

Faculty of Engineering and Environment

Department of Computer and Information Sciences

CM0645/KV6003 Individual Project

2018-2019

MICHAEL BUGLASS

BSc. (Honours) Computer Science

SOFTWARE ENGINEERING PROJECT

SUPERVISOR: Shelagh Keogh

2ND MARKER: Nick Dalton

A Mapping Application to Collect and Visualise Crowdsourced GPS Data

# Authorship Declaration

**DECLARATIONS**

I declare the following:

(1) that the material contained in this dissertation is the end result of my own work and that due acknowledgement has been given in the bibliography and references to **ALL** sources be they printed, electronic or personal.

(2) the Word Count of this Dissertation is ...................................

(3) that unless this dissertation has been confirmed as confidential, I agree to an entire electronic copy or sections of the dissertation to being placed on the eLearning Portal (Blackboard), if deemed appropriate, to allow future students the opportunity to see examples of past dissertations.  I understand that if displayed on eLearning Portal it would be made available for no longer than five years and that students would be able to print off copies or download.

(4) I agree to my dissertation being submitted to a plagiarism detection service, where it will be stored in a database and compared against work submitted from this or any other School or from other institutions using the service.

In the event of the service detecting a high degree of similarity between content within the service this will be reported back to my supervisor and second marker, who may decide to undertake further investigation that may ultimately lead to disciplinary actions, should instances of plagiarism be detected.

(5) I have read the Northumbria University/Engineering and Environment Policy Statement on Ethics in Research and Consultancy and I confirm that ethical issues have been considered, evaluated and appropriately addressed in this research.

SIGNED:

# Acknowledgements

Application User Interface designs make use of the Material Design UI kit, a free resource available to download on the official Adobe XD CC website. (<https://www.adobe.com/products/xd/resources.html>)

IEEE SRS Template used in this project (Appendix D) is Copyright © 1999 by Karl E. Wiegers. (<https://www.processimpact.com/>) Permission is granted to use, and modify this template for use within personal projects.

Original data from the GeoNames geographical database (http://www.geonames.org/) is used within the product, this data is licensed under Creative Commons Attribution 4.0 International Public Licence (CC BY 4.0) (<https://creativecommons.org/licenses/by/4.0/>). This allows the adaptation and redistribution of the material in any medium or format. Data used is in the names of major global cities.

# Abstract

Crowd-sourcing is a practice in which information is collected through the medium of a large crowd of people and offers a cost-effective method of data collection. This project explores the way in which the crowdsourcing of data is conducted, following legal and ethical practices. As a platform on which to collect this data, the design and implementation of Android applications will be investigated in a way that adheres to guidelines and good practice. Mobile interfaces will be studied to investigate attributes that affect usability and user experience so that maximum user satisfaction can be achieved.

Combining this information, this project aims to produce an application for Android devices that will facilitate the collection of crowd-sourced data, and display submitted data to users in a way that is simple, intuitive, and functional. Sourced data will be in the form of locations of interest, which will be displayed to the user through a map interface using GPS technologies. Ideally, users will be able to use this application to share data about a location in a more personal way than is provided by commercial mapping solutions, and communities will be able to mark areas of interest they deem interesting or useful for other members. There are no limitations to what users can mark, allowing for a greater sense of freedom in the data they submit, and variety in the data sourced.

Lack of such a product available on a mobile platform was identified, and since the portability of mobile devices lends itself well to such GPS applications, it was considered undertaking this project would produce a useful and exciting product that people may find beneficial.

Following a tailored agile approach, the application was produced using Java and XML using the Android Studio IDE. To present and collect data usefully, the Google Maps for Android SDK has been implemented. Cloud database solutions have been implemented to allow concurrent access to shared data across several devices simultaneously.

Various forms of testing, including user testing conducted using volunteers, and well as automated JUnit testing have been carried out to conclude the product meets core requirements and the primary aim of the project.

Table of Contents

[Authorship Declaration i](#_Toc7469962)

[Acknowledgements ii](#_Toc7469963)

[Abstract iii](#_Toc7469964)

[1 Introduction 1](#_Toc7469965)

[1.1 Aims and Objectives 1](#_Toc7469966)

[1.2 Overview of Product 2](#_Toc7469967)

[1.3 Background to Problem Context and Justification of Project Undertaking 2](#_Toc7469968)

[1.3 Scope 3](#_Toc7469969)

[1.4 Main Features and Characteristics 3](#_Toc7469970)

[1.5 Approach and Tools Used 4](#_Toc7469971)

[2 Analysis 5](#_Toc7469972)

[2.1 Review of Literature Relevant to the Proposed Project 5](#_Toc7469973)

[2.2 Justification of Tools and Resources Required 11](#_Toc7469978)

[2.3 Overview of the planned approach 13](#_Toc7469986)

[2.4 Justification of Analysis Deliverables 13](#_Toc7469987)

[2.5 MoSCoW Analysis of Application Requirements 14](#_Toc7469988)

[2.6 User Interface Designs 19](#_Toc7469991)

[3 Synthesis 20](#_Toc7469992)

[3.1 Class Diagram 20](#_Toc7469993)

[3.2 Database Specification and Design 21](#_Toc7469994)

[3.3 Product Code 22](#_Toc7469995)

[3.4 Testing 24](#_Toc7469996)

[4 Evaluation and Conclusions 30](#_Toc7469999)

[4.1 Evaluation of Product 30](#_Toc7470000)

[4.2 Evaluation of Project Process 36](#_Toc7470004)

[4.3 Conclusion and Recommendations 38](#_Toc7470005)

[5 References 40](#_Toc7470006)

[6 Glossary 42](#_Toc7470007)

[7 Appendix 43](#_Toc7470008)

# Introduction

## Aims and Objectives

The project aims to:

* Investigate the design and implementation of Android applications that adhere to Google play terms of service and guidelines. The use of good practice and recommended solutions will be explored along with the criterion that dictates the effectiveness of user interface designs. Integration of external SDKs will also be researched to make the best use of the available tools.
* Analyse, design and build of an Android application that collects and displays crowdsourced GPS location data, through the use the of Maps for Android SDK, in an intuitive and functional way.

The objectives of this project are:

* Research and review literature covering the process and techniques used when crowdsourcing data, and how this data is processed, to ensure the product can perform this task in a way that uses good practice and is ethically just.
* Research and review literature covering the analysis, design and development of Android applications using Java, to improve knowledge and familiarity with such technologies, taking into consideration usability and accessibility requirements.
* Construction of an IEEE requirement specification, for determining the functionalities of the product.
* Improve knowledge and understanding of the use of the Android Studio IDE, Java in the context of Android development, and integration of the Google Maps for Android SDK.
* Design and build the product using good practice, making use of JavaDoc comments where possible to improve code clarity and aid future maintenance.
* Analyse product requirements using use case diagrams and descriptions, interaction storyboards and entity relationship diagrams.
* Construct and carry out a testing strategy, employing the use of volunteers for user testing and using automated JUnit testing where possible to implement efficient industry practice.
* Evaluate the product against requirements and assess the project process, making recommendations to the future of the product produced.

## Overview of Product

The product produced is an Android application, compatible with versions 4.0.3 and higher, that allows the collection and presentation of crowdsourced data using a Google map. Development of this project aims to fill a gap-in-the-market for a mobile, community-based mapping tool to share points of interest with other users. Many existing applications that provide locational information only use official data explicitly chosen by developers, not the users. Some crowd-sourced mapping applications that do exist are generally limited to specific purposes, such as Waze; a satellite navigation application that allows drivers to submit data on road conditions and uses this information specifically for navigation purposes.

Allowing users to mark points of interest covering various categories, the product hopes to provide a platform for locational data to be shared on a mass scale. Through filtering of the displayed data, users can find specific things they may be looking for. In large unfamiliar cities, for example, it can be challenging to locate facilities such as restrooms; while in the countryside it may prove challenging to find a suitable place to dispose of rubbish.

## 1.3 Background to Problem Context and Justification of Project Undertaking

Initially, the idea for the application stemmed from attempting to locate street art in Newcastle-upon-Tyne to help with a friend’s project. However, there was no online sources or map that provided any information on where to find such artwork. Initially, the first concept of the product would have filled this gap and provided a platform for artists and enthusiasts to share locations of street art. This concept was expanded to allow users to mark anything they desire, rather than limiting markers to artwork – but allow the users to filter marker to meet their needs when viewing the map.

The undertaking of this project was worthwhile to provide insight into the vast industry of mobile development, an industry that has seen little coverage throughout university thus far. Many unfamiliar technologies were used, such as a new development environment, new languages, new libraries, and software development kits that offered a significant challenge. Opportunities were present to expand upon existing skills, such as object-orientated programming, in a different context to had previously been experienced, as well as learn new ones; such as mobile user interface design and testing of mobile software using automated JUnit classes. Crowd-sourcing the data also offered an exciting area of research, which proved to give inspiration to many new features for the application that had never been considered; such as allowing crowd curation of content through a voting system and using artificial intelligence to suggest areas of interest based upon patterns detected in the previously visited locations.

## Scope

The scope of this project is to produce a functional prototype for the application described, which incorporates all core functionalities to allow for the collections and presentation of data in a way that is useful to the user. More complex features will be considered where possible, but the application produced will not ready for release – this will be left to the future of the project.

## Main Features and Characteristics

The main functional components contained in the product are:

* A functional user management system, allowing secure account creation and log in – as well as allowing users to change their password in the application either through the setting or using the ‘forgot password’ button, which sends a password reset link to the user’s email address.
* A real-time cloud database, used to store information about users and locations and allow this data to be accessed concurrently by up to 1,000,000 users, allowing the application to be used across many devices simultaneously.
* A Google map that visualises the data stored in the database and allows users to search locations around the world using the autocompleting search bar, that dynamically suggests locations to the user as they type. Settings are also available to turn the map to ‘dark-mode’, to conserve battery life, and to turn default labels on or off depending on the user’s personal preference.
* The ability to view details of an existing marker, including title, category, description and the user who placed the marker – as well as allowing users to delete markers that belong to them.
* The ability for users to add new markers to the database, by holding down on the map location where they want the marker to appear, then entering information about this marker for others to see.
* The ability to filter markers using any desired combination of categories, to limit the markers displayed solely to what the user is interested in.

## Approach and Tools Used

The approach used is a tailored agile methodology; as these are known to perform well in projects using new and unfamiliar technologies. Testing will be performed frequently as the code is added to the project, allowing bugs to spotted early and often thus preventing them from becoming harder to correct as the system grows. This testing will be carried out in the chosen IDE, Android Studio, using built-in android emulation software as well as a physical Android device. Testing will be conducted both manually, using both black box and clear-box methodologies, as well as making use of automated JUnit testing.

Other tools used within this project will include:

* Android Studio IDE
* Google Maps SDK for Android
* Firebase SDK
* Microsoft Office Suite
* Adobe Creative Cloud Suite
* StarUML
* GitKraken

Justification for the use of these tools, as well as alternatives considered, is provided in the next section.

# Analysis

## Review of Literature Relevant to the Proposed Project

### Approaches to The Use of Crowd Sourced Data

A consistent theme among papers regarding the use of crowd-sourced data is the concern over the quality of harvested data and, in some cases the concerns of privacy. [Amsterdamer and Milo (2015)](#_References) outline the challenges that must be overcame when looking to crowdsource data, as users will have varied expertise and reliability; and their time, memory and attention span will be limited. Answers can also turn out to be approximate, contradictory, or even malicious, leading to data being of little use. However, to combat the unreliability of the harvested data it is proposed that the crowd can not only be used for the harvesting of new and missing data but also to check the quality of existing data. Allowing the crowd to self-critique data minimised the work on the development end compared to method requiring the harvester to perform such tasks, as all work can be left to the end user.

Handling quality control is also possible without using the crowd’s feedback. [Amarilli et al. (2004)](#_References) explore how to reduce uncertainty in crowd-sourced data, as quality control is essential to ensure data is reliable. One method discussed involves deriving estimates of the correct answer to a posed question through the analysis of all inputs to eliminate the uncertainty. By measuring the average and expected answers to questions, a system can be trained to classify answers and spot anomalies thus removing uncertainty automatically. One downside to this proposed method is the requirement for large datasets to train a machine learning system accurately, requiring a large crowd to answer seemingly identical questions. There also comes the issue of deciding the minimum data required to ensure a correct answer can be determined. This method is also most effective with questions that result in a Boolean answer, as opposed to qualitative answers. [Amarilli et al. (2004)](#_References) also state that deciding the next best question to ask a crowd member is also important, where some questions may require more data than others to determine the correct answer. Since an individual crowd member has limited engagement, it is not possible to assume that they will answer all questions, so asking the least answered questions improves the balance of question response rates.

Similarly to the methods mentioned by [Amarilli et al. (2004)](#_References), [Boim et al. (2012)](#_References) discuss the importance of asking the right questions, and present ‘AskIt!’; an implementation of a recommendation system that allows applications to determine which questions should be asked to which users to ensure the most exceptional quality in harvested data. The literature raises a point of how users are more likely to provide data given an incentive, whether it be monetary gain or, for example, writing a negative review to make a point and receive compensation for the incurred inconvenience. It is far less likely that someone would leave a positive review for no gain or participate in crowdsourcing processes for free. The review example can provide a negative skew to data that does not represent the sincere belief of a crowd; hence it is crucial to ensure data collected also comes from those on the opposite end of the scale, who are less likely to provide data voluntarily.

AskIt!, the system proposed in the literature, works by using a prediction algorithm for each user to determine missing answers to ask, filling the data set. For questions that do not have a rich data set, a second algorithm then predicts which answers the user would be unlikely to answer to collect data that will have greater weighting. It is suggested that any prediction algorithm may be used for these tasks, though [Boim et al. (2012)](#_References) opted to employ the Collaborative Filtering (CF) algorithm in both cases as it has been shown to yield high-quality predictions.

Active engagement of the crowd has been vital among other papers, however passively collecting crowd-sourced data can alleviate the issues of finding crowd members to actively provide information, hence removing the need to provide an incentive to the crowd. [Basiri et al. (2017)](#_References) investigated using CCTV cameras and Ambient Intelligence (AmI) to investigate the improvement of tourist guidance systems without the crowd explicitly providing information. Cameras monitored the trajectories of pedestrians in a gallery to determine points of interest. By determining patterns in which piece of art someone goes to next, the system can make suggestions on where a new individual with the same starting point should go by assuming that they have the same taste as others who followed the determined pattern. Spatio-temporal mining techniques [(Cressie and Wikle 2015)](#_References) were identified as being particularly helpful tools in the identification of these patterns. The proposed idea is a new method of implementing a recommendation system without asking the visitors for their preferences, improving the experience by reducing invasiveness.

Anonymity does become an issue though, as highlighted in the paper, as people would likely prefer their location was not tracked if asked, so would not likely permit for this data to be stored. Storage and processing of this data without consent would infringe data protection laws, so must be anonymised. In their paper, [Basiri et al. (2017)](#_References) use K-anonymity [(Kalnis et al. 2007)](#_References) to remove the ID of the user and cloak the user’s exact location. K-anonymity is one of the preferred methods of anonymising tracking data and makes passive crowdsourcing of data possible.

### Analysis, Design and Development of Android Applications

Even a well-made, well thought through, application does not guarantee success, as several factors can result in low download counts and poor user retention. [Rollins (2011)](#_References) discusses the importance of marketing an application to succeed on the app store, outlining how easy it is for applications to vanish amongst the thousands of others available on the app store. The issue of having an application stand out has only become more of an issue since the literature’s publication, as more and more applications are added to the store every day. Less strict approval processes on the Google Play store have led to more significant growth and variety compared to Apple’s iOS App Store. While the book does not go into specific depth regarding guidelines, both companies have significantly changed their approval process and guidelines since the publication of this literature. While the book suggests several IDEs that would work for cross-platform development, such as LiveCode and Appcelerator, it opts to provide tutorials using Eclipse IDE with Java JDK and Android SDK. The chosen IDE and development kits have, however, severely lost popularity over the years, with Android developers switching to the official Android Studio IDE when it was released in 2014, meaning the usefulness of the book's tutorials has severely decreased.

While code and IDE tutorials may be out of date, the information regarding application analysis and design are still very much relevant, as these processes have not changed much over the years. [Rollins (2011)](#_References) highlights how a useful application must be simple to use and, most importantly, solve a problem. Discussion into the application’s desired audience and how the application should be presented and categorised on the Play Store provides useful insight into the post-development phase of the application; however, this information is potentially out of scope for the prototype application although can be carried forward to the future of the project. Similarly, [Rollins (2011)](#_References) goes into detail about how logos and ideas may need trademarked or copyrighted, and specific ideas could even be patented to protect the developer’s intellectual property.

[Hoisington (2015)](#_References) provides a more recent publication, focusing mainly on providing tutorials to application development using Android Studio, although earlier versions of the book pre-date this IDE and instead use Eclipse (similarly to the literature by [Rollins (2011)](#_References)). Like other sources, [Hoisington (2015)](#_References) clarifies that Android Studio is now the “standard” for developers further securing its place as the IDE of choice for this project. A brief discussion into the state of the App Store with statistics confirms Android’s lead in sales with over 1 million handsets activated per day and seeing use in 73% of the world’s handsets compared to iOS’s 12.7% (at the time of publishing). Most of the book’s contents consist of Android Studio and Java tutorials which will prove useful in providing a greater understanding of the development process and implementation of application features. Information provided about publishing the application post-development would prove useful in the future of the application beyond the scope of this project, although it may not be of particular use in the prototype development.

[Clifton (2013)](#_References) gives basics on Android user interface (UI) design, and core principals including standardised components ensuring that the application design supports a range of devices of different size and resolution. Apple’s XCode IDE provides standardised components, such as back button locations and menu bars to ensure all applications are consistent across the App Store maximising user experience through increased familiarity. Android’s IDE has less strict standardisation, allowing greater freedom, although it is still recommended to stick to simple layouts to prevent consumers from being confused by the layouts. Wireframing of application designs is explained, with provided examples to provide greater understanding and clarity. Some of the more advanced techniques in the book give tutorials and example code for implementation of a variety of UI features, including splash screens and loading indicators as well as complex views. The importance of user feedback is discussed, which will be essential information for providing a satisfactory user experience

### Similar Products

[Winkler (2013)](#_References) looks at the future of Waze; a crowdsourced satellite navigation application, and how it manages to stay profitable. [Winkler (2013)](#_References) expresses how a crowd-sourced application is “only as good as the data users share”, meaning that an application with an in-active user base will fail to provide usefulness and grow its audience proposing a concerning issue with regards to the success of the project in the future being able to maintain a user base. Profit wise, Waze makes money through advertisements displayed on the screen while stationary and by allowing businesses, such as Dunkin' Donuts, to advertise their location on the application's map for users to see. Allowing advertising to be integrated in this way offers an interesting, and untraditional approach, to advertising through mobile applications unique to those using maps.

### Carried over to Product and Future Recommendation

As suggested by [Amsterdamer and Milo (2015)](#_References), it would be possible to allow the crowd to vote on the quality of markers within the produced application. Such a feature may see implementation in the form of a rating system, in which users can reward points to markers considered to be of high quality or usefulness and subtract points from ones they believe are of no use. This implementation is somewhat like a thumbs up/thumbs down system as is seen on social content sharing websites such as YouTube and Reddit. Scoring of a marker could potentially see consideration when deciding what to show on the map, markers below a certain score threshold could be hidden or removed from the map, while high scoring markers could be made more evident to users – as the user is more likely to appreciate a marker that has been considered high quality.

While the methods proposed by [Amarilli et al. (2004)](#_References) were not intended for the type of data crowdsourced within this project, the ideas still provide an exciting insight into gathering data. While the proposed application will not directly ask the user questions, guidance may be regarding the data users are expected to input, ensuring the right question is presented to the user. By suggesting categories to the user, such as ‘Event’, ‘Restroom’ or ‘Landmark’, they will be able to source data that is more useful to other users improving the quality of data at the source and reducing the need for quality control after the data is collected. Similarly to how [Amarilli et al. (2004)](#_References) attempt to spot anomalous results, it could be possible to spot any ‘fake’ markers created maliciously through the use of machine learning and sentiment analysis. By spotting patterns in fake markers, future markers that follow these identified patterns may be flagged for review to ensure they are not malicious in intent.

