University of Reading

On the design of smartphone based plant ID guides

Thesis submitted by
Andrew Bewsey
For the degree Doctor of Philosophy

Centre for Plant Diversity and Systematics,

School of Biological Sciences, University of Reading,

December 2018

Declaration

I confirm that this is my own work and the use of all material from other sources has been properly and fully acknowledged.

Andrew Bewsey

Acknowledgements

There are a number of people and organisations that, without their help, this this project would simply not have worked.

I thank the supervisors: A. Culham, P. Parslow, A. Black, and J. Mitchley. Without their help and time, and the voluntary help of their students, taking this project from a single idea to a fully-fledged and tested product would not have been possible. Thanks are also given to M. Christodoulou for her help correcting the thesis.

Without a loan of herbarium specimens from The Royal Botanic Garden Edinburgh Herbarium (**E**), the various rounds of testing shown here would not have been nearly as successful or thorough.

Thanks are given to my parents, friends, and anyone who was willing to listen while I talked through ideas for all parts of this PhD. The willingness of everyone to put up with the endless wittering has been constantly surprising.

Common acronyms used in this thesis

There are a number of different acronyms used in this thesis or in references. They are expanded here.

Acronym	**Expansion**
ANSI	Windows-1252
ASCII	American Standard Code for Information Interchange
BBC	The British Broadcasting Corporation
CPU	Central Processing Unit
CRLF	Carriage Return, Line Feed.
CSV	Comma Separated Value
HTML	HyperText Markup Language
GBP	Pound sterling
HSL	Hue Saturation Lightness
ID	IDentification
IE	Information Extraction
MSc	Master of Sciences
NLP	Natural Language Processing
OCR	Optical Character Recognition
PC	Personal Computer
PDF	Portable Document Format
PPPA	Per Person Per Annum
RAD	Rapid Action Development
RAM	Random-Access Memory
REGEX	REGular EXpression
RHS	Royal Horticultural Society
RRP	Recommended Retail Price
SDLC	Software Development LifeCycle
SQL	Structured Query Language
USD	United States Dollar
UTF	Unicode Transformation Format
XML	eXtensiable Markup Language

Abstract

The work presented here is the exploration and discussion of multi-access smartphone-based plant ID guides. The aim of this work was to explore the possibilities of using detailed multi-access keys on smartphones to identify taxa in the field to a professional standard. This involved establishing the most effective method of gathering data, deciding on the best design, evaluating whether users prefer this design over traditional media, and establishing whether or not continuation of this project would be financially sound. As they represented the best source of information, the decision was taken to extract data from extant paper-based ID guides. Two methods were attempted, automatic extraction through pattern matching techniques, and manual extraction. The automatic extraction technique was tested as paper-based ID guides were thought to use a very regular language. It was found, however, that the language used was not regular enough, and manual input of data was required. The design of the app was built from the designs of existing apps, and the principles of 'simple' and 'consistent', which were found to be the main themes of many design guides. When the design was evaulated, it was well received, only requiring refinements. After several rounds of updates, the app reached about 30% accuracy, similar to traditional paper-based ID guides. The similarity in accuracy to paper-based ID guides is indicative of the difficulty in identification of *Equisetum* specimens, thus is a positive indication of the quality of the app. The app was preferred by users, who found it easier to use. When the costs of continuation of this project were examined, it was quickly found that, despite a potential market of several thousand, the production costs were simply too high. Continuation was therefore deemed to require financial assistance, via grants or otherwise. This leads to a final suggestion that species x character databases are published with all novel taxonomic works in a standardised format, and that this is a requirement of grant funded work.

Contents

1 Introduction ... 1

1.1 Current floras and ID guides: general characteristics and coverages ... 2

1.2 Different key designs ... 9

1.2.1 Single access ... 9

1.2.2 Multi-access ... 15

1.2.3 Photorecognition ... 21

1.2.4 Picture matching ... 23

1.3 Smartphone ownership ... 24

1.4 What is missing from the current work? ... 24

1.5 Thesis overview ... 25

1.6 Aims and hypotheses ... 26

1.6.1 Hypotheses ... 26

1.6.1.1 With the rapid increase in power, natural language processing is no longer required to extract the data from floras ... 26

1.6.1.2 Everyday Smartphone technology is now sufficient to operate a complete technical key to the entire British Flora ... 26

1.6.2 App Design approach ... 26

1.6.2.1 If there is a best layout for an electronic flora designed to be used on smartphones, what is it? ... 26

1.6.3 Aims ... 26

1.6.3.1 Production of a useful database that can be used for species identification ... 26

1.6.3.2 Production of a high-quality smartphone app that can make use of this database ... 27

1.6.3.3 Investigation into whether production of the app would make a profit or not ... 27

2 Databases (part one): database size vs speed ... 28

2.1 Introduction ... 28

2.1.1 ANSI (Windows text encoding) ... 31

2.1.2 Search Design ... 31

2.2 Hypothesis tested and goal of these tests ... 31

2.2.1 Hypothesis ... 31

2.2.2 Aim ... 32

2.3 Materials and Methods ... 33

2.3.1 Test apps ... 33
2.3.2 Data analysis ... 35
2.3.2.1 Comparison of long vs wide database loading times ... 35
2.4 Results ... 36
2.4.1 Database handling method ... 36
2.4.2 Long vs wide databases ... 40
2.5 Discussion ... 41
2.6 Conclusion ... 44
3 Databases (part two): Natural Language Processing, Information Extraction, and their use in botanical database construction ... 45
3.1 Glossary ... 45
3.2 Introduction ... 45
3.3 Aim ... 49
3.4 Method ... 49
3.5 Results ... 52
3.6 Discussion ... 53
3.7 Conclusion ... 57
4 Databases (part three): Manual datasheet creation ... 58
4.1 Introduction ... 58
4.1.1 Different sources of morphological data ... 58
4.1.1.1 Paper based sources ... 58
4.1.1.2 Specimens ... 59
4.1.2 On databases and their content ... 59
4.1.2.1 Limiting the number of taxa ... 59
4.1.2.2 Data inclusion in this project. ... 60
4.1.2.3 Error induced in database creation ... 62
4.1.2.4 Database design ... 62
4.2 Aim ... 62
4.3 Method ... 62
4.3.1 Manual database ... 62
4.3.1.1 Materials ... 62
4.3.1.2 Procedure ... 63
4.3.2 Conversion to app-style database ... 64
4.3.3 Data testing ... 65

4.4 Results ... 65
4.4.1 Data inference ... 66
4.4.2 Data checking ... 67
4.5 Discussion ... 69
4.5.1 Data extraction ... 70
4.5.2 App database ... 75
4.6 Conclusion ... 76
5 App implementation (part one): design study of previous apps and best approaches to app design ... 77
5.1 Introduction ... 77
5.1.1 Prior work on multi-access keys ... 80
5.1.2 Previous electronic guides ... 86
5.1.2.1 List of questions ... 86
5.1.2.2 Figures ... 89
5.1.2.3 Search button ... 90
5.1.2.4 ActionBar contents ... 91
5.1.2.5 App navigation ... 91
5.1.3 Users ... 98
5.2 Method ... 98
5.2.1 Persona ... 98
5.2.2 Use case ... 99
5.2.3 Design ... 99
5.3 Results ... 103
5.4 Discussion ... 108
6 App implementation (part two): code design ... 110
6.1 Introduction ... 110
6.2 Aim ... 111
6.3 Materials methods, and results ... 112
6.3.1 Materials used ... 112
6.3.2 Practices used ... 112
6.3.3 Logic code ... 113
6.3.3.1 App use tutorial ... 117
6.3.4 Layout code ... 122
6.4 Results ... 126

6.5 Discussion ... 132
7 App implementation (part three): testing, evaluation, and incorporation of user feedback. ... 137
7.1 Introduction ... 137
7.1.1 Different approaches to product testing and evaluation ... 138
7.1.1.1 Usability test ... 138
7.1.1.2 Beta testing ... 139
7.1.1.3 Surveys ... 140
7.1.1.4 AB testing ... 140
7.1.1.5 Focus group testing ... 141
7.1.2 Electronic methods ... 142
7.1.3 Mixed methods ... 142
7.1.4 Software development lifecycles ... 143
7.1.4.1 Structured development SDLCs ... 145
7.1.4.2 Rapid action development (RAD) SDLCs ... 147
7.1.4.3 Agile ... 148
7.2 Aims ... 151
7.3 Method ... 151
7.3.1 Ongoing testing ... 151
7.3.2 SDLC used ... 151
7.3.3 General method for sessions ... 152
7.3.4 Session one – design students ... 153
7.3.5 Session two – herbarium volunteers ... 153
7.3.6 Session three – herbarium volunteers ... 156
7.3.7 Session four – experienced examiners ... 156
7.4 Results ... 156
7.4.1 General ... 156
7.4.2 Individual sessions ... 163
7.4.2.1 Session one – design students ... 163
7.4.2.2 Session two ... 166
7.4.2.3 Session three ... 168
7.4.2.4 Session four ... 169
7.4.3 SDLC (Software development lifecycle) ... 170
7.5 Discussion ... 171

7.6 Conclusions 178
8 Business study 180
8.1 Introduction 180
8.2 Aims 185
8.3 Method 185
8.3.1 Data from existing guides 185
8.3.2 Potential market size 186
8.3.3 Potential available grants 187
8.3.4 Cost of production 187
8.4 Results 187
8.4.1 Method of income for successful apps 187
8.4.2 Costs and sales of paper-based guides 190
8.4.3 Costs, sales, and/or downloads of electronic guides 193
8.4.4 Funding methods of previous works 197
8.4.5 Potential cost of running as a business 197
8.4.6 Potential market size 198
8.4.7 Potential available grants 198
8.5 Discussion 199
8.5.1 In app purchases 202
8.5.2 Funding production 203
8.5.3 Further development 205
8.6 Conclusions 206
9 Discussion and conclusions 208
9.1 Thesis contributions 208
9.2 General discussion 208
9.3 Limitations 213
9.4 Future work 214
9.5 Conclusions 215
10 Reference list 216
11 Appendices 243
Appendix A The App and full code 243
Appendix B Test photos and correct identifications reached by various photo-recognition apps 243
Appendix C Optics Maths 243

Appendix D Code 243
D.1 Autoextraction program 243
Appendix E Databases 243
E.1 Full database 243
E.2 Pines only database 243
E.3 Corrected *Equisetum* database 243
Appendix F A (very) brief introduction to programming 244
F.1 camelCase and underscore_case 244
F.2 Types of data 244
F.3 Java 244
F.4 XML 247
F.5 SQLite 250
F.6 The three languages and Android 251
Appendix G Additional information for chapter 6, App testing 254
G.1 Session one 254
G.2 Session two, Herbarium volunteers session one 256
G.2.1 Feedback generated through discussions with the volunteers during the session 256
G.2.2 Answers provided to the questionnaires 257
G.2.2.1 Results of question two (Identification of specimens) 257
G.2.2.2 Results of question three (Was the key easy to use?) 257
G.2.2.3 Results of question four (What aspect could be most improved when using this ID key under these circumstances?) 257
G.2.2.4 Results of question five (What aspect of this ID key did you think was the most useful?) 258
G.2.2.5 Results of question six (In the field, would you use this method?) 258
G.2.2.6 Results of question seven (If you feel that there are any other aspects of the ID key tested that need to be mentioned, please include them here) 258
G.3 Session three, Herbarium volunteers session two 259
G.3.1 List of specimens made available to the volunteers 259
G.3.2 Modified questionnaire used in session 260
G.3.3 Answers provided to the questionnaires 263
G.3.3.1 Results of question two (Identification of specimens) 263

G.3.3.2 Results of question three (Was the key easy to use?) 268
G.3.3.3 Results of question four 268
G.3.3.4 Results of question five 269
G.3.3.5 Results of question six 270
G.3.3.6 Results of question seven 270
G.4 Session four 270
G.4.1 Feedback generated during the session 270
Appendix H Additional data for Chapter 7, Business case 270
H.1 Full data for free apps 270
H.2 Full data for non-free apps 270
Appendix I Gantt Chart of timescale/milestones 271

1 Introduction

Modern smartphones are very powerful, highly versatile, extremely portable computers. They are provided with a wide range of equipment, including multiple cameras (GSMArena, 2018a), iris scanners (GSMArena, 2018c), and squeezable sections (Brown, 2018). There are a wide variety of exciting apps available to the user. Google maps is available on Android, providing the user with: highly detailed maps allowing world-wide navigation, details on traffic conditions, and in-depth information about locations (Google LLC, 2018b). Shazam (Apple INC, 2018) and Google Sound search (Lyon, 2018) allow the user to ask their device to listen to and automatically identify a piece of music that is being played through an external source. There are a great number of social media apps, such as Twitter (Twitter Inc., 2018) and Facebook allowing people to stay in contact with others. Smartphones are part of daily life for many people and in many jobs. Development of smartphones is being driven by user demands, for more powerful and capable gaming phones (Murray, 2018), or simply for taller screens to display more text at once (Dauer, 2018). Smartphone apps are easily and frequently updated, and therefore remain current. The most well used tools in botanical identification, however, remain a book and a hand-lens. If the book should become out of date, through taxonomic revision for example, then the user must buy the new version.

If your app, or book, is popular, then large sums of money can be made. J.K.Rowling, author of the Harry Potter series, is on the Sunday Times Rich list (The Sunday Times, 2018). Top apps can earn over 800Million USD (Wagner, 2018). A computer science degree is one of the highest earning professions, at the same level as economics, and law (Kirk and Scott, 2018). Publishing apps on the Google Play store has only a 25USD fee (Google, 2018e). Yet publishing academic work is famously expensive (Noorden, 2013), with the academic only being paid a small percentage of the price of textbooks (Kurtzleben, 2012).

Not long after the advent of the electronic computer in 1946 (Burks, 1947), their use in plant biosciences was considered (Hall, 1954), with the first computer based identification keys (ID keys) being designed a few years later (Hall, 1970) . Improvements were made over time, with systems such as DELTA (Dallwitz, 1980) and Lucid (LucidCentral, 2015) becoming widely used favourites. With the advent of smartphones, attempts have been made to produce national floras as apps, but due to the limited power of smartphones available at the time, these were limited in usability (Morrison, 2011).

This project originated as an MSc dissertation (Bewsey, 2014), where a similar key to the flora of Whiteknights campus, University of Reading was created. At the time the MSc project originated, the range of apps available for plant ID on Android were of limited quality. The aim of the MSc was the production of a simple to use guide capable of plant specimen identification to family level. The success of the MSc work lead to its continuation as a PhD project shown here.

1.1 Current floras and ID guides: general characteristics and coverages

There is a wide variety of different ID (IDentification) media including: floras, field guides, electronic guides, photo recognition guides, and others. They cover different regions at different levels of specificity. They have been produced for many years and for different user skill levels. The number of taxa included in a guide can vary greatly. They have been presented in different forms of media.

Species Plantarum (Linnaeus, 1753) can be argued to be the first 'modern' ID guide. It is the first book where the binomial nomenclature system is used. Species are arranged by numbers of floral parts; if the user examines their specimen they are capable of turning to the right section of the book, from there identification is a matter of choosing the right description. Before this, herbals tended to have a haphazard arrangement, where identification was simply a matter of finding an appropriate picture matching your specimen. The first true single-access key was made by Lamark in 1788 (Walter and Winterton, 2007), but between then and 1962 there was little research was done on their design (Osborne, 1963). Several methods generating single-access keys via computer programs were investigated in the late 1960's to early 1970's (Dallwitz, 1974). Other forms of ID guide, such as multi-access keys have existed since at least 1927 (Swain, 1926; Stucky, 1984). Development of multi-access keys has led to different programs being available for the development and use of electronic multi-access keys (Edwards and Morse, 1995; LucidCentral, 2015).

As ID guides have existed for a considerable amount of time, it is not surprising that several guides have been written for many areas of the world (Frodin, 2001). The British Isles have been particularly thoroughly covered (A. Culham, pers. com), with various individual counties having multiple floras (Druce, 1897; Bowen, 1968; Crawley, 2005), let alone the existence of multiple simultaneous national floras by different authors (some of which have had multiple editions (Stace, 1991, 1997, 1999, 2010c; Sell and Murrell, 1997, 2006, 2009, 2014, 2018).

When this project began, there were relatively few apps that could be considered plant ID keys (of any of the forms described later in this introduction). During this project, more have been released. To choose which Android-based ID keys to review, a casual investigation of different possible search terms in the Google Play store was undertaken, with the goal of finding the term that produced the greatest number of keys. On 19/06/2019, this appeared to be 'Plant ID key'. The store page of each app was investigated to find apps that were genuinely ID keys. Of the 249 results this search returned, 147 were clearly ID keys of one of the types described later in the introduction. From the remaining apps, 60 were not related to with plants at all, 22 were related to plants but were not ID guides, with the others being related to plants but where it was not clear whether or not they included an ID key. Only apps that clearly indicated they included ID guides were considered for review. The top result (i.e. highest in the results list) of each type of ID key detailed earlier was chosen, as were 7 randomly chosen (using the RND function of a Casio fx-83GT PLUS). This list was then supplemented with keys that have been found to be interesting and/or keys that have been recommended for investigation. These supplemental keys were not restricted to the Google Play store, but were restricted to electronic media. This quick search indicated that the most popular type of ID key was multi-access keys, with photo recognition as the second most popular.

Examples of the full variety of available ID guides, including those chosen for review from the Android store, are shown in Table 1-1.

While there is a considerable variety of different guides available, and information throughout this thesis has been drawn from as many relevant guides as possible, there are several 'standout' guides. These 'standout' guides will also be mentioned frequently throughout the thesis, either as directly, or as exemplars of the relevant variety of available guides.

Table 1-1 - Details of various existing ID guides. [1]RRP rather than actual store price wherever possible. If store price has been used, date and location where the price was gathered from were included. [2] Each geographical region in the key has its own set of photos and species. The number of species shown here was established by combining the numbers covered in each region; it is presumed there is some overlap, and the total number of species is slightly lower

Guide	Number of taxa	Geographical coverage	Depth of coverage	Media	Type(s) of key	Price[1]	Aims of coverage
Stace (2010c)	>4,800	Britain and Ireland	Sub-species and aggregate	Paper	Single- and multi-access	£52.57 (correct on 2/12/2018) (Stace, 2010c)	All native and naturalised, including frequent casuals
Poland and Clement (2009)	3,000	British Isles	Sub-species and aggregate	Paper	Single-access	£20.99 (correct on 13/12/2018) (Poland and Clement, 2018a)	Identification of most UK species without relying on floral characters
Key: PlantFamilies (CRinUS, 2012)	274	China	Flora of China plant families	Android and Web	Multi-access	Free	Unclear
Tutin *et al.* (1964a, 1968, 1972, 1976, 1980; 1993)	11,557 (Govaerts, 2001)	Europe	Sub-species and aggregate	Paper	Single-access	Over £100 for all volumes	All species present throughout Europe
Streeter *et al.*, (2009)	>1,900	British Isles	Species	Paper	Single-access	£19.99	All 'well established' species (no distinction within

							species aggregates).
Rose and O'Reilly (Rose and O'Reilly, 2006)	>1,600	British Isles	Species	Paper	Single access	£15.79 (correct on 13/12/2018) (Rose and O'Reilly, 2018)	All native and well established species (Grasses, rushes, sedges, ferns and fern allies, and 'lower plants') are excluded.
Crawley (2005)	2071	Berkshire (Vice county area)	Subspecies and hybrids	Paper	Single-access	£50	Full and detailed coverage of the vicecounty of Berkshire
ArbolApp (Real Jardin Botanico, 2015)	143	Iberian peninsula and Balearic Islands	Species	Android	Single- and multi-access	Free	Native and frequently established trees
Page (1982)	106	British Isles	Species	Paper	Single- and multi-access	£15.00	Full coverage of all ferns of the British Isles
Identifying Commonly Cultivated Palms (idtools.org, 2018)/Palm ID key (LucidMobile, 2018b)	92 (Web) /82 (App)	Continental USA, Hawaii, and Caribbean Islands.	Species	Android and Web	Multi-access	Free	Commonly cultivated palm species

Pla@ntNet(2017)	41,416[2]	Various worldwide geographical regions	Species	Android and iOS	Photo-recognition	Free	Photorecognition of various species found globally
MAKAQueS (2017)	3221	British Isles	Species	Android and PC	Single-access multi-palthway	£9.99	Coverage to species level of the British Flora
Tree ID - British Trees (Woodland Trust, 2019)	76	British Isles	Species	Android	Single-access	Free	Common native and non-native tree species
Forest Tree Identification (Discovery Green Lab, 2019)	290	Unclear	Species	Android	Single-access	Free	Unclear
Plant Identification (Learn and Grow Inc, 2018)	Unclear	Unclear (Presumed USA)	Species	Android and Web	Multi-access	Free	Unclear
Aquarium & Pond Plant ID (LucidMobile, 2018a)	270	Cultivated pond and aquarium plants	Genus	Android and Web	Multi-access	Free	All trade freshwater taxa in 2017
NZ Trees (AUT Ventures Limited, 2018)	>50	New Zealand	Species	Android	Multi-access	Free	Tree species native to NZ
iFlora (Trackenberg, 2019)	Unclear	The European Union	Species	Android, web, and book	Multi-access	Free to £78	Coverage of selected parts of the EU flora

The conifers of Britain guide (Parratt, 2017)	50	British Isles	Species	Web	Multi-access	Free	All naturalised conifers in Britian
Flora Incognita (Technische Universität Ilmenau, 2019)	>4800	Europe	Species	Android and iOS	Photo-recognition	Free	Wild flora of Europe
PlantFinder – Flower & Plant Identification (Sleep Sounds Studio, 2019)	90% of known plant and tree species	Worldwide	Species	Android	Photo-recognition	Free	The world
plant id (Divers Dias, 2019)	20000	Unclear	Species	Android	Photo-recognition	Free	Unclear

1.2 Different key designs

1.2.1 Single access

Single access keys are 'guided' keys; the user answers a chain of questions, in order, to reach an identification. Single-access is often incorrectly treated as a synonymous term to dichotomous keys (Penev *et al.*, 2009). This is not necessarily correct. 'Dichotomous' refers to a choice between only two, and many single access keys include steps where there are many choices (Penev *et al.*, 2009). For the purposes of this thesis, the term 'single-access' is used to refer to all keys of this sort, unless other terms are directly used in the reference material and maintaining their term assists with context. There are different designs of single-access key.

Drinkwater (2009) considers bracketed and indented (nested) keys as the only two choices. Bracketed keys are where all parts of the couplet are located together, with indication of either which couplet to proceed to, or the identification reached. Bracketed keys are used by (among others) Stace (2010c), Streeter (2009), Watts (1972), and Rose (Rose and O'Reilly, 2006). An example of a bracketed key is shown in Figure 1-1. Indented keys take the other approach, instead following each option in each couplet to its conclusion at an identification. Indented keys are used in (among others) Tutin *et al.* (1964a, 1968, 1972, 1976, 1980; 1993), Poland and Clement (2009). An example indented key is shown in Figure 1-2. With electronic single-access keys it can be harder to define whether a key is indented or bracketed, and this distinction can sometimes become meaningless. Arbolapp (Real Jardin Botanico, 2015) includes a single-access key where the user is shown only one couplet at any one time, thus it follows neither design. An example of an electronic single-access key that is neither indented nor bracketed is shown in Figure 1-3. The alternative to showing the user one question at a time for electronic keys would be to show them previous and/or subsequent questions but set them un-answerable (similar to how these questions are visible in paper-based keys), but no keys were found to use this method.

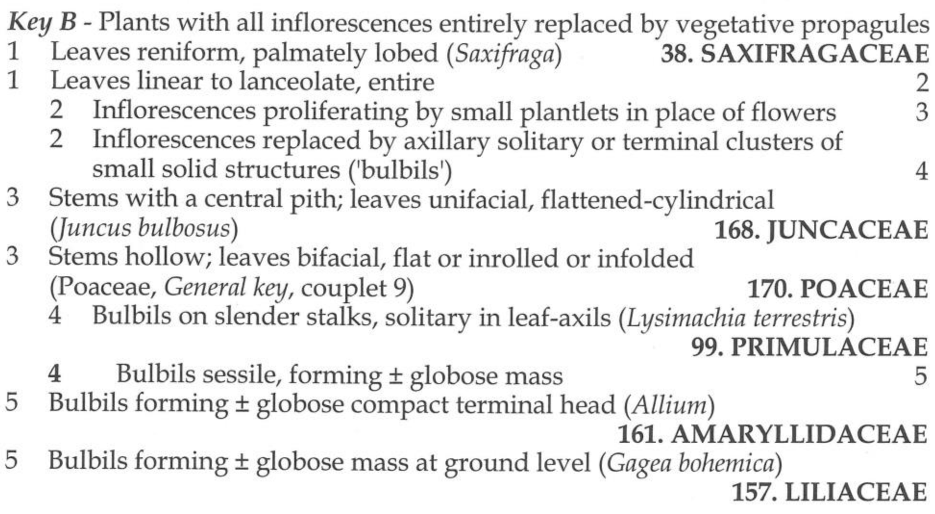
If a key has many end-points, then it can reach a considerable length (for taxonomic ID keys, end points are typically species, but keys may also only identify to family level). Assuming all 4,800 taxa in Stace (2010c) are species, and assuming a perfect binary tree, between 8191 and 16383 couplets would be required (calculation based on Harder

(2011), there is no perfect binary tree with 4,800 nodes, hence the range of couplets required). Some authors solve this problem by splitting keys into sub-sections, ensuring each sub-key is shorter. Several authors (Stace, 2010b; Streeter *et al.*, 2009; Tutin *et al.*, 1964a; for examples) provide two levels of key; a general key to families, and within each family a key to species. Poland and Clement (2009) take a different approach to splitting their key, instead splitting based on morphological similarity. The initial key to families is sometimes further split to reduce its length. Stace's key to Angiosperm families (Stace, 2010a) keys out 146 families. It split into several sub-keys, with the longest reaching 75 couplets. Tutin *et al.* (1964b) key out 172 Angiosperm families, but the key is not split into sub-keys, and reaches 302 couplets.

Couplet design and within couplet language can also vary, although these differences can frequently be seen within a single key. Couplets typically are of either the 'A/Not A' (see couplet 108 of Figure 1-2) or 'sub-set 1/sub-set 2' style (see couplet 2 of Figure 1-1). Individual options within couplets generally have multiple characters (see couplet 107 of Figure 1-2) (Drinkwater, 2009), but this does not always occur (see couplet 116 of Figure 1-2).

There are few single-access keys available as apps. Of the 147 keys found, only 12 were clearly single-access, with eight of these being dichotomous. From this set, Tree ID - British Trees (Woodland Trust, 2019) and Forest Tree Identification (Discovery Green Lab, 2019) were chosen for review. These apps cover similar numbers of species, with British Trees at 76 and Forest Tree Identification at 290 (note that while these numbers seem different, other apps claim coverage of thousands of species). Both these keys are single-access, but not dichotomous. There are a number of similarities and differences between the two apps. The first noticeable similarity is the approach taken to question design. They are both polychotomous keys, thus have multiple choices for each question. Tree ID - British Trees (Woodland Trust, 2019) is slightly different, the first question follows the multi-access style, allowing you to choose any number and combination of options, but this first question is not diagnostic, instead asking about what features of the specimen are available, and therefore is arguably not a part of the 'true' key. They both follow the usual approach of electronic single-access keys of only showing one question to the user at a time. Both keys allow you to see the current list of possible identifications at any stage, and both show the list of results as a scrolling list with photographs of the possibilities. From here, however, there are a number of differences. The most important of these is that Forest Tree Identification (Discovery Green Lab, 2019) lacks crucial information about geographical coverage of the flora either in the app or on the app store page (Discovery Green Lab, 2019). This information is vital for ID guides; you need to know that the taxa in the guide match up to the taxa in your geographical location. Tree ID - British Trees (Woodland Trust, 2019), on the other hand, has a geographic description of coverage both in the app and on the store page (Woodland Trust, 2019). Both apps provide click-through information about possible results; Tree ID - British Trees (Woodland Trust, 2019) providing off-line information, while Forest Tree Identification (Discovery Green Lab, 2019) links through to Wikipedia (Wikipedia, 2015). Linking through to Wikipedia has the advantage that the further information provided will always be current and updated without having to maintain the app, and that storage required for the app can remain small; however, it means an internet connection is required for this further information. Ensuring the app data is up-to-date can be incorporated as part general app updates. The final significant difference is how diagnostic information is displayed to the user. Both apps use figures for each option; Tree ID - British Trees (Woodland Trust, 2019) using diagrams only, while Forest Tree Identification (Discovery Green Lab, 2019) uses both diagrams and photographs.

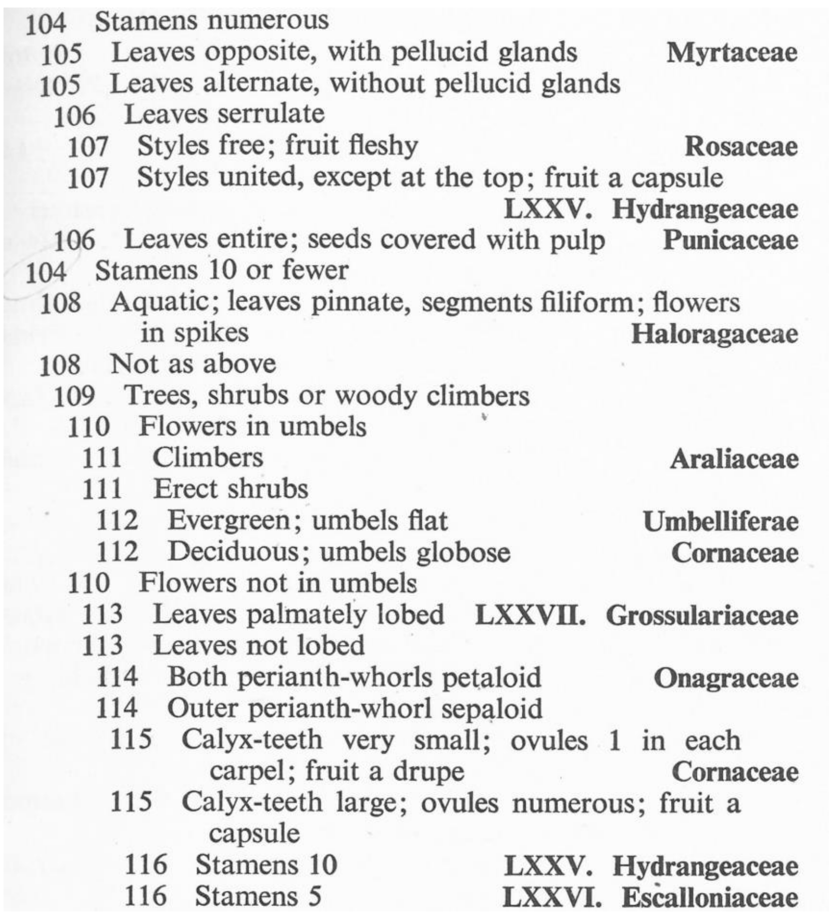
It has often been acknowledged that single-access keys are imperfect. Questions can be rendered un-answerable when specimens do not display the feature requested in the key, or when the user is not able to answer the question (e.g. an question relating to scent is unanswerable for a person with anosmia) (Taylor, 1995). Atypical specimens may result in a couplet being answered incorrectly, leading the user down the wrong path in the key (Taylor, 1995; Drinkwater, 2009). They also risk being slow to use; the user is forced to answer a chain of questions before an identification is produced.



Key B - Plants with all inflorescences entirely replaced by vegetative propagules

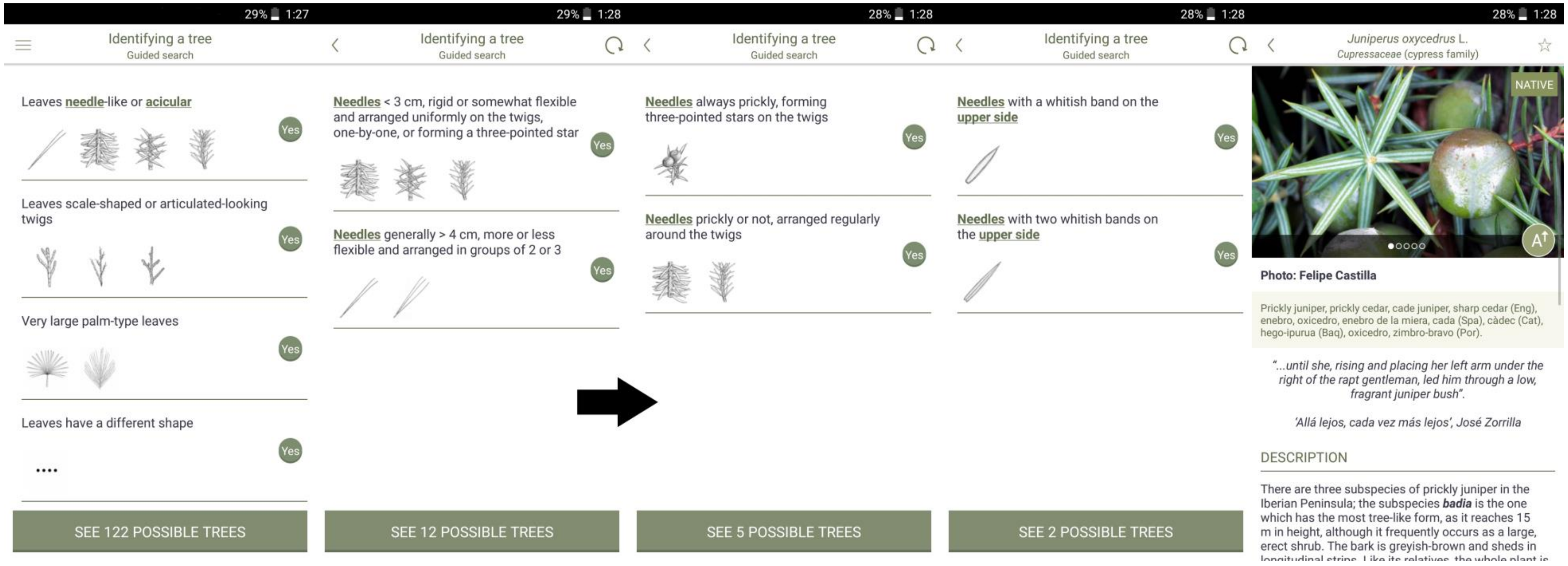
1 Leaves reniform, palmately lobed (*Saxifraga*) **38. SAXIFRAGACEAE**
1 Leaves linear to lanceolate, entire 2
  2 Inflorescences proliferating by small plantlets in place of flowers 3
  2 Inflorescences replaced by axillary solitary or terminal clusters of small solid structures ('bulbils') 4
3 Stems with a central pith; leaves unifacial, flattened-cylindrical (*Juncus bulbosus*) **168. JUNCACEAE**
3 Stems hollow; leaves bifacial, flat or inrolled or infolded (Poaceae, *General key*, couplet 9) **170. POACEAE**
  4 Bulbils on slender stalks, solitary in leaf-axils (*Lysimachia terrestris*) **99. PRIMULACEAE**
  4 Bulbils sessile, forming ± globose mass 5
5 Bulbils forming ± globose compact terminal head (*Allium*) **161. AMARYLLIDACEAE**
5 Bulbils forming ± globose mass at ground level (*Gagea bohemica*) **157. LILIACEAE**

Figure 1-1 - Example of a bracketed key. This is sub-key B from the Keys to Families of Angiosperms in Stace (2010a).



104 Stamens numerous
 105 Leaves opposite, with pellucid glands **Myrtaceae**
 105 Leaves alternate, without pellucid glands
  106 Leaves serrulate
   107 Styles free; fruit fleshy **Rosaceae**
   107 Styles united, except at the top; fruit a capsule **LXXV. Hydrangeaceae**
  106 Leaves entire; seeds covered with pulp **Punicaceae**
104 Stamens 10 or fewer
 108 Aquatic; leaves pinnate, segments filiform; flowers in spikes **Haloragaceae**
 108 Not as above
  109 Trees, shrubs or woody climbers
   110 Flowers in umbels
    111 Climbers **Araliaceae**
    111 Erect shrubs
     112 Evergreen; umbels flat **Umbelliferae**
     112 Deciduous; umbels globose **Cornaceae**
   110 Flowers not in umbels
    113 Leaves palmately lobed **LXXVII. Grossulariaceae**
    113 Leaves not lobed
     114 Both perianth-whorls petaloid **Onagraceae**
     114 Outer perianth-whorl sepaloid
      115 Calyx-teeth very small; ovules 1 in each carpel; fruit a drupe **Cornaceae**
      115 Calyx-teeth large; ovules numerous; fruit a capsule
       116 Stamens 10 **LXXV. Hydrangeaceae**
       116 Stamens 5 **LXXVI. Escalloniaceae**

Figure 1-2 - Example of a nested key. This is a section of the Key to Angiospermae in Tutin et al. (1964b)


29% 1:27

Identifying a tree
Guided search

Leaves needle-like or acicular — Yes

Leaves scale-shaped or articulated-looking twigs — Yes

Very large palm-type leaves — Yes

Leaves have a different shape — Yes

....

SEE 122 POSSIBLE TREES

29% 1:28

Identifying a tree
Guided search

Needles < 3 cm, rigid or somewhat flexible and arranged uniformly on the twigs, one-by-one, or forming a three-pointed star — Yes

Needles generally > 4 cm, more or less flexible and arranged in groups of 2 or 3 — Yes

SEE 12 POSSIBLE TREES

28% 1:28

Identifying a tree
Guided search

Needles always prickly, forming three-pointed stars on the twigs — Yes

Needles prickly or not, arranged regularly around the twigs — Yes

SEE 5 POSSIBLE TREES

28% 1:28

Identifying a tree
Guided search

Needles with a whitish band on the upper side — Yes

Needles with two whitish bands on the upper side — Yes

SEE 2 POSSIBLE TREES

28% 1:28

Juniperus oxycedrus L.
Cupressaceae (cypress family)

NATIVE

Photo: Felipe Castilla

Prickly juniper, prickly cedar, cade juniper, sharp cedar (Eng), enebro, oxicedro, enebro de la miera, cada (Spa), càdec (Cat), hego-ipurua (Baq), oxicedro, zimbro-bravo (Por).

"...until she, rising and placing her left arm under the right of the rapt gentleman, led him through a low, fragrant juniper bush".

'Allá lejos, cada vez más lejos', José Zorrilla

DESCRIPTION

There are three subspecies of prickly juniper in the Iberian Peninsula; the subspecies ***badia*** is the one which has the most tree-like form, as it reaches 15 m in height, although it frequently occurs as a large, erect shrub. The bark is greyish-brown and sheds in longitudinal strips. Like its relatives, the whole plant is

Figure 1-3 - Example of an electronic single-access key from ArbolApp (Real Jardin Botanico, 2015).

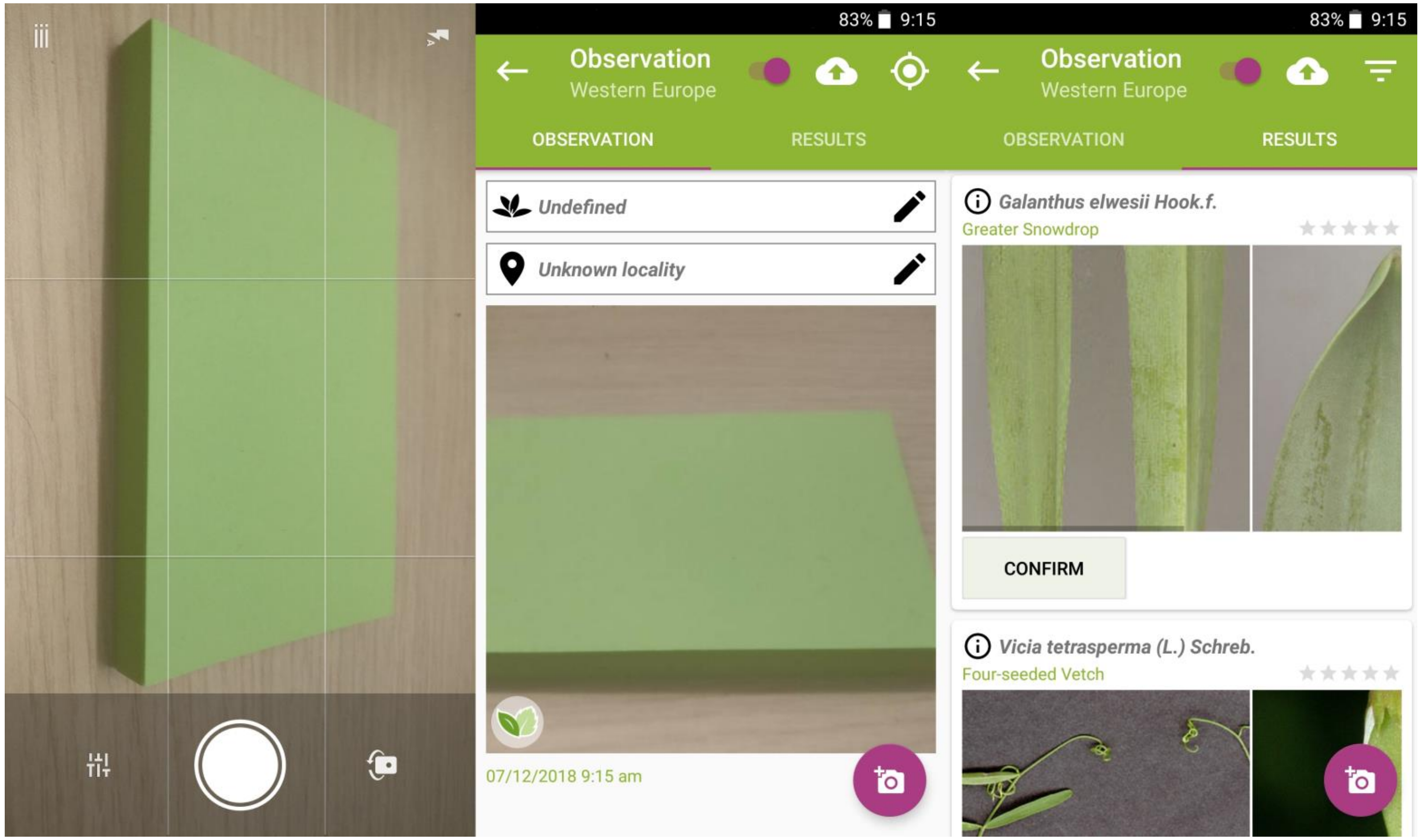


Figure 1-4 - Example of the general 'flow' of a photo-recognition app. The app used in this example is Plant.NET (Plantnet-project.org, 2015). *In this example a photograph of a stack of green notepaper is matched with a range of different green leaves on a similar colour background. All matches are noted as being low quality.*

1.2.2 Multi-access

Multi-access keys are 'open' keys; the user is presented with a list of questions, the answers of which will narrow down a list of potential identifications until (ideally) a single possibility is left. They are considered much more flexible for the user (Drinkwater, 2009).

Multi-access keys may also be called polyclaves or on-line identification programs (Edwards and Morse, 1995). Of these two options, 'on-line identification program' is incorrect through a lack of specificity. There is no requirement for an electronic guide to be multi-access (Drinkwater, 2009); electronic single-access and photo-recognition based guides exist. Multi-access keys do not require an electronic medium, but electronic guides are more commonly multi-access (Drinkwater, 2009).

There are frequently recognised advantages of multi-access keys over single access:

- The clearest features can be input into the key first (Tardivel and Morse, 1996; Drinkwater, 2009)
- Questions on unclear or missing features can be avoided entirely (Stucky, 1984; Edwards and Morse, 1995; Tardivel and Morse, 1996; Drinkwater, 2009; Penev *et al.*, 2009)

Design of a multi-access keys depends on the media they are presented on. Paper-based multi-access keys take the form shown in Figure 1-5. Here the user works through the list of statements, choosing the most appropriate from each set (or ignoring sets if no confident decision can be made). This provides them with a combination of letters, which they then match to the set of possible identifications below.

Figure 1-6 and Figure 1-7 show different electronic based multi-access keys, but both work in a similar manner. Data for electronic multi-access keys are typically stored as a species x character database (Edwards and Morse, 1995; Tardivel and Morse, 1996; Penev *et al.*, 2009; Dallwitz *et al.*, 2013), and generally work by using this data and the user input to eliminate non-matching taxa (Edwards and Morse, 1995; Drinkwater, 2009; Dallwitz *et al.*, 2013). The key shown in Figure 1-6 is PC based, and is designed to take advantage of the large amount of available screen space. A long list of potential characters to use in the identification process is displayed, along with the characters that have been used, and lists of both the possible taxa and the taxa that have been eliminated. The key in Figure 1-7 is smartphone based and has a different design to account for the smaller screen. Here only the list of questions is displayed, with the tabs along the bottom of the screen allowing the user to quickly view the potential taxa or features selected. An in-depth look at the design of smartphone based multi-access guides follows in chapter 5.

Multi-access keys do not solve all the problems of single-access keys. If the user answers a question incorrectly, an incorrect identification will still be made (Drinkwater, 2009). This mistake, or an aberrant feature on the specimen may reduce the list of potential taxa to zero (Edwards and Morse, 1995). Even if the user does answer the question correctly, it may have little effect on the list of potential taxa (Tardivel and Morse, 1996; Drinkwater, 2009). They also have the possibility of being less specific than a single-access key. If the user cannot discern enough characters, then the list of potential taxa may not be reduced to one (Edwards and Morse, 1995).

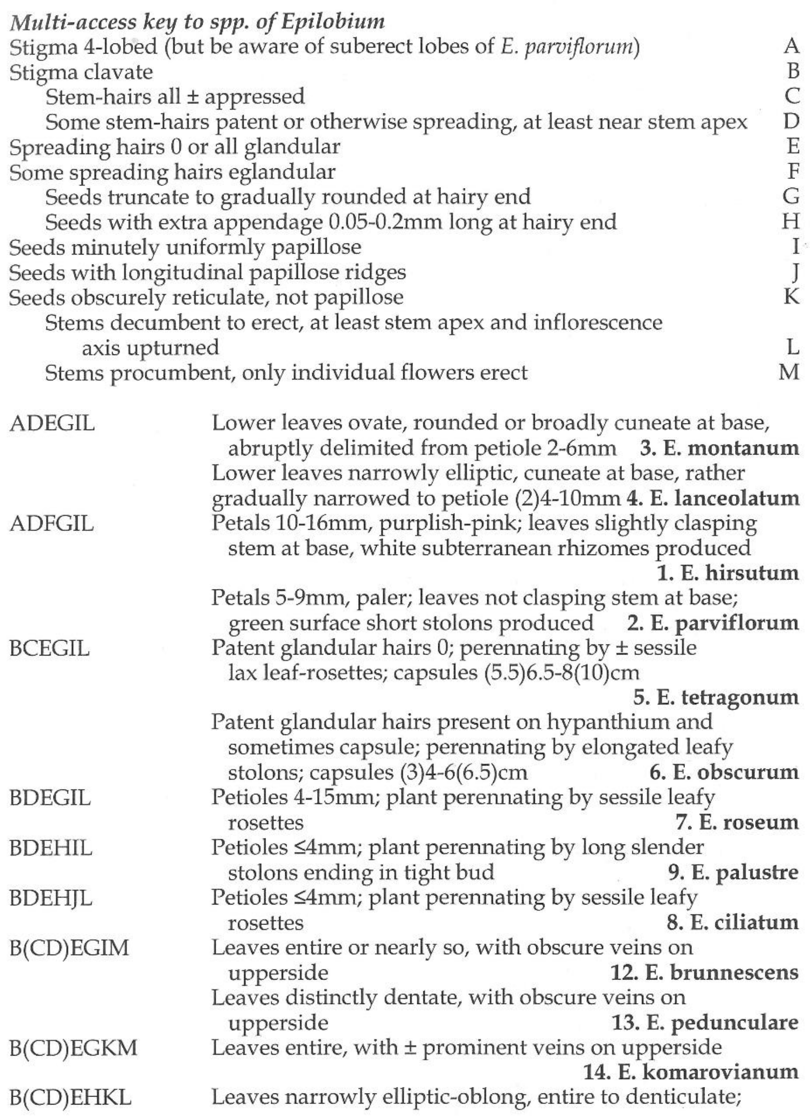
The majority of keys found in the Google Play store search were multi-access keys, at 98 of the 145 confirmed as Plant ID guides. Of the 98 multi-access apps, 76 were made by three groups: 48 by Wildflower Search, 20 as Lucid keys adapted for mobile phones, and eight by High Country Apps LLC. These 76 apps have reached a combined total of at least 48050 downloads. Wildflower Search apps cover separate states of the USA and parts of Canada with several thousand taxa included in each app, High Country Apps LLC covering some states and some wildparks in the USA and generally several hundred taxa, and Lucid keys covering a variety of different geographical ranges and number of taxa.

Four app-based guides were chosen for review here; Plant Identification (Learn and Grow Inc, 2018), Aquarium & Pond Plant ID (LucidMobile, 2018a), NZ Trees (AUT Ventures Limited, 2018), and iFlora (Trackenberg, 2019). The conifers of Britain guide (Parratt, 2017) is web-based and was included as it has an otherwise unknown design. As with all other keys, their coverage varies (but is generally lower than photo recognition keys), with NZ Trees (AUT Ventures Limited, 2018) covering over 50 tree species native to New Zealand, Aquarium & Pond Plant ID (LucidMobile, 2018a) covering 270 cultivated pond and aquarium plants, iFlora (Trackenberg, 2019) covering most of Europe in various modules. Plant Identification (Learn and Grow Inc, 2018) links through to two web based keys (see Portti and NatureGate (2019) and the U.S. Department of Agriculture (2019) for these keys) allowing it to cover a considerable number of taxa but meaning it is the only one of the smartphone based keys requiring an internet connection to work. The conifers of Britain guide (Parratt, 2017) covers the 50 naturalised or native conifers in Britain.

All of these keys follow, in very general terms, the list of questions design as shown in Figure 1-7, but have varying levels of functionality in the search page. All allow you to answer questions and produce identifications. NZ Trees (AUT Ventures Limited, 2018) does not provide additional help to questions, and only indicates the number of remaining results after pressing search. Aquarium & Pond Plant ID (LucidMobile, 2018a) improves on this by providing general information about the question (but not individual answers), and by indicating how many possible identifications have been removed by an answer and how many remain. iFlora (Trackenberg, 2019) disables answers that are no-longer diagnostic, shortening the list of options the user has to consider, and indicates the number of possible identifications remaining. The two sites linked to through Plant Identification (Learn and Grow Inc, 2018) provide very different experiences. Portti and NatureGate (2019) has few pictoral questions and attempts to provide a constantly updated list of possible identifications; while The U.S. Department of Agriculture (2019) keys provide very basic functionality, with a list of possible answers without separating questions, and no auto-updating list of results. The conifers of Britain guide (Parratt, 2017) does not eliminate identifications, instead scoring every taxa based on how many answered questions they match and how many they do not match. This is used to provide a ranking of all possible identifications, and an indication of which taxa match which feature. See Figure 1-8 for an example of this.

The results page differ between different keys, but with less variety than the search page. iFlora (Trackenberg, 2019), Aquarium & Pond Plant ID (LucidMobile, 2018a), and NZ Trees (AUT Ventures Limited, 2018) all providing brief descriptions and extra images on device. The conifers of Britain guide (Parratt, 2017) provides the same information as these keys, but on the website. Portti and NatureGate (2019) (linked through from Plant

Identification (Learn and Grow Inc, 2018)) provides a longer description and many more photos, while The U.S. Department of Agriculture (2019) provides only nomenclatural and distribution data.

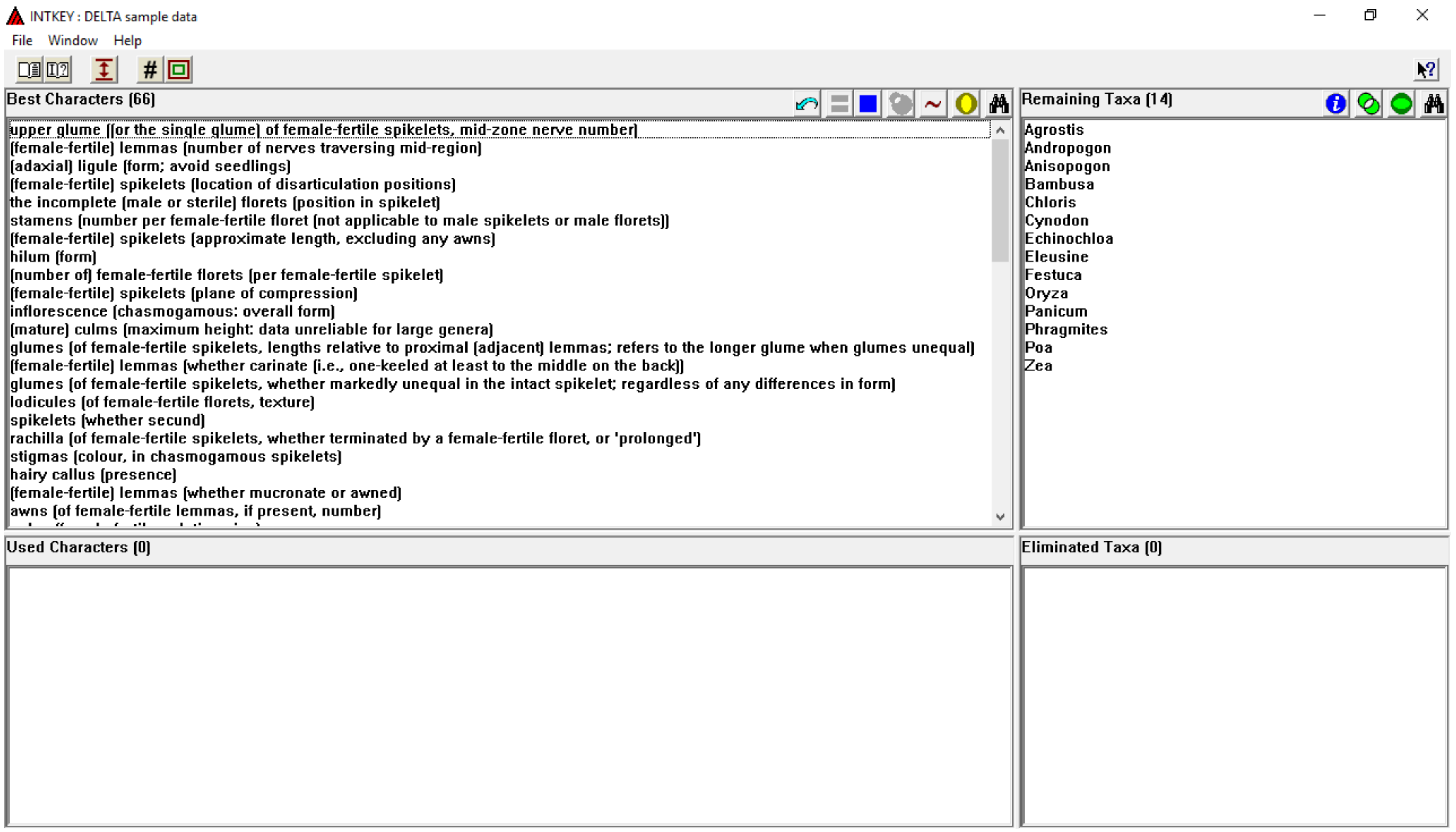


Multi-access key to spp. of Epilobium

Stigma 4-lobed (but be aware of suberect lobes of *E. parviflorum*)	A
Stigma clavate	B
Stem-hairs all ± appressed	C
Some stem-hairs patent or otherwise spreading, at least near stem apex	D
Spreading hairs 0 or all glandular	E
Some spreading hairs eglandular	F
Seeds truncate to gradually rounded at hairy end	G
Seeds with extra appendage 0.05-0.2mm long at hairy end	H
Seeds minutely uniformly papillose	I
Seeds with longitudinal papillose ridges	J
Seeds obscurely reticulate, not papillose	K
Stems decumbent to erect, at least stem apex and inflorescence axis upturned	L
Stems procumbent, only individual flowers erect	M

ADEGIL	Lower leaves ovate, rounded or broadly cuneate at base, abruptly delimited from petiole 2-6mm	**3. E. montanum**
	Lower leaves narrowly elliptic, cuneate at base, rather gradually narrowed to petiole (2)4-10mm	**4. E. lanceolatum**
ADFGIL	Petals 10-16mm, purplish-pink; leaves slightly clasping stem at base, white subterranean rhizomes produced	**1. E. hirsutum**
	Petals 5-9mm, paler; leaves not clasping stem at base; green surface short stolons produced	**2. E. parviflorum**
BCEGIL	Patent glandular hairs 0; perennating by ± sessile lax leaf-rosettes; capsules (5.5)6.5-8(10)cm	**5. E. tetragonum**
	Patent glandular hairs present on hypanthium and sometimes capsule; perennating by elongated leafy stolons; capsules (3)4-6(6.5)cm	**6. E. obscurum**
BDEGIL	Petioles 4-15mm; plant perennating by sessile leafy rosettes	**7. E. roseum**
BDEHIL	Petioles ≤4mm; plant perennating by long slender stolons ending in tight bud	**9. E. palustre**
BDEHJL	Petioles ≤4mm; plant perennating by sessile leafy rosettes	**8. E. ciliatum**
B(CD)EGIM	Leaves entire or nearly so, with obscure veins on upperside	**12. E. brunnescens**
	Leaves distinctly dentate, with obscure veins on upperside	**13. E. pedunculare**
B(CD)EGKM	Leaves entire, with ± prominent veins on upperside	**14. E. komarovianum**
B(CD)EHKL	Leaves narrowly elliptic-oblong, entire to denticulate;	

Figure 1-5 - Example of a paper based multi-access key. This example is the multi-access key to Epilobium in Stace (2010b).



Best Characters (66)

upper glume ([or the single glume] of female-fertile spikelets, mid-zone nerve number)
(female-fertile) lemmas (number of nerves traversing mid-region)
(adaxial) ligule (form; avoid seedlings)
(female-fertile) spikelets (location of disarticulation positions)
the incomplete (male or sterile) florets (position in spikelet)
stamens (number per female-fertile floret (not applicable to male spikelets or male florets))
(female-fertile) spikelets (approximate length, excluding any awns)
hilum (form)
(number of) female-fertile florets (per female-fertile spikelet)
(female-fertile) spikelets (plane of compression)
inflorescence (chasmogamous: overall form)
(mature) culms (maximum height: data unreliable for large genera)
glumes (of female-fertile spikelets, lengths relative to proximal (adjacent) lemmas; refers to the longer glume when glumes unequal)
(female-fertile) lemmas (whether carinate (i.e., one-keeled at least to the middle on the back))
glumes (of female-fertile spikelets, whether markedly unequal in the intact spikelet; regardless of any differences in form)
lodicules (of female-fertile florets, texture)
spikelets (whether secund)
rachilla (of female-fertile spikelets, whether terminated by a female-fertile floret, or 'prolonged')
stigmas (colour, in chasmogamous spikelets)
hairy callus (presence)
(female-fertile) lemmas (whether mucronate or awned)
awns (of female-fertile lemmas, if present, number)

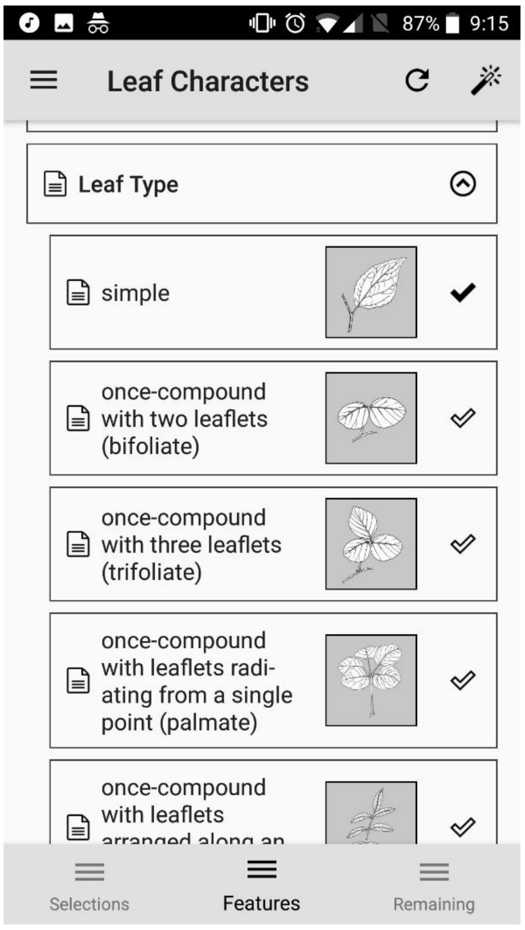
Remaining Taxa (14)

Agrostis
Andropogon
Anisopogon
Bambusa
Chloris
Cynodon
Echinochloa
Eleusine
Festuca
Oryza
Panicum
Phragmites
Poa
Zea

Used Characters (0)

Eliminated Taxa (0)

Figure 1-6 - Example of a computer based multi-access key. This example is a screenshot of the Delta/Intkey program (Dallwitz et al., 2000), *showing the '4 panel' key design. The top left panel shows the characters that can be used in identification, the top right shows the list of remaining taxa, the bottom left shows the list of 'in use' characters, and the bottom right shows the taxa currently excluded.*


Leaf Characters

Leaf Type

simple ✓

once-compound with two leaflets (bifoliate)

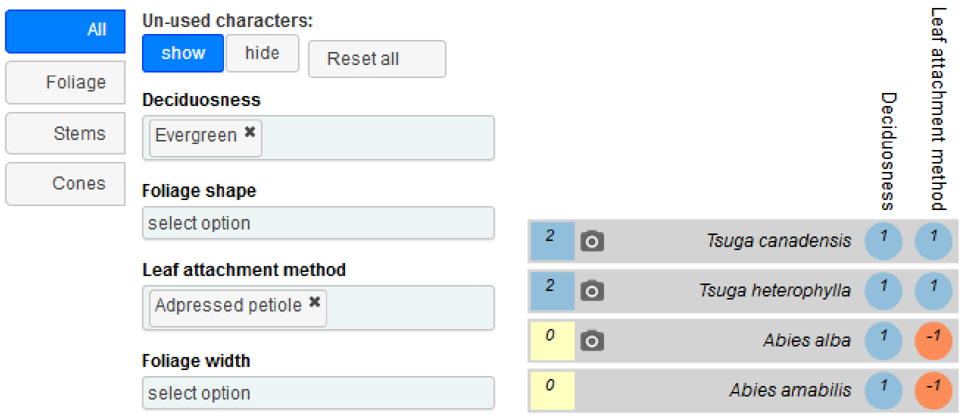
once-compound with three leaflets (trifoliate)

once-compound with leaflets radiating from a single point (palmate)

once-compound with leaflets arranged along an

Figure 1-7 - Example of a smartphone based multi-access key. This example is a screenshot from Weeds of South East QLD and Northern NSW (LucidMobile, 2015b)

This single-column visualisation of the multi-access key displays taxa in a single information-rich column.



Deciduosness: Evergreen ✖

Foliage shape: select option

Leaf attachment method: Adpressed petiole ✖

Foliage width: select option

Score		Taxon	Deciduosness	Leaf attachment method
2	📷	*Tsuga canadensis*	1	1
2	📷	*Tsuga heterophylla*	1	1
0	📷	*Abies alba*	1	-1
0		*Abies amabilis*	1	-1

Figure 1-8 – Partial screenshot of The conifers of Britain guide (Parratt, 2017) showing two questions answered and four taxa, two of which match both answers and have a score of two, two of which match only one answer and have a score of zero.

1.2.3 Photorecognition

Photo-recognition keys work by taking user photos, then through various image recognition techniques, producing either a single or list of potential identifications.

An example of the general 'flow' of a photorecognition app is shown in Figure 1-4. This example workflow is the general workflow; apps generally include some variety of other steps, including (but not limited to) an app opening page that lets you choose between the camera and list of taxa, or a page of further information about possible taxa.

Image-recognition (in general terms) is an increasingly popular area of research and development. Products such as Google Lens provide the general public with the ability to identify various products, locations, and potentially plant and animal species (Google, 2018b). Google lens has only been widely available since early 2018 (Googlephotos, 2018).

If it is assumed that the ability to see 1mm sized details is required for successful photo recognition to species level, smartphone cameras are only capable of producing good enough quality images in some conditions. To resolve the details, the image of these 1mm parts must be sufficiently sperated to remain as separate images on the sensor. If photos in the app are assumed to be taken at 1m, 10cm, and 1cm, and camera details of an HTC One (focal length of 3.82mm and pixel size of 2μm x 2μm (Klug, 2013)), photos taken at 1m will not be able to resolve 1mm details, while photos at 10cm and 1cm will (see Appendix C for maths). With the human eye, cone cells are the equivalent of pixels and are ≥0.5μm (Emsley, 1953). The human eye is capable of seeing 1mm at 2m (Emsley, 1953). When held at roughly the normal distance for specimen observation, the human eye is easily capable of seeing details smaller than 0.5mm. There are several assumptions that have been made here. The maths relating to smartphone optics assuming an all air system and thin lenses, the human eye calculations assume the 'reduced eye' (A way of modelling the optics of the human eye (Emsley, 1953)). These two assumptions will change the exact values but will not change the outcome. A more problematic assumption with the optics shown here is that a resolving power of one millimetre is required; much smaller parts may be required to identify to species level. Details of the hairs of yellow rosette *Asteraceae* are highly useful for identification, for example (Dr M goes wild, 2013).

There are a range of other problems, however, with photo recognition to species level. Other projects aiming to use photo-recognition approaches to identify specimens to species level (or similar) use a contrasting background so the subject can be reliably isolated (Cope *et al.*, 2012; Christodoulou *et al.*, 2018). It is suspected that app users are extremely unlikely to use such backgrounds in real life, and smartphone cameras have very long depths of focus. Subject isolation is therefore going to be much harder. Users are not going to be taking 'perfect' photos. While photo recognition software should be capable of handling imperfect photos, all species will require photos of specific specimen parts, and it cannot be guaranteed that users will photograph these parts. Any postprocessing of the images (either pixel-grouping directly in the sensor, or postprocessing such as sharpening) risks losing small details.

Despite these difficulties, a considerable number of ID key apps are photo-recognition type at 26 of the 147 keys found. 24 of these apps are from separate authors. Of the first 20 results found in the Google Store search, 11 were photo-recognition type, suggesting that despite the limited accuracy, users like the convenience of being able to simply take a photo and receive a specimen identification. Finer details of this preference, such as whether users are aware that photo recognition risks accuracy, or wether this apparent preference is a result of casual users are not examined in this thesis.

Of these 26, four were selected for review; PlantNet (Plantnet-project.org, 2015), Flora Incognita (Technische Universität Ilmenau, 2019), PlantFinder – Flower & Plant Identification (Sleep Sounds Studio, 2019), and plant id (Divers Dias, 2019). There are differences between these apps, including; geographical coverage, number of species, and requirements for the photo(s). Flora Incognita (Technische Universität Ilmenau, 2019) covers the smallest area and number of species, with approximately 4,800 species in Europe. PlantNet (Plantnet-project.org, 2015) covers approximately 20,000 species of various locations. The same number of species, 20,000, is the claimed coverage of plant id (Divers Dias, 2019), but they give no indication of geographical coverage. This induces the same difficulties as detailed earlier in the introduction with Forest Tree Identification (Discovery Green Lab, 2019). The largest claimed coverage is PlantFinder – Flower & Plant Identification (Sleep Sounds Studio, 2019), which claims to cover '90% of all known plants and trees' (Sleep Sounds Studio, 2019). There is further variety between apps with how much information is provided by the user for identification; (Sleep Sounds Studio, 2019) and plant id (Divers Dias, 2019) both take a single photo in app and use it to attempt to identify the specimen, while PlantNet and Flora Incognita both let you use multiple photos while specifying which part of the plant they belong to. Providing multiple photos and specifying which part of the specimen they are is likely to be an attempt to reduce the 'imperfect photo' problem. When investigating the accuracy of the different guides, this resulted in a clear difference in accuracy between apps. PlantNet (Plantnet-project.org, 2015) and Flora Incognita (Technische Universität Ilmenau, 2019) were able to either accurately identify to species, identify to genus, or have the correct identification as a 'second choice' in all test photos. The other two keys had some correct identifications, but also had some incorrect, with most incorrect answers either showing the correct ID as a second choice or showing the correct genus. See Appendix B for test photos and table of identifications reached. On further investigation with the two 'multi-photo' apps, it was found that if you do not have a flower, their ability to confidently identify specimens can decrease rapidly for both apps. It must be noted that most of the test photos are of well-known taxa from the UK, yet only two of the apps produced accurate identifications.

At the time this thesis is written casual use seems to be working; Google Lens adverts indicating plant species recognition tend to include photos of large and distinct flowers. With the issues highlighted above, the detailed and precise identification capable with a well-constructed single- or multi-access key seems currently impossible to achieve with photo recognition app. When smartphone cameras are more capable, and tools such as Google Measure are more mature (Google LLC, 2018c; Siu, 2018), this approach should absolutely be re-visited.

1.2.4 Picture matching

A different approach to plant identification is comparison of specimens to a set of images. This can range from simply comparing specimens to an unsorted set of images, to books where taxa are organised by flower colour and shape (where the user finds the right section of the book, then compares the specimen to all included photographs) such as the National Audubon Society Field Guide to North American Wildflowers–E

(National Audubon Society, 2001). This approach does not require much prior knowledge (Drinkwater, 2009). There are difficulties with this approach; it risks being time consuming, sources often do not include all species in the covered geographic area, and only the most common forms of species are included (Drinkwater, 2009).

1.3 Smartphone ownership

Over 90% of people between the ages of 18-44 in the United States of America (USA) own smartphones, only dropping below 80% ownership in the over 65 age group (DIGITAL, 2016). Ownership rates are similar in the UK, with 90% of 16-24 year olds owning one, and 50% of 55-64 year olds owning one in 2015 (Ofcom, 2015). Smartphones are nearly always much smaller than books, and are ever-increasingly powerful. Their use across the population is only increasing (Ofcom, 2018). Globally, Android (Google, 2019a) controls approximately 76% of the market, with iOS (Apple, 2019) at approximately 24% (correct as of July 2019) (statcounter GlobalStats, 2019). Android is the only operating system developed for and investigated in this thesis, as only devices running Android were available.

1.4 What is missing from the current work?

There are a variety of different ID guide designs. There has been some research into their design. There is also some research into which specific design is better (Tardivel and Morse, 1996; Randler and Zehender, 2006; Sharma, 2016). There are electronic ID keys, a few of which are smartphone based. There is, however, surprisingly little research into the best ID-guide design given how long they have existed, with no research into the best smartphone ID guide design. Here the potential for smartphone based ID guides is investigated. This investigation will focus on the use of multi-access guides; while there are benefits and difficulties to both designs, it was felt that multi-access guides have the significant advantage that they cannot be rendered unusable if a single charictor is missing on a specimen as the user can simply ignore questions relating to it. If the first question in a single-access key asks about smell, and the user is anosmic, then all paths will need to be followed, making the key considerably more difficult. This simply would not happen in a multi-access key. Their use in smartphones specifically is investigated as it is felt that the power and portability of smartphones, along with their widespread adaptation has not yet been properly exploited in ID-guide creation. Approaches to gathering the data for this guide will be investigated; there are a large number of different, already available, guides containing the required data. If data can be extracted from these guides automatically, this opens up exciting possibilities for the use of these data. Finally, the potential costs and income of full-production of these guides will be investigated. It has already been noted that there is the potential for significant income from apps, and their publishing costs are much lower than for academic papers. The question remains, however, of production costs. Thus, the following hypothesis and aims are proposed.

1.5 Thesis overview

This thesis covers three main areas: how do we get the data for a plant ID app; how should a plant ID app actually look; and whether or not production of such an app at higher quality would make financial sense.

The first three chapters investigate app databases and how to fill them. This starts with an investigation of how big an Android app's database can be. An investigation into automatic extraction of data from pre-existing paper based sources follows. The third of these chapters reverts back to manual extraction of data from the pre-existing sources, providing a detailed method on merging information from several sources with the aim of producing a thorough database.

The second three chapters follow the complete process of designing, implementing, and testing and evaluating an app. The first of these chapters looks at the user interface; investigating the design of pre-existing apps, and building a design that hopes to be simple for the end-user. The design of the code is detailed in the following chapter. The last of these three chapters details all rounds of user testing and evaluation that took place, and what changes were put in place to answer the feedback generated.

A single chapter then examines the economics of such an app are examined in depth.

The final chapter draws all these preceding ones together, providing an overview of the project. This chapter includes the conclusions and a suggestion on an appropriate path for continuations of such projects.

1.6 Aims and hypotheses

1.6.1 Hypotheses

1.6.1.1 With the rapid increase in power, natural language processing is no longer required to extract the data from floras

Given that there have been attempts using natural language processing techniques to extract the data from floras, is this necessary, or is a brute force approach now viable?

1.6.1.2 Everyday Smartphone technology is now sufficient to operate a complete technical key to the entire British Flora

With the rapid increase in power of modern smartphones, are they now capable of storing and processing the required data for a key to the entire Flora?

1.6.2 App Design approach

1.6.2.1 If there is a best layout for an electronic flora designed to be used on smartphones, what is it?

There are several extant electronic ID keys, many of which have a similar design theme. Is this design theme a good approach?

1.6.3 Aims

The main aim of this thesis is the design, construction, and testing of a smartphone-based multi-access ID guide. This will allow investigation of all salient points, thus the following sub-aims are proposed.

Note that from this point onwards, the smartphone-based multi-access ID guide constructed and discussed in this thesis shall be referred to as 'the app', unless otherwise inferred or required by context (such as when discussing a pre-existing app), in which case every attempt for clarity has been made.

1.6.3.1 Production of a useful database that can be used for species identification

The app will need a data-source it can search, thus a database will be needed. Investigation of the best approach to construction of this database will be required; a database to the whole British isles would require at least 4,800 entries (Stace, 2010c), and construction would take a considerable amount of time. Methods that can reduce this will therefore be investigated.

1.6.3.2 Production of a high-quality smartphone app that can make use of this database

In order to test ID guide design, an ID guide must exist. There are pre-existing smartphone-based guides and the MSc work (Bewsey, 2014); from these we can investigate, test, and evaluate smartphone based ID guide design.

1.6.3.3 Investigation into whether production of the app would make a profit or not

If it is possible to make a profit, then more work in this project should be undertaken with little hesitation. With a profit, work of this sort will not only have scientific value, but monetary value as well.

2 Databases (part one): database size vs speed

2.1 Introduction

There is evidence that the delay that humans perceive as instant or near instant is below 100ms (Card *et al.*, 1991), potentially as low as 13ms (Potter *et al.*, 2014). Google have found that inducing an artificial delay of lower than half a second when returning results from a web-search reduces the number of searches a user undertakes in a day, with various levels of delay giving different levels of impact (Brutlag, 2009). This artificial delay can result in a reduced number of searches after the delay has been removed (Brutlag, 2009). More than three seconds can lead to approximately 40% of shoppers leaving a website (Google, 2015). Smartphones, as opposed to computers, are used in shorter bursts with information wanted as soon as possible (Google Inc., 2012).

SQL (Structured Query Language) is a programming language designed for use with, and for the management of, databases (w3schools, 2019b). For a brief further introduction to SQL, see Appendix F . SQLite is an implementation of SQL as a C-language library, designed to be implemented within other applications (SQlite, 2019). SQL is a popular databasing language (Stackoverflow, 2017). Android has full support for the SQLite database format (Android Developers, 2017).