While doing well to highlight the importance of the incentive aspect of crowdsourcing, the examples provided by [Boim et al. (2012)](#_References) are not systems similar to the application produced in this project. As a result, it does not provide a solution to incentivising users to contribute to the map without providing monetary gain which would not be possible to supply in this case. Although the AskIt! system does succeed in effectively matching questions to those likely to answer; it is not possible to implement this within the project as there is no variation in questions to be asked as the application only asks users to mark points of interest. Though, if a user only marked points of interest under a single category, the application could suggest they make use of other categories, which would relate to [Boim et al. (2012)](#_References)’s method of suggesting the option the individual is least likely to voluntary use.

Since crowdsourcing and locational data play a large part in both this project and the investigation by [Basiri et al. (2017)](#_References), the idea of identifying crowd preference based on trajectory could work similarly within the application. By checking places users have been, and where they go next, patterns could be identified, and points of interest suggested in a similar way to the gallery studied in the paper. However, this would require many users to be monitored within the same area to identify patterns in specific areas as well as the implementation of sophisticated artificial intelligence algorithms with the ability to anonymize the data provided, likely using the K-anonymity method suggested. Due to the complexity of this feature, it would be out of scope for the prototype application, although in the future of the project it could be implemented providing the application possessed a large enough crowd to provide sufficient training data.

The proposal of the importance of marketing and proper categorisation by [Rollins (2011)](#_References) does provide essential points for the future of the project post development, as does discussion into the presentation of the application on the app store including branding and logos, and the protection of logos and other intellectual property. Alternative IDEs for cross-platform development, such as LiveCode, also propose possibilities for the application’s production on iOS devices without the need to entirely re-develop the application in the future. The key points to consider while considering an application also provide a sound basis for discussion in the analysis, ensuring the provided application will be considered useful and usable and ensure that it can solve the discussed problem.

Examples and tutorials provided by [Hoisington (2015)](#_References) will be referenced throughout the development of the application to familiarise with the IDE and the coding processes associated with Android applications, ensuring that efficient code and good practice is present throughout. Similarly, ideas provided by [Clifton (2013)](#_References) will see use in both the design and analysis, particularly in designing a consistent user interface for the application as well as producing wireframes among other designs. Some of the code tutorials may also be used in the implementation of specific features that may see implementation within the application where the tutorials can be considered relevant to such features.

## Justification of Tools and Resources Required

### Android Studio IDE

Android Studio will be used over any alternative IDE as the “standard” tool for this task, as outlined in reviewed literature by [Hoisington (2015)](#_References). Its developer support and the large community of Android developers who have produced many learning resources making this the most accessible IDE for Android development. While an alternative program such as LiveCode or Appcelerator would allow for cross-platform development, as outlined by [Rollins (2011)](#_References), the project intends to produce an application solely for Android. So, while these may have been better choices in the future of the project if it were to be decided to produce an iOS version, for this project Android Studio was deemed the most sensible choice as the other IDEs lack a similar quantity of resources.

### Google Maps SDK for Android

Similarly, to justifications made for Android Studio, Google Maps SDK was chosen as the standard tool being developed by the same company as the Android operation system, as well being the most well-known mapping software, with users opting to use the Google Maps application over competitors with a 67% market share [(Panko 2018)](#_References). The Google Maps SDK/API also has a commanding market share of 83.18% seeing use on over 4 million websites to date [(Datanyze.com 2018)](#_References). The functionality required by the application, to show custom markers on a map, is present in this development kit.

### Firebase SDK

For database functionality, Firebase SDK will be used to allow the implementation of a Firebase Real-time Database. Firebase will be used due to its integration with Android Studio, allowing for more straightforward implementation within the project. Firebase provides a NoSQL real-time cloud database that stores data as JSON. Firebase also allows cross-platform integration, so if an iOS version of the application is considered for development, it would be able to share the same database instance and automatically share data in real time. When the device’s internet connection is lost, Firebase applications remain responsive due to persisting the data on the disk allowing them to remain responsive. Scalability is also a key factor, with Firebase allowing data to spread across several database instances within a single project, supporting the growth of the application in the future.

### Microsoft Office Suite

Microsoft Office has always been the standard software suite throughout academic studies and is available for free through the university. Due to having a vaster array of features and support, as well as being the most straightforward and familiar software, this software has been chosen without much consideration for an alternative.

### Adobe Creative Cloud Suite

The standard suite for image manipulation and illustration software, the Adobe suite was chosen for its familiarity as the most used in academic studies thus far. Offering a wide range of applications that host many advanced tools, the software selected is more than capable of meeting the requirements of this application. Alternatives, GIMP and Pixelmator, offer similar performance but have their own downsides making them less viable option. GIMP has a less user-friendly design and has less powerful tools than the other two applications. Pixelmator’s downside is that it is only available on OSX, and work would not be transferable to a windows computer making working on different machines less fluent.

Adobe XD, however, is quite a unique piece of software that does not have a particularly large selection of alternatives. The available alternatives often cost money except for inVision, which appears to offer similar performance. Adobe XD has been chosen over inVision as a result of its wide variety of free UI kits with specific kits tailored for Android, iOS and other officially supported layouts. XD also integrates better with other applications in the Adobe suite and keeps the choice of applications more consistent.

### StarUML

Originally, Microsoft Visio was considered for the creation of UML diagrams within this project, as a simple to use tool that integrates well with other Microsoft Office Applications, allowing diagrams to be pasted directly into a Word document from Visio and can later be edited within the Word document without needing to open Visio again. However, Visio is only available on specific university computers and not available for home use without a paid subscription. StarUML however, is an alternative that also available on university computers but can be used at home for free as an evaluation copy. StarUML has many built-in UML templates, as is used by many industry-leading companies including Apple, IBM, Amazon, Intel, and Oracle (staruml.io).

### GitKraken

While the official GitHub client saw consideration, previous projects have shown GitKraken to be superior for managing a GitHub project with greater automation, and an informative interface indicating the projects time scale. While this is not available on university computers, it will see use on personal ones. Where necessary the GitHub website or official client will be used on university machines where GitKraken is not available.

## Overview of the planned approach

The project development will use a tailored agile development approach. While requirements will be outlined before the development of the application, volunteer feedback will be collected throughout the development to ensure the implementation of these requirements in a way that is easily usable and effective. Embracing changing requirements as the project progresses according to the feedback provided on prototypes will be critical to the development process and will uncover any design oversights that would lead to poor usability.

Since unfamiliarity with technology will be an issue within this project, in the form of the Android Studio IDE and Google Maps SDK, agile methodology proves to be the best choice due it is programming-centric nature which allows the learning new-technologies in less time than is possible in alternative methodologies. Agile development alleviates some of the risk associated with having to learn these new technologies for the implementation of the product, such as the time constraint.

Due to the limited, and inflexible, schedule available for the implementation of this task, an agile methodology will prove advantageous as it allows a faster delivery than other methodologies. By frequently evaluating progress, schedule visibility can be maintained as a principal criterion, allowing application features to be prioritised where necessary to ensure the achievement of the best implementation within the timescale. Unnecessary work can be avoided to ensure that the critical features are delivered in the final product.

Rapid application development methodology was considered for its similar performance; however, this requires a more significant amount of time to be set aside to learn the new technologies that will be required. As a result, it was considered more efficient to choose an agile methodology so that the system can see further development in the time that would otherwise be set aside for learning these technologies.

## Justification of Analysis Deliverables

Requirements analysis [(Appendix D)](#_d)__IEEE) will provide information regarding the functional and non-functional requirements of the system in order to measure their feasibility and will be used to justify the requirements present in the final product.

The produced use case diagram and descriptions [(Appendix K)](#_K)_Use_Case) have been used to identify the main activities to be performed by the product to create a scope for the product. The activities produced in these use cases will be considered the core functionalities of the product and therefore the most vital. Features extending the functionality described in use cases will not be considered crucial to the project; therefore their possible absence will not result in the product being considered a failure.

## MoSCoW Analysis of Application Requirements

### Functional Requirements and Justification

**Must**

* Provide a functioning log-in and account creation interface
  + Necessary in the full application, an account will be mandatory in the application so that the crowd will be deterred from maliciously inputting ‘fake’ markers, which do not contribute to the map. Considered to be a primary use case [(Appendix K)](#_K)_Use_Case).
* Show a map containing nearby markers to the user
  + The application will locate the user and show the markers in their area, providing convenience over having to search for their location. This feature is one visible in many existing map products including Waze, as users may not know how to find their current location manually. The ability to view markers placed by other users is one of the primary application functionalities [(Appendix D, section 2.2)](#_d)__IEEE) and a primary use case scenario [(Appendix K)](#_K)_Use_Case). The map will allow data to be shared between users and allow a convenient interface for precisely marking locations.
* Allow users to add a new marker, allowing them to choose a category and add a description
  + A primary function of the application as discussed in the requirements specification [(Appendix D, section 2.2)](#_d)__IEEE) and primary use case scenario [(Appendix K)](#_K)_Use_Case). This feature is fundamental for the collection of data from the crowd.
* Allow users to remove markers that they have made themselves
  + Editing markers is one of the primary use cases [(Appendix K)](#_K)_Use_Case). However, a workaround to editing a marker directly may be to allow the user to delete and replace the marker. Deleting markers will be considered essential so that low-quality data may be deleted to improve the overall quality of the application – as [Winkler (2013)](#_References) the application is only as good as the data it provides.
* Allow user to change account details such as password
  + Changing passwords is an excellent way to ensure security and allow a user to protect their account if their password becomes compromised, as discussed in [(Appendix D, section 4.3)](#_d)__IEEE)
* Safeguard against potential errors in data entry.
  + To prevent database errors and application instability, the application will ensure all data is entered where necessary, and user input is to ensure no invalid characters are entered. This could also lead to the improvement of data quality by preventing submissions that do not meet a certain length.
* Store data securely within an online database.
  + While a local database would be suitable for the application prototype, the final product would require an online database to allow data to be shared [(Appendix D, section 3.2)](#_d)__IEEE). Such a feature makes a desirable quality to enhance the quality of the prototype. The database used should be secure to meet the security requirements outlined [(Appendix D, section 4.3)](#_d)__IEEE).

**Should**

* Allow for concurrency in the database.
  + While the application prototype will be able to function appropriately without concurrency, it will be necessary for the application to make use of such technologies if released; otherwise it may prove difficult for users to make simultaneous use of the application [(Appendix D, section 3.2)](#_d)__IEEE).
* Use password encryption techniques to ensure storage within the database is secure.
  + Not considered essential in the prototype due to the application not storing any delicate data is stored within the database. However, the feature is desirable to protect users who use the same password for other logins [(Appendix D, section 4.3)](#_d)__IEEE).
* Allow filtering of markers based on category and score.
  + Allowing users to filter by chosen attributes somewhat extends the voting feature discussed in the ‘could’ section, building upon the literature of [Amsterdamer and Milo (2015)](#_References) allowing for viewing content based on its quality as judged by the crowd. However, even without voting may provide significant convenience to users who are looking only for specific information such as markers that fall under a specified category. As such, this has been considered a main functionality of the product [(Appendix D, section 2.2)](#_d)__IEEE), though the level of filtering that the user may perform may vary.

**Could**

* Allow users to vote on markers placed by other users, which will impact the user’s overall score.
  + Allowing users to vote on markers is inspired by social content sharing websites, such as Reddit and YouTube, as well as from literature by [Amsterdamer and Milo (2015)](#_References) as a method to allow the crowd to control the quality of the content sourced.
* Ensure passwords meet a specified criterion to ensure a higher standard of security.
  + This feature would assist in protecting data; however, since data is not very sensitive, it is not considered a feature of significant importance [(Appendix D, section 4.3)](#_d)__IEEE).
* Suggest that users mark places of interest using their least used categories to provide a better balance across categories.
  + Inspired from literature by [Amsterdamer and Milo (2015)](#_References) and [Biom et al. (2012)](#_References) which outlines the importance of asking the least answered question to ensure that no question lacks sufficient responses. This feature has not been considered an essential function within the prototype, as the core functionality of the application would not face considerable impact in its absence.
* Display Adverts in the form of on-screen banners using Google Mobile Ads SDK.
  + Inspired by the many free applications on the Play Store that use this technology, as well as reviewed literature by [Winkler (2013),](#_References) who discusses how similar crowdsourcing application Waze stays profitable using sponsored locations and pop up adverts. Not necessary for the prototype, as no users would be present to see the adverts, but it could be implemented to demonstrate the potential to commercialise the product and gain user feedback on how adverts are displayed to see if it negatively affects the user experience.
* Allow users to report markers that may be offensive, inappropriate or suspicious to prevent misuse of the application.
  + It is essential that offensive, explicit content is not placed on map markers which could offend and disgust users, leading to a negative experience with the application and possibly lead to the reporting of the application on the Google Play Store. To prevent markers being abused to aid criminal activity, such as arranging meeting places for such activities to occur, it should be possible to flag suspicious markers to be reported and removed as a precaution. If the feature to allow images were to be implemented, as discussed previously in this requirement list, the concern over explicit images would make this feature even more necessary. Since a controlled volunteer group will be used for the prototype development, these issues are unlikely to occur, and this feature is more likely reserved until the possibility of a full release.

**Won’t**

* Provide recommendations on places based upon previously viewed or visited locations using artificial intelligence techniques.
  + A feature inspired by [Basiri et al. (2017)](#_References) and their investigation into making tourist guidance systems more intelligent using ambient AI. This feature is not intended to be implemented in the prototype application due to its substantial complexity and requirement for a large data set, which will not be available, from which to train the AI recommendation system. This feature would be more likely to be implemented once the application had gained a substantial user base and held a vast amount of marker data that could be used.
* Allow filtering by/following creating user or communities.
  + A concept that comes from the application idea that originally stemmed this project [(Appendix A)](#_Project_Terms_of), which would have allowed users to mark street art. The feature would have allowed users to filter by the user who placed the marker, so in the scenario where an artist marked their work on the map, it would be possible to filter by markers which they had placed to see only their artwork. Similarly, allowing users to follow each other would have allowed users to keep up with new works by their favourite artists. While this feature will be considered for the future of the project, it is not going to be present in the prototype application as it does not add much to the core functionality of the application and is not useful until a significant user base is established.
* Allow pictures of specific file size to be attached to markers to convey what is there visually.
  + Not considered a necessary feature and will likely be left to the future of the project. Inspiration for the implementation of this feature comes from the street art community to show artwork on a map, partially connected to the origins of the application idea [(Appendix A)](#_Project_Terms_of). Such a feature has received support from volunteers who provided feedback on the project, though one volunteer did highlight the potential for misuse of such a feature by which a malicious user may upload explicit content. As a result, this feature may become challenging to curate in future. The large file size of images would also cause complications with the database as the size requirement would be significantly increased. Cloud Firestore limits the maximum size of a single document to 1 MiB [(Firebase, 2019b)](#_References) so such an increase would not be supported by the database used and an alternative method of data storage would have to be found.

### Non-Functional Requirements

**Should**

* Provide a convenient, aesthetically pleasing and user-friendly interface.
  + This will be tested by gathering feedback from volunteers throughout the development phase [(Appendix D, section 4.4)](#_d)__IEEE).
* Be reliable, and bug-free and not encounter any crashes.
* Be maintainable, with well structured, commented, and documented code
  + Following good practice and coding standards as well as making use of JavaDoc commenting will allow for code to be understood with greater ease, allowing for simpler maintenance and expansion of the project.
* Be efficient in storage and querying of data.
* Provide efficient operation in order conserve battery life of the mobile device.
  + As mentioned in the requirement specification [(Appendix D, section 4.1)](#_d)__IEEE), it is important, particularly for a mobile device, to minimise power consumption where possible as to preserve the battery life of the device. An unnecessarily draining application may lose its appeal to users and result in the product being uninstalled.
* Be available to a broad audience, by ensuring the application works on a wide range of Android versions.
  + Application availability is one of the essential attributes to consider with mobile development, as there is a much larger variety of software versions to consider [(Appendix D, section 4.4)](#_d)__IEEE). To ensure the availability of the application to a broad audience, the project will be developed using Android 4.03 (API 15), which Android Studio suggests will provide 100% device coverage, and tested with versions 4.03, 5.01 and 9.0 [(Appendix D, section 2.4)](#_d)__IEEE).

## User Interface Designs

One of the critical differences in the UI of a mobile application and that of a desktop application is the simplicity; while desktop applications will have tens, maybe hundreds, of buttons on a single screen this is not possible with a mobile device. The reason for this is due to the significantly smaller form factor and less precise input methods when compared to a mouse and keyboard. Such an issue is not something that has been an issue in any previous projects due to being developed for desktop computers; however, will need to be considered when designing the interfaces for this project.

When using a minimal number of buttons on the screen at any one time, it is possible to make the buttons larger and more decorated to make them easier to recognise and click, especially while on the move. Standardised icons will be used to make the purpose of buttons prominent without using words; these icons will be from the Material Design UI kit and Android Studio, which contain standardised, recognisable icons and layouts for Android development. Use of standardised icons promotes greater synergy between this application and other applications on the user’s device.

A consistent colour scheme is essential for branding will be used throughout the application to create an identity for the application using its colours ([Odgis 2015](#_References)). Colour schemes such as this have been observed among many other applications (Waze uses light blue, YouTube uses red, Snapchat uses yellow). This standardised house style will increase the aesthetics of the UI and improve the recognisability of the application.

The colour scheme to be used can be seen in [Appendix E](#_e)_User_Interface). Consideration was taken for the usability of these colours for user’s who are colour blind. Colour filters have been applied to the base colour scheme to simulate how they would appear to people who are colour blind; this can be seen in [Appendix F](#_f)_Colour_Palette). From these filtered views, it was observed that the colours used are significantly different and identifiable in many cases except for users with Achromatopsia. Because of these findings, care will be taken to ensure that the colours ‘Toolbox’ and ‘Fuzzy Wuzzy’ are not placed too close together due to being more difficult to differentiate for these users.

# Synthesis

## Class Diagram

Throughout the development process, the class diagram has changed several times as an understanding of Android development was formed. Initial designs of the class diagram contained classes for the database and classes to store marker information and information about the logged in user. While this initial design provided a good idea of the general classes needed, and structure of the user interface classes, some classes were considered unnecessary throughout development and were removed from the diagram or repurposed. The finalised diagram and class structure can be seen in [Appendix H](#_H)_Class_Diagram). Rather than a database class acting a medium between the current activity and the cloud database; it proved more efficient to access the cloud database from the activity class. Accessing the database in this way did not require much code as the API allows relatively simple database access – as a result, the database class was removed. Since user details can be accessed easily from the Firebase authentification method in the API, the user class was repurposed for creating an object for insertion of a new user into the Firestore database.

Similarly, rather than using the marker class to store details for each marker retrieved from the database, it was more efficient to add each marker to the map directly. By not storing each marker as a separate object instance, less memory is used by the application at run time which may prove beneficial on older phones with less memory. Rather than removing the marker class, it was repurposed for creating a marker object to insert into the database when the user creates a new marker.

Due to the initial unfamiliarity of some components used, the initial class diagram did not contain classes that would prove necessary in the final product, such as the SettingsFragment and FilterFragment classes. These classes are necessary to populate the SettingsActivity and FilterActivity screens with preference switches dynamically using values from the strings.xml file.

Generally, apart from the changes discussed, the finalised diagram follows the same structure as the original having been followed as closely as possible to ensure the final application remained within the general scope and size that was initially expected.

As a deliverable, this diagram was considered essential to convey the design and structure of the product and to allow a greater understanding of the application. In the future of the product, the class diagram will be highly beneficial for maintenance and further development allowing the code to be understood with greater ease.

## Database Specification and Design

Since the application relies heavily on the use of the cloud database, it is crucial that the database be efficient and straightforward. Additionally, to allow further development of the database, it is essential that the initial structure is sensible as not to compromise its further usability. The two database options recommended and supported by Google and Android are Cloud Firestore and the Firebase Real-time Database – these were considered above third-party choices for their ease-of-use and simple integration as well as a vast selection of learning materials. Cloud Firestore was chosen over Real-time Database due to its simplicity, more powerful queries and automatic scaling – allowing for greater futureproofing of the application especially when considering a full release of the product.

The entity relationship diagram which represents the structure of the document-based database is available at [Appendix L](#_L)_Entity_Relationship). It should be noted that that Cloud Firestore’s NoSQL document-based structure does not make use of foreign keys hence the exclusion of this key from the diagram. However, the relationship shown in the diagram highlights that a given location will have exactly one user who placed the marker, while a single user may have created zero to many locations. Example data for this structure can be viewed in the screenshots in [Appendix I](#_I)_Cloud_Firestore).

Initially, it was considered that the user table would be used to store the user’s password, preferably hashed using a secure encryption method; however, it was made apparent that the recommended method of storage for user login information was using Firebase Authentication [(Firebase, 2019)](#_References). This led to the removal of the password field from the user table, and instead, the Firebase Authentication manager is used to store user details. This proves to be beneficial for several reasons, as it provides more convenient verification methods through the API allowing for more efficient and secure code. Also, while the current application only makes use of the email and password sign in provider, it is possible to allow users to sign in using a variety of existing accounts they may have, including their Google, Facebook and Twitter accounts. Allowing users to sign in with existing accounts could prove beneficial in the future expansion of the project by providing a more convenient and connected user experience as well as potentially allowing more significant implementation of social features. Firebase stores passwords securely using encryption methods (Firebase Open Source, 2019) and hides the hashed values from both users and developers so that even with the correct decryption method it would not be possible for someone to gain access to another user’s password.

## Product Code

Throughout the development process, when faced with various choices on how to implement a feature the Google recommended solutions were chosen where possible to provide a product that makes use of the best practice. One example of this is the user login, which could have been manually coded to extract a hashed value from the database and decrypt this value on the device, but instead is handled by API as discussed in the previous section. To stick to good coding practice, Java documentation has been used throughout the project to provide clarity to any methods as well as their parameters and returns.

Creation of user interfaces was initially expected to be done using java classes, as has been practised throughout academic studies and other projects, however Android Studio makes use of XML files for user interface construction. The use of XML presented a requirement for an unexpected new language, which had never seen use throughout studies thus far. While the drag-and-drop design view provided a simple solution, it became necessary to edit code in this unfamiliar language directly. As a result, it was necessary to do some research into the different methods layout types available in XML, especially in the context of Android development. Luckily, the similarities between XML and HTML, which was extensively covered in previous academic work, are relatively significant and lead to the language being picked up quickly. Other than creating the UI, XML files were also used for the storage of resources, such as values for colours frequently used strings and application styles that were used for reference in other code, as it is considered poor-practice to hard code such values.

Overall, the prototype contains fourteen java classes, of which ten are activities responsible for displaying the user interface and handling user actions. Of the other four, two are used for populating user preferences, and two are used as objects for insertion into the database. Fourteen XML files are present, responsible mostly for the design of user interface and storage of strings and arrays. Four JSON files are used for the styling of the map, including a ‘dark mode’ style.

One of the significant features that has not been implemented is the ability for users to vote on markers, due to the unforeseen complexity of recording which users had voted on which posts. It was considered that a solution in which the only marker stores the score, but no information is stored about which specific user voted on the marker, was flawed. The reasoning behind this is that a user would be able to upvote a marker, click off the marker, click back on the marker, then upvote it again - thus falsely inflating the score for the marker. Likewise, it would allow a single user to produce infinite down votes on a marker maliciously, which would jeopardise the entire purpose of the feature – hence it was decided that the feature was better omitted than poorly implemented. The workaround that was conceived was to store an array for upvoted markers and an array for downvoted markers in the user table. However, this would produce problems in situations where markers were deleted as the array would now point to a null value. Significant increase in the number of queries required to implement such a feature lead to the implementation being put on hold until further development of the project, as it was considered the development time could be better spent implementing more pressing features.

Due to the deprecation of the Places SDK for Android part way through the project development, the autocomplete feature has been implemented in a way that, while functional, does not quite meet the quality that was initially hoped for. Originally, it was hoped that the autocomplete would provide location suggestions dynamically based upon their relevance as determined by their proximity to the user. However, the tutorials followed to implement such a feature no longer produce functional code due to the deprecation of the SDK – and no tutorials had been published at the time of implementation that would allow the same functionality with the succeeding SDK. A workaround was found by reading suggestion values from a text file using a buffered reader, as was taught in university studies, and populating the autocomplete suggestions with values from this list. The result was satisfactory, and for the prototype conveys the functionality well. This method may see re-consideration in the future, however, when more information is available regarding the implementation of this feature using the replacement Places SDK.

## Testing

### Test Plan and Strategy

#### Scope

The testing plan aims to ensure the robustness of the application when subject to user interaction, to prevent possible crashes that may be caused by unexpected interaction. Interfaces will be tested to ensure that all buttons function correctly and that at no point will the application crash upon clicking any interactable item on the screen. All buttons should perform their intended action and provide responsive feedback to the user using button animations and progress bars to indicate an action is taking place. All inputs must also be thoroughly tested to ensure that there are no inputs that will result in the application crashing and that all invalid input attempts will inform the user what they must change to make their input valid.

The performance will be tested to measure the use of computational resources by the application at run time such to ensure that the application makes efficient use of available resources to improve performance on older, less powerful, devices and reduce the consumption of battery power on all devices. This testing is especially crucial due to the mobile nature of the application, as unnecessarily draining the battery of the host device would deter people from continuing to use the application due to the inconvenience it causes to users. Ensuring the responsiveness of the application is also essential, as an unresponsive application may lead to unsatisfactory user experience and increase the likeliness of the application crashing. Internet bandwidth can also be monitored to measure the impact of the application on mobile data usage, as excessive data transferring will prove costly to users who have limited mobile bandwidth.

Testing will be carried out using the Android Studio Android Emulator for devices using Android 9.0, the latest version of the Android operating system released on August 6th, 2018 [(Android, 2019)](#_References) as well as for version 4.0.3 (API 15), the version used for development. A physical Android device will be used to test the usability of the device with participants, as well as allowing testing to be carried out on a native Android device. The device used is a Samsung Galaxy S4 model number GT-19505 running Android version 5.0.1.

#### Test Approach

To test validation of user inputs, automated testing will be carried out using JUnit test classes to ensure that all valid inputs are correctly accepted, and all invalid inputs are correctly denied – and identify the reasoning for any issues that arise. Automated testing can quickly and easily replay pre-defined tests and compare the results to expected outcomes. Since testing will occur continuously throughout the development cycle, automating testing will save significant time when re-testing classes after changes have been made to code after a sprint, ensuring no errors are created when editing code. Using such testing also adheres to the modern professional practice of larger companies and will aid the maintenance and future development of the application. Values used for input validation will include null and empty values, to ensure blank data cannot be entered into the database, values each side of the upper and lower boundaries will be used to test input length is correctly limited. Non-traditional and problematic values will be used to observe how the system handles such inputs, including the use of special characters, quotation marks that may break strings, and emoji which are a data type not traditionally considered outside of mobile development.

Integration testing will be used to ensure that each component of the application is functional before connecting it to another component, to ensure that errors are not carried forward from one class into another making it more challenging to identify the source of an error.

Performance testing tools will be used within the Android Studio IDE to measure the applications use on various resources. CPU profiler will measure processor activity to gauge the optimisation of code computation; memory profiler allows inspection of the Java heap and memory allocations; network profiler allows network traffic to be inspected to see which processes require internet access and the data usage of these processes; energy profiler monitors various components and sensors to identify unnecessary use of energy.

White box testing will be performed frequently throughout the development after each section of code is added to ensure its functionality, so errors can be spotted early, and often, to prevent them causing issues later down the line when they are harder to spot and resolve.

Black box testing will be performed less frequently by a selection of volunteers at their convenience. Volunteers will have little to no knowledge of the programming or implementation of the application, so trial and error may be present and will highlight any usability issues, such as difficulty navigating the application or vagueness and lack of guidance on how features work, as well as uncovering potential functional issues. Usability testing will be performed by volunteers, who will be asked to carry out tasks without guidance to see if they can navigate the application independently, such tasks may include changing their password, adding a marker to the map and deleting a marker which they have previously placed.

Validation of code will be carried out where possible using online validators, such as those offered by W3 Schools, to ensure that no errors are overlooked throughout the development processes.

### Test Results

Testing was performed on all activity classes for both the user interface and Java code using both manual testing, performed both personally and through the assistance of volunteers, as well as using JUnit testing on activities containing input validation. The complete results of these tests can be found in [Appendix J](#_J)_Testing_Plan) – throughout this section reference will be made to the tests and results in the appendix.

Early in the development, it was decided that the Samsung mobile phone would become the primary testing device, as the Android Studio emulator faced many stability issues which caused the host computer to freeze frequently and the application to crash despite the code being correct. Using an external Android native device also provided testing benefits, as performance statistics could be measured while testing and computational resources were limited to those available on the phone – where the emulator would take resources from the far more computationally-powerful host computer.

Throughout development, one of the key challenges was the use of new technologies, libraries and application programming interfaces (API) which resulted in some compatibility issues early on. Test LA-1 was the crucial first example of this, in which the Firebase User Authentication library produced errors in the ‘build.gradle’ file due to various other libraries also attempting to access the library ‘com.google.android.gms:play-services-base’ at the same time, resulting in conflicts. Similarly, test LA-2 produced similar issues because of mismatched and missing libraries which caused the application to crash upon attempting to open the map activity. Both issues were solved by altering the version numbers of implemented libraries in the ‘build.gradle’ file as well as adding missing libraries.

Unfamiliarity with the setup of Google API services led to issues, as can be seen in tests LA-1.2, MAC-1 and MAC-1.2. Failure of test MAC-1 was a result of the Google API key being incorrect and being unable to authenticate the application services. Failures LA-1.2 and MAC-1.2 were the results of services not being activated on the Firebase console settings. Having gained a greater understanding of the Google Play services integration, these types of issue are ones that are less likely to cause an issue in the future.

One major issue caused by using external libraries was the implementation of the autocomplete feature in the search text box; this was due to the deprecation of the Google Places SDK for Android midway through the development process on the 29th of January 2019. Initially, this SDK was to be used to predict entered location names based upon the user’s current location to enable greater relevance of search results, however, due to the service’s deactivation, the code provided in learning materials for such a feature became obsolete and non-functional – as can be seen in test MAS-1. Attempting to use the newer Places API in conjunction with the Google-powered search bar within the app also produced errors, as the search bar would continuously encounter an error for which a fix was unable to be found as can be seen in test MAS-1.2. As a workaround for this issue, a file containing approximately 23,000 major global cities was located online and converted into a text document containing only the city names. After a failed attempt to read this file using a file reader (test MAS-1.3), a solution was eventually found in which a buffered reader is used to read the contents of the text file into an array of strings from which the text view locates its suggestion results.

Custom markers are a design aspect that had to be reconsidered during testing, as the custom markers that were initially intended for use in the project produced visual errors when added to the map in the form of unsightly black boxes around the image (test MAMC-1). Initially, it was believed this was due to the nature of bitmap images not supporting alpha channels in most image editors, but eventually, a way to produce alpha-supported bitmap images was found in Pixelmator. However, even with the new images, the issue continued to occur as it appears that the Google Maps API does not support alpha channels on the marker images. Instead, it was discovered that it was possible to change the default marker type to a range of different colours, although somewhat limited, to differentiate categories, which resulted in a more straightforward implementation of this feature. Since marker design is tied to the API, not custom assets, markers will not require a redesign in the future for modernisation of the GUI, making the product more maintainable in the future.

While the un-conventional no-SQL database structure also posed unfamiliarity, its ease of use produced very little in the way of errors. The only tests in which the database is connected to failure were CAA-1, NMA-2 and CAAT-13. However, the first two issues were due to incorrectly inserting data into the database through the application meaning the data was stored incorrectly. The third issue, CAAT-13, was caused by Firestore database refusing to accept a password less than six characters in length, which at the time was a requirement that was not known but quickly rectified through the addition of extended input validation. The application performance test also indicated that the network usage incurred because of querying the cloud database was minimal, with typical peaks of a maximum of 2MB/s or less. Observation of performance metrics showed that the most resource consuming action performed using the database was retrieving all markers from the database and placing them on the map, although this had a negligible impact on the performance of the test device.

Thorough use of automated validation testing highlighted many mistakes in the logic of the code and revealed oversights in the initial implementation of validation methods. Primarily, failures were the result of the inability for validation methods to detect null inputs (as seen in tests LAT-3, CAAT-20) which resulted in several failures on all forms that accepted user input. Automated testing aided in uncovering of several logical issues, in which usernames and passwords that were not within the allowed length boundaries would be considered valid due to missing return statements in several places; examples of tests that failed due to this issue include CAAT-29 and CPAT-7. Despite the convenience of automated testing, some known compatibility issues occurred when combining JUnit testing with the ‘TextUtils’ library. This compatibility issue caused the production of false negatives when checking if an input string was empty, as can be seen in test NMAT-5, NMAT-7 and NMAT-9.

Performance evaluation of the application concluded that the most computationally intensive aspect of the program, and therefore the one that will have the most significant impact on power consumption, is the map component. Due to a large amount of data required, especially on satellite mode, this component incurred the highest network usage at the peak of 4.8MB/s when loading a location on the map. Due to the graphical nature of the map, it also proved to be the most strenuous activity on the CPU and memory as it rendered the map, with CPU usage peaking at 64.9% and peak memory usage at 202.4MB (approximately 10% of the test device’s memory). No way was found to reduce the polling rate of the GPS location, as was previously discussed in an earlier chapter as this is known to be severely draining on battery life. However, it was not possible to measure precisely how much effect this had on the battery life of the test device, as the energy profiler is only available on devices that use Android version 8.0 (API level 26) and higher, while the test device uses Android version 5.0.1 (API level 21). Overall, no performance issues were highlighted by the profiler and all readings were within a reasonable limit, with all volunteers stating that the application performance was adequate and within their expectations.

Usability testing carried out by volunteers was done at a time and place of their convenience and consisted of an introduction to the prototype application and a hands-on test of the prototype. To test usability, volunteers were asked to perform tasks without guidance to check if the design of the application was straightforward and intuitive enough for them to perform tasks without guidance. Informal discussions took place to discuss design choices and volunteers were given to opportunity to provide constructive criticism on the application. Volunteers provided generally positive feedback although it was made clear that there needed to be more guidance given on how to add a marker to the map as without being explicitly told, users were not aware on how to perform this action. Because of this feedback, a message was added upon opening the map reading “press and hold to add new marker”, which volunteers agreed as a favourable way to provide guidance and improved clarity on the feature. When given other tasks to perform, all users were able to carry out their task independently, which indicated good overall usability and clarity.

Unfortunately, not all errors uncovered by testing were able to be rectified, and one specific error is still known to exist within the system. The error in question is an issue with the validation of email inputs, as can be seen in tests LAT-9 and CAAT-11. Validation of emails within the application is performed using the JavaMail API. However, it has been made apparent that emails that follow a structure similar or identical to “foo@bar“will be considered valid despite missing a top level domain (TLD). This issue is unable to be fixed as the API code cannot be altered. However, if the API is updated to fix this issue, it may eventually be solved. One method that could be used to prevent user’s exploiting this error to use fake email addresses would be to prevent access from the application until the user verifies their email address using the verification email which is sent upon account creation. This feature will not be added to the prototype; however, as it is considered unnecessary and would inconvenience users if implemented.

# Evaluation and Conclusions

## Evaluation of Product

### Project Aims and Objectives

The project aims, as outlined in the terms of reference [(Appendix A)](#_Project_Terms_of), have been successfully fulfilled.

Much was learned about the design an implementation of Android mobile applications throughout the process of development. Unfamiliar coding practices, such as referring to external resources in code rather than hard-coding, were adhered to as was recommended in official tutorials. Significant use of recommended technologies is present throughout the project, making use of the Maps SDK for Android, as well as a Cloud Firestore database and Firestore Authentication manager.

As intended, the product built can collect and display crowdsourced data in a way that volunteers to this project have agreed is simple, intuitive, and useful which testing suggests contains minimal bugs or performance issues [(Appendix J)](#_J)_Testing_Plan).

### Comparison of project performance against functional requirements

The following requirements are taken from the MoSCoW analysis available in section 2.5 of this report. However, requirements listed under the ‘won’t’ have not been included as their inclusion in the project has been previously omitted.

#### Requirement: MUST Provide a functioning log-in and account creation interface

This objective has been met successfully using Firebase Authentication methods to create accounts using an email address and password and store these values securely in the online database. Once an account is created, the user can log in on any device with the application installed – meaning they will be able to transfer this account to a new phone where necessary. Use of validation ensures the validity of email addresses before attempting account creation. Upon account creation, a validation email is sent to the email provided to ensure that the email address belongs to the user using it within the application. In this sense, the implemented functionality exceeds the initial requirement. Results of testing show that this functionality works as intended, except a minor issue allowing some invalid emails to make it through validation [(Appendix J)](#_J)_Testing_Plan).

#### Requirement: MUST show a map containing locations near the user’s current location

Upon opening the map activity, the user’s current location will be centralised on the screen showing nearby markers. In that sense, this requirement has is met. However, markers that appear on the map are not limited to those nearby to the user – all existing markers are presented on the map. By placing all markers on the map, users can search the entire map for markers, not just nearby areas.

#### Requirement: MUST Allow users to add a new marker, allowing them to choose a category and add a description

Holding down on the map will create a marker which is able to be added to the map by the user. The add marker activity allows users to enter a title and description for their marker as well as choosing from nine pre-determined categories. Validation requires that neither the title or category are left blank, which safeguards against empty values being entered into the database. Upon marker creation, the entered values are stored in the location table alongside the username of the user and the geopoint coordinates of the marker. When viewing a marker, these details are presented to the user. Testing had ensured validation and marker creation work as intended [(Appendix J)](#_J)_Testing_Plan). This requirement is therefore considered to be met.

#### Requirement: MUST Allow users to remove markers that they have made themselves

When viewing a marker on the account used to place the marker, a delete marker button will be visible beneath the marker details. When viewing a marker placed by another user, this button will not appear preventing users from deleting each other's markers. The success of testing this functionality [(Appendix J)](#_J)_Testing_Plan) means this requirement is met.

#### Requirement: MUST Allow user to change account details such as password

Through the settings activity, it is possible for the user to change their password providing they can provide their current password for security purposes. Testing of this functionality has shown that validation correctly prevents users from entering invalid passwords and does not allow the user to change their new password to their current password. It has also been tested to ensure that the password is successfully changed [(Appendix J)](#_J)_Testing_Plan). It is also possible for a user to change their password by using the forgot password functionality to send a password reset email to their email address from the login screen. It is not possible for users to change their email or username, as changing email addresses may cause complication with login and changing username would require every change marker to be edited. For simplicity, it was decided these values would be fixed. In some ways, it is considered this requirement has been exceeded for the extra security and ability to reset a password via email – however, the ability to change other details may have been desirable to some.

#### Requirement: MUST Safeguard against potential errors in data entry.

JUnit testing has been carried out on all classes that allow for user input, to ensure that nothing the user enters will cause the application to encounter an error. This testing has been generally successful except for email addresses missing a top-level domain being able to make it through validation – however, the issue lies with the external API used for email validation [(Appendix J)](#_J)_Testing_Plan). Since this exception will not cause any errors with the application functionality, this requirement is mostly satisfied. If an update to the JavaMail API removes this issue, the requirement will become fully satisfied.

#### Requirement: MUST Store data securely within an online database.

The primary purpose of the product is to allow users to share data with each other, which would not be possible without an online database. Since data is stored securely within databases recommended and hosted by Google and is only accessed through recommended methods using the official API, it can be assumed all data transfers will be secure. As mentioned in the requirement specification [(Appendix D)](#_d)__IEEE), testing was done using separate devices which were all able to access the database as intended; hence this requirement is satisfied.

#### Requirement: SHOULD Allow for concurrency in the database.

While a cloud database allows users to share data across various devices, without database concurrency, they would not be able to do so while simultaneously using the application as the database access would be exclusive to the user who is currently connected [(Docs.oracle.com, 2019)](#_References). Therefore, while it is acceptable for the prototype not to meet this requirement, it is vital that a release version of the application must have this capability; otherwise, the entire project could be considered a failure. The databases implemented, Cloud Firestore and Firebase Authentication, allow concurrent access for up to 1,000,000 users per database and possesses a maximum write limit of 10,000 writes per second (10 MiB per second) [(Firebase, 2019b)](#_References). These usage limits far exceed the requirements of the product, and therefore this requirement has been exceeded.

#### Requirement: SHOULD Use password encryption techniques to ensure storage within the database is secure.

As has been mentioned in previous sections, passwords are hashed when entered into the Firebase Authentication database. Encryption is carried out using a modified version of scrypt [(Firebase Open Source, 2019).](#_References) The requirement is therefore satisfied.