A range of approaches for increasing SQL database handling speed have been published (Koch, 2017; Zaitsev, 2009, as a minimal example set). These articles typically relate to server databases, however, which have different hardware to mobile devices (A modern server can have 1.5TB of ram (Dell Inc, 2018), 375 times the 4GB of the Galaxy S8 (GSMArena, 2017), for example). The difference in data read speeds for different hardware protocols is shown in Table 2-2, showing that in most cases, desktop hardware can read data significantly faster than smartphone hardware. Despite superficially similar specifications when comparing mobile processors against typical desktop processors (see Table 2-3), comparisons are difficult because of their different designs (Sims, 2014 and Techquickie, 2015 serve as an introduction to the subject area). Benchmarking software is frequently considered imperfect for comparison purposes, with different suites giving different results (Ung, 2015, for example). Fortunately, Geekbench 4 (Primate Labs Inc, 2018d) a suite of tests available for cross-platform compute power comparison, is considered better quality than previous benchmarks

(Torvalds, 2016), and is used by many companies in the industry (Primate Labs Inc, 2018d). This can then be used to compare different smartphone and desktop grade processors. A comparison of GeekBench4 (Primate Labs Inc, 2018d) single- and multi-core scores are shown in Table 2-1 for pairs of similar age desktop and mobile device chips.Note that due to the more variable design of smartphones and mobile processors, no average score is available on the Geekbench 4 (Primate Labs Inc, 2018d) website , so the score reported is from Ruddock (2017), to act as a 'same place same time' reference for each test.

Table 2-1 - Geekbench 4 (Primate Labs Inc, 2018d) scores for various CPUs. [1](Intel, 2018c), [2](Intel, 2018d), [3](Intel, 2018f), [4](GSMArena, 2015b), *[5](Zimmerman, 2016), [6](Cheng, 2016), [7](Primate Labs Inc, 2018a), [8](Primate Labs Inc, 2018b), [9](Primate Labs Inc, 2018c), [10](Ruddock, 2017).*

Processor type	**Processor**	**Model year**	**Single-core**	**Multi-core**
Desktop	i7-6700k	(Q3) 2015[1]	5337[7]	17399[7]
	i7-7700k	(Q1) 2017[2]	5700[8]	18793[8]
	i7-8700k	(Q4) 2017[3]	5914[9]	25945[9]
Mobile	msm8996 (SD820)	2015[4]	1450[10]	3800[10]
	msm8996 (SD821)	2016[5]	1840[10]	4032[10]
	msm8998(SD835)	2017[6]	2059[10]	6461[10]

A modern work on database handling speed in Android is Feinstein's (2017) blog on getting the best performance possible out of SQLite in Android. He concluded that batch inserting in a single transaction using SQLiteStatement was the fastest method of inserting data to an SQLite database (Feinstein, 2017). Individual SQLite statements require a transaction, and Android allows you to wrap multiple statements into a single transaction (Dubey, 2015).

Table 2-2 - Data transfer speeds of various hardware types. [1] (PCI-SIG, 2017) in (Oh, 2017)*; [2] (SATA-IO, 2018); [3] (Wimmer, 2013); [4] (Wimmer, 2014a); [5] (Masiero, 2015); [6] (Osthoff, 2016); [7]* (Herbrich and Schmidt, 2017)*; [8] (Sold, 2013); [9] (Wimmer, 2014b); [10] (Wimmer, 2014c); [11] (Moser, 2015); [12] (Wimmer, 2016); [13] (Schwabe, 2017). For NotebookCheck sourced data, the AndroBench 3-5 Sequential Read 256KB read has been used.*

Type	Model	Specific	Speed (MB/s)
Desktop Hardware protocol	PCIe 1.0	1x	250.00[1]
		16x	8000.00[1]
	PCIe 2.0	1x	500.00[1]
		16x	16000.00[1]
	PCIe 3.0	1x	1000.00[1]
		16x	32000.00[1]
	PCIe 4.0	1x	2000.00[1]
		16x	64000.00[1]
	PCIe 5.0	1x	4000.00[1]
		16x	128000.00[1]
	SATA	1.x	150.00[2]
		2.x	300.00[2]
		3.x	600.00[2]
		3.2	1969.00[2]
Mobile phone	Samsung Galaxy	S4	75.10[3]
		S5	83.53[4]
		S6	319.00[5]
		S7	483.82[6]
		S8	792.86[7]
	Sony Xperia	Z	48.79[8]
		Z2	62.09[9]
		Z3	149.00[10]
		Z5	219.84[11]
		ZX	281.00[12]
		ZX1	679.13[13]

Table 2-3 - Core count and CPU clock speed of a selection of example processors. [1] (Intel, 2018a); [2] (Intel, 2018b); [3] (Intel, 2018e); [4] (Qualcomm, 2018a); [5] (Qualcomm, 2018b); [6] (Qualcomm, 2018c)

Manufacturer	**Chip**	**Number of CPU Cores**	**CPU clock speed (GHz)**
Intel	i3-8100	4[1]	3.60[1]
	i5-8400	6[2]	2.80[2]
	i7-8700	6[3]	3.20[3]
Qualcomm	MSM8940	8[4]	1.40[4]
	SDM630	8[5]	2.20[5]
	835	8[6]	2.45[6]

2.1.1 ANSI (Windows text encoding)

ANSI, or Windows-1252 or CP-1252 is a Windows character set (Microsoft, 2018a) that encodes for 256 potential characters (Microsoft, 2018a). Each character requires 1 byte of storage (Microsoft, 2018c). More characters of text cause larger files, but with text compression this increase may not be linear.

2.1.2 Search Design

The generic SQLite search query can be described as 'SELECT * from *', with '*' representing a wildcard character (SQLite, 2018). As a search term, this is widely understood to be bad practice as a search term for various reasons, including slow performance (Winand, 2013; Köhntopp, 2016; DPriver, 2017).

A more specific SQLite query can be described as 'SELECT * from * where x=y and a=b', where x and a are columns and y and b are the desired values from these columns (SQLite, 2018). This leads to the question of whether number of columns affects load and/or search time of SQLite databases in Android, an area that appears to have not been investigated.

2.2 Hypothesis tested and goal of these tests

2.2.1 Hypothesis

On Android devices, databases of the same number of cells load at the same speed irrespective of shape.

2.2.2 Aim

Determine the database handling method and database size that provides a rapid search speed when using the app. For the purposes of this chapter, a maximum load or search time of 3 seconds will be used. While instant (or near instant) would be preferred, proving this speed has been reached across all possible devices would be impossible, and increasing device power over time will help reduce the time required.

2.3 Materials and Methods

The devices used for this set of tests were an LG Gpad 8.3 (GSMArena, 2016) and a LG Google Nexus 5 (GSMArena, 2015). The database from chapter 3 was loaded in Microsoft Excel (Microsoft, 2016), and modified by removing columns from the right hand side, or removing rows from the bottom. Modified databases were saved in .CSV (Shafranovich, 2005) format for consistency and compatibility reasons.

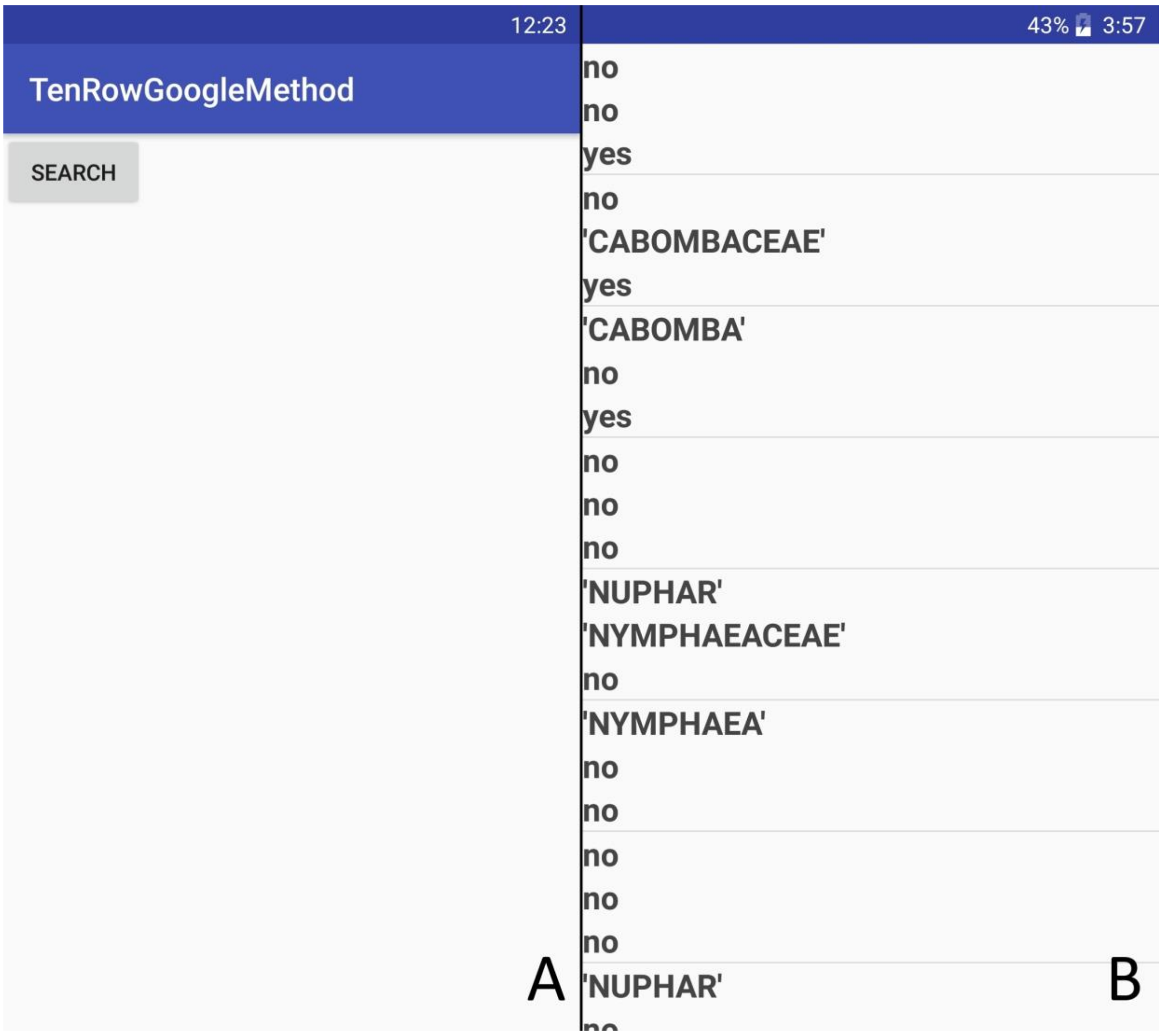
For the purpose of this chapter long and wide databases are defined here:

- Long databases are defined where there are many rows, and a reduced number of columns
- Wide data bases are where there are many columns, and a reduced number of rows.

In order to test handling speed differences between wide and long databases, this method was used to create 7 different database 'sizes', the full size database; database variations with the size reduced to either 2/3 or 1/3 the number of rows or columns, and a minimal 10 rows or columns. This gave long databases of 4028 rows by either 10, 260, 550, or 785 columns, and wide databases of 785 columns by either 10, 1342, 2685, or 4028 rows. Note that the largest two sizes on each 'shape' are identical.

2.3.1 Test apps

A separate test app was created for each combination of database size, shape, and handling method. Test apps were designed to load mainActivity page on start-up, displaying a single standard button. This button performed the search, and loaded the results page. These pages are shown in Figure 2-1. The results page was designed to be a self-filling list view of all returned results from the search, using a separate results display for each entry in the list. This design ensures minimal work is needed to load either page, and any delay is the effect of the database design.



```
no
no
yes
no
'CABOMBACEAE'
yes
'CABOMBA'
no
yes
no
no
no
'NUPHAR'
'NYMPHAEACEAE'
no
'NYMPHAEA'
no
no
no
no
no
'NUPHAR'
```

Figure 2-1 - Example mainActivity (A), shown on start up, and results list (B), shown after pressing search, from a database test app.

Each app was coded to load the database on app launch. Databases for this project were stored in apps using the SQLite format (Hipp *et al.*, 2018). Apps were programmed to handle these databases using either hardcoded insert statements (where the SQLite insert statement is directly encoded into the apps code), premade SQLite databases (Fluxà, 2009), or the method described by Google (Android Developers, 2015). Hardcoded insert statements were generated via a bespoke Java (Oracle, 2017b) program created on IntelliJ IDEA (JetBrains, 2018). Premade SQLite databases were created using SQLite expert (Sqlitebrowser, 2017). The search term used in the test apps will be 'SELECT * from (table name)'.

Database size vs handling speed tests were run twice; using the long set and wide set of databases in order to test whether 'shape' has any effect on handling speed. For every individual database, the device running the test app was re-started, and left for 5 minutes to finish start up procedures. After start-up, the test app and the stopwatch (the stopwatch on the Google clock (Google LLC, 2017) was used, allowing timing to 10ms accuracy) were started simultaneously. The stopwatch was run on the device not currently running the test app. As soon as the app finished loading, the stopwatch was stopped, and the time as displayed was noted. The search button was then pressed, and time required was measured using the same approach. This procedure represented a single test of a single database on a single device. Each single test was repeated three times, and all databases were tested on both devices. App crashes were defined as unexpected forced app closure events, and their occurrences were noted. Regressions of size vs handling speed were compared following the advice described by Frost (2016).

2.3.2 Data analysis

Mean app load and search times, and variances around means were plotted against relevant database size. Times were transformed to $\log_{10}(s)$ to ensure that all data were visible on the graph. Appropriate database sizes were chosen from these graphs, with an upper bound of 3 seconds for either function. This approach was chosen over a more exact approach as different conditions (i.e. different devices in different setting running different databases and different searches) will not perform exactly the same, so there was not enough data to set an exact limit with any confidence.

2.3.2.1 Comparison of long vs wide database loading times

In order to compare between wide and long databases, database size was normalised to load time per cell. This was done by dividing the average total load or search time by number of cells in the test database the app was handling.

This resolves the issue that the asymmetrical database produced in chapter 3 (used in these tests) does not produce identically sized databases when reduced in size in width and height by the same ratio independently. I.e. due to the shape of the original, the 10 row test database did not have the same number of cells as the 10 column database.

Note that this test was on s data, and not done on the $\log_{10}$(s) data used for the graphs in section 2.4.1; the $\log_{10}$(s) was used there to ensure visualisation was possible.

Homoscedasticity of regressions being compared was tested using the spreadsheet running Bartlett's test for homogeneity of standard deviations provided by McDonald (McDonald, 2014). Homo- and hetero-scedasticity are ways of describing standard deviations of a group of measurements, with homoscedasticity meaning the standard deviations are all the same, and heteroscedasticity meaning they are different from each other (McDonald, 2014). If heteroscedasticity is exhibited, the chance of obtaining a false positive becomes unacceptably high (McDonald, 2014). In order to remove any potential issues, should heteroscedasticity be exhibited by your data, the advice is to attempt different data transformations, and if that fails, use different testing methods (McDonald, 2014). Regressions were compared using the Comparing Coefficients in Regression Analysis described by Frost (2016). Change in loading time with changing database size for the same device, using the same handling method, running the same test, but with different shaped databases were compared in these regressions (See Figure 2-4 for an example of compared regressions).
The null hypothesis in this test is shown in 2.2.1.

2.4 Results

2.4.1 Database handling method

Hardcoded insert statements had the fastest combined time to load and search, taking approximately 9 seconds(s) maximum to search, and consistently less than 1 second to load. The other methods took longer, up to over 1000 seconds (approximately 16 minutes) in some cases. Hardcoded statements were also the only method that consistently caused app crashes at the largest database sizes, with the LG Gpad being unable to load the largest database tested. All other methods successfully load all database sizes on all devices.

These data suggest approximately 1.5 million cells (approximately either 370 columns x 4028 rows or 1900 rows x 785 columns based on the test database) as a recommended maximum size flat database. With the predominantly binary data in this dataset, this is roughly equivalent to a 4mb .csv (Shafranovich, 2005) file. This gives a search time of approximately 3 seconds on the test devices. The timing data are shown in Figure 2-2 and Figure 2-3. The times are all shown in $\log_{10}(s)$ for consistency; the Google Method and Premade database method graphs all required the use of $\log_{10}(s)$ for data to remain visible. Without transformation, there would be no way to easily plot the graphs. 1000 seconds is roughly equivalent to 16 minutes, which on Figure 2-2 start-up timing, for example, would push the faster start times to being indistinguishable from the y=0 line.

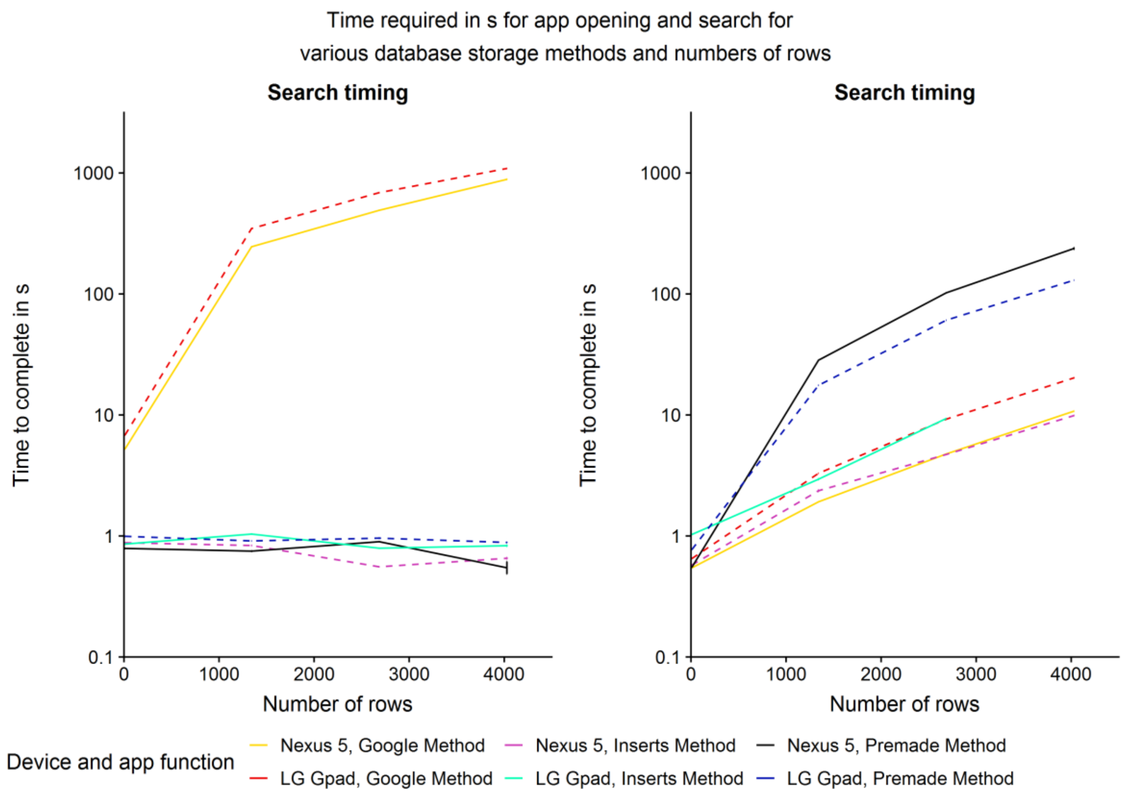


Figure 2-2 - App opening and search times in a Nexus 5 (GSMArena, 2015a) and LG Gpad (GSMArena, 2016) for various combinations of database sizes and database handling methods, when the database is expanded by modifying the number of rows. Error bars are ±1 standard error.

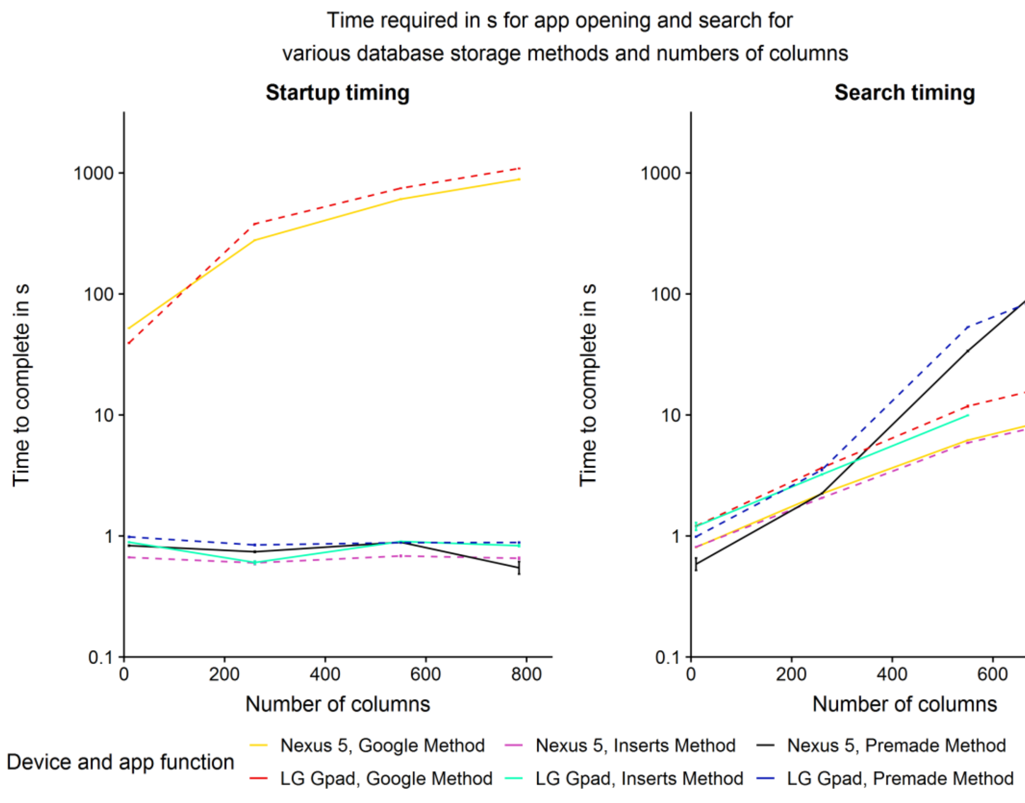


Figure 2-3 - App opening and search times in a Nexus 5 (GSMArena, 2015a) and LG Gpad (GSMArena, 2016) for various combinations of database sizes and database handling methods, when the database is expanded by modifying the number of columns. Error bars are ±1 standard error.

2.4.2 Long vs wide databases

Bartlett's test indicated no transformation of the seconds per cell data was required. Each separate pair of regressions for the same method on the same device (see Figure 2-4 for an example) was compared with no significant difference in app start or search times found between databases where size had been increased via length vs database where size had been increase via width. This accepts the null hypothesis that database shape has no effect on load time. These data are shown in Table 2-4.

Table 2-4 - Bartlett's test P values and coefficient comparison in regression analysis P values when comparing change in load time per cell for increasing size of long vs wide databases

Device	**Method**	**Action**	**Regression comparison P value**	**Barlett's P value**
Nexus 5	Google	Start	0.395	0.490
Nexus 5	Google	Search	0.298	0.418
Nexus 5	Inserts	Start	0.218	0.538
Nexus 5	Inserts	Search	0.284	0.453
Nexus 5	Premade	Start	0.249	0.613
Nexus 5	Premade	Search	0.407	0.097
Gpad	Google	Start	0.808	0.896
Gpad	Google	Search	0.35	0.461
Gpad	Inserts	Start	0.246	0.614
Gpad	Inserts	Search	0.262	0.53
Gpad	Premade	Start	0.243	0.609
Gpad	Premade	Search	0.265	0.537

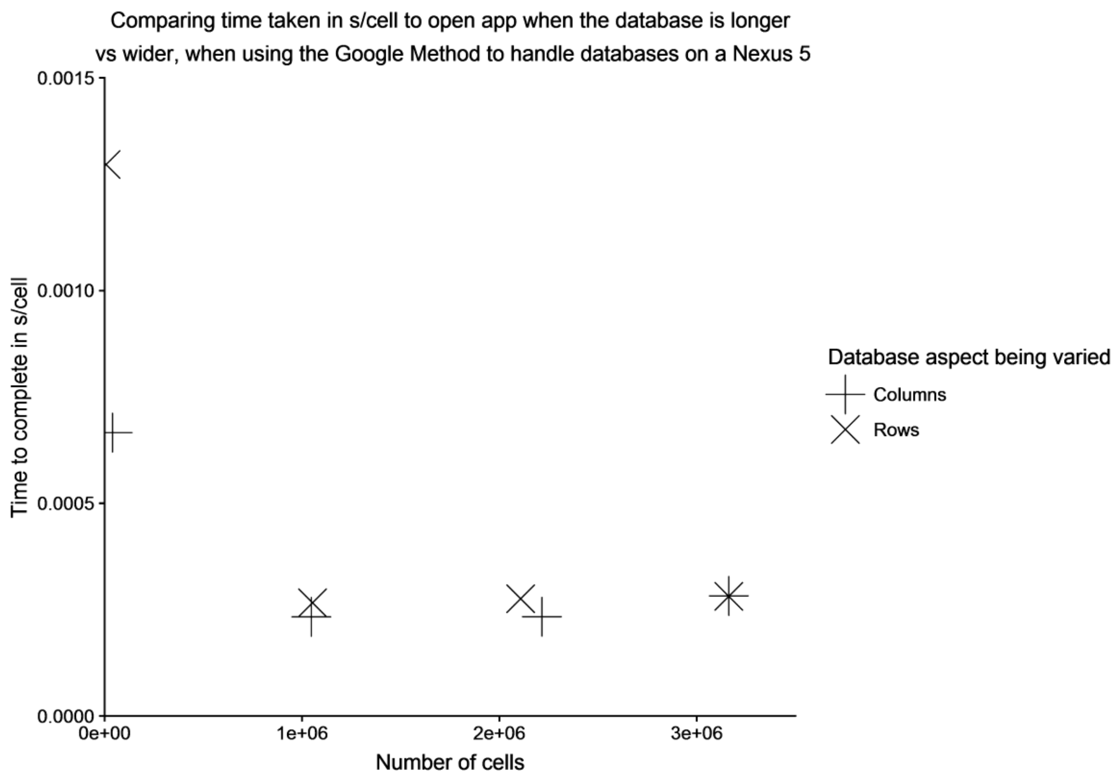


Figure 2-4 - Comparing app opening times for wider vs longer databases when times are adjusted to per cell

2.5 Discussion

The results suggest a database size of approximately 1.5 million cells, or a 4mb .CSV (Shafranovich, 2005) file. This database size gives 3s as the longest the user is likely required to wait when using devices with equivalent processing power to the test devices. With smaller datasets, this drops below 1s rapidly. As this test was running the worst-case search query in terms of search speed, this is very positive. This was not below the suggested 100ms (Card *et al.*, 1991), but with increasing device power this mark will eventually be reached. Using Geekbench scores (Geekbench Browser, 2018) and data transfer speeds (Schuster, 2013) for the LG Nexus 5, and the data from tables Table 2-1 and Table 2-2, it can be predicted that this will happen between 2030 and 2050. It must be noted that this is a crude approximation based on a linear improvement in scores and speed, and that significant changes in technology or programming will likely affect this.

The devices used in this test were available when the tests were being ran, and represented a good approximation of mid to mid/high range devices available. The unavailability of lower powered devices necessitated the decision to aim for a nonspecific rapid loading speed on mid-ranged hardware; lower powered devices would then hopefully be able to successfully run the app.

As the file size suggested is 4mb, with a time requirement of approximately 3s, it can be inferred from the data transfer speeds in Table 2-2 that processing rate is the limiting factor, rather than data transfer speed. The failure of the LGGpad 8.3 (GSMArena, 2016) to complete with the Insert method search at the largest database size, as shown in figures Figure 2-2 and Figure 2-3, was unfortunately un-explained, but the much lower times required by this method suggest that it should be the preferred method. Care should be taken to ensure that low power devices will still safely run the app.

The full dataset tested (4028 rows x 785 columns, or 3.1 million cells) was based on the work in chapter 3, and was a partially complete dataset extracted from Stace (2010c). Every taxon present in Stace (2010c) was included, but the set of features was incomplete. A full national flora on a single flat database is thus not yet easily achievable on mobile devices. This would also not be wise; as will be discussed later, it would lead to a clumsy design.

Pleasingly, testing longer (more rows) vs wider (more columns) flat databases shows that databases of the same total cell count are handled at the same speed. This is unsurprising, disc performance is frequently cited as having the biggest effect on search performance (DPriver, 2017; Köhntopp, 2016; Winand, 2013, for example), rather than processing the search query. Wider databases, with more columns and potentially more complex searches, are unlikely to affect app performance if the database is below the suggested maximum size.

In all tests, seconds per cell loading load or search time change with increasing database size showed no sign of heteroscedasticity ($P \geq 0.418$ on Bartlett's test). Lower P values occur with either very long (several minute) or very short (at or below 100ms) start or search times. This is evident with app start time when using the Google method and direct insert statements. This is likely to be due to the difficulties in producing consistent timing data by hand at these scales. As discussed by Card *et al.* (1991) and Potter *et al.* (2014), below 13ms is approaching the time that humans perceive as instant. External influences will affect accuracy when timing over several minutes. The reduced confidence in the data in these tests at these time scales is acceptable, the time scales are either much faster or much slower than the targeted 3s.

Given the contents of each cell in this test database (almost entirely either 'y' or 'n'), there is justification for re-running the test with more data in each cell; both random data with a predetermined average specific string length (to eliminate any 'hotspots' of long strings) and a normal database should be used. The size of these data should be increased from the minimums used in this chapter until the load time increases above the 3s threshold. The different database contents (random data vs normal data) will act as a check against unexpected effects of uneven databases. If a different database design is used (i.e. full words in each cell, not 'y' or 'n'), this will aid investigation of the highest possible cell count and therefore largest useful database size.

Re-running with transactions, as described by Feinstein (2017), would also improve the results, and would almost certainly suggest a larger flat database can be used. As the flat database size suggested already allows enough data storage for the rest of the work in this thesis, this re-test is not necessary as it is likely to allow inclusion of even more cells. Unfortunately this (Feinstein, 2017) work was only available after the tests described in this chapter were undertaken and they were not considered.

Re-running this test in the future, when mid-range devices are more powerful is also unlikely to be strictly necessary unless significant changes to the Android operating system occur, for exactly the same reason of already plentiful flat database size. Re-running on more modern hardware, and with transactions (Feinstein, 2017), may however bring the possibility of a national flora on a single flat database closer, should the suggested flat database size become large enough.

The aim of this test was to establish the largest viable flat database that can easily be handled by Android devices, as there is no guarantee of internet access in the field. Load times from the internet were not relevant to this test, and will also depend on the internet connection speed as well as the database size. It should not, however, be completely ignored, if the app is published. Users will download the app from the Google Play Store (Google, 2018d). This is also where additional modules will be available, should the data be modularised. While unlikely to cause a noticeable difference, having a smaller database size will result in smaller module sizes and faster download and install speeds.

2.6 Conclusion

The tests in this chapter show that an approximately 1.5million cell/4mb .csv (Shafranovich, 2005) database and converting to hard-coded insert statements is by far the best experience for the user, based on these tests. While not large enough for a national flora, work later in this thesis will demonstrate that this is plenty of room for a database describing *Equisetum*, therefore is likely enough for individual families or small sections of a national flora.

The further work and refinements to the method suggested are likely to increase the suggested database size, as are increases to device power.

3 Databases (part two): Natural Language Processing, Information Extraction, and their use in botanical database construction

3.1 Glossary

To avoid any potential confusion, definitions (and the source these definitions are based on) of some words as used in this chapter and its references are displayed here.

- Information extraction (IE): The process of gathering structured factual information from unstructured text (Piskorski and Yangarber, 2013).
- Lexicon: A vocabulary, or list of all meaningful units of a language (Lexico, 2019), including information about what categories the unit belongs to (i.e. whether the word can be used as a noun, adjective, e.t.c.) (Wilson, 2012).
- Natural Language Processing (NLP): Computerised programmatic analysis of natural human written language (Liddy, 2001). Generally NLP uses IE techniques to process the text.
- Ontology: specifications of concepts and the relationships between them (Gruber, 1995). An example general ontology covering Plantae, Animalia and Fungi, providing, amongst other factors, consistent labels for organism axis can be found in Dahdul *et al.* (2014).
- Parsing: process of analysing text to describe roles of separate component parts (Oxford English Dictionary, 2018)

3.2 Introduction

NLP and IE systems typically are designed to work on 'true' natural language, or freely created human language. Taxonomic descriptions are semi-structured, but attempting NLP and IE on these texts is still challenging (Lydon *et al.*, 2003). An example description is shown in Figure 3-1, where it can be seen that the language is a list of phrases, (typically) with a feature and descriptive words. This has been noted before, for example in Taylor (1995) and Wood *et al.* (2004). Given the amount of botanical descriptive literature, the only realistically viable approach is automation of either some or all the steps required to transform the data to electronically searchable forms (Lydon *et al.*, 2003).

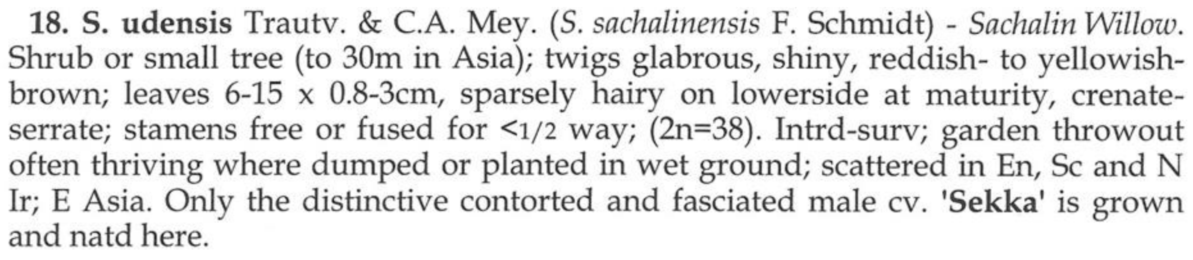

18. S. udensis Trautv. & C.A. Mey. (*S. sachalinensis* F. Schmidt) - *Sachalin Willow*. Shrub or small tree (to 30m in Asia); twigs glabrous, shiny, reddish- to yellowish-brown; leaves 6-15 x 0.8-3cm, sparsely hairy on lowerside at maturity, crenate-serrate; stamens free or fused for <1/2 way; (2n=38). Intrd-surv; garden throwout often thriving where dumped or planted in wet ground; scattered in En, Sc and N Ir; E Asia. Only the distinctive contorted and fasciated male cv. **'Sekka'** is grown and natd here.

Figure 3-1 - An example description of a taxon from Stace (2010c)*, in this case* S. udensis*. Note that this text has been reproduced by scanning from Stace* (2010c).

The variety of NLP and IE systems in biodiversity are discussed thoroughly by Thessen *et al.* (2012). These systems range from simple approaches to more sophisticated, and often different approaches IE and NLP are used within the same system to complete the required task (Thessen *et al.*, 2012).

Simpler approaches include syntactic parsing (Thessen *et al.*, 2012). Syntactic parsing identifies the function of different parts of a sentence, with shallower (simpler) parsing identifying different 'chunks' of the sentence, while deeper parsing can identify function of individual words. In NLP systems this information about function of different words is then used to extract meaning.

Wood *et al.* (2004) and Taylor (1995) used the simpler structure of botanical literature to help with shallow parsing; discourse segments (segments of text containing a head (feature) and its set of characteristics) were extracted by splitting descriptions into component parts. Both then used to fill a pre-made ontology with head/character states. Taylor (1995) manually created a set of 70 definite clause grammar rules. These rules were combined with a lexicon built from plant systematics textbooks to extract character/state pairs from the texts that had previously been 'chunked' (Taylor, 1995). McGee Wood *et al.* (2004) used an ontological gazetteer (a lookup list, where each feature in the ontology has a set of options that may be found in the text) on this 'chunked' text to link heads (features such as leaf, or colour) with features (options that relate to the head, such as frond, or yellow). This ontological gazetteer filled in the pre-made ontology (McGee Wood *et al.*, 2004).

Use of pattern matching can also range from simpler to more complex. In its basic form, pattern matching is the process where a string of characters is searched for in a long text (e.g. 'ana' would be matched within 'banana' but would not pattern match within 'apple'). This specific approach does not extract meaning, it simply highlights occurrences in the text. Krauthammer *et al.* (2000) use pattern matching to locate a list of protein or gene names within journal articles. Their approach was to convert each letter of both the gene/protein name and the target journal article to predetermined nucleotide combinations, and use BLAST software to locate incidences of the gene/protein name. The 94.6% accuracy recorded was lower than the expected 100% as fragments of gene names were matched incorrectly within the full gene (e.g. *notch* being marked as a partial match for *notchl*) (Krauthammer *et al.*, 2000). Pattern matching can be used in IE and NLP; Thessen *et al.* (2012) describe a situation where the rule 'x activates y' and a list of enzymes are used in a program analysing a relevant text. When the program finds an enzyme in the text, it assumes the rule 'activates y', and can highlight the 'y'.

There are a number of more complex methods (Thessen *et al.*, 2012). These are: full parsing, probability based, mixed syntactic-semantic, and sub language-driven approaches (Thessen *et al.*, 2012). These provide different approaches to gathering greater and greater levels of information about full sentences with reduced reliance on prior knowledge (Thessen *et al.*, 2012). An example of this greater level of information is shown in Figure 3-2, which has been reproduced from Thessen *et al.* (2012), where the grammatical function of each word within the sentence is extracted.

The basic functionality of this app is based largely on the preliminary work in the MSc project (Bewsey, 2014). The multi-access key search from this previous work functioned correctly, so was retained. This meant that the final datasheet had to retain the same flat structure, with 'yes' or 'no' cell contents, and the morphological data being encoded as the column heading. All that is required to fill in the database is presence or absence of features, achievable with either basic pattern matching techniques or the discourse segment analysis shown by (McGee Wood *et al.*, 2004). As a basic example, the description shown in Figure 3-1 contains the clause "Shrub or small tree (to 30m in Asia)". If the column headings shown in Table 3-1 are used, then the headings "shrub" and "small tree" would match, while "herb" and "tall tree" would not. This would produce Table 3-2. With enough headings, all possible features may be covered. Thus the more advanced NLP IE methods provide unnecessary information, it not a requirement of this database to know the function of specific words within a sentence.

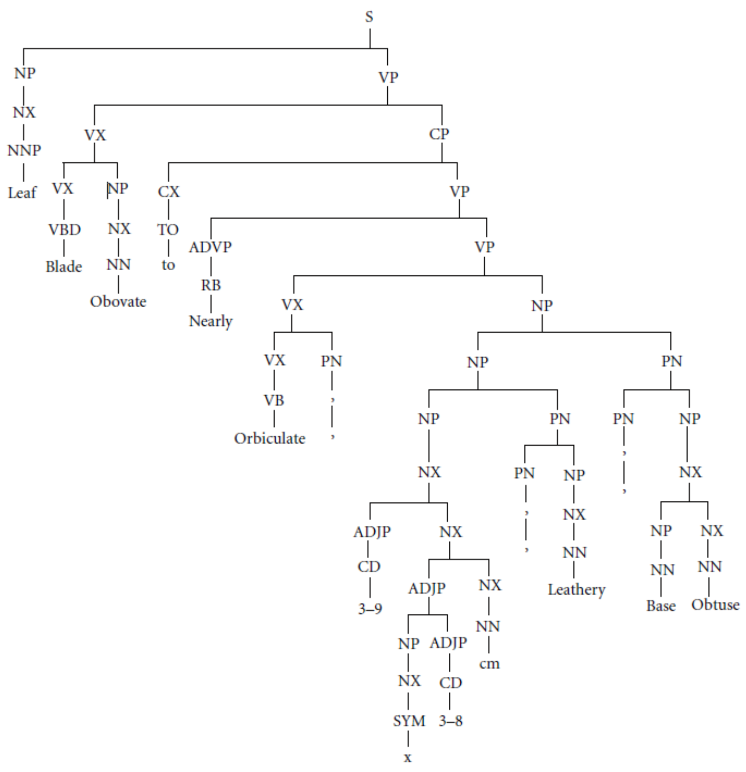


Figure 3-2 - The results of a deep parsing analysis using Enju Parser, showing the identification and function of each word and punctuation mark. Figure adapted from Thessen (Thessen *et al.*, 2012).

Table 3-1 - Small example set of column headings. These column headings would act as patterns, for a pattern-matching based IE program.

Taxon	Herb	Shrub	Small tree	Tall tree

Table 3-2 - The resulting table when a pattern matching IE program uses the headings in Table 3-1 on the text from Figure 3-1

Taxon	Herb	Shrub	Small tree	Tall tree

S. udensis Trautv, C.A. Mey	No	Yes	Yes	No

3.3 Aim

The aim of the work in this chapter is to investigate whether a database suitable for the app produced later in this project can be generated using simple pattern matching techniques and a single data source.

Pattern matching was chosen as there is no requirement for exact word meaning to be extracted from the text. As has been detailed before, taxonomic descriptions are lists of features and their descriptions; all that is required for this project is establishing whether the feature and its exact description are present. While this method will result in false matches where modifier words are present (e.g. 'not hairy' will still match the pattern 'hairy'), further columns with the modifier words are part of the pattern (e.g. a pattern of 'not hairy) should allow for accuracy.

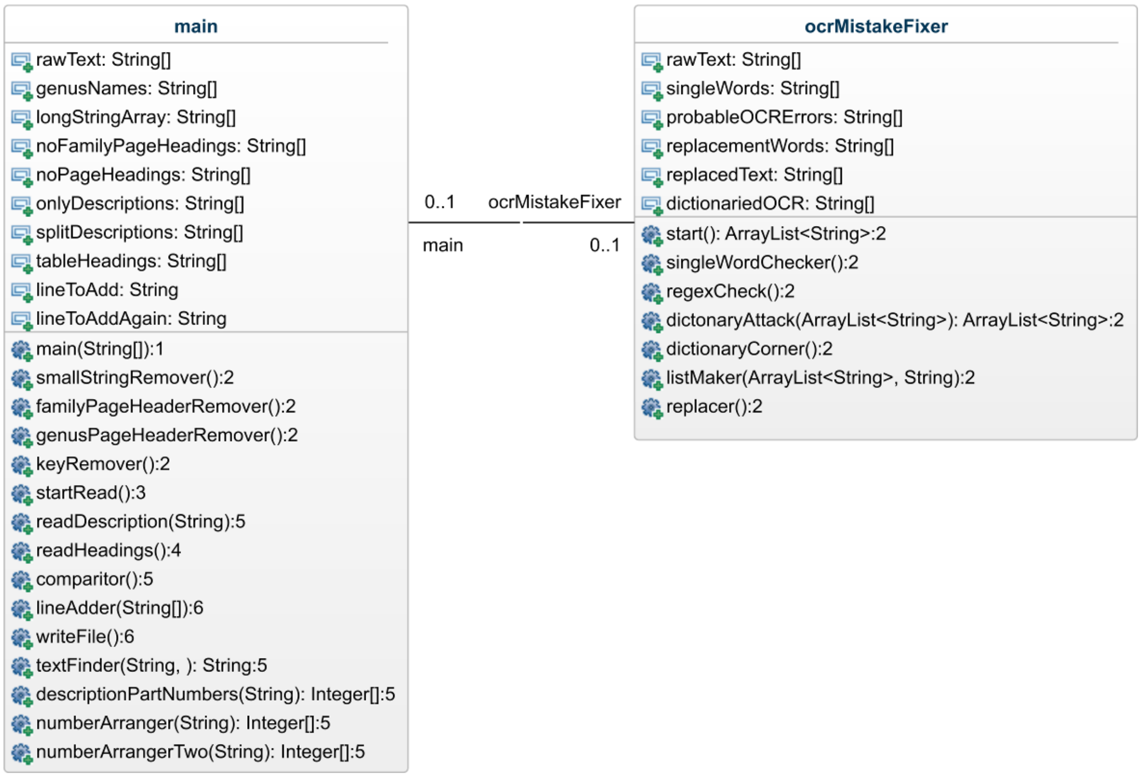
This use of a single data source allows proof of concept testing. Extraction from multiple data-sources and post-extraction data merger will follow if this test is successful.

3.4 Method

Stace (2010c) was scanned using an Epson Expression 1640XL (Epson, 2017) connected to an HP Elite Desk 800 SFF (CNet, 2018) running versions of Windows 7 (Microsoft, 2009) provided and modified by the University of Reading. Scanned pages were processed in ImageJ (Schneider *et al.*, 2012), isolating and separating individual pages. These isolated pages were saved in order in .PDF format. These PDF files underwent OCR processing in Adobe Acrobat (Adobe, 2015), with the output compiled into a single .UTF8 .txt file.

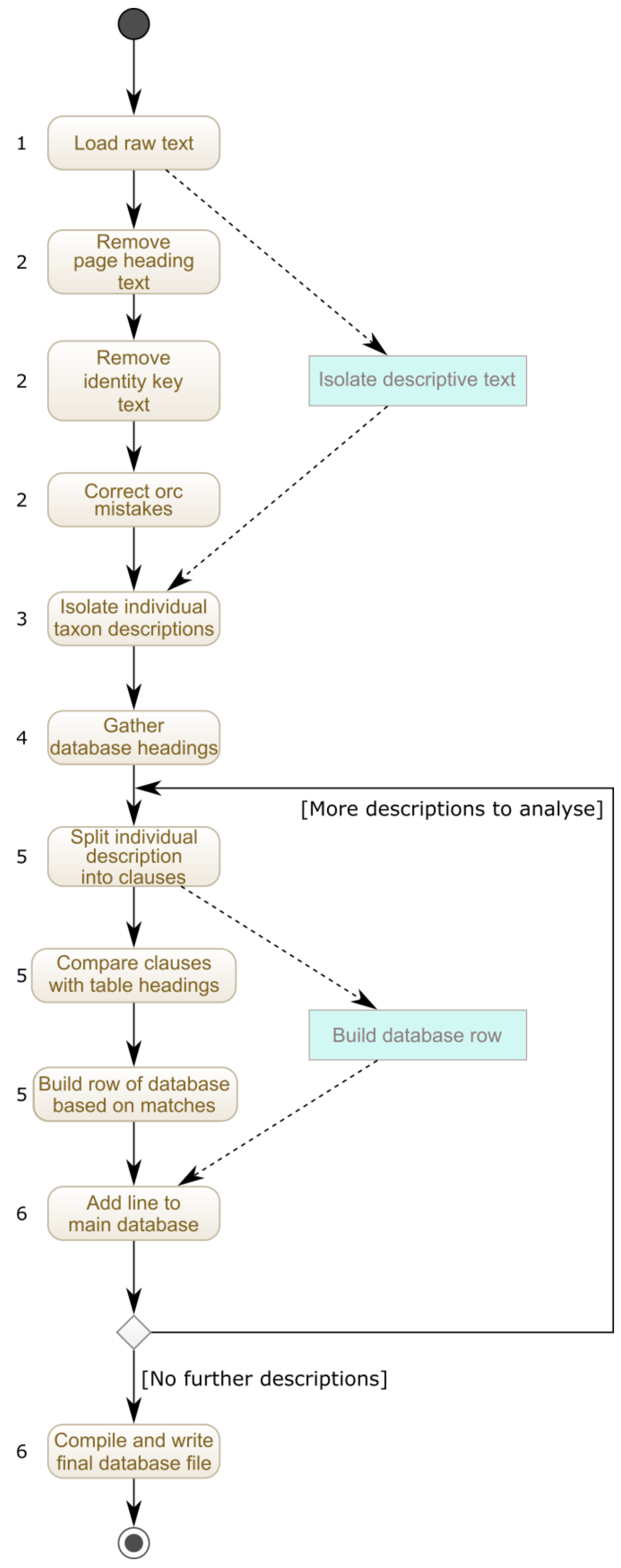
A program was created in Java on InteilJ Idea (JetBrains, 2018) which isolated family, genus, and species descriptions, including removal of 'extra' text such as ID keys. This program then compared the isolated descriptions to the set of pre-prepared column headings, and built up a database based on whether matches were found between the column heading and the description. The class diagram of this program is shown in Figure 3-3, with an activity diagram for the program shown in Figure 3-4. Note in both diagrams, the numbers close to each step or method relate to the numbered points later in the method. This program is available in appendix D.1 on the attached DVD.

This program was run on an HP Elite Desk 800 SFF (CNet, 2018), and a custom Viglin Genie with an i7-7700k and 32Gb RAM. The HP ran Windows 7 (Microsoft, 2009) and the Viglin ran Windows 10 (Microsoft, 2015). Both versions of Windows were provided by the University of Reading. Both operating systems were maintained in an up-to-date state by the University of Reading.



main
rawText: String[]
genusNames: String[]
longStringArray: String[]
noFamilyPageHeadings: String[]
noPageHeadings: String[]
onlyDescriptions: String[]
splitDescriptions: String[]
tableHeadings: String[]
lineToAdd: String
lineToAddAgain: String
main(String[]):1
smallStringRemover():2
familyPageHeaderRemover():2
genusPageHeaderRemover():2
keyRemover():2
startRead():3
readDescription(String):5
readHeadings():4
comparitor():5
lineAdder(String[]):6
writeFile():6
textFinder(String,): String:5
descriptionPartNumbers(String): Integer[]:5
numberArranger(String): Integer[]:5
numberArrangerTwo(String): Integer[]:5
0..1
ocrMistakeFixer
main
0..1
ocrMistakeFixer
rawText: String[]
singleWords: String[]
probableOCRErrors: String[]
replacementWords: String[]
replacedText: String[]
dictionariedOCR: String[]
start(): ArrayList<String>:2
singleWordChecker():2
regexCheck():2
dictonaryAttack(ArrayList<String>): ArrayList<String>:2
dictionaryCorner():2
listMaker(ArrayList<String>, String):2
replacer():2

Figure 3-3 - Class diagram of the automatic text extraction program. Note that numbers after ':' refer to numbered points in the method, where the behaviour is explained. Class diagram made using GenMyModel (GenMyModel, 2019) *and modified on InkScape* (2019).



1
Load raw text
2
Remove page heading text
2
Remove identity key text
Isolate descriptive text
2
Correct orc mistakes
3
Isolate individual taxon descriptions
4
Gather database headings
[More descriptions to analyse]
5
Split individual description into clauses
5
Compare clauses with table headings
Build database row
5
Build row of database based on matches
6
Add line to main database
[No further descriptions]
6
Compile and write final database file

Figure 3-4 - Activity diagram of the data-extraction program. Note that numbers to the left of each step refer to the steps in the method, where the behaviour is expanded on. Activity diagram made using GenMyModel (GenMyModel, 2019) *and modified on InkScape* (2019).

1. The raw text was provided to the program as a UTF8.txt file containing the full text of Stace (2010c) as separate lines (i.e. continuous text separated by 'cr/lf' characters).
2. This stage takes the full OCR text, and removes the unwanted information. This includes text interpretations of images, ID keys, page numbers, and other unrequired data. This involved checking the text against a set of REGEXs.
3. At this stage the program held only a list of strings comprising the descriptive text of each taxa in Stace (2010c). In order to ensure that each taxon description was handled separately, the list of strings were compiled into full separate descriptions for each individual taxon. This is done by examining each string for a handmade set of REGEXs. If the start of a family, genus, or species description was found, it was assumed that all following text was part of the description, until the next start of a description.
4. The program required a .CSV with the column heading features provided only. This stage created a list from the set of heading.
5. In this stage, individual complete descriptions were split to clauses, with each clause then split to individual words. This set of words was then consecutively compared to each heading. If a numerical range was found in the clause, this was automatically expanded so every possible value was compared to the headings. If the word(s) and/or numbers in the heading were contained in the descriptive clause, it was deemed a match. An array marking matches or not matches the same length as the set of headings was built.
6. The database was built up by appending each array of results to the array of headings. After every taxon had been analysed, this array was then converted to a .CSV file.

3.5 Results

The program ran successfully, performing exactly what it was programmed to do. One area where the program was mostly successful was the number of entries produced: Stace (2010c) states that there are 4800 taxa covered in the book; the program created a database with 4806 rows (heading row included). As the program was designed to give each taxon its own row in the database, this meant a near perfect extraction and isolation of each taxon and its description had been achieved.

Running on the HP Elite Desk, with approximately 100 column headings and the full text of Stace (2010c), the program required approximately 3 days to complete and generated 4806 entries.

Running on the Viglen Genie, using on the Gymnosperm text from Stace (2010c), and 3045 headings, the program required approximately 1.5 days to complete and generated 57 entries. In both cases, the processor was never overwhelmed, on the HP the total available ram was quickly filled.

The scanning took several days to complete, and required manually aligning the book each time. The databases produced are available on the attached disc as appendixes E.1 and E.2.

3.6 Discussion

Unfortunately, the process and results described in this chapter highlighted issues that mean this approach to IE from floras is unlikely to ever be completely successful and useful in isolation. The first of these issues is the time required to create the program; it may have been quicker to hand-create the database. The exact behaviour of the extraction and the style of database 'accuracy' were not quite as required for the app.

One of the design considerations of this program was that, providing it does not take several weeks to complete, program run time was not a major consideration. The two reasons for this design decision were that computer hardware is only becoming more powerful, and that processing floras by hand will easily take several weeks per book. Both of these were highlighted during the PhD; the higher power of the Viglen computer increased the speed at which a much higher heading count database was completed; the work required to manually extract data from text and complete a database is shown in chapter 4.

Construction of the program was a different time issue, taking several months of work before it would successfully complete. There were several reasons for this, some of which would be improved with successive versions for different sources, but some of

which are inherent to the method.
Use of OCR was effective, as it would have taken considerably longer to hand copy Stace (2010c). The imperfections, however, caused considerable problems while coding the program. There were two areas where this was highlighted among others; isolating descriptive text from other text (program stage 2), and finding starts to descriptive text (program stage 4). The two big issues from the OCR were mistaken characters (where a character in text was recognised as a different character) and mistaken line breaks (where new lines in the book either were or were not recognised correctly, or new line breaks were input in incorrect positions). As these issues were prevalent throughout the text, the decision was taken to correct them in the program through an expanded set of REGEXs. The number of different OCR errors, however, was larger than expected, meaning a considerable amount of time was required to create these REGEXs. If new attempts are made using this approach, then it is advised that either use of already digitised sources (either ones that have alternatively been released as an e-book or with better scans), or a higher quality approach to scanning the book should be used (this will hopefully create better quality images for the OCR system to work with). If the scan and OCR quality are improved, this stage of program creation should hopefully become much faster, with far fewer REGEXs required.

The general flow of the program took a large amount of time to finalise and implement. With the general design of the program now mostly stable, modification or re-creation of the program for new sources should require less time. With greater experience of programming, the time required to create the program will only decrease.

As suspected, splitting descriptions to individual clauses seems to be a mostly solved problem, assuming these processes stick to floras. Sticking with flora style descriptions means the clauses are reliably split with punctuation. With more natural language texts, this punctuation splitting should still be present, but it is less guaranteed. With the reliable punctuation splitting of clauses, it is a simple set of REGEXs required to split descriptions while avoiding issues (e.g. full stops between clauses vs full stops in species epithets). With the structured, compact nature of flora texts, this does not appear to lose any detail; all descriptors of features are kept with their feature. With longer form botanical literature, such as monographs or expert guides, this splitting to clauses is probably still achievable (Page (1982) keeps roughly to one character to a sentence), but may run into issues if character descriptions run over several sentences. In these cases it may be more effective to work similar to Hong Cui *et al.* (2010) where if a new descriptive element is found, the program looks back to the previous character to determine what is being described.

Current techniques can lose details, for an example see Wood *et al.* (2004). This method does not 'forget' detail; it looks for the words specified, and nothing more. Data can be preserved without ambiguity in this method with extra columns; in order to determine whether a taxon has 'leaves', 'large leaves', or 'large leaves on branches', these three columns need to be included in the database. This does, however, lead to one of the large issues with this approach. In order to avoid accidently losing relevant data, and to extract data across an entire book, a huge number of headings will be required. This has several issues: it risks redundancy of headings; it will require a considerable effort if these headings are manually created; and if these headings are manually created there is a risk of missing a relevant combination of words causing a gap in the data. With long lists of headings, it would then be a manual task to either reduce the list by merging highly similar columns, or selecting relevant ones for the required database.
It may be possible to automate creation of these headings based on the glossary, and some common rules, along with some automatic learning of common phrases in the text. This again runs the risk of generating a very long list of headings, requiring considerable work to reduce the database size. It also is barely removed from the previous IE attempts, negating much of the ease of this rule based approach.

The other large problem with this approach is that it only finds a match if the word/set of words is in the text for each taxon. When working from a single source, this can lead to unusual results. For example, consider using a database with the heading 'cone'. When tested against the gymnosperm text from Stace (2010c), only the descriptions for the Pinaceae, *Cupressus macrocarpa* Hartw. ex Gordon, and *Juniperus* L., had the word 'cone' in them. If the resulting database was used in its raw state, and the user indicated the specimen had cones, this would eliminate most members of the Gymnosperms. Even with an improved database, where members of lower taxonomic ranks inherit the data from parental taxa, this issue remains with the majority of the Cupressaceae apparently not having cones. In this example this is because the Cupressaceae description mentions ‘cone**s**’; with the column heading ‘cone’, this was missed. While this particular example may be easy to solve, if there are at least tens of thousands of headings, spotting mistakes such as this will be considerably trickier.

The other significant issue this method has is that some phrases used in floras are a hard problem for IE. One of the best examples of this with this method is the ‘x longer than y’ style phrase. This style of phrase does not have any absolute information on the length of x, meaning pattern matching IE cannot be completed. All that is known is that ‘x’ is longer than ‘y’; there is no numerical data on the length of ‘y’, and therefore do not have a numerical value for the lower limit of the length of ‘x’. Importantly, this style of phrase seems likely to be difficult for modern IE methods to deal with. The issue is that without looking at the data for ‘y’, there simply is no absolute information on the length of ‘x’. This requires a more advanced system, capable of understanding that the description of ‘y’ needs to be investigated. In this example, investigation of the description of ‘y’ will still only provides a minimum length. Again, this example is easy to spot and solve on its own, but with the possible number and variety of cases throughout a large flora this again becomes a hard problem.

These problems combine to suggest that yes, pattern matching IE in this chapter can work, and in the right circumstance, will. Care must be taken to avoid or mitigate the issues highlighted above. It is, however, only realistic to expect it to work as intended where the amount of text being worked on is small; issues are inevitable, they will only be detected if the database is small enough. It suggested, therefore, that a NLP based system is almost certainly going to be the best approach.

The data extraction stage is clearly the difficult and most complicated stage. The current techniques lose detail, so this method was designed to instead requested all the detail requested by the user if possible.

The output of the approach outlined here is essentially custom designed for the app. This required a plain database, with taxon name(s) in the first column(s), and other columns with the character data, in a .CSV file. This is, luckily, a fairly general purpose design. This should allow the program to be used to create databases for any purpose.

The output from other works, such as the markup based solution from Cui (2010), may be more useful in other situations. All the same data is available, and is in a format that can fairly quickly be converted to the standard database used in this project. The most appropriate output format should therefore be decided on a case-by-case basis. This is an advantage of digitally stored data, as format manipulation is much easier.

3.7 Conclusion

Given the amount of literature available, the high amount of work in NLP and IE is expected. Given that it is a hard problem, it is perhaps understandable that a general solution is not yet available. What is a little surprising is that given the more rigid nature of the language of botanical texts that a solution has not yet been found in this area, but as this chapter demonstrates there are good reasons for this.

Given the volume of literature and data available, it is recommended that more work is undertaken in this area.

4 Databases (part three): Manual datasheet creation

4.1 Introduction

There are a variety of sources of morphological information on plant taxa, including; floras, field guides, specialist guides, herbarium and living specimens. Each source holds morphological data for the same potential set of taxa, but different sources have different goals, coverages, and styles in their structure. For this project, a searchable database needs to be created; data will be extracted from these sources for this purpose.

Data extraction specifically for ID guide databases has previously been undertaken by Drinkwater (2009), who extracted the data from a single source only; Lowrie's Carnivorous Plants of Australia (Lowrie, 1987, 1989, 1998). Drinkwater (2009) found that use of pre-existing literature to create the key worked, but limited the accuracy. For information extraction (IE) purposes, Wood *et al.* (2004) suggest merging data from several sources can greatly improve the quality of the resulting database. Data for previous multi-access keys have previously been stored in the DELTA format (Dallwitz, 1980) where information in stored in character/state pairs for each taxa, and 'lookup lists' are used to store character information. Generally, multi-access key data are stored in species x character databases (Edwards and Morse, 1995); as an SQLite database is required for Android functionality, this is the format that will be used here.

4.1.1 Different sources of morphological data

4.1.1.1 Paper based sources

Different sources may rely on different features to describe different taxa: Stace (2010c) specifically only includes what he considers 'the most important characters', while Page (1982) provides a much fuller description of each taxon. There may also be disagreement on which characteristics are necessary for accurate identification. An example of this is *E. fluviatile* L. Its description in Streeter (2009) includes information about the habitat the species is likely to be found in, but with incomplete morphological detail. The description of the same species in Stace (2010c) still includes some details relating to the habitat, but expands on the range of morphological details when compared with Streeter (2009). The description in Poland and Clement (2009) contains almost no information relating to the actual habitat you would expect to find the species in, and focuses on different aspects of the morphology.

These differences between sources, using *Ranunculus* species descriptions as examples, were studied by Lydon, *et al.* (2003). They found absolute agreement in only 9% of examples, with 36% of the data overlapping in different sources and approximately 55% of the data only available from a single source.

4.1.1.2 Specimens

Herbarium specimens are examples of dried plants or parts of plants, typically held within a herbarium (Beentje, 2010). While most of a specimen's morphology is preserved, colour and small details can frequently be lost, as well as other aspects being affected by the process of preparing the specimen for the herbarium. Live specimens can be found in the field, or as part of living collections. They can be easier to work with, but care must be taken as morphology can vary between wild and cultivated specimens.

4.1.2 On databases and their content

4.1.2.1 Limiting the number of taxa

Not all sources attempt to include all taxa for a given area: for these sources it is usual to give a brief written definition of the coverage (e.g. Poland and Clement (2009) and Streeter (2009)).

Limiting the number of taxa potentially reduces the success of the ID guide. Even if the user is unlikely to encounter the taxa that have been removed, they may yet still encounter them. If the user then tries to identify them using the restricted guide, the guide will fail. With dichotomous keys, this is likely to involve the key being impossible to complete; with multi-access keys, this is likely to result in the key producing a list of 0 possible identifications. Restricting the included taxa however aids the use of the key. With fewer taxa, typically fewer and more distinct morphological features are required to distinguish each taxon individually, as the number of cryptic species will be reduced. This can help the key be accessible to the user less familiar with either the taxa in the key or botanical terminology; the reduced complexity of the key means they will hopefully find it easier to use. The key will be easier to construct successfully. Note that this simplification is not guaranteed, as the reduced list of taxa may still be morphologically very similar.

Inclusion of all known taxa in the covered geographical area ensures that whatever the user encounters, they should theoretically be able to identify it correctly. This assumes that there have been no changes to the species composition of the community since the key was created; novel invasives will not work in the key, local extinctions may result in a false-positive identification. Sources that approach this include Stace (2010c) and Tutin *et al.* (1993), with Sell and Murrell (1997) aiming to achieve this coverage. Inclusion of more taxa has the inverse effect of limiting the number; a greater variety of morphological features will be needed to distinguish morphologically similar taxa, risking a harder-to-use guide.

4.1.2.2 Data inclusion in this project.

As there are benefits to both limiting the number of taxa and including all taxa, attempts to follow and evaluate both approaches will be made. Data included in this project will be extracted from various books with taxa descriptions for the Equisetaceae covering the British Isles.

Protologues are available for the taxa. While they are diagnostic and incomplete descriptions of the taxa, they might contain enough information to distinguish the taxon from morphologically similar ones. This is, however, not a requirement of a protologue. Page (1981) for example explicitly states very little information, instead describing many characteristics in the style of 'intermediate between *E. fluviatile* and *E. palustre*'. Protologues also do not show the morphological range of the taxon within a specific area only. If the data is available, they show the full range of morphology expressed by the taxon globally. If this is the case, there is no way of determining whether individuals outside of the British Isles are generally larger. It is more typical for species descriptions in protologues to rely on a few specimens, thus the description is not likely to show the full range of morphology expressed by the taxon either globally or within a single country.

One of the areas of interest in this investigation is whether or not the data from book based descriptions is enough to create a useful database for the app as this would help indicate whether automatic data extraction from the various book based sources is viable. Another area of interest is whether the descriptions in the various book-based sources are enough to distinguish taxa anyway.
As a result of these points, protologues were not used to generate data, instead they were used to check and amend it if necessary.

As data are being collected from multiple different sources for this project, this will lead to missing data being generated. Some taxa are only described in a few sources, typically no sources cover all aspects of morphology, and different sources do not cover exactly the same set of aspects of morphology. This is unlike in Lydon et al. (2003); they were investigating parallel descriptions of taxa that occur in every source they were investigating. While data for these gaps could be established from available live and herbarium specimens, there is no guarantee you have viewed the complete range of morphology exhibited by the taxon with the available specimens. Extracting data from herbarium specimens is risky due to their degradation rate. In order to mitigate these issues, you need to ensure you have viewed a significant number of the available specimens for each taxon, as well as live in-situ specimens. Pre-existing sources aim to allow identification of all typical morphological variants found in the British Isles. Stace (2010c) states that the extreme values that are given in brackets are not always the most extreme that can be encountered in the British Isles. Hybridisation can cause 'vigour' of the offspring (Shull, 1948), but *Equisetum* hybrids frequently are intermediate to their parents (C.N. Page, 1973 and C.N. Page, McHaffie, & Butler, 2007 serve as examples); for hybrids with missing data, inferences based on parental data may be a viable way to determine the potential range of morphology exhibited in the British Isles.

With the issues that have been highlighted, data included in this project will only come from the literature. Herbarium specimens will be used for testing only. Protologue information will be used to check data where possible.

This merger of data from multiple sources and inferences of gaps also serves as a test case for extended automatic data extraction, similar to Lydon et al. (2003); if this case study is successful, and the resulting database is found to be good quality, then data extraction from books with post extraction merger of the data can still be considered viable, and further investigation should continue.

4.1.2.3 Error induced in database creation

There are different types of human error outlined by Read et al., (2016) that need to be accounted for, as they may affect the work. These include: slips (where an action goes wrong); lapse (where an action is forgotten), rule mistake (where an action is taken based on an incorrect decision), and knowledge mistake (where there is not enough information to make a decision, but an action must occur anyway) (*Human failure types*, 2016). A zero human error rate is possible with automatic data extraction (machine/programmatic errors may still persist). As discussed in chapter 3, automatic data extraction is currently not a working technique.

4.1.2.4 Database design

The basic functionality of this app is based largely on the preliminary work in the MSc project (Bewsey, 2014). The multi-access key search from this previous work (Bewsey, 2014) functioned correctly, so was retained. This meant that the final datasheet had to retain the same structure, with 'yes' or 'no' cell contents, and the morphological data being encoded as the column heading.

4.2 Aim

To investigate means of production of an error-free database based on the available data from book-based sources only.

4.3 Method

4.3.1 Manual database

4.3.1.1 Materials

The database was initially created in Microsoft Excel (Microsoft, 2016) for simplicity of use, with data being drawn from: New Flora of the British Isles (Stace, 2010c); The Vegetative Key to the British Isles (Poland and Clement, 2009); Collins flower guide (Streeter *et al.*, 2009); Flora Europaea (Tutin, Heywood, *et al.*, 1993); The Ferns of Britain and Ireland (Page, 1982). The database was converted to SQL insert statements using the bespoke Java program from section 2.3.1.

4.3.1.2 Procedure

Data extraction from multiple sources were chosen to increase the variety of morphological data available, and to ensure checking induced by repeated morphological data in different floras. This would reduce error rate by relying on the wisdom of crowds' effect. If stem height had been entered incorrectly when working from the first source, for example, all subsequent authors would give a different value, and the cell would be corrected.

Two separate lists of taxa were generated; a full list of every taxon, and a restricted list. The full list was created by inclusion of every taxon included in every book, the restricted list was limited to taxa described in more than one book. Separate databases were created for both these lists.
Each individual character was added in no particular book order. For every new characteristic described, a new column was added to the database. Similar characteristics were kept close together in the database for ease of use later. Numerical data (e.g. height if measured in cm) was input at the resolution used by the source data (e.g. 0 decimal points).

Differences in character data between books were handled dependant on the data type. Range data were adapted to accommodate the maximum range in all books. For example if book A described the diameter as 3-5mm, and book B described the diameter as 4-10mm, then a range of 3-10mm was input into the database. Categorical data where multiple options were described and possible, either due to multiple ways to interpret the character, or multiple phenotypes being viable, were adapted to allow for all options. For example, if book A describes the tooth margin width as wide, and book B describes tooth margin width as medium, then a margin width of medium to wide was input into the database. Categorical data where multiple options were described but unlikely used the description from the majority of books describing the feature for the taxon. For example if books A and B describe the taxon as branching, but book C describes the taxon as not branching, then the taxon was described as branching in the database.
If taxa were entirely missing data for characteristics, these data were approximated using the following method. If the taxon with missing data was a hybrid, these data

were inferred from the parents, following the same steps for adapting different descriptions of the same characteristic for the same taxon from different sources. For example, if parent A had a stem width described variously as 1-3 cm and 2-4 cm, and parent B had stem width described variously as 5-9 cm and 7-11cm, then the hybrids stem width was input as 1-11 cm. This hopefully would account for all possible variation based on parental genetics. This did not account for potential hybrid vigour, but without examination of specimens in the field, this would have been impossible to estimate accurately. This induces the effect that hybrids will be harder to remove with a multi-access key; if you are trying to isolate a specimen using a characteristic with inferred data, the range shown by the hybrid will cover the range of both parents. If the taxon with missing data was a species, every option possible was included, with the intention of limiting the range based on either herbarium data or data from further books later. For example, if stem diameter was not described in any book used, then the full range of possible diameters (i.e. 0-200mm in this database) was input into the database. If more than 3 characters required approximation for a species, then the species was removed until further notice, as it would negatively affect the app ability to correctly remove the taxon from the list of results.

4.3.2 Conversion to app-style database

The design for the search feature of the app was carried forward from the app produced in the preliminary MSc work (Bewsey, 2014). This required the database to be in a different design than was created in section 4.3.1. Each character was separated into discrete headings for each possible variant. For ranged data, this involved creation of a column for each possible value in the range, with columns created at the same resolution available in the source data (e.g. if the source data was accurate to 0.1mm, a new column was created for every possible 0.1mm). For categorical data each variant was separated into its own column. An example conversion is shown in Table 4-1 and Table 4-2, with Table 4-1 showing a sample prior to conversion, and Table 4-2 showing the results of conversion. Note that 'no branches' is coded as 'branches 0', in order to maintain an unambiguous dataset. Not including the 'branch angles 0' column risks the appearance of incomplete data, where *E. hyemale* L. would have the appearance of no information encoded about the branch angles. Once the database had been converted to the required design, it was then converted to SQLite insert statements using the same bespoke java program as used in 2.3.1.

Table 4-1 - Example section of the database after hand creation

Species	Number of branches
E. hyemale L.	No branches
E. sylvaticum L.	4

Table 4-2 - The same example section of the database after conversion to the searchable database. Note that no branches is coded as having branch angles at 0, rather than only having 'n' in all branch angle options.

Species	Branch angles 0	Branch angles 1	Branch angles 2	Branch angles 3	Branch angles 4
E. hyemale L.	y	n	n	n	n
E. sylvaticum L.	n	n	n	n	y

4.3.3 Data testing

Herbarium specimens for each taxon were gathered, and all characteristics which were not affected by the herbarium specimen preparation process or by age of the herbarium specimen were compared against the database. After database construction and completion of data adjustments based on herbarium specimens, experts in the area were asked if they were willing to volunteer to check the database for inaccuracies. The expert was provided with a printed copy of the database, and asked for feedback. After the datasheet was modified based on expert feedback, it was compared against relevant taxa protologues, and the datasheet updated if differences were found.

4.4 Results

The two taxon lists generated by investigation of different sources are shown in Table 4-3. Note that *E. scirpoides* Michx. is missing from both lists; it was only enumerated in Tutin et al. (1993) where not enough information was included for 'safe' inclusion. For species, a 'safe' inclusion requires approximation of less than 3 characters at most. Unsurprisingly, the full list includes more taxa than the short list; some sources deliberately do not cover all taxa that may be present in the area covered, thus some taxa were only included in a single source.

Table 4-3 - Taxa present in each list generated by the investigation

Taxon	**App full list**	**More than once source**
E. hyemale L.	Yes	Yes
E. x moorei Newman	Yes	Yes
E. x trachyodon A. Braun	Yes	Yes
E. ramosissimum Desf.	Yes	Yes
E. x meridionale Milde	Yes	No
E. variegatum Schleich. Ex F. Webber & D. Mohr	Yes	Yes
E. scirpoides Michx.	No	No
E. fluviatile L.	Yes	Yes
E. x mchaffieae C.N. Page	Yes	No
E. x dycei C.N. Page	Yes	No
E. x willmotii C.N. Page	Yes	No
E. x litorale Kühlew. ex Rupr	Yes	Yes
E. arvense L.	Yes	Yes
E. x rothmaleri C.N. Page	Yes	No
E. x robertsii Dines	Yes	No
E. pratense Ehrh.	Yes	Yes
E. x mildeanum Rothm.	Yes	No
E. sylvaticum L.	Yes	Yes
E. x bowmanii C.N. Page	Yes	No
E. palustre L.	Yes	Yes
E. x font-queri Rothm.	Yes	No
E. temateia Ehrh.	Yes	Yes

4.4.1 Data inference

Following the decision not to use herbarium material to generate data, data inference for some characteristics was required for several hybrids, and would have been required for most characteristics of *E. scirpoides* Michx.. Ten out of 21 characters would have required the full range of data for *E. scirpoides* Michx., which meant that it was excluded from the database.

The required data inference for hybrid taxa was not consistent between different taxa. *E. x meridionale* Milde, *E. x mchaffieae* C.N. Page, and *E. x robertsii* Dines were included in Stace (2010c) only, and without morphological descriptions. They therefore required inference of all characteristics. *E. x dycei* C.N. Page, *E. x willmotii* C.N. Page, *E. x rothmaleri* C.N. Page, *E. x mildeanum* Rothm., *E. x bowmanii* C.N. Page, and *E. x font-*

queri Rothm. were all described in at least one source with at least some details included, but did not have every characteristic described, thus requiring estimation of at least one characteristic. The sets of characteristics requiring estimation varied throughout the taxa, with every character requiring estimation for at least one taxon.

4.4.2 Data checking

Using the datasheet produced by this method allowed the correct identification of herbarium and live specimens. An initial examination using the human-readable datasheet format against **RNG** herbarium specimens suggested a few corrections were required. Adjustments to data as a result of this testing are shown in Table 4-4.

As a check on hybrid taxon data approximation quality, the database with Table 4-4 modifications incorporated was then checked against the various protologues for the relevant hybrid taxon. *E. x bowmanii* C.N. Page was the only taxon where characters were described in the protologue with a range that was not present in the database; these characters were the ridge count and the number of teeth. In the datasheet, these were both originally 14 only, data from the protologue updated these to 8-14.