#### Requirement: SHOULD Allow filtering of markers based on category and score.

Filtering of categories is possible through the filter activity; the user can choose to filter using any combination of the nine pre-determined categories. Testing of this functionality has shown the feature now works as intended [(Appendix J)](#_J)_Testing_Plan). However, since the feature to allow users to vote on markers has not been implemented, there is no way to filter markers by score. Therefore, this requirement has only been somewhat met.

#### Requirement: COULD Allow users to vote on markers placed by other users, which will impact the user’s overall score.

This requirement was not met, as has been discussed in the synthesis chapter, due to the complexity it would add to the database and view marker activity. Without storing all markers that a user has voted on in an array and checking this array each time a marker is viewed, it would be possible for a user to vote on a marker an infinite amount of times. However, implementing the feature in this way would be very time consuming and was put on hold as a result as it was considered a better use of time to work on features that had a more significant effect on core functionality.

#### Requirement: COULD Ensure passwords meet a specified criterion to ensure a higher standard of security.

It was uncovered during the testing process that Firebase Authentication requires that passwords are a minimum of six characters long [(Appendix J, test CAAT-13)](#_J)_Testing_Plan). Validation was implemented to ensure passwords are therefore between 6 and 20 characters long. While stricter validation requirements could be implemented to ensure inputs contain mixed case or special characters, this would inconvenience users hoping to use an existing password that does not satisfy these requirements. Since stored data is not sensitive, it did not seem worth inconveniencing the user for this reason. Therefore, this requirement has only somewhat been satisfied.

#### Requirement: COULD Suggest that users mark places of interest using their least used categories to provide a better balance across categories.

This requirement has not been met, as implementation was considered unnecessary for the prototype and may prove challenging. Since not enough data will be present, it will not be possible to accurately suggest categories that significantly lack data – as all categories would have very little data.

#### Requirement: COULD Display Adverts in the form of on-screen banners using Google Mobile Ads SDK.

This requirement was not met; however, user feedback was gained on visual a mock-up [(Appendix M)](#_M)_Mock_Advert) and potential ideas for using advertisements within the applications. As can be seen in the mock-up, the banner would not cover the buttons added to the map as to not hinder usability and does not cover the Google logo, which would breach the Google Maps Platform terms of service [(Google Cloud, 2019).](#_References) Two users expressed that they would not mind such adverts, while the other four expressed such banner-based adverts negatively impacted the application experience. In the analysis section, an alternative method of advertising was considered in the form of sponsored markers, inspired by the satellite navigation application, Waze. The idea of sponsored markers was explored with volunteers in discussion to which all six volunteers agreed that this method of advertising would provide a less invasive form of commercialisation for the application. Neither method was implemented as the product is, for now, just a prototype and therefore would not be able to generate any income from displayed advertisements. While banner adverts would be able to be generated automatically by the Google AdMob [(Admob.google.com, 2019)](#_References), sponsors would have to be located manually for the implementation of paid markers.

#### Requirement: COULD Allow users to report markers that may be offensive, inappropriate or suspicious to prevent misuse of the application.

This requirement has not been met, as the prototype application will only be used by a group of trusted volunteers who are unlikely to produce any data that could be considered inappropriate. However, this requirement is likely to be implemented within the future of the product before a possible release can be considered as it is important that an effort is made to prevent such content from being contained within the application.

### Comparison of project performance against non-functional requirements

#### Requirement: SHOULD Provide a usable, aesthetically pleasing and user-friendly interface.

Usability testing and volunteer feedback imply the application provides a user-friendly experience, with test users managing to complete tasks with ease. The one usability issue that was encountered was the lack of guidance on placing markers, which proved an issue with all volunteers. This issue has now been rectified by adding an additional informative message to the map activity, which appears upon starting the activity, informing the user on how to add a marker to the map. Volunteers were questioned as to whether they believed the application satisfied this requirement, all six considered the requirement had been met.

#### Requirement: SHOULD Be reliable, and bug-free and not encounter any crashes.

The results of testing [(Appendix J)](#_J)_Testing_Plan) show that the application only has one known-bug with email validation, as has been mentioned in section 3.4.2 of this report. However, this bug does not result in the instability of the application. Therefore, this requirement can be considered mostly satisfied.

#### Requirement: SHOULD Be maintainable, with well structured, commented, and documented code

JavaDoc commenting has been used throughout the code to improve the clarity of methods and the parameters they use, along with regular comments describing various sections of the code that may be considered complicated. Validation of XML and JSON files indicates that no errors have been found in the non-java code [(Appendix J)](#_J)_Testing_Plan). While every effort has been made to produce maintainable code, this criterion is difficult to judge until maintenance is carried out in the future of the project. Since it is not possible to completely confirm the satisfaction of this requirement, It will only be considered somewhat satisfied.

#### Requirement: SHOULD Be efficient in storage and querying of data.

As can be seen in the entity relationship diagram produced for the Cloud Firestore database [(Appendix L)](#_L)_Entity_Relationship), the produced database stores only the minimal amount of required data. Database connections are only established when necessary for queries and closed once the containing method is complete, as to reduce the number of concurrent connections at any single time. Performance testing [(Appendix J)](#_J)_Testing_Plan) did not highlight any abnormal metrics that may have suggested the querying of data was inefficient in any way. The requirement is considered to be satisfied.

#### Requirement: SHOULD Provide efficient operation in order conserve battery life of the mobile device.

Performance testing [(Appendix J)](#_J)_Testing_Plan) did not indicate the product causes any unexpected performance issues. It was deduced that the component that consumed the most power was the map. This is due to the map requiring a large amount of data, primarily when used in the satellite view, and being the most graphically intensive component in the entire application. Powering the display is one of the significant contributions to battery drain in mobile devices, and research has suggested one way to combat this issue to provide a dark mode in applications [(xda-developers, 2019)](#_References). Taking this research into account, a dark mode was added to the application to the application, changing the colour of the map component to elongate potential battery life. The effect of this feature on battery life is not able to be measured, as energy consumption statistics are not available on the test device. Attempts to reduce the polling rate of the GPS were unsuccessful as no method could be found to perform such an action. Overall, this requirement is satisfied.

#### Requirement: SHOULD Be available to a broad audience, by ensuring the application works on a wide range of Android versions.

As mentioned in the requirement specification [(Appendix D, section 2.4)](#_d)__IEEE), the Android version used for development is version 4.0.3, which Android Studio suggests will provide coverage to approximately 100% of Android phones in use. Testing [(Appendix J)](#_J)_Testing_Plan) was performed across versions 4.0.3, 5.0.1 and 9.0 and it was observed that the application performed identically on all versions; the requirement is therefore considered to be met.

## Evaluation of Project Process

The tailored agile methodology used throughout the development process proved to support the development of the application, allowing familiarisation with the new technologies to be implemented into the development process, rather than learned as a prerequisite to the application development. Carrying out testing frequently also allowed any errors in the code to spotted early and often, which helped mitigate potential risks further down the line. Generally, this method of development allowed the project plan to be strictly followed throughout most of the project. However, complications arose towards the end of the project when commitments to other projects and assignments meant falling behind on the planned schedule, causing some work to become more rushed than would have been preferable.

Android Studio and the official tutorials, as well as reviewed literature by [Hoisington (2015)](#_References) and [Clifton (2013)](#_References), provided insightful information into how to get started with the development of the project, including setting up the IDE and required libraries. However, the tutorials offered by reviewed literature did not offer much information on the map component, which is central to the application so were used less in the later parts of the development process. For many of the issues faced because of testing failures, various forums were used to browse solutions that had been submitted to users who faced similar, and in sometimes identical, problems, as tutorial literature often lacked such information. In some respects, the problems faced throughout development and testing because of the lack of knowledge on the new technologies required significantly improved the understanding of these technologies compared to if everything had worked the first time. Having initially knew nothing about the development of mobile applications, confidence and understanding has been built significantly to the point where creating another product for the same platform would be considered far less challenging and intimidating than was initially the case. While previously honed skills with Java were beneficial through development, the way in which the language is used in the context of Android development shows significant differences to what has been observed in other projects. For example, tasks such as displaying messages to the user and creating new UIs use very different code compared to development for Windows. Prior to this project, XML was a language that had gone untouched, but its use in this project for the development of user interfaces has greatly improved knowledge of this language. Skills gained in HTML through academic studies were of great use when editing XML files, as the two languages have a lot in common in terms of code structure and styling.

In terms of what would be done differently, if the project were to be done again, a more significant effort would be made to increase the number of volunteers participating in the study by widening the search for participants. This would allow for an increase in the amount of user testing that could be performed throughout the project and widen the range of opinions offered. One issue faced by using volunteers who are friends and/or fellow students is the potential for bias opinions to lead to legitimate criticism not being provided from the test group.

Chosen tools were overall useful; however, in some cases, alternatives were used. For the design of some graphical elements, for example, Pixelmator was used as an alternative to the Adobe Creative Cloud suite due to its availability at the time of use – as the Adobe suite is only available through university computers. GitKraken had initially been planned for use in version control management; however, the version control system built into the Android Studio IDE proved to be more convenient at performing this task and replaced GitKraken.

The database used in the project was initially one of the more intimidating requirements, as previous projects have all used non-concurrent local databases – this would be a first experience using an online database with Java. However, the chosen tool proved to be very beneficial, as many of the complexities that require lengthy code in local databases are carried out in the back end of the Firestore API – such as intricacies of complex SQL queries. As a result, the switch to using an online database was far smoother than expected and has dramatically increased confidence with the use of cloud storage for future projects where this feature may prove beneficial.

## Conclusion and Recommendations

Overall, the primary aims of the project outlined in the terms of reference [(Appendix A)](#_Project_Terms_of) have been met. The final product has met all critical (must) functional requirements and has mostly met all preferable (should) functional and non-functional requirements, although the filtering system could be expanded to add a broader range of filters to improve its usefulness; such filters may include the distance from the user or the user who placed the marker. Generally, requirements listed under the ‘could’ section have not been met, the only one that has somewhat been met is a password criterion that ensures passwords are greater than the minimum length of 6 characters. Many of these requirements were, admittedly, unrealistic for the scope of the prototype, and listing so many of these requirements under the ‘could’ category was overly optimistic. Instead, more of these features should have been ruled out earlier on to reduce any uncertainty about the final product throughout the development phase.

The approach taken to development, and the tailored agile methodology used were beneficial to the project, by learning unfamiliar technologies alongside the development crucial time was saved. While difficulty was met keeping to the pre-determined schedule near the end of the project, an alternative methodology may likely have worsened this issue and resulted in a less than adequate product.

In the future development of the project, the unmet requirements should be considered and implemented where possible. The feature that would be most beneficial to add would be the scoring system, which was omitted due to complications. Taking inspiration from literature by [Amsterdamer and Milo (2015)](#_References), this feature would add an engaging method of crowd curation to data as well as measuring data usefulness. With a scoring system in place, it would then be possible for users to filter markers based on their usefulness as determined by the crowd. By prioritising useful data, the application itself may become more useful to users; as mentioned by [Winkler (2013)](#_References), the application is only as useful as the information it provides.

Taking inspiration from the origins of the project's conceptualisation, as disused in the terms of reference [(Appendix A)](#_Project_Terms_of), more research should be done into the addition of images to markers for the sharing of artwork. Currently, such a feature is limited by database file-size requirements, however, if an alternate storage method could be determined, this feature could add considerable substance to the product and increase its usefulness with the art community. Similarly, the addition of social aspects, such as viewing markers filtered by the user who placed them, and the ability to ‘follow’ users (similarly to Twitter or Instagram), could see a shift to the applications use as a more creative platform -encouraging social exploration and sharing of unique experiences.

To improve application accessibility to those who require additional aid, features including a screen reader and colour blind options could be added to the application. Screen reading could either be activated in the settings or appear as a loud-speaker icon on the view marker screen that users could press to enable dictation.

Availability of the application is a non-functional requirement of significant importance to this project, and while steps were taken to ensure the maximum availability of the project on Android devices, further research will be required into the development of an iOS version. Due to the significant architectural and design differences between operating systems, this recommendation is the most significant; as it may require an entire overhaul of the user interface and code. Research will be conducted into various cross-platform development options and techniques for porting software to various platforms to determine the feasibility of this process. By making the product available on a second major platform, the number of potential users is increased – which in turn means the amount of sourced data has potential to grow thus improving the usefulness of the application. Due to the increased maintenance that would be required for two separate versions of the application, an iOS version of the application will not be attempted until the implementation of other future recommendations is complete.

Funding of future development may be made possible through the implementation of advertainment, as was discussed in the requirements. Further research should be completed into this feature in ways that are not obtrusive to the user – such as the possibility of sponsored markers. Locating sponsors to fund such markers is likely to prove to be a more challenging route than using banner advertisement, so this feature may have to be put on hold until a sponsor could be located. Until then, banner advertisements with similar proportion the mock-up used to gauge user reaction [(Appendix M)](#_M)_Mock_Advert) may have to be used as a compromise to ensure the commercial viability of the application.

# References

Admob.google.com. (2019). *Google AdMob - Mobile App Monetization*. [online] Available at: https://admob.google.com/home/ [Accessed 13 Apr. 2019].

Amarilli, A., Amsterdamer, Y. and Milo, T., 2014, April. Uncertainty in crowd data sourcing under structural constraints. In International Conference on Database Systems for Advanced Applications (pp. 351-359). Springer, Berlin, Heidelberg.

Amsterdamer, Y. and Milo, T., 2015. Foundations of crowd data sourcing. ACM SIGMOD Record, 43(4), pp.5-14.

Android. (2019). *Android 9 Pie*. [online] Available at: https://www.android.com/versions/pie-9-0/ [Accessed 6 Apr. 2019].

Basiri, A., Amirian, P., Winstanley, A. and Moore, T., 2017. Making tourist guidance systems more intelligent, adaptive and personalised using crowd sourced movement data. Journal of Ambient Intelligence and Humanized Computing, pp.1-15.

Boim, R., Greenshpan, O., Milo, T., Novgorodov, S., Polyzotis, N. and Tan, W.C., 2012, April. Asking the right questions in crowd data sourcing. In 2012 IEEE 28th International Conference on Data Engineering (pp. 1261-1264). IEEE.

Clifton, I.G., 2013. Android user interface design: turning ideas and sketches into beautifully designed apps. Addison-Wesley.

Cressie, N. and Wikle, C.K., 2015. Statistics for spatio-temporal data. John Wiley & Sons.

Datanyze.com. (2018). [online] Available at: https://www.datanyze.com/market-share/mapping-and-gis/google-maps-api-market-share [Accessed 20 Dec. 2018].

Docs.oracle.com. (2019). *Data Concurrency and Consistency*. [online] Available at: https://docs.oracle.com/cd/B19306\_01/server.102/b14220/consist.htm [Accessed 13 Apr. 2019].

Firebase Open Source. (2019). *Firebase Authentication Password Hashing*. [online] Available at: https://firebaseopensource.com/projects/firebase/scrypt/ [Accessed 4 Apr. 2019].

Firebase. (2019). *Firebase Authentication  |  Firebase*. [online] Available at: https://firebase.google.com/docs/auth/ [Accessed 13 Apr. 2019].

Firebase. (2019b). *Usage and limits  |  Firebase*. [online] Available at: https://firebase.google.com/docs/firestore/quotas [Accessed 11 Apr. 2019].

Google Cloud. (2019). *Google Maps Platform Terms of Service  |  Google Maps Platform  |  Google Cloud*. [online] Available at: https://cloud.google.com/maps-platform/terms/ [Accessed 11 Apr. 2019].

Hoisington, C., 2015. Android Boot Camp for Developers Using Java: A Guide to Creating Your First Android Apps. Cengage Learning.

Kalnis P, Ghinita G, Mouratidis K, Papadias D (2007) Preventing location-based identity inference in anonymous spatial queries. IEEE Trans Knowl Data Eng 19(12):1719–1733

Odgis, J. (2015). *Keeping Color Consistent Is a Vital Element of Branding*. [online] HuffPost. Available at: https://www.huffingtonpost.com/janet-odgis/keeping-color-consistent-\_b\_8749334.html [Accessed 20 Dec. 2018].

Panko, R. (2018). *The Popularity of Google Maps: Trends in Navigation Apps in 2018 | The Manifest*. [online] Themanifest.com. Available at: https://themanifest.com/app-development/popularity-google-maps-trends-navigation-apps-2018 [Accessed 20 Dec. 2018].

Rollins, M., 2011. The business of Android apps development: making and marketing apps that succeed. Apress.

StatCounter Global Stats. (2018). *Mobile Operating System Market Share Worldwide | StatCounter Global Stats*. [online] Available at: http://gs.statcounter.com/os-market-share/mobile/worldwide [Accessed 20 Oct. 2018].

Winkler, R. (2013). Mapping the Hazy Way Forward for Waze. [online] Wall Street Journal. Available at: https://www.wsj.com/articles/SB10001424127887324682204578517200613837078 [Accessed 11 Nov. 2018].

xda-developers. (2019). *Google wants developers to add dark themes to save battery life*. [online] Available at: https://www.xda-developers.com/google-wants-developers-to-add-dark-themes-to-save-battery-life/ [Accessed 9 Apr. 2019].

# Glossary

**AI:** *Artificial Intelligence*

**API:** *Application Programming Interface*

**CCTV:** *Closed Circuit TV*

**CF:** *Collaborative Filtering*

**CPU:** *Central Processing Unit*

**GPS:** *Global Postioning System*

**GUI:** *Graphical User Interface*

**HTML:** *Hypertext Markup Language*

**ID:** *Identity*

**IDE:**  *Integrated Development Environment*

**IEEE:** *Institute of Electrical and Electronics Engineers*

**JDK:** *Java Development Kit*

**JSON:** *JavaScript Object Notation*

**OSX:** *Macintosh Operating System X*

**SDK:** *Software Development Kit*

**SQL:** *Structured Query Language*

**SRS:** *Software Requirements Specification*

**UI:** *User Interface*

**UML:** *Unified Modeling Language*

**XML:** *eXtensible Markup Language*

# Appendix

## Project Terms of Reference

#### A Mapping Application to Collect and Visualise Crowdsourced GPS Data

##### Background to Project

Upon completion, the project aims to produce a mobile application that combines GPS maps with crowdsourced data with social aspects. The application will present users with a map in which they mark the location of anything they desire, and other users will be able to see these marked locations on their own devices. The application will suggest some categories for users to choose from, such as facilities (i.e. toilets, bins), points of interest or locations of events. However, the user can choose to mark miscellaneous points on the map for whatever they wish. Icons for the different categories will appear differently on the map, for easy identification, and will show more detail about the marker when pressed. Filtering of categories will also help a user when they need to find something specific. Users will be able to vote on whether a marker is helpful or not, and the user who created the marker will receive points for their contribution.

The reason this project was chosen is to fill a gap in the market for a crowd-sourced mapping application which offers users with freedom over what they mark. While there are applications that exist to inform users of points of interest, the information provided is often picked by the developer and not the users. This can lead to a lack of information, or misinformation, regarding points of interest, particularly in rural areas where the application developer may be less informed than local residents. Related applications, such as satellite navigation application WAZE allow users to mark from a selection of categories; however, all of the categories relate to road hazards and traffic. This is an example of providing the user with the ability to choose what is marked but limiting their choices for a specific purpose. The proposed application will have no such limitations.

Personally, I have experienced difficulties while abroad with locating facilities, such as toilets, in large unfamiliar cities and a struggle to find places to dispose of waste in rural areas where bins are scarce. As well as this, the passing of many points of interest that would only be known by stumbling across them, such as street art, exciting sculptures or local events that many will miss as they are unmarked on most maps.

I first thought of the idea for a similar application in early January 2018 while helping a friend locate street art in Newcastle for a project she was working on. However, there was no online sources or maps which provided any kind of information as to where this artwork could be found. Instead, we resorted to using Google Maps and its street view feature to check streets for any artwork that could be useful, as well as recalling areas which we had previously noticed while walking around the city. From there, it was a case of planning a primary route and exploring, sometimes to find hidden gems, others to end up in the entirely wrong direction. This led to the initial idea of an application which would allow users to take pictures, and mark the GPS location, of street art to share with a broader community and allow others to visit the artwork in person. Since then, I decided that user’s contributions should not be limited to just one thing and that they should be given even more freedom to provide usefulness to a broader range of communities and individuals.

This app will provide a solution to these several dilemmas in a single place, providing users with useful information they otherwise would not have known. In some cases, this information may save them much time wandering around aimlessly to locate something, or simply give them something to do and see.

I imagine the application will be most useful to those who frequently travel to new, unfamiliar locations, or communities who wish to share points of interest.