Table 4-4 - Data changed as a result of checking against ***RNG*** *and* ***E*** *herbarium specimens. Changes to data are shown in black prior to change, and red after the change.*

Taxon	**Stems to**	**Ridge count**	**Stem diameter (mm)**	**Cone diameter (mm)**	**Teeth shape**	**Teeth fusion**	**Teeth count**
E. hyemale L.	100	10-30	4-6/8	8-15	small/round to not present	no	10-30
E. x moorei Newman	60/75	1-15	2/4-7	8-16	narrow straight	No/occ yes	10-25
E. variegatum Schleich. Ex F. Webber & D. Mohr	80	4-10	0-3	5-7/10	triangular to short	no	4-10
E. fluviatile L.	150	10-30	2-12	10-20/23	triangular to short/medium	no	10-30
E. x litorale Kühlew. ex Rupr	100	6-20	2-12	5/10-20	triangular to short	no	6-20
E. x mildeanum Rothm.	80	16	1-4	15-40	Fine straight	Yes	6-14
E. x robertsii Dines	200	4-/6-40	1-20	4-80	Triangular and slender	no	6-40
E. x font-queri Rothm.	65/70	8-12	1-20	10-80	tapering	no	4-40

During testing against herbarium specimens, notes were taken for the ‘help text’ for: teeth colour, stem smoothness, cone tip shape, cone measurement, and sheath colour. The teeth for *E. telmateia* Ehrh were occasionally much smaller than expected, and may require use of a hand lens. Stem smoothness was related to feeling like relevant grades of sandpaper in the notes for the ‘help text’. Cones can be damaged, making apiculate tips look like rounded tips. The bases of cones can also be partially obscured, so the ‘help text’ would require a note to ensure measurement of the full cone. Sheath colour is affected by age of the sheath, with younger ones appearing greener, the banding can also be slightly ambiguous and affect the apparent colour of the sheath teeth, so notes in the ‘help text’ were required to ensure users paid close attention to these aspects.

The second round of testing was against an expert in the field, where areas of missing data were expanded on. This included splitting the data for fertile and vegetative stems out from a single stem column, as *E. telmateia* Ehrh, *E. arvense* L., and *E. palustre* L. have differing stem colour and branch rates between fertile and vegetative stems. Other data gathered were further refinements to sheath teeth shape and border information.

Subsequent datasheet testing took place after it had been included in the app. At this point, rapid testing (unstructured testing using the app to check against named herbarium or live specimens) suggested there were no obvious errors in any taxa being tested against. Errors in identification at this stage resulted from poorly worded questions or aspects of design. The datasheet produced is available in appendix E.3 (available on the attached DVD).

4.5 Discussion

This method took several days to produce a complete database for the Equisetaceae of the British Isles, and many times that to check and modify for accuracy. It did, however, eventually produce a database that could be used to identify members of the Equisetaceae.

For this isolated case, much less time was taken than would have been required to create a fully working automatic extraction system, as detailed in chapter 3. If this method, or one similar to it, is used to create a database to the whole of the UK, this time difference may well vanish as inclusion of more data and more sources will increase the time required to complete this method. If a future project requires a larger scale database, at least some time should be spent ensuring automatic data extraction methods are still not successfully working before this approach is undertaken.

4.5.1 Data extraction

Data extraction from the dichotomous keys in the floras was avoided; their structure makes accurate complete data extraction impossible. The first couplet encodes for the entire set of taxa covered by the key, further couplets only cover sub-sets of these taxa. In Figure 4-1, you can say with confidence that taxa 'a' and 'b' share the same form of a characteristic, whereas taxon 'c' shows the alternate form; all three taxa are reached through question 2 where this characteristic is queried. With this same key, there is typically no way to know if taxon d, highlighted in red, presents the characteristic asked about in question 2, and if so which form it presents.

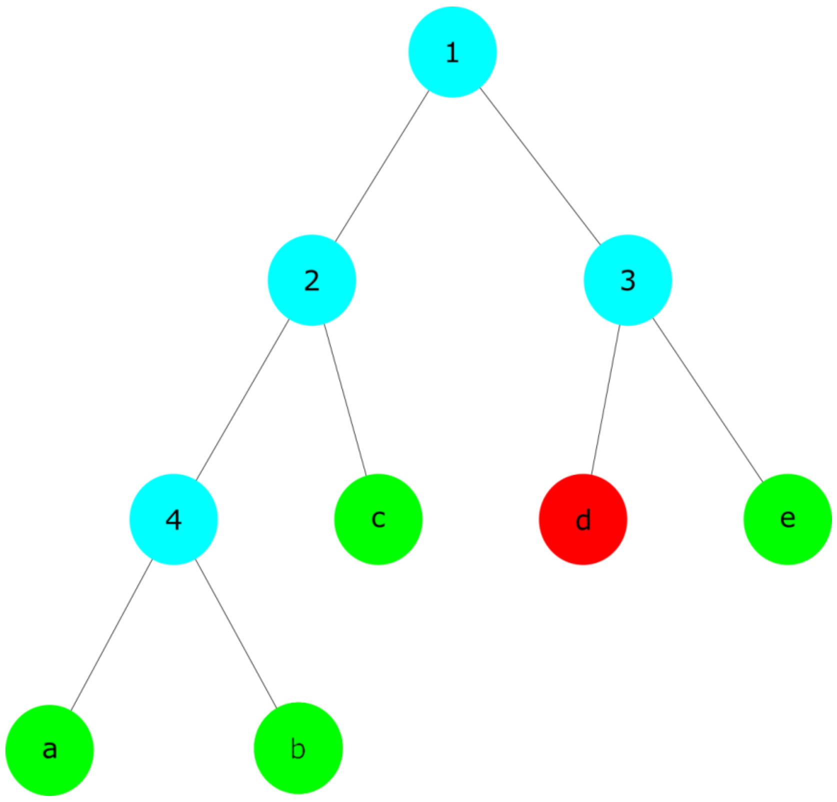


Figure 4-1 - Small example dichotomous key. Blue circles with numbers represent questions, red and green circles with letters represent possible taxa identifiable with the key.

In general, data extraction from books was a relatively simple process. There were no major usability issues with the procedure described in the method. This method was time consuming and training more individuals in the procedure would reduce the time required if data are going to be extracted from more books.

This method was also vastly more successful at extracting the entire set of data in a smaller space than the automatic method attempted in chapter 3 would ever be. As this method has to be done by hand, the meaning of the language used can be interpreted. Issues such as the 'cone' vs 'cones' problem or 'x longer than y' problems highlighted in chapter 3 discussion will be solved as they arise. In the automatic extraction method shown in chapter 3, they either would have required many more columns to account for all linguistic variations present, or would have not worked due to a lack of data.

If books had used the same source (e.g. gathered data for their text from the same set of herbarium specimens, or based their data on a common reference), their data may have shared the same error. As data was gathered from several sources of different forms (two floras, two field guides, and one expert guide), and from sources covering a range of institutions and dates, it was assumed that at least one of the sources would have worked from a different set of original data. Thus all data gathered from the sources was included. While this risks false positives in identification (where taxa are hard to eliminate using the key), this was felt to be acceptable for two reasons. 1: Sources working from different original data are unlikely to have the same error, thus missing data from one source will get completed by another source. 2: False positives (where errors mean ranges are too large) were preferred over flase negatives; false negatives are eliminated from the list incorrectly, and ensuring questions have been correctly answered is unlikely to reinstate them, while false positives can still be eliminated by answering an extra question correctly.

Despite the intention to include *E. scirpoides* due to its presence in the United Kingdom as a garden plant (A. Culham Pers com), it was not described in enough British Isles or European books to provide a full set of data for the database. There are descriptions of *E. scirpoides* in sources from other geographical regions, but these were outside the scope for this key, as geographical differences in morphology may have induced errors. As it is a species, estimation of the missing data from parents was impossible. While there was procedure set for missing data at species level, there were too many gaps for inclusion to be ‘safe’. Inclusion of full ranges for several different aspects of morphology would likely have resulted in it appearing in the results list far more often than it was accurate. It was thus not included in the database.

As not all hybrid taxa had full descriptions in every book, some data had to be inferred from their parents. This approach is detailed in 4.3. This approach proved useful, as it fairly quickly allowed for a completed dataset, and one that should reduce accidental exclusion of hybrid taxa as much as possible with the data available. This approach was chosen despite hybrid vigour potentially meaning some hybrid taxa exceeded the ranges displayed by their parents (e.g. the hybrid growing taller than either parent is likely to), as features of *Equisetum* hybrids generally are described as ‘intermediate between their parents’ (Page, 1981; Dines, 2002; Page *et al.*, 2007).

This estimation approach has drawbacks, however. If all you know is the stem width, and this has had to be inferred from parental data, this risks a false positive identification, as it will be impossible to separate the hybrid from the parents. They will be sharing the same range, thus both parent and hybrid will show up in the list of results. If several commonly used characters are estimated for a hybrid, this approach risks it may showing up with its parents frequently. When combined with other morphological factors, however, the hope was that there would have been few enough categories inferred that this effect would allow the hybrids to have been isolated from their parents successfully. As will be shown in chapter 6, this approach has had different levels of success for different taxa. If taxa with missing data were instead removed from the datasheet, they will not be included as a list of possible identifications even if the user has correctly entered all characters.

In the future the best way to solve this issue will be either investigation of every possible source, in order to gather as much data about each taxon as possible, with the aim of hopefully filling in each missing category, or (ideally both if the resources are available) traditional investigation of herbarium and live specimens of taxa with missing data to fill in the blanks.

Herbarium specimens were also available from **RNG** and **E**, but the decision was taken to use these as initial test subjects to ensure the accuracy of the key instead of incorporating the morphological information into the datasheet.

Herbarium specimens are not like wild specimens; some morphological aspects are damaged by the herbarium preparation process, and occasionally specimens are degraded by time, incomplete, or damaged. The following list covers all separate potential issues with herbarium specimens, and whether this information can be established from the specimen.

- Specimen being fixed to mounting card means ridge count, sheath teeth, and number of ridged on branches was not countable, as half of each value was hidden.
  Full counts for each category for each specimen can be approximated by counting the visible data on half the specimen, and doubling the result. The risk for this approach is that some parts may be partially obscured, leading to a false approximation. This may then artificially inflate the range input into the database for the characteristic, increasing the overlap between taxa and reducing the effectiveness of the key.
- Incomplete gathering of each individual, or damage over time may have resulted in an incomplete stem. This can be seen with either a break in the stem, no roots being present, or a clear break at the top of the stem.
  Without an obviously complete specimen, there is no way to accurately record height.
- Specimens may be simply incomplete. There may be no cone present on the specimen, for example, which may lead to the false definition of an infertile taxon.
- Specimens may be damaged, destroying the datum for the character for that specimen. With these cases there is no way to approximate the datum.
- On investigation it rapidly became apparent that Equisetaceae stems lose their colour and turn brown as the herbarium specimen ages.
- Evergreenness and the ratio of the central hollow diameter to the full stem diameter were impossible to establish from herbarium specimens unless

specifically recorded. Evergreenness requires investigation of a specimen over time; there no reliable way to establish whether a stem dies back at any time of the year from a pressed specimen.
The central hollow to full stem diameter may have been changed during specimen drying, the central hollow is also likely to have been crushed during the specimen preparation process.

Damaged herbarium specimens did however provide useful information on what should be included in the 'help text' for each question. Specimen damage is a problem for all keys but with electronic keys there is the potential to explain what damage may occur close to the question. This will not negate the effect of specimen damage, but will hopefully aid the user into making the correct choice of answer.

Different sources included very different lists of taxa, giving rise to the required creation of two distinct databases. This was successful, as this generated the two separate lists of taxa shown in Table 4-3. The differences between books is highlighted typically by the number of hybrid taxa included, for example with Stace (2010c) showing the greatest number of taxa (other than Tutin et al (1993) which covers all of Europe) Streeter et al (2009) including all the non-hybrid taxa from Stace (2010c), and Page (1982) including most of these hybrids, but not all of them. This is partially due to the fact that Page (1982) was written before some of these taxa were described, such as *E.* x *mchaffieae* in 2007 (Page, McHaffie, & Butler, 2007). It also relates to what taxa are decided to include in different books, with Streeter et al (2009) explicitly stating they have left out taxa that are short term casuals.

One taxon appeared once, and should have been included in the 'full' taxon list: *E. scirpoides* Michx.. This taxon was included in Tutin et al (1993), and has been described as a naturalised garden escape. This should have justified it for inclusion in the full taxon list. Due to the limited data available for it, however, it remained excluded.

The limited taxon list tended to exclude taxa described as having more limited ranges in Page (1982). This may be useful information, or worth determining whether or not this is accurate in the future, as a method of rapidly estimating the geographic coverage of different taxa within an area based on the coverage of the taxa in different books.

4.5.2 App database

During the conversion to the searchable version of the datasheet, the design of the app had to be considered carefully. Some morphological features were only in a sub-set of the species. In the human readable datasheet, all variations of a feature were included in a single column. Taxa that did not display the feature were included with 'not present' being recorded. In the searchable datasheet, the individual features were split into multiple columns, each column only showing a single variant. The cells in the column then contained Boolean data as to whether or not the taxon showed the variant of the feature. This necessitated an extra column being introduced to establish the presence or absence of the feature. This avoided ambiguity between whether the taxon displayed the feature and the variant was not encoded in the datasheet, or whether the taxon did not display the feature at all. This was reflected in the app, where presence or absence of the feature is queried separately from any variation within the feature.

An alternative approach to this problem would be to modify the search function in the app, allowing for the variants or absence of a feature to be included in a single column of the datasheet. This approach was not taken, as this would require ranged data to also be included in a single column. This would have broken the search function, as SQLite only accepts a single value in a cell. Ranged data requires a minimum, a maximum, and a method of determining the intermediate values. SQL can accept secondary datasheets if multiple values are required in a single cell, but as this would have required the same number of columns for each potential value in ranged data as if they had been included in the main datasheet. The decision was therefore taken to continue using the multiple column single datasheet approach, instead of modifying the search function to work with the ranged data.

Conversion of range data to separate columns for each possible value for the searchable database meant that continuous data (e.g. height, where a specimen can be 10 cm tall, 11 cm tall, or any value between, such as 10.6 cm tall) was categorised to pseudo-categorical data/integers (where the database format suggests that specimens can be either 10cm tall, or 11cm tall, but a height of 10.6cm was invalid). This was felt to be acceptable. The categories/integers used matched the resolution provided in the source data, and as such there had been no loss of detail. It was suspected that the users of the app would simply ‘round’ to the nearest whole number from the measurement they took. If the inability to choose the exact measurement was highlighted in testing, putting an indication of this behaviour in the help-text pop up (see chapter 5 for details of these pop-ups) would be investigated. If this issue was affecting accuracy of identification, then a slight fuzzing of the data by expanding the range a single point higher and lower than the correct data should account for all rounding errors.

4.6 Conclusion

Data extraction was successful, if slow. With the linguistic complexity still present in these books, as discussed in chapter 3, this is still the only realistic method to extract this dataset. If full botanical coverage of the British Isles is required (for example), it will be worth training several people on this method to ensure the process can be undertaken in a reasonable time.

5 App implementation (part one): design study of previous apps and best approaches to app design.

This chapter deals with the decisions made on the design and coding of the app prior to the first large scale test and evaluation sessions. The results of these sessions and subsequent changes to the design are outlined in chapter 6. With this stipulation, the layout of this chapter is roughly going to follow the typical experimental chapter layout. Attempts will be made for the style of contents and size of each section to stay close to the traditional expectations of an experimental chapter, although they will be adapted if required.

This chapter will reference specific elements of Android design. In Android programming (both XML and Java) these elements have specific names, and as a result of space (" ") characters being typically reserved in programming languages, these names have no spaces. This results in names having the appearance similar to 'textBox' for a user text input field. In order that references to these design elements are visible and clear through this chapter, this way of naming design elements is maintained where required.

For clarity, the version of the app described in this chapter is the original version prior to any major testing and evaluation. It therefore contains descriptions of design elements that are no longer extant in the app.

5.1 Introduction

Product design is a huge subject area, even when considering only Android app design. For this project, two basic ideas will be considered: keep it simple, and keep it consistent. While these two ideas are not always explicitly stated, they are the underlying theme of many sources and guides on Android app design, and user interface design in general (see, among many others: Babich, 2018b, 2018a; Cooper, Reimann, Cronin, & Noessel, 2014; Creative Blog Staff, 2012; Doris, 2017; Gove, 2016; Natoli, 2014; Sam, 2017; So, 2017; Teo Siang, 2018; Tidwell, 2010; "User Interface Design Basics," 2018) (There are many more design books and blogs relating to Android apps than are referenced here. Collection or further references was limited at this point by the desire to not overwhelm the chapter with endless references that cover minor variations of the same point.).

1. Simple

Users should not notice the interface (Teo Siang, 2018; U.S. Department of Health & Human Services, 2018); they should be able to get on with their task. Users should not only be showed the information they want, but only the information they want, with as little clutter (Babich, 2018a) or unnecessary information (Sam, 2017) as possible. Users should be able to understand what to do the first time they use the app (Natoli, 2014).

For example, if a user needs to enter a phone number, show the number keyboard rather than a text keyboard (Creative Bloq Staff, 2012; Gove, 2016).

2. Consistent

This relates to similar apps, and to the operating system. It is linked to the previous idea; if an app is consistent with others, then the user will be able to use it correctly without having to think.

When compared with similar apps (apps that have the same general purpose, i.e. the identification of plant specimens) familiar screens (Babich, 2018a) and familiar user patterns (Babich, 2018b) should be maintained. This will help the user understand your app if they have previously used similar ones.

The Operating System (OS) will have a specific design guide. Google use 'Material Design' (Google, 2017b), while Apple use 'Human Interface Design' (Apple, 2017). These guidelines should be consulted and understood (So, 2017). There are other general areas of app design that need to be respected, including 'thumb' zones (Creative Bloq Staff, 2012; Ingram, 2016).

Dennis et al (2015) list another set of principles for product design, similar to the ones detailed above, and are distinct enough that they warrant being detailed separately. Their set of principles are split into several categories:

- Navigation design
    - Prevent mistakes
      Design the product so that there are as few choices as possible and that

each choice is clearly labelled to avoid confusion. Do not include commands that cannot or should not be used. Ensure users know when a process is irreversible.

    - Simplify recoveries
      As people are going to make mistakes, wherever possible ensure there is a way to un-do any action.
    - Use consistent grammar order
      Use either ‘action-object’ (e.g. choose delete, then choose which file) or ‘object-action’ order (e.g. choose file, then choose delete option). Be consistent within the app and similar systems, noting that most use ‘object-action’.
- Input design
    - Online vs batch processing
      Online processing is where details about an action are captured directly as the action takes place (e.g. records of a product being sold being taken immediately), while batch processing captures these details in ‘chunks’ (e.g. payroll data may only be processed once per month). Different processes will be more appropriate for different products.
    - Capture data at the source
      Capturing data at the source is generally beneficial to a system; it generally results in less work, less processing time, and importantly fewer chances for errors to ingress.
    - Minimize keystrokes
      For similar reasons to capturing data at the source, ensuring as few keystrokes as possible are required improves the usability of a product and the quality and accuracy of the data being input.
- Output design
    - Understand report usage
      Understand how the report is going to be used, and design it accordingly.

- Manage information load

  In order that a report is useful, it should present only the required and relevant information in a useful order.
- Minimize bias

  The arrangement of data in a report can induce bias, promoting some results over others.

Smartphone apps assist in ID key design all the areas highlighted both earlier in the introduction and by Dennis et al (2015). Availability on a smartphone means the key can easily be available at the source and allow for online rather than batch processing. The set of haptic tools for data-entry allow for as few keystrokes as possible, making the design simpler to use and more constant. This more familiar design should help reduce the number of mistakes users make, as will a high-quality design. Any mistakes the users do make can be amended more easily than with a paper-based dichotomous key due to the nature of multi-access key design, and back button meaning the user can go directly from page to page in the app rather than having to search for a specific page number. All other points can be accommodated with careful design.

5.1.1 Prior work on multi-access keys

There has been some work towards a set of best practises for computer based ID guide design (Dallwitz *et al.*, 2013). Some of these principles are relevant only to computer use, as they assume things that are not necessarily reliably present with smartphones, such as larger screens or internet connectivity. The suggestion that being able to run the guide without having to install it, or allowing a user-definable tool bar seem unlikely to ever be correctly implementable in an app, for example. Android is capable of running apps without installation, but there is a size limit of 10MB for this function, thus is unlikely to be available here (Android Developers, 2018d). Toolbars in Android are simply generally unchangeable by the user. Some of the other points are clearly relevant, and are effectively covering the same ideas as the app design sources (see the suggestion that the interface should be simple for example). There are some other points, such as the suggestion that help understanding characters should be made easily available or that characters should be made available, which are relevant to ID key design in general, and to which attention should definitely be paid.

A number of different identity keys have been tested and evaluated before. These examinations have had a variety of goals, and have employed a range of methods. One thing they all do not report, where it would have been relevant, is the SDLC used in the key's development.

The first study on different forms (single- vs multi-access) of botanical identity keys appears to be Stucky (1984). Here a key of each form was created to seedlings of 40 weed species in North Carolina using pre-existing instructions. 25 volunteers (split into low- and high-knowledge groups) identified between three and six species, using both keys for every species and alternating the order they used the keys. Stucky (1984) found that both types of keys when used by both groups gave an accurate identification rate of 70%. They also found no significant time differences when using either key by either group. This lead to the conclusion by Stucky (1984) that more care should be taken when designing single-access keys, in order to improve their reliability and efficiency. They also conclude that multi-access keys are preferable in at least two situations: less experienced users can input a greater variety of information at any point; and where data sets are rapidly increasing in size, as re-writing the key will not be necessary.

Tardivel and Morse (1996) examined the efficacy, accuracy, and usability of single- and multi-access keys to the same group of taxa. To do this, they created a hyper-text and electronic multi-access key from a paper-based single-access source (in this case Hopkins (1991)). 72 students were randomly assigned one of the three keys, then initially asked to identify a known specimen (*Ligia oceanica* L.) to familiarise themselves with their key. The students were then asked to identify an unknown live woodlouse. Students were asked to note down their identification of the unknown woodlouse (three woodlouse species were included), how long it took them to reach the identification, confidence in their identification, and to fill in a questionnaire relating to previous woodlouse identification experience and experience of using the key. Students managed an average correct identification rate of 74%, with the multi-access key having the highest rate at 79% and hypertext having the lowest. In the paper it is claimed that accuracy across all types of key is 66%, this appears to be a mistake. Tardivel and Morse (1996) found that the multi-access key was slowest to use, and the paper-based version of the single-access key the fastest. User feedback on the paper-based single-access key focused on difficulties navigating the key. Feedback on hyper-text based single-access, and multi-access keys highlighted the lower quality or lack of images increased the difficulty in use of the key.

A dichotomous key and an identification book (where species are described using natural language text and images) were used by Randler and Zehender (2006) to evaluate which design was more beneficial in an educational setting. Participants were shown a slide show of different species, and were asked to identify each species using one of the media. Six reptile species were used in their evaluation. This evaluation was run again after the initial evaluation both immediately after the class and 4 weeks later in order to evaluate retention level. No significant difference in identification ability or retention was found between different media.

Tarkus, Maxl, and Kittl (2010) used either focus groups of interviews with teachers at various education levels to investigate the teachers views on various identification guides including whether they fit into the education framework, and what presentation medium is the most appropriate. They found that dichotomous keys were preferred for teaching, as less information was provided to the students at one time. Multi-access keys were preferred for more advanced students, and when there were fewer images to illustrate characteristics. Keys for younger pupils were found to need more work to ensure that appropriate language and an age specific design were used; keys for younger pupils were otherwise considered an appropriate and interesting tool. A required use of a computer to access the ID guides was not seen as a problem at any level, but smartphone variants were preferred as their portability allowed their use in the field. Tarkus, Maxl, and Kittl (2010) noted time limitations to their study; teachers were presented with tools only once, and were not allowed to examine the tools over an extended period of time.

As part of a project developing a multi-media tool to teach botany, Silva, Pinho, Lopes, Nogueira, and Silveira (2011) produced an interactive dichotomous key to various vascular plants from Ria de Aveiro. The key was used in classes with fresh specimens, and students were encouraged to use the key outside of school hours. In order to evaluate whether the interactive key had a positive effect on student grades, the number of correct species identifications before and after the introduction of the interactive key was analysed. Students were also asked to complete a questionnaire about their opinion of the key after evaluating it. It was found that the interactive key can improve learning plant identification by making it more accessible and appealing to students. It was again suggested that the capability to use the key in the field was a benefit of an electronic key.

Various visual keys (where the key has as few words as possible, instead relying on images to describe the different couplet options) to the *Quercus* L. of the south eastern USA was created by Kirchoff et al. (2011). Different keys relied on either: leaves, buds, bark, or fruit. This was done to investigate the effectiveness of an image-based key to a difficult to identify group of species. The authors who did not participate in the specific keys creation evaluated it by using it to identify either live specimens or images of species within the key. Decisions made at each stage of the key were explained, and this information was fed back into the design of the key to improve it. Kirchoff et al. (2011) found that people new to plant identification tended to focus on the wrong aspect of images, for example focusing on colour of leaf rather than shape. It was also found that the production of image-based keys is becoming easier and is opening further opportunities.

Vollbrecht, Rush, and Cottenie (2013) created a key to Ephemeroptera families. Their aim was to improve the quality of key available in their university module, as students were finding the current key hard to use. Vollbrecht et al.'s (2013) improved key introduced illustrations within the text of the key (as opposed to figures being collected in a separate location), and introduced definitions of more complicated terms. The new key was also restricted to family level identification only, as no deeper identification accuracy was required for the course. The 11 families in the rivers covered by the key were included. Vollbrecht et al. (2013) evaluated their key against 58 student volunteers. Volunteers were randomly assigned the key currently in use in the module, or the new key created by Vollbrecht et al. (2013). Volunteers were asked to key out one mayfly to family level, and record their result. They found that their new key produced an 80% correct identification rate, while the previous key managed only 21%. This greater success rate leads to their suggestion that design of key is much more effective. Vollbrecht et al. (2013) note that the various differences between the two keys under comparison mean there is no way to be confident in which adaptation provided the most benefit, and particularly note that the shorter length of the new key (due to family level identification only) may have been of particular benefit.

The use and accuracy of recognition based guides to tropical rainforest plants by specialists and non-specialists was investigated by Hawthorne, Cable, and Marshall (2014). User accuracy with the guide compared to prior knowledge was explored, along with fitness of various illustration types. Guides consisting of images of different species were prepared for trials. Guides varied in style including various types of image (such as photographs of dried specimens, line drawings, and photographs of fresh specimens) and use of dried leaf specimens. Trials of the various guides involved asking users to identify around 20 different plants to species level (guide styles were randomly allocated to different species). During the trials, participants identifications of the species were recorded by the evaluator. After the trials, participants were asked to complete a questionnaire relating to their opinion of the guides they had evaluated. Hawthorne et al. (2014) suggest that the use of this style of guide may allow typical users to reach between 70-95% accuracy, noting that participants reached this level within the day of trialling guides. Hawthorne et al. (2014) also note that despite the high potential level of accuracy, difficult species remained tricky to identify, and identification accuracy while using the guides on evaluation fell to below 50%. Format of image made little difference to the accuracy of the guide, and Hawthorne et al. (2014) suggest that the most valuable approach is inclusion of the easiest and cheapest to produce images. A high number of false positives noted by Hawthorne et al. (2014) lead to a suggestion that all species in the geographical area covered by a guide are included in said guide.

In order to test the hypothesis that a visual and holistic approach for character representation in an ID guide will allow users to identify specimens more accurately, as part of a project aiming to produce a higher quality key that was useful for both experts and novices, Dellinger-Johnston (2015) created an image based key to 43 *Quercus* L. species. Images for this key were chosen through an electronic questionnaire where participants were presented with two random images of leaves, and were asked to rank their similarity. The results of this questionnaire were used in the construction of the key. This key was compared with the U.S. Forest Service Field Guide to Native Oak

Species of the Eastern North America (2012), a paper based illustrated key to the same set of species as the Dellinger-Johnston (2015) key. Keys were evaluated by providing participants with 10 herbarium specimens of various species within both keys. Participants were asked to use either one of the keys to identify all 10 specimens, or to split key use to five specimens with one key and five specimens with the other key. Participants were either biological sciences university students or *Quercus* L. experts. In all sessions, correct and incorrect identifications were recorded. Participants were given questionnaires relating to their opinion of the guides after using them. Dellinger-Johnston (2015) found that in all sessions (either with student or expert participants) identification accuracy was higher when using the visual key. Both sets of participants also rated the visual key as easier to use. When adjusted for possible causes of variation (different numbers and types of characters and different numbers of species between keys, for example) the higher accuracy of the visual guide persisted. This higher accuracy and user preference towards visual keys leads to Dellinger-Johnston (2015) concluding they are more effective.

Citizen science requirements for ID guides were investigated by Sharma (2016). Citizen sciences involvement of people with different levels of experience, as well as a typical aim of data collection were suggested as the main challenges for such guides. A pre-existing field-guide to 22 bumblebee species was modified into single- and multi-access designs. It was found that the multi-access design had a higher mean accuracy rate (53.7%) that the other keys, with the suggestion that the ability to only input features that were visible on the photograph provided. Sharma (2016) also found that users felt that the multi-access key had a lower workload, despite taking longer to use.

Stagg and Donkin (2017) examined various iOS apps vs printed guides to explore their usability. Guides used were chosen based on their suitability for novices (excluding, as a result, Rose et al. (Rose and O'Reilly, 2006) and Streeter et al. (2009) among others). Plant species chosen for the study had to: be present on Cornwall College campus; not have a sparse population; be identifiable by beginners; and not degrade rapidly when cut. Usability studies were carried where participants were asked to identify a plant group using a single guide (either paper or electronic). Participants were asked to complete a recording sheet for the specimens in their trial, and a user experience questionnaire after the trail was ran. Stagg and Donkin (2017) found that ID guide quality is more important than media. They found that users preferred guides with objective and unambiguous characters, and that diagrams and colours should provide enough detail without becoming simplistic. It was found that the participants' preferred choice of media for guides was highly influenced by the quality of guides they had used during the sessions; if they evaluated high quality electronic guides, this was their preference, and visa-versa. One third of participants noted that smartphone based guides would be more portable, and it is noted by Stagg and Donkin (2017) that smartphones can carry multiple guides simultaneously and that users typically carry their smartphones at all times. Stagg and Donkin (2017) did not note any limitations to their study.

5.1.2 Previous electronic guides

With the design principles highlighted earlier, and the information presented by Dallwitz et al. (2013) in mind, an examination of pre-existing multi-access ID keys is a necessary step to ensuring as good quality app design as possible is reached. The search page of several pre-existing Android-based guides are shown in Figure 5-2 through to Figure 5-13. A prototypical generalised amalgamation is shown in Figure 5-1. Please note that this is not an exhaustive set of examples; the aim of the apps shown is to illustrate the range of designs currently implemented. To include an example from every guide available would require considerable amounts more ink and paper than is required. There are cases where a single producer makes several apps with the same design, LucidMobile (2015a) have several apps in the store for example, so only two have been included here. Where a producer has only produced a single app, but the design is very similar to other apps, it has also not been included.

5.1.2.1 List of questions

All of the examples here show a 'list of questions' design. The most basic of these are shown in Figure 5-2 and Figure 5-3, where all questions and all options for each question are shown directly to the user. This design however risks a very long list being presented to the user if many different characters are needed in the key. Figure 5-6 and Figure 5-8 are an advancement on this, where the options are still visible, but require the user to scroll horizontally within each question to view them all. Figure 5-7 is the same design as Figure 5-6, but with fewer options per question, meaning little if any horizontal scrolling. Figure 5-9 is another similar design, but eliminates the requirement for horizontal scrolling by having multiple lines of answers if reqired. This design reduces the amount of vertical scrolling, however if a question has many answers a great deal of horizontal scrolling may be needed. Each option is also required to fit in a box. Figure 5-4, Figure 5-5, Figure 5-10, and Figure 5-11 show an alternative approach to reducing list length; in each of these examples, the user must actively chose the question they would like to answer. When the user has selected a question, the app then expands the design around the question to show the possible answers. In Figure 5-4 and Figure 5-10, the question expands 'downwards' to show all possible answers. The user does not leave the list of questions page, but the list becomes longer. Figure 5-5 provides a pop-out window for each option for the answers to each question. The user selects their preferred option, presses 'ok' and is returned to the list of questions. Two similar variants of multi-level lists are shown in figures Figure 5-11 and Figure 5-12. Here the user is initially presented with a list of general options (wider categories such as location or leaf). Once the user selects an option, they are presented with a sub-list of options (leaf size or type, for example). The user then selects their answer (leaves simple, for example), and then must navigate back 'up' to the list of general options. Figure 5-11 and Figure 5-12 show the steps required to navigate these lists.

When considering 'simplicity' and 'consistency', there are aspects of the various designs that are preferable over others.

The use of multiple layers of lists (Figure 5-11 and Figure 5-12) feels 'confusing' (i.e. not 'simple'); the user may easily become lost in the list of options, unable to understand the hierarchical arrangement of the key. If click-actions are considered as keystrokes, this design requires a lot of keystrokes by the user to navigate all possible lists. The set of available questions may, as a result, be forgotten by the user, as they cannot easily see the contents of another branch. The user may end up not entering a vital piece of morphology, and the effectiveness of the key is reduced. Thus, the user may make mistakes which may not be trivial to notice or amend. When the mistake is in a sub-menu, the user may not realise it, making recovery from the mistake difficult. Of the two, Figure 5-12 is probably the better design as the options chosen within lower level lists are shown in the general list.

Figure 5-4 and Figure 5-10 are a better design. The user cannot get lost in sub-lists, and there is a mechanism of maintaining a short list even if there are a lot of questions. Using the above substitute for keystrokes, this reduces the number the user is required to make, but does not eliminate them as the user still needs to open menus. There is also a possibility that if a user does not know the full list of options for a question, they will not consider that morphology exhibited on their specimen is relevant to a question, and the question remain un-answered. This allows users to make mistakes by omission, Viewing all the options for a question requires the user to actively select the question; if they do not, they will not be shown the possible answers.

Horizontal scrolling (as shown in Figure 5-6 and Figure 5-7) has several disadvantages. In terms of pure design, it requires each option to fit within the pre-determined box. Help images cannot be made larger, whether or not they would benefit from greater room. It requires treating text as an image, and ensuring that it also fits within the pre-determined box. More importantly, horizontal scrolling is inconsistent with many modern computer programs and apps. Modern websites use responsive web design to ensure that websites look good on all devices (w3schools, 2018). This nearly always results in a website that scrolls vertically, but not horizontally. In chat and email apps, conversations scroll vertically. Smartphone design has responded to this, and modern devices have a tall 18:9 aspect ratio so users will have to scroll less (Bhagat and Bajaj, 2018).

The very basic list of questions shown in Figure 5-2 is the simplest, as everything that is available is shown directly to the user. There is no way for the user to get lost, and there is no way for any possibilities to become accidently hidden or forgotten about. It is simple for the users to navigate, all they need to do is scroll. The largest difficulty with its design is that if many questions are included in the key, and/or if the questions have many possibilities, there is a risk that the list will become very long. This risks providing the user with too much information at once. The user may have to spend a great deal of time scrolling, and may forget what questions are available at different places on the list.

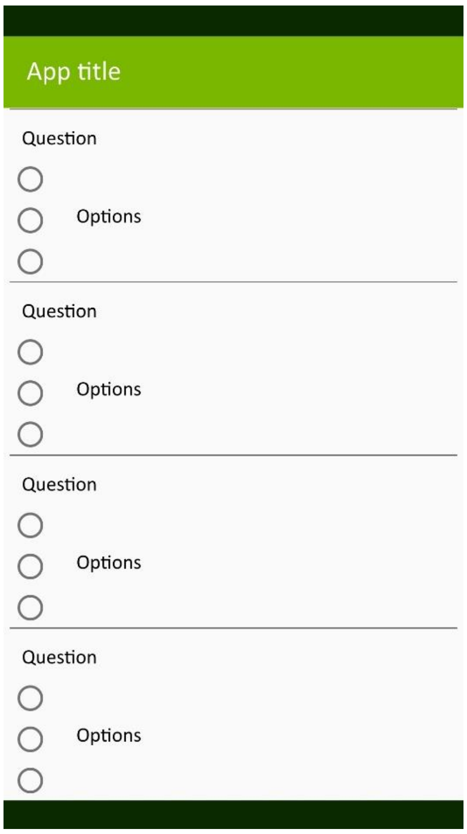


Figure 5-1 - Generalised app design. Note that this is highly generalised and simplified.

5.1.2.2 Figures

Nearly every example includes figures to illustrate and assist user understanding of questions (the only two without such images are Figure 5-2 and Figure 5-5). Nearly all of these help images are line-drawings/diagrams, with some using colour and some remaining black and white (see Figure 5-4, Figure 5-6, Figure 5-7, Figure 5-10, Figure 5-11, and Figure 5-12). The size and complexity of these figures can vary considerably, from relatively small and simple in Figure 5-6, Figure 5-7, and Figure 5-10, to considerably larger and more complex in Figure 5-12. Full photographs are rarely used (see Figure 5-13).

Looking at the practicality of different image forms, there is a clear advantage for line-drawings over photographs. With the relatively small screens of smartphones, the detail is lost in photographs as the image is shrunk. Well-designed line drawings/diagrams are shrinkable further before this problem occurs. With the greater number of apps using line images, it was felt that it would be more consistent to stick with their use. Work by

Hawthorne, Cable, and Marshall (2014) says that the exact nature of images (i.e. photo vs painting vs line drawing) has little effect on the quality of a key, and work by Stag and Donkin (2017), Dellinger-Johnston (2015), Hawthorne et al. (2014), Milward et al. (2017), and Wu et al. (2017) suggests that high quality images are preferable.

5.1.2.3 Search button

There is a surprising variety in forms of 'search' button.

Figure 5-2 (visible in Figure 5-14), Figure 5-6, Figure 5-7, and Figure 5-12 all have a button that takes you from the search page to a new, separate page, with the results. No apps share any common detail for this button.

Figure 5-5, Figure 5-10, and Figure 5-11 show two variations on the 'tab' design. This is most clearly demonstrated in Figure 5-10, where above the list of questions, there is what appears to be a button (currently selected in this figure) marked 'criteria', and a button currently marked '50 results'. Answering an additional question in the 'criteria' tab reduces the number of results reported on this 'results' button. Selecting the 'results' tab functions identically to a search button; the one key difference is that the presence of the 'criteria' tab makes it very simple for the user to switch back to the list of questions. While this behaviour theoretically makes it easier for users to recover from mistakes, switching back to the search page from results should also occur when the user presses the 'back' button (a default button on all Android devices).

Figure 5-4 shows an unusual design, where the user must navigate a navigation draw menu to find the search results page. The 'magic wand' icon in the top right of the app does not show the results, instead showing the best questions left to ask. Note that this design has been updated to the design shown in figures Figure 5-11 and Figure 5-13.

It was felt that the simplest and most consistent design was the search button. Search buttons are very well understood and widely used, being found in nearly every instance where there user will need to search for data.

The 'tab' design is almost as useful as the search button design, but had slight difficulties when compared to search buttons. Given the extremely wide use of search button, and the use of 'tabs' in all modern computer-based internet browsers, it was felt that 'tab' as a search mechanism would be a slightly less consistent design. It was felt that with the back button functionality in Android, and the similarity in behaviour to a search button (i.e. press on it and get a list of results), that in terms of simplicity both 'tabs' and search buttons were felt to be equally simple to use.

Placing the results within a navigation draw was unusual and felt confusing. It was felt to be 'hidden', and required the user discovering the function. It was the only app design where the search function was hidden so far away from the set of options.

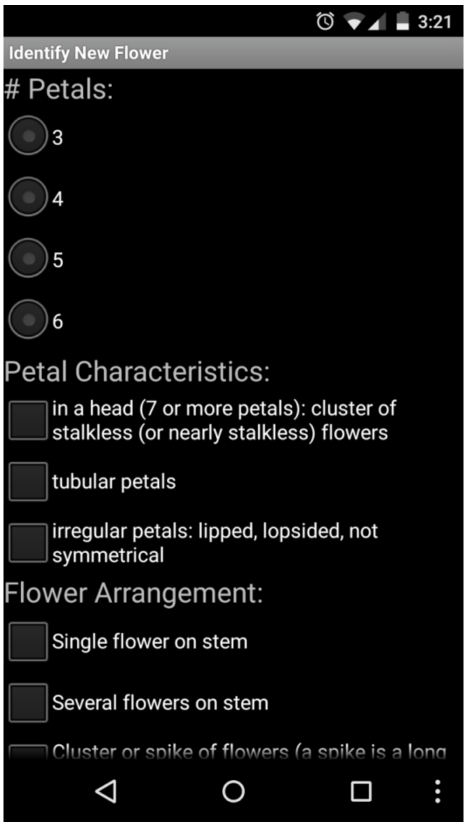
5.1.2.4 ActionBar contents

The actionBar is the bar at the top of each page of an app, designed to display the activity title and other useful items, and assist in application navigation(Android Developers, 2018a). Most apps display one in some form, with only Figure 5-5 and Figure 5-13 missing one entirely. ActionBars are used to either identify the app being used, or the section of the app used. As they are also common among Android apps in general (except when full screen, e.g. videos, is required), their use is both simple and consistent design.

5.1.2.5 App navigation

If these aspects are considered together, how users navigate the app can be considered. This includes the number of actions a user is required to take to get what they want.

Figure 5-13 and Figure 5-14 compare the number of steps required on two different apps to input a single feature, while checking any help images, and search for results. There are two notable differences. In Figure 5-13 the help image could be expanded (thus it was included as a step). In Figure 5-14 the list was long enough, with the search button at the bottom, that this scrolling was included as a step. In Figure 5-14 it was also possible to press on individual entries in the results and obtain more information, which was again included as a step. Figure 5-13 uses the multi-level list design, so entering a second piece of data about a different part of the plant may require four extra steps (two 'up a level' navigations, and two 'down'). In Figure 5-14, entering an extra feature requires navigating the list, only adding a single step.



3:21

Identify New Flower

Petals:

( ) 3

( ) 4

( ) 5

( ) 6

Petal Characteristics:

[ ] in a head (7 or more petals): cluster of stalkless (or nearly stalkless) flowers

[ ] tubular petals

[ ] irregular petals: lipped, lopsided, not symmetrical

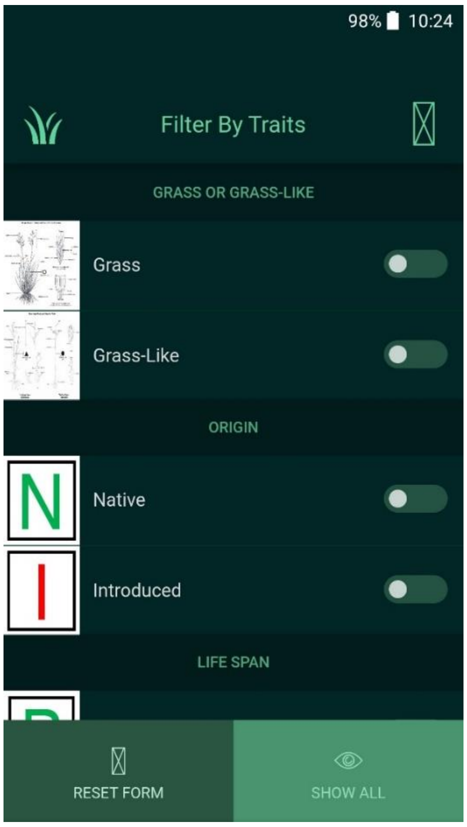
Flower Arrangement:

[ ] Single flower on stem

[ ] Several flowers on stem

[ ] Cluster or spike of flowers (a spike is a long

Figure 5-2 - Wildflower identification by Ryan Haines (2013)



98% 10:24

Filter By Traits

GRASS OR GRASS-LIKE

Grass

Grass-Like

ORIGIN

Native

Introduced

LIFE SPAN

RESET FORM

SHOW ALL

Figure 5-3 - Idaho Grasses by Night Fox Digital (2018)

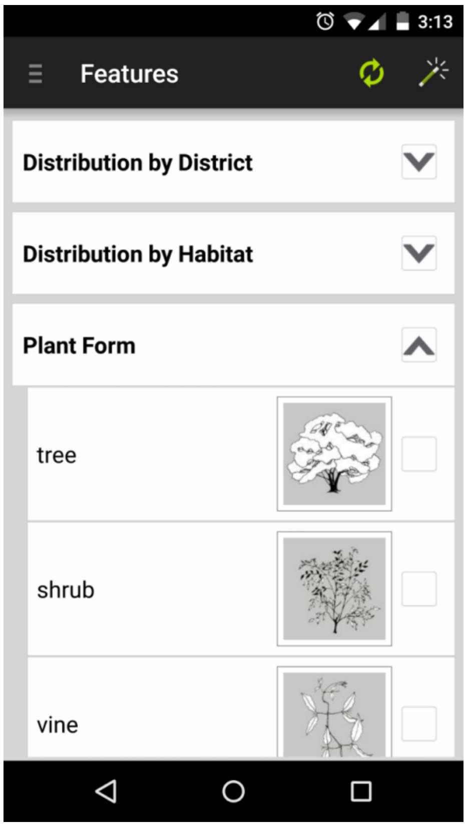


Figure 5-4 - Weeds of South East QLD by LucidMobile (2015b) *(Old LucidMobile app design).*

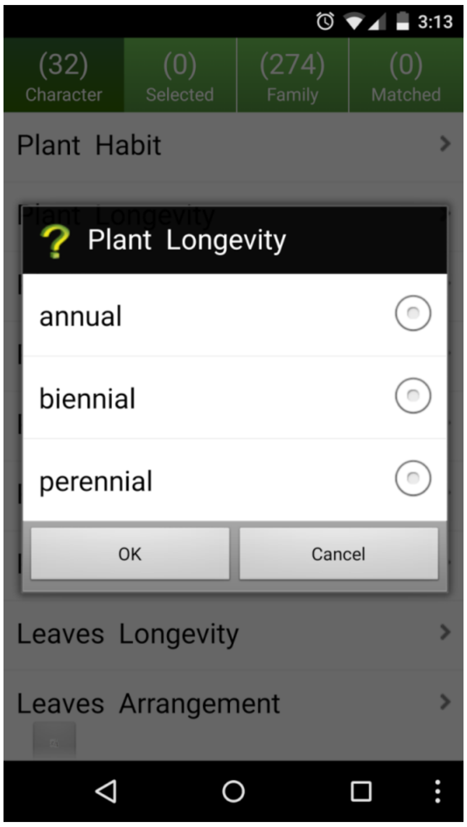


Figure 5-5 - Key: Plant Families by CRinUS (2012)

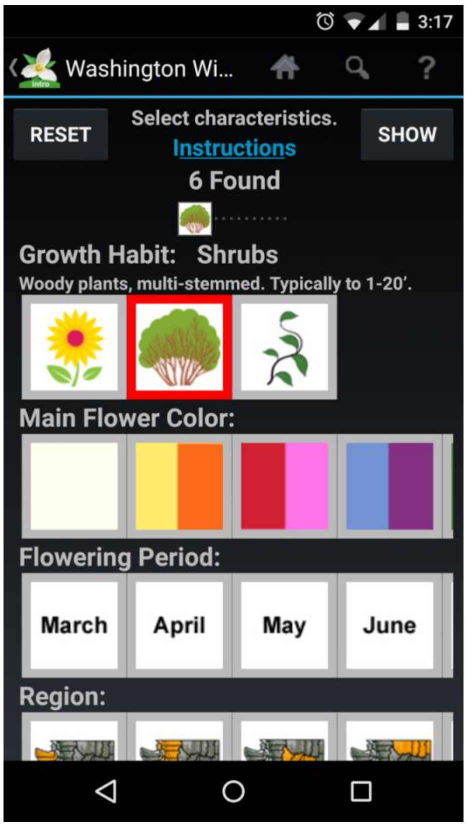


Figure 5-6 - Washington Wildflowers Intro by HighCountry Apps, LLC (2015).

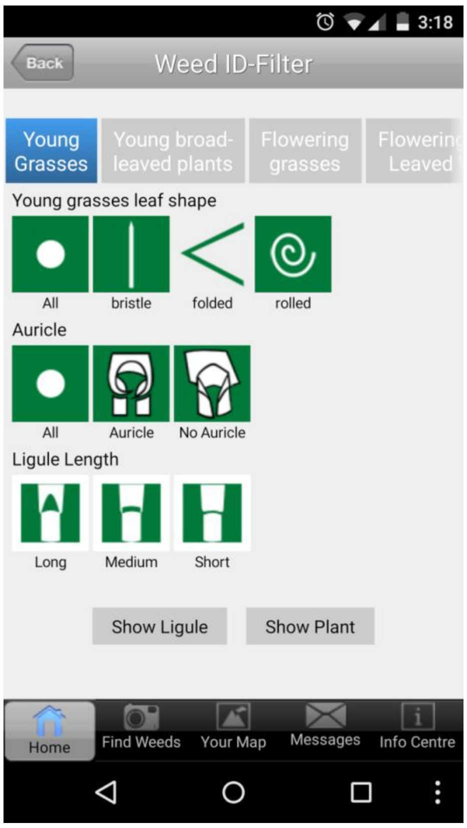


Figure 5-7 - Weed ID by BASF (2013).

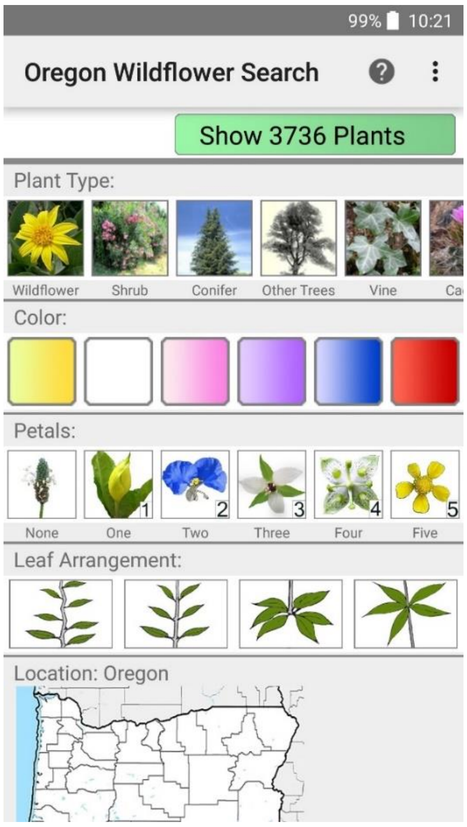


Figure 5-8 - Oregon Wildflower Search by Wildflower Search (2018)

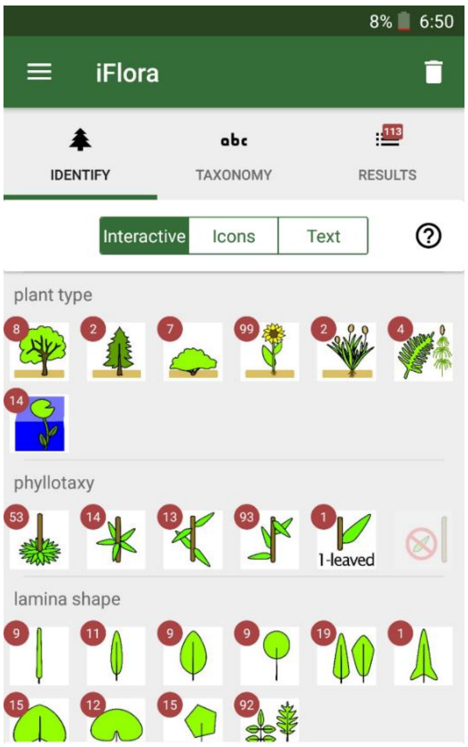


Figure 5-9 – iFlora by Dr Oliver Tackenberg (2019)

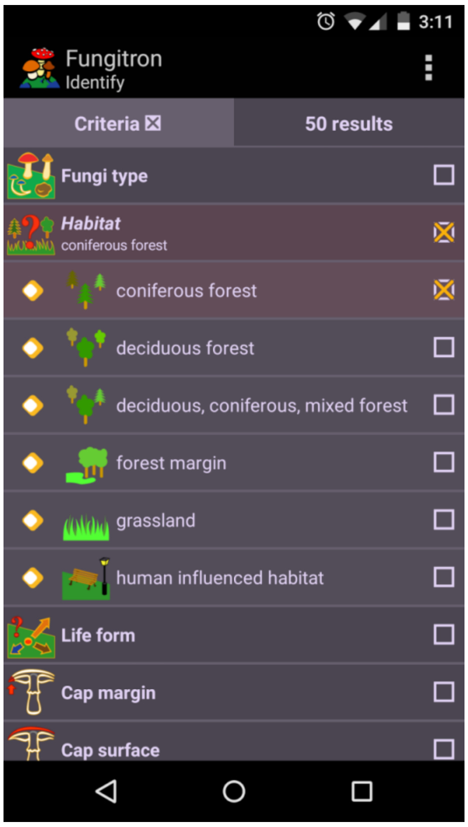


Figure 5-10 - Fungitron - mushroom guide by Apptent Studios (2014).

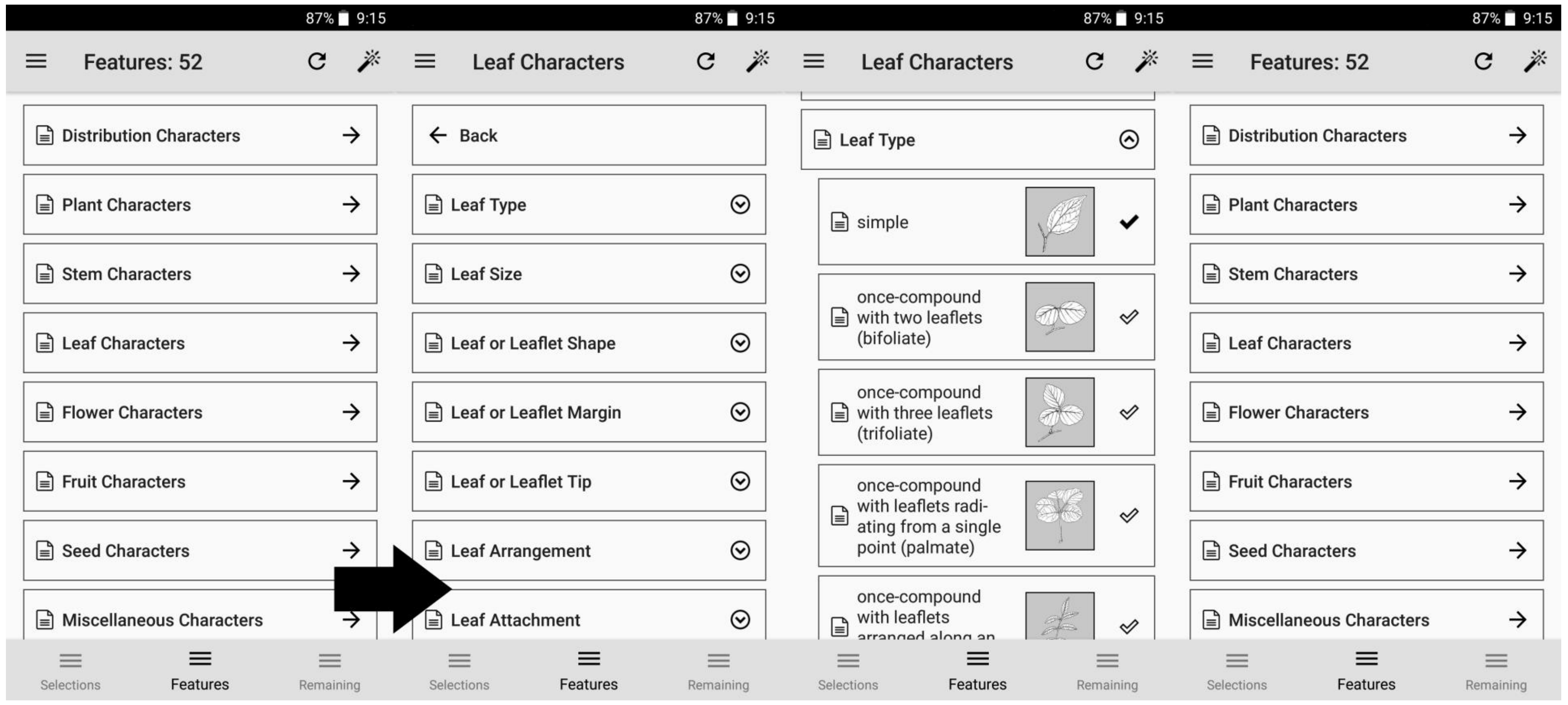


Figure 5-11 - An example of the 'list of checkboxes' design from Weeds of SEQ (LucidMobile, 2015b). *Here the user selects a general area of the plant. They are then shown a list of more specific options relating to the chosen section of plant. Selecting one of these specific options will show the list of possible variants to choose from. The user then can view the number of remaining taxa, or can input more data. Note that the arrow represents the direction of the steps.*

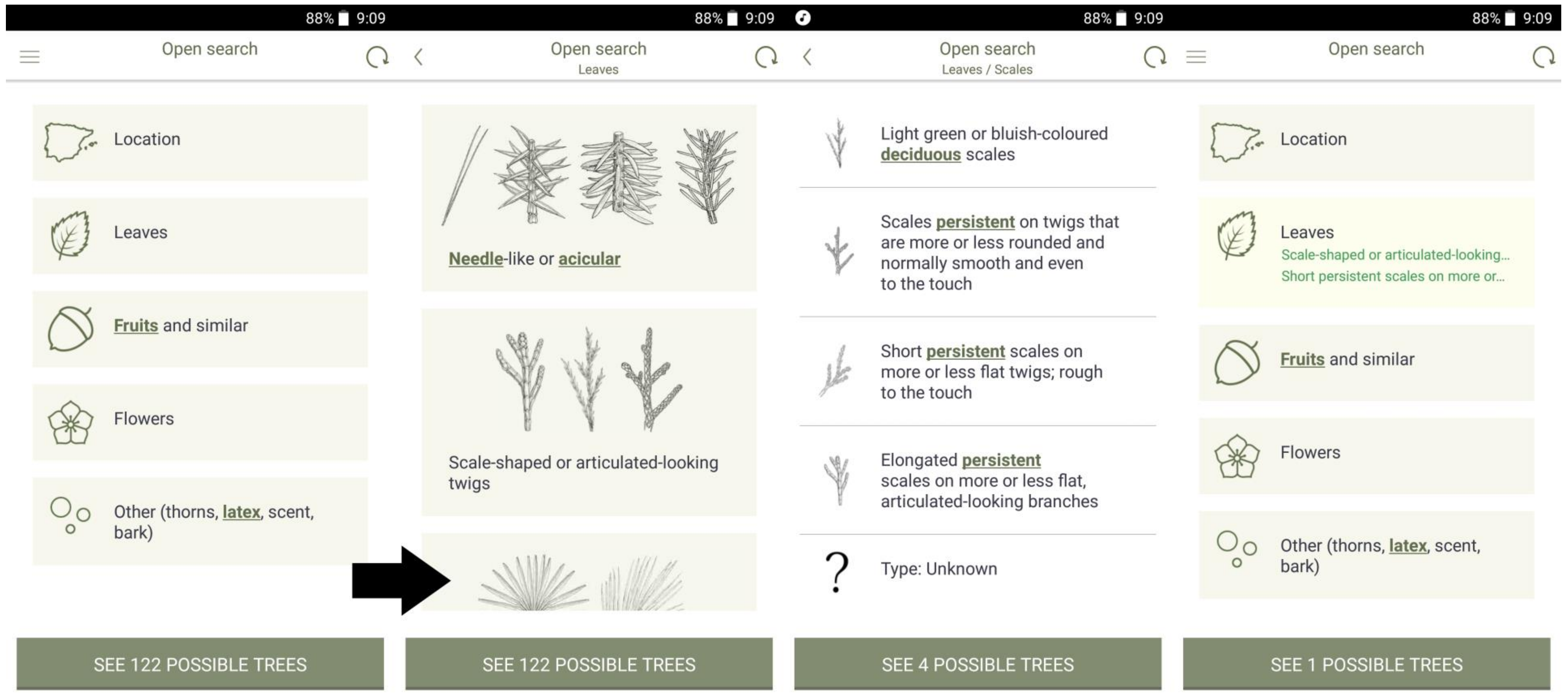

88% 9:09
Open search
Location
Leaves
Fruits and similar
Flowers
Other (thorns, latex, scent, bark)
SEE 122 POSSIBLE TREES
Open search
Leaves
Needle-like or acicular
Scale-shaped or articulated-looking twigs
SEE 122 POSSIBLE TREES
Open search
Leaves / Scales
Light green or bluish-coloured deciduous scales
Scales persistent on twigs that are more or less rounded and normally smooth and even to the touch
Short persistent scales on more or less flat twigs; rough to the touch
Elongated persistent scales on more or less flat, articulated-looking branches
Type: Unknown
SEE 4 POSSIBLE TREES
Open search
Location
Leaves
Scale-shaped or articulated-looking...
Short persistent scales on more or...
Fruits and similar
Flowers
Other (thorns, latex, scent, bark)
SEE 1 POSSIBLE TREES

Figure 5-12 - A slightly different approach to showing a multi-level question structure. The general functionality is the same as shown in Figure 5-11, with the key difference that when the user navigates back to the main list, the options they have chosen are displayed. Note that the arrow represents the direction of the steps.

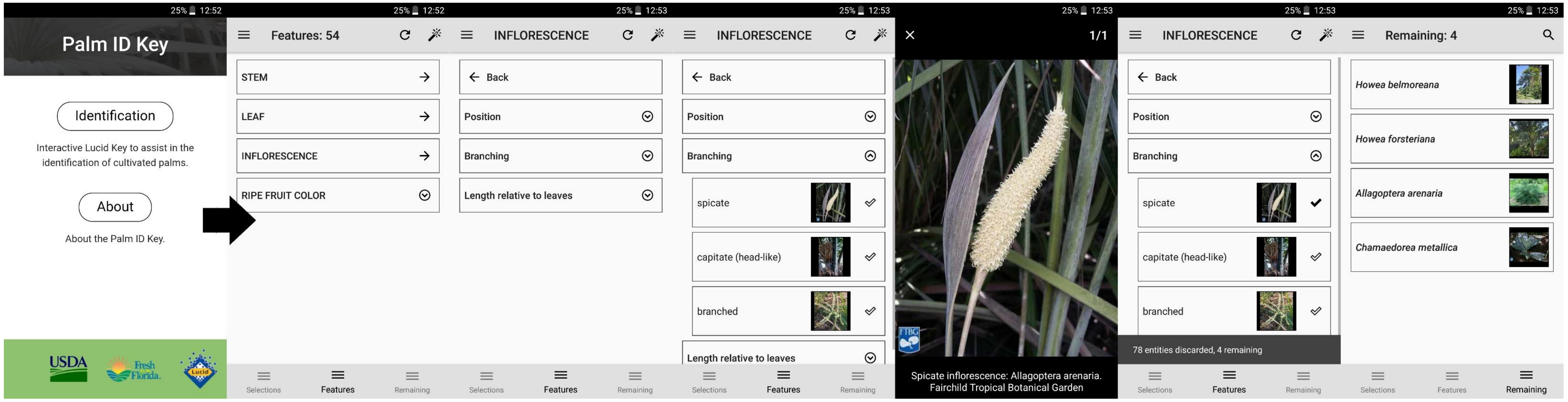
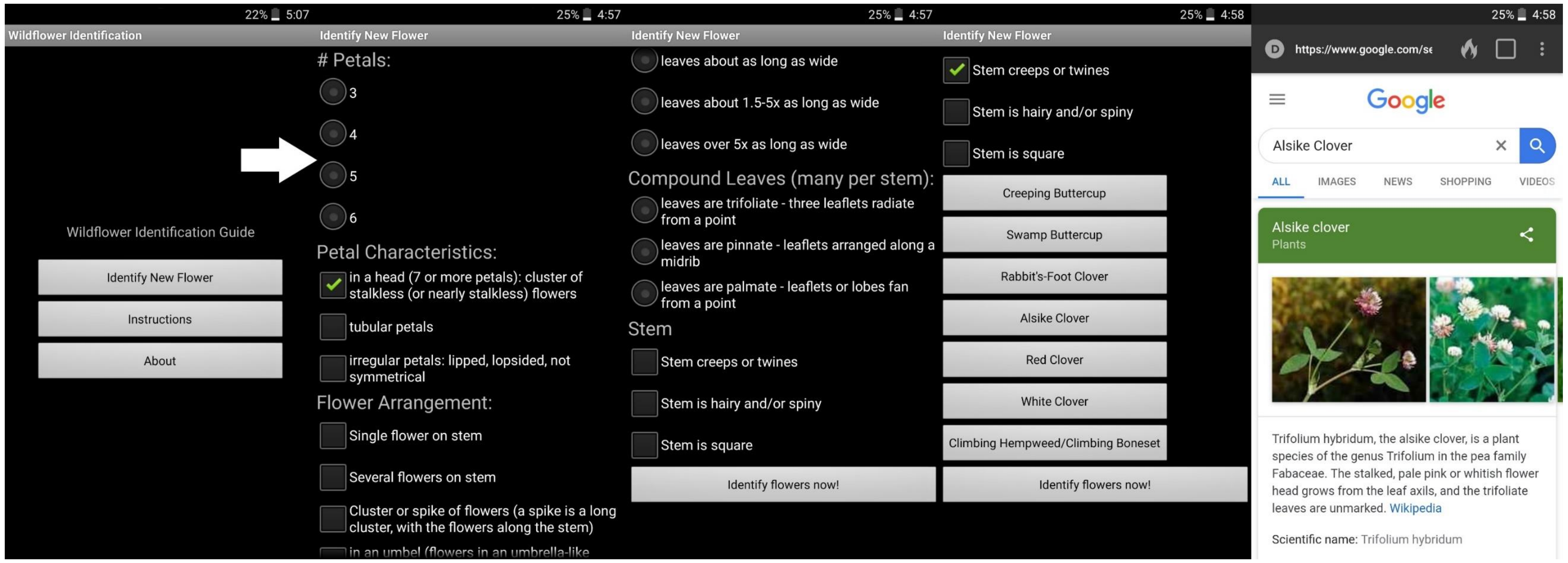


Figure 5-13 - The steps when searching with a single character in the Palm ID key (LucidMobile, 2018b)*. This is the new LucidMobile app design, and shows a single option selected before the search function is used. Note that the arrow represents the direction of the steps, and that the step of expanding the help image can be skipped.*


22% 5:07

Wildflower Identification

Wildflower Identification Guide

Identify New Flower

Instructions

About

25% 4:57

Identify New Flower

Petals:

3

4

5

6

Petal Characteristics:

[✓] in a head (7 or more petals): cluster of stalkless (or nearly stalkless) flowers

tubular petals

irregular petals: lipped, lopsided, not symmetrical

Flower Arrangement:

Single flower on stem

Several flowers on stem

Cluster or spike of flowers (a spike is a long cluster, with the flowers along the stem)

in an umbel (flowers in an umbrella-like

25% 4:57

Identify New Flower

leaves about as long as wide

leaves about 1.5-5x as long as wide

leaves over 5x as long as wide

Compound Leaves (many per stem):

leaves are trifoliate - three leaflets radiate from a point

leaves are pinnate - leaflets arranged along a midrib

leaves are palmate - leaflets or lobes fan from a point

Stem

Stem creeps or twines

Stem is hairy and/or spiny

Stem is square

Identify flowers now!

25% 4:58

Identify New Flower

[✓] Stem creeps or twines

Stem is hairy and/or spiny

Stem is square

Creeping Buttercup

Swamp Buttercup

Rabbit's-Foot Clover

Alsike Clover

Red Clover

White Clover

Climbing Hempweed/Climbing Boneset

Identify flowers now!

25% 4:58

https://www.google.com/se

Google

Alsike Clover

ALL IMAGES NEWS SHOPPING VIDEOS

Alsike clover

Plants

Trifolium hybridum, the alsike clover, is a plant species of the genus Trifolium in the pea family Fabaceae. The stalked, pale pink or whitish flower head grows from the leaf axils, and the trifoliate leaves are unmarked. Wikipedia

Scientific name: Trifolium hybridum

Figure 5-14 - The steps when searching using a single character in wildflower identification (Haines, 2013). *Note that the arrow represents the direction of the steps.*

5.1.3 Users

In order to ensure a product is designed correctly, the users of the product need to be considered. In design, this involves creating a 'persona'. They are a way of providing discrete methods of discussing potential user behaviour, and how the product should function (Cooper *et al.*, 2014). They help design products correctly for specific people, as designs aimed at 'everyone' tend not to be useful for anything specific (Cooper *et al.*, 2014).

5.2 Method

5.2.1 Persona

For this app, the simple persona in use was: the 'enthusiastic non-expert' who has a smartphone and is likely to want to use it in the field. The clearest example of this are students of the MSc plant diversity at the University of Reading – they have chosen to undertake a botanically focused course, but may not necessarily have a great level of botanical knowledge.

This person was thought to be happy to use a 'no frills' and accurate guide that can be used quickly and easily, in slightly difficult conditions. However, this user may not be completely confident in answering the questions, and would appreciate a good quality 'help'. It was suspected that this person may use the guide in the field and in a lab or herbarium setting.

The desire for a 'no frills' and accurate guide is based on the suspicion that person will have used existing paper based guides before. These paper based guides typically ask short, specific questions, and do not frequently provide illustration. This is maintained here. As the user is an amateur, it is suspected they will not know 'everything', thus there may be some questions they cannot answer directly. This is where the high quality help is appreciated by the user. When working in the field, the user is likely to need to use the guide at arm's length, half way down a cliff, or other difficult situations, so speed and ease are requirements.

5.2.2 Use case

The relatively simple overall behaviour of an ID key, where you answer questions about a specimen in order to identify the specimen, along with the 'no frills' guide detailed in 5.2.1, means that the use case diagram for the app at this stage is shown in Figure 5-15.

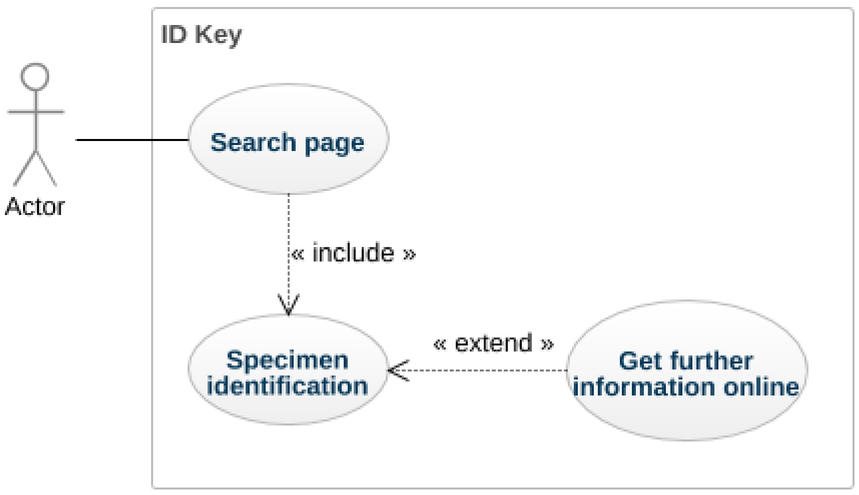


Figure 5-15 - Use case diagram of the app as initially designed. Usecase diagram made in GenMyModel (2019).

5.2.3 Design

At this stage, the app contained data for the four sections of Pteridophytes as shown in Stace (2010c). These different groups are morphologically very different. This would have led to a long list of questions to allow all required morphology to be input, and risked counterintuitive behaviour where some questions deactivated others. Number of stem ridges is a key feature in Equisetaceae identification, but would be a meaningless question for rosette-forming *Isoetes*. In order to maintain a logical and short list of questions, the taxa were split into the four groups as shown in Stace (2010c). This split was shown as four buttons, each button is only a little smaller than a quarter of the whole screen. This meant that at any screen size, the button would be large, allowing a lot of room for images within the button. For the user, it is very quick and easy to press the correct quarter of the screen. At this stage of app implementation and testing, these images had not been created, so simply the name of the section was used.

The plain list of questions design (as seen in Figure 5-2) was chosen as the base of the design. It is the simplest design that ensured that the users were presented with all options for each question and had no risk of any information accidently being hidden or forgotten, helping prevent mistakes through omission. This bought in a requirement that the list of questions and answers remained short, so the user did not ‘get lost’. This fed into the decision to split the data into the four groups. This also fed into the design of the list of questions.

With the design aimed at the ‘enthusiastic amateur’ and the desire to maintain a short list of questions and options influenced several aspects of the design of the list of questions.

The questions were intentionally worded to contain all required information while ensuring their length remained short. Any extra information required by the user to understand the question was put in the help box, along with any images. This simplified and kept the list of questions as plain as possible. Where question answers made other questions incompattable (e.g. if the user indicates that the specimen does not branch, questions about details of the branch are not compattable), the incompattable questions were greyed out and disabled, preventing user mistakes.

The use of a alertDialogues as help-message boxes meant that the length of help and glossarial text was not limited by the design of the list, it could be as long as was required and formatted appropriately. This meant that all possible detail about all possible variants could be included. This was felt to be beneficial over all other designs, including books. Other app designs have minimal help text, if any is available at all. Most ID guide books include a glossary, but space for each entry is typically limited to a couple of lines of text.

Images were used to help illustrate the meaning of question in several places. These help images were also placed in the alertDialogue help-message boxes. This meant their size limit was more relaxed, being limited only by the size of the help box. This greater amount of space meant that images could be larger, and illustrate the desired point more clearly. Putting the help images in the pop-out box is not consistent with previous designs (see Figure 5-4 and Figure 5-6 to Figure 5-13), but it was felt the benefits outweighed the costs of unconformity.

The intention was to use linedrawings/diagrams as help images. They are familiar to users, being in wide use in many paper- and electronic-based guides. Their use was preferred over photographs due to the smartphones small screen size. Photographs reduced in size loose detail as sets of pixels showing part of an image are merged to a single pixel. Line diagrams are easier to reduce in size as not every pixel provides data. Diagrams can be drawn to show exactly the form of morphology required, and do not require a 'perfect' specimen to be found. An initial set of images was gathered from Page (1982). These were deemed good quality and representative of the images that should be used in the key. As these images are subject to copyright, the intention was these images would be placeholders until better ones could be obtained.

SeekBars were used to provide specificity to users when selecting values for ranged data. An example seekBar is shown in Figure 5-16. The requirement for specificity ruled out radioButtons, as they would have required the user to choose between ranges of data. SeekBars allow the user to choose the exact number they desire. The user is provided with a minimum and a maximum at either end of the bar, and has to select how far between the two the value they want. Numberpickers (shown in Figure 5-17) were ruled out for two reasons; their size and their limited usability. They are tall and narrow design elements, increasing the length of the list of questions while not using horizontal space. SeekBars are vertically very small, but can take full advantage of the available horizontal space. NumberPicker behaviour was deemed unintuitive as only 3 numbers were visible at any one time. There is no information about the range of options. They also take a considerable amount of time to scroll through when there are hundreds of options to choose from. In order to prevent the 'endless scrolling' problem a number picker could have been provided for each factor of 10 but this was dismissed as an option as it was thought that users would find this confusing. SeekBars are quick and simple to use, every number is displayed simultaneously, and it is a single movement of the users thumb to select any number. There is one inherent difficulty with seekBars: if there are a lot of different values, they become 'fiddly'. A slightly extreme example that easily describes the issue with too many options is if there are too many options, the risk is that there are not enough pixels to hold each of the options. This would make it impossible for the user to choose exactly the option they want. In reality, the difficulty is that a user's thumb is not precise to a single pixel, meaning allowing several hundred options typically has the same difficulty.

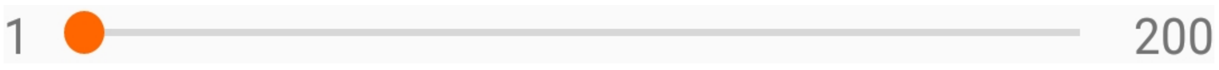


Figure 5-16 - An example seekBar, allowing the user to select any value between 1-200. SeekBar isolated from Figure 5-19

Discrete options were placed into radioGroups. RadioButtons are very widely used, so will be familiar to users. RadioGroups are groupings of radioButtons where the user can only select a single option. While this means users cannot choose multiple options if a specimen is unclear, it importantly restricts the user from inputting contradictory options. Individual radiobuttons are also space efficient, with the default design taking up only a little more room than standard text. Thus little extra space is needed over the list of options.