This project has a healthy mix of research potential as well as a variety of methods for implementation. To create a working prototype should be reasonably complex and should consist of several well-thought-out features to improve the usability of the product.

Domains of this project will be social media and application design and development. I have previous experience developing simple applications for Android using a reasonably necessary development environment, most notably a simple drawing application. This simple application was made in an IDE that required little to no code. I have no experience with the design or creation of a social platform or with the use of GPS mapping and am unfamiliar with more complex, mainstream development environments for mobile applications. As a result, this project will provide several problems which will need to be overcome throughout the development process, in the form of unfamiliar technologies, development environments and languages. As well as the front end of the application, a back-end will require development to store the crowdsourced data and communicate with the application allowing queries to display correct and relevant data within the application based upon the user’s location and applied filters. Ideally, the majority of data will be stored externally on the database to ensure the application has a minimal footprint on the device on which it is installed.

The most important feature of this application is the ability to mark a location and have it shared with other users, however for the application to become frequently used it must have a simple, intuitive design and easy-to-use, aesthetically pleasing interfaces, additional features as well as good branding. People are unlikely to download an application that is poorly branded on the app store, and shows unattractive user interfaces, and are likely to stop using one that does not provide excellent usability.

While there are many desirable features to implement that could benefit the application overall, it is crucial to stick to the time-constraint and ensure the core functionality of the application is fully functional without allowing scope creep to significantly change the application or result in poor implementation of its features. To ensure the management of the projects scope, IEEE Software Requirement Specification will be used to clearly outline all possible features based on importance to functionality and difficulty of implementation as well as other criterion so there is never any doubt as to which features can and must be implemented within the timeframe, and which are out of scope. This ensures the creation of the application follows a structured and efficient process to maximise time.

##### Proposed Work

Proposed work for this project is primarily the development of a useful mobile application for Android devices that allows users to log on and mark points of interest on an interactive map, and view markers placed by other users, based on GPS location and categorical filters.

Application development will be completed using Android Studio, the official Integrated Development Environment for Android application development due to its extensive features and support. Use of Android Studios fast emulator also allows for faster testing and alleviates the need for an android device to be present.

Java will be the language of choice throughout development; as the official language of Android development, it will have the most significant availability of learning resources as well as being a programming language that I am familiar with through university teaching and projects.

For map functionality, Google Maps SDK for Android will be used as it is the most popular, therefore straightforward, mapping software with extensive and frequently updated mapping information and satellite imaging. The SDK also allows for the anchoring of icons to specific positions on the map, which is fundamental to the functionality of the proposed application. As well as this, the integration of Google Maps with Android will be simpler than other mapping solutions due to both being products of Google LLC.

Throughout development, it will be ensured that the application adheres to the Google Play policies and guidelines to ensure the app can be submitted to and remain available on the Google Play store. Otherwise, the development off the app would be redundant as it would be unusable.

Android’s greater availability as an open source operating system, being available in a broader variety of devices, lead to the decision for using this as the primary operating system for development. Android’s higher market share worldwide also provides more significant potential for the application to be used in a greater range of foreign territories where Apple phones are less common. This would aid the generation of crowdsourced content globally. While the future of the project could see the potential for an iOS counterpart of the application, this is out of scope for the permitted time.

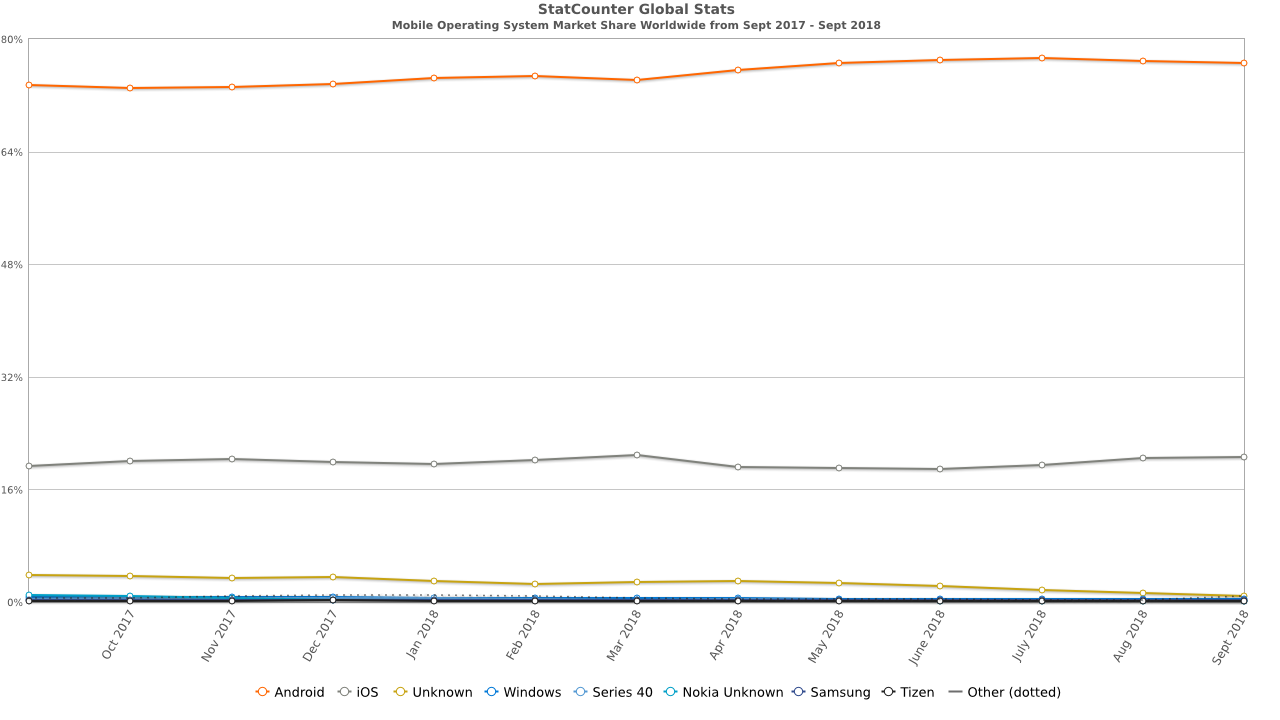


Figure 1 – Mobile Operating System Market Share Worldwide (gs.statcounter.com, 2018)

While the application will be developed in English, crowdsourced content can be entered in the user’s native or preferred language and will be displayed to other users as it was initially entered. However, the user interface will be mostly visual with minimal text, so it should not impede the user’s ability to make use of the app if they do not understand English. Extended language support and the ability to translate user’s markers, most likely using Google Translate, are both features that could be implemented in the future of the project however are out of scope within the development process.

Design and implementation of an SQL database to hold user information and marker information will be required. Data stored for users will include usernames, passwords and scores. Marker data will include the marker id, location, title, description, score and the user who created the marker. This will be implemented by using a university-based MySQL web server. The database will be frequently queried and edited in a live version of the application so should be scalable and able to handle significant frequencies of simultaneous queries; however, this is not required during the initial development and testing of the prototype. Communications will need to be established using the mobile application to connect to and query the database. Data returned will need to be accurate and relevant to the query and displayed in a clear and useful fashion within the application.

Areas of investigation will consist of the design and development of mobile applications on Android, including good practices of design and implementation, mobile interface design and HCI, integration of Google Maps interface and use of Maps SDK for Android, design and implementation of SQL web server for use with a mobile application.

##### Aims of the Project

The aims of the project are:

* Investigate the design and implementation of Android applications that adhere to the Google Play policies and guidelines, making use of good practices and well-designed user interfaces and additional SDKs.
* Design and build an Android application that collects and displays crowdsourced data submitted by users using GPS and Google Maps SDK in a simple, intuitive, functional way.

##### Objectives

1. Write a literature review and construct a Requirements Specification (Analysis)

a) GPS and crowd-sourced support utilisation techniques and tools (Section in Chapter 2)

b) Usability and accessibility tools and techniques (Section in Chapter 2)

c) Construct a summary of tools and techniques to be used in the project and tools and techniques to be carried forward to future recommendations (Section in Chapter 2)

d) Construct a Requirements Specification using IEEE Template (Section in Chapter 2/3 and appendix document

Deliverable: - Chapter 2 in the report, and one appendix

2. Complete tutorials, read literature and make use of other resources to improve knowledge and familiarity with (Analysis):

a) Android Studio IDE

b) Google Maps SDK

c) Using Java for mobile applications

3. Design, build and test the Product (Synthesis)

a) Analyse the Requirements, using Use Case Diagrams, Descriptions and Class diagrams and any other relevant UML tools.

b) Design the product, using wireframes, storyboards and any other relevant UML tools. Incorporate good practice from the literature review

c) Create a test strategy and implement through testing.

d) Build and user test the product.

4. Evaluate the project and make recommendations for the future of the product (Evaluation and Conclusions).

a) Evaluate the product against the requirement specifications and using feedback from volunteer users.

b) Evaluate the process, time, tools, methods and skills used.

c) Conclude if the aim and if each objective has been met and how well each one has been met.

d) Make recommendations for the future of the product.

5. Write Abstract and Introduction (Abstract and Introduction)

Deliverable: - Abstract and Chapter 1 in the report

##### Skills

Throughout this project, I will make use of several skills I have learned previously as well as being required to learn new skills, particularly within the development aspect of the project. Some of the skills previously attained were done so in a different context but are transferable to this project.

Most of the skills required are more focused on the development stage of the project, with some lesser focus on the design and testing phases. To gain these new skills I will use online resources, such as tutorials and guides, as well as making use the university and public libraries for relevant books regarding the listed topics.

###### Existing Skills

* Simple Android application development

Through studies at GCSE computing, I gained a brief introduction into mobile development using MIT’s App Inventor for Android, a basic, code-minimal IDE. I can build upon these through this project by becoming familiar with an IDE of greater complexity.

* Mock user interface designs and wireframing

Through previous software engineering projects at A-Level, as well as through university modules, such as Web Programming, Program Design and Development, and Software Engineering Practice I have had previous experience with modelling interface designs.

This skill will be used throughout the design phase of my project, to design an appropriate user interface for the application, ensuring that an interface is made for all possible screens to avoid ambiguity during the construction of the application. While these skills were learned in the context of desktop applications and website designs, they will be transferable to my project.

* Database design and integration

Learning basic SQL at A-Level, then furthering this knowledge at university through the Relational Databases module, as well as implementing a MySQL database with a simple website in the Web Technologies module. I have gained skills with SQL queries and designing databases that have been correctly normalised using Entity Relationship Diagrams (ERDs). These skills will be used in both the design and implementation of the application, to ensure that the database is functional, efficient and adequately normalised, to ensure robust implementation.

* Program design using UML and good practice

Gaining the required skills to analyse the requirements of a system properly, and represent findings using appropriate UML was learned initially through the Systems Analysis module and was reinforced throughout the Program Design and Development and Software Engineering Practice modules. These skills will be used during the planning and design phase of the project to ensure that the application is well thought out and functional as well as operating in a useful way before implementation.

* Procedural and object-oriented programming skills and good coding practice

Basic procedural programming skills from GSCE computing, advancing through A-Level computing and the Procedural Programming module in university. Object-oriented programming skills gained from university teachings in Object-Oriented Programming modules and reinforced in Software Engineering Practice modules. Good coding practice, especially regarding layout and commenting has been reinforced through all stages of learning. During the implementation phase of the project these skills will be critical, as Android development will be done in Java, the object orientated programming language of choice throughout my university studies. Good coding practice, including layout and thorough commenting, will also aid the development of the product as well as future-proofing it, allowing for code to be easily understood when adding further features in the future of the project.

* Project management skills

Being responsible for managing an A-Level computing project, including communications and meetings with a client, has given me the skills to oversee an individual project. Through Software Engineering Practice I learned how to structure a team project, as well as the various software development methodologies, as well as putting one into practice in the team project assignment. Time management skills were required to ensure elements of the project were functioning and were on time. Project management skills will be crucial throughout the entirety of the project to ensure deadlines are met with work meeting a high standard and ensuring that all core features can be implemented appropriately within the permitted time.

* Creating and carrying out precise testing plans

Demonstrating this skill in my A-Level project to ensure the provided solution was functional before being delivered to the client, then again in the Software Engineering Practice module to ensure that the program was fully functional and was free of bugs before being demonstrated to the marking lecturer. This skill will be carried out throughout both the development stage and the dedicated testing stage of the application, to ensure the application is robust and functional.

###### New Skills Required

* Developing more complex mobile applications using Android Studio IDE

As discussed in the existing skills section, the previous experience with Android application development was done using a simple IDE and little to no coding. Throughout the project, I will be required to use Android Studio, the official IDE of Android Application Development, alongside Java to create an application of higher complexity than previously experienced.

* Using the Google Maps SDK for Android

To provide the core functionality of the application, I will be required to use the Google Maps SDK for Android to provide users with an interactive map on which they can see markers displayed in a clear and useful way, as well having the ability to add their own markers to the map. The program will have to query the database to find markers near to the user’s location and add them as markers to the map.

* Testing mobile applications

While I have experience with testing plans and testing software created for use on a desktop environment, I have little experience with testing applications specifically tailored to a mobile platform, and the additional tests that may be required due to the hardware and software differences of the target operating system/device.

* Designing and implementing a database for use with a mobile application

While I have experience with designing and implementing a MySQL database for use in local databases for desktop applications, as well as web server-based databases for use on a website, I have no experience with databases accessed from mobile platform or using a database alongside an application to display data in a visual manner, such as on an interactive map.

##### Sources of Information

Android Developers. (2018). *Meet Android Studio  |  Android Developers*. [online] Available at: https://developer.android.com/studio/intro/ [Accessed 20 Oct. 2018].

StatCounter Global Stats. (2018). *Mobile Operating System Market Share Worldwide | StatCounter Global Stats*. [online] Available at: http://gs.statcounter.com/os-market-share/mobile/worldwide [Accessed 20 Oct. 2018].

##### Resources

* A suitable computer for application development

An up to date Windows 10 or OSX computer with decent performance will be needed for application development to ensure the project goes smoothly, and the tools used are not out of date. I will make use of my own laptop and desktop, as well as university computers where necessary. Use of a laptop or university provided computer would also allow for the demonstration of the product within the university.

* Android Studio IDE

The official integrated development environment of Android applications, which is free to download from the software’s official website. While alternatives are available, this IDE offers the most exceptional support of features and has a broader range of learning resources which may be required when starting to use the software.

Simulation of an android phone is also possible within the IDE so alleviates the need for additional hardware or simulation software.

* Google Maps SDK for Android

Necessary for the integration of Google Maps within the application. Will require a project to be registered for testing in the Google Cloud Platform Console to receive an unrestricted API key.

* Microsoft Office Suite

The preferred software suite for word processing, Microsoft Word will be used to write up all reports and compile all documentation regarding this project. Microsoft Visio will be used to create the database ERD and UML diagrams. All necessary Microsoft Office applications are available for use on university-provided computers, with some software being accessible from personal computers using a university log in. Free alternatives are also available online if necessary, such as Google Documents.

* Adobe Creative Cloud Suite

For any graphic design, and some of the user interface design, Adobe’s creative cloud suite of products offers several products that may come in useful. Photoshop and Illustrator are useful for creating graphics for use in the user interface and creating detailed mock-ups. Adobe XD is an application for designing and prototyping websites and mobile applications, so could be used to draft the user interface for my application. The Adobe suite of products is available on university-provided computers and can be accessed from home with a valid Adobe Creative Cloud subscription. Alternative image manipulation software is available, such as GIMP, which is free, and Pixelmator which is a paid application available on the Mac App Store.

* StarUML

A free tool for creating UML diagrams, which will be particularly useful through the design phase of the project for specific UML diagrams. This is available on university computers and on personal computers.

* GitKraken

A free desktop client that acts as an alternative for the official GitHub client, GitKraken will be used to upload all documentation to GitHub so that it can be accessed anywhere, if necessary. This also provides a convenient method to back up my project as it progresses, to prevent any possible data loss and provide a helpful timeline to check which work was done and when to allow me to manage my project better. Alternatively, the official GitHub desktop app can be used on university computers.

##### Structure and Contents of the Project Report

##### Abstract (Objective 5)

##### Introduction (Objective 5)

* Aims and objectives
* Overview of the proposed product and work
* Reason for choosing and undertaking this project
* Tools and resources required, and reason for choosing
* Overview of the planned approach to the project

##### Analysis (Objective 1)

* Review of literature relevant to the proposed project
* GPS integration and tools
* Use of crowdsourced data
* Tools and resources required, and reason for choosing
* Overview of the planned approach to the project
* Requirement specification

##### Synthesis (Objective 3)

* MoSCoW analysis of requirements
* Application user interface designs
  + The reasoning for design choices
* Application code and assets
* Use Case Diagrams
* Class Diagrams
* Database design
  + Normalisation
  + Entity Relationship Diagrams
* Database SQL Code
* Testing
  + Test Plan and Strategy
  + Results

##### Evaluation and Conclusions (Objective 4)

* Evaluate the product against the requirement specifications and using feedback from volunteer users
* Evaluate the process, time, tools, methods and skills used
  + How suitable were the processes and tools used?
  + Would alternatives have been better?
* Discussion of any issues that arose throughout the project
* Evaluation of which objectives have been met, and to what extent
* Discussion of recommendations for the future of the project

##### Marking Scheme

Software Engineering Project

Overview

Report: 40%

Abstract & Introduction 5%

Analysis 30%

Synthesis 30%

Evaluation & Conclusions 30%

Presentation 5%

Product 50%

Fitness for Purpose 40%

Build Quality 60%

Viva 10%

##### Project Plan

Below is a Gantt chart representing the initial plan for project timings. This may change throughout the project, as some things may take more, or less, time than initially expected however provides an introductory guide to when I should expect to work on specific aspects of the project ensuring it will stay on schedule within the time constraint of this project.

Building the application is expected to be the most time-consuming process, as it requires the most learning and will produce the most problems which will need to be solved and overcome.



##### Appendix

1. Usability and accessibility criteria
2. Requirement Specification
3. Use case diagrams
4. Use case descriptions
5. Class diagram
6. Wireframes
7. Storyboards
8. Test Results
9. User Feedback

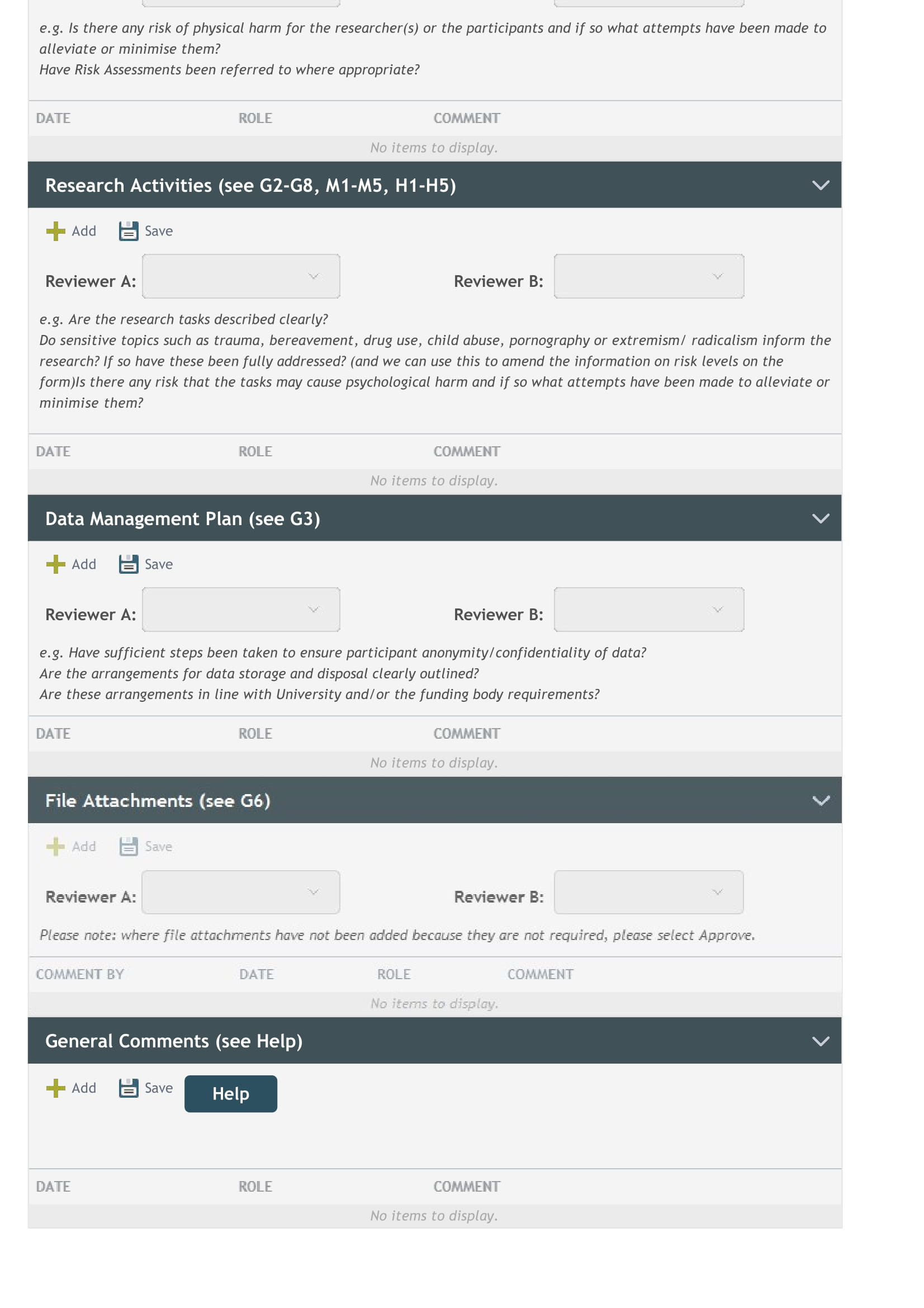
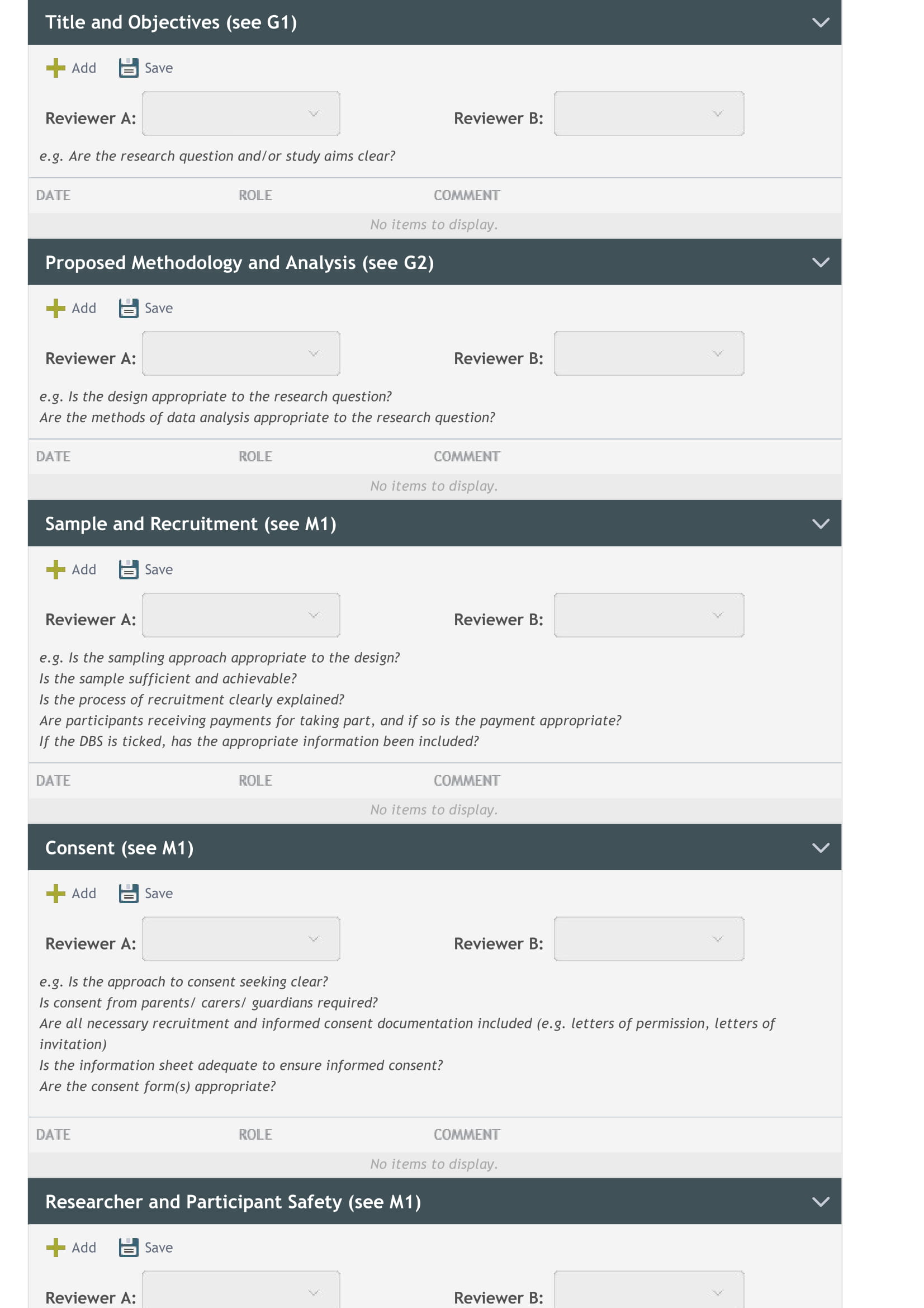
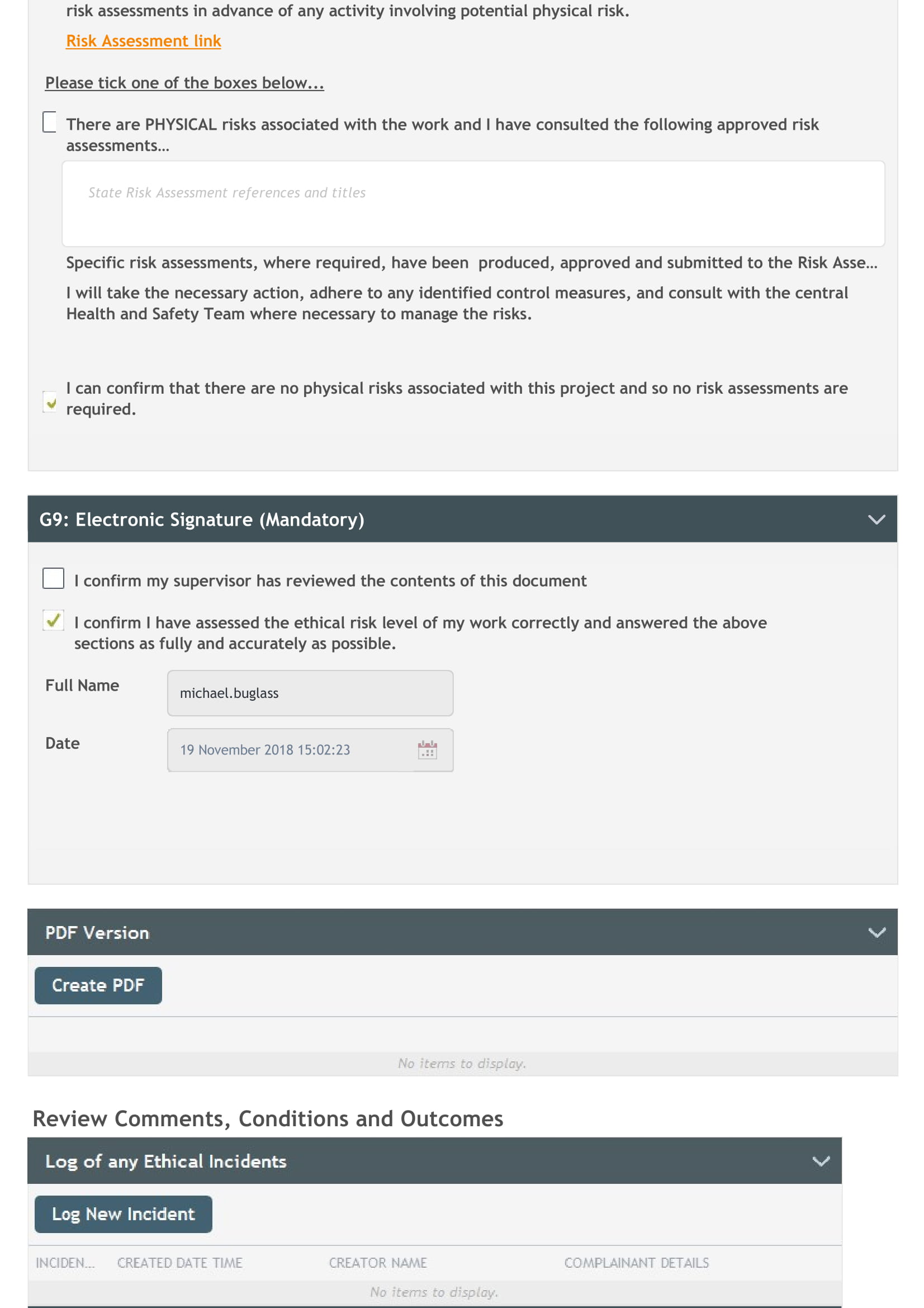
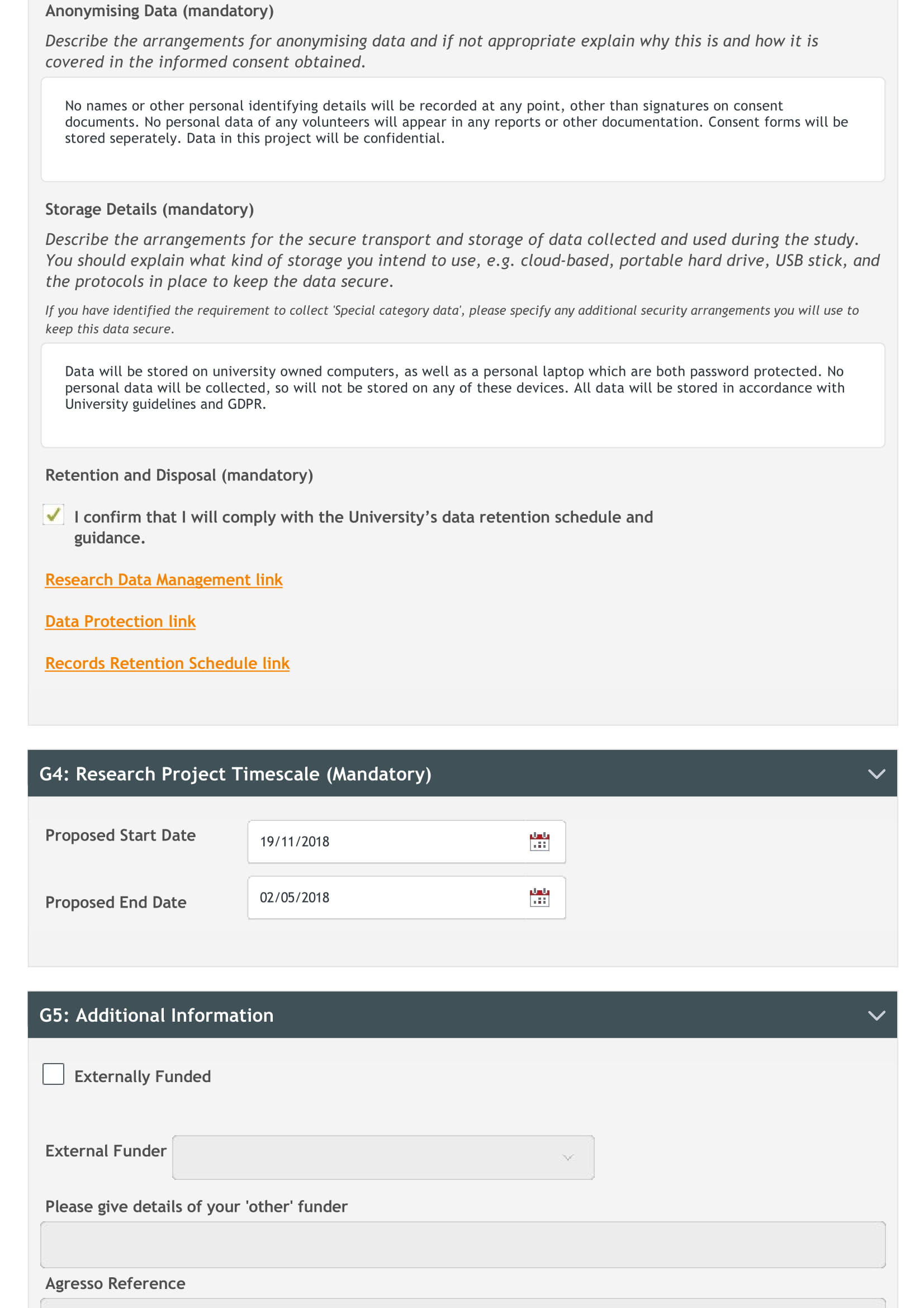
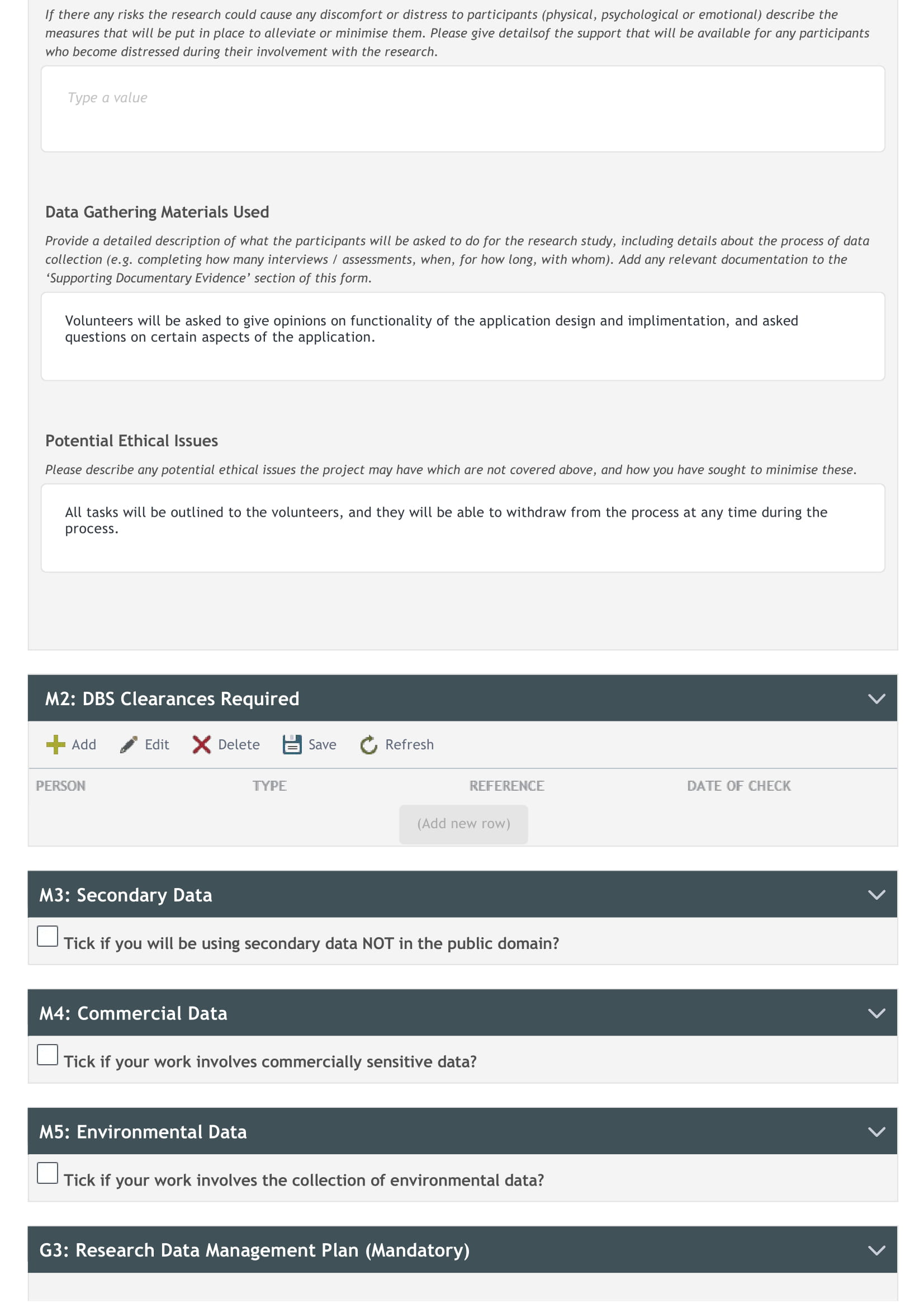
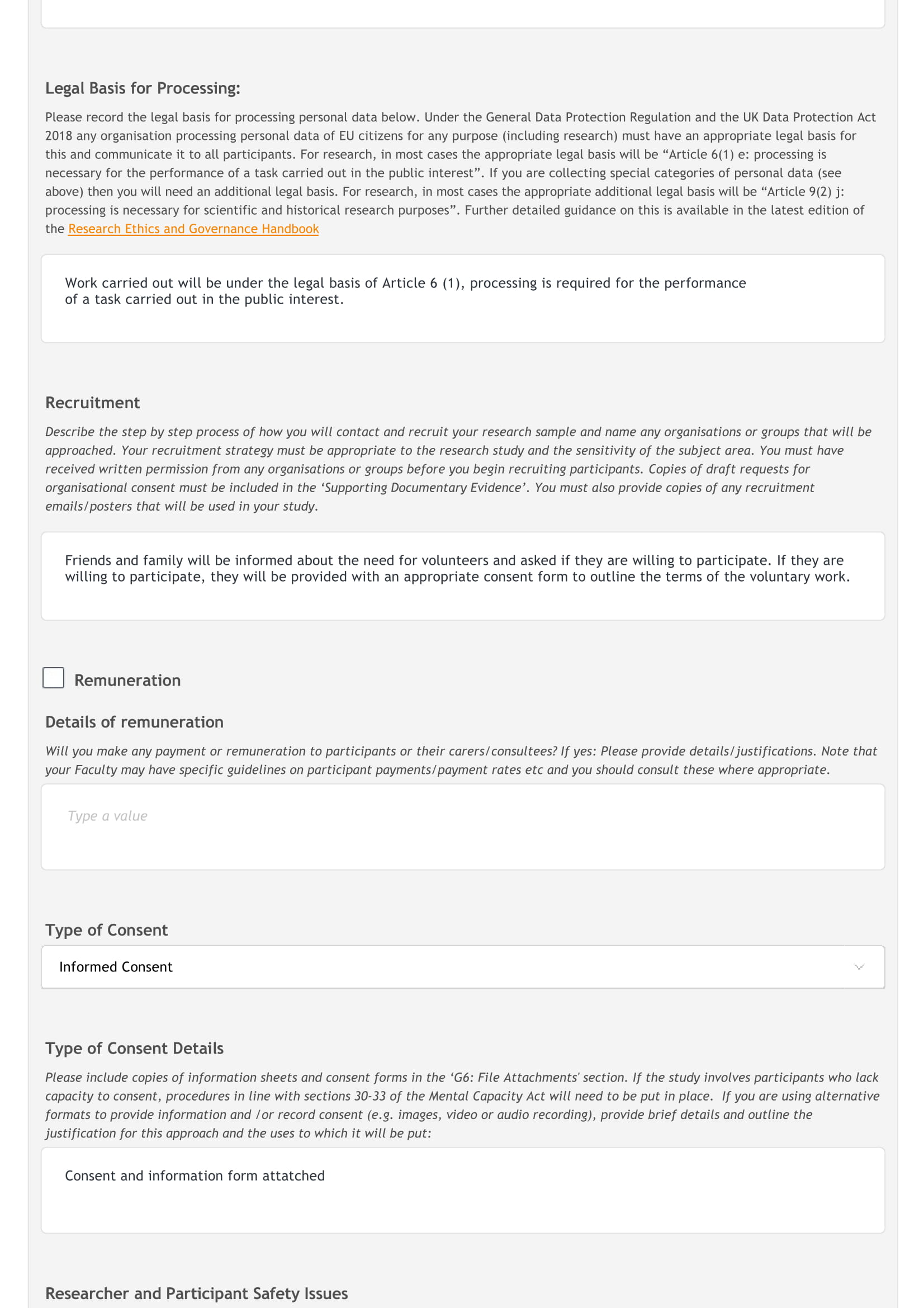
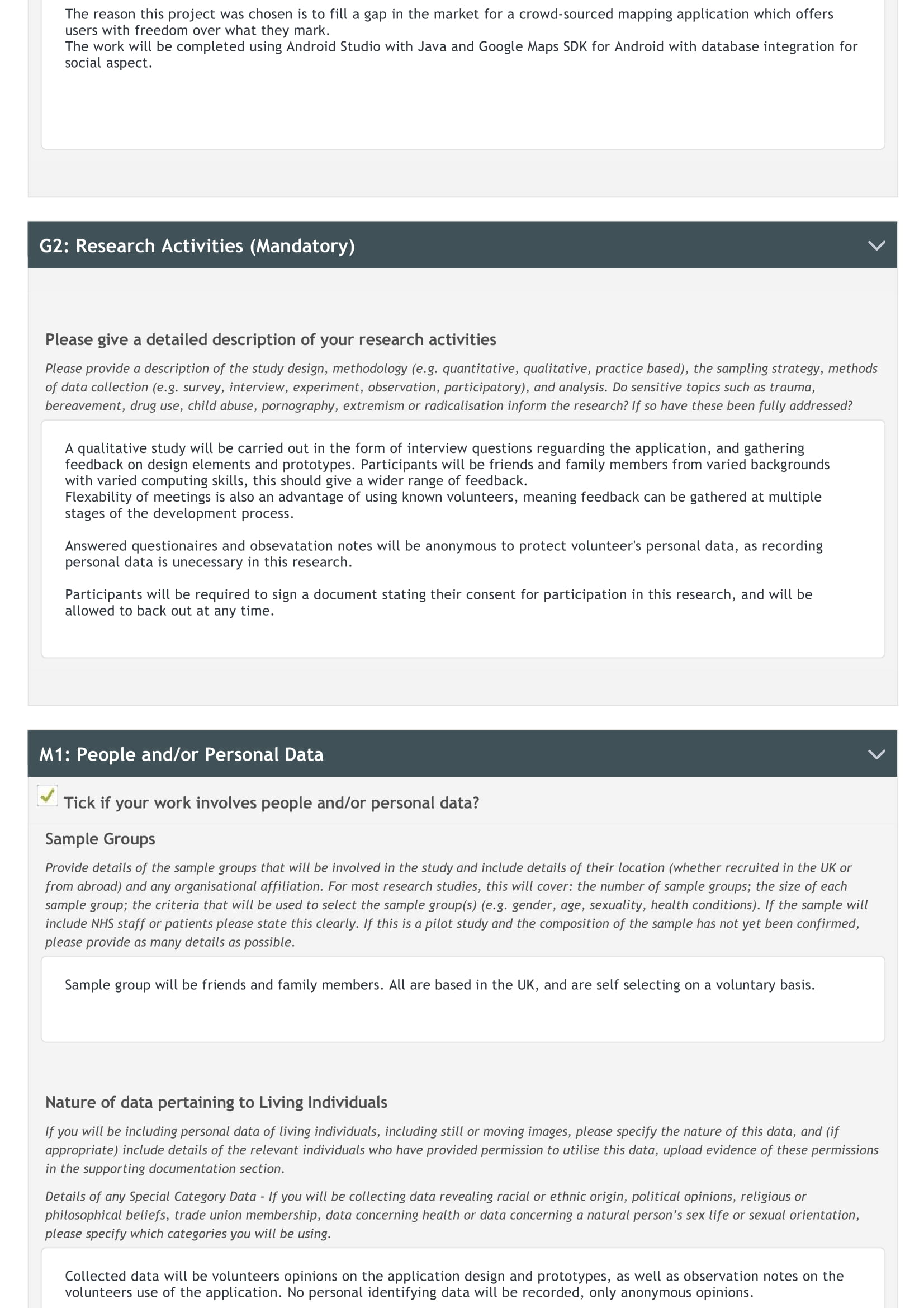
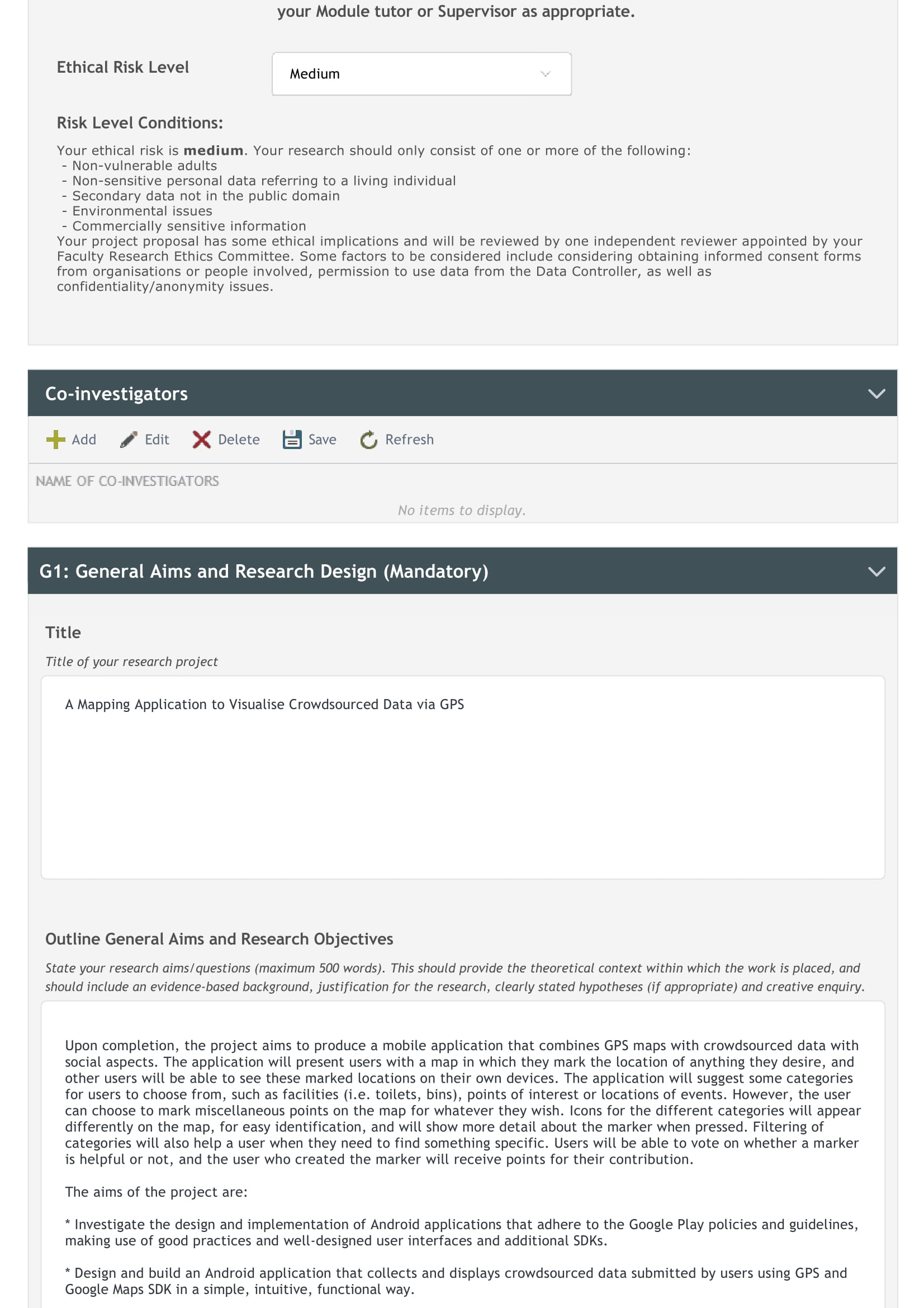
###### Ethics Form