Text inputs are a similar size to seekBars, and would have been smaller than radioGroups and numberPickers. They are well understood and would have provided a consistent method of data entry for the user. They would, however, have required a way to instruct what form of input was expected from the user, and the range limits of what could be input. While a text input to search for specific taxa may be appropriate, it was felt that its use in the multi-access key would have been confusing to the user. They would also likely be slower, requiring the user to type in all relevant information.

In previous apps there was no consistent search button design or location. The decision was therefore taken to put the search button in a clear and stable place, and to label it with a familiar phrase. It was assumed that it was placed as part of the action bar, then its significance would have been reduced. The search and re-set buttons were therefore placed in their own static bar at the top of the page.

For various details, the decision was taken to follow available material design guidelines and resources. This includes the design of the refresh button and 'help' button found with each question. The colour scheme used for the app was based on the material design suggested palette.

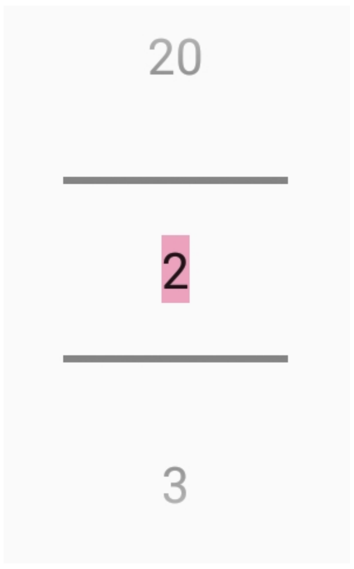


Figure 5-17 - An example numberPicker, allowing the user to select any value between 2 and 20.

5.3 Results

Consideration of the various pre-existing app based guides, and the ideas of simplicity gave rise to the app shown in Figure 5-18, Figure 5-19, and Figure 5-20. Note that all four of the buttons in Figure 5-18 work, and take the user to their search page; the *Equisetum* search page is shown here as a representation these search pages. A simplified storyboard for the app is shown in Figure 5-21, with a complete windows navigation diagram shown in Figure 5-22. Windows that are shown in Figure 5-22 but not shown in Figure 5-21 are all alertDialogs, and aside from the copyright images window, are all not necessary for the minimal functionality of the app (simple identification, without gathering any further information). The copyright images window would not be in the final version of the app, however new non-copyright images will need to be gathered before this window can be removed.

The intended use of the app is as follows:

1. The user opens the app, and they are presented with the four-way choice.
2. When the user makes a choice, they are presented with a search page that is similar to Figure 5-19 but specifically for their chosen section.

a. For any question in the search page, the user can press on a 'help' button, marked with a question mark '?' symbol in a circle.

3. When the user has answered all questions they can with confidence on this search page, they press the search button and are presented with a results list similar to Figure 5-20.
4. Should the user want to know more information about a specific result, they can press on the result, and a pop-up window presents them with the first paragraph of information from Wikipedia.

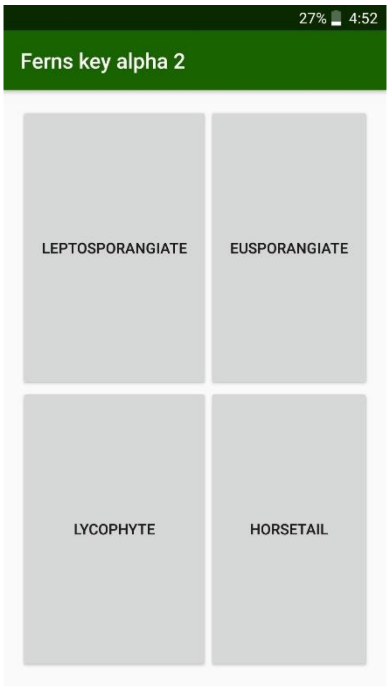


Figure 5-18 - The four way split shown to the users when the app was opened Pressing on any of the four options would take the user to the specific search page for the section they chose.

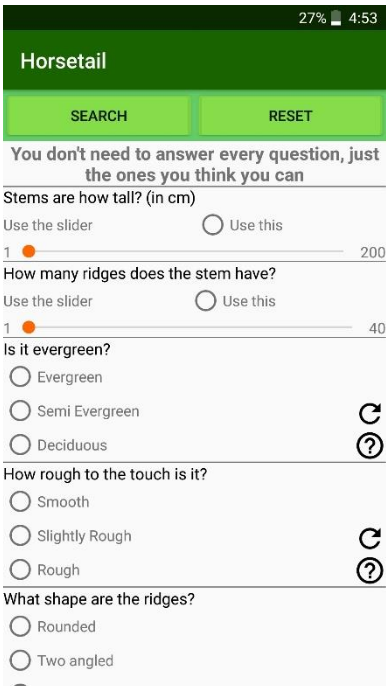
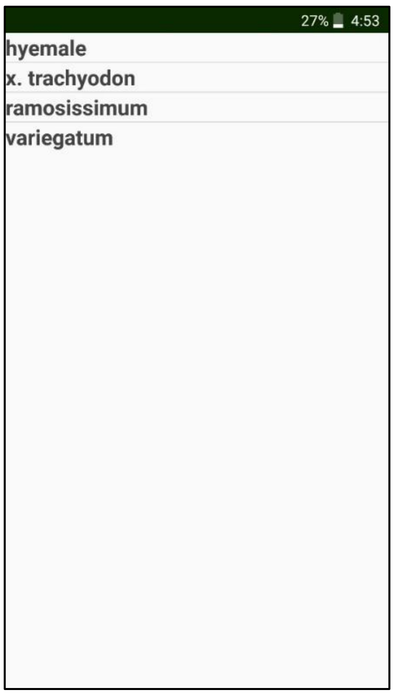

27% 4:53
Horsetail
SEARCH
RESET
You don't need to answer every question, just the ones you think you can
Stems are how tall? (in cm)
Use the slider
Use this
1
200
How many ridges does the stem have?
Use the slider
Use this
1
40
Is it evergreen?
Evergreen
Semi Evergreen
Deciduous
How rough to the touch is it?
Smooth
Slightly Rough
Rough
What shape are the ridges?
Rounded
Two angled

Figure 5-19 - An example search page, in this case the search page for horsetails is chosen. This search page incorporates all the preferred versions of different design elements.


27% 4:53

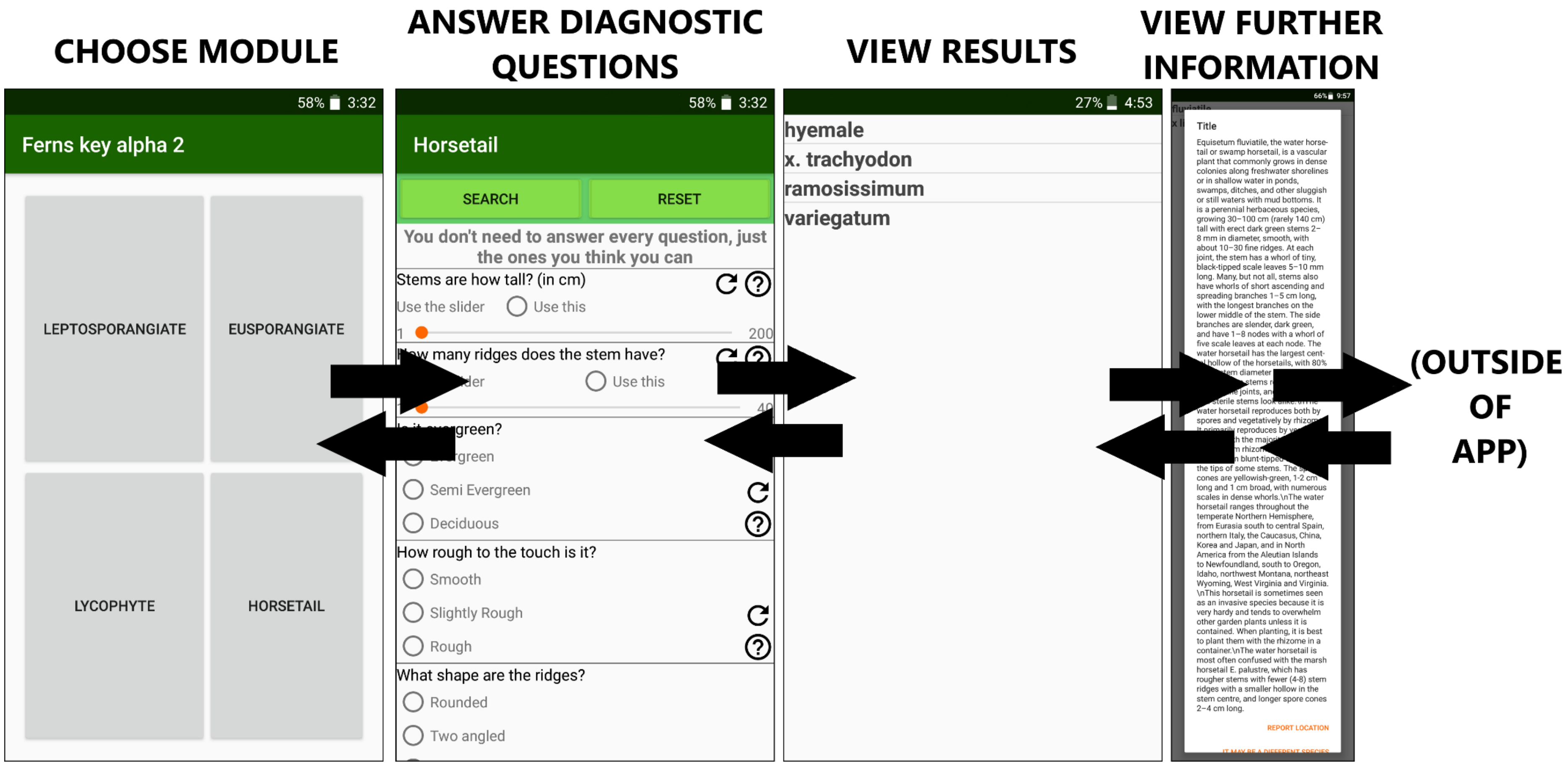
hyemale

x. trachyodon

ramosissimum

variegatum

Figure 5-20 - The results page shown to the users when they have searched in the app.



Equisetum fluviatile, the water horsetail or swamp horsetail, is a vascular plant that commonly grows in dense colonies along freshwater shorelines or in shallow water in ponds, swamps, ditches, and other sluggish or still waters with mud bottoms. It is a perennial herbaceous species, growing 30–100 cm (rarely 140 cm) tall with erect dark green stems 2–8 mm in diameter, smooth, with about 10–30 fine ridges. At each joint, the stem has a whorl of tiny, black-tipped scale leaves 5–10 mm long. Many, but not all, stems also have whorls of short ascending and spreading branches 1–5 cm long, with the longest branches on the lower middle of the stem. The side branches are slender, dark green, and have 1–8 nodes with a whorl of five scale leaves at each node. The water horsetail has the largest central hollow of the horsetails, with 80% [illegible] stem diameter [illegible] stems r[illegible] joints, an[illegible] sterile stems look alike.\nThe water horsetail reproduces both by spores and vegetatively by rhizom[illegible] It primarily reproduces by ve[illegible] h the majori[illegible] m rhizo[illegible] n blunt-tipped [illegible] the tips of some stems. The sp[illegible] cones are yellowish-green, 1-2 cm long and 1 cm broad, with numerous scales in dense whorls.\nThe water horsetail ranges throughout the temperate Northern Hemisphere, from Eurasia south to central Spain, northern Italy, the Caucasus, China, Korea and Japan, and in North America from the Aleutian Islands to Newfoundland, south to Oregon, Idaho, northwest Montana, northeast Wyoming, West Virginia and Virginia. \nThis horsetail is sometimes seen as an invasive species because it is very hardy and tends to overwhelm other garden plants unless it is contained. When planting, it is best to plant them with the rhizome in a container.\nThe water horsetail is most often confused with the marsh horsetail E. palustre, which has rougher stems with fewer (4-8) stem ridges with a smaller hollow in the stem centre, and longer spore cones 2–4 cm long.

Figure 5-21 – Storyboard for the app as designed. Diagram designed as a hybrid windows navigation diagram and windows layout diagram. Note this diagram shows the use of a single search page; the user can select any of the four options in the 'choose module' window, and will be shown a similar set of windows. Note that this diagram is simplified, with pop-out windows not shown. For a complete windows navigation diagram for this module, see Figure 5-22.

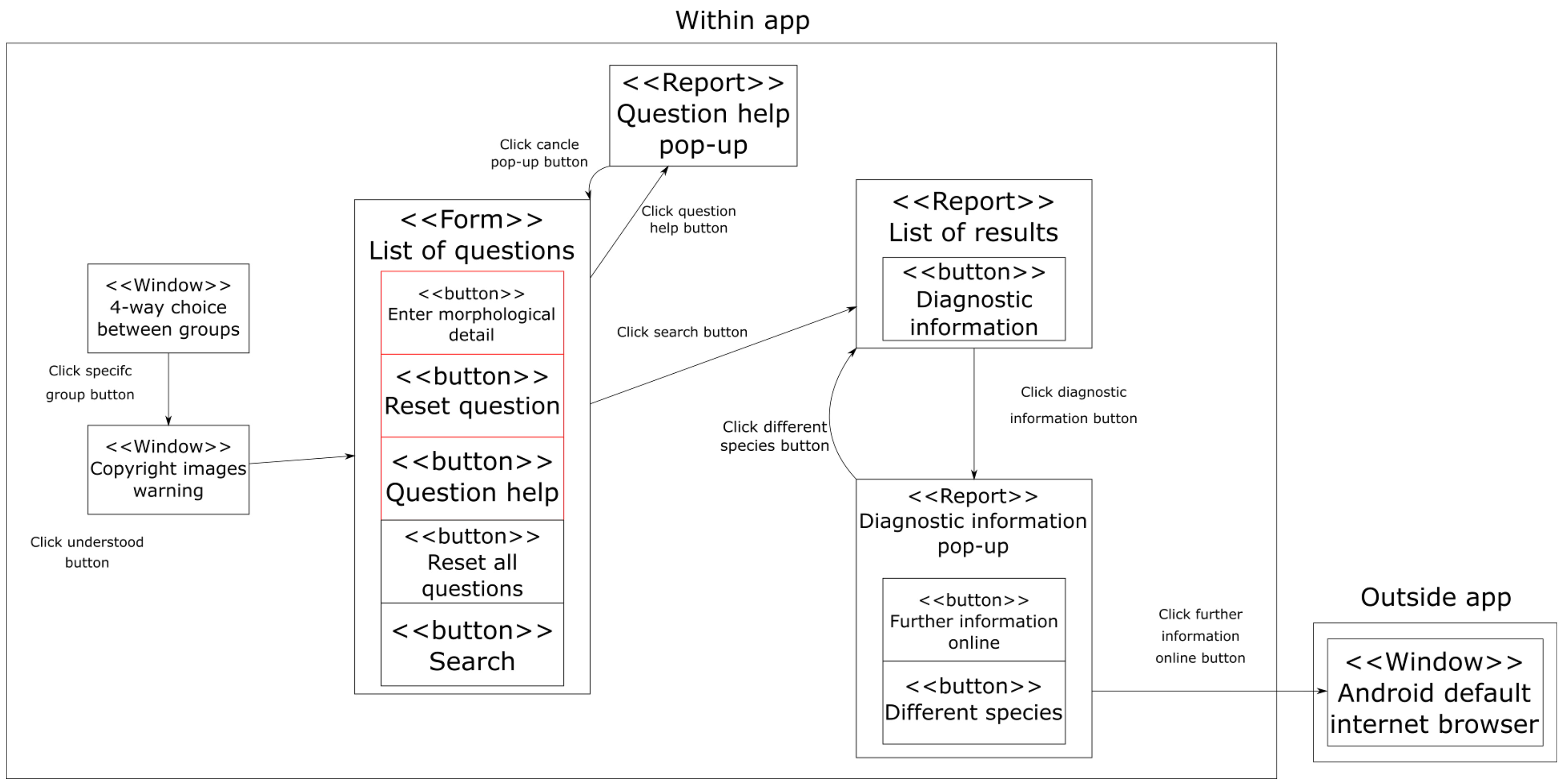

Within app
<<Report>> Question help pop-up
Click cancle pop-up button
Click question help button
<<Form>> List of questions
<<button>> Enter morphological detail
<<button>> Reset question
<<button>> Question help
<<button>> Reset all questions
<<button>> Search
<<Window>> 4-way choice between groups
Click specifc group button
<<Window>> Copyright images warning
Click understood button
Click search button
<<Report>> List of results
<<button>> Diagnostic information
Click diagnostic information button
Click different species button
<<Report>> Diagnostic information pop-up
<<button>> Further information online
<<button>> Different species
Click further information online button
Outside app
<<Window>> Android default internet browser

Figure 5-22 - Windows navigation diagram for a single module of the app as initially designed. Note that the set of windows available to the user would be the same regardless of which option is chosen in the '4-way choice' window. Note that buttons with red outlines are repeated as often as required to show all buttons in the form, they have intentionally only been shown once to ensure the diagram can fit on a single page.

5.4 Discussion

Following the design guidelines detailed in 5.1 and examining pre-existing apps provides a logical approach to designing a multi-access smartphone-based ID key. The user is presented with a simple key, with a design that is consistent with other smartphone based keys. This key is easy to use in the field, thus allowing online and at the time data entry. This key provides the user with all the information they need to answer questions, without excess clutter in the main multi-access key page. If the user requires more information to answer a question correctly (or to decide to ignore the question), extra information is rapidly available. As is an expected benefit with electronic keys, especially smartphone based and multi-access based keys, it is considerably easier for the user to go back and amend their answers to questions, thus allowing easy recovery from mistakes. Testing and evaluation of this design will follow, in chapter 7. This testing will show whether the design constructed here is good quality, and where improvements can be made.

This design allows easy expansion of the key, should new taxa be included. The four-way split shown to the users can be expanded to include more options, or another layer of split can be included. Use of this multi-way split allows the new taxa to be sold in bundles, adding them to the key as separate 'modules'. If more questions are required to differentiate taxa within a module, it is a simple case of adding another question in the design. This is far simpler than single-access keys, where addition of new taxa or a requirement for new questions will generally mean a re-write of the key is required.

Care would need to be taken when deciding what taxa to put into each bundle. Grouping morphologically similar taxa is required to help maintain short lists of questions in the modules. The common use of the morphological species concept would suggest following a pre-existing taxonomy (such as APGIV (Chase *et al.*, 2016)) as an appropriate approach. Care would need to be taken with this approach, as some taxa are not very morphologically consistent (for example, the description of Rosaceae contains the line 'Exceptions to virtually all the above occur' in Stace (2010d), suggesting a very varied morphology within the family), and there is considerable variety in taxa size.

If a single database to the flora of a country is created, it may be possible to arrange the taxa in modules such that the number of splits is as minimal as possible. Given the number and morphological variety of species in the UK flora, however, it is suspected that a considerable number of groups, and therefore splits would still be required. This risks simply creating a large single-access key. Other arrangement of taxa into modules should therefore be investigated, such as arrangement by vice county, to find the most useful. At this point it is unclear what arrangements will need to be considered.

6 App implementation (part two): code design

Note that this is intended as a brief technical document detailing the design of the code, and is not for direct publication.

Note that this chapter details code at all stages in the project, including the initial versions of the app, and the final versions.

6.1 Introduction

Various programming languages are required to produce an Android app. The logic (i.e. how the app works) can be coded in Kotlin, Java, and C++ (Android Developers, 2019a). XML (eXtensible Markup Language) is a markup language designed to store data using 'tags' (w3schools, 2019a). Android uses XML as a method to define the layout of the app (i.e. it says how the app should look) (Android Developers, 2016b). Android supports SQLite (Android Developers, 2016a), and recommends storage of appropriate data in SQL databases (Android Developers, 2019b) For a full introduction into various programming languages see Appendix F . For a brief introduction to SQLite, see chapter 2.

Java and C++ are examples of object oriented (OO) programming languages. Programs made in OO languages are generally made of several 'classes'. A class is a template, or blueprint, of code, containing methods and variables required to make a specific 'object' (for example, for the app produced in this project, a single object could be a list of results, with the programmatic object interacting with the XML based layout code) (Rouse, 2005; Programiz, 2019). These classes interact when a program runs. Class diagrams provided details about what classes are present in a program, details about the classes, and how these classes interact (Dennis *et al.*, 2015). In a class diagram, individual classes are represented by rectangles made of three parts. The name of the class is shown in the top part, the middle part shows the attributes, and the bottom part the operations (methods) (Dennis *et al.*, 2015). Visibility of attributes and operations is shown by a plus (+) for public (not hidden), hash (#) for protected (only visible for the class and its subclasses), tilde (~) for package (only visible to a group of related classes, not necessarily the entire program), and a minus (-) for private (only visible to the class)

(Dennis *et al.*, 2015; Lucidchart, 2019). Relationships between classes are depicted as lines between different classes, the line is labelled with either the name of the relationship or the role (roles are indicated by a plus (+) symbol on the label) the classes play in the relationship (Dennis *et al.*, 2015). The number of times a class can be associated with another class is shown on this association line with numbers separated by two dots (e.g. 1..2) on the class being associated with (Dennis *et al.*, 2015). In Figure 6-1,both classes can be associated with between zero and one of each other. Classes can be generalisations of others, such that one superclass provides properties for several subclasses. These associations can either be aggregations (where the subclass is part of one to potentially many superclasses) or compositions (where the subclass is associated with one and only one superclass) (Dennis *et al.*, 2015).

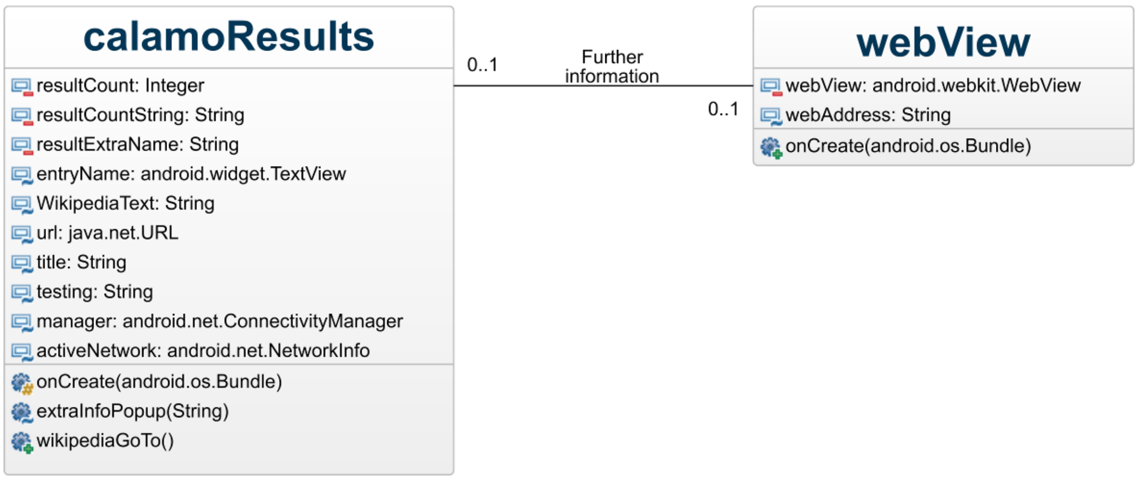

calamoResults
resultCount: Integer
resultCountString: String
resultExtraName: String
entryName: android.widget.TextView
WikipediaText: String
url: java.net.URL
title: String
testing: String
manager: android.net.ConnectivityManager
activeNetwork: android.net.NetworkInfo
onCreate(android.os.Bundle)
extraInfoPopup(String)
wikipediaGoTo()
0..1
Further information
0..1
webView
webView: android.webkit.WebView
webAddress: String
onCreate(android.os.Bundle)

Figure 6-1 - Example class diagram, using two classes taken from Figure 6-4. Diagram made using GenMyModel (2019).

In order to produce readable code, companies and people generally have 'best practices' for code design (Nikishaev, 2017; Google, 2019b). These generally aim to ensure developers produce clear, readable, and concise code. Ideas such as ensuring code is readable to others are often included.

6.2 Aim

Produce logical, clear, and short code for the app produced in this project.

6.3 Materials methods, and results

6.3.1 Materials used

The app was developed for Android devices only. The app code was written in Android Studio (Google and JetBrains, 2018), using all available versions between January 2015 and June 2019, and the program maintained in an up-to-date state on the 'stable' channel. Various computers were used in development, all running either Windows 7 (Microsoft, 2009) or Windows 10 (Microsoft, 2015) depending on the computer in use. The majority of the development took place on an HP Elite Desk 800 SFF (CNet, 2018) running Windows 7 (Microsoft, 2009) and a custom Viglen Genie with an i7-7700k and 32Gb RAM running Windows 10 (Microsoft, 2015). Both desktops and their operating systems were maintained and updated by the University of Reading IT department.

The primary Android devices used during development were a LG Nexus 5 (GSMArena, 2015a), a OnePlus 3t (GSMArena, 2018b), and a LG Gpad 8.3 (withdrawn from use half way through development) (GSMArena, 2016). These devices were the only ones available and within the budget during development. These devices were maintained in an up-to-date state, with updates as provided by the relevant companies (e.g. Google for the Nexus 5, or OnePlus for the 3T). This meant the app was developed on all versions of Android from Android 4.2 JellyBean to Android 8.0 Oreo, with exact Android versions used at any point dependant on the upgrade cycle of the devices used.

6.3.2 Practices used

The following practices were used where possible:

- Long, descriptive, and consistent variable names, always starting with the first approximately six letters of the module they contain
  - The exemplar module here works only on horsetails, so where necessary the names begin with 'calamo', as they are in the order Calamophyta. If a module contained all taxa in the Asteraceae found in the south west region, 'swAster' would be appropriate.
  - E.g. if a radioButton that provides the answer that 'the sheath teeth are long', then an appropriate ID would look like "calamoSheathTeethLongRadioButton".

    - E.g. a class holding a set of SQL queries would have a name like "calamoSQLqueries".
- Comments on code should aim to make things clear as quickly as possible
    - Comments on every line of code in a method can get in the way of seeing how the method works, so if possible, explain the method not the lines of code.
    - If a method is repeated with only mild changes (e.g. an alert dialog that has one button in the first version and two buttons in the second), it only needs to be explained in the first repeat.
- General code layout should be consistent
    - E.G. always as below, rather than having the else clause begin on a separate line

      ```
      if {
      } else {
      }
      ```
- Avoid, if possible, code becoming difficult to maintain. This can either happen through code becoming unnecessarily long and/or repetitive, or otherwise poorly written.

6.3.3 Logic code

The code was developed from the app produced during the MSc. (Bewsey, 2014). This app worked without any major fault, so the decision was taken to build and improve from it. This code was initially based on the tutorial on building an 'Address book' (Banas, 2013a to 2013b).

The search function was similarly based on the 'Address book' tutorial (Banas, 2013a to 2013b) and on the basic SQL search query shown in chapter 2. The program was designed to start with a query that would return all entries in the database; in this case 'SELECT * FROM {database name} where 1=1'. Individual answers input by the user would then appended clauses to query. For example, selecting the answer 'sheaths short', would append 'AND sheathsShort = 'y'' to the basic query. The database design for this app is shown in chapter 4; each column represents a possible variation of a feature the taxa in the database show. Building up the search query in this approach therefore selected only the taxa that displayed the specific feature variants selected by the user. The ‘where 1=1’ clause was included in the default query for this app as it is an ‘always true’ statement, and allowed the additional clauses to be appended in any order; ‘SELECT * FROM {database name} AND sheathsShort = ‘y’’ is not a valid SQL query. This freedom to append clauses as required allowed the users to answer questions in any order.

Each question answer (i.e. each radioButton or seekBar) was provided with its own distinct method. Each method followed the same general layout. The general pattern for methods relating to radioButtons is shown with the method for ‘only one stem type’ in Code box 6-1. These methods check that the radioButton has been turned on, as a confirmatory step. If it has been turned on, it appends the correct clause to the searchQuery and removes all other options from radioButtons in the same radioGroup. The questionText associated with the answer pressed is then set to bold, to signify to the user that the question has been answered. Finally, the method numberOfResultsUpdate is called, so the user is updated about the number of possible identifications.

```
public void calamoOneStemType(View view) {
    boolean on = ((RadioButton) view).isChecked();
    if (on) {
        searchQuery += " AND one_stem_type = 'y'";
        searchQuery = searchQuery.replace(" AND 
two_stem_types = 'y'", "");
        stemTypeCountQuestion.setTypeface(null, 
Typeface.BOLD);
    }
    numberOfResultsUpdate();
}
```

Code box 6-1 - Exemplar method for a radioButton answer

A few different methods were required for seekBars. Standard onClickListeners were implemented in onCreate, and were set to update the number of results continuously as the user updated the answer. They were also set to programmatically update the value being shown to the user as the current value the seekBar is set to (see Figure 7-10 for an example of this behaviour). The seekBars were not set to permanently update the searchQuery on use (unlike radioButton behaviour), as this induced unplanned behaviour when the user used a seekBar twice (for example when correcting an answer). When the search button was pressed, whether a seekBar had been used was checked, and a new clause was appended to the searchQuery for each one in use. In order to append the correct clause, the position of the seekBar immediately at the time the searchButton was pressed was queried, and this data used to build up the clause.

Where seekBars were included as the primary option for answering a question, an option was included in the actionBar menu to toggle the question between seekBar layout and radioButton layout.

Reset buttons reset all questions to an un-answered state. For radioButton-based questions, this removed all possible clauses from the search query, reset all relevant radioButtons to off, and reset the question text to non-bold text. For seekBar based questions, the seekBar was reset to zero, it was set to ‘not used’, the textView displaying the current position of the seekBar was set to display ‘No value chosen yet’, and the question text was set to non-bold font.

Help buttons were set to create an AlertDialog pop-out box displaying glossarial help relating to the question. The first step of this was to check if the Android version of the device the app was running on was running. If the version of Android was more recent than Lollipop, then the AlertDialog was set to display a custom XML layout; lower than Lollipop, a simple text string was used as the body of the AlertDialog. A single button, acknowledging the help, was set to dismiss the pop-up. An exemplar of this code is shown in Code box 6-2, showing the code for help button for the shape of the ridges on the main stem.

```
public void calamoRidgeShapeHelp(View view) {
    final AlertDialog.Builder builder = new
AlertDialog.Builder(originalHorsetailsSearch.this);
    if (Build.VERSION.SDK_INT >=
Build.VERSION_CODES.LOLLIPOP) {
        builder.setView(R.layout.help_calamo_ridge_shape);
    } else {
        builder.setMessage(R.string.calamo_ridge_shape);
    }
    builder.setPositiveButton("understood", new
DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int
which) {
            dialog.cancel();
        }
    });
    builder.show();
}
```

Code box 6-2 - Exemplar method for a help pop-out box showing the code for help button for the shape of the ridges on the main stem

The code was initially built up by repeating working blocks as required for each question, with a single class holding all methods required for each layout page of the app.

The database was held in a separate class. As per the results of chapter 2 where this was found to be the preferred approach, the onCreate of this class contained and executed all the required database creation and data insert SQL statements. The database holding class also included the method that actually performed the database search, and returned the list of results. This method received the searchQuery from the results page, searched using the rawQuery method implemented in Android, and used the results to build the list of results that was returned to the results class.

The results class received the list of results from the searchQuery, and used this to fill in the list of results. This was achieved by iterating through this list, and using the data in each element to fill out a custom view. Each view was then added to a scrollable listView, which was displayed to the user.

The re-written code aimed to reuse as much code as possible. Thus, a single method was written for each of: a radioButton being pressed, a reset button being pressed, and a help button being pressed. Each of these methods generally followed the same behaviour as in the original version of the code (i.e. same general layout of code within the method), but modified to gather the specific data required from the helper classes.. They typically gather the ID of the button pressed, and use this information to look up the correct data from the relevant helper class. For example, when a reset button was pressed, the method would gather which elements of the layout needed to be reset from its helper class, and reset all of them. The reset everything button simply gathered all the data from the helper class. Some parts of the code were not easily reusable, such as the onClickListeners for the seekBars, and were retained. The database initiation was delayed slightly, which allowed the database choice alertDialogue to be displayed to the user beforehand. The results of this choice were then passed to the database class, which then executed the correct set of insert statements based on this information. This meant only a single database class was required.

6.3.3.1 App use tutorial

The app and ID key tutorial requested in the feedback during app testing and evaluation (see chapter 7) was included in the re-written app. In the class diagram shown in Figure 6-4, this tutorial uses the final nine methods at the end of mainActivity. These methods all create alertDialogues, with similar behaviour to the help boxes. Due to differing button requirements between different boxes, each new box required a new method. The exception to this was the set of boxes required for each chain of instruction. The windows navigation diagram for this introduction is shown in Figure 6-2, with additional information for this figure provided in Table 6-1 and Table 6-2.

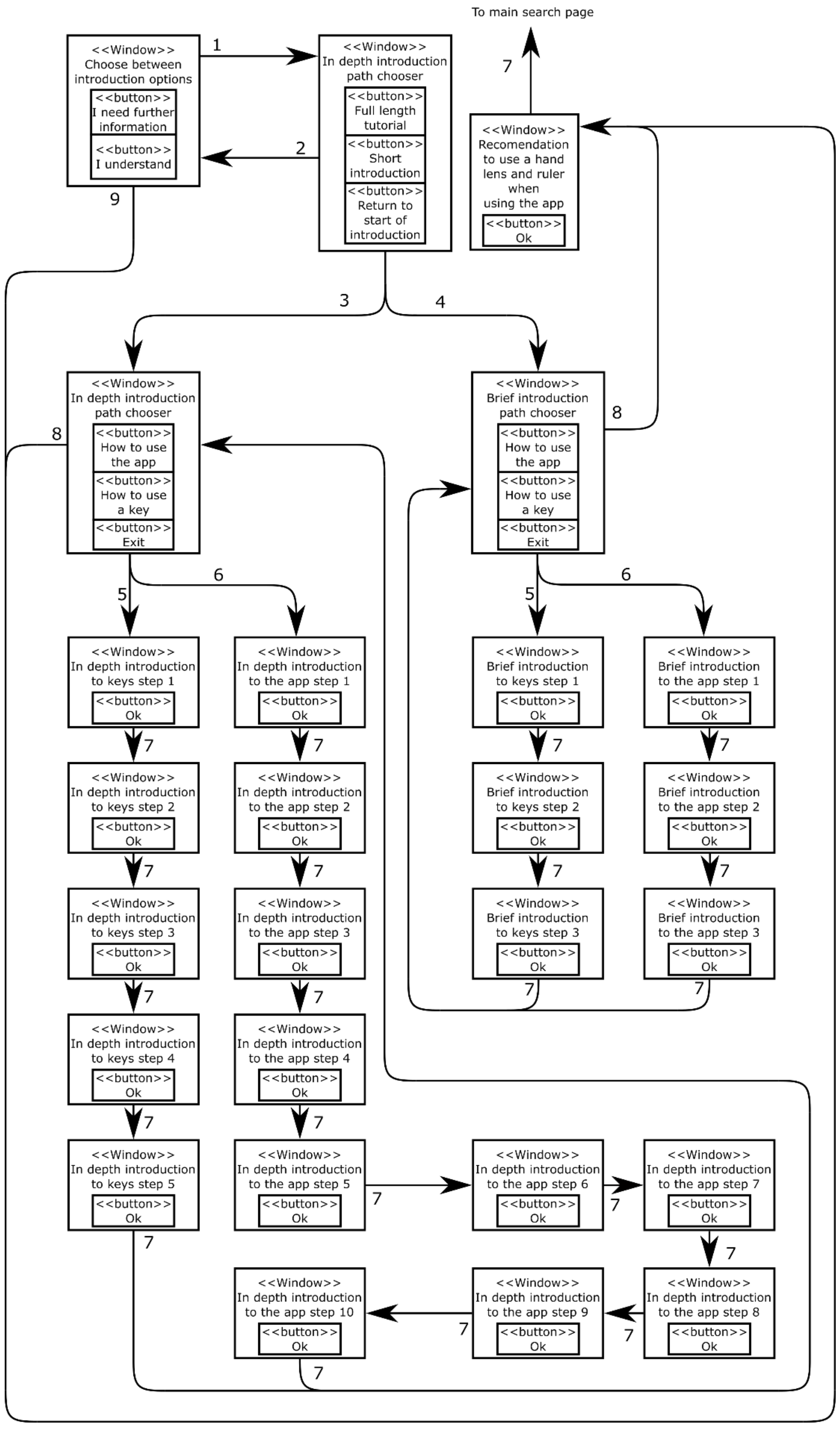

To main search page
<<Window>> Choose between introduction options
<<button>> I need further information
<<button>> I understand
<<Window>> In depth introduction path chooser
<<button>> Full length tutorial
<<button>> Short introduction
<<button>> Return to start of introduction
<<Window>> Recomendation to use a hand lens and ruler when using the app
<<button>> Ok
<<Window>> In depth introduction path chooser
<<button>> How to use the app
<<button>> How to use a key
<<button>> Exit
<<Window>> Brief introduction path chooser
<<button>> How to use the app
<<button>> How to use a key
<<button>> Exit
<<Window>> In depth introduction to keys step 1
<<Window>> In depth introduction to keys step 2
<<Window>> In depth introduction to keys step 3
<<Window>> In depth introduction to keys step 4
<<Window>> In depth introduction to keys step 5
<<Window>> In depth introduction to the app step 1
<<Window>> In depth introduction to the app step 2
<<Window>> In depth introduction to the app step 3
<<Window>> In depth introduction to the app step 4
<<Window>> In depth introduction to the app step 5
<<Window>> In depth introduction to the app step 6
<<Window>> In depth introduction to the app step 7
<<Window>> In depth introduction to the app step 8
<<Window>> In depth introduction to the app step 9
<<Window>> In depth introduction to the app step 10
<<Window>> Brief introduction to keys step 1
<<Window>> Brief introduction to keys step 2
<<Window>> Brief introduction to keys step 3
<<Window>> Brief introduction to the app step 1
<<Window>> Brief introduction to the app step 2
<<Window>> Brief introduction to the app step 3
<<button>> Ok
1
2
3
4
5
6
7
8
9

Figure 6-2 - The steps the user can choose to go through in the apps tutorial. For the full text of each window, see Table 6-1. For the labels for each arrow, see Table 6-2.

Table 6-1 (Over the next 3 pages) - Full text of each window in the app and ID key tutorial. Window navigation diagram of the tutorial shown in Figure 6-2

Step in figure	**Full text**
Choose between introduction options	If you have used Android apps before, and understand that you do not have to answer every question in a multi-access key for them to work successfully, press the 'I understand' button below. Otherwise, press the 'I need further information button'.
Tutorial depth chooser	If you would like a full-length tutorial, please press the 'full-length introduction' button below. This is recommended for people who are mostly unfamiliar with plant identification keys or android apps. If you have used either of these things before, but would like a brief introduction to the app, press the 'short introduction' button below. If you do not feel like you need an introduction to the app at all, please press the 'back' button.
In depth introduction path chooser	Would you like instructions on how to use this app, or an introduction to identification keys? If you are happy to use the app, press 'exit'.
In depth introduction to keys step 1	1/5 In order to identify an object using a key, you need to answer questions about it. In identification keys, this will typically relate to how the object looks. This information is then used to eliminate possibilities, and narrow down the list to (ideally) a single result.
In depth introduction to keys step 2	2/5 In multi-access identification keys (like this app), you do not need to answer every question. They are designed such that you can answer the questions are most confident about first. If there are still a large number of results remaining, you then begin answering questions you are less confident about, until there is one result remaining.
In depth introduction to keys step 3	3/5 When you are answering questions in any key, precision in answering is required. For example, the number of ridges on the stem may be critical in distinguishing between two similar species.
In depth introduction to keys step 4	4/5 There may be cases where two or more taxa are very similar, and distinguishing between them requires a small feature that is either impossible to measure or not present on the specimen. Do not be alarmed if you reach a list of a few possible identifications.

In depth introduction to keys step 5	5/5 It is recommended that for this key you have a hand-lens and a ruler available, as this will facilitate precise answering of several questions. Please note that this is not a requirement as questions can be avoided.
In depth introduction to the app step 1	1/10 You will find buttons under many of the questions. You need to read the question carefully, select the correct answer, and press on the button next to the most appropriate answer.
In depth introduction to the app step 2	2/10 This is a slider. These are used when the answer can be any single number from a large range. You need to move the orange dot to the appropriate place along the bar. Your current choice is displayed above the bar.
In depth introduction to the app step 3	3/10 If you would prefer to use buttons instead of sliders, you can press this button in the top bar of the screen. From there, you can choose which questions you would like to change from sliders to buttons.
In depth introduction to the app step 4	4/10 The question mark icon indicates that additional help, such as explanatory diagrams, glossarial information, or further information about the question is available. You can press on this icon to receive this further information in the form of a 'pop-out' box. Once understood, this box can be dismissed.
In depth introduction to the app step 5	5/10 The refresh icon indicates that a question can be 'reset', should you want to remove your answer to the question from the current search.
In depth introduction to the app step 6	6/10 The clear all button has the same effect as the 'reset' button, but instead of relating to a single question, it resets ALL questions. This can be useful, for example, if you are trying to identify a new specimen.
In depth introduction to the app step 7	7/10 Pay attention to the text at the top – it is displaying how many possible results there currently are, and will constantly update based on your current answers. You do not have to get to a single result; two possible results may be similar, and your specimen may not show the required feature to distinguish between them.
In depth introduction to the app step 8	8/10 Once you have answered all the questions you feel sure of, press the search button to see the results.
In depth introduction to the app step 9	9/10 When you see the list of results, you can press on each different one. This will open a new 'pop-out' box, with information about the result. This includes how to tell the taxon from ones that look similar. If you are connected to the internet, you can press on the 'further information' button to be taken to a website containing some further information about each taxon.

In depth introduction to the app step 10	10/10 Remember that you can press the back button, and return to the list of questions if you would like to adapt your answer (for example if you discover you have miscounted a feature).
Brief introduction path chooser	Would you like an introduction to the app, or to identification keys? If you are happy to use the app, press 'exit'.
Brief introduction to keys step 1	1/3 To identify specimens using an identification key, you need to answer questions about the specimen. The key uses this information to narrow down the list of potential identities.
Brief introduction to keys step 2	2/3 When you are answering questions in any key, precision in answering is required. The number of ridges on the stem may be critical in distinguishing between two similar species for example.
Brief introduction to keys step 3	3/3 It is recommended that you have a hand-lens and ruler, as they may be required for several of the answers. Note that they are not required, the key will still function if many questions are un-answered.
Brief introduction to the app step 1	1/3 Press this icon if you would like to read further guidance about a question. Press this icon to reset a question to an unanswered state. Press this button if you would like to reset all questions to an unanswered state. Slider-style questions can be toggled to buttons in the menu.
Brief introduction to the app step 2	2/3 This text at the top displays the current number of possibilities based on the current answers you have input to the key. You do not have to get to a single result; two possible results may be similar, and your specimen may not show the required feature to distinguish between them.
Brief introduction to the app step 3	3/3 Once you have answered all questions you are confident in the answers to, press the search button to see the results. If you would like further information about a result, press on it in the result list to find out more. This information includes how to tell each taxa from ones that look similar. Remember you can press back to modify your answer to a question if required.
Recommendation to use a hand lens and a ruler while using the app	For this key, it is recommended that you have a hand-lens and a ruler with millimetre markings. Please note that as any question can be avoided, these are not necessarily a requirement

Table 6-2 - Labels for arrows in the windows navigation shown in Figure 6-2.

Arrow	Label
1	Click 'I need further information' button
2	Click 'Return to start of introduction' button
3	Click 'Full length tutorial' button
4	Click 'Short introduction' button
5	Click 'How to use a key' button
6	Click 'How to use the app' button
7	Click 'OK' button
8	Click 'Exit' button
9	Click 'I understand' button

6.3.4 Layout code

The layout code for questions was based on the same design of a textView showing the question, a radioGroup of the required number of radioButtons for the possible answers (with seekBars as the primary option when the answers were numerical and covered a continuous range of greater than five values), a help button (where appropriate), and a reset button. A pseudocode example of this design is shown in Code box 6-3. This question was repeated with a bespoke layout for each question as required. These questions were put in a scroll view, so the user could scroll through the questions, and combined with the actionBar and topBar (showing the number of results remaining), and the bottom buttons, to produce the layout shown in Code box 6-4 (note that this is a pseudocode representation of the actual code for brevity).

```
<Text view
      {question}/>
<Button
      {help button}/>
<Button
      {reset button}/>
<RadioGroup
      <RadioButton
            {answer}/>
      <RadioButton
            {answer}/>
                        />
```

Code box 6-3 – Pseudocode XML layout for individual questions. Note that the exact order of elements within the questions is not fixed, Android XML includes layout tags, which indicate where the different elements of the layout will go.

```
<RelativeLayout
       {Necessary code}>
              <TableRow
                     {Necessary code}>
                            <Button
                                   {Search button layout}>
                            <Button
                                   {Reset button layout}>
              </TableRow>
              <TextView
                     {“You don’t need to answer every question, just the ones you think you can”}
              </TextView>
              <ScrollView
                     {Necessary code}>
                            <RelativeLayout
                                   {Necessary code}>
[Repeat below as required]
                                          <TextView
                                                 {Question text}>
                                          </TextView>
                                          <Radiobuttons or seekbar as required
                                                 {Necessary code}>
                                          <Button
                                                 {Reset individual question}>
                                          </Button>
                                          <Button
                                                 {Individual question help}>
                                          </Button>
[End of repeating section]
                            </RelativeLayout>
              </ScrollView>
</RelativeLayout>
```

Code box 6-4 - Pseudo-code layout of the search page

The layout of the results page is shown in Code box 6-5, and is an empty list. Results were gathered from the SQLite search. Each individual result supplied a single species name. The empty list was filled by using each separate species name from the results list to fill in the 'android:text' tag in Code box 6-6. The tag was filled in with a single species name, then the whole layout was placed into the list. This was repeated until every species name in the set of results was in the list. The ability for the user to press on an individual result and be presented with a pop-up box with further information about the taxa was provided through an onItemClickListener.

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
<ListView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@android:id/list">
</ListView>
</TableLayout>
```

Code box 6-5 - The full code for the results page.

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <TableLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">

        <TableRow
            android:layout_width="wrap_content"
            android:weightSum="3"
            android:layout_height="wrap_content">
              <TextView
                     android:id="@+id/speciesName"
                     android:layout_width="fill_parent"
                     android:layout_height="fill_parent"
                     android:text=""
                     android:textColor="#444444"
                     android:textSize="20sp"
                     android:layout_weight="1"
                     android:textStyle="bold" />
       </TableRow>
      </TableLayout>
</RelativeLayout>
```

Code box 6-6 - The text view layout used to fill the results list

6.4 Results

The class diagram for the app, prior to the core re-write, is shown in Figure 6-3. Note that due to the significant number of attributes and methods required for this version the font size for these boxes has been shrunk significantly to allow the diagram to fit on A4.

Figure 6-4 shows the class diagram for the code after the re-write. As can be seen, there are several different helper classes, and the main logical-code class has considerably fewer methods and attributes. Figure 6-4 is expanded in Figure 6-5, Figure 6-6, Figure 6-7, and Figure 6-8 for clarity.

For the final code, see Appendix A (available on the attached DVD).

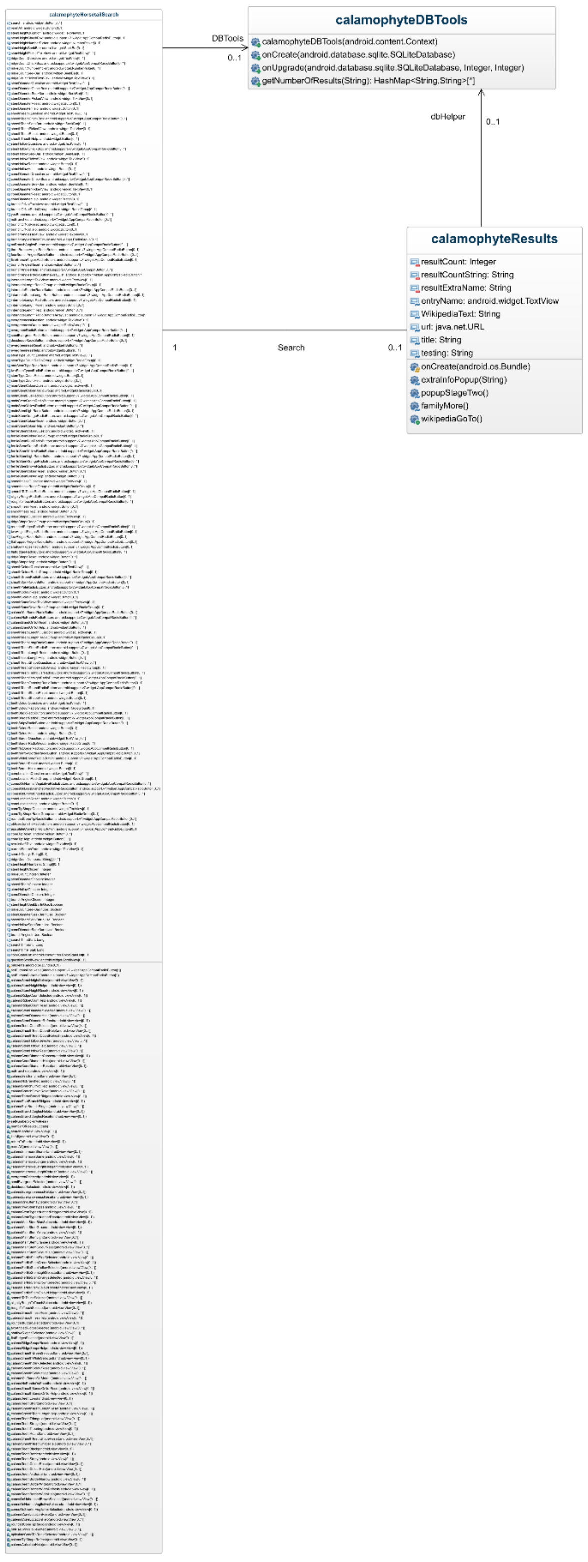

calamophyteDBTools
calamophyteDBTools(android.content.Context)
onCreate(android.database.sqlite.SQLiteDatabase)
onUpgrade(android.database.sqlite.SQLiteDatabase, Integer, Integer)
getNumberOfResults(String): HashMap<String,String>[*]
DBTools
0..1
dbHelper
0..1
calamophyteResults
resultCount: Integer
resultCountString: String
resultExtraName: String
entryName: android.widget.TextView
WikipediaText: String
url: java.net.URL
title: String
testing: String
onCreate(android.os.Bundle)
extraInfoPopup(String)
popupStageTwo()
familyMore()
wikipediaGoTo()
1
Search
0..1

Figure 6-3 - Class diagram of a module of the app before the re-write of the app. Diagram made using GenMyModel (2019) *and modified using InkScape* (2019). *Note that the text in the classes is not intended to be readable, instead is illustrative of the number of methods and attributes in the classes.*

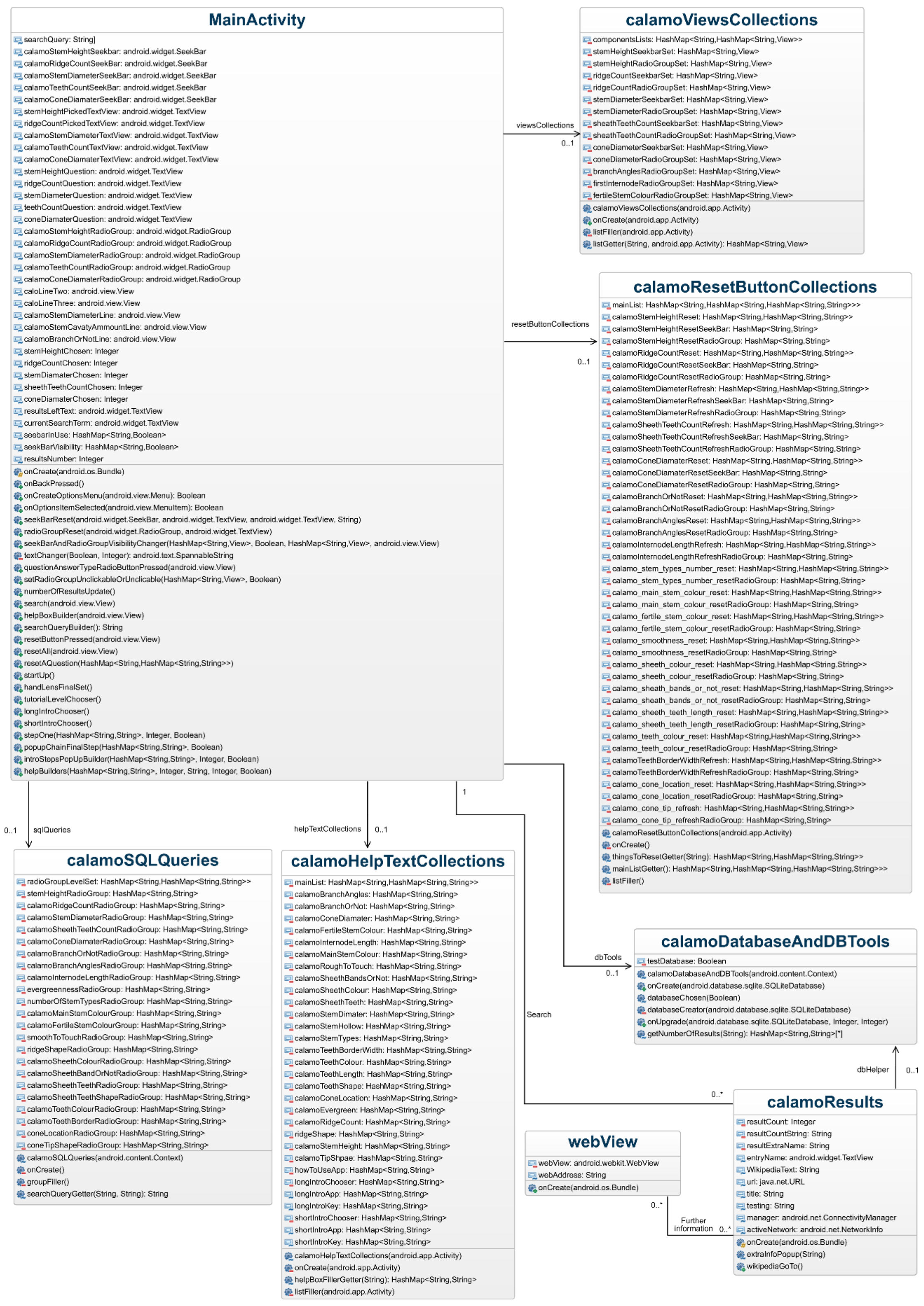


Figure 6-4 - Class diagram of a 'module' of the app, or the full app as in Appendix A , after the code re-write. Diagram made using GenMyModel (2019) *and modified using InkScape* (2019)*. Note that this figure has been deliberately reduced in size to fit on an A4 page. This diagram is expanded in Figure 6-5, Figure 6-6, Figure 6-7, and Figure 6-8 for clarity.*

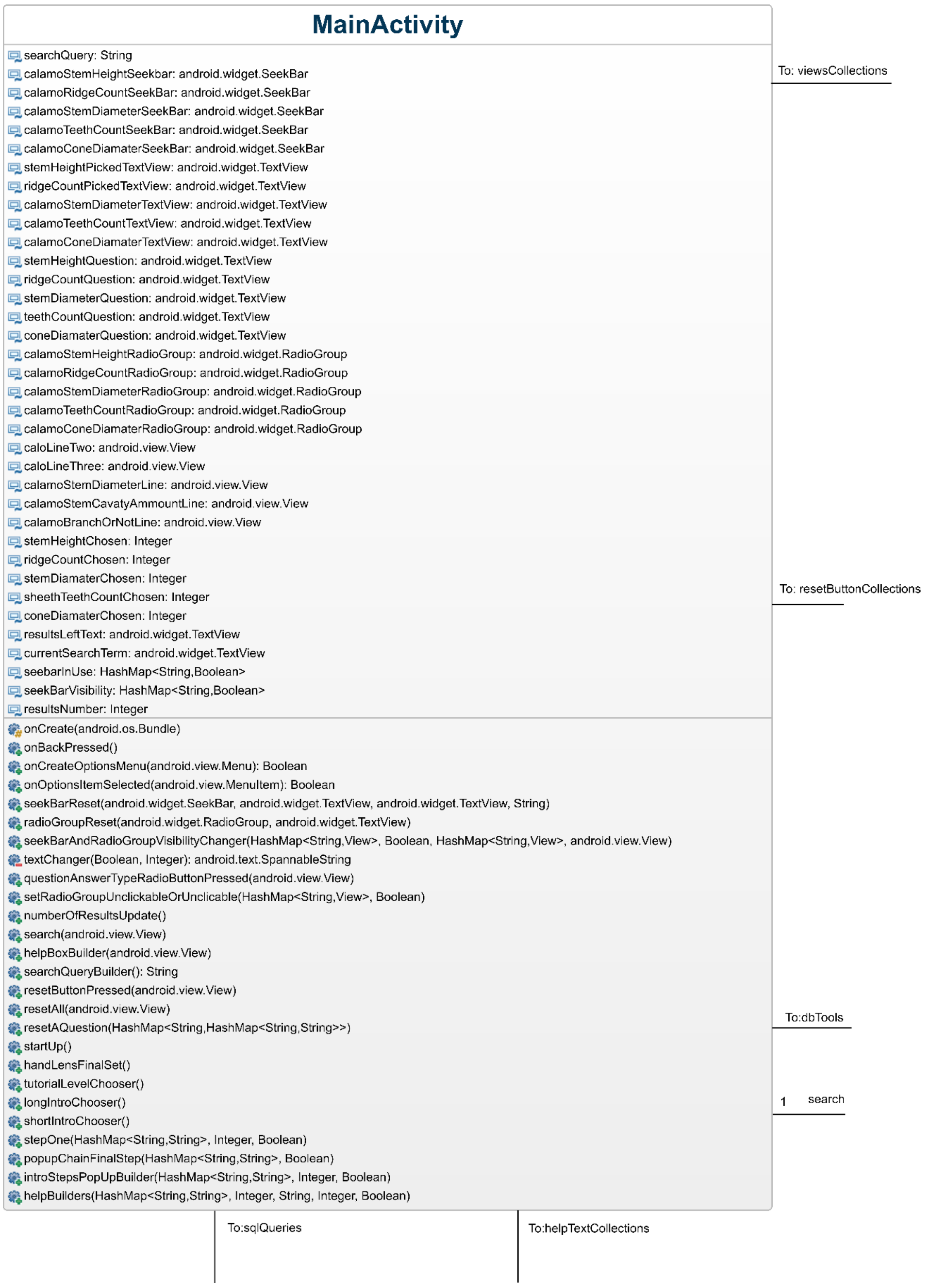

MainActivity
searchQuery: String
calamoStemHeightSeekbar: android.widget.SeekBar
calamoRidgeCountSeekBar: android.widget.SeekBar
calamoStemDiameterSeekBar: android.widget.SeekBar
calamoTeethCountSeekBar: android.widget.SeekBar
calamoConeDiamaterSeekBar: android.widget.SeekBar
stemHeightPickedTextView: android.widget.TextView
ridgeCountPickedTextView: android.widget.TextView
calamoStemDiameterTextView: android.widget.TextView
calamoTeethCountTextView: android.widget.TextView
calamoConeDiamaterTextView: android.widget.TextView
stemHeightQuestion: android.widget.TextView
ridgeCountQuestion: android.widget.TextView
stemDiameterQuestion: android.widget.TextView
teethCountQuestion: android.widget.TextView
coneDiamaterQuestion: android.widget.TextView
calamoStemHeightRadioGroup: android.widget.RadioGroup
calamoRidgeCountRadioGroup: android.widget.RadioGroup
calamoStemDiameterRadioGroup: android.widget.RadioGroup
calamoTeethCountRadioGroup: android.widget.RadioGroup
calamoConeDiamaterRadioGroup: android.widget.RadioGroup
caloLineTwo: android.view.View
caloLineThree: android.view.View
calamoStemDiameterLine: android.view.View
calamoStemCavatyAmmountLine: android.view.View
calamoBranchOrNotLine: android.view.View
stemHeightChosen: Integer
ridgeCountChosen: Integer
stemDiamaterChosen: Integer
sheethTeethCountChosen: Integer
coneDiamaterChosen: Integer
resultsLeftText: android.widget.TextView
currentSearchTerm: android.widget.TextView
seebarInUse: HashMap<String,Boolean>
seekBarVisibility: HashMap<String,Boolean>
resultsNumber: Integer
onCreate(android.os.Bundle)
onBackPressed()
onCreateOptionsMenu(android.view.Menu): Boolean
onOptionsItemSelected(android.view.MenuItem): Boolean
seekBarReset(android.widget.SeekBar, android.widget.TextView, android.widget.TextView, String)
radioGroupReset(android.widget.RadioGroup, android.widget.TextView)
seekBarAndRadioGroupVisibilityChanger(HashMap<String,View>, Boolean, HashMap<String,View>, android.view.View)
textChanger(Boolean, Integer): android.text.SpannableString
questionAnswerTypeRadioButtonPressed(android.view.View)
setRadioGroupUnclickableOrUnclicable(HashMap<String,View>, Boolean)
numberOfResultsUpdate()
search(android.view.View)
helpBoxBuilder(android.view.View)
searchQueryBuilder(): String
resetButtonPressed(android.view.View)
resetAll(android.view.View)
resetAQuestion(HashMap<String,HashMap<String,String>>)
startUp()
handLensFinalSet()
tutorialLevelChooser()
longIntroChooser()
shortIntroChooser()
stepOne(HashMap<String,String>, Integer, Boolean)
popupChainFinalStep(HashMap<String,String>, Boolean)
introStepsPopUpBuilder(HashMap<String,String>, Integer, Boolean)
helpBuilders(HashMap<String,String>, Integer, String, Integer, Boolean)
To: viewsCollections
To: resetButtonCollections
To:dbTools
1
search
To:sqlQueries
To:helpTextCollections

Figure 6-5 - Partial view of Figure 6-4, with the 'mainActivity' class expanded for clarity. Diagram made using GenMyModel (2019) *and modified using InkScape* (2019).

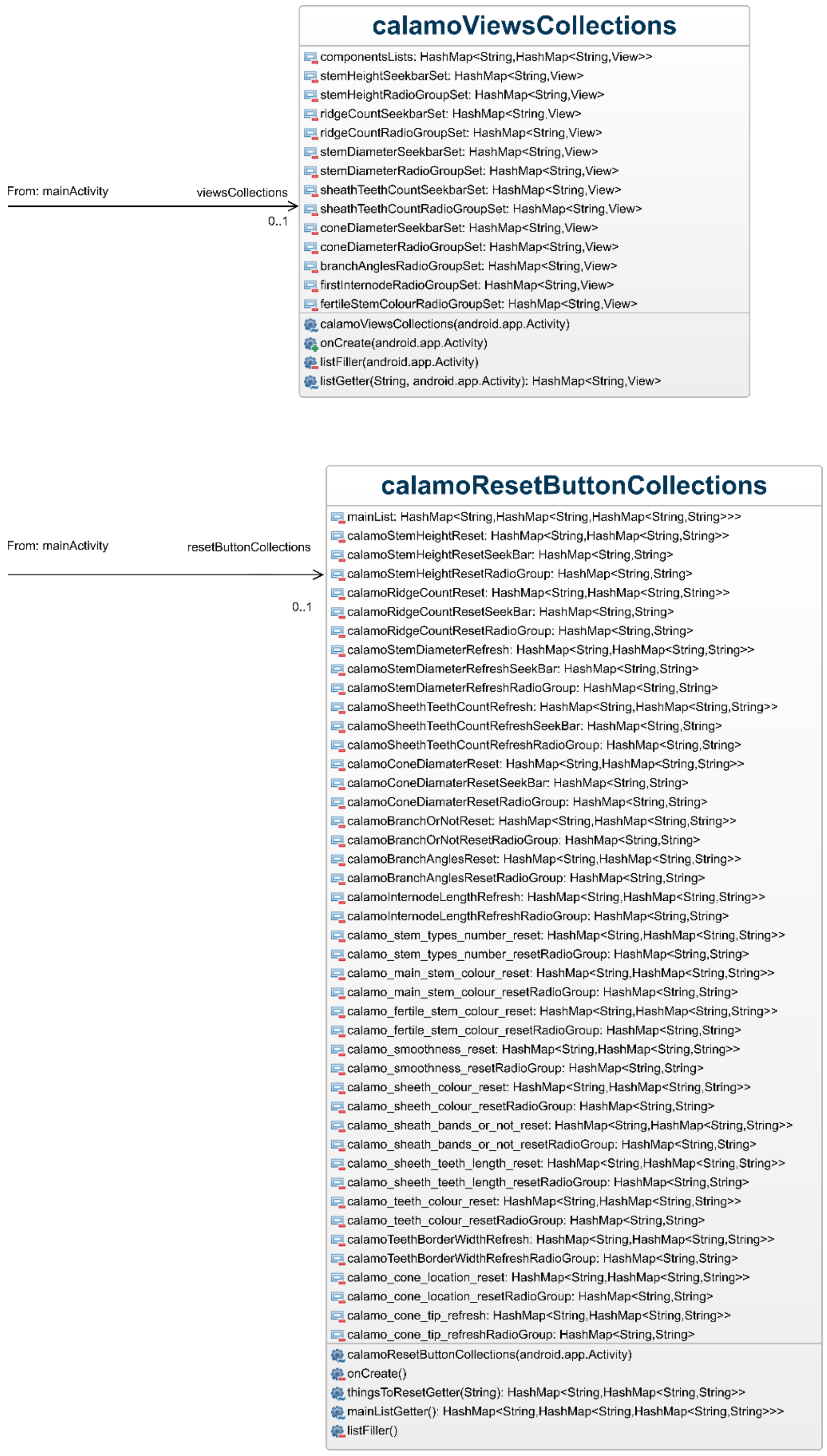

calamoViewsCollections
componentsLists: HashMap<String,HashMap<String,View>>
stemHeightSeekbarSet: HashMap<String,View>
stemHeightRadioGroupSet: HashMap<String,View>
ridgeCountSeekbarSet: HashMap<String,View>
ridgeCountRadioGroupSet: HashMap<String,View>
stemDiameterSeekbarSet: HashMap<String,View>
stemDiameterRadioGroupSet: HashMap<String,View>
sheathTeethCountSeekbarSet: HashMap<String,View>
sheathTeethCountRadioGroupSet: HashMap<String,View>
coneDiameterSeekbarSet: HashMap<String,View>
coneDiameterRadioGroupSet: HashMap<String,View>
branchAnglesRadioGroupSet: HashMap<String,View>
firstInternodeRadioGroupSet: HashMap<String,View>
fertileStemColourRadioGroupSet: HashMap<String,View>
calamoViewsCollections(android.app.Activity)
onCreate(android.app.Activity)
listFiller(android.app.Activity)
listGetter(String, android.app.Activity): HashMap<String,View>
From: mainActivity
viewsCollections
0..1
calamoResetButtonCollections
mainList: HashMap<String,HashMap<String,HashMap<String,String>>>
calamoStemHeightReset: HashMap<String,HashMap<String,String>>
calamoStemHeightResetSeekBar: HashMap<String,String>
calamoStemHeightResetRadioGroup: HashMap<String,String>
calamoRidgeCountReset: HashMap<String,HashMap<String,String>>
calamoRidgeCountResetSeekBar: HashMap<String,String>
calamoRidgeCountResetRadioGroup: HashMap<String,String>
calamoStemDiameterRefresh: HashMap<String,HashMap<String,String>>
calamoStemDiameterRefreshSeekBar: HashMap<String,String>
calamoStemDiameterRefreshRadioGroup: HashMap<String,String>
calamoSheethTeethCountRefresh: HashMap<String,HashMap<String,String>>
calamoSheethTeethCountRefreshSeekBar: HashMap<String,String>
calamoSheethTeethCountRefreshRadioGroup: HashMap<String,String>
calamoConeDiamaterReset: HashMap<String,HashMap<String,String>>
calamoConeDiamaterResetSeekBar: HashMap<String,String>
calamoConeDiamaterResetRadioGroup: HashMap<String,String>
calamoBranchOrNotReset: HashMap<String,HashMap<String,String>>
calamoBranchOrNotResetRadioGroup: HashMap<String,String>
calamoBranchAnglesReset: HashMap<String,HashMap<String,String>>
calamoBranchAnglesResetRadioGroup: HashMap<String,String>
calamoInternodeLengthRefresh: HashMap<String,HashMap<String,String>>
calamoInternodeLengthRefreshRadioGroup: HashMap<String,String>
calamo_stem_types_number_reset: HashMap<String,HashMap<String,String>>
calamo_stem_types_number_resetRadioGroup: HashMap<String,String>
calamo_main_stem_colour_reset: HashMap<String,HashMap<String,String>>
calamo_main_stem_colour_resetRadioGroup: HashMap<String,String>
calamo_fertile_stem_colour_reset: HashMap<String,HashMap<String,String>>
calamo_fertile_stem_colour_resetRadioGroup: HashMap<String,String>
calamo_smoothness_reset: HashMap<String,HashMap<String,String>>
calamo_smoothness_resetRadioGroup: HashMap<String,String>
calamo_sheeth_colour_reset: HashMap<String,HashMap<String,String>>
calamo_sheeth_colour_resetRadioGroup: HashMap<String,String>
calamo_sheath_bands_or_not_reset: HashMap<String,HashMap<String,String>>
calamo_sheath_bands_or_not_resetRadioGroup: HashMap<String,String>
calamo_sheeth_teeth_length_reset: HashMap<String,HashMap<String,String>>
calamo_sheeth_teeth_length_resetRadioGroup: HashMap<String,String>
calamo_teeth_colour_reset: HashMap<String,HashMap<String,String>>
calamo_teeth_colour_resetRadioGroup: HashMap<String,String>
calamoTeethBorderWidthRefresh: HashMap<String,HashMap<String,String>>
calamoTeethBorderWidthRefreshRadioGroup: HashMap<String,String>
calamo_cone_location_reset: HashMap<String,HashMap<String,String>>
calamo_cone_location_resetRadioGroup: HashMap<String,String>
calamo_cone_tip_refresh: HashMap<String,HashMap<String,String>>
calamo_cone_tip_refreshRadioGroup: HashMap<String,String>
calamoResetButtonCollections(android.app.Activity)
onCreate()
thingsToResetGetter(String): HashMap<String,HashMap<String,String>>
mainListGetter(): HashMap<String,HashMap<String,HashMap<String,String>>>
listFiller()
From: mainActivity
resetButtonCollections
0..1

Figure 6-6 - Partial view of Figure 6-4, with the two helper classes 'calamoViewCollections' and 'calamoResetButtonCollections' expanded for clarity. Diagram made using GenMyModel (2019) *and modified using InkScape* (2019).

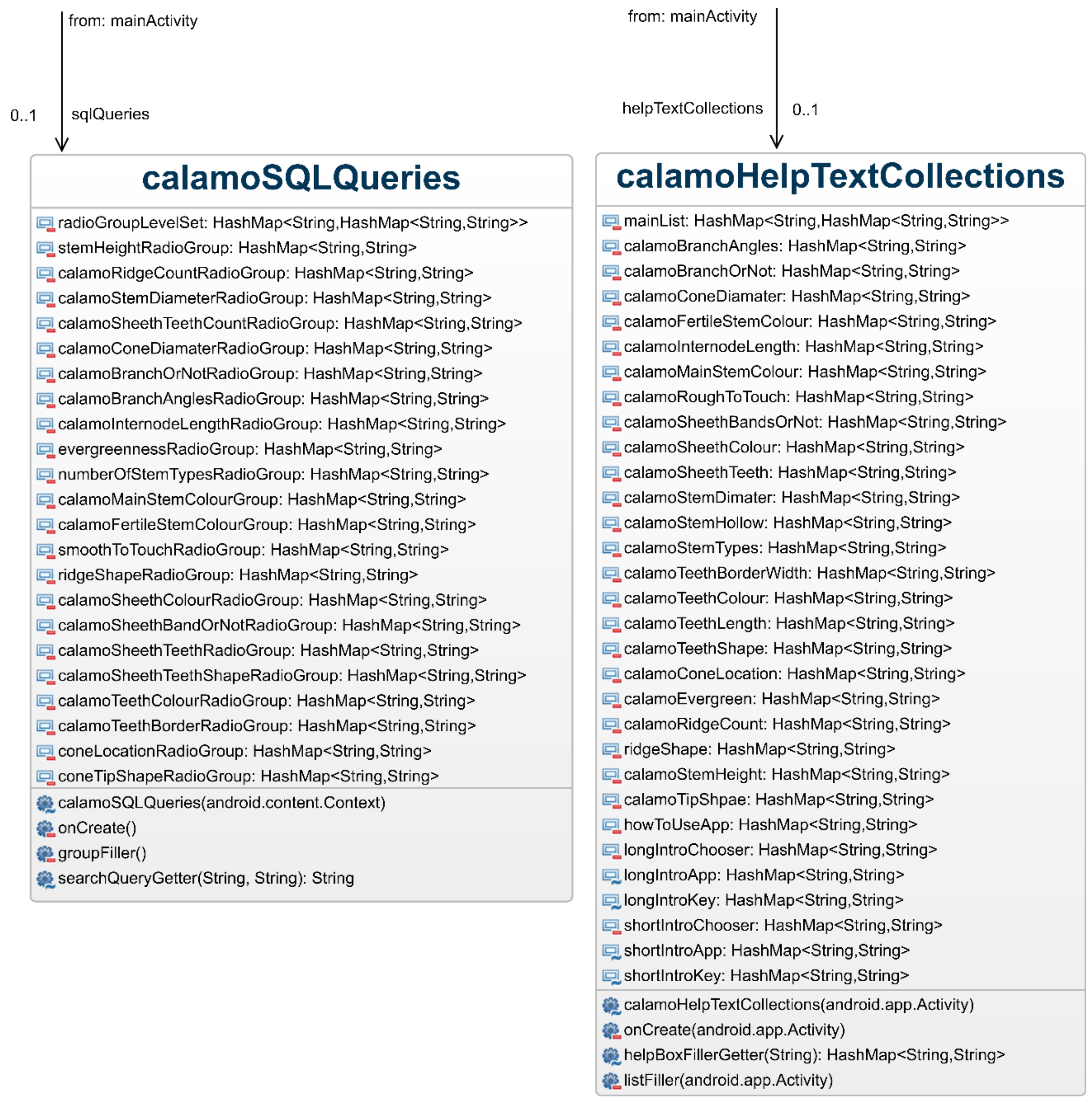

from: mainActivity
0..1
sqlQueries
from: mainActivity
helpTextCollections
0..1
calamoSQLQueries
radioGroupLevelSet: HashMap<String,HashMap<String,String>>
stemHeightRadioGroup: HashMap<String,String>
calamoRidgeCountRadioGroup: HashMap<String,String>
calamoStemDiameterRadioGroup: HashMap<String,String>
calamoSheethTeethCountRadioGroup: HashMap<String,String>
calamoConeDiamaterRadioGroup: HashMap<String,String>
calamoBranchOrNotRadioGroup: HashMap<String,String>
calamoBranchAnglesRadioGroup: HashMap<String,String>
calamoInternodeLengthRadioGroup: HashMap<String,String>
evergreennessRadioGroup: HashMap<String,String>
numberOfStemTypesRadioGroup: HashMap<String,String>
calamoMainStemColourGroup: HashMap<String,String>
calamoFertileStemColourGroup: HashMap<String,String>
smoothToTouchRadioGroup: HashMap<String,String>
ridgeShapeRadioGroup: HashMap<String,String>
calamoSheethColourRadioGroup: HashMap<String,String>
calamoSheethBandOrNotRadioGroup: HashMap<String,String>
calamoSheethTeethRadioGroup: HashMap<String,String>
calamoSheethTeethShapeRadioGroup: HashMap<String,String>
calamoTeethColourRadioGroup: HashMap<String,String>
calamoTeethBorderRadioGroup: HashMap<String,String>
coneLocationRadioGroup: HashMap<String,String>
coneTipShapeRadioGroup: HashMap<String,String>
calamoSQLQueries(android.content.Context)
onCreate()
groupFiller()
searchQueryGetter(String, String): String
calamoHelpTextCollections
mainList: HashMap<String,HashMap<String,String>>
calamoBranchAngles: HashMap<String,String>
calamoBranchOrNot: HashMap<String,String>
calamoConeDiamater: HashMap<String,String>
calamoFertileStemColour: HashMap<String,String>
calamoInternodeLength: HashMap<String,String>
calamoMainStemColour: HashMap<String,String>
calamoRoughToTouch: HashMap<String,String>
calamoSheethBandsOrNot: HashMap<String,String>
calamoSheethColour: HashMap<String,String>
calamoSheethTeeth: HashMap<String,String>
calamoStemDimater: HashMap<String,String>
calamoStemHollow: HashMap<String,String>
calamoStemTypes: HashMap<String,String>
calamoTeethBorderWidth: HashMap<String,String>
calamoTeethColour: HashMap<String,String>
calamoTeethLength: HashMap<String,String>
calamoTeethShape: HashMap<String,String>
calamoConeLocation: HashMap<String,String>
calamoEvergreen: HashMap<String,String>
calamoRidgeCount: HashMap<String,String>
ridgeShape: HashMap<String,String>
calamoStemHeight: HashMap<String,String>
calamoTipShpae: HashMap<String,String>
howToUseApp: HashMap<String,String>
longIntroChooser: HashMap<String,String>
longIntroApp: HashMap<String,String>
longIntroKey: HashMap<String,String>
shortIntroChooser: HashMap<String,String>
shortIntroApp: HashMap<String,String>
shortIntroKey: HashMap<String,String>
calamoHelpTextCollections(android.app.Activity)
onCreate(android.app.Activity)
helpBoxFillerGetter(String): HashMap<String,String>
listFiller(android.app.Activity)

Figure 6-7 - Partial view of Figure 6-4, with the two helper classes 'calamoSQLQueries' and 'calamoHelpTextCollections' expanded for clarity. Diagram made using GenMyModel (2019) *and modified using InkScape* (2019).