To be confirmed when further instructions are received, but this project will include people to inform requirements and to test prototype products – Amber. To ensure the protection of personal data, all participants will be given an information sheet and asked to sign a consent form. No personal data will be stored at any time. Any communications will be done via Northumbria email, and if meeting in person, no data will be taken at the time of the meeting.

###### Risk Assessment Form

No health and safety issues are anticipated. At no time will the researcher be in a situation with strangers, all participants will be friends or fellow students.

## Ethics Form



## Consent Form Template

**Research Project Statement of Information, Intent and Confidentiality**

Thank you for agreeing to participate in this study, to produce a user-friendly and useful crowd-sourced mapping application.

I (Michael Buglass, [michael.buglass@northumbria.ac.uk](mailto:michael.buglass@northumbria.ac.uk)) am the sole researcher on this project.

The aims of this project are:

* Investigate the design and implementation of Android applications that adhere to the Google Play policies and guidelines, making use of good practices and well-designed user interfaces and additional SDKs.
* Design and build an Android application that collects and displays crowdsourced data submitted by users using GPS and Google Maps SDK in a simple, intuitive, functional way.

Participants in the study would be required to provide feedback on application designs, and usability of the application prototype to support development. Any participation would be at your own convenience and times will be agreed in advance.

It will be possible to withdraw from participation at any point during this project should you change your mind. All information will be securely stored, and password protected.

All data will be anonymous and at no point will responses be linked to any personal data.

**Consent Section**

I consent to the University of Northumbria recording and processing information resulting from my responses to the questions set in the above research project. I understand that this information will be used for the purposes set out in the information sheet and consent is conditional upon Northumbria University complying with the obligations under the Data Protection Act 1998.

Name: Date:

## d) IEEE Systems Requirement Specification

IEEE SRS Template used is Copyright © 1999 by Karl E. Wiegers (https://www.processimpact.com/). Permission is granted to use and modify this document to meet the needs of personal projects.

**Software Requirements Specification**

**for**

**A Mapping Application to Collect and Visualize Crowdsourced Data**

**Version 1.0**

**Prepared by Michael Buglass**

**Northumbria University**

**2018**

**Table of Contents**

**Table of Contents ii**

**Revision History ii**

**1.** **Introduction 1**

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

**2.** **Overall Description 2**

2.1 Product Perspective 2

2.2 Product Functions 3

2.3 User Classes and Characteristics 4

2.4 Operating Environment 4

2.5 Design and Implementation Constraints 4

2.6 User Documentation 5

2.7 Assumptions and Dependencies 5

**3.** **External Interface Requirements 6**

3.1 User Interfaces 6

3.2 Hardware Interfaces 7

3.3 Software Interfaces 7

3.4 Communications Interfaces 7

**4.** **Other Nonfunctional Requirements 8**

4.1 Performance Requirements 8

4.2 Safety Requirements 8

4.3 Security Requirements 8

4.4 Software Quality Attributes 8

**Appendix A: Glossary 9**

# Introduction

* 1. **Purpose**

The purpose of the product is to provide a prototype application that can collect and display locational data provided by users through crowdsourcing. Reasoning behind this project was a gap-in-the-market for a mobile application that allows users to share locations on a communal map. Due to the portability of the platform, it is the ideal device for which to produce such an application as it can be used anywhere, providing an internet connection is available.

Ideally the application will provide a platform for various communities to share locational information across various categories. Such locations may include practical facilities, such as bins or restrooms; locations of memorials or statues; the location of street art and murals or even a building of architectural or historical significance.

Inspiration from the project came from a research project into street art in Newcastle-upon-Tyne, in which an attempt was made to locate areas containing such art. However, no consistent source could be found which contained any form of locational data for this artwork and locations were subsequently discovered through guesswork.

The early idea for the project would have provided a platform for artwork to be marked on a communal map for all users to see, allowing artists and enthusiasts to share work.

However, it was later decided that the application would be more useful if its purpose was not limited to sharing of artwork location, but a location that could be deemed interesting by the user.

* 1. **Intended Audience and Reading Suggestions**

The intended audience for this application is not limited to any specific group, rather anyone who wished to share interesting locational data and potential get to know more about the area around them using information provided by other users.

* 1. **Product Scope**

The produced prototype is not to be considered a final implementation of the product, but rather display the core functionalities of the concept.

Primary functionalities will include:

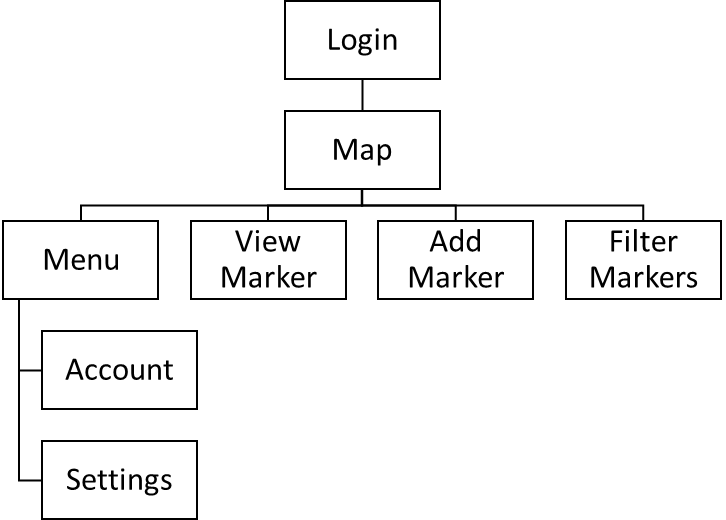
* Creation of accounts and secure login capabilities.
* Ability to change account details and application settings.
* Addition of markers to an online database, including information about what is located at each marker.
* Visual presentation of data stored within the database on a map.

1. **Overall Description**
   1. **Product Perspective**

The product to be created is a new, self-contained application for Android mobile devices that will use the community to source geographical data, specifically points of interest, and visualise this data in a way that is considered useful to the user. After the user has logged in, a compulsory step for the use of this application, they will be presented with a map, centered on their current location, that will be the core of the application. From this map, the user will be able to place markers on the map for everyone to see, highlighting points of interest, whether it be landmarks, facilities or a local event. Alternatively, the user can browse markers place by other users, either by dragging the map to view around them or using the search feature to search for another area.

This application originated from an earlier application idea that would allow users to mark street art on a map, with a picture, title, and description to allow such artwork to be located with greater ease as it often difficult to locate, hidden away in backstreets of large cities where people would not think to look. However, this application would only have served a niche audience while having provided the base for a more useful application. By allowing users to mark anything they like, and then allowing them to filter this information as they wish, it opens the base idea of the application up to a broader audience. The street art community that the original idea would have catered to would be able to choose the ‘Street Art’ category, for example, while someone else might need help to locate a toilet or a bin and filter by these facilities accordingly.

The diagram below illustrates the primary functional screens of the application. Since the application is used on a mobile device, only one screen will be visible at a time, but the user is able to access all screens by following the hierarchy of the diagram.

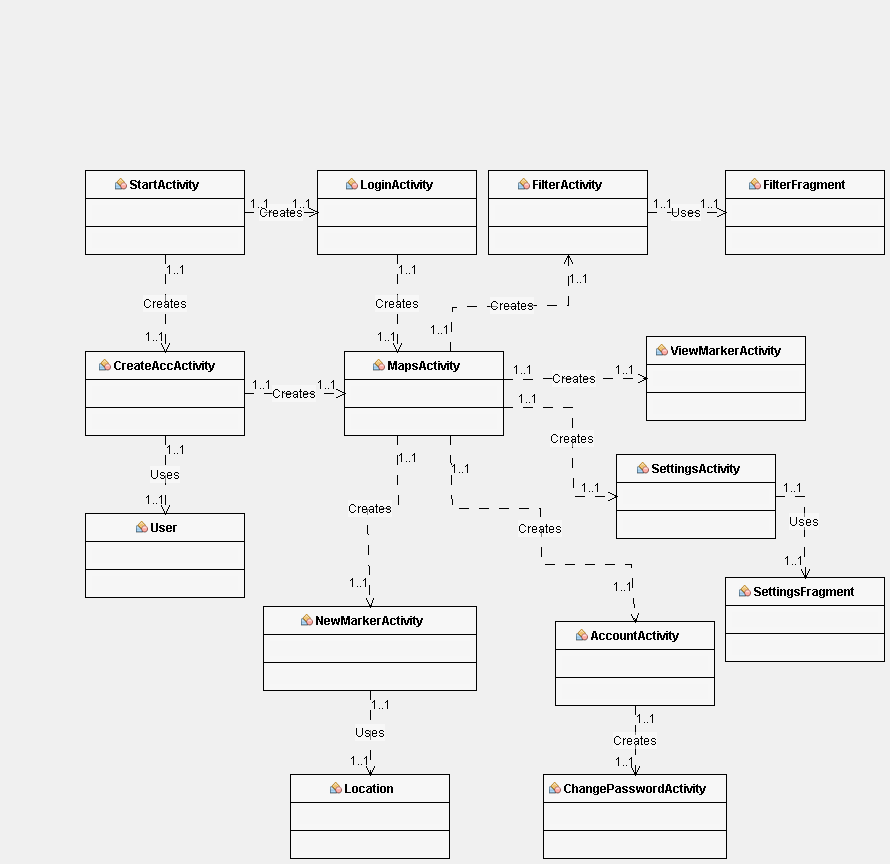
**

* 1. **Product Functions**

The major functionalities of the application will be:

* The ability to view a map containing nearby markers
* The ability to place markers that other users can see
* The ability to filter markers based on a specific criterion

The class diagram below illustrates the intended communication between application classes and UI screens, class methods have been omitted.



* 1. **User Classes and Characteristics**

There will only be one user class in this system, which will be the base user. This class will be able to perform all critical functionalities of the application. Any administrative tasks will be done outside of the applications, such as editing the database manually to remove malicious content or to ban users. As there is only one user class, it is essential that the functionality of this class is appropriately satisfied as it will account for the entire user base.

* 1. **Operating Environment**

The application will operate on Android mobile devices. Specifically, testing will be carried out on three versions of the operating systems. The first two versions listed below will be tested using an android emulator due to the lack of availability of several physical test devices.

Firstly, ‘Pie’, Android version 9.0, the latest version of the Android operating system released on August 6th, 2018 (<https://www.android.com/versions/pie-9-0/>).

The second operating system version used for testing will be ‘IceCreamSandwich’, Android version 4.03, the version at which Google determines as providing approximately 100% device coverage for applications developed using this version. As such, this will be the version the application is developed to support.

Finally, version 5.0.1 will be used for testing on the native Android mobile device that will be used for volunteer testing and field testing.

Operating on mobile devices, the application can be used potentially anywhere. However, the main scenarios in which it is expected to be used is while the user is stationary, potentially sitting at home looking for somewhere to go and, secondly, while the user is on the move looking to reach a marker they have found, likely in a city or other area that may attract tourists.

For map functionality, the application must coexist with the Google Maps SDK for Android. This allows the map to be used within the application and allows markers to be displayed on the map. When a marker is created it will be added to the database, and then the map will refresh to display the newly added marker on the map.

* 1. **Design and Implementation Constraints**

The application must run on a wide range of Android devices, so should be as efficient as possible to run on older phones that may have less memory and slower processors than newer high-end phones. Since battery life is a concern with mobile devices, efficiency is vital to ensure that unnecessary computation does not reduce the battery life of the phone significantly. The polling rate of the GPS should be limited to a value that is somewhat frequent, but not so frequent that it could be considered unnecessary as constantly checking GPS locations can be a severe battery drain.

Functionality of the application will be limited to what is achievable in Android Studio IDE and with the Google Maps SDK; however, these technologies will be able to support all considered features of this application. For commercialisation, the Google Mobile Ads SDK could also be implemented to provide banner adverts on the application that would produce revenue when seen and pressed.

The application will store data both locally, for device-specific information, and online, for storing users and markers. Local data will consist of settings and auto-login data so that the user does not have to sign in each time they wish to use the application.

Language requirements are to be reduced by making the application as visual as possible. While the application will only be developed in English, by using clear icons it should be usable for those who do not understand the language without much difficulty. Since data is crowd-sourced, the user will be able to create a new marker in their own language and have it displayed to other users as it had initially been input. By minimizing language constraints in this way means the application will be usable by everyone regardless of language.

To ensure that user’s passwords are secure, they should be encrypted on the database so that they cannot be viewed directly in the case of a security breach. The amount of data stored about users will be limited to what is essential, their desired username, their email and their password.

* 1. **User Documentation**

The application will not come with any separate documentation or manuals; however, a help section may be provided in the menu section outlining specific tasks if it is considered necessary to do so. Volunteers will be used to assess the usability of the application, and if they are unable to figure out a particular function of the program through the user interface, this function will be considered for an explanation in the help screen.

* 1. **Assumptions and Dependencies**

Testing of the application will assume that the performance of the application prototype on the Android Studio emulator and any test hardware that may be used will reflect the performance of the software overall. This is because it is not cost effective to purchase a wide variety of Android-based hardware for testing, as some companies do, as this application is a prototype and is non-commercial in this state.