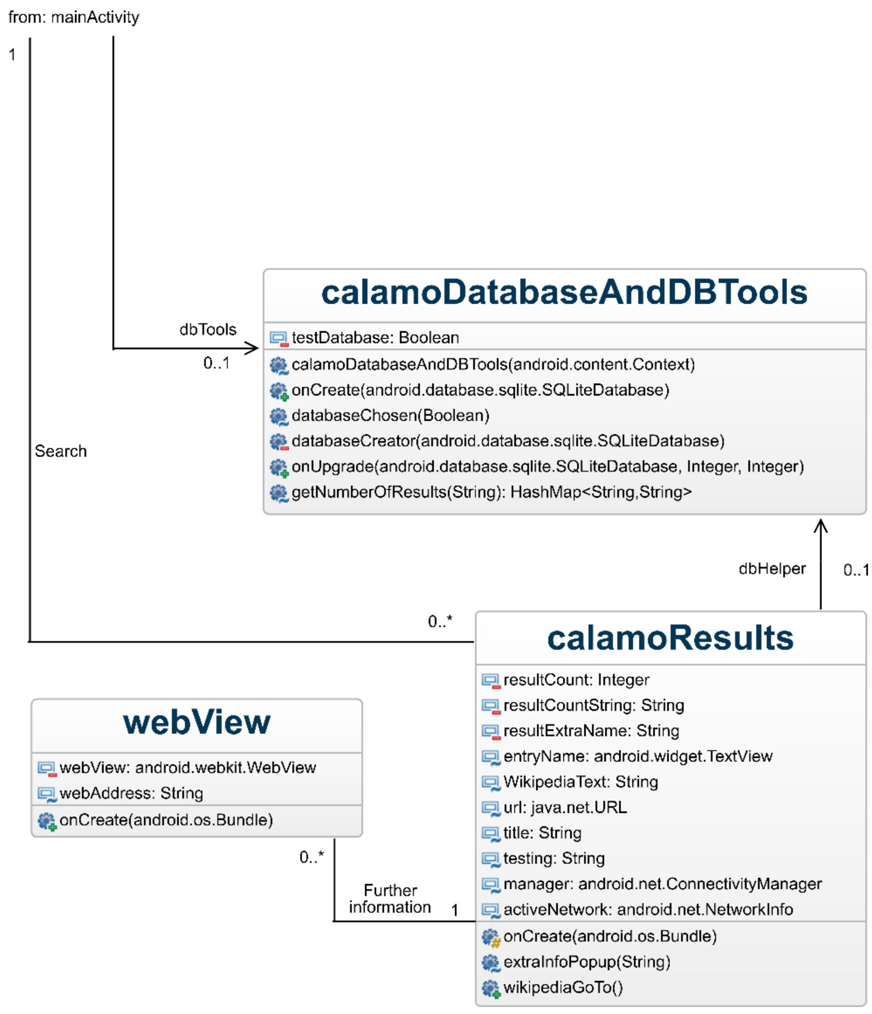

from: mainActivity
1
dbTools
0..1
Search
calamoDatabaseAndDBTools
testDatabase: Boolean
calamoDatabaseAndDBTools(android.content.Context)
onCreate(android.database.sqlite.SQLiteDatabase)
databaseChosen(Boolean)
databaseCreator(android.database.sqlite.SQLiteDatabase)
onUpgrade(android.database.sqlite.SQLiteDatabase, Integer, Integer)
getNumberOfResults(String): HashMap<String,String>
dbHelper
0..1
0..*
calamoResults
resultCount: Integer
resultCountString: String
resultExtraName: String
entryName: android.widget.TextView
WikipediaText: String
url: java.net.URL
title: String
testing: String
manager: android.net.ConnectivityManager
activeNetwork: android.net.NetworkInfo
onCreate(android.os.Bundle)
extraInfoPopup(String)
wikipediaGoTo()
webView
webView: android.webkit.WebView
webAddress: String
onCreate(android.os.Bundle)
0..*
Further information
1

Figure 6-8 - Partial view of Figure 6-4, with the classes 'calamoDatabaseAndDBTools', 'calamoResults', and 'webView' expanded for clarity. These classes hold the database for the Calamophyes 'module', the search method, the results page, and the method for opening a webpage to view further information about the selected taxon. Diagram made using GenMyModel (2019) *and modified using InkScape* (2019). *Note that the '1' relationship indicator for search has been repeated from Figure 6-5.*

6.5 Discussion

The best practices used here helped produce readable and useful code. However, as the app was being developed and different approaches to programming various aspects were investigated, these were not always followed strictly. The best practices here are also not necessarily as thorough as other practices. The use of a limited set of best practices was an active choice, as it was felt that it would allow easier development of the app. By the end of the development it was clear that this was the wrong approach, and that having a thorough and strict set of coding practices would have helped produced more well-designed code.

The layout code was simple to produce; a set of questions, with two buttons at the bottom of the screen, and a menu in the action bar. The questions were all based on the same design of a textView showing the question, a radioGroup of the required number of radioButtons for the possible answers (with seekBars as the primary option when the answers were numerical and covered a continuous range of greater than five values), a help button (where appropriate), and a reset button. The layout code was therefore simple to build up quickly by repeating the same question units. Given this repetitive nature, it may well be entirely possible to automate the generation of the layout. This could either be done during app code implementation (For example using a bespoke java program to generate the required XML), or at app run-time (Potentially by including a helper-class with the relevant data). This approach may aid implementation of a question list that re-arranges as the user answers questions. The approach outlined here, with a hand-made XML layout was ‘good enough’, it took an acceptable amount of time, and was maintainable. If layouts are required for modules to cover the entire flora of the British Isles, it may be worth investigating auto-generation of layout code. If adaptive lists (e.g. with questions being made invisible if they do not reduce the number of possible answers) are implemented in the future, the layout code may again need to be revisited.

The approach to searching the database used here is simple. The database was taxa x feature variants, thus the simple query of ‘select all where (requested feature) is true’ could be used. This is similar to manual multi-access card-based keys. Unfortunately, it was not capable of any more complicated search features, such as searching based on a range of measurements, or adjusting the list of questions based on the current set of possible results. Improving the database design and search function should be a priority for further work; a more capable search function should allow some of the functions suggested in feedback from users in chapter 7 such as highlighting the ‘best’ questions to answer next.

The initial design for the logic code had a unique method for every function required in the app. This design was an expansion of the code in the MSc app (Bewsey, 2014), where the use of separate methods for each radioButton was based on the tutorial used to create the app. Inclusion of other parts of the layout that were not found in the original app, such as seekBars, followed this separate method approach for simplicity. This is what led to the large and repetitive initial code layout.

The initial code design had a unique method for every possible function required in the app. This design was an expansion of the MSc (Bewsey, 2014) app, where separate methods for each radioButtons was based on the original tutorial. The app was built up by simply writing a new method for every new radioButton required. Other parts of the app, such as seekBars, that were not found in the original app, were also included with separate methods. This easily produced a working app and fairly useable code-base, as a known to be effective and bug free method was simply repeated over and over. As can be seen in Figure 6-3, as the app grew in size, this led to a considerable number of methods in a single class. With calls for an alternative to seekBars, a considerable number of extra radioButtons would have been required, thus requiring many more methods, lengthening the code. It was felt the code was becoming unnecessarily long and very repetitive, thus hard to maintain.

The code re-write aimed to re-use and reduce code wherever possible. Thus, for example, instead of having a unique method for each radioButton, a single method was written that gathered the identity of the radioButton pressed, gathered the required information from the helper classes (all radioButtons added parts to the search query), and perform the required actions. Most processes could be adapted this way, with a single method gathering the required data from a helper class based on the identity of the button pushed. As pop-ups in different parts of the app required different buttons with different actions, they were generally left with bespoke methods. This meant that the amount of logical code (as opposed to data-holding code) was considerably reduced, making the code easier to maintain and understand. As a result, while Figure 6-4 superficially appears to be more complicated than Figure 6-3 (many classes compared to one single, large class), all of the novel helper classes simply act as data stores, with the logic code contained to 'mainActivity' only.

A method to search for all layout parts related to a question was not implemented. This meant that when a function was pressed (e.g. reset question), rather than being able to look up which question the button related to and gather the correct data using this information, buttons needed to look up the required information from their own helper class (e.g. reset question button looks up what components to reset from the reset functions helper class). This meant that several helper classes were required as each question had several separate functions (answer question, help, and reset). With either further improvements to the layout, helper class data storage method, or a method of searching for the correct data, it seems likely that only one helper class of a useful layout would be required. Future work should include an investigation on how to optimise the helper classes while ensuring that further data (and thus questions) can be included as easily as possible.

Despite the re-write of the code, many classes still contain a considerable number of methods, suggesting further optimisations could be made. As already discussed, there are multiple alertDialogue methods; these could be merged if an effective way to generate the required buttons and functionality can be achieved, and it may be possible to re-organise the data held in the various helper-classes into a single helper-class. There may be a considerable number of other improvements, but they were not apparent at the time of re-write. It must be noted that any code refinement and reductions in length must be balanced against code-comprehensibility.

The methods used for the introduction tutorial were placed in the main activity. In a complete version of the app (i.e. one with several modules installed) they should be set to display when the app is first started. This may require either the use of an onboardingFragment, or behaviour similar to what is currently implemented. Use of an onboardingFragment would be more consistent with other apps introducing the functionality of their app. With the length of the longer tutorial, this may not be the most appropriate use of an onboardingFragment and maintaining the current behaviour might be preferable. This decision will need to be taken in the future, but may be more appropriate for the future work in chapter 5.

7 App implementation (part three): testing, evaluation, and incorporation of user feedback.

7.1 Introduction

Testing and evaluating a product are two distinct, but related activities. Testing relates to determining whether functions in a product behave correctly (e.g. does pressing a reset page button cause the page to reset, or does it cause some other behaviour), while evaluating a product relates to whether users find a product appropriate (e.g. is the refresh button in a useful position).

Testing a product is an essential part of software engineering (Farnsworth *et al.*, 2013), with the aim of identifying issues and ensuring the product works as intended (Kauppinen, 2003; Bertolino, 2007; Da Mota Silveira Neto *et al.*, 2011). This testing can encompass many levels of activity, from testing small pieces of code to final user acceptance (Bertolino, 2007; Android Developers, 2018c). This can be achieved either manually or automatically (Bhuarya *et al.*, 2016).

Product evaluation generally requires users to interact with the product (or an analogue of it) in order for the users to provide feedback. There are several different approaches to allow users to interact with the product when evaluating it, including: Usability tests, where users are observed actually using the product; Focus group testing, where users are bought together to discuss a product; Beta testing, where users are provided with near complete forms of the product for them to give feedback on; A/B testing, where users are provided similar versions of the product to help decide the best approach; and Surveys, where users are provided with a list of questions to answer (Babich, 2017). Many of these approaches are explored in Lawrence and Hawthorne (2006). Depending on the exact method used when users are evaluating the product, these approaches can be any of the following types of evaluation: walkthrough evaluation, where users are generally walked through a storyboard, windows navigation diagram, or similar to provide feedback on a system (Dennis *et al.*, 2015); interactive evaluation, where users work with a prototype, generally by going through use scenarios, and provides evaluative feedback to the session operator (Dennis *et al.*, 2015); or formal usability testing, which is similar to interactive evaluation, but instead of a session operator being present, the user is recorded using the prototype (Dennis *et al.*, 2015). If users are not present, heuristic evaluation requires developers to examine the interface by comparing it to a set of pre-defined lists of principles (Dennis *et al.*, 2015).

7.1.1 Different approaches to product testing and evaluation

In their guide to field guide production, Lawrence and Hawthorne (2006) highlight several general factors that need to be carefully considered before these sessions can commence.

- It must be ensured that the method chosen is appropriate for the participants. If the key is for scientific use, using members of the general public are unlikely to understand the technical language; if the key is smartphone based, participants should be capable of smartphone use.
- Participants need to be comfortable and in the correct setting. If the participants feel like they are being examined, rather than the key, then they may modify their answers and induce bias.
- If the setting is completely unlike the intended use setting of the guide (e.g. session ran in a herbarium for an in the field only guide), then the participants may be unable to evaluate the guide correctly.

7.1.1.1 Usability test

Usability tests can be a very generic idea (Rubin and Chisnell, 2008), but are typically where the user is observed in some manner (typically either directly or remotely) using the product (Babich, 2017). In 1994 this was a commonly used approach (Nielsen, 1994).

The observer effect may influence the feedback from usability testing, however, with users behaviour changing from the normal (Sauro, 2017). This is also known as the Hawthorne effect (Fox *et al.*, 2008; Sauro, 2017), and despite debate on the effects existence (O' Sullivan *et al.*, 2004; Kompier, 2006), there are studies that show its existence (Guerin, 1986; McCarney *et al.*, 2007). If the observer is not in the room, care must again be taken to ensure that every required aspect is observed correctly (Babich, 2017).

In Wu, Tombor, Shahab, and West (2017) and Stinson et al. (2010), a 'think aloud' interview was used: participants were asked to verbalise their thoughts while using the app under examination. Wu et al. (2017) found that an anti-smoking app needs to be visually appealing. They (Wu et al. (2017)) considered their study to be limited for two linked reasons: it was a single session examining a specific app, and that the sample size was small. They suggest that testing and evaluating apps needs to be done iteratively. They also suggest two limitations of this style of study. Despite the participant being briefed prior to the session, interviewers may still need to provide input, which may induce bias. Wu et al. (2017) suggest that generalisation beyond the sample group is difficult with this approach, but accept that it has previously been accepted. Stinson et al. (2010) iteratively evaluated an internet based program to support sufferers of Juvenile Idiopathic Arthritis (JIA) using this method. The iterative approach allowed areas inducing user errors to be amended, and allowed participants to suggest how the user interface could be improved. They (Stinson et al. (2010)) again suggest that the small sample size in their study may be a limitation on their results, but highlight that even a single cycle can highly reduce the number of usability problems. The small sample size limited examination of whether different factors (participant age gender and education level, disease severity and duration, or experience with computers) will have affected usability of the program. The final limitation Stinson et al. (2010) highlight is the lack of consensus on evaluation method for health care related apps.

When performing usability tests on plant ID guides, Lawrence and Hawthorne (2006) suggest the most appropriate setting is in the field as this is the realistic product use setting.

7.1.1.2 Beta testing

Beta testing is where users are provided with a near complete version of the product, and are asked to provide feedback (Babich, 2017). Beta tests allow the producer to 'try out' an app, in order to see whether the design works (Dolan and Matthews, 1993; Babich, 2017). It has been well used for over 20 years (Dolan and Matthews, 1993). Care needs to be taken with beta testing to ensure that the users are not frustrated with an incomplete product, otherwise you will receive feedback on bugs, rather than software features (Babich, 2017). Due to the nature of beta tests, sample sizes will be smaller than the full user base (Dolan and Matthews, 1993), and therefore cannot guarantee success (Ozer, 1999).

De La Vega et al. (2014) consider two cycles of beta testing to be enough to develop the product to satisfactory levels, . Licskai, Sands, and Ferrone (2013) used a prolonged beta test (users were provided with access to the app for 81 days) to assist development of an app designed to aid the self-management of asthma. Licskai et al. (2013) noted different limitations in their study, two of which are relevant to general app studies. The first of these was that there was no control group, so bias could not be excluded from the results. The other is that in their study, end users were not involved in the design phase of the app. It is suggested by Licskai et al. (2013) that these limitations should be fixed in future studies.

7.1.1.3 Surveys

Surveys are not as useful for studying user behaviour, you do not get to see user interactions with the product (Babich, 2017). You will also only get answers to the questions you ask, meaning that valuable information may be missed. Milward et al. (2017) for example consider this approach limited in fine-grain detail.

7.1.1.4 AB testing

A/B testing is another term for controlled experiments, where an aspect of a product is changed for a portion of users in order to see how the change affects a specific use area (Lind, 1753; Kohavi *et al.*, 2009; Kohavi and Longbotham, 2015). A form of A/B test is the clinical trial (Bhat, 2017), and clinical trials have been in use since the 1700's (Lind, 1753). It is a standard and very widely used approach to gathering feedback about a product (Lind, 1753; Deng *et al.*, 2016; Miikkulainen *et al.*, 2017), with many large websites using it thousands of times a year on various aspects of their products (Kohavi and Longbotham, 2015). As a result of this wide use, there are many resources describing its use (Kohavi and Longbotham, 2015).

A/B experiments need to be done with care, as you are only testing between 2 variants (Babich, 2017), and thus if other options are preferable to option A or B, they will not be discovered with this method.

7.1.1.5 Focus group testing

Focus group testing typically involves a small group of people, representative of the intended users of the target audience, gathered to provide feedback about a product prior to making the product more widely available (Ozer, 1999; Babich, 2017). They allow individual opinions to be expressed as well as allowing views to be developed and discussed (Milward *et al.*, 2017). These discussions may also stimulate discussion about aspects that would otherwise not have been discussed in interviews (Milward *et al.*, 2017). As suggested by Basch (1987), this technique was discussed by Bogardus in 1926 and mentioned in subsequent years. The target audience for this app is discussed in chapter 8.

Much like beta testing, focus groups may not represent the target audience exactly, and in addition can have the results influenced by the quality of the members and the moderator (Ozer, 1999), meaning the results may not be representative (Babich, 2017). Focus groups should not be used as the only source of research; users may not get the chance to fully explore the product (Babich, 2017).

Milward et al. (2017) used focus group style discussions to investigate user opinions on an alcohol drinking intervention app. 20 participants attended over 3 focus groups. Milward et al. (2017) found that, for this app, several factors were important to the participants. These were: ease of use; visually appealing design; social connectivity; and personalisation. They also noted limitations to this study: the majority (90%) of participants were female, which may have introduced bias; participants were not screened on their intentions to change levels of alcohol use, potentially biasing behaviour while using the app; and that focus groups can be affected by participants saying what they thing the researchers want.

7.1.2 Electronic methods

Firebase is an analytical tool that can be included in Android apps, and allows tracking of many different aspects of the app (Firebase, 2017). Firebase can also easily be used to run tests on the products, such as A/B testing (Bakshi, 2018; Firebase, 2018a), as well as whether the app runs on a very wide variety of Android devices (Firebase, 2018b). Beta tests of a product can be set up via the Google Play Console and distributed to selected users (Android Developers, 2018b).

7.1.3 Mixed methods

Use of more than one method may be appropriate in longer studies.

In development of an app designed to aid self-management of type one diabetes (T1DB), Castensøe-Seidenfaden et al. (2017) employed several of the methods shown above over multiple cycles of user examination. This included: workshops; prototyping; a mail panel; 'think aloud' studies; and a feasibility study. The workshops involved 11-21 participants, and generally involved idea development around pre-determined themes. Workshops and their output were recorded for subsequent analysis. All prototypes were built by an external IT company based on feedback from these workshops and input from diabetes care professionals. For the mail panel, participants were mailed either screenshots of the app or test versions, and questionnaires relating to layout content and functions. The feedback generated was then passed to the IT company to modify the app. There were 6 cycles of the mail panel. 'Think aloud' usability tests were ran in similar approach to Wu et al. (2017) and Stinson et al. (2010). The feasibility study involved health care professionals and prior participants with T1DB being presented with the app, and asked to use it in either work or daily life for 5 weeks. Participants were then presented with a questionnaire to gather final feedback.

Castensøe-Seidenfaden et al. (2017) felt that the mixed method approach provided a highly detailed understanding of the users preferences prior to app design, as well as providing a thorough feedback while refining the app. Where end users participated in designing the app, it was found that users would contribute crucial aspects of the end product. There was, however, enough feedback generated that available resources limited how much could be acted on. Castensøe-Seidenfaden et al. (2017) also comment that there is a lack of guidelines on what data to collect in this session style. The mail panel was found to be more useful on correcting levels and accuracy of app content, while the 'think aloud' sessions were more effective at showing how users explored the app. Combining both sets of feedback provided effective input to improve the app; running these sessions in an iterative cycle showed considerable reductions in issues highlighted between cycles. Feasibility testing showed that 'real life' required some final modifications relating to accessibility while in clinics and social aspects of the app.

While Castensøe-Seidenfaden et al. (2017) highlighted many benefits of the mixed method approach, there were also some limitations. Resource limitations results in participants returning at different stages, potentially biasing results towards individual participants preferences. Resource limitation also resulted in parents being excluded from this study. A lack of available validated questionnaires limited data collection about participant knowledge on health apps. The app was examined over a short period of time, meaning any longer-term effects are not known. The final limitation was that a single author conducted the majority of the tests, potentially biasing the results in an unknown manner.

Despite the number of limitations, Castensøe-Seidenfaden et al. (2017) strongly recommend the use of a mixed methods approach in the future.

7.1.4 Software development lifecycles

Software development lifecycles (SDLC/SDLCs) describe the stages of the development of a system (Ruparelia, 2010; Hijazi *et al.*, 2012; Leau *et al.*, 2012; Dennis *et al.*, 2015). There are a number of different SDLCs (Hijazi *et al.*, 2012; Dennis *et al.*, 2015). Each SDLC follows the same set of major phases, with different methodologies varying the order and number of times each step is taken (Dennis *et al.*, 2015). These phases are: planning, analysis, design, and implementation (Dennis *et al.*, 2015). Different authors expand this list to include different phases: Royce (1970) for example has; System requirements, Software requirements, Analysis, Program Design, Coding, Testing, and Operations. The phases in Dennis et al. (2015) are more generalised, with each of their phases usually encompassing the different phases of other authors. Thus the Dennis et al. (2015) phases are briefly explained and used here.

- Planning. This includes the 'why' stage, where the reason a piece of software is required is expanded upon. For example, the software may make entering data into a database considerably more convenient, saving a company time and therefore money. This stage also includes the initial management stages on how a project should be run.
- Analysis. More in depth information about the system is provided in this stage, particularly relating to details on the intended user group and the situation it will be used in. Thus this stage answers the 'who', 'what', 'where', and 'when' questions. This stage also includes research into whether or not there are any pre-existing systems, and how well they are received.
- Design. This is the 'how' stage. The design of all parts of the system and how it is going to be implemented are decided on at this stage. This typically begins with how the system is going to be developed (e.g. with the companies own programmers or through a third party), and proceeds through increasing levels of specificity, starting at the general architecture of the program, and finishing at the finer details.
- Implementation. This final stage involves the actual creation of the system, installing, and maintenance of the system. This stage is usually the longest as it requires the greatest amount of work.

SDLCs can be placed in several categories, with different authors providing different arrangements. Dennis et al. (2015) suggest three different general categories; structured development, rapid application development, and agile development. These general categories each contain different SDLCs. The previously mentioned testing and evaluation sessions take place at different phases of the SDLC and differing numbers of times depending on the SDLC undertaken.

7.1.4.1 Structured development SDLCs

Structured development methods are logical processes, where one phases of development are only began after the previous ones completion (Dennis *et al.*, 2015). Dennis et al. (2015) include both the waterfall and parallel methods in this category.

The waterfall method is the original methodology (Dennis *et al.*, 2015) and is still in use (Munassar and Govardhan, 2010; Dennis *et al.*, 2015). There are differing opinions on who first described the waterfall model, with Royce (1970) being suggested by Davis and Bersoff (1988) and Hijazi et al. (2012), and Benington (1956) being suggested by Ruparelia (2010). The basic waterfall SDLC, illustrated in Figure 7-1, is where the development process proceeds from one phase to the next, in order, until the product has been developed. The original version of this SDLC did not have any feedback between phases, meaning that errors early in the process would carry over to subsequent stages (Ruparelia, 2010; Hijazi *et al.*, 2012). To reduce the risk of this happening, different versions of the waterfall SDLC provide feedback between the phases to varying levels, or allow phases to overlap or even be repeated (Royce, 1970; Davis and Bersoff, 1988; Munassar and Govardhan, 2010; Ruparelia, 2010; Hijazi *et al.*, 2012; Dennis *et al.*, 2015). There are two main advantages to the waterfall approach; that system requirements are described early in the development process, and that changes to these requirements over the project are minimised (Dennis *et al.*, 2015). There are disadvantages and risks to this development method, however. The requirements and design stages being completed and not revisited before any programming takes place means that if a requirement changes, it is very difficult to go back and accommodate changes (Hijazi *et al.*, 2012; Dennis *et al.*, 2015). As no product is available until the whole process is complete, and as each stage can take a considerable length of time (as each stage discusses the whole product, rather than a part of it), there can be a considerable amount of time between the project commencing and the product being made available to the user (Hijazi *et al.*, 2012; Dennis *et al.*, 2015). There are adaptations of the waterfall approach including but not limited to parallel development (Dennis *et al.*, 2015), the V-shaped method (Munassar and Govardhan, 2010; Ruparelia, 2010; Morris, 2018), and the b-Model (Birrel and Ould, 1988) in (Ruparelia, 2010).

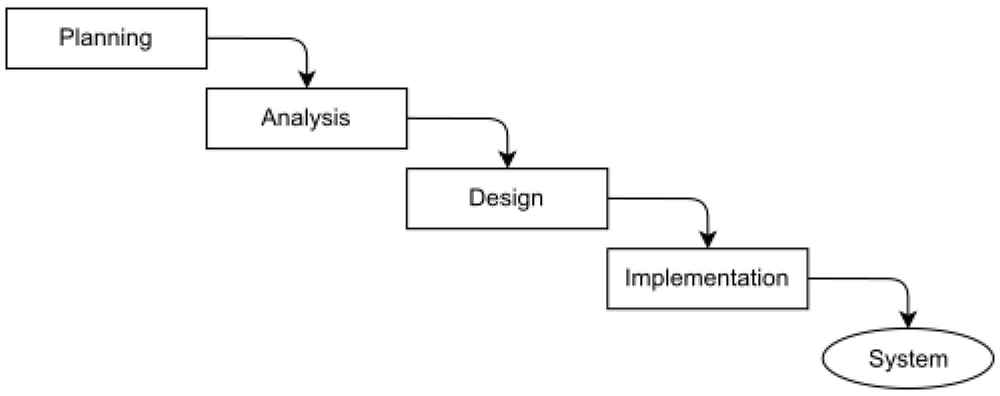


Figure 7-1 - Phases of development in the waterfall software development lifecycle. Redrawn from Dennis et al. (2015).

Parallel development, described in Dennis et al. (2015), adapts the waterfall method by introducing a general design phase, where the project is devided into into 'subprojects'. These subprojects have both design and implementation phase, and are completed simultaneously. An integration phase is introduced after the subprojects have been completed, combining them into the final product. While this approach can save time (as parts of the project are completed simultaneously), not all sub-projects are truly independent, potentially causing difficulty in both completing subprojects and potentially the integration stage. Parallel development is illustrated in figure 1-3 of Dennis et al. (2015).

The V-shaped method (Munassar and Govardhan, 2010; Ruparelia, 2010; Morris, 2018) is a modification and development of the waterfall method. Instead simply requiring linear progress, the V-shaped model defines the process such that stages follow a decompositional progression. I.e. the process begins with the requirements of the system as a whole, and then works down through various stages to individual components. When everything has been defined and designed, the implementation phase works in the reverse; individual components are created first, with the system being completed last. Thus, the two 'arms' of the V are implemented. The other major modification from the waterfall approach is that the tests of each 'level' are generally developed during the design phase, are used to help inform subsequent design stages, and phase is generally designed to interact with the equivalent stage on the other 'arm' of the V.

The b-Model (Birrel and Ould, 1988) in (Ruparelia, 2010) extends the Waterfall model to include a maintenance cycle. This adds a set of phases similar to the initial waterfall, designed to ensure improvement of the software became part of the development lifecycle.

7.1.4.2 Rapid action development (RAD) SDLCs

Rapid action development (RAD) SDLCs maintain the structured nature of waterfall-based SDLCs. The key difference is that instead of a single pass-through of each phase of development, RAD SDLCs are designed to incorporate user feedback about the product, and incorporate it into the next version of the product. This helps resolve some of the biggest drawbacks about structured development methods; the time between a system being requested and users getting to use at least some part of it is reduced, and incorporating the idea of revisions directly into the SDLC means changes to the program (either from changes in the scope of the product, user feedback, or other discoveries during the development process) are less likely to result in large delays and increases in cost. Dennis et al. (2015) include three different SDLCs under the RAD category; phased development, prototyping, and throwaway prototyping.

Under phased development, multiple versions of the product are produced, with each version being fully functional (Dennis *et al.*, 2015). Development of each version works like a self-contained waterfall, with a product produced at the end. The first version typically is designed to only have fundamental features, with subsequent versions amending any faults and introducing extra features based on user feedback. Each cycle of the phased development builds on the previous version of the product, and these cycles continue until the product is complete and development is finished. This SDLC has the advantage over the waterfall approach that a functioning product is delivered to the user sooner than would otherwise occur, and the users gets to see development and changes in the product based on their feedback. The disadvantage, however, is that the initial versions that the user gets to use are incomplete, thus the feedback generated may not initially be accurate.

Prototyping is similar to phased development, in that a basic version of the product is developed and shown to the users in order to gather feedback (Dennis *et al.*, 2015). The same phases as waterfall development are present, the difference is that speed is given a higher priority. Prototyping begins with a planning phase, then continuously loops through the analysis, development, and implementation stages, producing a prototype with each loop. These prototypes allow user interaction so feedback can be generated. Feedback about each prototype is incorporated into the next 'loop'. These loops continue until it is felt that the protoype is functional, and the system is installed. This approach is a step further in getting the users a product sooner, and incorporate their feedback faster. However, with the increasing speed of development, it can become harder to control and maintain the quality of the program being developed, with the potential for significant problems not being spotted until later.

In throwaway prototyping SDLCs, both the planning and initial analysis phases are thorough (Dennis *et al.*, 2015). The initial analysis phase here aims to generate a concept for the system, and accepts that full details of each function of the system may not yet be fully rationalised. Once the initial analysis phase has been completed, a similar loop of analysis, design, and implementation phase begins. These loops generate 'design prototypes', where non-functional but 'close enough' prototype is developed to understand a specific issue. For example, in order to evaluate whether users prefer a nested or indented key, the key itself may be relatively trivial (leading to preferred breakfast items, say), but would fully implement the two design options. Once all issues have been resolved, and a complete idea of the product can be built based on the design prototypes, the final product can be developed and implemented. Throwaway prototyping can take longer than normal prototyping SDLCs, but is likely to result in a more stable product.

7.1.4.3 Agile

Agile development methods aim to streamline the development process (Dennis *et al.*, 2015). They generally break development of the project down into smaller, faster cycles (Ruparelia, 2010; Dennis *et al.*, 2015; Morris, 2018). These cycles allow for incremental development of the product (Leau *et al.*, 2012; Kumar and Bhatia, 2014; Dennis *et al.*, 2015; Morris, 2018; Virkus, 2019). In Agile SDLCs, acceptance and adaptation to change in the product, caused either by changing customer requirements, or more generally as a result of the development process, are given a very strong emphasis (Leau *et al.*, 2012; Dennis *et al.*, 2015). The customer is generally involved early and continuously in the development process (Morris, 2018; Virkus, 2019).

Agile SDLCs differ from RAD SDLCs. Instead of formal cycles of development and testing (as in phased development), agile SDLCs work on an effectively continuous bases of modifying the product. A fully working product is produced and modified, unlike in the various prototyping SDLCs where incomplete (and not necessarily functional) prototypes of the product are shown to the clients at regular intervals, but a fully working product is not necessarily available from the beginning.

Agile SDLCs main advantages are the continuous development and involvement of the customer. Revisions and updates to the product are built into the SDLC with the iterative development, so such changes are never going to be as expensive and time consuming compared to more linear (e.g. waterfall) SDLCs (Leau *et al.*, 2012). Agile methods also allow for development of the product to begin at an earlier stage, before the customer has fully defined what they want from the product; changes or refinements are built into the SDLC (Leau *et al.*, 2012). The Agile Manifesto lists 12 principles (Agile Manifesto, 2019), which Leau et al. (2012) condense to:

- Early customer involvement
- Iterative development
- Self-organising teams
- Adaptations to change

Many sources detail that agile SDLCs are not necessarily a panacea, and that caution and planning are needed if they are used. If development of part of the system is outsourced in any manner, the co-location of development teams and interaction with the client required in agile SDLCs becomes considerably more difficult (Dennis *et al.*, 2015). Agile SDLCs require careful managing in order to avoid 'programmers gone wild' development, but this does not necessarily fit with the definition of agile development (Dennis *et al.*, 2015). The increased speed and iterative nature of the development, along with the SDLC design putting less emphasis on it means that the amount of documentation produced in agile workflows is considerably reduced compared to other

SDLCs, which may lead to difficulty in the future (Leau *et al.*, 2012; Dennis *et al.*, 2015). The number of meetings and short cycles of development can cause the developers considerable stress (Leau *et al.*, 2012). Changes in the requirements of one module may cause a considerable number of changes in other modules to be required if modules are strongly dependent on one another, slowing development (Leau *et al.*, 2012). The requirement for client interaction can cause difficulty if there are a large number of clients, or if a developers 'have poor social skills'(Leau *et al.*, 2012) and are thus unable to interact with the clients to gather feedback successfully (Ruparelia, 2010; Leau *et al.*, 2012; Virkus, 2019).

There are a number of different agile development methods, including but not limited to: extreme programming (Ruparelia, 2010; Kumar and Bhatia, 2014; Dennis *et al.*, 2015), Scrum (Ruparelia, 2010; Kumar and Bhatia, 2014; Dennis *et al.*, 2015; Virkus, 2019), and Lean (Ruparelia, 2010; Virkus, 2019).

Scrum and extreme programing are similar (Cohn, 2007; Visual Pradigm, 2019), but there are a number of differences. Both work in iterations, but extreme programming's iterations are generally one to two weeks, while Scrum's last three to four (Cohn, 2007; Visual Pradigm, 2019). Within each iteration, Scrum does not allow changes to the requirements of the program, while extreme programming does (Cohn, 2007; Bowes, 2015; Visual Pradigm, 2019). There are no Scrum team leaders, instead teams organise themselves (Dennis *et al.*, 2015). In extreme programming the priority and order of which features are developed is set by the customer, with the development team following this (Cohn, 2007; Bowes, 2015; Visual Pradigm, 2019). The customer still indicates the priority in Scrum development, but the team may choose a different development order (Cohn, 2007; Bowes, 2015; Visual Pradigm, 2019). Extreme programming prescribes a set of engineering practices, unlike Scrum, which include but are not limited to test driven development, simple design, and close interaction with the end users (Cohn, 2007; Bowes, 2015; Dennis *et al.*, 2015; Visual Pradigm, 2019). Lean development is less specifically an SDLC, instead is the principle of delivering the minimum viable product fast, working on the theory that less of a product now is better than more of a product later (Cohn, 2007; Visual Pradigm, 2019).

7.2 Aims

The aim of the test and evaluation sessions described in this chapter is to improve the design of the app and the quality of the database. These sessions will also aim to establish whether users prefer the app over the paper-based ID guides.

7.3 Method

7.3.1 Ongoing testing

The app was tested (i.e. determining whether the correct effect followed the cause; for example when pressing a button, did the expected response happen, or did the app crash) in a continuous manner, usually after the code had been modified. If the modification was a new set of buttons, for example, this continuous testing involved pressing each of the buttons in order (scrolling through the list) and seeing if the button performed the correct function. Other ongoing continuous testing involved 'playing' with the app at opportune moments, either casually simulating using the app (e.g. answering questions haphazardly without any known good data or specimen to gather data from) and seeing if everything worked, or pressing unexpected combinations buttons on the screen to see if unusual behaviour could be solicited.

7.3.2 SDLC used

Opportunities for co-ordinated sessions with users who were relatively unfamiliar with the product yet well suited to test and evaluate the app were rare, with three key opportunities presenting themselves during the project, and a fourth after. As these opportunities were rare, it was considered more useful for a fully working key to be tested and evaluated at these opportunities, thus all features present in the app could be scrutinised as many times as possible. These co-ordinated sessions were therefore designed to accommodate both testing and evaluation. Thus, a phased development SDLC was employed, with the feedback from these key sessions being analysed, the app modified and updated, and the next session being ran. The cycle is shown in Figure 7-2.

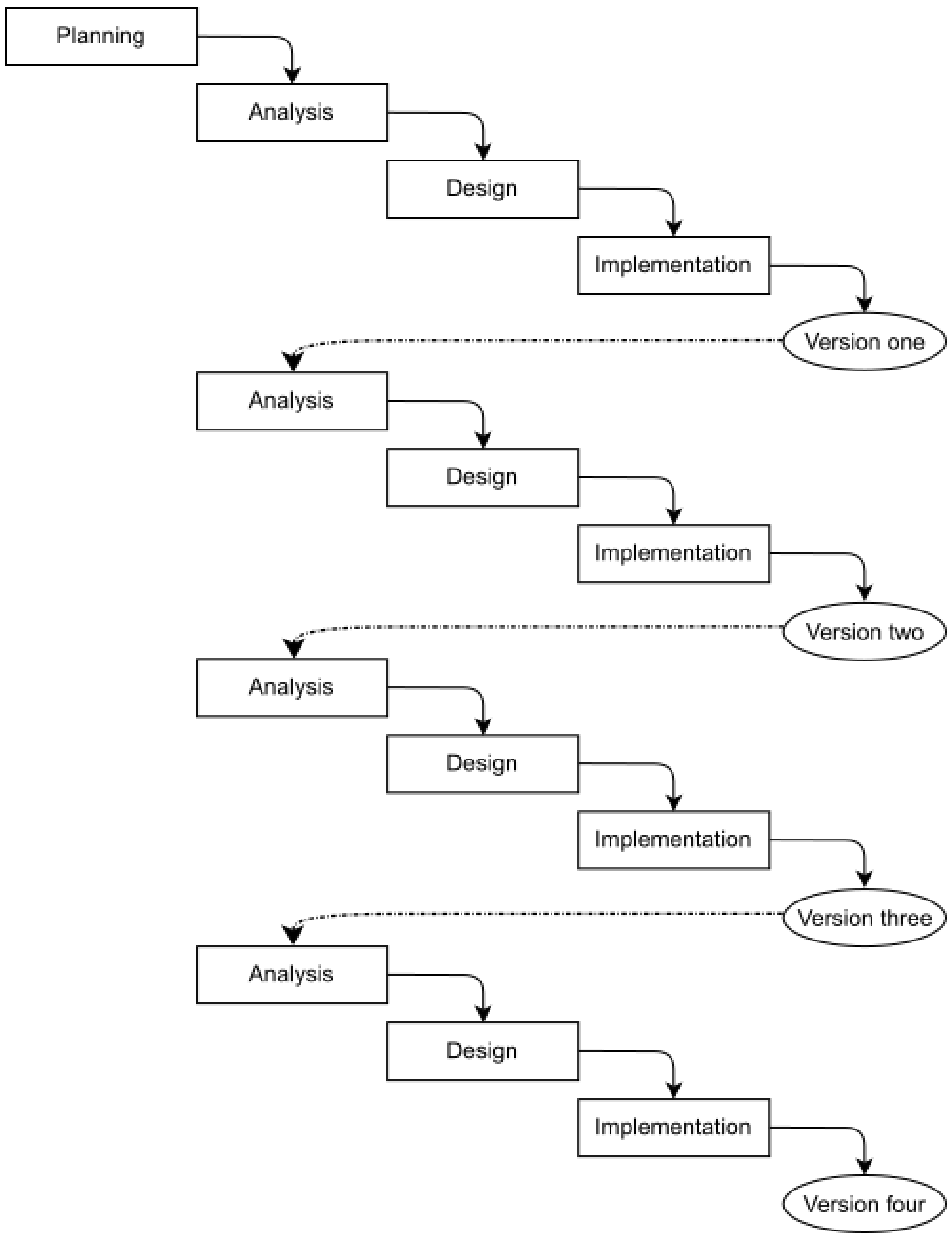


Figure 7-2 - Phased software development cycle used here. Redrawn and adapted from Dennis et al. (2015)

7.3.3 General method for sessions

These key sessions were set up as hybrid focus group/usability tests. Volunteers were provided with access to the app, one or many paper-based ID guides, and fresh or herbarium *Equisetum* specimens. Volunteers were typically split into small groups and asked to use the paper-based ID guide and the app to identify various specimens. All sessions were set up as interactive evaluation sessions, and modified to accommodate many-to-one, rather than one-to-one interaction between volunteers and coordinator. For lists of specimens used in each session, see sections 7.3.5 and G.3.1 .

7.3.4 Session one – design students

The session was set up as a focus group. This session had Dr Alison Black, two Design and Typography students at the University of Reading supervised by Dr Alison Black, Dr Alison Black, and one coordinator (the author). The students were provided with the app a few days before the feedback session.

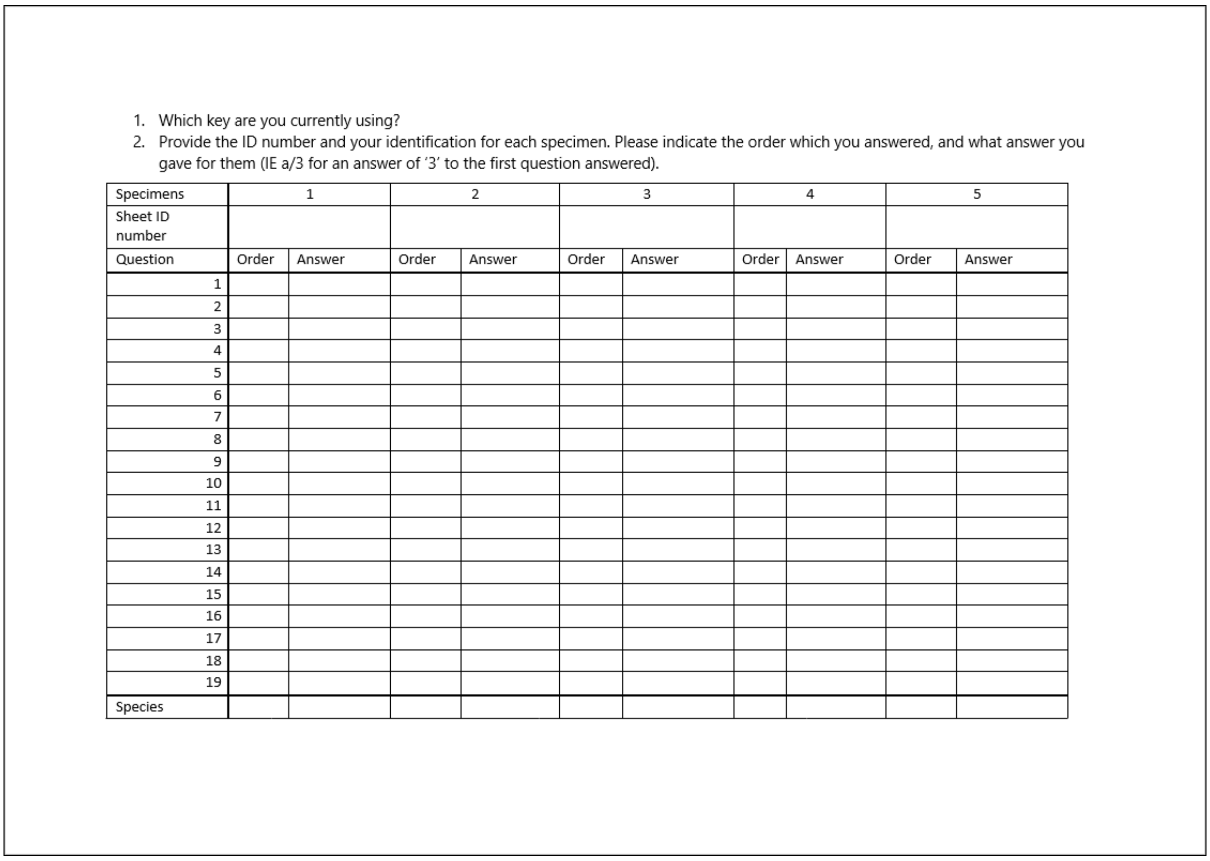
At the session, a copy of the New Flora to the British Isles (Stace, 2010c) was provided, along with fresh *Equisetum* L. specimens retrieved from the Whiteknights campus of the University of Reading. During the session, the volunteers and Dr Black attempted to identify the specimens using both the app and the copy of the New Flora to the British Isles (Stace, 2010c). All volunteers used both keys, but keys were used separately (i.e. one key for one specimen). Feedback was then gathered as part of a discussion focusing on evaluating the app.

7.3.5 Session two – herbarium volunteers

This session had Dr Alastair Culham (**RNG** herbarium curator), 12 **RNG** herbarium volunteers (all students at the University of Reading), and one coordinator (the author).

The volunteers were provided with either a method to download the most recent version of the app to their personal Android device, or a device running the most recent version of the app. The app was set to use the smaller database (see sections 4.3.1.2 and 4.4). They were also provided with a copy of the New Flora of the British Isles (Stace, 1997, 2010c). Herbarium specimens from five taxa in the database were selected from available specimens with their label covered by a number. Specimens selected for *E. x litorale* Kühlew. ex Rupr. were from **RNG**, and were collected by E.C. Wallace (29/08/1953, 26/08/1951, and 30/08/1947) and J.E.Lousley (11/7/1937 and 12/9/1942). Specimens selected for *E. arvense* L. were from **RNG** and were collected by E.C. Wallace (2/7/1930 and 17/8/1935), J. Donald Grose (10/6/1939), and F.S.E (29/5/1927, and 8/7/1929). Specimens selected for *E. telmateia* Ehrh. were from **RNG** and were collected by E.C.Wallace on 31/08/1941, 20/08/1939, 24/06/1933, 24/05/1920, and 24/06/1950.

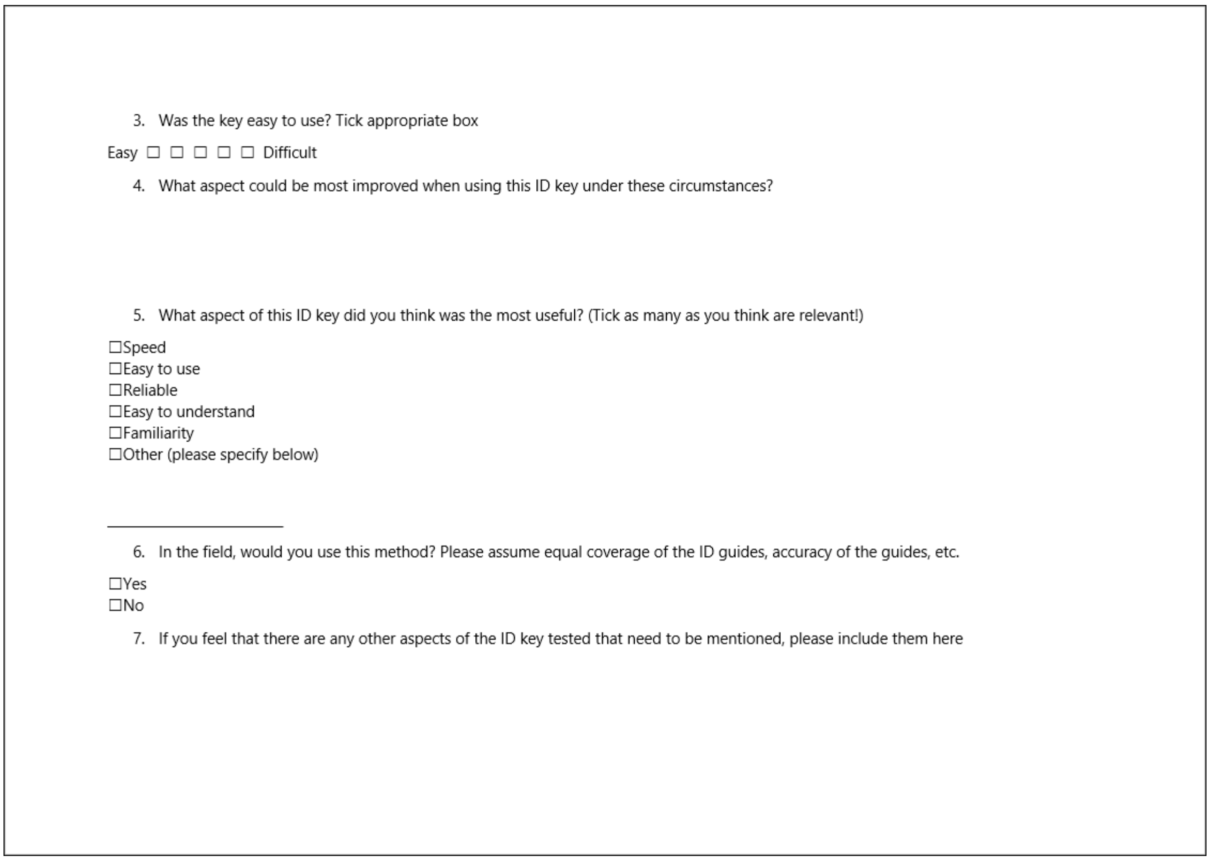
Specimens of *E. x mchaffieae* C.N. Page were from **E** and are object numbers 00664537, 00664533, 00664534, 00664535, and 00664536. Specimens of *E. x robertsii* Dines were from **E** and are object numbers 00834289, 00834288, 00834287, 00834286, and 00834285. Volunteers used both the app and copies of the New Flora of the British Isles (Stace, 1997, 2010c) to identify these specimens. Volunteers were asked to choose one specimen from each taxon (thus five total) and identify them with either the app or the New Flora of the British Isles (Stace, 1997, 2010c). Volunteers were then asked to fill in a questionnaire (see Figure 7-3 and Figure 7-4 for this questionnaire), answering based on their experience using the key. The volunteers then repeated this, choosing 5 other specimens and the other key. This questionnaire was designed to gather both test and evaluative feedback. Feedback was also gathered from direct discussion with the volunteers during the session.



1. Which key are you currently using?
2. Provide the ID number and your identification for each specimen. Please indicate the order which you answered, and what answer you gave for them (IE a/3 for an answer of '3' to the first question answered).

Specimens	1		2		3		4		5	
Sheet ID number										
Question	Order	Answer	Order	Answer	Order	Answer	Order	Answer	Order	Answer
1										
2										
3										
4										
5										
6										
7										
8										
9										
10										
11										
12										
13										
14										
15										
16										
17										
18										
19										
Species										

Figure 7-3 – Page one of the questionnaire used in session two



3. Was the key easy to use? Tick appropriate box

Easy ☐ ☐ ☐ ☐ ☐ Difficult

4. What aspect could be most improved when using this ID key under these circumstances?

5. What aspect of this ID key did you think was the most useful? (Tick as many as you think are relevant!)

☐Speed
☐Easy to use
☐Reliable
☐Easy to understand
☐Familiarity
☐Other (please specify below)

____

6. In the field, would you use this method? Please assume equal coverage of the ID guides, accuracy of the guides, etc.

☐Yes
☐No

7. If you feel that there are any other aspects of the ID key tested that need to be mentioned, please include them here

Figure 7-4 - Page two of the questionaire used in session two

7.3.6 Session three – herbarium volunteers

After feedback from session two was examined and incorporated into the app, another session with the same volunteers was scheduled. The previous session was modified such that volunteers could access the larger database on the app, a slightly modified version of the questionnaire was provided (see appendix G.3.2), volunteers were asked to identify fewer specimens from a wider range of taxa, and volunteers were able to use their preferred paper-based ID key. For the list of specimens available, see appendix G.3.1 . Two PhD students under Dr Culham volunteered for this session. Only six **RNG** volunteers were available. Dr Culham was again available for the session, and one coordinator (the author) was present.

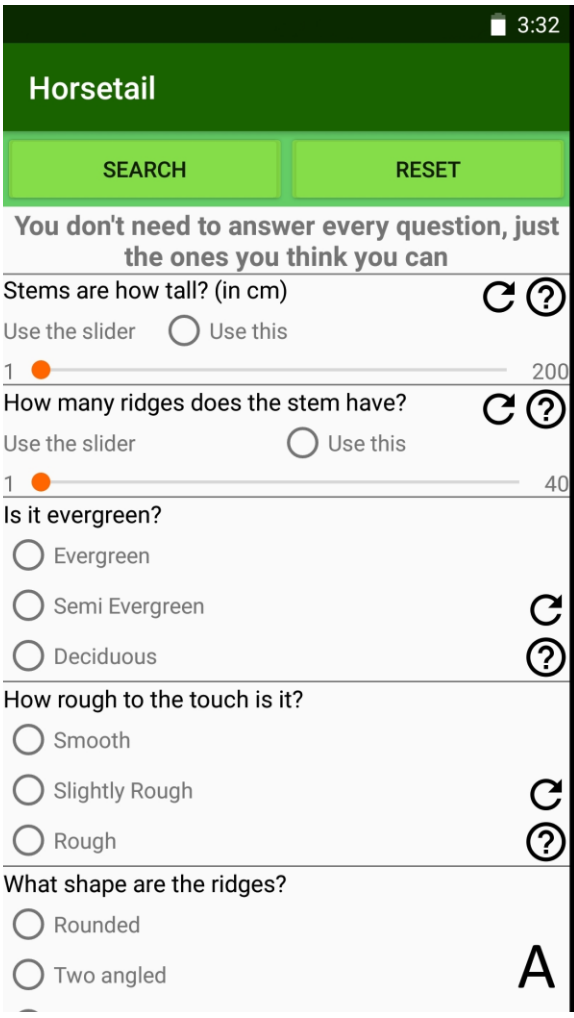
7.3.7 Session four – experienced examiners

The additional fourth session happened during the PhD viva and followed a similar 'focus group' method to the first session, with the app being made available to the users prior to the session. Differences from the first session were that herbarium specimens were provided instead of live specimens, no copy of the New Flora of the British Isles (Stace, 2010c) was provided, and the users were 'experienced botanists', rather than design students. While this was not a scheduled evaluation, the feedback was valuable and key to some design changes.

7.4 Results

7.4.1 General

As a result of the four sessions, the design of the app was updated. Figure 7-5 shows how the design of the search page was changed in response to the feedback. Figure 7-6 shows how the design of the results was changed in response to the feedback.


3:32

Horsetail

SEARCH RESET

You don't need to answer every question, just the ones you think you can

Stems are how tall? (in cm)

Use the slider ◯ Use this

1 200

How many ridges does the stem have?

Use the slider ◯ Use this

1 40

Is it evergreen?

◯ Evergreen

◯ Semi Evergreen

◯ Deciduous

How rough to the touch is it?

◯ Smooth

◯ Slightly Rough

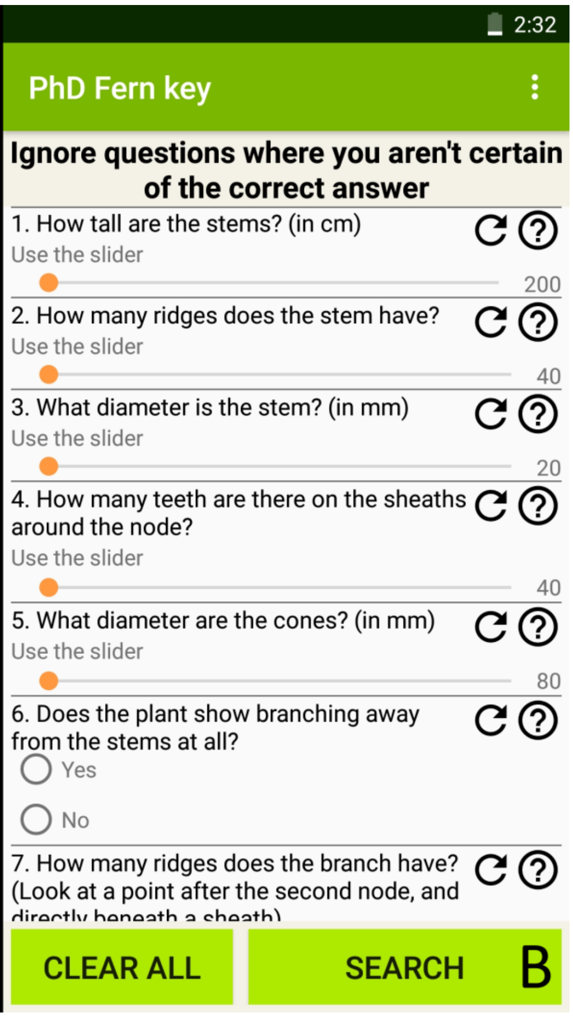
◯ Rough

What shape are the ridges?

◯ Rounded

◯ Two angled

A


2:32

PhD Fern key

Ignore questions where you aren't certain of the correct answer

1. How tall are the stems? (in cm)

Use the slider

200

2. How many ridges does the stem have?

Use the slider

40

3. What diameter is the stem? (in mm)

Use the slider

20

4. How many teeth are there on the sheaths around the node?

Use the slider

40

5. What diameter are the cones? (in mm)

Use the slider

80

6. Does the plant show branching away from the stems at all?

◯ Yes

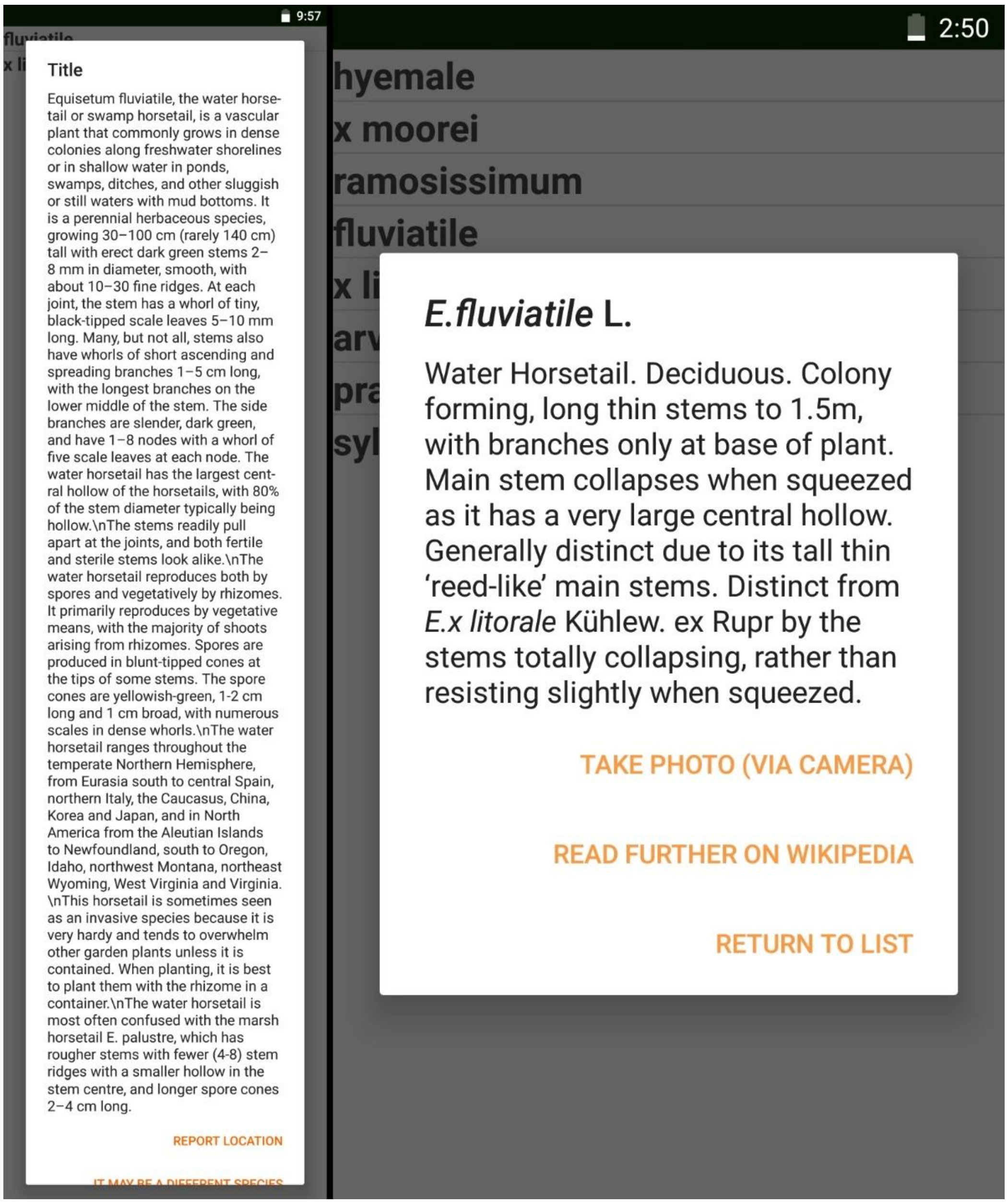
◯ No

7. How many ridges does the branch have? (Look at a point after the second node, and directly beneath a sheath)

CLEAR ALL SEARCH

B

Figure 7-5 - The multi-access search page of the app. A shows the design during the first session. B shows the updated design.


9:57

fluviatile

x li

Title

Equisetum fluviatile, the water horsetail or swamp horsetail, is a vascular plant that commonly grows in dense colonies along freshwater shorelines or in shallow water in ponds, swamps, ditches, and other sluggish or still waters with mud bottoms. It is a perennial herbaceous species, growing 30–100 cm (rarely 140 cm) tall with erect dark green stems 2–8 mm in diameter, smooth, with about 10–30 fine ridges. At each joint, the stem has a whorl of tiny, black-tipped scale leaves 5–10 mm long. Many, but not all, stems also have whorls of short ascending and spreading branches 1–5 cm long, with the longest branches on the lower middle of the stem. The side branches are slender, dark green, and have 1–8 nodes with a whorl of five scale leaves at each node. The water horsetail has the largest central hollow of the horsetails, with 80% of the stem diameter typically being hollow.\nThe stems readily pull apart at the joints, and both fertile and sterile stems look alike.\nThe water horsetail reproduces both by spores and vegetatively by rhizomes. It primarily reproduces by vegetative means, with the majority of shoots arising from rhizomes. Spores are produced in blunt-tipped cones at the tips of some stems. The spore cones are yellowish-green, 1-2 cm long and 1 cm broad, with numerous scales in dense whorls.\nThe water horsetail ranges throughout the temperate Northern Hemisphere, from Eurasia south to central Spain, northern Italy, the Caucasus, China, Korea and Japan, and in North America from the Aleutian Islands to Newfoundland, south to Oregon, Idaho, northwest Montana, northeast Wyoming, West Virginia and Virginia.\nThis horsetail is sometimes seen as an invasive species because it is very hardy and tends to overwhelm other garden plants unless it is contained. When planting, it is best to plant them with the rhizome in a container.\nThe water horsetail is most often confused with the marsh horsetail E. palustre, which has rougher stems with fewer (4-8) stem ridges with a smaller hollow in the stem centre, and longer spore cones 2–4 cm long.

REPORT LOCATION

IT MAY BE A DIFFERENT SPECIES

2:50

hyemale

x moorei

ramosissimum

fluviatile

x li

arv

pra

syl

E.fluviatile L.

Water Horsetail. Deciduous. Colony forming, long thin stems to 1.5m, with branches only at base of plant. Main stem collapses when squeezed as it has a very large central hollow. Generally distinct due to its tall thin 'reed-like' main stems. Distinct from *E.x litorale* Kühlew. ex Rupr by the stems totally collapsing, rather than resisting slightly when squeezed.

TAKE PHOTO (VIA CAMERA)

READ FURTHER ON WIKIPEDIA

RETURN TO LIST

Figure 7-6 - The information provided when you press on a result. A shows the initial full-length first paragraph provided via Wikipedia; the design during the first session. B shows a pre-written short piece of diagnostic text stored locally on the device (thus not requiring internet connectivity); the updated design. Note that A has been shrunk in resolution to fit in this figure. Note also that in A, the '\n' characters were part of the data supplied through the Wikipedia API, and the app had not been programmed to remove them.

Accuracy of identification is shown in Table 7-1. Accuracy was only measured in sessions two and three. Accuracy was measured by first checking whether the specimen was present in the key used at the time. If it was and the volunteers put the correct identification down as either of their answers (volunteers only put down two possibilities as a maximum, this behaviour was unprompted), then this was marked as an accurate identification as it was assumed that users would then be able to decide between two results using further information about the two taxa. The percentage accuracy was then based on the number of attempted identifications and correct identifications.

Table 7-1 - Accuracy rates of identifications from the app and from paper based keys during sessions two and three.

Session	App accuracy	Paper based accuracy
1	33.3%	0%
2	7.6%	30%

The questionnaires included questions asking: 1, whether the user would use the app in the field and whether they would use the paper-based key in the field; and 2, whether they found the app and the paper based key easy to use (on a scale of 1(easy) to 5(difficult)). The results of these questions are shown in Table 7-2.

Table 7-2 - Reponses to questions on whether users preferred the app or a paper-based key. Users were asked whether or not they would use the key in the field, with the aggregated score representing the number of users who said they would. Users were also asked whether they found the key easy to use on a scale of 1(easy) to 5(difficult)

Question	App	Paper-based key
Would you use the key in the field?	100%	60%
Was the key easy to use? (Lower is better)	1.8/5	3.6/5

The final version of the app is provided as an .APK file as Appendix A on the included disc.

As a result of the changes implemented to the app in response to the feedback generated through the testing and evaluation sessions, the windows navigation, use case, and storyboard diagram from chapter 5 can be updated. The new versions of these diagrams are shown in Figure 7-7, Figure 7-8 and Figure 7-9. The major changes in these diagrams are the inclusion of the ability to toggle the design of questions, on-device diagnostic information about taxa, and the additional option to take a photo being presented to the user when they are viewing the diagnostic information about a taxon.

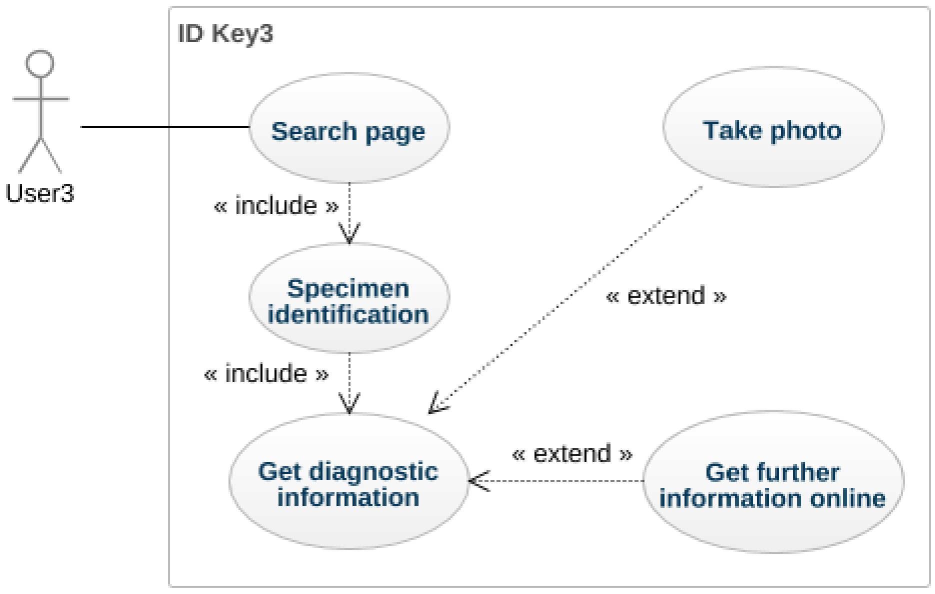


Figure 7-7 - Use case diagram after updates and modifications to the app based on feedback. Diagram made using GenMyModel (2019).

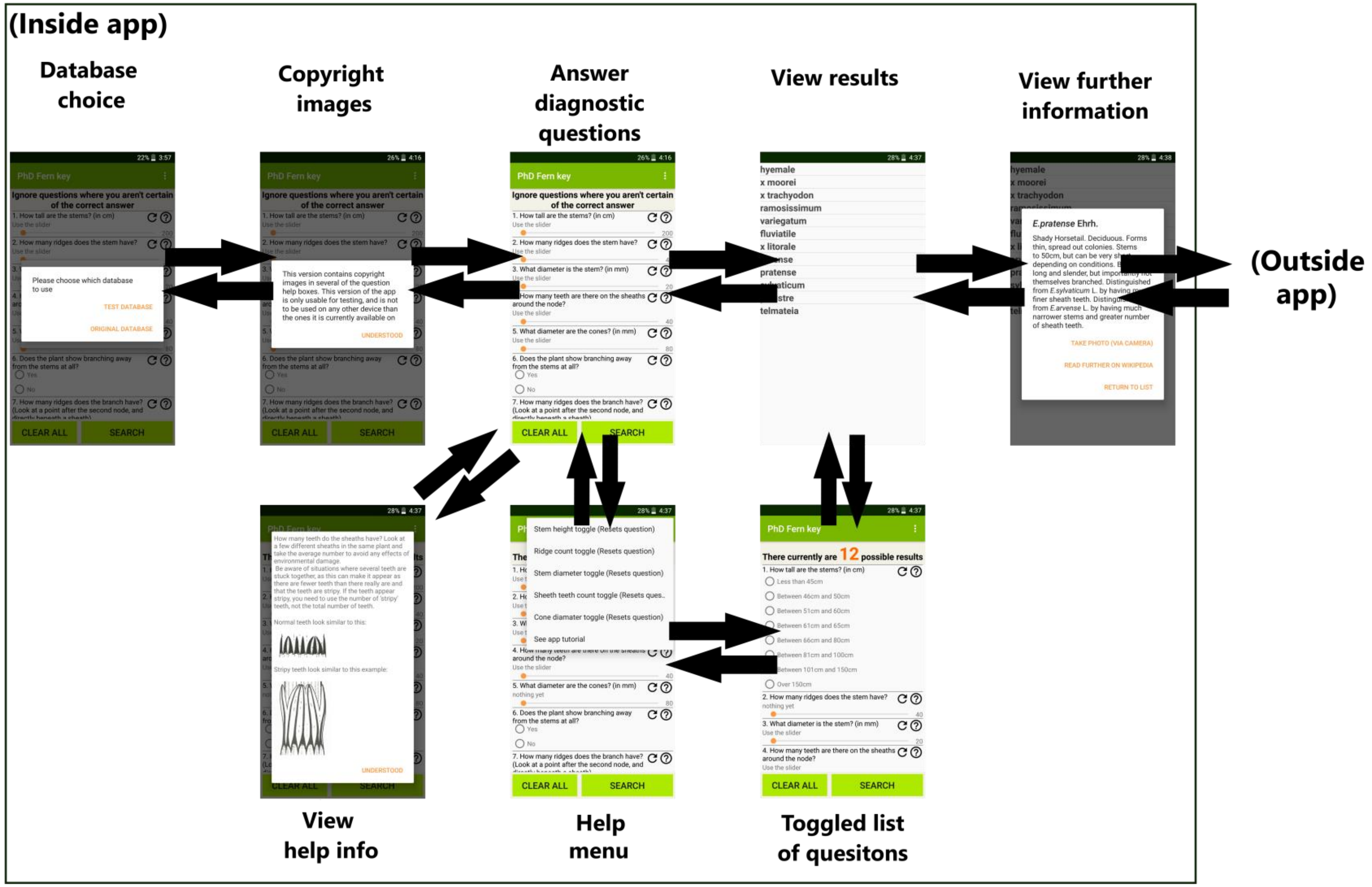

(Inside app)
Database choice
Copyright images
Answer diagnostic questions
View results
View further information
(Outside app)
View help info
Help menu
Toggled list of quesitons
Please choose which database to use
TEST DATABASE
ORIGINAL DATABASE
This version contains copyright images in several of the question help boxes. This version of the app is only usable for testing, and is not to be used on any other device than the ones it is currently available on
UNDERSTOOD
PhD Fern key
Ignore questions where you aren't certain of the correct answer
CLEAR ALL
SEARCH
hyemale
x moorei
x trachyodon
ramosissimum
variegatum
fluviatile
x litorale
pratense
telmateia
E.pratense Ehrh.
TAKE PHOTO (VIA CAMERA)
READ FURTHER ON WIKIPEDIA
RETURN TO LIST
Stem height toggle (Resets question)
Ridge count toggle (Resets question)
Stem diameter toggle (Resets question)
Sheeth teeth count toggle (Resets ques...
Cone diamater toggle (Resets question)
See app tutorial
There currently are 12 possible results

Figure 7-8 - Storyboard navigation diagram of the app after changes based on feedback

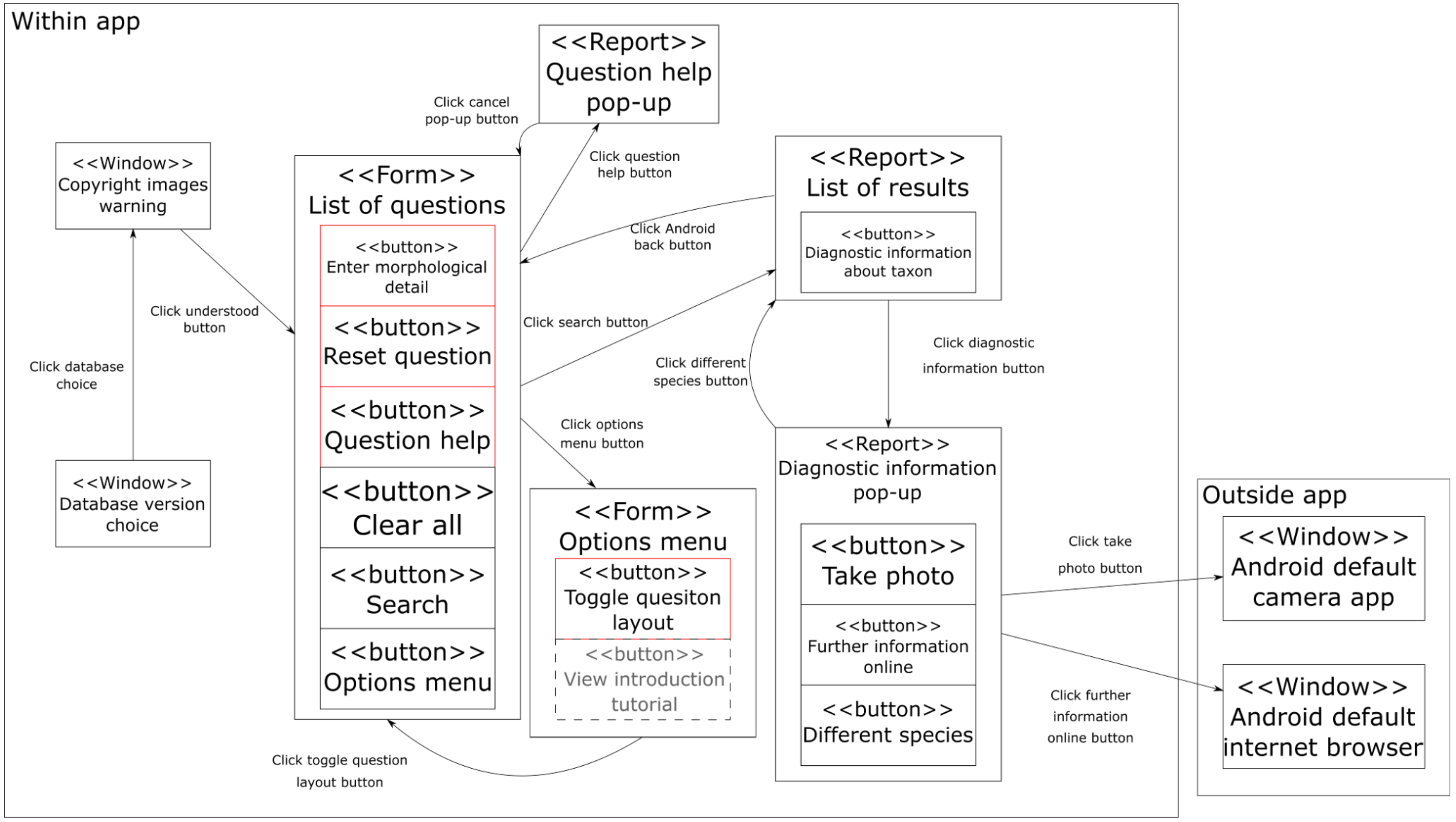

Within app
<<Report>> Question help pop-up
Click cancel pop-up button
Click question help button
<<Window>> Copyright images warning
<<Form>> List of questions
<<button>> Enter morphological detail
<<button>> Reset question
<<button>> Question help
<<button>> Clear all
<<button>> Search
<<button>> Options menu
<<Report>> List of results
<<button>> Diagnostic information about taxon
Click Android back button
Click understood button
Click database choice
Click search button
Click different species button
Click diagnostic information button
Click options menu button
<<Window>> Database version choice
<<Form>> Options menu
<<button>> Toggle quesiton layout
<<button>> View introduction tutorial
<<Report>> Diagnostic information pop-up
<<button>> Take photo
<<button>> Further information online
<<button>> Different species
Click take photo button
Click further information online button
Click toggle question layout button
Outside app
<<Window>> Android default camera app
<<Window>> Android default internet browser

Figure 7-9 – Windows navigation after updates to the app based on feedback. Note that red outlines mean the item is repeated as required for the design and isn't shown here, grey dashed outlines show items that are in the version of the app in Appendix A , but would be in an alternative location in a published version of the app.

7.4.2 Individual sessions

The individual answers to questionnaires and feedback generated during sessions are shown in Appendix G . Nearly all feedback generated in all the sessions is evaluative, as the continuous testing had caught most programmatic issues. The only significant testing feedback generated during these sessions was that modifications to the app required the minimum required version of Android to be increased between sessions one and two. This unfortunately meant that the LG Gpad (GSMArena, 2016) was no longer capable of running the app.

Feedback generally fell into four general categories:

- single point problems, such as typos
- suspected data errors, which were corrected
- Missing or poorly formatted information, for example the suggestion that stem height should indicate measuring from ground level resulted in a re-evaluation of all help texts
- app behavioural changes, which were considered, and generally acted on

Single point issues were corrected as they were highlighted in the feedback. Data errors were investigated, and if it was found to be a genuine data error was corrected. Missing or poorly formatted information presented to the user was amended as highlighted in the feedback, and general lead to a review of all similar bits of the app. Behavioural changes were generally acted on if the same feedback was repeated by either more than one group of volunteers, or if it was repeated over several sessions.

7.4.2.1 Session one – design students

User interface (UI) close to the edge of the display can appear as if the UI is 'running over the edge' and hidden from the user. It can also make interactive elements of the UI harder to use. Thus, a more prominent gap between the UI elements and the edge of the screen was introduced. The total length of each seekBar was reduced slightly thus reaching the end was more comfortable. Seekbars with many possible values still had 'tricky' behaviour due to the number of possible values making it hard to select the desired value. When using the seekBars, the UI element showing the number currently selected was frequently lost under the users hand. The behaviour was changed so that during use the number was increased in size and made blue, raising the content awareness of this UI element, and shrinking to the same size as the question text after the user stops interacting with the seekBar. This ensured that the value was still more visible after use; but helped the overall list of questions remain short. This new behaviour is shown in Figure 7-10. The amount of information retrieved from Wikiepdia for the click-through extra information box when the user presses on a potential identification in the list of results was reduced from the first paragraph of the relevant article to only the first sentence. The line spacing in these boxes was also increased. It was indicated that these boxes could be very long and very dense. An example of the improved further information pop-up is shown in Figure 7-6 B. The first sentence of a taxon's Wikipedia article is often not enough to confirm an identity, so this length of text should be used as a guide for hand-written diagnostic texts.

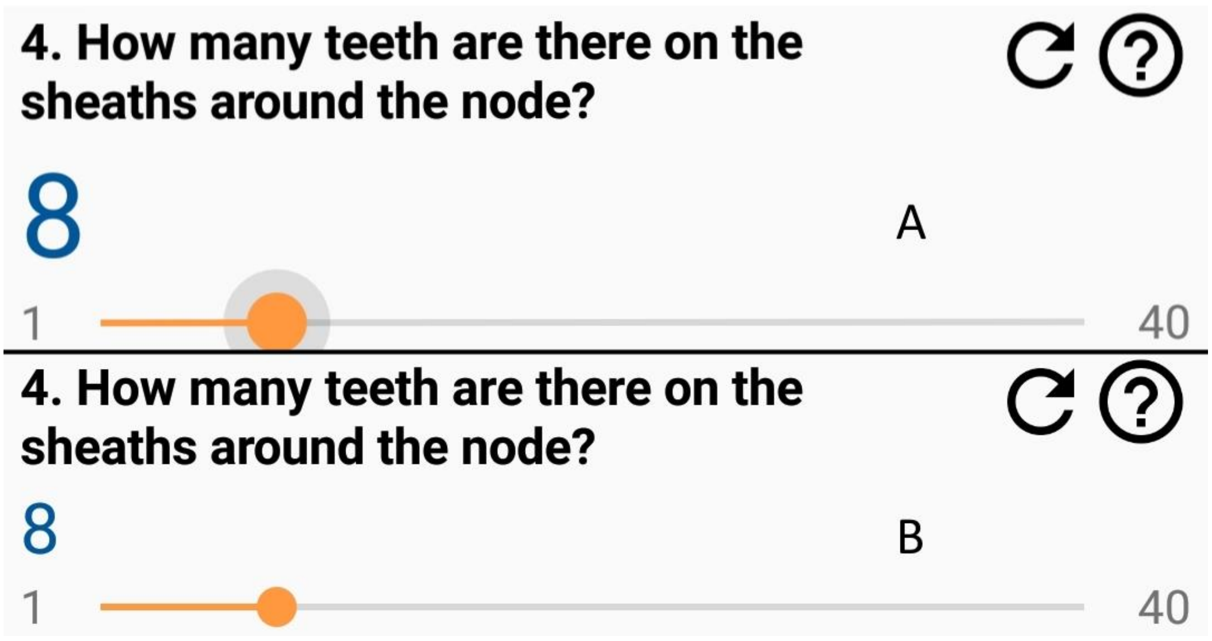

4. How many teeth are there on the sheaths around the node?
8
A
1
40
4. How many teeth are there on the sheaths around the node?
8
B
1
40

Figure 7-10 - The display of the current value of a seekBar. Shown during use (A) where the value is larger, and after the value has been selected (B) where the number has shrunk to save room.

Other adaptations were made to improve the behaviour and comprehend-ability of the search page. Some questions were considered 'self-evident', with the tap-through help box providing no further information. For these questions, these help boxes were simply removed. This removed a potential source of confusion where users may think that some of the help box had not loaded, or that there was further information in the help box that they had not understood. The questions were numbered in order, and questions where information was either hard to gather or confirm (e.g. central hollow to total stem diameter ratio) were removed. This provided the users with a shorter list to navigate, and reference points for the users location in the list.

To ensure that each component of the 'list of questions' screen was visually distinct and that the intended functionality of each component was clear, changes were made to the layout and colour scheme of this page. The 'reset' and 'search' buttons were moved to the bottom of the page, as shown in Figure 7-5, with the 'reset' button being changed to say 'clear all'. This made it clearer that the user was resetting the currently answered questions to an unanswered state, and put both buttons in a more natural position after the list of questions. The design of these two buttons was also improved to ensure that they were clearly buttons. Rather than only modifying the background colour for the 'results left' text, the colour scheme of the whole app was updated, improving the overall aesthetics. This new colour scheme can be seen in Figure 7-5B. In the new scheme, the background to the 'results left' text and the 'clear all' and 'search' buttons are the same, highlighting that they are not part of the list of questions, instead a separate UI component.

As part of making better use of the tools available on smartphones, an additional button opening the device's default camera was added to the extra information boxes when the user clicks on a potential identification. This allowed the users to photograph and keep a visual record of specimens they have identified.

Finally, the exact wording of the text and design was updated to correct typos and irregularities in font size. A number of different areas of feedback were considered, but were not acted on. Despite suggestions that the images used in click-through help boxes could be improved (for example with scale bars, being redrawn to make them more schematic, or simply having images in more boxes), no changes were made. The images were taken from Page (1982) with the intention to use these as a guide for the quality and style required, then replace them with non-copyright versions in the future. Scale bars were not included in the images from Page (1982); there would be no way to ensure accuracy. The non-copyright versions would include scale bars wherever possible. The list of questions was a static layout, so re-arranging on the fly to have the most diagnostic questions always at the top of the list would have required a complete re-development of the search page layout and behaviour, and the data returned by the search function. The behaviour of the search function also meant that it was not possible to give probabilities to results. The questions were not re-organised to show the easiest or most frequently answered at the top of the list. It was not clear which questions were easiest to answer, and there was not enough information gathered about the popularity of different questions to indicate which should go at the top of the list. It was felt that having questions about similar aspects of morphology closer to each other in the list was easier to understand. The design of the results page also meant it was not clear how to show information about all the results if sufficiently few remained.

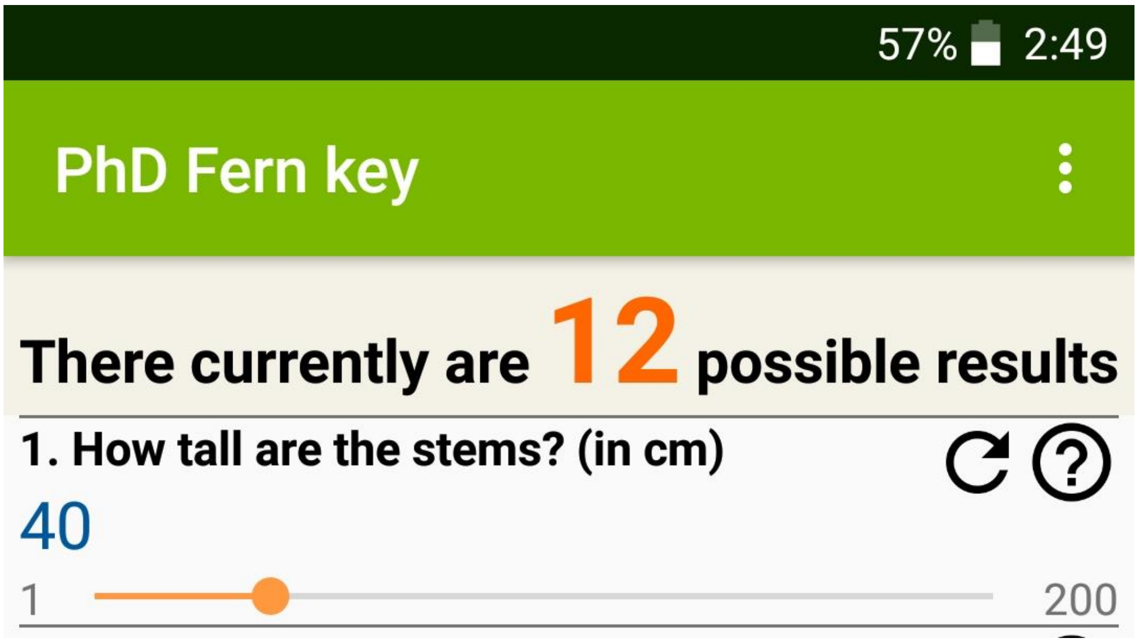
SeekBars were not replaced with radioButtons at this stage for two reasons. ID keys are generally intended to be accurate; questions requiring accuracy are expected. 1; With morphologically similar taxa, the level of accuracy implied by a SeekBar may be required. 2; A considerable number of radioButtons would be required to replace a SeekBar, greatly increasing the length of the list of questions.

It was felt reducing the size of the results left text at the top of the page would not be beneficial, as it would be smaller and less visible, thus users may be more likely to ignore it. However, the exact wording of this text was adjusted to make it more accurate and useful. It was felt that the users did not notice they had reached zero results, they would have to go back and double check their answers, thus reaching a more accurate identification

7.4.2.2 Session two

Users were often reaching zero results without realising, and felt that the display at the top of the page was not 'obvious enough'. To increase awareness of the number of results remaining, the font size of the number was increased relative to the rest of the text and the number was changed to the same orange used for 'checked' radioButtons or the drag-able part of seekbars. This is shown in Figure 7-11. When the user hit zero results, the wording of the text changed and all text was black and the same font size.

Note that a suggestion to 'freeze' the app at zero results was not used as it was felt this would be more frustrating to the user.


57% 2:49

PhD Fern key

There currently are 12 possible results

1. How tall are the stems? (in cm)

40

1 200

Figure 7-11 – Partial screenshot of the app showing the top bar and question one. This screenshot highlights the number of text remaining, with the large and noticable orange number '12'.

Some questions and help texts were indicated as being poorly worded, and some unexpected app behaviour, typos, and inaccuracies in the database were highlighted. All were corrected and adjusted as required, aside from a suggestion to indicate which teeth should be measured. It was noted in the volunteer feedback that teeth can vary depending on location on the stem. As this issue had not been highlighted before, and paper-based keys do not indicate which teeth to use, it was presumed this was a specimen issue.

Despite further indication that seekBars with many options were fiddly to use, they were not amended at this stage. there was no obvious alternative design that maintained the two key aspects of the seekBars: 1, the ability to set the minimum and maximum values, with users quickly selecting the correct value between these; 2, their small vertical size in the layout.

Futher calls for more images in help-boxes has the same problem of limited copyright source material as previously. A call for more information and an image of the taxon in the extra information box when the user presses on a result was not acted on as this would have been contrary to the suggestions in the previous session. This information is also available through the ‘read more online’ button. If this request came up again, it would have been revisited.

The design of the search page and behaviour of the search function meant that there was no clear way to highlight the most diagnostic questions. Highlighting these questions, rather than moving them to the top of the page as per suggestions from the previous session, was considered a better approach, thus if a method to achive this was found it would have been acted upon.

7.4.2.3 Session three

The further database inaccuracies were corrected. In this session no poorly worded questions, typos, or similar problems were highlighted.

The behaviour of questions that had been disabled based on the answer to other questions was found to be faulty. If a question had been answered (adding its ‘where x=y’ clause to the search query) and subsequently was disabled, the clause remained in the search query. These questions could not be reset individually and remained answered when the ‘reset all’ button was pressed. This was changed so if a question had an answer, but was subsequently disabled, the clause was removed from the search query but the radioButton indicating the chosen answer remained checked. If the question was subsequently re-enabled, the clause was re-added to the search query. Ensuring the radioButton remained checked while a question was disabled gave the user a visual indication of their choice.

Another request for more pictures faced the same problem as previous sessions, thus was not acted upon. A lack of extra images also meant that a request for greater detail in help-boxes was not acted on. It was felt more appropriate to revisit these helpboxes after extra images had been created, as these images may alleviate the highlighted issue.

Two bits of feedback highlighted areas that would require considerable amounts more data to improve. The reduced accuracy of the key when hybrids are included in the database is a limitation of the available data about the hybrids. In chapter 3 data for hybrids is inferred from their parents if no sources list it, thus hybrids come up frequently when using this version of the database in the key. While use of subjective variables was avoided, the nature of some characters meant they were required as the tools required for precise answers were not widely available.

Emailing users their answers to the key was avoided to ensure there were no data security issues in the app.

7.4.2.4 Session four

As there had previously been feedback highlighting seekBars as difficult, further calls for an alternative including the explicit suggestion of radioButtons was acted on. A set of toggle switches were placed in the actionBar options menu for the search page, toggling seekBar questions to radioButton questions. The required radioButtons were established by grouping variants of a feature that distinguished the same set of taxa. This resulted in a considerable number of radioButtons being required for each question, thus seekBar layout was set as the default and radioButtons as an alternative to avoid an excessively long list of questions. The seekBars were felt to be slightly misleading, with the '1' next to the starting position of the seekBar suggesting they were already set at '1' (see Figure 7-11) despite the user not having interacted with them. The '1' was thus removed, giving the appearance shown in Figure 7-12.

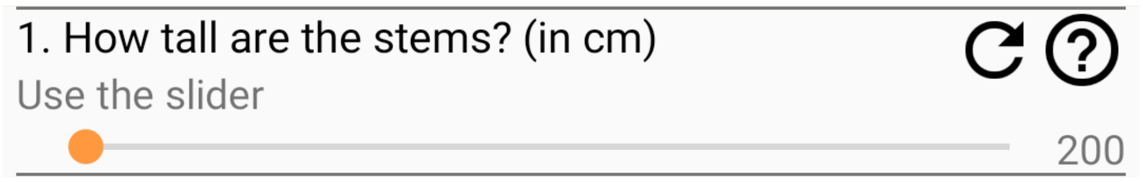


Figure 7-12 - Updated seekBar following the feedback from session four

Instruction guides for the app and ID keys were written. This instruction guide is detailed in section 6.3.3.1. In the version of the app available in Appendix A , the instruction guide is available in the actionBar options menu. In a production version of the app with several modules, it would be more appropriate to have this introduction shown first when a new user opens the app. sharedPreferences could then be used to store whether the user had seen the introduction, and the option to re-view the tutorial could be placed in the action-bar menu. Short diagnostic texts were written for each taxon, and were used in the extra information boxes. This resolved the issue that ‘further information’ boxes required a working network connection. If a network connection was not available, then the ‘Read further on Wikipedia’ button was disabled. A request for images of the taxa to be placed in their ‘further information’ box was not acted on. At the start of the project, app size and available storage on user devices was a consideration in the design, and images of the taxa are available online. It was also felt that if images were provided users would try to match with the picture, a potentially less accurate technique compared to use of the key and diagnostic information. It is also effectively impossible to display the complete morphological range expressed in a taxon in a single image, thus these images would not necessarily be accurate. An installation guide was also written, and can be found as part of appendix A on the attached DVD.

The behaviour of the search button was changed such that if zero results remained, an alertDialogue suggesting the user checks their answers was displayed. This approach was felt to be preferable over simply blocking the search button as it provided some context to the user.

Despite the continued calls for more pictures, the same limitation from previous calls for more pictures still applied. With the limited financial and time budgets remaining at this stage of the project, there was no way to commission the required diagrams from a professional source. The suggestion to include an introductory mini-guide to *Equisetum* features was not acted on for the same reason; this guide would cover the same area as the help-boxes. As there had been previous suggestions to include more images in the help-boxes, it was felt more appropriate to see if the improved help-boxes fulfilled the request for an introductory mini-guide.

7.4.3 SDLC (Software development lifecycle)

Use of a phased development style SDLC, with different styles of testing and evaluating sessions, different user-groups, and 4 cycles was effective. Each session gave different feedback, with different groups focusing on appropriate areas. Later sessions generated fewer points of feedback, suggesting that the approaches taken on acting on different types of feedback (e.g. single point issues acted on directly, significant changes only after repeated calls) was appropriate.

7.5 Discussion

The use of a phased development SDLC was clearly successful: the app received positive reviews, suggestions for improvement became increasingly limited to areas where work was less justifiable at this stage, and database accuracy improved as errors were found. This agrees with the findings of De La Vega et al. (2014), Kirchoff et al. (2011), Stinson et al. (2010), and Castensøe-Seidenfaden et al. (2017) where gathering feedback from users at multiple points in an iterative process is beneficial. The four major cycles in this project however indicate the two suggested by De La Vega et al. (2014) may not be enough. This project did not involve the prolonged sessions of Licskai et al. (2013); this is unlikely to have caused any issues, ID guides are not used for extended periods of time, instead being used in sessions, which these sessions simulated multiples of.

Choice of which SDLC to implement was relatively simple. The waterfall approach (and directly related SDLCs) was not appropriate; key goals for this project included that the users would find the product easier to use than a traditional paper-based key, and that the product was high quality. These goals necessitated incorporation of feedback from the users, meaning that a single 'run' through each phase of development was unlikely to be successful. Working as a sole developer and on a product with a relatively simple structure effectively ruled out parallel development methods. There were some parts of the project which could have been developed simultaneously, such as the user interface and the database contents. Development of significant portions of these areas simultaneously would likely have proved difficult (you need to know what aspects and variant of morphology are in the database before you can write the questions and incorporate them into the user interface, for example). Working as a sole developer also restricted the number of areas that could be developed simultaneously. As user testing and evaluation sessions required a fully working version of the app, rather than a partially working prototype, it was felt that a phased development SDLC rather than an agile SDLC was more appropriate. With the above conditions, the most appropriate SDLC was felt to be a phased development approach. If the continuous testing process is included as part of the SDLC used, rather than simple code quality conformation, then an AGILE SDLC is a more appropriate description of the process. Given that this continuous testing was informal (not structured), and importantly undertaken by the developer rather than users, it was felt that while this was part of the development process, it did not qualify as a formal session.

The key result from the feedback was that the original 'list of checkboxes' design from the MSc project (Bewsey, 2014) did not require major work, instead only requiring neatening. This was clear throughout the sessions, as while there were criticisms of details of the design, there were no criticisms of the overall design. Importantly, there were no cases of people being completely unable to use the app with the chosen design. The only user error on the overall design was that several users answered every question, rather than only the questions they were able to answer confidently. This lead to unnecessary elimination of taxa from the results lists due to misidentified feature variations. In order to ensure users were aware they could ignore 'difficult' questions the size of the hint at the top of the screen was increased over subsequent versions and modifications to the exact wording were made, but this user error persisted. While this may be a symptom of a design problem, it may also be a symptom of user unfamiliarity with multi-access keys. Future work should establish what the leading cause was. This future work will likely require at least some sessions where users are directly asked whether they know how to use electronic multi-access keys successfully, this will provide useful data on where the issue is. This was not asked in these sessions, as the aim was not to establish whether people knew how to use this style of key, the aim was to produce a high quality version of this style of key.

Most feedback points were either single point issues (either design or database accuracy), or simple systematic issues (e.g. not indicating where to measure stem height from). These were fixed after the specific session, and generally were not highlighted again. The effectiveness of this is particularly highlighted by the decreasing amount of feedback over several sessions. That subsequent sessions still highlighted single point issues may indicate one of two effects occurring: one; developing a complicated program requires a considerable amount of testing and evaluating, and four sessions may not be enough to catch all such issues; or two; different people using the app have different opinions on the best design, and that it is impossible to make the perfect product for everyone. While single point issues were highlighted in all sessions, no individual part of the app was bought up in more than one session, suggesting that effect number one was more likely. This would indicate that more testing and evaluation sessions would be beneficial; this would need to be balanced against the need to release the app, and the risk of never-ending-sessions.

Some feedback would require more significant behavioural changes, such as the changing from seekBars to radioButtons. The decision to act on these feedback points only if they were bought up over more than one session and by more than one person was based on two ideas: that acting on the feedback as soon as it was bought up would risk overfitting the app to a single person; and that it risks removing behaviour that the majority of users like. There were two notable points of feedback of this style. There were calls for the question page to highlight the 'best' questions based on the current set of results. This behaviour was not implemented, despite several calls for it, as there was not a consistently expressed desire for a specific new behaviour. It would also require an update to the search function in a manner that was not clear. The seekBars 'fiddlyness' and the suggestion that radioButtons would be more effective was bought up twice, and was acted on. As detailed in the results, it was not acted on exactly as suggested by the feedback, as this would have resulted in an especially long list of radioButtons.

There were two points of feedback that came up repeatedly, but were unfortunately unactionable.

Improved quality and range of pictures were frequently requested. This is echoed in the findings of Wu et al. (2017) and Milward et al. (2017) where users expressed a desire for more images. It is also similar to the findings of Kirchoff et al. (2011), Dellinger-Johnston (2015), and Hawthorne et al. (2014) where primarily visual rather than word guides were discussed favourably. Feedback requesting more images for this guide again suggests a user preference for as many images as possible. The set used here were adapted from Page (1982), as they were good quality images and could serve as a reference for the intended quality of the final image set if the app was made publicly. As these images were subject to copyright, they were not modified in form (the only action used was cropping to extract the required image/part of image), and were used as sparingly as possible. While attempts were made to create new images to fill in these gaps, they were not high quality or detail. This would have not improved the app, as user feedback and prior research from Stag and Donkin (2017), Dellinger-Johnston (2015), Hawthorne et al. (2014), Milward et al. (2017), and Wu et al. (2017) all suggest that use of higher quality images is very important in app design. While photographs could have been included, using the approaches mentioned by Kirchoff et al. (2011), there was not enough fresh material available of a wide enough range of species so it would not have been possible to illustrate all possible variation of the specimens. As per (Hawthorne *et al.*, 2014), exact nature of the image does not directly impact the quality of the key, and it was felt that when using a smaller device (such a smartphone) instead of a larger device (such as a tablet or a computer), line drawings are more useful.

Linking outside the app to include other available functionality was requested. This included linking to the camera and location data to both help eliminate taxa based on location and record spotting locations. Linking outside the app was avoided for lack of necessity, species migration, data security, and preservation of rare taxa reasons. The data used in this app was gathered from sources that do not use specific location to help identify specimens (they only assume you are in the correct country), and species are generally not defined by absolute location. Species may not have their complete distribution recorded, and are able to spread and migrate. Location data is therefore not necessary for species identification. If user location is used to identify taxa, it risks false eliminations when the user has found a taxon outside of its recorded range. If a user accidently leaves on the option to eliminate taxa based on real-time location data, identifying herbarium specimens would become considerably more difficult! As the app is aimed at the 'enthusiastic amateur', it was felt that it would be more appropriate behaviour to list all morphologically possible taxa. If multiple taxa remain (for example a morphologically more-different but location appropriate taxa, and a morphologically more-similar taxa that has not been recorded in the users location), then it would be the users choice to choose the appropriate identification from the list generated by the app. If only one possibility is generated by the app, but this taxa is not recoded at the users location, it is again the users choice whether to accept this identification, and suggest that the taxa has not been recoded here (either simply been missed, or has migrated). In future versions of the app, it may be appropriate to include an indication in the results list that a taxon has not been yet recorded in the users location (as long as the taxon is still included in the list). This is not possible in the current version of the app; the decision to not include location data was taken very early in development, no location data was included in the database. Inclusion of rare taxa location data has a risk of advertising the location to potential species 'hunters'. It was felt there would be no issue including the use of the camera; the user is aware of where they are, so there is no problem with location data being stored. The ability to go directly to the camera was included in the 'further information' about individual results.

Hawthorne et al. (2014) and Dellinger-Johnston (2015) both found that use of visual keys can result in a greater accurate identification rate. For Hawthorne et al. (2014) difficult species were still hard, and reduced accuracy. With further work to this key, it should be possible to reach the accuracy of the visual keys generally, rather than just for 'easy' taxonomic groups. Well-designed images in the help boxes should allow the same behaviour as the visual keys for 'easy' taxonomic groups, while carefully worded questions combined with the images should hopefully be enough for 'hard' taxonomic groups.

As shown in Table 7-1, the accuracy levels were shown to be situationally dependant and broadly similar to the traditional paper-based single-access key, but suggest that more work is going to be required on the volume and accuracy of data for better performance in all situations than traditional paper-based keys. The accuracy of identifications here was lower than previous investigations of ID guides (which ranged between 53.7% (Sharma, 2016) and 80% (Vollbrecht *et al.*, 2013)). It is suspected that the lower overall accuracy here is due to *Equisetum* herbarium specimens being harder to identify. Previous investigations noted a similarity in identification accuracy between single- and multi-access keys. This similarity in accuracy between keys is again recorded here but the accuracy is lower, hence the suggestion that *Equisetum* herbarium specimens are 'harder'.

When the restricted set of taxa (section A of Table 7-1) were used, accuracy was higher than Stace (1997 and 2010). There were two probable reasons for this. The first is that the data were better quality; this database had no inferred data (see section 4.3), thus reducing risks of artificial overlaps between taxa. The lower number of taxa also means that fewer questions needed to be answered correctly before a single taxon remained in the results list. The second reason is that the session these data were generated from highlighted that single-access keys were prone to failure when a specimen did not display the required aspect of morphology. This style of failure occurred frequently, causing the low recorded accuracy of Stace (1997 and 2010). When the database with the full set of taxa was used (section B of Table 7-1), the accuracy of the app fell. This was most likely due to the lower quality of the data; some of the taxa had some estimated data (see section 4.3) and will have been artificially harder to eliminate due to the overlapping character data. The increased number of taxa meant that more questions needed to be answered to reduce the results list to a single taxon. These two aspects combined will have increased the difficultly in successfully using the app key.

The number of taxa in this ID key app was similar to higher and/or more taxonomically similar than most other studies where different forms of ID key have been compared. There were 21 taxa in the full set and 12 in the restricted set, both sets including hybrids (see Table 4-3). The highest number of taxa in a previous studies is 43 in Dellinger-Johnston (2015). All 43 species are from *Quercus*. Sharma (2016) bee species, and state that 'some species are considerably harder to identify than others'. Stucky (1984) used 40 species, but these 40 came from a variety of different genera, suggesting a much greater morphological variation between the 40. Hawthorne et al. (2014) included around 20 species, but they do not indicate which species were used. Tardivel and Morse (1996) use only three species of woodlouse, Vollbrecht et al.'s (2013) use 11 Ephemeroptera families, and Stagg and Donkin (2017) used groups of five to six species. The number and taxonomic similar of species here is appropriate and not a limitation.

The results shown in Table 7-2 are very positive. They show that by the end of the development cycles, people found it easier and were happier to use the app. This is highlighted by feedback shown in G.2 and G.3, where the app was found to be easier to use and understand than paper based guides. This is similar to the findings of Dellinger-Johnston (2015), where users prefer the electronic key. This is also similar to Tardivel and Morse (1996) and Sharma (2016) The increased portability of a smartphone based guide was mentioned in the user feedback for the app; this point has been raised before by Stag and Donkin (2017), Silva, Pinho, Lopes, Nogueira, and Silveira (2011), and Tarkus, Maxl, and Kittl (2010).

This leads to the suggestion that one of the biggest issues holding back the wide-spread use of electronic based keys is their reputation; if the reliability and usability can be maintained, and the coverage improved, evidence from this chapter suggests electronic keys have the potential to become the preferred version. If they become more widespread, people will become more familiar with their use, leading to reduced user errors.

The sessions were run as a hybrid focus-group/user-test questionnaire as this was the most practical approach with the available resources and volunteers. During the sessions, volunteers spent most of the time split into smaller groups examining the app. This reduced the risk of the Hawthorn effect (Fox *et al.*, 2008; Sauro, 2017), as the person coordinating the session could only be with one group at any given moment. While bias may still have been introduced by coordinator intervention when participants encountered issues (a similar issue was noted by Wu et al. (2017)), the inability for the coordinator to work with all groups simultaneously will have hopefully reduced this. When questionnaires were used, they were designed to focus on the areas most important to the session, with more closed ended questions in order to gather more quantifiable data about the area being focused on. The questionnaires still allowed comment on areas of the app not directly focused on through the more open-ended questions. This questionnaire design aimed to maintain specificity and usefulness of answers, and reduce the risk of low detail answers highlighted by Milward et al. (2017). While groups tended to work separately, some larger discussions did occur and generated novel feedback similar to Milward et al. (2017). The fragmented nature of these larger discussions will have hopefully reduced bias induced by louder people.

There were limitations to this study.

- The app was designed to be used in the field as well as in herbaria, labs, and university teaching settings for example. Unfortunately, none of the sessions took place in the field. Evaluating with as many separate taxa as possible was required to attempt to ensure there was no ‘edge case’ behaviour with less common taxa. As there was not enough funding available to transport volunteers around the country, and specimens of all required taxa were available from **RNG** and **E** and fresh specimens of some taxa available locally, sessions took place in lab or herbarium conditions only. In similar future user testing and evaluation, in-field sessions would likely prove very beneficial, and should be undertaken.

- The two sessions with herbarium volunteers saw returning participants, potentially inducing some bias in a similar manner to Castensøe-Seidenfaden et al. (2017) where participants returning for more than one study may have bias the results to these participants preferences.
- Participants were not directly screened on their ability to identify *Equisetum* taxa or use plant ID guides. If all participants were highly- or completely un-qualified in either area this would have induced bias in the feedback. In these sessions, participants were recruited by asking for volunteers from groups where all potential participants should be capable of ID guide use, but not experts in *Equisetum* identification.

Several areas that may have induced bias or had been limitations in previous studies were not limitations here.

7.6 Conclusions

Direct evaluation with users is, obviously, a necessary step in app development. This was previously very well known, and has been shown yet again here. At the start of the process, the app was based on pre-existing apps, and the app produced in the MSc study (Bewsey, 2014). This initial version of the app was received well, but it was clear there were areas where improvements that could be made. During the testing and evaulation process, these areas were established, and the app was improved. The success of this process is evidenced by the final questionnaire data shown in Table 7-2, where users clearly preferred the app over the paper based key. This evaluation highlighted a relatively low accuracy of the key compared to previous keys (electronic or paper). The use of herbarium specimens and an inherent difficulty in identifying *Equisetum* probably account for this; the paper-based keys tended to be about as accurate as the app.

The evaluations here add to the mounting evidence that ID guides of this sort are preferred by the user. They are easier to use, less likely to go wrong if the specimen is imperfect, and are capable of more functions than a piece of paper. As the design being evaluated here was based on previous works, it seems that a good ‘base’ has been found.

8 Business study

8.1 Introduction

Apps are known to be expensive to create (Mrva-Montoya, 2015). A previous survey discovered that apps can cost over $700K to produce (Craigmile, 2015). This survey was based on an hourly app-creation rate of between $100-150, and found that the total required hours varied between 252-4850. Of these hours, only 200 max were assigned to research and discovery. It must be noted that the survey did not include maintenance costs (Craigmile, 2015).

There are several different business models that apps can follow to fund app development. These include: free, freemium, subscription, paid, and paymium (Apple Developers, 2018). There are a variety of different methods of pricing; consumers can be charged directly for access to the content, consumers can sell information for access to the content, or consumers can sell attention (Lambrecht *et al.*, 2014). Digital publication and monetisation of books specifically are discussed by Horne (2012). Different approaches for these aspects are investigated, including: selling the rights, selling the software, and selling the content (Horne, 2012). The conclusion drawn by Horne (2012) is that the best approach for all authors (potentially aside from best-sellers) is to publish as an e-book (i.e. sell the content), as it is a low risk approach, while maintaining a large potential market.

One-time fees are the traditional way of charging for a product and are well understood however the market has diversified to try to break down the barrier caused by potentially high app prices. The 'freemium' model is where users are offered a limited service for free, with the full product being available for a fee (Kumar, 2014; Oh and Min, 2015; Yan and Wakefield, 2018). This model has been discussed and called 'freemium' since at least 2006 (Emilio and Gayo, 2009; Yan and Wakefield, 2018). Freemium Programs can follow the 'in app purchase' model. In this model the user can unlock additional features of the program (Roma and Ragaglia, 2016). These in app purchases can have a wide range of prices, for non-game apps these can range at least from $0.99 to $299.99, with a mean of $28.70 and a median of $12.99 (McGregor, 2015) (note that these prices are not necessarily the entire range). An alternative name for in app purchases is micro transactions. This term is in more common use in the video game industry, where micro transactions are an unpopular model for game pricing (Flores, 2017; Pearce, 2017; Anderton, 2018). Despite this negative opinion about their use, in-app purchases are known as an effective method of pricing the 'freemium' model (Kearl, 2016; Nelson, 2018) accounting for 79% of app revenue in 2014 (Hsiao and Chen, 2016) . The only app found during the project offering in-app purchasable modules was iFlora (Trackenberg, 2019), where various modules are available. These modules include national floras sold for £34.99, types of plants (such as alpine or trees) for various prices. An 'unlock all' option is also available for £77.99.

Subscription based models for e-commerce are becoming increasingly widely adopted by users (Chen *et al.*, 2018). Examples of software available for subscription including the Adobe suite (Adobe, 2018) or Microsoft Office (Microsoft, 2018b) (these are examples of Software as a service (SaaS) (Freedman, 2018). It must be noted that consumers are not necessarily fans of subscription based services (Chen *et al.*, 2018)). An alternative to the direct subscription model is Patreon (Patreon, 2018), where members of the public act as 'patrons' of members of the website. This funding model is becoming increasingly popular among creators and consumers (Hern, 2018). Another similar approach is crowdfunding websites, such as Kickstarter, where projects are advertised online for public funding, and members of the public can put forward different amounts of money, typically for different levels of 'reward' if the project is funded and completed (Kickstarter, 2018). This funding model is becoming increasingly popular among creators and consumers (Hern, 2018).

Selling of information involves selling user data (Lambrecht *et al.*, 2014). This approach is used by companies such as Facebook (The Star, 2012), but is increasingly unpopular with consumers (McRae, 2018). The General Data Protection Regulation (GDPR, implemented in the UK as The Data Protection act (Data Protection Act 2018, 2018;

Gov.UK, 2018)) was bought in to increase protection and allow users more control of data relating to their selves (Burgess, 2018).

Consumer attention is sold by providing space for advertising (Lambrecht *et al.*, 2014). Apps can be designed to include advertising as a method of monetisation (Google, 2018c). It is not easy to determine how much income this can generate, exact details are hidden, and are presumably trade secrets. There are, however, case studies available that suggest Google AdMob can be used to generate business-sustaining income (Google, 2018a). The declining effectiveness of traditional methods of advertising (pop-up or side-bar for example) has been discussed since at least 2009 (Clemons, 2009). Use of an advertising income based model relies on reaching a 'critical mass' of users to attract relevant third parties (Roma and Ragaglia, 2016). Various approaches to app funding and their success are discussed by Roma and Ragaglia (2016), with the success of different methods shown in Table 8-1.

Table 8-1 - Hypothesis and findings from Roma and Ragaglia (2016). *Table adapted by merging the contents of their table 1 and 8.*

Hypothesis	Hypothesis statement	Status	Explanation
H1a	Paid apps are associated with better daily revenue ranks than free apps.	Not confirmed	Hypothesis H1a is not confirmed as paid apps are not always associated with better daily revenue ranks than free apps. Indeed, in Google Play, no significant differences in revenue performance arise. The reason is due to the fact that users have relatively low willingness to pay. Hence, in spite of the general tendency of tilting away from the free revenue model in online markets, this model can still be as effective as the paid model in environments characterized by relatively low willingness to pay.
H1b	Freemium apps are associated with better daily revenue ranks than free apps. This improvement in the daily revenue rank is even higher than that obtained using the paid revenue model, so that freemium apps are associated with better daily revenue ranks than paid apps as well.	Not confirmed	Hypothesis H1b is not confirmed, as freemium apps are not always associated with better daily revenue ranks than free apps. Indeed, in Google Play, the opposite occurs, which makes the freemium revenue model inferior to both free and paid models. The rationale is that the presence of a free version in the freemium model can cannibalize the paid version in this platform given that Android users have on average limited willingness to pay and thus may decide to utilize the free version without upgrading to the full paid version. Also the free version will not be able to generate enough advertising revenue because, differently from the free model, the freemium model is not fully centered on generating revenue from ads.
H2a	It is more likely that paid apps are associated with better daily revenue ranks than free apps in the Apple's App Store.	Confirmed	Hypothesis H2a is confirmed because results show that it is more likely that paid apps are associated with better daily revenue ranks than free apps in the Apple's App Store. Indeed, paid apps are associated with better revenue rank than free apps in this store, whereas no significant difference emerges in Google Play. The reason is that Apple's App Store players are more willing to pay that Android users, thus they are less worried about spending their money if quality is delivered to them.
H2b	It is more likely that freemium apps are associated with better	Confirmed	Hypothesis H2b is confirmed because results show that it is more likely that freemium apps are associated with better daily revenue ranks than free apps in the Apple's App Store. The rationale is the same as that provided for H2a.

	daily revenue ranks than free apps in the Apple’s App Store		Moreover, no significant differences in terms of revenue performance between paid and freemium apps emerge in Apple’s App Store.
H3	The effect of the different types of revenue models on app daily revenue rank depends on the app category.	Confirmed	H3 is confirmed as our results show that certain revenue models are more (or less) preferable in certain categories. This is because different categories satisfies different needs, attract different consumer segments, and are associated with different levels of uncertainty with regard to product value.
H4	Apps enabling in-app purchase are associated with better daily revenue ranks.	Not confirmed	H4 is not confirmed as apps enabling in-app purchase are not always associated with better daily revenue ranks. Indeed, in Google Play the opposite occurs. That is, ceteris paribus, apps without in-app purchase leads to higher revenue performance. The rationale is that, according to theory, in-app purchase can be optimal in presence of two segments with distinct valuations for high-end version and low-end version. As the highly valuable segment is on average not adequately developed in Google Play, the in-app purchase strategy turns out to be inferior in this store.
H5	The option of in-app purchase is more likely to be effective in terms of daily revenue rank in the Apple’s App Store than in Google Play.	Confirmed	H5 is confirmed because the option of in-app purchase is more likely to be effective in terms of daily revenue rank in the Apple’s App Store than in Google Play. The rationale follows from above. Indeed, differently from Google Play, the highly valuable segment is largely developed in Apple’s App Store due to the higher average consumers’ willingness to pay in this store.
H6	The effect of in-app purchase on app daily revenue rank depends on the app category	Confirmed	H3 [H6] is confirmed (although to a less extent as compared with H3) as our results show that in-app purchase can be more (or less) preferable in certain categories. The explanation follows that provided for H3.

8.2 Aims

The aim of this chapter is to determine whether-or-not production of an app ID guide to the UK using the process described in the rest of this thesis is financially viable. For this purpose, a number of areas were investigated.

- Success of different funding models for apps in general
- Sales/downloads and prices/method of generating income of current plant ID guides (both paper- and electronic-based)
- Potential market size for this project
- Potential cost of continuing this project

8.3 Method

In order to establish the most successful app pricing method, the Google Play (Google, 2018d) 'Top Grossing Apps' section was investigated. While there is no guarantee that these apps are run commercially, their position as the top grossing apps is strongly suggestive of this.

8.3.1 Data from existing guides

In order to find existing plant ID guides for the Android platform, various search terms to find plant ID keys were searched in the Google Play Store (Google, 2017a), with the following data recorded in 2015 and 2017:

- number of downloads
- Average ratings (data shown in appendices H.1 and H.2)
- number of ratings (data shown in appendices H.1 and H.2)
- Last update date (data shown in appendices H.1 and H.2)

Data were recorded for apps where it was either clear they had plant identification capabilities, or appeared as if they might. Download data for apps available on Google Play (Google, 2018d) are displayed in the apps page as a range (e.g. between 1,000 and 5,000 downloads). In this chapter, the upper and lower bounds of this range are averaged to give an approximated number of downloads.

Paper based guides were chosen based on prior knowledge and feedback from Dr Culham. Sales data for paper based guides and e-books was gathered from Amazon UK. While sales data for Amazon purchases are not directly available, for this chapter relative sales levels are estimated from number of comments using the 1300 multiplier established by Spool (2009) in Equation 1. This is a crude method of estimating sales, as different categories of product will have different sales/comments ratios, but is the only method currently available.

Equation 1 - Equation to predict number of sales through Amazon

Sales = number of comments * 1300

Current prices at time of writing for paper based and e-book style projects were gathered from Amazon UK (price at physical stores was assumed to track the Amazon web-price closely enough that they could be assumed to be the same), prices for Android based projects were gathered from the Google Play store (Google, 2018d). Historic prices for paper based projects were gathered from CamelCamelCamel (2017). If multiple listings were available on Amazon UK for an individual project, the listing with Amazon UK as the first party seller was used in CamelCamelCamel (2017). Where necessary in the chapter, historic prices were adjusted to account for inflation using the Bank of England (BOE) inflation calculator (Bank of England, 2017).Pre-existing project funding (i.e. grant, self-funding business, etc.) method was established by examination of available material for the sources.

8.3.2 Potential market size

In order to establish a potential market size, the size/membership of existing relevant organisations was investigated. Organisations were chosen based on prior knowledge of their purposes, and size/membership was established by investigation of their website.

8.3.3 Potential available grants

Potential grant availability and value was investigated in a similar way to potential market size; grants advertised on the websites were investigated. Organisations were chosen on prior knowledge and suggestions, and were chosen to give examples of the possible range of values, rather than the complete array of potential grants.

8.3.4 Cost of production

Due to the relative ease of estimating some costs and difficulty of estimating others, some real word data were combined with 'placeholder' values. A staff of 3 (1 full time programmer (produce the app, data extraction program, and any other program as required); 1 full time botanist (key design, data management and accuracy checking); with the final member of staff with mixed skills to help where needed (assistance of either of the other two members of staff, or to act as the public facing member, for example)) was assumed based on experience from this project, and information available from previous works. Average UK wage rates were gathered and adapted from the Office for National Statistics (Smith, 2017). National insurance (NI) rates were gathered from the Gov.UK website (Gov.UK, 2017b). The UK has an automatic enrolment pension service (The Pensions Advisory Service, 2018b), which the employer at time of writing pays 2% towards, rising to 3% in 2019 (The Pensions Advisory Service, 2018a). Business rate tax costs were estimated by finding equivalent properties at appropriate locations and seeing what business rate they were listed at on the UK government business rate evaluation website (Gov.UK, 2017a). Rental costs were estimated by finding properties of a similar size in appropriate locations advertised for rent at time of writing and using their rental fees as placeholder estimates. All internet prices have been gathered from Virgin media business (Virgin media Business, 2017). The cheapest available broadband was included.

8.4 Results

8.4.1 Method of income for successful apps

The top 10 apps in the 'Top Grossing Apps' on the Google Play store (Google, 2018d) are all 'freemium'. They vary in exact manner of income generation, with many advertising 'in-app purchases. This top ten is shown in Table 8-2. The market segment descriptors are unclear at face value, as everything aside from Tinder and Google Drive are games. The number of installs is cumulative, rather than current, so removal from a device does not reduce the total count. The lowest number of installations is 10 million.

The first app in the 'Top Grossing Apps' list that has a direct fee is 'Driving Theory Test 4 in 1 Kit – Hazard Perception', with a cost of £4.99. This is in the 'education' category, and is a companion app to help people pass part of the UK driving licence test. It is at position 118 in the 'Top Grossing Apps' list (correct on 4/11/2018), with only 100,000+ installs.

Table 8-2 - The top ten 'Top Grossing Apps' on Google play, along with various areas of information about the app. [1](Tinder, 2018) [2](King, 2018) [3](Niantic Inc, 2018) [4](Google LLC, 2018a) [5](Playrix Games, 2018a) [6](Supercell, 2018) [7](Moon Active, 2018) [8](Playdemic, 2018) [9](Roblox Corporation, 2018) [10](Playrix Games, 2018b). *All data correct on 4/11/2018.*

Ranking	**App**	**Market segment**	**Income method**	**Installs**	**App cost**
1	Tinder®[1]	Lifestyle	3 tier model: free, 'Plus' subscription with some added features, and 'Gold' with further unlocked features	100,000,000+	Free
2	Candy Crush Saga[2]	Casual	Undescribed 'optional in-game items'	500,000,000+	Free
3	Pokémon GO[3]	Adventure	Undescribed 'in-game purchases'	100,000,000+	Free
4	Google Drive[4]	Productivity	Not stated	1,000,000,000+	Free
5	Gardenscapes[5]	Casual	Undescribed 'in-game items	50,000,000+	Free
6	Clash of Clans[6]	Strategy	Undescribed 'game items'	100,000,000+	Free
7	Coin Master[7]	Casual	Undescribed 'in-app purchases'	10,000,000+	Free
8	Golf Clash[8]	Sports	Undescribed	10,000,000+	Free
9	ROBLOX[9]	Adventure/Action & Adventure	In-game purchases via in game currency. A subscription to the game	100,000,000+	Free
10	Homescapes[10]	Casual	Undescribed 'in-game items'	50,000,000+	Free

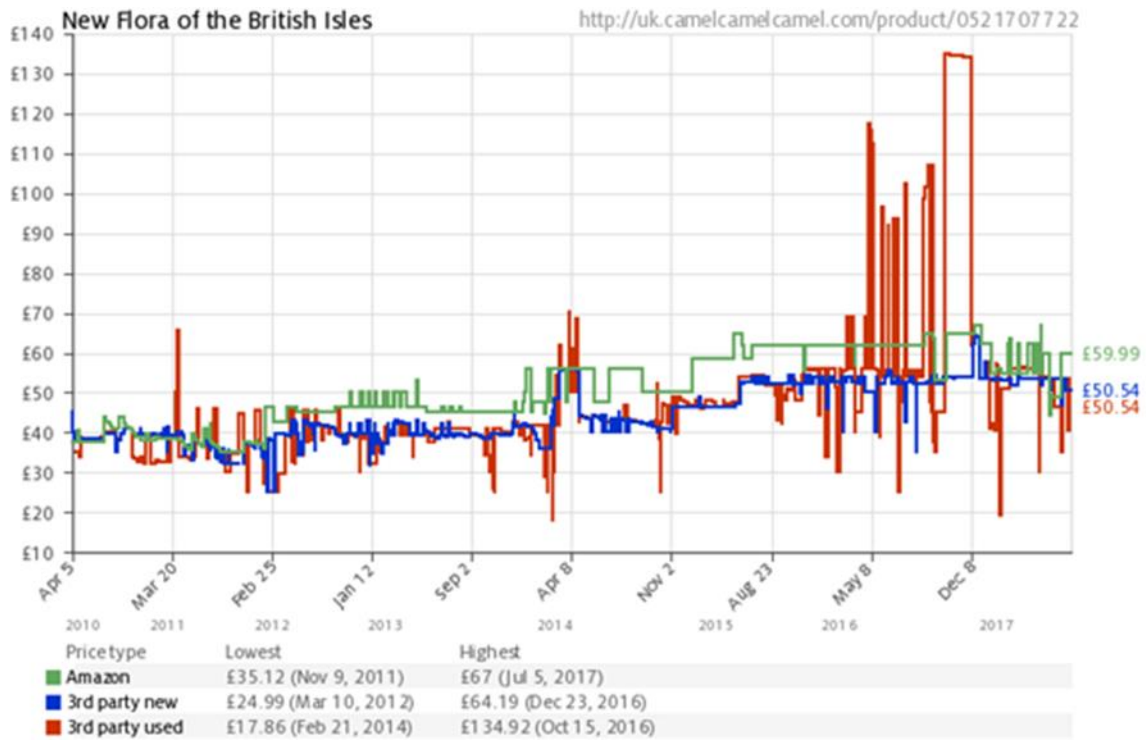
8.4.2 Costs and sales of paper-based guides

The price of different guides available on Amazon UK for paper based guides ranged from approximately £15 to £175 for individual books at the time of investigation (Table 8-3).

Table 8-3 - Prices of various paper based guides on Amazon UK

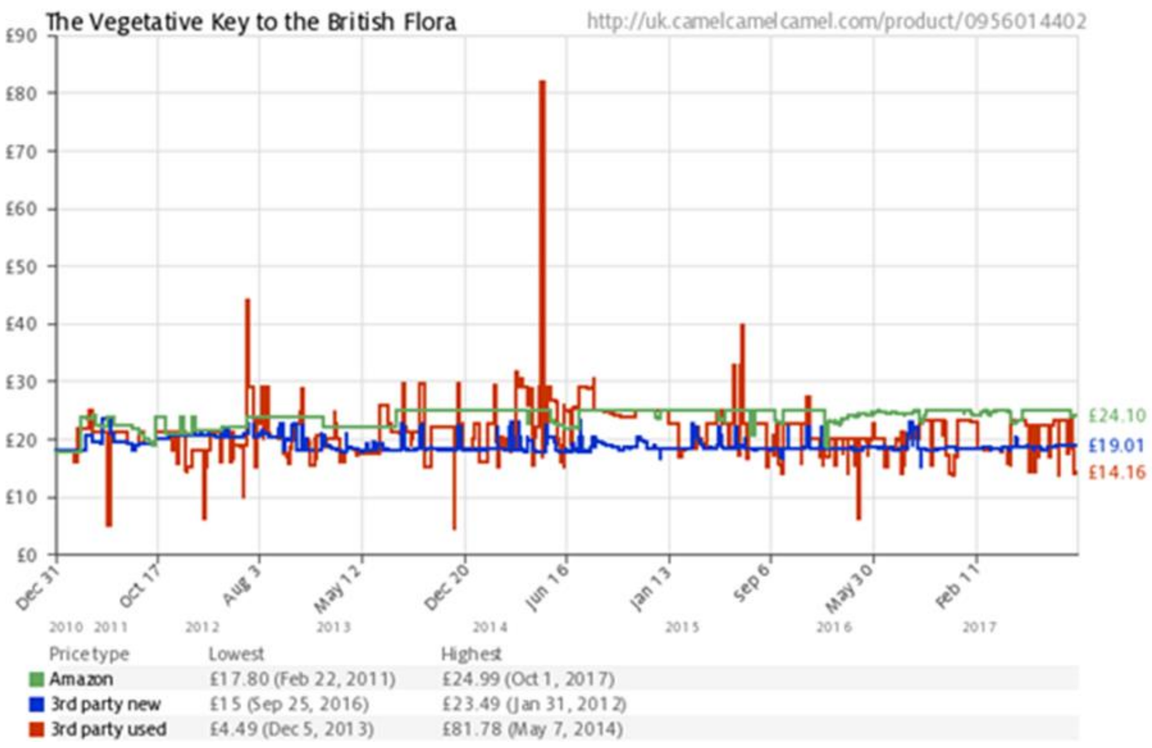
Source	**Price**	**Price correct on**
New Flora of the British Isles (Stace, 2010c)	£52.57 (Stace, 2010c)	26/10/18
Collins Wild Flower Guide (Streeter *et al.*, 2018c)	£34.95 (Streeter *et al.*, 2018b)	26/10/18
The Wild Flower Key (Revised Edition) - How to identify wild plants, trees and shrubs in Britain and Ireland (Rose and O'Reilly, 2006)	£15.66 (Rose and O'Reilly, 2018)	26/10/18
Flora of Great Britain and Ireland (Sell and Murrell, 1997, 2006, 2009, 2014, 2018) (Individual volumes)	Between £82.23 and £175 (Amazon UK, 2018c)	26/10/18
Flora Europaea (Individual volumes) (Tutin et al., 1964, 1968, 1972, 1976, 1980, 1993)	£40.99 (Typical Price) (Amazon UK, 2018b)	26/10/18
Flora Europaea (Full set) (Tutin et al., 1964, 1968, 1972, 1976, 1980, 1993)	£150 (Amazon UK, 2018a)	26/10/18
The Ferns of Britain and Ireland: Second Edition (Page, 1997)	£82 (Page, 2018)	26/10/18

The first party price of Stace (2010c) has remained fairly stable over the period where data is available, with a slight increase in price over time (Figure 8-1). When adjusted for inflation using the BOE inflation calculator (Bank of England, 2017) the 2011 first party price of £40 for Stace (2010c) on Amazon is approximately £44 in 2016 (the most recent dates available using the BOE inflation calculator (Bank of England, 2017) at the time of writing). The two third party options have consistently been lower than the first party price in most cases, with the third party used cost occasionally jumping very high during 2016 (Figure 8-1). Poland and Clement (2009) shows similar price stability and positioning for first party, and third party new and used when compared to Stace (2010c) (Figure 8-2). The first party price appears not to have increased, but it must be noted that the time period tracked is lower.



Price type	Lowest	Highest
🟩 Amazon	£35.12 (Nov 9, 2011)	£67 (Jul 5, 2017)
🟦 3rd party new	£24.99 (Mar 10, 2012)	£64.19 (Dec 23, 2016)
🟥 3rd party used	£17.86 (Feb 21, 2014)	£134.92 (Oct 15, 2016)

Figure 8-1 - Amazon UK first party new, third party new, and third party used price for Stace (2010c) *over time. Data gathered from CamelCamelCamel* (2017)*on 13/10/2017*



Price type	Lowest	Highest
🟩 Amazon	£17.80 (Feb 22, 2011)	£24.99 (Oct 1, 2017)
🟦 3rd party new	£15 (Sep 25, 2016)	£23.49 (Jan 31, 2012)
🟥 3rd party used	£4.49 (Dec 5, 2013)	£81.78 (May 7, 2014)

Figure 8-2 – Amazon UK first party new, third party new, and third party used price for Poland and Clement (2009) over time. Data gathered from CamelCamelCamel (2017) *on 13/10/2017*

The number of reviews and predicted number of sales using Spool's (2009) multiplier is shown in Table 8-4.

Table 8-4 - Number of reviews and predicted number of sales for a range of different projects on Amazon UK. Note that reviews of individual volumes of Flora Europaea were combined to give a total volumes sold value through Amazon UK. [1](Streeter *et al.*, 2018c). [2](Streeter *et al.*, 2018a). [3](Poland and Clement, 2018b). [4](Rose and O'Reilly, 2018). [5](Stace, 2010c). [6](Tutin *et al.*, 2018).[7] (Amazon UK, 2018b) [8] (Amazon UK, 2017).

Project	**Number of reviews**	**Predicted number of sales**
Collins Wild Flower Guide (2nd edition) [1]	26	338000
Collins Wild Flower Guide (1st edition) [2]	65	845000
The Vegetative key to the British Flora [3]	43	559000
The Wildflower key (Revised edition) [4]	164	2132000
Stace (3rd edition) [5]	30	390000
Flora Europaea (5 volume set) [6]	1	1300
Flora Europaea (Total reviews on all Amazon options for all volumes) [7]	4	5200
Flora of Great Britain and Ireland (All volumes) [8]	0	0

8.4.3 Costs, sales, and/or downloads of electronic guides

Most apps monitored were downloaded a few thousand times over the two-year period (see Figure 8-3). Note that the number of downloads in 2015 and 2017 are averaged; the exact number of downloads is not displayed on the Google Play Store (Google, 2018d), a range is displayed.

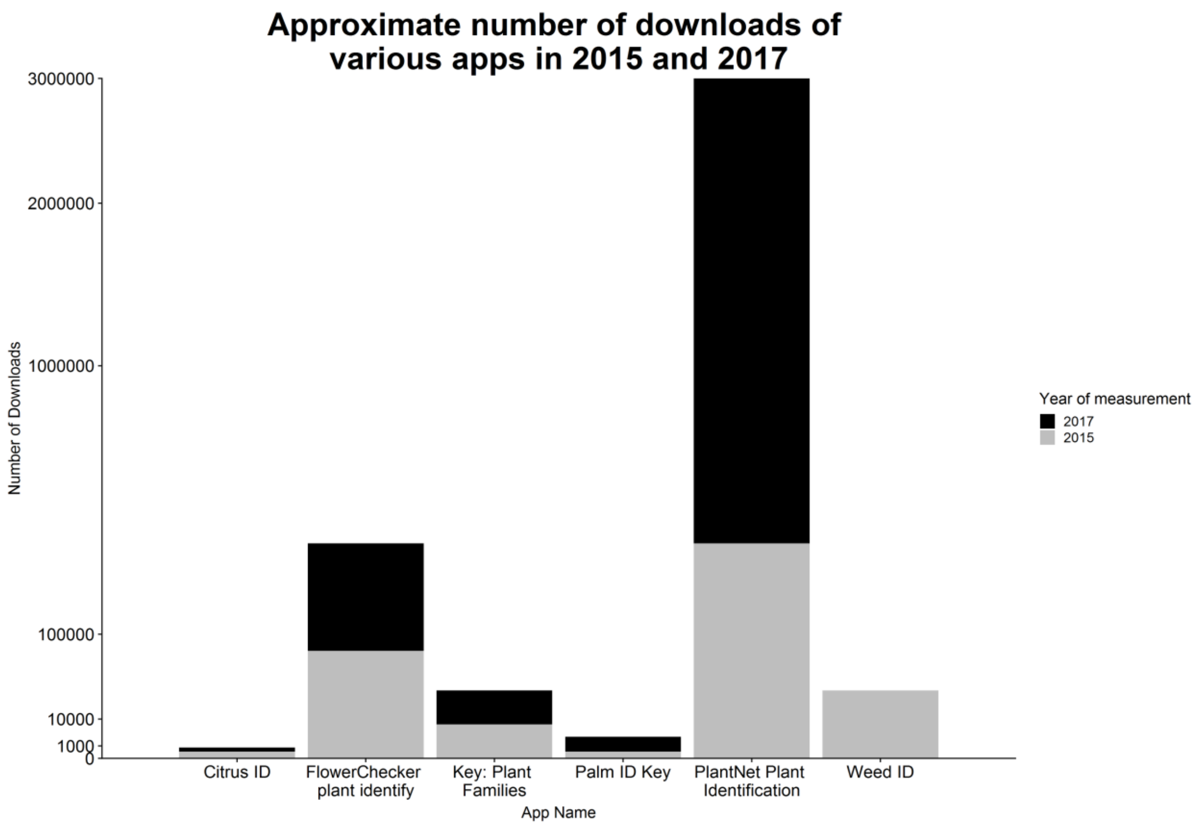


Figure 8-3 - Approximate numbers of downloads of various freely available ID keys available on the Google Play store (Google, 2018d)*. Download data are approximated as number of downloads is displayed on the Google Play store as a range rather than specific download counts, so the median was taken. The Google Play store* (Google, 2018d) *records cumulative downloads rather than current downloads.*

App price had no noticeable effect on number of purchases (see Figure 8-4 and Figure 8-5). Figure 8-4 isolates apps that clearly are or contain plant ID keys, while Figure 8-5 contains all apps that may include plant ID keys. Point colour on Figure 8-5 represents funding method.

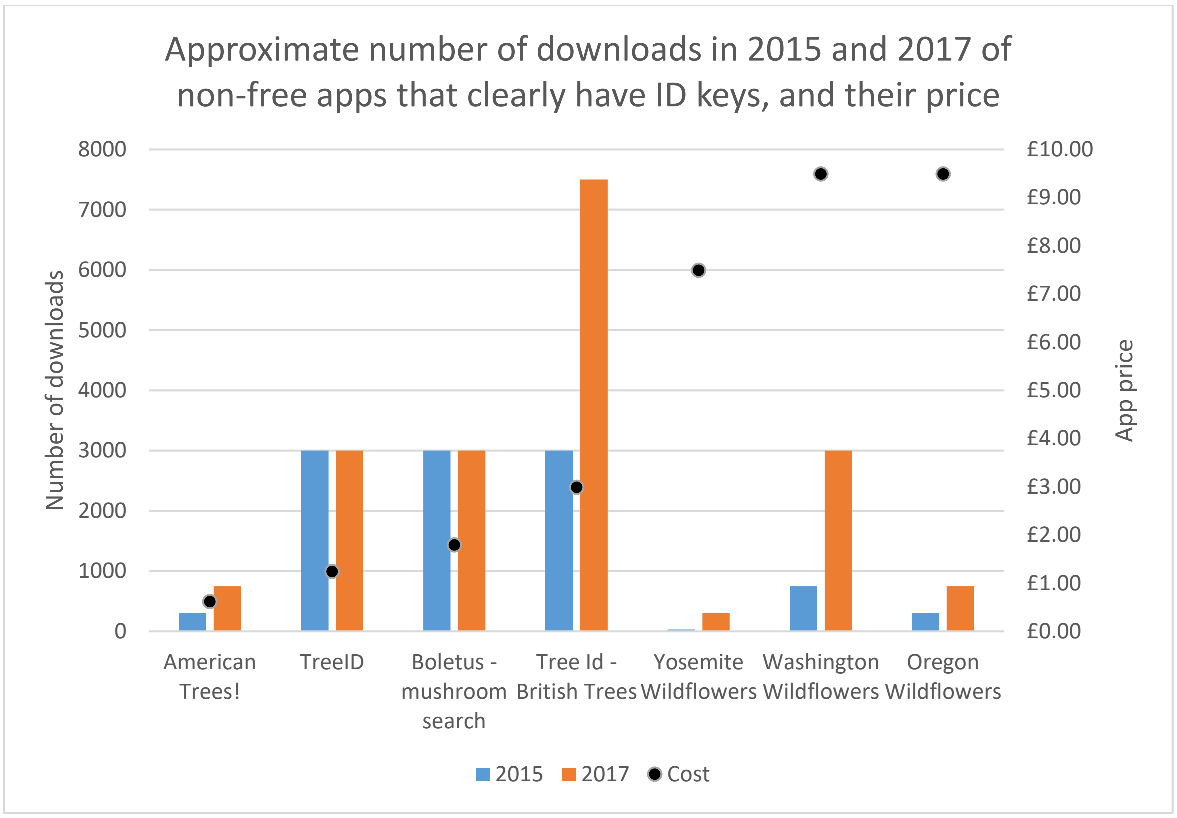


Figure 8-4 - The number of downloads (recorded in 2015 and 2017) of various apps that clearly contain ID keys, and their associated price. Number of downloads is approximate as Google Play does not display exact numbers of downloads, instead displaying a range. Number of downloads has been approximated by averaging this range.

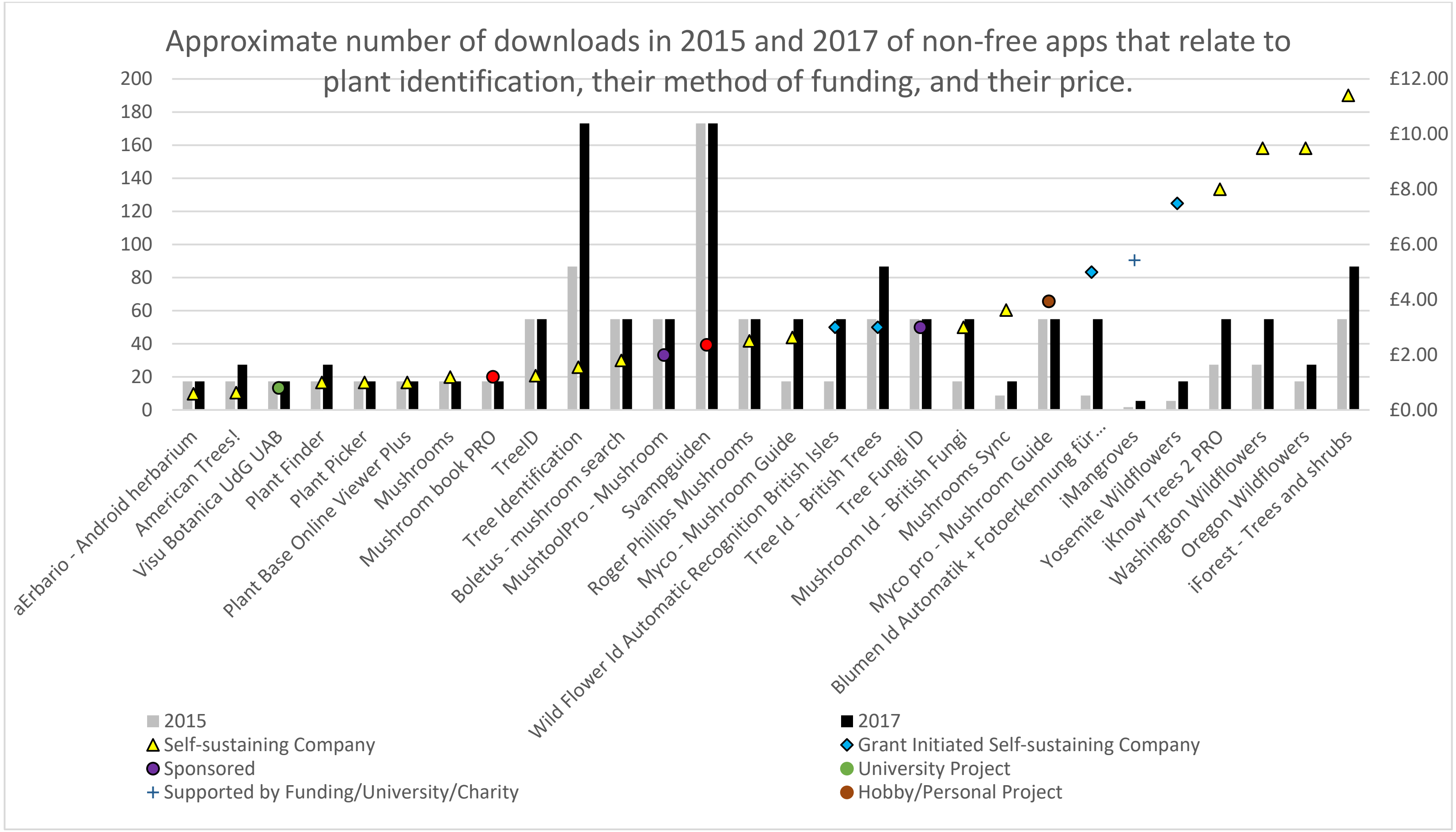


Figure 8-5 - The number of downloads (recorded in 2015 and 2017) of various app that related to plant identification, but where it was not necessarily clear that they contained an ID guide, their method of funding, and their price. The graph shows no obvious trend for any possible influence. Number of downloads is approximate as Google Play does not display exact numbers of downloads, instead displaying a range. Number of downloads has been approximated by averaging this range. Number of downloads is displayed as square root so apps with fewer downloads remain visible.

8.4.4 Funding methods of previous works

Of the paper based guides examined, only Poland and Clement (2009) and Tutin et al. (1993) clearly state their works have received external funding. All the other sources are unclear as to whether they are solely the author's work that has subsequently been published, or whether they have been financially supported. Due to this lack of information it is impossible to know if any of these are financially sustainable publications.

8.4.5 Potential cost of running as a business

The Business costings suggest a value of around £325K. These are shown in Table 8-5. Note that categories with placeholders have been given values of £1,000 and are highlighted by *italic text*. Categories with placeholder values were considered more variable (for example greater price volatility or greater effects of personal preference), and would be better served by deciding on these costs closer to the set-up of the business. Salaries are based on the average UK wage of £28.6Kpa (data gathered and adapted from the Office for National Statistics (Smith, 2017)). This salary would fall into class 1 National Insurance (NI) payments (Gov.UK, 2017b). The combined employer and employee rates are 25.8% (Gov.UK, 2017b), giving a required additional budget for wage of £19.35Kpa. For simplification, pensions were assumed static at 2% only. It was assumed that approximately 50m2 of office would be required.

Table 8-5 - The costs for each aspect of the project, with the total. All values are in £. Note that the final total has been adjusted to assume a 3 year project, and as such repeating costs have been accounted for in the total. Objects in italic text have placeholder values.

Object	**Minimal**
Pay PPPA	25,000.00
National Insurance PPPA	6450.00
Pension PPPA	500.00
Workstation per person	1500.00
Mobile phones (various)	2000.00
Office set up	*1000.00*
Business rates	6800.00
Utilities	*1000.00*
Internet	360.00
Office Rent (2ndry location)	2000.00
Insurance	*1000.00*
Total	**325,530.00**

8.4.6 Potential market size

The organisations investigated had a wide range of membership (or equivalent), from in the thousands to in the millions (Table 8-6).

Table 8-6 - Various groups of people who may be interested in an electronic plant ID guide

Organisation	**Membership**
The Royal Horticultural Society (RHS)	449,000 members (Royal Horticultural Society, 2015)
The British Broadcasting Cooperation (BBC) show Gardeners World (BBC, 2017)	2 million viewers (O'Conner, 2016)
The National Trust	4.5 million members (The National Trust, 2016)
The Botanical Society of Britain and Ireland (BSBI)	3,000 (The Botanical Society of Britain and Ireland, 2018a)
Chartered Institute of Ecology and Environmental Management (CIEEM)	5668 (CIEMM, 2018)
National Farmers' Union (NFU)	>55,000 (NFU, 2018)
British Ecological Society (BES)	6,500 (Weiss, 2018)

8.4.7 Potential available grants

There are a variety of botany focused grants available, with most being around £1,000 (Table 8-7).

Table 8-7 - The grants advertised by different groups

Organisation	Typical grant size
The Linnean Society of London	Between £1,000 and £3,000 (The Linnean Society of London, 2018)
The BSBI (BSBI specific grants)	Around £1,000 (The Botanical Society of Britain and Ireland, 2018b)
Other grants advertised by the BSBI	Between £200-£1,500 (The Botanical Society of Britain and Ireland, 2018b)
The British Ecological Society	Up to £5,000 (small projects) Up to £20,000 (Early career ecologists) (The British Ecological Society, 2018)

8.5 Discussion

The business costing indicates a required sales/price of 20,000/£18 or 10,000/£36. This would generate £360K gross, £34,470 net profit. These values are much higher than both the sales numbers and cost of typical apps, and would require competing with paper-based guides directly. As per Zhong and Michahelles (2013), whilst the google play store is mostly a long tail market (where sales drop off gradually as price increases), an island exists where much higher priced apps (>$10 in 2013) outsell lower priced ones. Zhong and Michalles (2013) suggest that these are professional apps, such as navigation aids, and as such will have been marketed differently to lower priced apps such as games. It may be possible to place the app in this 'island' and compete with paper-based keys directly. The majority of apps with a direct cost monitored gathered less than 5000 downloads, as shown in Figure 8-5. Figure 8-5 also shows no major trend towards any level of pricing being indicative of a greater number of downloads. Figure 8-3 shows the same number of downloads for most free apps. The highest number of downloads reached by a tracked app was three million which was Plant.Net (Plantnet-project.org, 2015), but it started at a much higher number of downloads already, which may well have affected the number.

The highest number of estimated sales for books shown in Table 8-4 is over 2 million for The Wildflower key (Rose and O'Reilly, 2006). While reaching only 1% of these sales would easily ensure a profit was made on this project, The wildflower key (Rose and O'Reilly, 2006) has been available for over 10 years at time of publication, rather than the three years outlined for this project. The Wildflower key (Rose and O'Reilly, 2006) was the stand-out for paper-based guides, however, with most others shown in Table 8-4 having a predicted number of sales in the 100,000-999,999 scale. While this is a large range, it must be remembered that the method to estimate these sales is relatively crude. Having sales within the same factor of 10 was therefore deemed to be similar. With both paper- and electronic- based projects, price did not appear to directly affect number of sales. Only the Flora of Great Britain and Ireland (Sell and Murrell, 1997, 2006, 2009, 2014, 2018) has a predicted 0 sales, and a much higher price of over £120 per volume. This lack of reviews and high price is because it is aimed at libraries and similar collections (e.g. herbarium libraries), who are less likely to review products on Amazon, and will be more willing to pay higher prices. Figure 8-1 and Figure 8-2 indicate that pricing for paper based projects is typically stable at the prices shown in Table 8-3, indicative of either low changes in demand (i.e. steady sales over time), or a pricing strategy resulting in stable prices, rather than one reactive to demand levels. Unfortunately price over time data was not available for Stace (1997), as changes in price with the introduction of Stace (2010c) would have provided valuable information on price stability when newer options are introduced.

This lack of any obvious link between price and number of downloads or predicted purchases suggests that greater emphasis on the price for each source is based on the coverage, quality, and target audience and market size of the source, than the physical size or other factors. This is similar to the findings shown in chapter 6, and in particular (Stagg and Donkin, 2017), where user opinion on ID guides was typically more affected by quality rather than media.

It is expected that at least some members of different botanical and/or horticultural organisations would be interested in the theoretical guide being costed up, and that different groups will have different percentages of people who would be interested. Table 8-6 shows the memberships of these different organisations. Given the botanically 'focused' nature of the BSBI, it is expected that a large percentage of membership would purchase such a guide; assuming 20% gives a potential market of 600. Assuming a similar 'focused' nature for the CIEEM gives a potential market of 1,133, and for the BES gives 1,300. A less botanically 'focused' nature was assumed for the NFU, thus a lower conversion rate of 1% and a potential market of over 5,500. While the RHS is less taxonomy focused, the membership is much higher. A 1% conversion rate would give a market size of approximately 50,000. Inclusion of garden plants would likely increase the conversion rate, but there are considerable difficulties associated with production of an ID guide to garden plants. A 1% conversion rate for National Trust members would give a market of 450,000, but without investigation of visitation numbers for different sites, it is harder to say how likely and valid that is. In order to reach all potential RHS and National Trust members, an advertising campaign would be required, increasing costs. If the app is only marketed at the University of Reading, a smaller market size is possible. At the University of Reading, the MSc Plant Diversity accepts around 8 new students each year while the MSc Species Identification and Survey Skills accepts around 20-50. Note that there will be overlaps in membership (particularly RHS and National trust membership), meaning these numbers are not exact and are probably overestimates.

Using sales/downloads of existing guides as a way to establish a potential market size gives different results. Apps that showed any change were typically downloaded less than 500 times, giving a market of less than 200 sales a year (apps were monitored over several years). ‘Standout’ apps had a much higher download count, but this was rare. Book sales were a lot higher, at between approximately 300,000 to 2,000,000. The only book investigated that had reached over 1,000,000 sales was The Wildflower key (Rose and O’Reilly, 2006). While reaching only 1% of these sales would easily ensure a profit was made on this project, The Wildflower Key (Rose and O’Reilly, 2006) has been available for over 10 years at time of publication, rather than the three years outlined for this project. Combining the high and low estimates for potential market size gives a range of approximately 20-450,000 (Note that ‘standout’ apps and books have been excluded from this range). Both extremes are less likely, with the number of downloads of previous apps suggesting that 450,000 is particularly unlikely. Even if these extremes are discounted, the potential range is still very wide, and more work should be done on establishing a potential market size. These different areas combined suggest that, as long as the price is not excessively high, and the app is high quality and there is some way for users to sample the app to confirm its quality before-hand, then an appropriate price can be charged to make a profit.

8.5.1 In app purchases

Using the in-app purchase model works nicely with the modularisation suggested in chapter 2. There are different logical ways to split the flora of the British Isles, including different families and different vice counties. Different families would provide at most 170 modules. Charging 50p would mean the complete key would cost £85 at most. This is a much lower cost per module than iFlora (Trackenberg, 2019), but a greater cost to activate the entire key. Families also provide a clear way to divide the modules. The inherent difficulty is that different families have vastly different numbers of species, and it would be unusual behaviour to charge the same amount for very large and very small families. To solve this problem 'bundles' of families could be sold, and bundles would not necessarily need to cost the same amounts. Examples of these bundles could include; grouping families with fewer species in order to reach the same number of species as a larger family, or a larger bundle of families so all tree species of the UK are covered. If the target audience has some experience with paper-based guides, they are likely to figure out that total cost when buying individual family modules is higher than most traditional keys.

Dividing based on vice county would ensure that the user could identify all species in their area. With 112 different vice counties in mainland Britain (Stace, 2010c), this would again allow charging 50p per module to generate a profit. This approach again has difficulties. Different modules for different vice counties can remain separate. This is clear for the user, as they select the module based on location. However, different vice counties, in particular geographically adjacent vice counties, will cover similar sets of taxa. Users may discover this, and may question the value if few/no taxa have been added to the guide overall. Having a different modules for each vice county also requires a bespoke key and database to be constructed, which would be time consuming. The modules for different vice counties could also behave in an 'additive' manner; this has the same risks associated with geographically adjacent vice counties adding few taxa to the key, and has the risk that the value of location specificity built into vice-county derived modules is lost. In order to maintain pricing sanity compared to existing paper-based keys, an option to purchase the entire set of modules at once for a fee of, say, £30 should be made available. This is both similar to the cost to purchase most paper-based guides, and can be marketed as a ‘good value bundle’. Further work would be needed to decide the best method to implement this total unlock fee, as this could include; offering it as a one-time purchase along with individual modules, unlocking the whole key when enough modules have been purchased, or arranging and pricing modules so this fee is reached when all modules have been purchased.

Further work would be required including surveying potential users to determine how many modules they would be likely to buy, and how much they would be willing to pay for these modules. It may be that the ability to create a semi-custom app with only modules they consider important, for a lower fee than a traditional guide may be more popular than anticipated, and thus more likely to create a profit. The opposite may also be true, potential users may not be interested in a piecemeal guide. If the in-app purchase model is chosen, the ‘basic’ version would need to be free in order to avoid any potential risks similar to micro-transactions.

8.5.2 Funding production

The costs shown in Table 8-5 are high. They are, however, of the same scale as found by Craigmile (2015). It is suspected that the more expensive apps surveyed by Craigmile (2015) are likely to be ‘headline’ apps, such as Bloons (ninja kiwi, 2018) or the Microsoft Office mobile suite (Microsoft Corporation, 2018f, 2018a, 2018e, 2018d, 2018c, 2018b). The app in this project is clearly not as ‘headline’ as these apps; however, it is a vastly different style, requiring a much greater research and data preparation time. This suggests that while high, the £325K costing is at the appropriate scale.

There are different ways to lower the total costs. If costs are lowered, then the either the number of sales, or the total price can be reduced.

The wages suggested were chosen based very simply on the average UK wage. The total wages could be reduced in several ways: The total number of staff could be reduced; the number of hours of work could be reduced; parts of the project could be undertaken by PhD students; or the nature of the project, and therefore the requirement for the normal approach to wages could be modified. It would be possible to run this project with fewer staff. Reducing the staff level by one reduces the total cost by £95,850 (with a resultant total cost of £229,680). This means the price of the guide could be reduced to £12 for a profit to be made (assuming 20,000 sales again for simplicity of comparison). This approach has the advantage that remaining staff would still be full time. Reducing the number of hours staff are required to work on the project would have the same cost-adjusting effect as reducing the number of staff. It has two advantages over reducing the number of staff: the range of staff skills is maintained, and the amount of work done by staff can be more 'fine-tuned'. It may be possible that some of the work could be undertaken by PhD students, as combined stipend and fees can be less than the wages suggested here. While this is not a reduction in staff levels, as-such, PhD students typically have other commitments relating to their PhD reducing the amount of time they will have available for this project. It may also be possible to crowd-source some of the work, such as gathering morphological information. This approach has previously been used in the Wildflower Identification website and apps (S. Sullivan, 2019).

If adjusting the staffing level is the approach chosen to generate cost reduction, it must be noted that the suggestion of three full time staff was based on experience from this project. This may seem excessive; it was possible to produce and test a working app within approximately two years with a single member of staff equivalent, but this was only to *Equisetum* L.. Creating a key to the flora of the British isles involves a much greater number of species, and a much greater number of data sources (i.e. there are a greater number of books to extract data from), thus it will require a greater number of man hours to create the database if no method of automation is discovered. Leafsnap (Columbia University, 2015) is developed by volunteers and acknowledges 62 contributors over four institutions (Leafsnap, 2011). Stace (2010c) is authored by a single person, but acknowledges the help of a considerable number of people, as does Streeter (2009). Adjusting the staff level down therefore risks producing the key in the same time frame, or reducing the quality of the key as fewer man-hours are available.

In order to reduce location costs (such as office rent/tax rates, office set up, etc.), it may be possible to undertake this project entirely remotely. In order to establish the total cost with no office, the following were removed from the total: Office set up, business rates, utilities, Internet, office rent, and insurance. Assuming 3 staff at full time, this reduces the total cost to £291,050. Assuming the reduced staff costs are incorporated as well, is further reduced to £195,200. This gives a required price of £15 (no office) and £10 (no office and reduced staff) (again assuming 20,000 sales for ease of comparison). If this approach to cost reduction is chosen, it must again be noted that this will induce difficulties including that no central location will highly reduce the potential for group working and collaboration between staff.

Care must therefore be taken if cost reduction methods are employed. As has been discussed, they are likely to induce compromise in the quality of the product; either from reduced ease of collaboration, risk of reduced quality of work, reduced number of test devices, or any number of other areas. These risks to the quality of the product are still present in a fully funded project, but the intention would be that enough funding would allow work to happen with these risks reduced as much as possible.

8.5.3 Further development

Given the nature of the work required, some of the grants mentioned earlier may be available. Given the typical totals available, however, it is not going to make a significant difference in total cost of app production. If grant funding is going to be sought, therefore, it will need to be during the initial stages of the project to get it off the ground; once there are any large costs incurred, the relatively small amounts of money these grants make available are unlikely to make significant differences. It is also worth noting that many of these grants have conditions which may mean they are not available for projects of this nature; the British Ecological Society grants are awarded to individuals only (The British Ecological Society, 2018), but here it has been suggested a three person team would be required. Grants may specify that products of the work are made available for free, meaning other forms of monitisation are no longer possible.

Continued work in the area will likely increase the revenue generated by this project; a study has suggested continually developed apps are frequently potentially better received (Danova, 2015). Many sources suggest that frequently updating your app is beneficial as users are receptive to bugs being fixed rapidly, it also allows the developer to adapt the app to the user far more specifically, and because it keeps the app in the mind of the user (Danova, 2015; Yarmosh, 2016; Evgeniy, 2017; Genty, 2017). It is well known that taxonomy is not a static subject, and that the exact contents of taxonomic groups changes over time. With traditional paper-based guides, this in an easy (but expensive to the user) problem to solve – update the text then print and sell a revised edition of the book. Stace has published three versions of the 'New Flora to the British Isles', with a fourth to be published (Stace, 1991, 1997, 2010c, 2018), Each new version is a revision and update of the previous text (generally incorporating new alien species and taxonomic revisions), rather than a full re-write, yet has been sold. Sales of previous editions will have provided an income, assisting development of later editions. With apps, most users expect updates for free. The question then becomes how long these updates are provided for, and how well the taxonomy in the app is maintained? Taxonomic revisions are rare compared to the rate at which most apps are updated, but given the higher app cost suggested throughout this chapter, users may expect updates for a considerable period. This will necessitate some app maintenance, which will have an associated cost. It seems necessary that some investigation into this problem is undertaken before a business of this nature is created.

It may simply be possible to develop the app further for free, using a non-profit making system. The Wildflower Identification Website (S. K. Sullivan, 2019) and associated apps (Wildflower Search, 2019) cover a considerable amount of the flora of Northern America, and are available for free. The data and images used in their website and apps is comes from a variety of sources, including crowdsourcing of both images and data. Crowdsource data would reqire maintenance to ensure quality (The Wildflower Identification Website (S. K. Sullivan, 2019) notes that you cannot save data without permission), and would require a critical mass of interested participants to produce enough data. Given the costs of website maintainace and potential costs of data maintannace, this approach is not necesasarly cost free. It may, however, result in a much lower overall cost. This would mean a grant may cover all costs, or the cost to the user can be significantly reduced.

8.6 Conclusions

While it may be possible to create a profit-making business creating and selling an electronic ID guide to the UK flora, the results of this chapter suggest that financial success is unlikely. The potential market is not large enough that similar revenue to the top grossing apps is guaranteed, but the costs and difficulties in production would be higher than a ‘hobby’ app. Various cost reduction/income supplementing methods have been examined, but none are large enough to make a realistic difference. It is probably a wiser approach to produce this key as part of a much wider work, such as part of a large company's portfolio, or part of a University research output.

9 Discussion and conclusions

9.1 Thesis contributions

The first, and slightly surprising, contribution from this thesis is the investigation into database sizes on Android. While there has been extensive work into SQL databases, and optimising apps on Android, there appeared to be no investigations providing actual numbers on Android SQLite database sizes. Chapter 2 may be the first. With some further work on more modern midrange smartphones, and incorporation of transactions (as per Feinstein (2017)), this may be a publishable work.

While it is not fully novel work, the work on automatic extraction effectively confirms that despite the appearance of more-regular language when compared to natural language, automatic extraction of data from paper-based sources is going to require advanced NLP techniques.

The creation of the app and the feedback from users on rounds of testing shows clearly that apps of this style are preferred by users over their traditional paper-based counterparts.

The final chapter confirms that production of such an app is never going to be financially successful, and will have to rely on grants or other forms of support.

9.2 General discussion

There is clearly a demand for good quality smartphone-based ID guides. There are already a number of extant ID guides (see Table 1-1, which shows a sample of the current availability). Some of these, such as Stace (1991, 1997, 2010c) or Streeter (Streeter *et al.*, 2009, 2018c), have had multiple editions. As was demonstrated in chapter 8, there is a potential market of thousands of people for such an app (see Table 8-6). This market is clearly active, ID-guide apps in the Google Play store (Google, 2017a), typically gather around 3,000 lifetime downloads (see Figure 8-3 and Figure 8-4). Plant.NET (Plantnet-project.org, 2015) has done exceptionally well and is currently at around 3 million downloads (see Figure 8-3)! However in chapter 1 at the time this project began it was felt that the smartphone-based options were not taking full advantage of their platform. The work presented here therefore aimed to examine smartphone-based multi-access ID guides: investigating the best approach to gather data for such guides, testing their potential best design, and determining whether or not production of these guides would return a profit.

An accurate database containing information about each taxon was required for the app to work; the app searches through this database to produce a list of potential identifications. The variety of available pre-existing guides was a potentially very rich data source. There were two ways to gather these data; automatically or manually. Previous attempts at automatic data extraction from floras have tended towards using NLP techniques (Cui *et al.*, 2010; Wood *et al.*, 2011). The database format required for my project suggested that this process was unnecessarily complex, and that the regularity of botanical descriptions (noted many times before (Taylor, 1995; Mary McGee Wood *et al.*, 2004; Cui *et al.*, 2010)), and the higher power of modern computers lead to an attempt to use brute force and pattern matching to extract the data. This did not work. When viewed in the context of a whole guide (rather than individual descriptions), the text was not as regular as hoped. This highlighted numerous issues including: aspects included in family descriptions are true for the whole family and are often not carried forwards to species descriptions, and descriptions can include relational information requiring the examination of the description of a different taxon. A system involving a 'look back', similar to Cui et al. (2010), and merging data from multiple floras, similar to Lydon et al. (2003), will help here, as help fill in as many gaps as possible. It is very clear that books are written for humans by humans (for now). A manual approach, involving a human, was therefore taken. This process was slow, but allowed data to be combined from several sources with different lists of taxa. This process was based on the 'by hand' process described by Lydon et al. (2003), but expanded on it. In their work, taxa were present in all sources used; here, some taxa were not included in some sources, meaning data had to be inferred. It seems reasonable to assume that continuation of this method, covering the entire national flora and using all possible sources (including, but not limited to: Poland & Clement, 2009; Sell & Murrell, 1997, 2006, 2009, 2014, 2018; Stace, 2010; Streeter et al., 2016), would be successful. The amount of labour and funding required to do this thoroughly may, unfortunately, be prohibitive. An interesting area of investigation that may follow is the 'tracking' of various taxa; the UK as a whole, and most counties have had multiple versions of floras written about them, giving 'snapshots' of what was recorded where and when. Extraction of the data from all the various sources will allow a much more detailed examination of their data.

Testing whether data could automatically be extracted from floras was not a complete failure: it provided a database that (while not tremendously accurate) held data about the entire UK flora. As Android phones are lower power than desktop computers (see section 2.1), this database was used to check how big a database Android phones could work with. Figure 2-2 and Figure 2-3 showed that databases of about 1.5 million cells load quickly enough to be called ‘user friendly’. These two figures also show that you can use considerably bigger databases, but eventually these will cause lower power devices to crash, rendering the app unusable. While smartphones are currently not able to hold a national flora, when the *Equisetum* database was created, there proved to be enough room for a complete section of the flora. Thus, to create a national flora, the data would need to be ‘sectioned’ or ‘modularised’, to keep the databases small enough that smartphones can run the ID-guide app at an acceptable speed.

The decision to stick with a multi-access key design was made very early on. They have clear advantages over single access keys: it is almost impossible to render them un-usable (compared to single-access keys where an unresolvable couplet risks the entire key), the clearest features can be input into the key first (Tardivel and Morse, 1996; Drinkwater, 2009), and questions on unclear or missing features can be avoided entirely (Stucky, 1984; Edwards and Morse, 1995; Tardivel and Morse, 1996; Drinkwater, 2009; Penev *et al.*, 2009). Multi-access guides are simply easier for users than single-access guides, they are more forgiving and they do not require all the data to work correctly. Despite the difficulties with multi-access keys (Edwards and Morse, 1995; Tardivel and Morse, 1996; Drinkwater, 2009), there has been no indication they are the wrong choice anywhere in this project. The number of multi-access compared to other designs of smartphone-based ID guides backs this decision up. See Figure 5-2 through to Figure 5-10 for a set of examples of smartphone-based multi-access guides, compared to Figure 1-3 which is one of very few smartphone based single-access keys found during this project), and Figure 1-4 which is one of few photo-recognition smartphone-based ID guides found.

While none of the available smartphone-based multi-access ID guides was perfect, the variety that are available allowed determination of the most useful design. As can be seen in the figures in chapter 5, there were several themes running through the various designs. There were also two strong ideas about optimal app design running through the literature: keep it simple, and keep it consistent (Babich, 2018b, 2018a; Cooper, Reimann, Cronin, & Noessel, 2014; Creative Blog Staff, 2012; Doris, 2017; Gove, 2016; Natoli, 2014; Sam, 2017; So, 2017; Teo Siang, 2018; Tidwell, 2010; "User Interface Design Basics," 2018). 'Keep it simple' was important for this project, one of the key aims was to produce a guide that users found easier. These themes and ideas were used as the basis to create the first version of the app (seen in Figure 5-18, Figure 5-19, and Figure 5-20). Given the favourable response to even the first version of the app (see chapter 7 and Appendix G), this approach clearly was successful. During all rounds of testing and evaluation there was no feedback suggesting that a significant redesign was required; feedback about design and database accuracy only ever called for refinements. The first version of the app incorporated the 'four way split', where users were a single single-access style question was asked. This split lead the user to their chosen set of taxa, allowing both the questions in the search page to remain more relevant to their specimen, and the database to be split into small-enough parts that the app would load quickly.

The feedback generated when evaluating the app (see chapter 7) gave good indications of why people preferred the app over paper-based guides. Questions five and seven of both questionnaires used gave the volunteers opportunities to suggest specifically what they liked about the app and any other points they felt were useful to mention. The feedback here showed that the app was felt to be faster, easier to use and understand, and that the improved portability of a phone over a paper-based flora was beneficial.

With the number of extant apps and guides (see Table 1-1), the number of apps that are clearly making a lot of profit (see Table 8-2), and with how well this app was received, the financial suitability of continuation of this project was considered. Unfortunately, despite the clear presence of a market, it is not a big enough market to turn a profit. The market size established here is a few thousand (see Table 8-4); for example Tinder (2018) has been downloaded over 100 million times at time of writing.

With the amount of work required to make an app like this, making a profit will require a market several factors of 10 larger. This is backed up by the 'large' smartphone-based guides clearly showing that they are supported by governmental grants. Plant.NET (2017) is funded by CIRAD, INRA, INRIA and IRD, and the Agropolis Fondation (Plantnet-project.org, 2017). ArbolApp (Real Jardin Botanico, 2015) is an initiative of the Consejo Superior de Investigaciones Científicas funded by Fundación Española para la Ciencia y la Tecnología (Arbolapp, 2018). Large paper-based guides are also often grant supported, such as Poland and Clement (2009) who were supported by eight separate organisations (including the BSBI) or Tutin et al. (Tutin et al., 1964, 1968, 1972, 1976, 1980, 1993) who were funded by The Royal Society, the Botanical Research Fund, and the Royal Horticultural Society, amongst others.

In the introduction a few issues with photo-recognition ID guides were raised. The two most important of these were that smartphone cameras were not as good at resolving small details as the human eye (whether or not the eye is assisted with a jeweller's loupe), and that users are not capable of taking 'good' photographs for identification. The first issue will eventually be resolved (Xiaomi are planning on producing a smartphone with a 48 megapixel camera (Jansen, 2018)). Without details on the exact specifications on all smartphone cameras, and without knowledge of the smallest detail that will be required for species identification, it is impossible to say when smartphones will be able to resolve sufficient detail. The second problem is far more difficult to solve. It will be impossible for the ID-guide to identify to species level if the user has inadvertently left out the necessary detail from photographs. As the required details are not in a consistent place at a consistent time on all taxa, there is no easy way to train users on what to photograph, without including as many instructions as are in a question-based ID guide. Thus, for 'in the field' work, question-based ID guides are extremely likely to remain the only reliable method of identification to species level.

In the introduction, one of the reasons raised for this work was that it was felt that at the time this project was started, Android was very underprovided for with good quality ID guide apps. It was felt that this was a wasted opportunity, hence the preliminary work (Bewsey, 2014) was began. The situation has improved since this point, and as demonstrated through this thesis, there are a number of apps now available. Therefore, while every attempt has been made to include and refer to every multi-access smartphone-based ID guide available, it must be stressed that new apps are being constantly produced (even in botany!). Future work should ensure they undertake a careful re-examination the available literature to make sure nothing novel since this project has been missed.

9.3 Limitations

The use of Equisetaceae as the case study group was a notable limitation to this study. The decision to limit the data during app testing to the Cryptogamae was taken after it was found that the automatic extraction methods were unlikely to properly produce an accurate database. The scope was again limited afterwards to the Equisetaceae after the four way split was found to be successful. The intention of the further scope-restriction was to allow enough time to fully extract and merge data from the sources, and to test the database. Unfortunately the decision to limit the scope to the Equisetaceae came late enough that, when combined with budget limitations, it would have been a significant challenge to gather and maintain live specimens of every taxa present in the British Isles. With the very limited ranges and population sizes of some of the smaller taxa (*E. x mchaffieae* C.N. Page at time of recording was limited to a small bay in a single loch (Page *et al.*, 2007)), time and fore-thought would have been required to ensure there was enough of each taxon to take. Herbarium specimens had clear advantages in availability and longevity, with the downside that specimens are clearly not quite as found in the wild. Herbarium specimens from **RNG** and **E** were therefore used. Every attempt was made to obtain a specimen of *E. x willmotii* C.N. Page; it appeared that the only specimen was held at Glasnevin herbarium **DBN**, who did not respond to a request for its loan. A future project working along the same lines should, if possible, aim to include live specimens for all taxa. This will ensure that all ID guides are tested under as accurate conditions as possible.

This project did not examine all possible designs of smartphone-based ID guide. Single-access and photo-recognition designs were deliberately excluded. There were a number of designs or design parts of multi-access keys (such as the multi-layer list design) that did not make it to the final design (see chapter 5 for examples of these).

The behaviour of the search function limited some of the possibilities that could be explored. The database had columns representing feature variants and cells containing binary data as to whether the taxa displayed that variant. The search function then simply selected each taxon that matched at all columns specified by the user. This meant all results were considered equally 'valid', and no probabilistic data could be generated as to which taxon in the results was most likely. It also meant that all missing data (i.e. questions that were un-answered) was simply ignored.

9.4 Future work

The future work suggested in the previous chapters can now be bought together to form a clearer image of the work that should be done. Any future work should begin with a through re-examination of available technology (both soft- and hard-ware). Cameras and image recognition may have advanced enough that photorecognition methods may be more successful, for example. Future work needs to examine the software, as there are a number of different designs for keys that were discounted, but may be preferred by users. Finally, future work should incorporate transactions (Feinstein, 2017) and more modern smartphones to establish a more-accurate maximum database size.

Future work on the contents of the key is more important. The work here is restricted to the *Equisetum*, a very small section of the flora. More modules should be written, at the very least ensuring that the four-way split described in chapter 5 can be re-introduced. When further modules unrelated to this four-way split are written, future work is going to need to examine the best way to incorporate them into the app. The only apparent way to do so currently is to have a 'key to modules', but it is not clear whether this is the best design.

Finally, if work on this project is going to continue, methods of funding are going to have to be investigated. The work in chapter 8 suggests that there is no way to successfully fund development of the app allowing it to cover the whole of the UK, assuming current programming speeds and market size. Alternatives such as costing up development of a single module and investigating whether relevant organisations can fund their development (e.g. a fern society for the fern modules) should be investigated, along with methods to r educe the costs of programming (either through increasing the speed or reducing the cost of labour), or increasing the market size (through adaptation for markets such as the RHS).

9.5 Conclusions

An alternative approach to production of ID guide apps and their required databases is thus suggested. It is reasonable to assume that novel natural language taxonomical descriptions (e.g. ID guides, monographs, taxonomic revisions) will have some form of electronic storage about each taxon. Even if these data are limited and not cohesive, it is strongly suggested that they are expanded on, and urge publication of a complementary species x character database with such publications. If such a database is available, and the work here on app design is used as an exemplar, then production of an associated ID-guide app is not a large step further (it is, of course, acknowledged that more work than production of the paper-based guide alone will be required). When funding is sought to aid production of novel paper-based guides, or other taxonomic revisions of taxa, then production of at least the species x character databases should be one of the requirements. It is suggested that an approach to this is standardised at a relevant TDWG (2018) meeting or similar, but it is acknowledged that this risks falling into the 'add another standard to the pile' trap (Munroe, 2011) and care should be taken. This seems the only realistic way to produce the datasets, and thus the apps. Given how much more useful and adaptable data stored in this style would be, this requirement should have been implemented long ago.

This work has shown that it is very possible to produce good smartphone-based ID guides. These guides are preferred by users over the paper-based counterparts. These guides will be with the users at all times. We are in a smartphone based society; it is time that taxonomy made use of such available and versatile tools.

10 Reference list

Adobe (2015) 'Adobe Acrobat DC Pro'. Adobe Systems. Available at: https://acrobat.adobe.com/uk/en/acrobat/acrobat-pro.html (Accessed: 17 July 2019).

Adobe (2018) *Discover the Creative Cloud experience*. Available at: https://www.adobe.com/uk/creativecloud/plans.html (Accessed: 28 October 2018).

Agile Manifesto (2019) *Principles behind the Agile Manifesto*, *Agile Manifesto*. Available at: https://agilemanifesto.org/principles.html (Accessed: 15 April 2019).

Amazon UK (2017) *Amazon search results for 'Peter Sell Gina Murrell'*. Available at: https://www.amazon.co.uk/s/ref=nb_sb_noss?url=search-alias%3Daps&field-keywords=Peter+sell+gina+murrell&rh=i%3Aaps%2Ck%3APeter+sell+gina+murrell (Accessed: 11 April 2018).

Amazon UK (2018a) *Flora Europaea (5 Volume Set)*. Available at: https://www.amazon.co.uk/Europaea-Heywood-Burges-Valentine-Walters/dp/B009C7X9D2/ref=sr_1_11?ie=UTF8&qid=1540565506&sr=8-11&keywords=flora+europaea (Accessed: 26 October 2018).

Amazon UK (2018b) *Search results for 'flora europaea' on Amazon UK*. Available at: https://www.amazon.co.uk/s/ref=nb_sb_ss_c_1_8?url=search-alias%3Daps&field-keywords=flora+europaea&sprefix=flora+eu%2Caps%2C130&crid=2VIFJ47IBXFB0&rh=i%3Aaps%2Ck%3Aflora+europaea (Accessed: 26 October 2018).

Amazon UK (2018c) *Search results for 'sell murrell' on Amazon UK*. Available at: https://www.amazon.co.uk/s/ref=nb_sb_noss_1?url=search-alias%3Daps&field-keywords=sell+murrell (Accessed: 26 October 2018).

Anderton, K. (2018) *The Ongoing Controversy Of Microtransactions In Gaming [Infographic]*, *Forbes*. Available at: https://www.forbes.com/sites/kevinanderton/2018/03/07/the-on-going-controversy-of-microtransactions-in-gaming-infographic/#146acb231d9c (Accessed: 31 October 2018).

Android (2017) *Android Developers*. Available at: https://developer.android.com/index.html (Accessed: 10 August 2017).

Android Developers (2015) *Saving Data in SQL Databases*, *Android Developers*. Available at: http://developer.android.com/training/basics/data-storage/databases.html (Accessed: 9 November 2015).

Android Developers (2016a) *Databases in Android*, *Android Developers*. Available at: https://developer.android.com/guide/topics/data/data-storage.html#db (Accessed: 15 August 2017).

Android Developers (2016b) *Layouts*, *Android Developers*. Available at: https://developer.android.com/guide/topics/ui/declaring-layout.html (Accessed: 14 August 2017).

Android Developers (2017) *Storage Options: Using Databases*, *Android Developers*. Available at: https://developer.android.com/guide/topics/data/data-storage.html#db (Accessed: 19 January 2018).

Android Developers (2018a) *ActionBar*, *Android Developers*. Available at: https://developer.android.com/reference/android/app/ActionBar (Accessed: 11 December 2018).

Android Developers (2018b) *Beta test your app with users to get invaluable early feedback*, *Android Developers*. Available at: https://developer.android.com/distribute/best-practices/launch/beta-tests (Accessed: 9 August 2015).

Android Developers (2018c) *Fundamentals of Testing*, *Android Developers*. Available at: https://developer.android.com/training/testing/fundamentals (Accessed: 6 May 2018).

Android Developers (2018d) *Native Android apps, without the installation*, *Android Developers*. Available at: https://developer.android.com/topic/google-play-instant/ (Accessed: 15 December 2018).

Android Developers (2019a) *Application Fundamentals*, *Android Developers*. Available at: https://developer.android.com/guide/components/fundamentals?hl=en (Accessed: 18 May 2019).

Android Developers (2019b) *Save data using SQLite*, *Android Developers*. Available at: https://developer.android.com/training/data-storage/sqlite (Accessed: 18 May 2019).

Apple (2017) *Human Interface*. Available at: https://developer.apple.com/design/ (Accessed: 8 November 2017).

Apple (2019) *iOS 12 More power to you*, *Apple*. Available at: https://www.apple.com/uk/ios/ios-12/ (Accessed: 17 July 2019).

Apple Developers (2018) *Choosing a Business Model*. Available at: https://developer.apple.com/app-store/business-models/ (Accessed: 8 November 2018).

Apple INC (2018) *Shazam*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.shazam.android&hl=en_GB (Accessed: 14 December 2018).

Apptent Studios (2014) 'Fungitron - mushroom guide'. Google Play Store. Available at: https://play.google.com/store/apps/details?id=org.gfx54b.android.fungitron&hl=en_GB (Accessed: 6 June 2015).

Arbolapp (2018) *About Arbolapp*. Available at: http://www.arbolapp.es/en/about-arbolapp/ (Accessed: 20 December 2018).

AUT Ventures Limited (2018) 'NZ Trees'. Google Play Store. Available at: https://play.google.com/store/apps/details?id=com.nztrees&hl=en (Accessed: 12 July 2019).

Babich, N. (2017) *The top 5 User Testing Methods*, *Adobe Blog*. Available at: https://theblog.adobe.com/the-top-5-user-testing-methods/ (Accessed: 8 May 2018).

Babich, N. (2018a) *A Comprehensive Guide To Mobile App Design*, *Smashing Magazine*. Available at: https://www.smashingmagazine.com/2018/02/comprehensive-guide-to-mobile-app-design/ (Accessed: 18 November 2018).

Babich, N. (2018b) *The 4 Golden Rules of UI Design*, *Nick Babich*. Available at: https://theblog.adobe.com/4-golden-rules-ui-design/ (Accessed: 16 November 2018).

Bakshi, K. (2018) *Deep Dive Into A/B Testing With Firebase Remote Config*, *AndroidPub*. Available at: https://android.jlelse.eu/deep-dive-into-a-b-testing-with-firebase-remote-config-6213e95bd024 (Accessed: 9 May 2018).

Banas, D. (2013a) *Android Development 10*, *New Think Tank*. Available at: http://www.newthinktank.com/2013/06/android-development-tutorial-10/ (Accessed: 19 May 2019).

Banas, D. (2013b) *Android Development 14*, *New Think Tank*. Available at: http://www.newthinktank.com/2013/06/android-development-14/ (Accessed: 19 May 2019).

Bank of England (2017) *Inflation Calculator*. Available at: http://www.bankofengland.co.uk/education/Pages/resources/inflationtools/calculator/default.aspx (Accessed: 26 October 2017).

Basch, C. E. (1987) 'Focus Group Interview: An Underutilized Research Technique for Improving Theory and Practice in Health Education', *Health Education & Behavior*, 14(4), pp. 411–448. doi: 10.1177/109019818701400404.

BBC (2017) 'Gardeners' World'. United Kingdom: British Broadcasting Corporation. Available at: http://www.bbc.co.uk/programmes/b006mw1h (Accessed: 28 July 2017).

Beentje, H. (2010) *The Kew Plant Glossary: An Illustrated Dictionary of Plant Identification Terms*. Royal Botanic Gardens Kew.

Benington, H. D. (1956) 'Production of Large Computer', *Proceedings, ONR Symposium on Advanced Programming Methods for Digital Computers*, pp. 15–17.

Bertolino, A. (2007) 'Software Testing Research: Achievements, Challenges, Dreams', *Future of Software Engineering (FOSE '07)*, (September), pp. 85–103. doi: 10.1109/FOSE.2007.25.

Bewsey, A. (2014) *An easy to understand electronic key to Whiteknights campus flowering plant families*. University of Reading.

Bhagat, H. R. and Bajaj, K. (2018) *The 18:9 display dilemma: Will the new smartphone screens make our lives easier or do the opposite?*, *The Economic Times*. Available at: https://economictimes.indiatimes.com/magazines/panache/the-189-display-dilemma-will-the-new-smartphone-screens-make-our-lives-easier-or-do-the-opposite/articleshow/62662023.cms (Accessed: 11 December 2018).

Bhat, N. (2017) 'Near Optimal A-B Testing', *Columbia Business School Research Archive*, pp. 1–50. Available at: https://www8.gsb.columbia.edu/researcharchive/articles/25462 (Accessed: 12 July 2019).

Bhuarya, P., Nupur, S., Chatterjee, A. and Thakur, R. S. (2016) 'Mobile Application Testing: Tools & Challenges', *International Journal Of Engineering And Computer Science*, 5(10), pp. 18679–18681. doi: 10.18535/ijecs/v5i10.57.

Binkley, D., Davis, M., Lawrie, D. and Morrell, C. (2009) 'To camelcase or under_score', *2009 IEEE 17th International Conference on Program Comprehension*, pp. 158–167. doi: 10.1109/ICPC.2009.5090039.

Birrel, N. D. and Ould, M. A. (1988) *A practical handbook to software development*. Cambridge: Cambridge University Press.

Bowen, H. J. M. (1968) *The Flora of Berkshire*. Hollywell Press Oxford.

Bowes, J. (2015) *Kanban vs Scrum vs XP – an Agile comparison*, *TPX Manifesto*. Available at: https://manifesto.co.uk/kanban-vs-scrum-vs-xp-an-agile-comparison/ (Accessed: 17 April 2019).

Brown, A. (2018) *Google confirms Pixel 3 will feature the return of squeezable sides*, *T3 Smarter Living*. Available at: https://www.t3.com/news/google-confirms-pixel-3-will-feature-the-return-of-squeezable-sides (Accessed: 14 December 2018).

Brutlag, J. (2009) 'Speed matters for google web search', *Google*, p. 2009. Available at:

https://ai.googleblog.com/2009/06/speed-matters.html.

BSAF (2013) ‘Weed ID’. Google Play Store. Available at: https://play.google.com/store/apps/details?id=com.weedid&hl=en_GB (Accessed: 4 May 2015).

Burgess, M. (2018) *What is GDPR? The summary guide to GDPR compliance in the UK*, *Wired UK*. Available at: https://www.wired.co.uk/article/what-is-gdpr-uk-eu-legislation-compliance-summary-fines-2018 (Accessed: 30 October 2018).

Burks, A. W. (1947) ‘Electronic Computing Circuits of the ENIAC’, *Proceedings of the IRE*, 35(8), pp. 756–767. doi: 10.1109/JRPROC.1947.234265.

CamelCamelCamel (2017) *camelcamelcamel uk*. Available at: https://uk.camelcamelcamel.com/ (Accessed: 16 October 2017).

Card, S. K., Robertson, G. G. and Mackinlay, J. D. (1991) ‘The information visualizer, an information workspace’, *Proceedings of the SIGCHI conference on Human factors in computing systems Reaching through technology - CHI ’91*, pp. 181–186. doi: 10.1145/108844.108874.

Castensøe-Seidenfaden, P., Reventlov Husted, G., Teilmann, G., Hommel, E., Olsen, B. S. and Kensing, F. (2017) ‘Designing a Self-Management App for Young People With Type 1 Diabetes: Methodological Challenges, Experiences, and Recommendations’, *JMIR mHealth and uHealth*, 5(10), p. e124. doi: 10.2196/mhealth.8137.

Chase, M. W., Christenhusz, M. J. M., Fay, M. F., Byng, J. W., Judd, W. S., Soltis, D. E., Mabberley, D. J., Sennikov, A. N., Soltis, P. S., Stevens, P. F., Briggs, B., Brockington, S., Chautems, A., Clark, J. C., Conran, J., Haston, E., Möller, M., Moore, M., Olmstead, R., Perret, M., Skog, L., Smith, J., Tank, D., Vorontsova, M. and Weber, A. (2016) ‘An update of the Angiosperm Phylogeny Group classification for the orders and families of flowering plants: APG IV’, *Botanical Journal of the Linnean Society*, 181(1), pp. 1–20. doi: 10.1111/boj.12385.

Chen, T., Fenyo, K., Yang, S. and Zhang, J. (2018) *Thinking inside the subscription box: New research on e-commerce consumers*, *McKinsey & Company High Tech*. Available at: https://www.mckinsey.com/industries/high-tech/our-insights/thinking-inside-the-subscription-box-new-research-on-ecommerce-consumers#0 (Accessed: 28 October 2018).

Cheng, F. (2016) *Get small, go big: Meet the next-gen Snapdragon 835*, *Qualcomm Blog*. Available at: https://www.qualcomm.com/news/onq/2016/11/17/get-small-go-big-meet-next-gen-snapdragon-835 (Accessed: 1 August 2018).

Christodoulou, M. D., Battey, N. H. and Culham, A. (2018) ‘Can you make morphometrics work when you know the right answer? Pick and mix approaches for apple identification’, *PLoS ONE*, 13(10), pp. 1–17. doi: 10.1371/journal.pone.0205357.

CIEMM (2018) *CIEEM List of Current Members*. Available at: http://events.cieem.net/home/Member-Directory.aspx (Accessed: 28 November 2018).

Clemons, E. K. (2009) ‘Business Models for Monetizing Internet Applications and Web Sites: Experience, Theory, and Predictions’, *Journal of Management Information Systems*, 26(2), pp. 15–41. doi: 10.2753/MIS0742-1222260202.

CNet (2018) *HP EliteDesk 800 G1 - Core i5 4570 3.2 GHz - Monitor : none. Series Specs*, *CNet*. Available at: https://www.cnet.com/products/hp-elitedesk-800-g1-core-i5-4570-3-2-ghz-monitor-none-series/specs/ (Accessed: 3 August 2018).

Cohn, M. (2007) *Differences Between Scrum and Extreme Programming*, *Mountain Goat Software*. Available at: https://www.mountaingoatsoftware.com/blog/differences-between-scrum-and-extreme-programming (Accessed: 17 April 2019).

Columbia University (2015) ‘Leafsnap’. iTunes. Available at: https://itunes.apple.com/gb/app/leafsnap/id430649829?mt=8 (Accessed: 17 April 2015).

Cooper, A., Reimann, R., Cronin, D. and Noessel, C. (2014) *About Face*. Indianapolis: Wiley.

Cope, J. S., Corney, D., Clark, J. Y., Remagnino, P. and Wilkin, P. (2012) ‘Plant species identification using digital morphometrics: A review’, *Expert Systems with Applications*. Elsevier Ltd, 39(8), pp. 7562–7573. doi: 10.1016/j.eswa.2012.01.073.

Craigmile, N. (2015) *Cost to build a mobile app: a survey*, *Clutch*. Available at: https://clutch.co/app-developers/resources/cost-build-mobile-app-survey-2015 (Accessed: 29 October 2018).

Crawley, M. (2005) *The Flora of Berkshire: With Accounts of Charophytes, Ferns, Flowering Plants, Bryophytes, Lichens and Non-lichenized Fungi*. Harpenden: Brambleby Books.

Creative Bloq Staff (2012) *The 10 principles of mobile interface design*, *CB Creative Bloq Art and Design Inspiration*. Available at: https://www.creativebloq.com/mobile/10-principles-mobile-interface-design-4122910 (Accessed: 10 December 2018).

CRinUS (2012) ‘Key: Plant Families’. Google Play Store. Available at: https://play.google.com/store/apps/details?id=com.bim.key.plantfamily198&hl=en (Accessed: 3 February 2015).

Cui, H. (2010) ‘Semantic annotation of morphological descriptions: an overal strategy’, *BMC Bioinformatics*, 11, p. 278.

Cui, H., Boufford, D. and Selden, P. (2010) ‘Semantic Annotation of Biosystematics Literature WithoutTraining Examples’, *Journal of the American Society for Information Science and Technology*, 61(3), pp. 522–542.

Dahdul, W. M., Cui, H., Mabee, P. M., Mungall, C. J., Osumi-Sutherland, D., Walls, R. L. and Haendel, M. A. (2014) ‘Nose to tail, roots to shoots: Spatial descriptors for phenotypic diversity in the Biological Spatial Ontology’, *Journal of Biomedical Semantics*, 5(1), pp. 1–13. doi: 10.1186/2041-1480-5-34.

Dallwitz, M. J. (1974) ‘A flexible computer program for generating identification keys’, *Systematic Biology*, 23(1), pp. 50–57. doi: 10.1093/sysbio/23.1.50.

Dallwitz, M. J. (1980) ‘A general system for coding taxonomic descriptions’, *Taxon*, 29(1), pp. 41–46. doi: 10.2307/1219595.

Dallwitz, M. J., Paine, T. a and Zurcher, E. J. (2000) ‘IntKey’. Available at: http://www.delta-intkey.com/.

Dallwitz, M. J., Paine, T. a and Zurcher, E. J. (2013) ‘Principles of Interactive Keys’, *Computer-based Species identification*, (January), pp. 1–20. Available at: http://delta-intkey.com/www/interactivekeys.pdf.

Damery, J. and Kopsell, D. (2009) ‘The Vegetative Key to the British Flora’, *HortScience*, 44(7), p. 2065.

Danova, T. (2015) *How Often Should I Update My App? Businesses Should Consider The Benefits Of Frequent App Updates*, *Business Insider UK*. Available at: http://uk.businessinsider.com/app-update-strategy-and-statistics-2015-1?r=US&IR=T (Accessed: 20 October 2017).

Data Protection Act 2018 (2018) *Data Protection Act 2018*. Available at:

http://www.legislation.gov.uk/ukpga/2018/12/pdfs/ukpga_20180012_en.pdf (Accessed: 5 December 2018).

Dauer, S. (2018) *Pros and cons of the trendy 18:9 screen format*, *AndroidPIT*. Available at: https://www.androidpit.com/18-9-screen-advantages-disadvantages (Accessed: 14 December 2018).

Davis, A. and Bersoff, E. (1988) 'A strategy for comparing alternative software development life cycle models', *Software Engineering,* 14(October), pp. 1453–1461. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6190.

Dell Inc (2018) *PowerEdge T630 Tower Server*. Available at: http://www.dell.com/en-uk/work/shop/povw/poweredge-t630 (Accessed: 22 January 2018).

Dellinger-Johnston, R. A. (2015) *A New Method for Creating a Visual Plant Identification Key.* The University of North Carolina at Greensboro.

Deng, A., Lu, J. and Chen, S. (2016) 'Continuous Monitoring of A/B Tests without Pain: Optional Stopping in Bayesian Testing', *CEUR Workshop Proceedings*, 1828, pp. 53–59. doi: 10.1145/1235.

Dennis, A., Wixom, B. H. and Tegarden, D. (2015) *Systems Analysis & Design An Object-Oriented Approach with UML*. New Jersey: Wiley. Available at: http://store.visible.com/Wiley.aspx.

DIGITAL (2016) *Mellennials are top smartphone users*, *Nielsen*. Available at: http://www.nielsen.com/us/en/insights/news/2016/millennials-are-top-smartphone-users.html (Accessed: 6 December 2017).

Dines, T. D. (2002) 'A new hybrid horsetail, *Equisetum arvense* x *E. telmateia* (*E. x robertsii*) in Britain', *Watsonia*, 24(July 2000), pp. 145–157.

Discovery Green Lab (2019) *Forest Tree Identification*, *Google Play Store*. Google Play Store. Available at: https://play.google.com/store/apps/details?id=com.kesiflerdunyasi.foresttreeidentification (Accessed: 24 June 2019).

Divers Dias (2019) 'plant id'. Google Play Store. Available at: https://play.google.com/store/apps/details?id=com.palnt.app.dias (Accessed: 12 June 2019).

Dolan, R. J. and Matthews, J. M. (1993) 'Maximizing the utility of customer product testing - beta-test design and management', *Journal of Product Innovation Management*, 10(4), pp. 318–330.

Doris (2017) *A List of User Experience Goals That UX Designers Should Set*, *Mockplus*. Available at: https://www.mockplus.com/blog/post/user-experience-goals (Accessed: 18 November 2018).

DPriver (2017) *A list of SQL best practices*, *DPriver Blog*. Available at: http://www.dpriver.com/blog/2011/09/27/a-list-of-sql-best-practices/ (Accessed: 29 November 2017).

Dr M goes wild (2013) *Those elusive yellow composites!*, *Dr M goes wild*. Available at: http://drmgoeswild.com/those-elusive-yellow-composites/ (Accessed: 20 December 2018).

Drinkwater, R. E. (2009) 'Insights into the development of online plant identification keys based on literature review: an exemplar electronic key to Australian Drosera', *Bioscience Horizons*, 2(1), pp. 90–96. doi: 10.1093/biohorizons/hzp007.

Druce, G. C. (1897) *The Flora of Berkshire*. Oxford: Clarendon Press Oxford.

Dubey, A. (2015) *SQLite Locking and Transaction Handling in Android*, *To The New Blog*. Available at: http://www.tothenew.com/blog/sqlite-locking-and-transaction-handling-in-android/ (Accessed: 27 November 2017).

Edwards, M. and Morse, D. R. (1995) 'The potential for computer-aided identification in biodiversity research', *Trends in Ecology and Evolution*. doi: 10.1016/S0169-5347(00)89026-6.

Emilio, J. and Gayo, L. (2009) *Web 2.0*. Edited by M. D. Lytras, E. Damiani, and P. Ordóñez de Pablos. Boston, MA: Springer US. doi: 10.1007/978-0-387-85895-1.

Emsley, H. H. (1953) *Visual Optics: Volume 1 Optics of Vision*. 5th edn. London: Butterworth.

Encyclopedia Britannica (2017) *SQL*, *Encyclopedia Britannica*. Available at: https://www.britannica.com/technology/SQL (Accessed: 15 August 2017).

Epson (2017) *Epson Expression 1640xl*. Available at: https://www.epson.eu/products/scanners/consumer-scanners/epson-expression-1640xl (Accessed: 2 June 2017).

Evgeniy (2017) *How Often Should You Update Your App*, *GBKSoft*. Available at: http://gbksoft.com/blog/how-often-should-you-update-your-app/ (Accessed: 20 October 2017).

Farnsworth, E. J., Chu, M., Kress, W. J., Neill, A. K., Best, J. H., Pickering, J., Stevenson, R. D., Courtney, G. W., VanDyk, J. K. and Ellison, A. M. (2013) 'Next-Generation Field Guides', *BioScience*, 63(11), pp. 891–899. doi: 10.1525/bio.2013.63.11.8.

Feinstein, J. (2017) *Squeezing Performance from SQLite: Insertions*, *Medium*. Available at: https://medium.com/@JasonWyatt/squeezing-performance-from-sqlite-insertions-971aff98eef2 (Accessed: 3 November 2017).

Firebase (2017) *Firebase*, *Firebase*. Available at: https://firebase.google.com/ (Accessed: 11 August 2017).

Firebase (2018a) *Create Firebase Remote Config Experiments with A/B Testing*. Available at: https://firebase.google.com/docs/ab-testing/abtest-config (Accessed: 9 May 2018).

Firebase (2018b) *Firebase Test Lab*, *Firebase*. Available at: https://firebase.google.com/docs/test-lab/ (Accessed: 9 May 2018).

Flores, J. (2017) *An Opinion On: The Maddening Micro-Transactions Conundrum*, *The Indie Toaster*. Available at: https://www.indietoaster.com/en/video-games/understanding-the-maddening-microtransaction/ (Accessed: 31 October 2018).

Fluxà, J.-M. (2009) *Using your own SQLite database in Android applications*, *Reign Design*. Available at: http://blog.reigndesign.com/blog/using-your-own-sqlite-database-in-android-applications/ (Accessed: 11 February 2016).

Fox, N. S., Brennan, J. S. and Chasen, S. T. (2008) 'Clinical estimation of fetal weight and the Hawthorne effect', *European Journal of Obstetrics Gynecology and Reproductive Biology*, 141(2), pp. 111–114. doi: 10.1016/j.ejogrb.2008.07.023.

Freedman, A. (2018) *Definition: SaaS*, *Computer Desktop Encyclopedia*. Available at: https://www.computerlanguage.com/results.php?definition=software+as+a+service (Accessed: 5 December 2018).

Frodin, D. G. (2001) *Guide to Standard Floras of the World*. 2nd edn. Cambridge University

Press.

Frost, J. (2016) *How to Compare Regression Slopes*, *The Minitab blog*. Available at: http://blog.minitab.com/blog/adventures-in-statistics-2/how-to-compare-regression-lines-between-different-models (Accessed: 1 November 2017).

Geekbench Browser (2018) *LG Nexus 5 Benchmarks*. Available at: https://browser.geekbench.com/android_devices/120 (Accessed: 28 August 2018).

GenMyModel (2019) *GenMyModel*. Available at: https://www.genmymodel.com/ (Accessed: 11 July 2019).

Genty, N. (2017) *How Often Should You Update Your App?*, *Iconic Solutions*. Available at: https://iconic-solutions.com/how-often-should-you-update-your-mobile-app/ (Accessed: 20 October 2017).

Google (2015) *Speed Is Key: Optimize Your Mobile Experience*. Available at: https://www.thinkwithgoogle.com/marketing-resources/experience-design/speed-is-key-optimize-your-mobile-experience/ (Accessed: 28 November 2017).

Google (2017a) *Google Play Store apps*, *Google Play Store*. Available at: https://play.google.com/store/apps?hl=en (Accessed: 20 November 2017).

Google (2017b) *Material Design*. Available at: https://material.io/ (Accessed: 8 November 2017).

Google (2018a) *Ever wondered how much AdMob Revenue you can earn?*, *AdMob by Google*. Available at: https://www.google.com/admob/resources/how-much-revenue-can-you-earn-from-admob.html (Accessed: 9 November 2018).

Google (2018b) *Google Lens*. Available at: https://lens.google.com/ (Accessed: 7 December 2018).

Google (2018c) *Google Mobile Ads SDK for Android*. Available at: https://developers.google.com/admob/android/quick-start (Accessed: 19 April 2018).

Google (2018d) *Google Play*. Available at: https://play.google.com/store?hl=en (Accessed: 15 February 2018).

Google (2018e) *Google Play Console Developer Agreement*. Available at: https://play.google.com/apps/publish/signup/ (Accessed: 15 December 2018).

Google (2019a) *android*, *Android*. Available at: https://www.android.com/intl/en_uk/ (Accessed: 17 July 2019).

Google (2019b) *Google Java Style Guide*, *Google*. Available at: https://google.github.io/styleguide/javaguide.html#s7.1-javadoc-formatting (Accessed: 19 May 2019).

Google Inc. (2012) 'The New Multi-Screen World: Understanding Cross-Platform Consumer Behavior', *Google Think Insights*, (August). Available at: http://www.thinkwithgoogle.com/research-studies/the-new-multi-screen-world-study.html.

Google and JetBrains (2018) 'Android Studio'. Google. Available at: https://developer.android.com/studio/releases/index.html#3-2-0 (Accessed: 26 September 2018).

Google LLC (2018a) *Google Drive*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.google.android.apps.docs (Accessed: 4

November 2018).

Google LLC (2018b) *Maps - Navigate & Explore*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.google.android.apps.maps&hl=en_GB (Accessed: 14 December 2018).

Google LLC (2018c) *Measure - Quick Everyday Measurements*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.google.tango.measure&hl=en_GB&rdid=com.google.tango.measure (Accessed: 8 December 2018).

Googlephotos (2018) *Rolling out today, Android users can try Google Lens to do things like create a contact from a business card or get more info about a famous landmark. To start, make sure you have the latest version of the Google Photos app for Android: http://goo.gl/8ZXW*. Available at: https://twitter.com/googlephotos/status/970788927765278720 (Accessed: 7 December 2018).

Gov.UK (2017a) *Estimate your business rates*. Available at: https://www.gov.uk/calculate-your-business-rates (Accessed: 23 October 2017).

Gov.UK (2017b) *National Insurance rates and categories*. Available at: https://www.gov.uk/national-insurance-rates-letters (Accessed: 27 October 2017).

Gov.UK (2018) *Data protection*. Available at: https://www.gov.uk/data-protection (Accessed: 5 December 2018).

Govaerts, R. (2001) 'How Many Species of Seed Plants Are There?', *Taxon*, 50(4), p. 1085. doi: 10.2307/1224723.

Gove, J. (2016) *Principles of Mobile App Design: Engage Users and Drive Conversions*, *think with Google*. Available at: https://www.thinkwithgoogle.com/marketing-resources/experience-design/principles-of-mobile-app-design-introduction/ (Accessed: 13 November 2018).

Gruber, T. R. (1995) 'Towards principles for the design of ontologies used for knowledge sharing', *International Journal of Human-Computer Studies*, 43(5–6), pp. 907–928. doi: doi.org/10.1006/ijhc.1995.1081.

GSMArena (2015a) *LG Nexus 5*. Available at: http://www.gsmarena.com/lg_nexus_5-5705.php (Accessed: 12 October 2015).

GSMArena (2015b) *Qualcomm announces Snapdragon 820 with Kryo CPU*. Available at: https://www.gsmarena.com/qualcomm_announces_snapdragon_820_with_kryo_cpu-news-11386.php (Accessed: 1 August 2018).

GSMArena (2016) *LG G Pad 8.3*. Available at: http://www.gsmarena.com/lg_g_pad_8_3-5673.php (Accessed: 9 March 2016).

GSMArena (2017) *Samsung Galaxy S8*. Available at: http://www.gsmarena.com/samsung_galaxy_s8-8161.php (Accessed: 8 June 2017).

GSMArena (2018a) *LG V40 ThinQ*. Available at: https://www.gsmarena.com/lg_v40_thinq-9300.php (Accessed: 14 December 2018).

GSMArena (2018b) *OnePlus 3T*. Available at: https://www.gsmarena.com/oneplus_3t-8416.php (Accessed: 22 February 2018).

GSMArena (2018c) *Samsung Galaxy S9+*. Available at: https://www.gsmarena.com/samsung_galaxy_s9+-8967.php (Accessed: 14 December 2018).

Guerin, B. (1986) 'Mere Presence Effects in Humans: A Review', *Journal of Experimental Social*

Psychology, 22, pp. 38–77.

Haines, R. (2013) ‘Wildflower Identification’. Google Play Store. Available at: https://play.google.com/store/apps/details?id=com.ourbenefactors.wildflowerIdentification (Accessed: 7 June 2015).

Hall, A. V. (1970) ‘A computer-based system for forming identification keys’, *Taxon*, 19(1), pp. 12–18. Available at: http://www.jstor.org/discover/10.2307/1217908?uid=3738032&uid=2129&uid=2134&uid=372420607&uid=1250888&uid=2&uid=70&uid=3&uid=5910784&uid=67&uid=19230&sid=21102295579177.

Hall, N. (1954) ‘Punched card systems and their application to forestry in Australia’, *Australian Forestry*, 18(2), pp. 141–152.

Harder, D. W. (2011) *4.5 Perfect Binary Trees*. University of Waterloo. Available at: https://ece.uwaterloo.ca/~cmoreno/ece250/4.05.PerfectBinaryTrees.pdf (Accessed: 13 December 2018).

Hawthorne, W. D., Cable, S. and Marshall, C. A. M. (2014) ‘Empirical trials of plant field guides’, *Conservation Biology*, 28(3), pp. 654–662. doi: 10.1111/cobi.12232.

Herbrich, M. and Schmidt, D. (2017) *Test Samsung Galaxy S8 Smartphone*, *Notebook Check*. Available at: https://www.notebookcheck.com/Test-Samsung-Galaxy-S8-Smartphone.209284.0.html (Accessed: 2 August 2018).

Hern, A. (2018) *The rise of Patreon - the website that makes Jordan Peterson $80k a month*, *The Guardian*. Available at: https://www.theguardian.com/technology/2018/may/14/patreon-rise-jordan-peterson-online-membership (Accessed: 29 October 2018).

High Country Apps LLC (2015) ‘Washington Wildflowers Intro’, *Google Play Store*. Google Play Store. Available at: https://play.google.com/store/apps/details?id=com.emountainworks.android.washingtonfieldguidedemo&hl=en (Accessed: 14 June 2016).

Hijazi, H., Khdour, T. and Alarabeyyat, A. (2012) ‘A Review of Risk Management in Different Software Development Methodologies’, *International Journal of Computer Applications*, 45(7), pp. 975–8887.

Hipp, D. R., Kennedy, D. and Mistachkin, J. (2018) ‘SQLite’. SQlite. Available at: https://sqlite.org/ (Accessed: 12 July 2019).

Hopkins, S. (1991) *A key to the woodlice of Britian and Ireland*. Shrewsbury: Field Studies Council.

Horne, L. K. (2012) ‘Apps: A practical approach to trade and co-financed book apps’, *Publishing Research Quarterly*, 28(1), pp. 17–22. doi: 10.1007/s12109-012-9257-4.

Hsiao, K. L. and Chen, C. C. (2016) ‘What drives in-app purchase intention for mobile games? An examination of perceived values and loyalty’, *Electronic Commerce Research and Applications*. Elsevier B.V., 16, pp. 18–29. doi: 10.1016/j.elerap.2016.01.001.

Human failure types (2016). Health and Safety Excecutive. Available at: http://www.hse.gov.uk/humanfactors/topics/types.pdf (Accessed: 17 July 2019).

idtools.org (2018) *Identifying Commonly Cultivated Palms*. Available at: http://idtools.org/id/palms/palmid/about.php (Accessed: 4 December 2018).

Ingram, S. (2016) *The Thumb Zone: Designing For Mobile Users*, *Smashing Magazine*. Available at: https://www.smashingmagazine.com/2016/09/the-thumb-zone-designing-for-mobile-users/ (Accessed: 10 December 2018).

Inksacpe (2019) 'Inkscape'. Available at: https://inkscape.org/release/inkscape-0.92.4/ (Accessed: 12 July 2019).

Intel (2018a) *Intel® Core^{TM} i3-8100 Processor*, *Intel ARK*. Available at: https://ark.intel.com/products/126688/Intel-Core-i3-8100-Processor-6M-Cache-3_60-GHz (Accessed: 12 February 2018).

Intel (2018b) *Intel® Core^{TM} i5-8400 Processor*, *Intel ARK*. Available at: https://ark.intel.com/products/126687/Intel-Core-i5-8400-Processor-9M-Cache-up-to-4_00-GHz (Accessed: 12 February 2018).

Intel (2018c) *Intel® Core^{TM} i7-6700K Processor*, *Intel ARK*. Available at: https://ark.intel.com/products/88195/Intel-Core-i7-6700K-Processor-8M-Cache-up-to-4_20-GHz (Accessed: 1 August 2018).

Intel (2018d) *Intel® Core^{TM} i7-7700K Processor*, *Intel ARK*. Available at: https://ark.intel.com/products/97129/Intel-Core-i7-7700K-Processor-8M-Cache-up-to-4_50-GHz (Accessed: 1 August 2018).

Intel (2018e) *Intel® Core^{TM} i7-8700 Processor*, *Intel ARK*. Available at: https://ark.intel.com/products/126686/Intel-Core-i7-8700-Processor-12M-Cache-up-to-4_60-GHz (Accessed: 12 February 2018).

Intel (2018f) *Intel® Core^{TM} i7-8700K Processor*, *Intel ARK*. Available at: https://ark.intel.com/products/126684/Intel-Core-i7-8700K-Processor-12M-Cache-up-to-4_70-GHz (Accessed: 1 August 2018).

Jansen, M. (2018) *Xiaomi is preparing to set records with 48-megapixel phone camera*, *Digital Trends*. Available at: https://www.digitaltrends.com/mobile/xiaomi-48-megapixel-phone-camera/ (Accessed: 19 December 2018).

JetBrains (2018) 'IntelliJIDEA'. JetBrains. Available at: https://www.jetbrains.com/idea/ (Accessed: 4 September 2018).

Kauppinen, R. (2003) 'Testing framework-based software product lines', *Master's thesis, University of Helsinki Department of …*. Available at: http://www.cs.helsinki.fi/group/rita/papers/pltesting.pdf.

Kearl, M. (2016) *30 Essential Stats on In-App Purchases and Monetization*, *braze*. Available at: https://www.braze.com/blog/in-app-purchase-stats/ (Accessed: 31 October 2018).

Kickstarter (2018) *Kickstarter*, *Kickstarter*. Available at: https://www.kickstarter.com/ (Accessed: 9 November 2018).

King (2018) *Candy Crush Saga*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.king.candycrushsaga (Accessed: 4 November 2018).

Kirchoff, B. K., Leggett, R., Her, V., Moua, C., Morrison, J. and Poole, C. (2011) 'Principles of visual key construction-with a visual identification key to the Fagaceae of the southeastern United States', *AoB PLANTS*, 11(1). doi: 10.1093/aobpla/plr005.

Kirk, A. and Scott, P. (2018) *Britain's highest paying degrees, according to graduate salary*, *The Telegraph*. Available at: https://www.telegraph.co.uk/education/0/uks-highest-paying-degrees-according-graduate-salary/ (Accessed: 15 December 2018).

Klug, B. (2013) *The HTC One Review*, *AnandTech*. Available at: https://www.anandtech.com/show/6747/htc-one-review/5 (Accessed: 14 December 2018).

Koch, R. (2017) *SQL Database Performance Tuning for Developers*, *toptal*. Available at: https://www.toptal.com/sql/sql-database-tuning-for-developers (Accessed: 3 November 2017).

Kohavi, R., Longbotham, R., Sommerfield, D. and Henne, R. M. (2009) 'Controlled experiments on the web: Survey and practical guide', *Data Mining and Knowledge Discovery*, 18(1), pp. 140–181. doi: 10.1007/s10618-008-0114-1.

Kohavi, R. and Longbotham, R. (2015) 'Online Controlled Experiments', in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '15*. New York, New York, USA: ACM Press, pp. 1–1. doi: 10.1145/2783258.2785464.

Köhntopp, K. (2016) *Which is faster: SELECT * or SELECT by column names in MySQL?*, *Quora*. Available at: https://www.quora.com/Which-is-faster-SELECT-*-or-SELECT-by-column-names-in-MySQL (Accessed: 29 November 2017).

Kompier, M. A. J. (2006) 'The "Hawthorne effect" is a myth, but what keeps the story going?', *Scandinavian Journal of Work, Environment and Health*, 32(5), pp. 402–412. doi: 10.5271/sjweh.1036.

Krauthammer, M., Rzhetsky, A., Morozov, P. and Friedman, C. (2000) 'Using BLAST, A DNA and Protein Sequence Comparison Tool, for Finding Gene and Protein Names in Journal Articles', *Proceedings of the AMIA Symposium*, 25(1 7), p. 1052.

Kumar, G. and Bhatia, P. K. (2014) 'Comparative analysis of software engineering models from traditional to modern methodologies', *International Conference on Advanced Computing and Communication Technologies, ACCT*. IEEE, pp. 189–196. doi: 10.1109/ACCT.2014.73.

Kumar, V. (2014) *Making 'Freemium' Work*, *Harvard Business Review*. Available at: https://hbr.org/2014/05/making-freemium-work# (Accessed: 12 July 2019).

Kurtzleben, D. (2012) *How Your Textbook Dollars Are Divvied Up*, *U.S. News*. Available at: https://www.usnews.com/news/articles/2012/08/28/how-your-textbook-dollars-are-divvied-up (Accessed: 15 December 2018).

De La Vega, R., Roset, R., Castarlenas, E., Sánchez-Rodríguez, E., Solé, E. and Miró, J. (2014) 'Development and testing of painometer: A smartphone app to assess pain intensity', *Journal of Pain*, 15(10), pp. 1001–1007. doi: 10.1016/j.jpain.2014.04.009.

Lambrecht, A., Goldfarb, A., Bonatti, A., Ghose, A., Goldstein, D. G., Lewis, R., Rao, A., Sahni, N. and Yao, S. (2014) 'How do firms make money selling digital goods online?', *Marketing Letters*, 25(3), pp. 331–341. doi: 10.1007/s11002-014-9310-5.

Lawrence, A. and Hawthorne, W. (2006) *Plant Identification Creating User Friendly Field Guides for Biodiversity Management*. London: Earthscan.

Leafsnap (2011) *Leafsnap About*. Available at: http://leafsnap.com/about/ (Accessed: 22 October 2018).

Learn and Grow Inc (2018) 'Plant Identification'. Google Play Store. Available at: https://play.google.com/store/apps/details?id=com.andromo.dev601172.app695529 (Accessed: 30 May 2019).

Leau, Y. B., Loo, W. K., Tham, W. Y. and Tan, S. F. (2012) ‘Software Development Life Cycle AGILE vs Traditional Approaches’, *International Conference on Information and Network Technology (ICINT 2012)*, 37, pp. 162–167. Available at: https://pdfs.semanticscholar.org/69b1/9ddc8a578f4c63d1dfe15252a465ee12fe5d.pdf.

Lexico (2019) *lexicon*, *Lexico*. Available at: https://www.lexico.com/en/definition/lexicon (Accessed: 7 July 2019).

Licskai, C. J., Sands, T. W. and Ferrone, M. (2013) ‘Development and pilot testing of a mobile health solution for asthma self-management: Asthma action plan smartphone application pilot study’, *Canadian Respiratory Journal*, 20(4), pp. 301–306. doi: 10.1155/2013/906710.

Liddy, E. D. (2001) ‘Natural Language Processing.’, in *Encyclopedia of Library and Information Science*. 2nd edn. New York: Marcel Decker.

Lind, J. (1753) *A treatise of the scurvy. In three parts. Containing an inquiry into the nature, causes and cure, of that disease. Together with a critical and chronological view of what has been published on the subject.* Edinburgh: Sands, Murray, and Cochran.

Linnaeus, C. (1753) *Species Plantarum*. Stockholm: Laurentius Salvius.

Lowrie, A. (1987) *Carnivorous Plants of Australia Vol. 1*. Nedlands, Western Australia: University of Australia Press.

Lowrie, A. (1989) *Carnivorous Plants of Australia Vol. 2*. Nedlands, Western Australia: University of Australia Press.

Lowrie, A. (1998) *Carnivorous Plants of Australia Vol. 3*. Nedlands, Western Australia: University of Australia Press.

LucidCentral (2015) ‘Lucid’. LucidCentral. Available at: http://www.lucidcentral.com/ (Accessed: 12 July 2019).

Lucidchart (2019) *UML Class Diagram Tutorial*, *Lucidchart*. Available at: https://www.lucidchart.com/pages/uml-class-diagram (Accessed: 27 May 2019).

LucidMobile (2015a) ‘Lucid Mobile Android Apps’. Google Play Store. Available at: https://play.google.com/store/apps/developer?id=LucidMobile&hl=en (Accessed: 7 June 2015).

LucidMobile (2015b) ‘Weeds of South East QLD’. Google Play Store. Available at: https://play.google.com/store/apps/details?id=com.lucidcentral.mobile.sew_full&hl=en (Accessed: 7 June 2015).

LucidMobile (2018a) ‘Aquarium and Pond Plant ID (Unreleased)’. Google Play Store. Available at: https://play.google.com/store/apps/details?id=com.lucidcentral.mobile.appw&hl=en (Accessed: 1 July 2019).

LucidMobile (2018b) *Palm ID Key*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.lucidcentral.mobile.palm_id&hl=en (Accessed: 15 November 2018).

Lydon, S. J., Wood, M. M., Huxley, R. and Sutton, D. (2003) ‘Data patterns in multiple botanical descriptions: Implications for automatic processing of legacy data’, *Systematics and Biodiversity*, 1(2), pp. 151–157. doi: 10.1017/S1477200003001129.

Lyon, J. (2018) *Google’s Next Generation Music Recognition*, *Google AI Blog*. Available at: https://ai.googleblog.com/2018/09/googles-next-generation-music.html (Accessed: 14

December 2018).

Makaques (2017) *MAKAQueS, UK/Irish flora key*. Available at: https://play.google.com/store/apps/details?id=com.makaques.makaques&hl=en_GB (Accessed: 13 October 2017).

Masiero, M. (2015) *Test Samsung Galaxy S6 Smartphone*, *Notebook Check*. Available at: https://www.notebookcheck.com/Test-Samsung-Galaxy-S6-Smartphone.139934.0.html (Accessed: 8 February 2018).

McCarney, R., Warner, J., Iliffe, S., Van Haselen, R., Griffin, M. and Fisher, P. (2007) 'The Hawthorne Effect: A randomised, controlled trial', *BMC Medical Research Methodology*, 7, pp. 1–8. doi: 10.1186/1471-2288-7-30.

McDonald, J. H. (2014) *Homoscedasticity and heteroscedasticity*, *Handbook of Biological Statistics*. Available at: http://www.biostathandbook.com/homoscedasticity.html (Accessed: 24 November 2017).

McGee Wood, M., Lydon, S. J., Tablan, V., Maynard, D. and Cunningham, H. (2004) 'Using parallel texts to improve recall in botany', (June), p. 237. doi: 10.1075/cilt.260.26mcg.

McGregor, C. (2015) *In-App Purchases of the Top Grossing apps*, *Medium*. Available at: https://growthbug.com/in-app-purchases-of-the-top-grossing-apps-db091584b69 (Accessed: 31 October 2018).

McRae, H. (2018) *Companies have been selling our data in exchange for 'free' products and services for a long time – Facebook's not so different*, *Independent*. Available at: https://www.independent.co.uk/voices/facebook-data-scandal-free-products-sheryl-sandberg-a8294006.html (Accessed: 30 October 2018).

Microsoft (2009) 'Windows 7'. Microsoft. Available at: https://www.microsoft.com/en-gb/software-download/windows7 (Accessed: 17 July 2019).

Microsoft (2015) *Windows 10*. Microsoft. Available at: https://www.microsoft.com/en-gb/windows/features (Accessed: 17 July 2019).

Microsoft (2016) 'Excel 2016'. Microsoft. Available at: https://products.office.com/en-us/excel (Accessed: 17 July 2019).

Microsoft (2017) *Capitalization Styles*, *Microsoft Developer Network*. Available at: https://msdn.microsoft.com/en-us/library/x2dbyw72(v=vs.71).aspx (Accessed: 23 August 2017).

Microsoft (2018a) *Code Page 1252 Windows Latin 1 (ANSI)*. Available at: https://msdn.microsoft.com/en-us/library/cc195054.aspx (Accessed: 13 February 2018).

Microsoft (2018b) *Get the most from Office with Office 365*. Available at: https://products.office.com/en-GB/compare-all-microsoft-office-products?tab=1 (Accessed: 28 October 2018).

Microsoft (2018c) *The Code-Page Model*. Available at: https://msdn.microsoft.com/en-us/library/cc194787.aspx (Accessed: 13 February 2018).

Microsoft Corporation (2018a) *Microsoft Excel*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.microsoft.office.excel (Accessed: 29 October 2018).

Microsoft Corporation (2018b) *Microsoft Office Lens - PDF Scanner*, *Google Play Store*.

Available at: https://play.google.com/store/apps/details?id=com.microsoft.office.officelens (Accessed: 29 October 2018).

Microsoft Corporation (2018c) *Microsoft OneNote*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.microsoft.office.onenote (Accessed: 29 October 2018).

Microsoft Corporation (2018d) *Microsoft Outlook*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.microsoft.office.outlook (Accessed: 29 October 2018).

Microsoft Corporation (2018e) *Microsoft PowerPoint*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.microsoft.office.powerpoint (Accessed: 29 October 2018).

Microsoft Corporation (2018f) *Microsoft Word*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.microsoft.office.word (Accessed: 29 October 2018).

Miikkulainen, R., Iscoe, N., Shagrin, A., Cordell, R., Nazari, S., Schoolland, C., Brundage, M., Epstein, J., Dean, R. and Lamba, G. (2017) 'Conversion Rate Optimization through Evolutionary Computation', pp. 1–7. doi: 10.1145/3071178.3071312.

Milward, J., Deluca, P., Drummond, C., Watson, R., Dunne, J. and Kimergård, A. (2017) 'Usability Testing of the BRANCH Smartphone App Designed to Reduce Harmful Drinking in Young Adults', *JMIR mHealth and uHealth*, 5(8), p. e109. doi: 10.2196/mhealth.7836.

Moon Active (2018) *Coin Master*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.moonactive.coinmaster (Accessed: 4 November 2018).

Morris, A. (2018) *6 basic SDLC methodologies: Which one is best?*, *Robert Half*. Available at: https://www.roberthalf.com.au/blog/employers/6-basic-sdlc-methodologies-which-one-best (Accessed: 15 April 2019).

Morrison, D. (2011) 'Why is taxonomy still presented to the world as books?', *Australasian Systematic Botany Society Newsletter*, 149(December), pp. 17–22.

Moser, M. (2015) *Test Sony Xperia Z5 Smartphone*, *Notebook Check*. Available at: https://www.notebookcheck.com/Test-Sony-Xperia-Z5-Smartphone.153759.0.html (Accessed: 8 February 2018).

Da Mota Silveira Neto, P. A., Carmo MacHado, I. Do, McGregor, J. D., De Almeida, E. S. and De Lemos Meira, S. R. (2011) 'A systematic mapping study of software product lines testing', *Information and Software Technology*. Elsevier B.V., 53(5), pp. 407–423. doi: 10.1016/j.infsof.2010.12.003.

Mrva-Montoya, A. (2015) 'Beyond the Monograph: Publishing Research for Multimedia and Multiplatform Delivery', *Journal of Scholarly Publishing*, 46(4), pp. 321–342. doi: 10.3138/jsp.46.4.02.

Munassar, N. M. A. and Govardhan, A. (2010) 'A Comparison Between Five Models Of Software Engineering', *International Journal of Computer Science*, 7(5), pp. 94–101. doi: 10.1.1.403.3201.

Munroe, R. (2011) *Standards*, *XKCD*. Available at: https://xkcd.com/927/ (Accessed: 18 December 2018).

Murray, A. P. (2018) *Razer Phone 2 review: Still the best gaming phone*, *PCWorld*. Available at:

https://www.pcworld.com/article/3314717/android/razer-phone-2-review-gaming-phone.html (Accessed: 14 December 2018).

National Audubon Society (2001) *National Audubon Society Field Guide to North American Wildflowers - E*. New York: Penguin Random House.

Natoli, J. (2014) *5 Crucial Principles for Great Mobile Design*, *Give Good*. Available at: https://www.givegoodux.com/5-crucial-principles-great-mobile-design/ (Accessed: 16 November 2018).

Nelson, R. (2018) *U.S. iPhone Users Spent An Average of $58 on Apps in 2017, 23% More Than the Year Before*, *SensorTower*. Available at: https://sensortower.com/blog/revenue-per-iphone-2017 (Accessed: 31 October 2018).

NFU (2018) *NFU About Us*. Available at: https://www.nfuonline.com/about-us/ (Accessed: 28 November 2018).

Niantic Inc (2018) *Pokémon GO*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.nianticlabs.pokemongo (Accessed: 4 November 2018).

Nielsen, J. (1994) 'Usability inspection methods', *Conference companion on Human factors in computing systems - CHI '94*, pp. 413–414. doi: 10.1145/259963.260531.

Night Fox Digital, L. (2018) *Idaho Grasses*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.uofidaho.idahograsses&hl=en_GB (Accessed: 13 July 2019).

Nikishaev, A. (2017) *13 Simple Rules for Good Coding (from my 15 years of experience)*, *Hackernoon*. Available at: https://hackernoon.com/few-simple-rules-for-good-coding-my-15-years-experience-96cb29d4acd9 (Accessed: 19 May 2019).

ninja kiwi (2018) *Bloons TD 6*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.microsoft.office.outlook (Accessed: 29 October 2018).

Noorden, R. Van (2013) *Open access: The true cost of science publishing*, *Nature*. Available at: https://www.nature.com/news/open-access-the-true-cost-of-science-publishing-1.12676 (Accessed: 15 December 2018).

O' Sullivan, I., Orbell, S., Rakow, T. and Parker, R. (2004) 'Prospective Research in Health Service Settings: Health Psychology, Science and the "Hawthorne" Effect', *Journal of Health Psychology*, 9(3), pp. 355–359. doi: 10.1177/1359105304042345.

O'Conner, R. (2016) *Will BBC ruin Gardeners' World? Monty set for revamp in bid to shake off Bake Off success*, *Daily Express*. Available at: http://www.express.co.uk/showbiz/tv-radio/705710/bbc-gardeners-world-hoping-share-bake-off-success (Accessed: 30 October 2017).

Ofcom (2015) *The UK is now a smartphone society*. Available at: https://www.ofcom.org.uk/about-ofcom/latest/media/media-releases/2015/cmr-uk-2015 (Accessed: 9 April 2018).

Ofcom (2018) *Adults' media use and attitudes report*. doi: 10.1016/j.csi.2015.08.004.

Oh, N. (2017) *PCI-SIG Finalizes and Releases PCIe 4.0, Version 1 Specification: 2x Bandwidth and More*, *AnandTech*. Available at: https://www.anandtech.com/show/11967/pcisig-finalizes-and-releasees-pcie-40-spec (Accessed: 7 February 2018).

Oh, Y. K. and Min, J. (2015) 'The mediating role of popularity rank on the relationship between advertising and in-app purchase sales in mobile application market', *Journal of Applied Business Research*, 31(4), pp. 1311–1322. doi: 10.19030/jabr.v31i4.9318.

Oracle (2017a) *Class Integer*, *Java documentation*. Available at: https://docs.oracle.com/javase/7/docs/api/java/lang/Integer.html (Accessed: 23 August 2017).

Oracle (2017b) 'Java'. Oracle. Available at: https://www.java.com/en/ (Accessed: 12 July 2019).

Osborne, D. V. (1963) 'Some Aspects of the Theory of Dichotomous Keys', *New Phytologist*, 62(2), pp. 144–160. doi: 10.1111/j.1469-8137.1963.tb06322.x.

Osthoff, A. (2016) *Test Samsung Galaxy S7 Smartphone*, *Notebook Check*. Available at: https://www.notebookcheck.com/Test-Samsung-Galaxy-S7-Smartphone.160891.0.html (Accessed: 8 February 2018).

Oxford English Dictionary (2018) *Parse*, *English Oxford Living Dictionaries*. Available at: https://en.oxforddictionaries.com/definition/parse (Accessed: 26 September 2018).

Ozer, M. (1999) 'A survey of new product evaluation models', *Journal of Product Innovation Management*, 16(1), pp. 77–94. doi: http://dx.doi.org/.

Page, C. N. (1973) 'Two hybrids in Equisetum new to the British flora', *Watsonia*, 9, pp. 229–237. Available at: http://archive.bsbi.org.uk/Wats9p229.pdf.

Page, C. N. (1981) 'A New Name For A Hybrid Horsetail In Scotland', *The Fern Gazette*, 12(3), pp. 178–179.

Page, C. N. (1982) *The Ferns of Britain and Ireland*. 1st edn. Cambridge: Press Syndicate of the University of Cambridge.

Page, C. N. (1997) *The Ferns of Britian and Ireland (2nd edition)*. Cambridge: Cambridge University Press.

Page, C. N. (2018) *The Ferns of Britain and Ireland: Second Edition*, *Amazon UK*. Available at: https://www.amazon.co.uk/Ferns-Britain-Ireland-Second/dp/0521586585/ref=sr_1_1?ie=UTF8&qid=1540565763&sr=8-1&keywords=the+ferns+of+britain+and+ireland (Accessed: 26 October 2018).

Page, C. N., McHaffie, H. and Butler, J. K. (2007) 'A new far northern hybrid horsetail from Scotland: *Equisetum x mchaffieae* CN Page (*Equisetum fluviatile* L. x *E. pratense* Ehrh.)', *Watsonia*, 26(3), pp. 339–346. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.669.8228&rep=rep1&type=pdf%0A http://www.archive.bsbi.org.uk/Wats26p339.pdf.

Parratt, M. (2017) *Matt's conifers*, *FSC Identikit*. Available at: https://conifers.fscbiodiversity.uk/?fbclid=IwAR0M0CSMEttvb_xr9-XR4YgxEg8SbeVnCuQ1_Pw6FPW-3bbBUGoCOGDuFw8 (Accessed: 5 July 2019).

Patreon (2018) *Patreon*. Available at: https://www.patreon.com/ (Accessed: 29 October 2018).

PCI-SIG (2017) 'DEVCON 2017 UPDATE'. PCI-SIG, p. 7. Available at: https://images.anandtech.com/doci/11967/pci-sig_devcon_2017_press_deck-07.png.

Pearce, A. (2017) *Opinion: Publishers Don't Realize Microtransactions Are Hurting Them, Too*, *IGN*. Available at: https://uk.ign.com/articles/2017/10/28/opinion-publishers-dont-realize-microtransactions-are-hurting-them-too (Accessed: 31 October 2018).

Penev, L., Sharkey, M., Erwin, T., van Noort, S., Buffington, M., Seltmann, K., Johnson, N., Taylor, M., Christian Thompson, F. and Dallwitz, M. J. (2009) ‘Data publication and dissemination of interactive keys under the open access model’, *ZooKeys*, 21, pp. 1–17. doi: 10.3897/zookeys.21.274.

Piskorski, J. and Yangarber, R. (2013) ‘Multi-source, Multilingual Information Extraction and Summarization’, pp. 23–50. doi: 10.1007/978-3-642-28569-1.

Pla@ntNet (2017) *Pl@ntNet*. Available at: https://plantnet.org/en/ (Accessed: 21 November 2017).

Plantnet-project.org (2015) *PlantNet Plant Identification*. Google Play Store. Available at: https://play.google.com/store/apps/details?id=org.plantnet (Accessed: 8 October 2015).

Plantnet-project.org (2017) *Pl@nt Net ‘About’*. Available at: https://identify.plantnet-project.org/api/about/credits (Accessed: 7 December 2017).

Playdemic (2018) *Golf Clash*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.playdemic.golf.android (Accessed: 4 November 2018).

Playrix Games (2018a) *Gardenscapes*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.playrix.gardenscapes (Accessed: 4 November 2018).

Playrix Games (2018b) *Homescapes*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.playrix.homescapes (Accessed: 4 November 2018).

Poland, J. and Clement, E. (2009) *The Vegetative Key to the British Flora*. Totton: Botanical Society of the British Isles.

Poland, J. and Clement, E. (2018a) *The Vegetative Key to the British Flora*, *Amazon UK*. Available at: https://www.amazon.co.uk/Vegetative-Key-British-Flora/dp/0956014402/ref=tmm_pap_swatch_0?_encoding=UTF8&qid=1544696969&sr=8-1 (Accessed: 13 December 2018).

Poland, J. and Clement, E. (2018b) *The Vegetative Key to the British Flora*, *Amazon UK*. Available at: https://www.amazon.co.uk/Vegetative-Key-British-Flora/dp/0956014402 (Accessed: 11 April 2018).

Portti, L. and NatureGate (2019) *Identification of plants*, *NatureGate*. Available at: http://kukkakasvit.luontoportti.fi/index.phtml?lang=en (Accessed: 24 June 2019).

Potter, M. C., Wyble, B., Hagmann, C. E. and McCourt, E. S. (2014) ‘Detecting meaning in RSVP at 13 ms per picture’, *Attention, Perception, & Psychophysics*, 76(2), pp. 270–279. doi: 10.3758/s13414-013-0605-z.

Primate Labs Inc (2018a) *Intel Core i7-6700K Benchmarks*, *Geekbench 4 benchmarks*. Available at: https://browser.geekbench.com/processors/1705 (Accessed: 1 August 2018).

Primate Labs Inc (2018b) *Intel Core i7-7700K Benchmarks*, *Geekbench 4 benchmarks*. Available at: https://browser.geekbench.com/processors/1779 (Accessed: 1 August 2018).

Primate Labs Inc (2018c) *Intel Core i7-8700K Benchmarks*, *Geekbench 4 benchmarks*. Available at: https://browser.geekbench.com/processors/2062 (Accessed: 1 August 2018).

Primate Labs Inc (2018d) *Introducing Geekbench 4*. Available at: https://www.geekbench.com/ (Accessed: 25 July 2018).

Programiz (2019) *Java Class and Objects*, *Programiz*. Available at: https://www.programiz.com/java-programming/class-objects (Accessed: 18 May 2019).

Qualcomm (2018a) *MSM8940 Processor*. Available at: https://www.qualcomm.com/products/msm8940-processor (Accessed: 12 February 2018).

Qualcomm (2018b) *SDM630 Processor*. Available at: https://www.qualcomm.com/products/sdm630 (Accessed: 12 February 2018).

Qualcomm (2018c) *Snapdragon 835 Mobile Platform*. Available at: https://www.qualcomm.com/products/snapdragon/processors/835 (Accessed: 12 February 2018).

Randler, C. and Zehender, I. (2006) 'Effectiveness of reptile species identification - A comparison of a dichotomous key with an identification book', *Eurasia Journal of Mathematics, Science and Technology Education*, 2(3), pp. 55–65. doi: 10.12973/ejmste/75464.

Real Jardin Botanico (2015) 'ArbolApp'. Google Play Store. Available at: https://play.google.com/store/apps/details?id=com.mo2o.csic.botanic (Accessed: 12 July 2019).

Roblox Corporation (2018) *ROBLOX*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.roblox.client (Accessed: 4 November 2018).

Roma, P. and Ragaglia, D. (2016) 'Revenue models, in-app purchase, and the app performance: Evidence from Apple's App Store and Google Play', *Electronic Commerce Research and Applications*. Elsevier B.V., 17, pp. 173–190. doi: 10.1016/j.elerap.2016.04.007.

Rose, F. and O'Reilly, C. (2006) *The Wild Flower Key*. London: Warne.

Rose, F. and O'Reilly, C. (2018) *The Wild Flower Key (Revised Edition) - How to identify wild plants, trees and shrubs in Britain and Ireland*, *Amazon UK*. Available at: https://www.amazon.co.uk/Wild-Flower-Key-Revised-identify/dp/0723251754/ref=sr_1_1?ie=UTF8&qid=1544697783&sr=8-1&keywords=francis+rose+wildflower+key (Accessed: 11 April 2018).

Rouse, M. (2005) *class*, *WhatIs.com*. Available at: https://whatis.techtarget.com/definition/class (Accessed: 18 May 2019).

Royal Horticultural Society (2015) *A year with the RHS 2015*. Royal Horticultural Society. Available at: https://www.rhs.org.uk/about-the-rhs/pdfs/about-the-rhs/mission-and-strategy/past-annual-reports/RHS-Annual-Review-2015-2016.pdf (Accessed: 12 July 2019).

Royce, W. W. (1970) 'Managing the development of large software systems', *IEEE Wescon*, pp. 338–382.

Rubin, J. and Chisnell, D. (2008) *Handbook of usability testing [electronic resource] : How to plan, design, and conduct effective tests (2nd ed.), Indianapolis, IN: Wiley Pub.* doi: 10.1007/s13398-014-0173-7.2.

Ruddock, D. (2017) *Qualcomm Snapdragon 835: The first benchmark results - comparisons to Pixel, OnePlus 3T, Galaxy S7, and more*, *Android Police*. Available at: https://www.androidpolice.com/2017/03/22/qualcomm-snapdragon-835-first-benchmark-results-comparisons-pixel-oneplus-3t-galaxy-s7/ (Accessed: 1 August 2018).

Ruparelia, N. B. (2010) 'Software development lifecycle models', *ACM SIGSOFT Software Engineering Notes*, 35(3), p. 8. doi: 10.1145/1764810.1764814.

Sam, B. (2017) *6 Necessary Elements For Designing A Perfect Mobile App User Interface*, *AppSamurai*. Available at: https://appsamurai.com/6-necessary-elements-for-designing-a-perfect-mobile-app-user-interface/ (Accessed: 16 November 2018).

SATA-IO (2018) *SATA Naming Guidelines*. Available at: https://sata-io.org/developers/sata-naming-guidelines (Accessed: 8 February 2018).

Sauro, J. (2017) *Do Observers Affect Usability Test Results?*, *MeasuringU*. Available at: https://measuringu.com/observer-effect/ (Accessed: 9 May 2018).

Schneider, C. A., Rasband, W. S. and Eliceiri, K. W. (2012) 'NIH Image to ImageJ: 25 years of image analysis', *Nature Methods*. Bethesda, Maryland: National Institutes of Health, 9(7), pp. 671–675. doi: 10.1038/nmeth.2089.

Schuster, D. (2013) *Review Google Nexus 5 Smartphone*, *Notebook Check*. Available at: https://www.notebookcheck.net/Review-Google-Nexus-5-Smartphone.105973.0.html (Accessed: 28 August 2018).

Schwabe, I. (2017) *Test Sony Xperia XZ1 Smartphone*, *Notebook Check*. Available at: https://www.notebookcheck.com/Test-Sony-Xperia-XZ1-Smartphone.260254.0.html (Accessed: 8 February 2018).

Sell, P. and Murrell, G. (1997) *Flora of Great Britain and Ireland, Volume 5:* Butomaceae - Orchidaceae. Cambridge: Cambridge University Press.

Sell, P. and Murrell, G. (2006) *Flora of Great Britain and Ireland, Volume 4:* Campanulaceae - Asteraceae. Cambridge: Cambridge University Press.

Sell, P. and Murrell, G. (2009) *Flora of Great Britain and Ireland, Volume 3:* Mimosaceae - Lentibulariaceae. Cambridge: Cambridge University Press.

Sell, P. and Murrell, G. (2014) *Flora of Great Britain and Ireland, Volume 2:* Capparaceae - Rosaceae. Cambridge: Cambridge University Press.

Sell, P. and Murrell, G. (2018) *Flora of Great Britain and Ireland, Volume 1:* Lycopodiaceae - Salicaceae. Cambridge: Cambridge University Press.

Shafranovich, Y. (2005) *RFC 4180 Common Format and MIME Type for Comma-Separated Values (CSV) Files*, *Internet Engineering Task Force*. Available at: https://tools.ietf.org/html/rfc4180 (Accessed: 1 February 2018).

Sharif, B. and Maletic, J. I. (2010) 'An eye tracking study on camelcase and under-score identifier styles', *IEEE International Conference on Program Comprehension*, pp. 196–205. doi: 10.1109/ICPC.2010.41.

Sharma, N. (2016) 'Species Identification in Citizen Science', *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA '16*, pp. 128–133. doi: 10.1145/2851581.2890382.

Shull, G. (1948) 'What is "heterosis"?', *Genetics*, 33(5), p. 439. Available at: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1209417/.

Silva, H., Pinho, R., Lopes, L., Nogueira, A. J. A. and Silveira, P. (2011) 'Illustrated plant identification keys: An interactive tool to learn botany', *Computers and Education*. Elsevier Ltd, 56(4), pp. 969–973. doi: 10.1016/j.compedu.2010.11.011.

Sims, G. (2014) *ARM vs X86 – Key differences explained!*, *Android Authority*. Available at: https://www.androidauthority.com/arm-vs-x86-key-differences-explained-568718/ (Accessed:

12 February 2018).

Siu, D. N. (2018) *Google Measure app is available (but glitchy) on even more phones*, *MashableUK*. Available at: https://mashable.com/article/google-measure-android-update-bad/?europe=true#rr5wQl74wmqO (Accessed: 8 December 2018).

Sleep Sounds Studio (2019) *PlantFinder - Flower & Plant Identification*, *Google Play Store*. Google Play Store. Available at: https://play.google.com/store/apps/details?id=com.plantfinder.identification (Accessed: 25 June 2019).

Smith, R. (2017) *Annual Survey of Hours and Earnings: 2017 provisional and 2016 revised results*, *Office for National Statistics Statistical bulletin*. Available at: https://www.ons.gov.uk/employmentandlabourmarket/peopleinwork/earningsandworkinghours/bulletins/annualsurveyofhoursandearnings/2017provisionaland2016revisedresults#regional-earnings (Accessed: 27 October 2017).

So, Y. (2017) *Designing for Mobile Apps: Overall Principles, Common Patterns, and Interface Guidelines*, *Mediu*. Available at: https://medium.com/blueprint-by-intuit/native-mobile-app-design-overall-principles-and-common-patterns-26edee8ced10 (Accessed: 16 November 2018).

Sold, F. (2013) *Test Sony Xperia Z Smartphone*, *Notebook Check*. Available at: https://www.notebookcheck.com/Test-Sony-Xperia-Z-Smartphone.88500.0.html (Accessed: 8 February 2018).

Spool, J. M. (2009) *The Magic Behind Amazon's 2.7 Billion Dollar Question*, *uie*. Available at: https://articles.uie.com/magicbehindamazon/ (Accessed: 11 April 2018).

SQlite (2019) *What Is SQLite?* Available at: https://www.sqlite.org/index.html (Accessed: 27 May 2019).

SQLite (2018) *SQL As Understood By SQLite*. Available at: https://www.sqlite.org/lang_select.html (Accessed: 23 January 2018).

Sqlitebrowser (2017) 'DB Browser for SQLite'. GitHub. Available at: http://sqlitebrowser.org/.

Stace, C. (1991) *New Flora of the British Isles*. 1st edn. Cambridge: Cambridge University Press.

Stace, C. (1997) *New Flora of the British Isles*. 2nd edn. Cambridge: Cambridge University Press.

Stace, C. (1999) *Field Flora of the British Isles*. Cambridge: Cambridge University Press.

Stace, C. (2010a) 'Keys to Families of Angiosperms', in *New Flora of the British Isles (Third Edition)*. Cambridge: Cambridge University Press, pp. 56–78.

Stace, C. (2010b) 'Multi-access key to spp. of Epilobium', in *New Flora of the British Isles (Third Edition)*. Cambridge: Cambridge University Press, p. 355.

Stace, C. (2010c) *New Flora of the British Isles*. 3rd edn. Cambridge: Cambridge University Press.

Stace, C. (2010d) 'ROSACEAE - Rose family', in *New Flora of the British Isles (Third Edition)*. Cambridge University Press, p. 187.

Stace, C. (2018) *Clive Stace | British Botanist*, *Clive Stace*. Available at: https://www.clivestace.com/ (Accessed: 6 December 2018).

Stackoverflow (2017) *Stack Overflow Developer Survey 2017*. Available at:

https://insights.stackoverflow.com/survey/2017 (Accessed: 27 November 2017).

Stagg, B. C. and Donkin, M. E. (2017) ‘Apps for angiosperms: the usability of mobile computers and printed field guides for UK wild flower and winter tree identification’, *Journal of Biological Education*. Routledge, 51(2), pp. 123–135. doi: 10.1080/00219266.2016.1177572.

statcounter GlobalStats (2019) *Mobile Operating System Market Share Worldwide - June 2019*, *statcounter GlobalStats*. Available at: http://gs.statcounter.com/os-market-share/mobile/worldwide (Accessed: 17 July 2019).

Stein, J., Binion, D. and Acciavatti, R. (2012) *Field Guide to Native Oak Species of Eastern North America*. Morgantown, W. Va: U.S. Forest Service, Forest Health Technology Enterprise Team.

Stinson, J., McGrath, P., Hodnett, E., Feldman, B., Duffy, C., Huber, A., Tucker, L., Hetherington, R., Tse, S., Spiegel, L., Campillo, S., Gill, N. and White, M. (2010) ‘Usability testing of an online self-management program for Adolescents with Juvenile Idiopathic Arthritis’, *Journal of Medical Internet Research*, 12(3). doi: 10.2196/jmir.1349.

Streeter, D., Hart-Davies, C., Hardcastle, A., Cole, F. and Harper, L. (2009) *Collins Flower Guide*. London: Collins Publishing.

Streeter, D., Hart-Davies, C., Hardcastle, A., Cole, F. and Harper, L. (2018a) *Collins Flower Guide (Britain and Ireland)*, *Amazon UK*. Available at: https://www.amazon.co.uk/Collins-Flower-Guide-Britain-Ireland/dp/0007183895/ref=sr_1_1?ie=UTF8&qid=1523445894&sr=8-1&keywords=collins+flower+guide (Accessed: 11 April 2018).

Streeter, D., Hart-Davies, C., Hardcastle, A., Cole, F. and Harper, L. (2018b) *Collins Wild Flower Guide*, *Amazon UK*. HarperCollins Publishers. Available at: https://www.amazon.co.uk/Collins-Flower-Guide-David-Streeter/dp/0008156751/ref=sr_1_5?ie=UTF8&qid=1509009115&sr=8-5&keywords=collins+wild+flower+guide (Accessed: 26 October 2017).

Streeter, D., Hart-Davies, C., Hardcastle, A., Cole, F. and Harper, L. (2018c) *Collins Wild Flower Guide (2nd edition)*, *Amazon UK*. London: HarperCollins Publishers. Available at: https://www.amazon.co.uk/Collins-Flower-Guide-David-Streeter/dp/0008156751/ref=pd_cp_14_1?_encoding=UTF8&psc=1&refRID=W4HRST2KDDVXFKT676CA (Accessed: 26 October 2017).

Stucky, J. M. (1984) ‘Comparison of Two Methods of Identifying Weed Seedlings’, *Weed Science Society of America*, 32(5), pp. 598–602.

Sullivan, S. (2019) *Outstanding Contributors*, *Wildflower Identification Website*. Available at: https://wildflowersearch.org/search?page=Thanks (Accessed: 2 July 2019).

Sullivan, S. K. (2019) *Wildflower Identification Website*. Available at: https://www.wildflowersearch.org/ (Accessed: 13 July 2019).

Supercell (2018) *Clash of Clans*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.supercell.clashofclans (Accessed: 4 November 2018).

Swain, E. H. F. (1926) ‘A Universal Index to Wood’, *Journal of Forestry*, 24(7). doi: https://doi.org/10.1093/jof/24.7.735.

Tardivel, G. M. and Morse, D. R. (1996) ‘The role of the user in computer-based species identification’, in Bridge, P. et al. (eds) *Information Technology, Plant Pathology and Biodiversity*. Oxford: CAB International.

Tarkus, A., Maxl, E. and Kittl, C. (2010) ‘User needs for interactive identification tools to organisms employed in the EU-Project KeyToNature', *Tools for Identifying Biodiversity: Progress and Problems. Proceedings of the International Congress.*, pp. 361–365.

Taylor, A. (1995) 'Extracting knowledge from biological descriptions', *Towards Very Large Knowledge Bases. Knowledge Building and Knowledge Sharing 1995*, pp. 114–120.

TDWG (2018) *Biodiversity Information Standards (TDWG)*. Available at: https://www.tdwg.org/ (Accessed: 18 December 2018).

Technische Universität Ilmenau (2019) 'Flora Incognita - automated plant identification'. Google Play Store. Available at: https://play.google.com/store/apps/details?id=com.floraincognita.app.floraincognita (Accessed: 12 July 2019).

Techquickie (2015) *ARM CPUs as Fast as Possible*, *YouTube*. Available at: https://www.youtube.com/watch?v=X4BxUiqWq8E (Accessed: 12 February 2018).

Teo Siang (2018) *Bad Design vs. Good Design: 5 Examples we can learn from*, *Interaction Design Foundation*. Available at: https://www.interaction-design.org/literature/article/bad-design-vs-good-design-5-examples-we-can-learn-frombad-design-vs-good-design-5-examples-we-can-learn-from-130706 (Accessed: 16 November 2018).

The Botanical Society of Britain and Ireland (2018a) *subscriptions*. Available at: https://bsbi.org/subscriptions (Accessed: 9 November 2018).

The Botanical Society of Britain and Ireland (2018b) *Training*. Available at: https://bsbi.org/training (Accessed: 28 November 2018).

The British Ecological Society (2018) *Research Grants*. Available at: https://www.britishecologicalsociety.org/funding/research-grants/ (Accessed: 28 November 2018).

The Linnean Society of London (2018) *Medals, Awards, Prizes, and Grants*. Available at: https://www.linnean.org/the-society/medals-awards-prizes-grants (Accessed: 28 November 2018).

The National Trust (2016) *National Trust Annual Report 2015/16*. Swindon. Available at: https://nt.global.ssl.fastly.net/documents/annual-report-pdf-version-201516.pdf (Accessed: 12 July 2019).

The Pensions Advisory Service (2018a) *Automatic Enrolment How much do I and my employer have to pay?* Available at: https://www.pensionsadvisoryservice.org.uk/about-pensions/pensions-basics/automatic-enrolment/how-much-do-i-and-my-employer-have-to-pay (Accessed: 21 May 2018).

The Pensions Advisory Service (2018b) *Pensions basics Automatic Enrolment*. Available at: https://www.pensionsadvisoryservice.org.uk/about-pensions/pensions-basics/automatic-enrolment (Accessed: 21 May 2018).

The Star (2012) 'Facebook selling users ' personal data for profit', 4 October, p. 3. Available at: https://search.proquest.com/docview/1086346665/abstract/1BB3E97399CF4CA8PQ/1?accountid=13460 (Accessed: 12 July 2019).

The Sunday Times (2018) *The Sunday Times Rich List 2018: JK Rowling net worth*, *The Sunday Times*. Available at: https://www.thetimes.co.uk/article/sunday-times-rich-list-2018-jk-rowling-net-worth-rhrbq7ctc (Accessed: 14 December 2018).

The Tech Terms Computer Dictionary (2017) *String*, *The Tech Terms Computer Dictionary*.

Available at: https://techterms.com/definition/string (Accessed: 23 August 2017).

Thessen, A. E., Cui, H. and Mozzherin, D. (2012) ‘Applications of natural language processing in biodiversity science’, *Advances in Bioinformatics*, 2012. doi: 10.1155/2012/391574.

Tidwell, J. (2010) *Designing Interfaces: Patterns for Effective Interaction Design (2nd Edition)*. 2nd edn, *O’Reilly*. 2nd edn. Edited by M. Treseler. Sebastopol: O’Reilly.

Tinder (2018) *Tinder®*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.tinder (Accessed: 4 November 2018).

Torvalds, L. (2016) *Geekbench 4*. Available at: https://www.realworldtech.com/forum/?threadid=159853&curpostid=159860 (Accessed: 25 July 2018).

Trackenberg, O. (2019) *iFlora - Flora of Great Britain and Europe*, *Google Play Store*. Google Play Store. Available at: https://play.google.com/store/apps/details?id=de.iflora.europe&hl=en_GB (Accessed: 12 July 2019).

Tutin, T. G., Heywood, V. H., Burges, N. A., Valentine, D. H., Walters, S. M. and Webb, D. A. (1964a) *Flora Europaea Volume 1 Lycopodiaceae to Platanaceae*. Cambridge: Cambridge University Press.

Tutin, T. G., Heywood, V. H., Burges, N. A., Valentine, D. H., Walters, S. M. and Webb, D. A. (1964b) ‘Key to Angiospermae’, in *Flora Europaea Volume 1 Lycopodiaceae to Plantanaceae*. Cambridge University Press, pp. xxvii–xxxii.

Tutin, T. G., Heywood, V. H., Burges, N. A., Moore, D. M., Valentine, D. H., Walters, S. M. and Webb, D. A. (1968) *Flora Europaea Volume 2 Rosaceae to Umbelliferae*. Cambridge: Cambridge University Press.

Tutin, T. G., Heywood, V. H., Burges, N. A., Moore, D. M., Valentine, D. H., Walters, S. M. and Webb, D. A. (1972) *Flora Europaea Volume 3 Diapensiaceae to Myoporaceae*. Cambridge: Cambridge University Press.

Tutin, T. G., Heywood, V. H., Burges, N. A., Moore, D. M., Valentine, D. H., Walters, S. M. and Webb, D. A. (1976) *Flora Europaea Volume 4 Plantaginaceae to Compositae*. Cambridge: Cambridge University Press.

Tutin, T. G., Heywood, V. H., Burges, N. A., Moore, D. M., Valentine, D. H., Walters, S. M. and Webb, D. A. (1980) *Flora Europaea Volume 5 Alismataceae to Orchidaceae*. Cambridge: Cambridge University Press.

Tutin, T. G., Heywood, V. H., Burges, N. A., Valentine, D. H., Walters, S. M., Webb, D. A., Ball, P. W., Chater, A. O., Moore, D. M., Ferguson, I. K., DeFilipps, R. A. and Richardson, I. B. K. (1993) *Flora Europaea 1-5*. Cambridge: Cambridge University Press.

Tutin, T. G., Burges, N. A., Chater, A. O., Edmondson, J. R., Heywood, V. H., Moore, D. M., Valentine, D. H., Walters, S. M. and Webb, D. A. (1993) *Flora Europaea Volume 1 Psilotaceae to Platanaceae (2nd Edition)*. 2nd edn. Cambridge: Cambridge University Press.

Tutin, T. G., Heywood, V. H., Burges, N. A., Valentine, D. H., Walters, S. M. and Webb, D. A. (2018) *Flora Europaea 5 Volume Paperback Set*, *Amazon UK*. Available at: https://www.amazon.co.uk/Flora-Europaea-5-Paperback-Set/dp/0521154065/ref=sr_1_1?ie=UTF8&qid=1541513112&sr=8-1&keywords=flora+europaea (Accessed: 6 November 2018).

Twitter Inc. (2018) *Twitter*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.twitter.android&hl=en_GB (Accessed: 20 December 2018).

U.S. Department of Agriculture (2019) *PLANTS Interactive ID Keys: Introduction*, *USDA Natural Resources Conservation Service*. Available at: https://www.nrcs.usda.gov/wps/portal/nrcs/detailfull/national/entsc/?cid=stelprdb1044954 (Accessed: 24 June 2019).

U.S. Department of Health & Human Services (2018) *User Interface Design Basics*, *usability.gov*. Available at: https://www.usability.gov/what-and-why/user-interface-design.html (Accessed: 16 November 2018).

Ung, G. M. (2015) *Tested: Why the iPad Pro really isn't as fast as a laptop*, *CIO*. Available at: https://www.cio.com/article/3008071/tablets/tested-why-the-ipad-pro-really-isnt-as-fast-as-a-laptop.html?page=3 (Accessed: 25 July 2018).

Virgin media Business (2017) *Voom fibre business broadband*. Available at: https://www.virginmediabusiness.co.uk/connectivity/internet-access/business-broadband/ (Accessed: 27 October 2017).

Virkus, M. (2019) *Software Development Explained With Cars*, *toggl*. Available at: https://toggl.com/developer-methods-infographic/ (Accessed: 15 April 2019).

Visual Pradigm (2019) *Extreme Programming (XP) vs Scrum*, *Visual Paradigm*. Available at: https://www.visual-paradigm.com/scrum/extreme-programming-vs-scrum/ (Accessed: 17 April 2019).

Vollbrecht, L., Rush, M. T. and Cottenie, K. (2013) 'Improving dichotomous keys for undergraduate teaching', *Surg*, 7(1), pp. 23–32. Available at: https://journal.lib.uoguelph.ca/index.php/surg/article/view/2750.

w3schools (2018) *HTML Responsive Web Design*, *w3schools*. Available at: https://www.w3schools.com/html/html_responsive.asp (Accessed: 11 December 2018).

w3schools (2019a) *Introduction to XML*, *w3schools*. Available at: https://www.w3schools.com/xml/xml_whatis.asp (Accessed: 18 May 2019).

w3schools (2019b) *SQL Tutorial*. Available at: https://www.w3schools.com/sql/ (Accessed: 27 May 2019).

Wagner, K. (2018) *Tinder's business will double this year to more than $800 million*, *recode*. Available at: https://www.recode.net/2018/8/8/17662746/tinder-revenue-match-group-q2-earnings (Accessed: 14 December 2018).

Walter, D. E. and Winterton, S. (2007) 'Keys and the Crisis in Taxonomy: Extinction or Reinvention?', *Annual Review of Entomology*, 52, pp. 193–208. doi: 10.1146/annurev.ento.51.110104.151054.

Watts, T. (1972) *Rocky Mountain Tree Finder*. Berkeley: Nature Study Guild.

Weiss, S. (2018) *Media Center*, *British Ecological Society website*. Available at: https://www.britishecologicalsociety.org/news-and-opinion/press/ (Accessed: 5 December 2018).

Wikipedia (2015) *Wikipedia*. Available at: https://en.wikipedia.org/wiki/Main_Page (Accessed: 26 October 2015).

Wildflower Search (2018) *Oregon Wildflower Search*, *Google Play Store*. Available at: https://play.google.com/store/apps/details?id=com.wildflowersearch.orwildflowers (Accessed: 13 July 2019).

Wildflower Search (2019) *Wildflower Search Apps*, *Google Play Store*. Available at: https://play.google.com/store/apps/developer?id=Wildflower+Search (Accessed: 13 July 2019).

Wilson, B. (2012) *lexicon*, *The Natural Language Processing Dictionary*. Available at: http://www.cse.unsw.edu.au/~billw/nlpdict.html#lexicon (Accessed: 7 July 2019).

Wimmer, F. (2013) *Test Samsung Galaxy S4 GT-I9505 Smartphone*, *Notebook Check*. Available at: https://www.notebookcheck.com/Test-Samsung-Galaxy-S4-GT-I9505-Smartphone.91942.0.html (Accessed: 8 February 2018).

Wimmer, F. (2014a) *Test Samsung Galaxy S5 Smartphone*, *Notebook Check*. Available at: https://www.notebookcheck.com/Test-Samsung-Galaxy-S5-Smartphone.115431.0.html (Accessed: 8 February 2018).

Wimmer, F. (2014b) *Test Sony Xperia Z2 Smartphone*, *Notebook Check*. Available at: https://www.notebookcheck.com/Test-Sony-Xperia-Z2-Smartphone.116233.0.html (Accessed: 8 February 2018).

Wimmer, F. (2014c) *Test Sony Xperia Z3 Smartphone*, *Notebook Check*. Available at: https://www.notebookcheck.com/Test-Sony-Xperia-Z3-Smartphone.126979.0.html (Accessed: 7 February 2018).

Wimmer, F. (2016) *Test Sony Xperia XZ Smartphone*, *Notebook Check*. Available at: https://www.notebookcheck.com/Test-Sony-Xperia-XZ-Smartphone.181041.0.html (Accessed: 8 February 2018).

Winand, M. (2013) *Myth: Select * is bad*, *Use the index, Luke*. Available at: http://use-the-index-luke.com/blog/2013-08/its-not-about-the-star-stupid (Accessed: 29 November 2017).

Wood, M M, Lydon, S. J., Tablan, V., Maynard, D. and Cunningham, H. (2004) 'Populating a Database from Parallel Texts Using Ontology-Based Information Extraction', *9th International Conference on Applications of Natural Language to Information Systems, NLDB 2004*, (June), pp. 254–264. doi: 10.1007/978-3-540-27779-8_22.

Wood, Mary McGee, Lydon, S. J., Tablan, V., Maynard, D. and Cunningham, H. (2004) 'Using parallel texts to improve recall in botany', (January 2003), p. 237. doi: 10.1075/cilt.260.26mcg.

Wood, M. M., Rydeheard, D., Bree, D., Wang, S., Huxley, R. and Sutton, D. (2011) *The Manchester Computational Flora Project*. Available at: http://www.cs.man.ac.uk/~david/flora/computational-flora.html (Accessed: 5 August 2018).

Woodland Trust (2019) *Tree ID - British trees*, *Google Play Store*. Google Play Store. Available at: https://play.google.com/store/apps/details?id=com.woodlandtrust (Accessed: 24 June 2019).

Wu, J., Tombor, I., Shahab, L. and West, R. (2017) 'Usability testing of a smoking cessation smartphone application ("SmokeFree Baby"): A think-aloud study with pregnant smokers', *Digital Health*, 3, p. 205520761770427. doi: 10.1177/2055207617704273.

Yan, J. and Wakefield, R. (2018) 'The freemium (two-tiered) model for individual cloud services: factors bridging the free tier and the paying tier', *Journal of Information Technology Management*, XXIX(1).

Yarmosh, K. (2016) *How Often Should You Update Your App?*, *Savvy Apps blog*. Available at: https://savvyapps.com/blog/how-often-should-you-update-your-app (Accessed: 20 October 2017).

Zaitsev, P. (2009) *How number of columns affects performance ?*, *Percona Database Performance Blog*. Available at: https://www.percona.com/blog/2009/09/28/how-number-of-columns-affects-performance/ (Accessed: 3 November 2017).

Zhong, N. and Michahelles, F. (2013) 'Google play is not a long tail market', in *Proceedings of the 28th Annual ACM Symposium on Applied Computing - SAC '13*. New York, New York, USA: ACM Press, p. 499. doi: 10.1145/2480362.2480460.

Zimmerman, S. (2016) *A Look at What Has Changed from the Snapdragon 820 to the Snapdragon 821 in the Google Pixel Phones*, *xdadevelopers*. Available at: https://www.xda-developers.com/a-look-at-what-has-changed-from-the-snapdragon-820-to-the-snapdragon-821-in-the-google-pixel-phones/ (Accessed: 1 August 2018).

11 Appendices

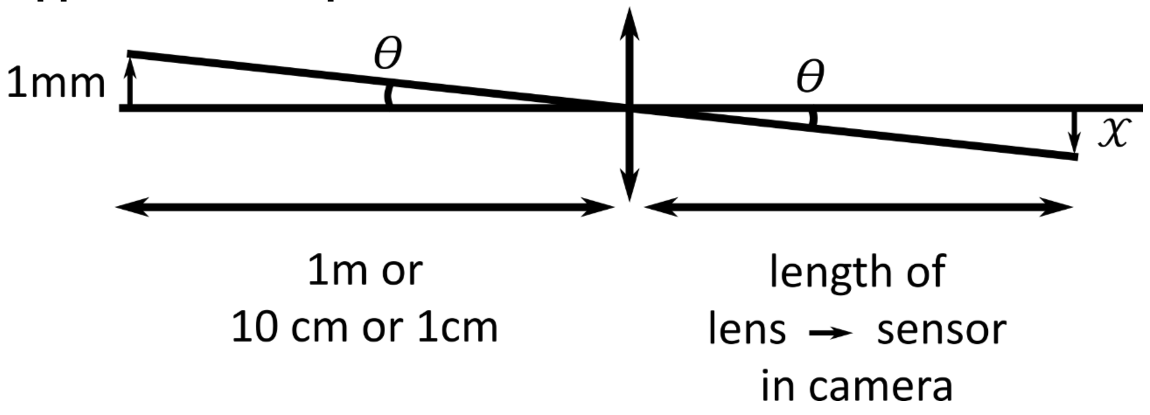
Appendix A The App and full code

Provided electronically on attached DVD.

Appendix B Test photos and correct identifications reached by various photo-recognition apps

Provided electronically on attached DVD.

Appendix C Optics Maths



Appendix figure C-1 - Optical diagram, showing an object of 1mm at a distance of 1m, 10cm, and 1cm. The object is to the left of the figure, the image is to the right of the figure.

With the lens -> sensor length of 3.82mm of an HTC One (see Chapter 1), therefore:

$$\frac{1mm}{1000mm\ or\ 100mm\ or\ 10mm} = tan\theta = \frac{\mathcal{X}}{3.82mm}$$

$$\frac{1}{1000\ or\ 100\ or\ 10} \times 3.82 = \mathcal{X}$$

$\mathcal{X} = 3.82\mu\text{m or } 38.2\mu\text{m or } 382\mu\text{m}$

Appendix D Code

D.1 Autoextraction program

Provided electronically on attached DVD.

Appendix E Databases

E.1 Full database

Provided electronically on attached DVD.

E.2 Pines only database

Provided electronically on attached DVD.

E.3 Corrected *Equisetum* database

Provided electronically on attached DVD.

Appendix F A (very) brief introduction to programming

The following is a very short introduction to the three major programming languages (Java, SQLite, and XML) used by this project. The goal is some familiarisation with what both languages 'look like', in order to aid understanding of the rest of the thesis.

F.1 camelCase and underscore_case

Camel case (stylised as camelCase) and underscore case (stylised as underscore_case) are both used in programming languages where a space ' ' character would imply each word was a different token (a token is a part of the code). camelCase avoids use of space characters by removing them and capitalising every word so they can be identified (Binkley *et al.*, 2009). CamelCase with the first letter capitalised is sometimes called PascalCase (Microsoft, 2017). underscore_case instead replaces every space with an underscore (Binkley *et al.*, 2009). There have been studies showing that while there is a trend towards the use of camelCase, underscore_case is preferable for readability (Sharif and Maletic, 2010).

F.2 Types of data

Programing languages store variables (data points) in a number of different types. Examples are shown below

- Byte: 8 bits (1s or 0s), any value between -128 and 127
- Boolean: 1 bit of information; either true or false
- Character: A single character (usually either 16-bit Unicode or ASCII)
- Integer: a numerical value, typically allowed to be any between -2,147,483,648 and 2,147,483,647.
- String: a set of Characters

F.3 Java

Java is an example of an Object oriented programming language (OOP language). These languages work on the concept of 'objects' that contain data; and code, forming procedures, typically called methods. Two example lines of code in Java are shown below, Appendix code box F-1 howing a generic example line of code, Appendix code box F-2 showing a functional example line of code.

```
Object objectName = Class.doThisMethod(toThisThing);
```

Appendix code box F-1 – an example line of basic Java

```
Pattern doubleQuote = Pattern.compile("\"");
```

Appendix code box F-2 - an example line of functional Java code, compiling a REGEX to a Pattern.

In Appendix code box F-2, a regular expression pattern (REGEX) is cast (created), and is named ‘double quote’. This is shown in the two words ‘Pattern doubleQuote’, before the ‘=’ sign. In Java, as with many programming languages, ‘=’ have very specific meanings. A single ‘=’ means that this named value (in our case the Pattern called doubleQuote) has the value set by the code afterwards. After the ‘=’, in our example, the next word we see is ‘Pattern’. Pattern can only execute one method, ‘compile’. Other classes, for example Integer, have many more methods available (Oracle, 2017a). ‘compile’ from our example takes the string (A data value representing a string of characters representing text, rather than numbers (The Tech Terms Computer Dictionary, 2017)) and compiles it into a REGEX. To refer to this REGEX in the program, ‘doubleQuote’ will need to be referred to. The brackets do not need to refer to a string themselves, and can instead refer to code that will return a string.

An example of a full method in Java is shown in Appendix code box F-3.

```
static String exampleString(String forExample){

        String working = forExample;
        String returnString = "";
        Pattern example = Pattern.compile("\"");
        Matcher matcherExample =
        example.matcher(working);
        if (matcherExample.find()){
                returnString = working;
        }
        else {
                returnString =
        String.valueOf("nothing");
        }
        return returnString;

}
```

Appendix code box F-3- an example generic Java method.

In Appendix code box F-3, exampleString, is passed a string. It accepts this string, and calls it 'forExample'. A second string is created, called 'returnString', and is initially set to have an empty value (set as a string of zero length). The double quote pattern from the previous example is created, and this time is called 'example'. A matcher is created called 'matcherExample', and is set to attempt to match the REGEX pattern 'example' to the string 'forExample'. An if/else option set is then created. This is code that performs a logical choice. In this example, the 'if' statement is followed if the matcher finds what's it has been set to look for (coded as 'matcherExample.find()' in the brackets. If this branch is taken, the code contained within the '{}' is executed. If not the code in the else statement branches '{}' is executed. if/else code can include more options, represented as if/elseif/elseif/…/else.

The code contained within both halves of the if/else statement in the example sets the value of returnString to a string, with both halves setting it to a different option.

The final line of code in the example returns 'returnString'. This is the value that will be returned when this method is called elsewhere in the code.

Native Java includes a large variety of pre-created methods, allowing the basic functionality (for example, the method 'compile' used with REGEX patterns). Third party methods can also be included.

F.4 XML

Extensible Markup Language (XML) is a programming language designed to encode documents in a human and machine readable format. It is similar in syntax to HTML (Hyper-text markup language). In Android, XML is used to encode the layout of various aspects of the app. A basic app page XML layout is shown in Appendix code box F-4.

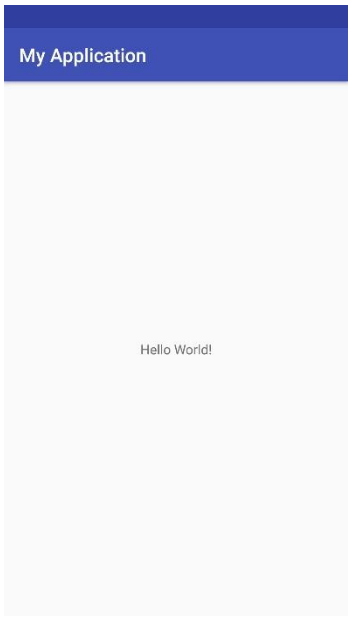
```xml
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context="com.example.sb823249.myapplication.MainActivity">

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Hello World!"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>
```

Appendix code box F-4 - Android XML layout for 'Hello world'.

Appendix code box F-4 codes for the page of the app shown in Appendix figure F-1.



Appendix figure F-1 - The layout coded for by Appendix code box F-4

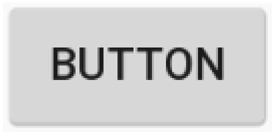
Appendix code box F-4 code is made of two elements, the Layout and its attributes, and a TextView and its elements. This explanation will focus on the TextView element, as while the layout can be modified, it is largely automatically constructed by Android Studio and does not need modification. The TextView element has been highlighted in yellow.

Every XML element opens with an '<' followed by the name of the element being described, in this case a way to display text, called a TextView.

The succeeding lines of code describe and define different aspects of the TextView, all following the same general pattern: name of aspect="how the aspect is defined".

In Android, there are many different possible design details to choose from. Each generally is referred to with an obvious nomenclature. Different XML elements typically have sensible nomenclature, elements displaying text are called textView, buttons are button. Android provides standard designs for most elements, but they can be modified to look however the developer wants. A standard XML element for a button is shown below in Appendix code box F-5, with the button displayed shown in Appendix figure

F-2.



Appendix figure F-2 - The generic Android button

```
<Button
   android:id="@+id/button"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:text="Button"
   tools:layout_editor_absoluteX="0dp"
   tools:layout_editor_absoluteY="16dp" />
```

Appendix code box F-5 – the XML code for the button in Appendix figure F-2

A modified element is shown below in Appendix code box F-6, which shows the button displayed in Appendix figure F-3.

Appendix figure F-3- A button with modified XML to display a refresh icon.

```
<Button
   android:id="@+id/modifiedButton"
   android:layout_width="30sp"
   android:layout_height="30sp"
   android:background="@drawable/ic_refresh_black_48dp"
   android:onClick="reset"/>
```

Appendix code box F-6 - the XML required to create the refresh icon in Appendix figure F-3.

The line of code from Appendix code box F-6 shown in Appendix code box F-7 is the line required to change from the default to the modified look. The other lines in Appendix code box F-6 modify the size and location, and provide additional information to the button.

```
android:background="@drawable/ic_help_outline_black_48dp"
```

Appendix code box F-7 - The single line of XML required to alter the look of a button in Android.

F.5 SQLite

SQL (Structured Query Language (Encyclopedia Britannica, 2017)) is a programming language designed to work with databases. SQLite is an implementation of SQL databasing system in the C (SQlite, 2019), designed to accept SQL commands and run as a database management system. Android fully supports the SQLite format of SQL databases and includes a tool called sqlite3 for debugging purposes (Android Developers, 2016a). SQL is structured as commands, or statements, and two basic examples are shown in Appendix code box F-8 and Appendix code box F-9.

```
INSERT INTO exampleTable (specificID, column_1, column_2) VALUES (1, a,
banana_pudding)
```

Appendix code box F-8 - Example SQLite insert statement, which inserts the values into the specified columns in the specified database.

```
SELECT * FROM exampleTable
```

Appendix code box F-9 - Generic example SQLite select statement.

Appendix code box F-8 is an INSERT statement. It will insert a row into the datasheet shown in Appendix table F-1 updating it to the datasheet shown in Appendix table F-2.

Appendix table F-1– Example table, prior to executing the command shown in Appendix code box F-8 - Example SQLite insert statement, which inserts the values into the specified columns in the specified database.

specific ID	column_1	column_2

Appendix table F-2 - Example table, after executing the command shown in Appendix code box F-8 - Example SQLite insert statement, which inserts the values into the specified columns in the specified database.

.

specific ID	column_1	column_2
1	a	banana_pudding

Appendix code box F-9 is a SELECT statement. This will search the table using the criteria specified. In this example, the search criteria is '*', and the table is 'exampleTable'. In SQLite '*' refers to a blank statement: when used as the search criteria it returns every entry in the table; when '*' is used to specify which table is to be searched, every table within range will be queried. In this case, the example search would return everything currently contained within 'exampleTable'. Typically search criteria are Boolean tests, in the form 'WHERE x=y', where x is the column name, and y is the requested variant.
Other commands can be included, the search criteria can be expanded, and multiple databases can be search simultaneously, allowing for a highly complex and potentially very efficient database structure.

F.6 The three languages and Android

Android apps are primarily programmed using Java (Android, 2017). It has been designed so that the three different languages are usable together to produce a functioning app.

Java classes can refer to individual XML layout files. This allows them to receive input from them, or manipulate what is displayed to the user. This is done using the line of code shown in Appendix code box F-10.

```
setContentView(R.layout.activity_main);
```

Appendix code box F-10 - line of Android where the current view is set to the XML layout code named 'activity_main'.

In Appendix code box F-10 the XML layout file ‘activity_main’ is being referred to.

One of the aspects an XML element can have is the ‘id’ aspect. This ‘id’ is referenceable in the Java class files. Two examples are shown in Appendix code box F-11 and Appendix code box F-12.

```
exampleTextView = (TextView) findViewById(R.id.example_text_view);
```

Appendix code box F-11 - a generic example line of Android Java instructing a textView to relate to the required XML labeled textView

```
exampleTextView.setText(“example_string”);
```

Appendix code box F-12 - an example line of code, setting a textView to display a string of text

In Appendix code box F-11, the object ‘exampleTextView’ is told to function as a TextView, and then is told to refer to the XML element ‘example_text_view’. If ‘example_text_view’ is not a TextView, then an error will probably be produced, and the app is unlikely to function correctly. Appendix code box F-12 sets the object ‘exampleTextView’ to display the value “example_string”. As ‘exampleTextView’ already refers to the XML element labelled ‘example_text_view’, ‘example_text_view’ will display the string “example_string”.

SQLite is run directly in the JAVA code. Two examples are shown in Appendix code box F-13 and Appendix code box F-14, both examples are two lines of code.

```
String insertOne = “INSERT INTO exampleTable (specificID, column_1,
column_2) VALUES (1, a, banana_pudding)”;
database.execSQL(insertOne);
```

Appendix code box F-13 - the two lines of java code required to run the SQLite example from Appendix code box F-8 in Android.

```
String query = “SELECT * FROM exampleTable”;
database.execSQL(query);
```

Appendix code box F-14 - the two lines of Android Java required to run the SQLIte example from Appendix code box F-9 in Android.

In both Appendix code box F-13 and Appendix code box F-14, the SQLite command is stored as a string. This allows you to create and modify the SQLite command as needed. The following line in each example then executes the SQLite command.

Appendix G Additional information for chapter 6, App testing

G.1 Session one

The students suggested the following feedback in the meeting. Points 1-9 were provided by the students as notes and are quoted directly, points 10-25 are notes from the meeting.

1) Slider endpoints too close to the edge of the screen.
2) Text at top need not be so long. e.g. “You do not need to answer all questions” or “Answer only the questions you can”
3) Would radio buttons be better for more of the questions? e.g. is the height so critical that it needs to be precise, or could it be (eg) 5-15cm, 15-25cm etc. Especially on smaller screen devices, the radio buttons are easier to use.
4) What order are the questions in? Given that you have on-the-fly analysis after each answer to produce a number of possibilities, would it be worth having those that are most defining at the beginning? Or the questions that are easiest to answer (perhaps this is what you have done)?
5) * ‘What colour is the **sheath?’** (typo)
6) For clarity, especially on smaller screens, some of the diagrams could do with redrawing to make them more schematic, or alternatively replaced with photos. Likewise, those variables (eg where are the cones located?) which are explained with text: it would probably make things easier for users if there were representative images of the options you are describing
7) Do you know whether this is likely to be used on mobile phone (ie small screen) or tablet (large screen) devices? If it is the former, some of the type is a bit small.
8) If you add sufficient variables to reduce the number of possibilities to a small number, e.g. 3, you are given information about the three
9) Not sure it is helpful to have a (?) tap-through for those variables that are self-evident, or provide no particularly useful further explanation (eg what is the colour of fertile stems)?

10) Remember phones are small
    a) Question list is long
11) Make it so questions are impossible to answer once you have reached a single possibility
12) If possible – give possibilities in results
13) Butterfly key can get down to three questions
14) Images need to be more schematic instead of illustrative
15) Include scale bars wherever possible
16) Text from Wikipedia can be very long (an example of extra information pop-up is shown in Figure 7-6 A)
    a) A little more space between the lines would be easier to read
17) It would be nice for people to be able to keep records of the locations they found species, especially if you can take photos!
    a) Maybe change one of the buttons on the extra text for each species to go directly through to the camera
18) Radio buttons with range questions
    a) Rather than sliders
        i) Easer to use, especially when cold
    b) Specificity makes it look like being so precise is necessary
19) Increase the size of the margin a bit, as a number at the edge of the screen can make it look like the app isn't being displayed properly.
20) Change the colour of the top where the number of results is displayed to a different colour from the search button.
21) Ensuring that the search button actually looks like a button.
22) What is the closest colour to, rather than what is the colour of {x}?
23) Possibly number the questions
    a) It takes space
    b) It gives a reference
24) The teeth length question appears to have smaller than normal text
25) The clear all button looks like it's clearing the Equisetum category, rather than the options
    a) Maybe move it to the bottom instead

b) Like the slightly older version, but at the bottom instead
c) Search button at the bottom right
d) Don’t bother with floating action buttons

G.2 Session two, Herbarium volunteers session one

G.2.1 <u>Feedback generated through discussions with the volunteers during the session</u>

1. Stem height question needs indication that it’s from ground level
2. Stem diameter doesn’t indicate at what point on the stem it is measured at
3. The GPad crashes on radio button use
4. ‘Internode’ was missing a ‘t’ in question eight
5. Branching away from the main stem can be interpreted as branches on branches/it isn’t clear what the meaning is of the question
6. Q6 cannot clear
7. Clearing questions individually doesn’t always appear to re-set the number of specimens
8. No indication of which teeth to measure. Teeth at different points on the stem can look different to other teeth
9. Rough to touch help text has an unexpected ‘r’ in it
10. Black band question is very hard to understand
11. Branching away from the stem question help text is also hard to understand
12. Cone diameter needs a unit
13. Not a pathed key if multi-access – you can’t as easily trace your work back and change your mind
14. Branches question – branches might be interpreted as leaves – might cause confusion
15. Ridges – need to look under the sheath to get the right value
16. Telmatia might have a matrix error – stem diameter can easily go down to 5mm
17. Branch density might be useful to have in the key

G.2.2 Answers provided to the questionnaires

G.2.2.1 Results of question two (Identification of specimens)

Appendix table G-1 - Number of complete (where users managed to successfully generate a potential identification) and incomplete (where users started to identify a specimen but couldn't generate a potential identification)

	Total attempts	Incomplete attempts	Completed attempts	Percent completed
App	17	3	14	82.35%
Stace (2010c)	16	5	11	68.75%

G.2.2.2 Results of question three (Was the key easy to use?)

Scoring the results from 1 (Easy) to 5 (Difficult), and averaging the score gives the results shown in Appendix table G-2

Appendix table G-2 - Usability score

App	2.25	4.25	Stace

G.2.2.3 Results of question four (What aspect could be most improved when using this ID key under these circumstances?)

The following were given as answers for this question.

Stace (1997 and 2010):

- Use easier language to understand
- More practice and a glossary to hand
- Vegetative characters

App:

- Use easy to understand language
    - Explain vocab
- Sliders are a bit sensitive – can be tricky to get the right number
- Slidy bar not touch sensitive enough
- Not obvious enough that answers have got down to 0, star warning could appear next to unanswered questions when this happens
- Something to indicate which questions distinguish the remaining species – was stuck on two species for ages!

G.2.2.4 Results of question five (What aspect of this ID key did you think was the most useful?)

Appendix table G-3 - Number of respondents indicated various aspects of the ID keys as 'most useful'

Question	App	Stace (Stace, 2010c)
Speed	3	0
Easy to use	4	0
Reliable	0	2
Easy to understand	3	1
Familiarity	0	2
Design	2	0

Feedback from the 'other' option for the app:

- Don't have to answer all questions
- Info buttons with descriptions are very useful
- Choice of questions

Feedback from the 'other' option for Stace (2010c):

- Gave at least two options for each couplet

G.2.2.5 Results of question six (In the field, would you use this method?)

Appendix table G-4 - Percentage of respondents who indicated tehy would use the ID key when in the field

App	100.00%	75%	Stace

G.2.2.6 Results of question seven (If you feel that there are any other aspects of the ID key tested that need to be mentioned, please include them here)

The following was given as answers for this question:

App:

- When [you] get to [a] final answer – a picture with notes about the specimen would help to see if [the] answer is likely to be correct

- More illustrations of features

G.3 Session three, Herbarium volunteers session two

G.3.1 List of specimens made available to the volunteers

Appendix table G-5 - List of specimens made available to the volunteers in this session

Taxa	Label number during the session	Identifiable number
E. hyemale L.	1	
	2	
E. x moorei Newman	3	McClintock 1956
	4	Jury and Rumsey 1983
E. x trachydon A. Braun	5	Mackechnie 1939
	6	Unidentified 1949
E. ramosissimum Desf.	7	
	8	
E. variegatum Schleich. ex F. Weber & D. Mohr	9	Wallace 1963
	10	Lousley 1972
E. fluviatile L.	11	Lousley 1926
	12	Wallace 1933
E. x litorale Kühlew. ex Rupr.	13	Not easily identifiable
	14	Boley 1943
E. arvense L.	15	Lousley 1924
	16	WAS (W. Arthur Sledge?) 1936
E. x rothmaleri C.N. Page	17	
E. pratense Ehrh.	18	Reylands 1858
	19	Lousley 1959
E. sylvaticum L.	20	Wallace 1940
	21	Wallace 1942
E. x bowmanii C.N. Page	22	Bowen 1989
	23	Garnet and Stanley 1998
E. palustre L.	24	Wallace 1973
	25	Wallace 1934

E. x font-queri Rothm.	26	C + S 1998
E. telmateia Ehrh.	27	Lousley 1933
	28	Mackechnie 1935
E. x meridionale (Milde) Chiov.	29	Girevd 1991
	30	Berger 1998
E. x dycei C.N. Page	31	EDIN 834099
	32	EDIN 834098
E. x mildeanum Rothm.	33	EDIN 31003
	34	EDIN 712013
E. x mchaffieae C.N. Page	35	EDIN 664535
	36	EDIN 664536
E. x robertsii Dines	37	EDIN 834287
	38	EDIN 834289

G.3.2 Modified questionnaire used in session

The questionnaire used in this session was slightly from the previous session, and is shown below

1. What key are you currently using?

   ☐App
   ☐Paper (please specify which one below)

   ____

2. Provide the ID number and your identification for each specimen.
   Please fill in only the questions you answered providing the answer you gave, and the order in which you answered. IE if you answered question 4 first, for question 4 put a '1' in the order box, and the answer you gave in the answer box. If you are using the couplet key, you only need to put what option you chose in the correct couplet box.

Specimens	1		2		3	
Sheet ID number						
Question/Couplet	Order	Answer	Order	Answer	Order	Answer
1						
2						
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						

18						
19						
Species						

3. Was the key easy to use? Tick appropriate box

Easy ☐ ☐ ☐ ☐ ☐ Difficult

4. What aspect could be most improved when using this ID key under these circumstances?

5. What aspect of this ID key did you think was the most useful? (Tick as many as you think are relevant!)

☐Speed

☐Easy to use

☐Reliable

☐Easy to understand

☐Familiarity

☐Other (please specify below)

____

6. In the field, would you use this method? Please assume equal coverage of the ID guides, accuracy of the guides, etc.

☐Yes

☐No

7. If you feel that there are any other aspects of the ID key tested that need to be mentioned, please include them here

G.3.3 Answers provided to the questionnaires

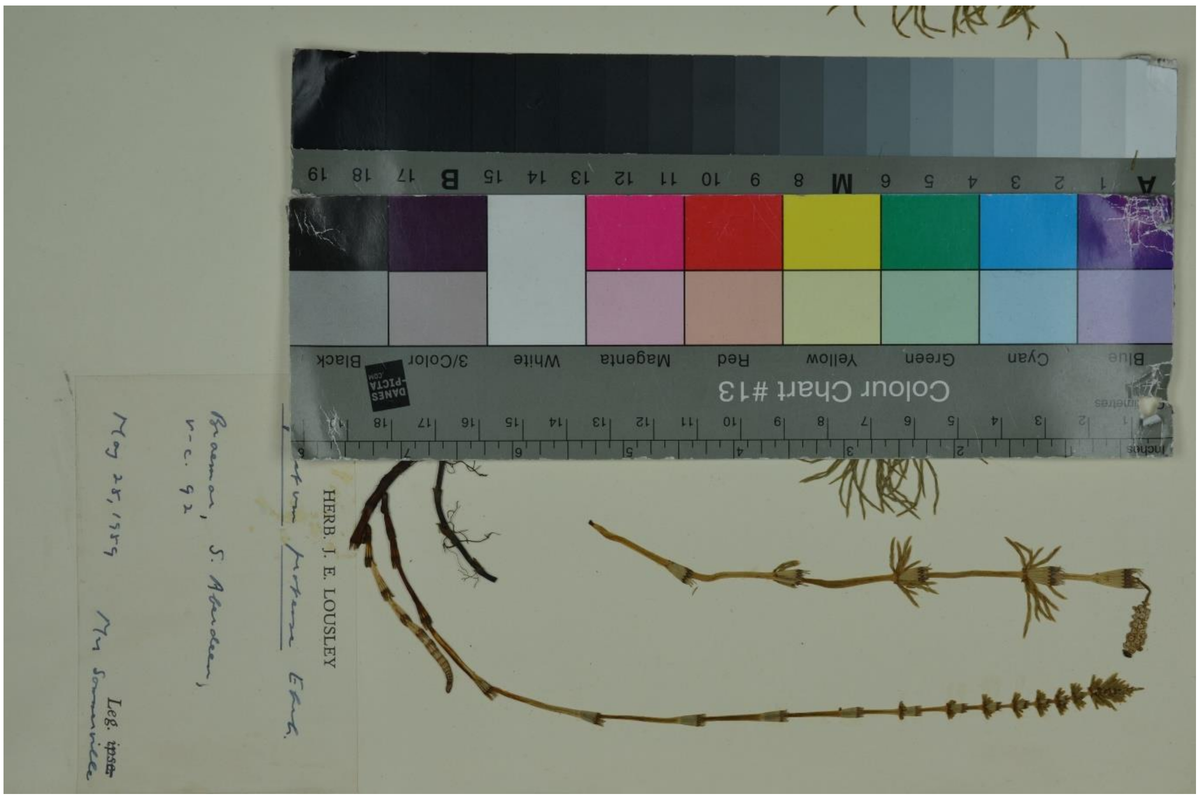
G.3.3.1 Results of question two (Identification of specimens)

Both paper based keys and the app had a 100% completed attempt rate and 13 attempts. The full set of results for each attempt, including probable causes for error are shown in Appendix table G-6.

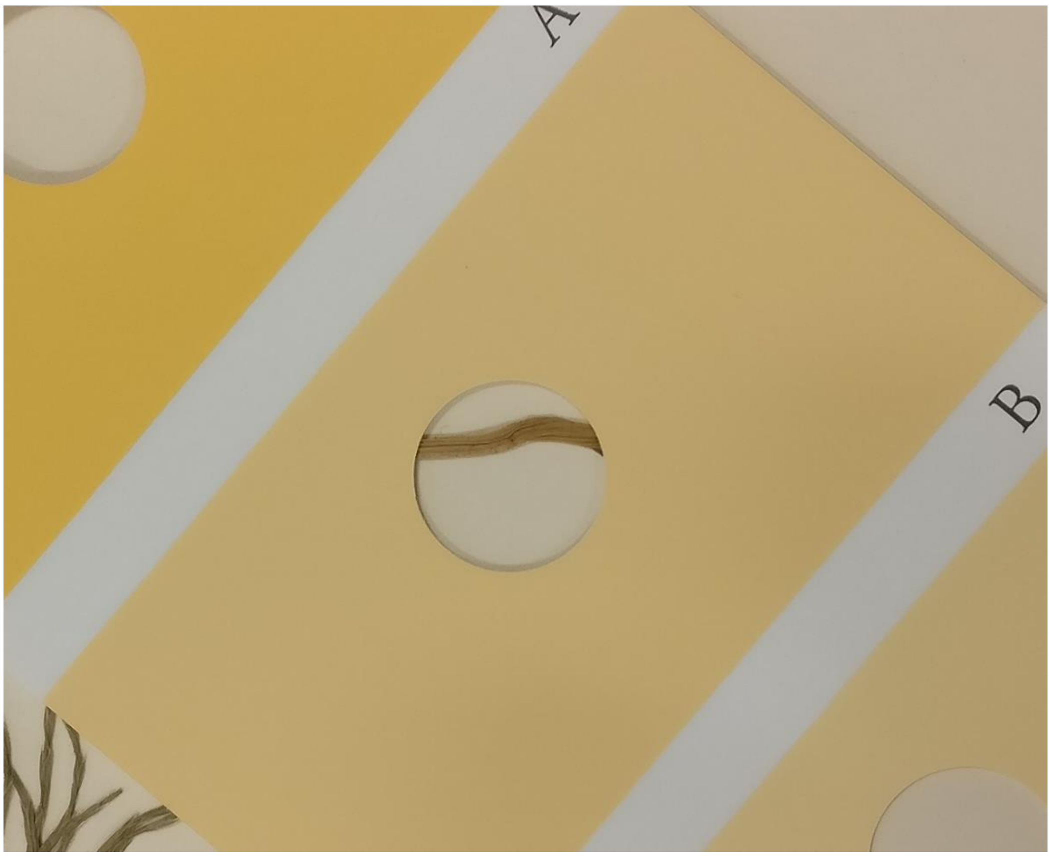
Appendix table G-6 - Correct identification, identification(s) reached, and probable causes of error where identifications were incorrect, for various herbarium specimens when identified by volunteers using the app.

Correct Identification	**Identification(s) reached**		**Error**
E. telmateia Ehrh.	*E. arvense* L.		User did not double number of ridges
E. palustre L.	*E. fluviatile* L.	*E. x mcaffieae* C.N.Page	User error, clearly input wrong stem width
E. palustre L.	*E. x robertsii* Dines		Herbarium specimen cones appear to be unusually small - either a damage issue or a data fuzziness issue. See Appendix figure G-3.
E. x bowmanii C.N.Page	*E. x mchaffeie* C.N.Page	*E. x robertsii* C.N.Page	correct that it was 8, possible data error
E. x font-queri Rothm.	*E. x litorale* Kühlew. ex Rupr.	*E. x robertsii* Dines	data error
E. telmateia Ehrh.	*E. x robertsii* Dines		Question badly answered
E. pratense Ehrh.	*E. x mchaffeie* C.N.Page		Herbarium specimen loss of colour. See Appendix figure G-1 and Appendix figure G-2.
E. arvense L.	*E. arvense* L.	*E. x robertsii* C.N.Page	n/a
E. sylvaticum L.	*E. x meridionale* (Milde) Chiov.		potential data error
E. x meridionale (Milde) Chiov.	*E. palustre* L.		Potential user error/Data fuzziness issue - 5mm is correct and maintains x merid, 6mm eliminates all options
E. arvense L.	*E. x robertsii* Dines		Question badly worded - teeth are longer on fertile stem

E. sylvaticum L.	*E. variegatum* Schleich. ex F. Webber & D. Mohr		Herbarium specimen damaged cone. See Appendix figure G-3.
E. x bowmanii C.N.Page	*E. x bowmanii* C.N.Page		n/a

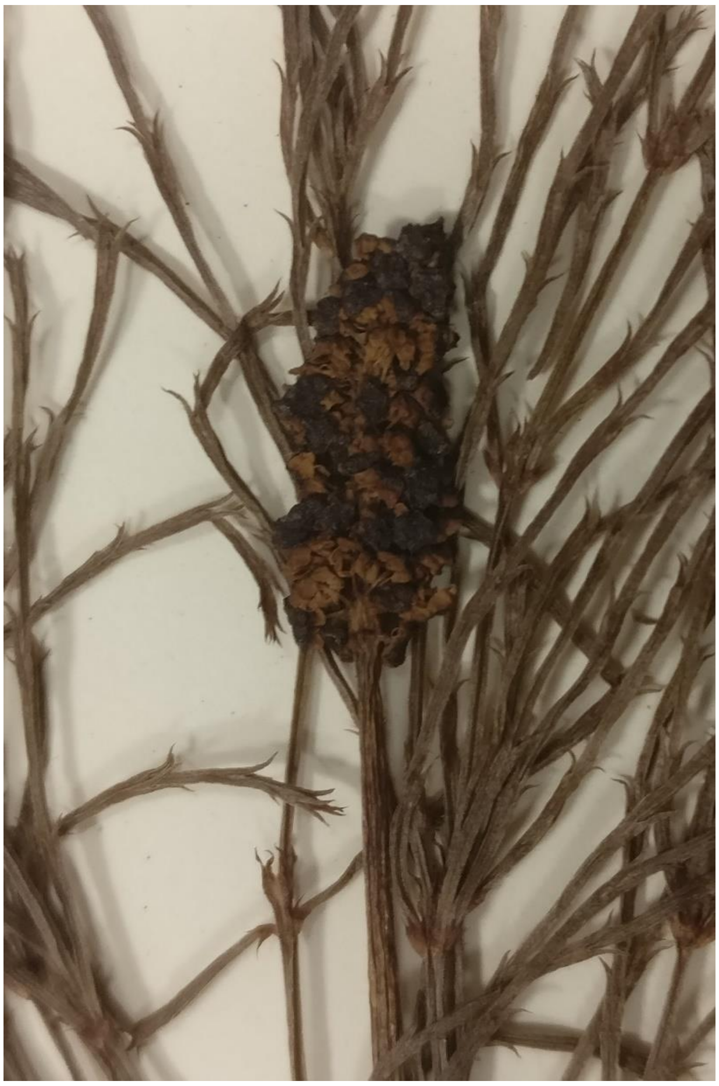


Appendix figure G-1- Specimen identified in a session as having a 'yellow' stem, pictured with a colour chart for reference



Appendix figure G-2 - A section of stem shown in Appendix figure G-1 compared to RHS colour card Yellow-Orange Group no. 19, option B

.



Appendix figure G-3 - Damaged cone from a specimen where the cone was described as being smaller than expected for the species

G.3.3.2 Results of question three (Was the key easy to use?)

Scoring the results from 1 (Easy) to 5 (Difficult), and averaging the score gives the results shown in Appendix table G-7

Appendix table G-7- Usability score

App	1.8	3.6	Book

G.3.3.3 Results of question four

The following points of feedback were given as answers for this question

Paper (unrecorded)

- “it is a perfect characters you are mention if in this keys”

New Flora of the British Isles (Stace, 2010c)

- "Wider use of characters. Avoiding perennial/annual stem choice"

Field flora of the British Isles

- "Key heavily reliant on colour, growth (e.g. annual) and therefore is difficult at some couplets. Especially the first."

Collins flower key (Streeter *et al.*, 2009)

- "Pictures/explanations of key features"

Vegetative key to the British Flora (Poland and Clement, 2009)

- "Probably best in the field"

App

- "Fuller explanation when using help"
- "Some of the variable[s] do seem subjective e.g. regarding teeth border width."
- "Quite difficult to narrow it down to anything other than a hybrid at times. "
- "More pictures"
- "Have photos as result not just name?"

G.3.3.4 Results of question five

Appendix table G-8 - Number of respondents indicated various aspects of the ID keys as 'most useful'

Question	App	Book
speed	3	3
easy to use	4	1
reliable	0	0
easy to understand	4	2
familiarity	0	1
design	1	2

G.3.3.5 Results of question six

Appendix table G-9 - Number of respondents indicated various aspects of the ID keys as 'most useful'

App	100%	60%	Book

G.3.3.6 Results of question seven

The following feedback was given about the app

- Having a phone-based key is really useful in terms of portability. It would be good to be able to record/email the key choices once ID has been completed.
- Reset button on couplet two is broken I think

The following feedback was given about Stace (2010c):

- The key is reliant on cones which are often missing.

The following feedback was given about Stace (1999)

1st specimens, was unsure about what ID at couplet 13 and therefore checked 14 to confirm specimen didn't fall out there.

G.4 Session four

G.4.1 <u>Feedback generated during the session</u>

1. An instruction guide would be helpful
2. A user guide would be helpful
3. More pictures if possible would be helpful
4. Sliders can move when you lift your finger
5. Finger hitbox on sliders might be too small
6. Possibly a bit flexible – ridge count is discrete might be possible
7. Wikipedia is the internet, the app requires the internet!

   Getting a list of zero results is irritating.

Appendix H Additional data for Chapter 7, Business case

H.1 Full data for free apps

Provided electronically on the attached DVD.

H.2 Full data for non-free apps

Provided electronically on the attached DVD.

Appendix I Gantt Chart of timescale/milestones

Month	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3	4	5	6	7	8	9	10	11	12
Year	2015												2016											
	Literature review	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
						6 month report	🟩																	
								Work on auto-extraction program	🟩	🟩	🟩	🟩	🟩	🟩	🟩									
																Manual data extraction	🟩	🟩	🟩					
																				Initial app creation	🟩	🟩	🟩	🟩
																								TDWG conference

1	2	3	4	5	6	7	8	9	10	11	12	1	2	3	4	5	6	7	8	9	10	11	12
2017												**2018**											
Literature review																							
Initial app creation																							
						Session 1																	
							Adjustments to app based on feedback																
												Session 2											
													Adjustments to app based on feedback										
															Session 3								
																Adjustments to app based on feedback							
						Thesis write up																	
																							Sumbission