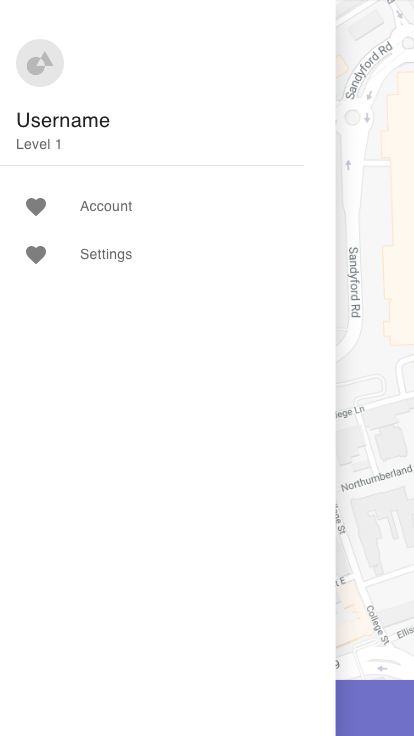
Similarly, it is assumed that the application will be backwards compatible with previous versions although will only be tested on the latest version due to the difficulty with testing on a large variety of older, unsupported, software versions.

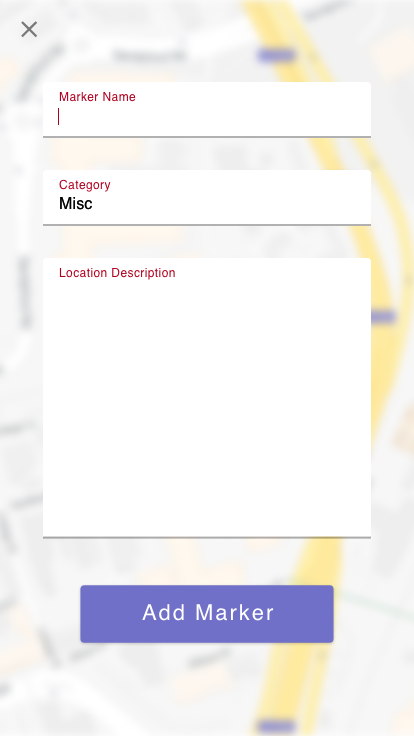
If in the future of the project the application was to be commercialised using Google AdMob adverts, in the form of banners on the application, or sponsors, which may be sponsored markers, it would be assumed that the advertisement that appears will be appropriate for the audience and not considered explicit. Google AdMob adverts are generated through the SDK and often take the user’s search data into account where available, so it may not be possible to predict what would appear.

1. **External Interface Requirements**
   1. **User Interfaces**

The user interface will use a simple layout consisting of informative, standardized buttons, such as a magnifying glass for search, a marker icon to place the marker and a hamburger menu button to bring up the side menu. Recurring colours will be used where possible throughout the application to form a consistent identity and house style for the application that users may associate with its colours.

For standardized component appearance, the Google Material Design UI kit has been used, with component designs specifically made to cater to the Android aesthetic for improved synergy across mobile applications.

**

**

(Concept User Interface Designs, not reflective of the final product)

* 1. **Hardware Interfaces**

The user interface will require a mobile phone with a touchscreen to function and will not use any audio devices or other input devices. The on-screen keyboard will be necessary when logging in, searching locations, entering data when creating or editing a marker, and changing personal information in settings. It will not be possible to use this application where a touchscreen is not available except for emulation software that simulates such hardware.

Communication with the online database will require the device to have an internet connection, so the application must be used on an internet capable device to function correctly. Use of an online database is necessary to allow data to be shared and viewed by other users. Database queries will use the NoSQL format used by the Cloud Firestore real-time database. To allow simultaneous use of the application across several devices, the database must be able to handle concurrency.

* 1. **Software Interfaces**

Logging into the system will require the input of user data into the system; this will be checked against the records in the online database and verified. Once logged in, the user's data is carried forward to the rest of the application screens so that their markers are linked correctly to their account, and so they can edit their personal data in the account menu.

The main screen of the application will connect to the online database to find markers in the user’s area; it will get locational data from the phones GPS, then the marker information will be used in the Google Maps SDK to show these markers on the map inside the application.

Any use of the Google Mobile Ads SDK will not require any data input as the SDK generates these adverts automatically.

* 1. **Communications Interfaces**

TCP protocol will be used in database communications using SQL. Data will be entered into and checked against database records using electronic forms presented in the application as editable text boxes.

Database communications do not need encryption for the most past as little sensitive data is stored; however, passwords should be encrypted in the application and stored in their hashed form within the database to prevent the password being exposed in a database breach.

1. **Other Nonfunctional Requirements**
   1. **Performance Requirements**

Timing of GPS polling should be limited to an amount that does not inhibit the usability of the application. By only polling the GPS where necessary battery life may be conserved by reducing the required computation.

Similarly, the map should only be refreshed when an event occurs, such as the user loading the map UI or adding, removing or editing a marker. This prevents unnecessary refreshing of the map, which could waste battery life and slow down the application, especially on older, less powerful devices.

* 1. **Safety Requirements**

There are no safety issues which would arise that are specific to using this application. The only time a user’s safety is at risk is in situations that arise from using a mobile device in an unsafe manner, such as while driving or not paying attention to surroundings.

* 1. **Security Requirements**

To comply with data protection laws, such as the Data Protection Act 2018 and the General Data Protection Regulation (GDPR), the minimum amount of data will be recorded. As a result, no personal identifying data will be stored as it serves no functional purpose. The only data recorded about users will be their username, email and password.

Authentication will be required to access the application; this will be done by logging in with an email and password. Authentication is required to deter malicious use of the application and ensure that the quality of crowdsourced data is of a higher quality.

The storage of passwords should be secure to reduce the risk of passwords being compromised should an attack take place on the database. This will ideally be accomplished using a secure encryption algorithm. It should be possible for the user to change their password to prevent unauthorized log in where their password may have been compromised

By implementing a password requirement criterion, the quality and security of user passwords could be improved at the source thus increasing the overall application security. However, this may prove inconvenient to users who wish to use an existing password that does not meet these requirements, and since minimal data is stored in the database, high security is not vital.

* 1. **Software Quality Attributes**

Availability is an important attribute to consider with mobile development, as many Android users will be using different versions of the operating system as a result of having phones of different performance and age. Ensuring the application can be used across a range of Android versions allows a more significant number of users to access the application, thus improving the quantity of crowdsourced data and therefore the usefulness of the application.

Portability of the application will be necessary for the success of this application, as such should be easily usable on the go. To aid this attribute, the UI should consist of few buttons that are of an excellent size to be usable on the move, and not difficult to click precisely as smaller buttons would be.

Reliability, robustness and maintainability will be a desirable attribute and will be enhanced by clearly laying out and labelling code so that it can be clearly understood. This allows for improved error checking of code and editing and improvement of features in future updates.

Usability will be tested using volunteers and improved based on feedback. This will be an essential feature to ensure that users continue to use the product and have a good initial first impression of the application. User interface designs will be shown to volunteers to allow for constructive criticism to produce an aesthetically pleasing design that is straight forward and easy to use. Through observing the volunteers use of the application it will be possible to gauge the ease-of-use of the application based upon how naturally they are able to perform tasks without guidance.

**Appendix A: Glossary**

*GPS: Global Positioning System*

*IDE: Integrated Development Environment*

*SDK: Software Development Kit*

*UI: User Interface*

*SQL: Structured Query Language*

*TCP: Transmission Control Protocol*

## e) User Interface Colour Palette

Colour scheme built using https://coolors.co/



## f) Colour Palette colour blind Views

All following images depict the appearance of the colour palette shown in Appendix E to those who possess the corresponding form of colour blindness. This should be used as a guide to imagine how the colours of the chosen palette will appear to such users. Colour scheme built using https://coolors.co/

Protanopia



Deuteranopia

Tritanopia



Achromatopsia



Achromatopsia (ordered from dark to light for comparison)



Protanomaly



Deuteranomaly



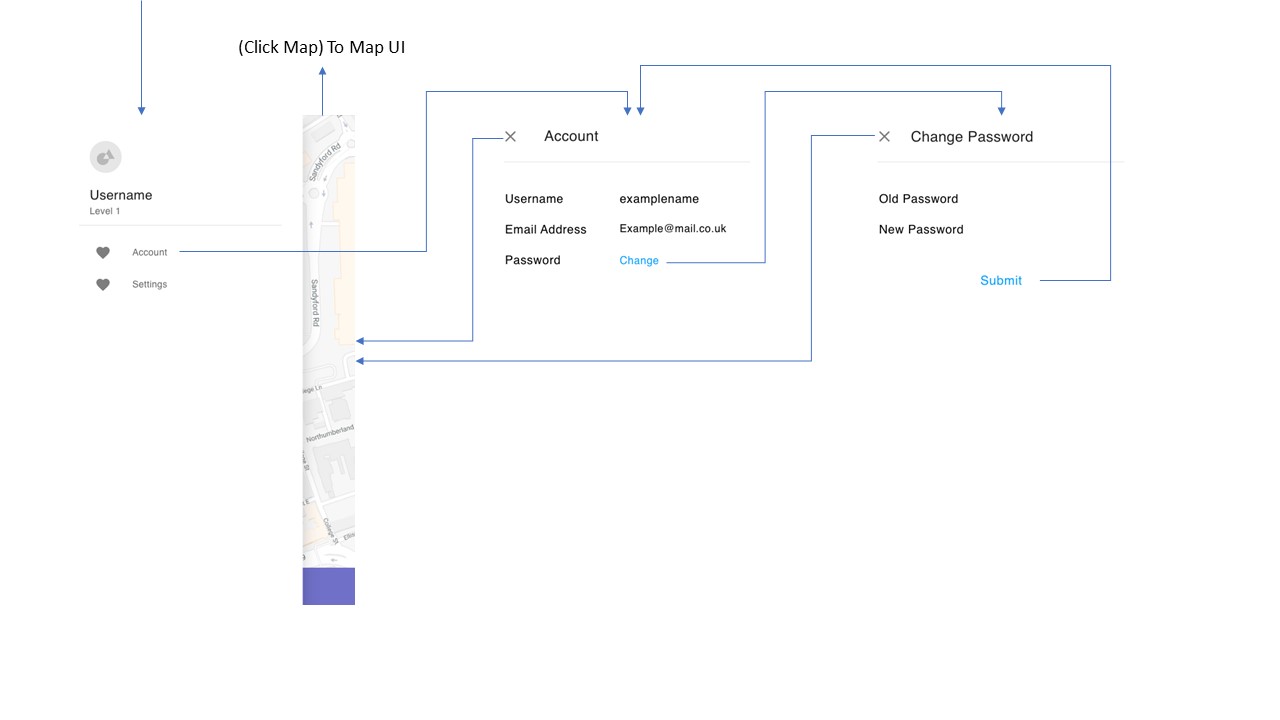
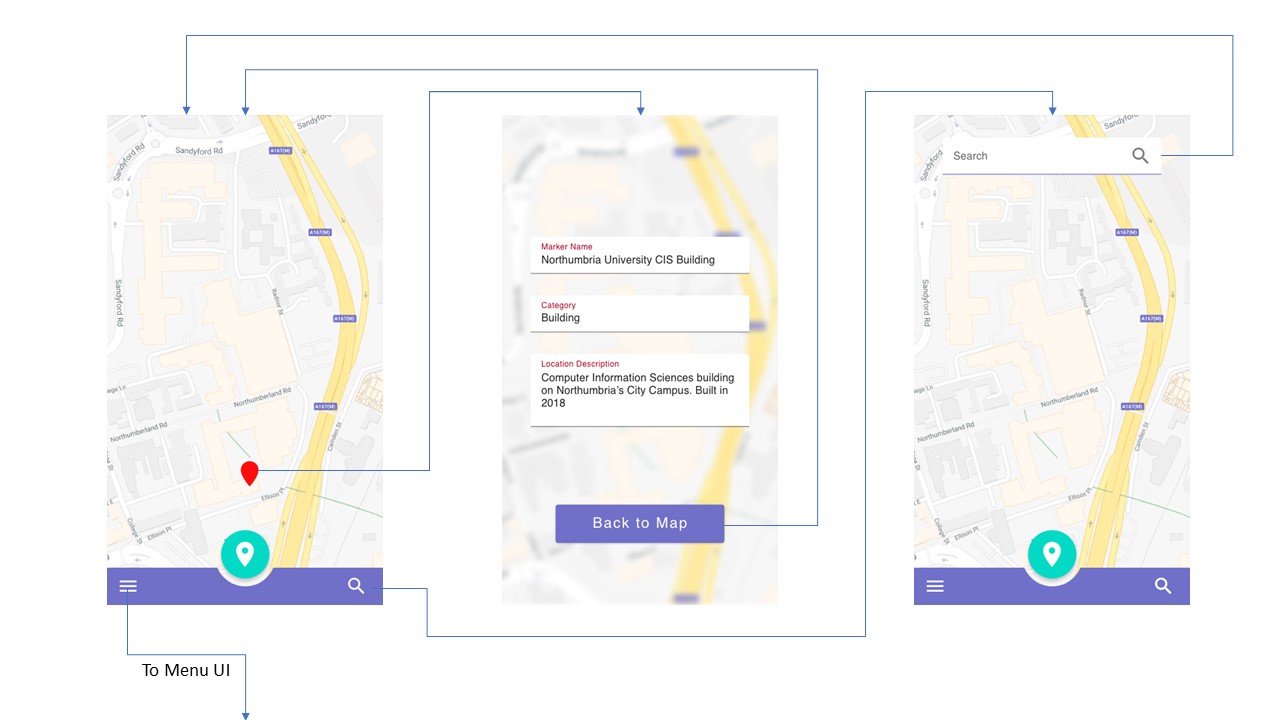
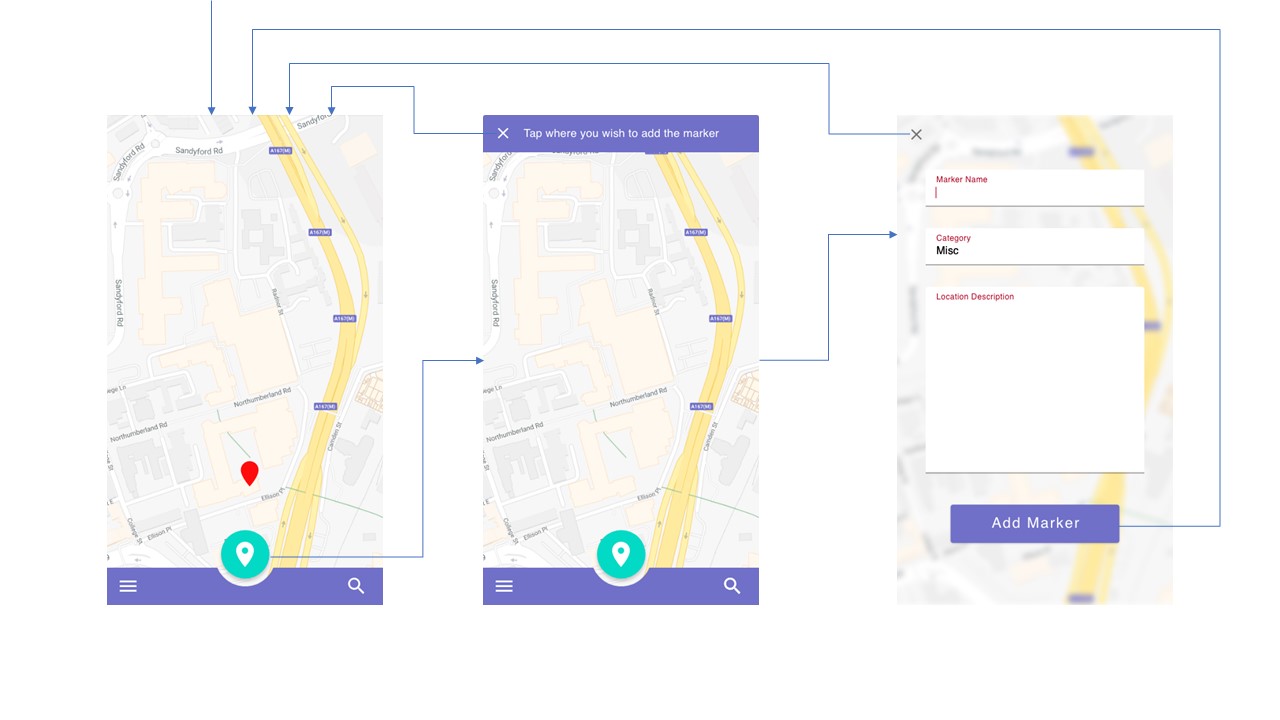
Tritanomaly



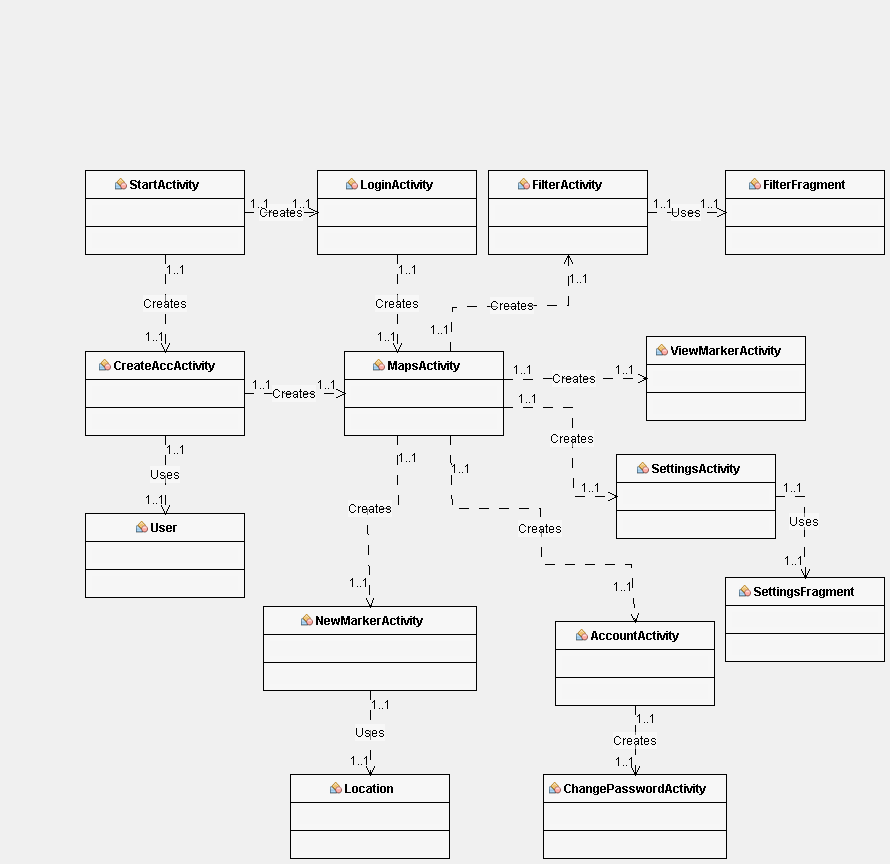
Achromatomaly

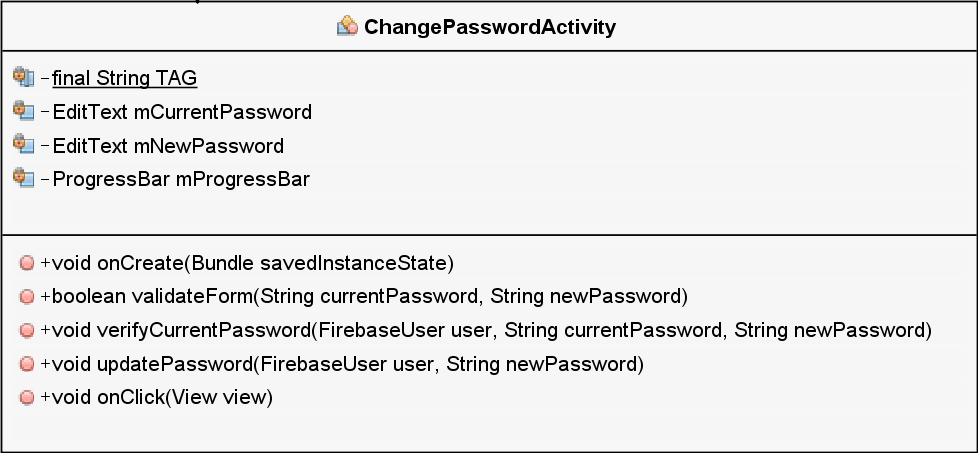
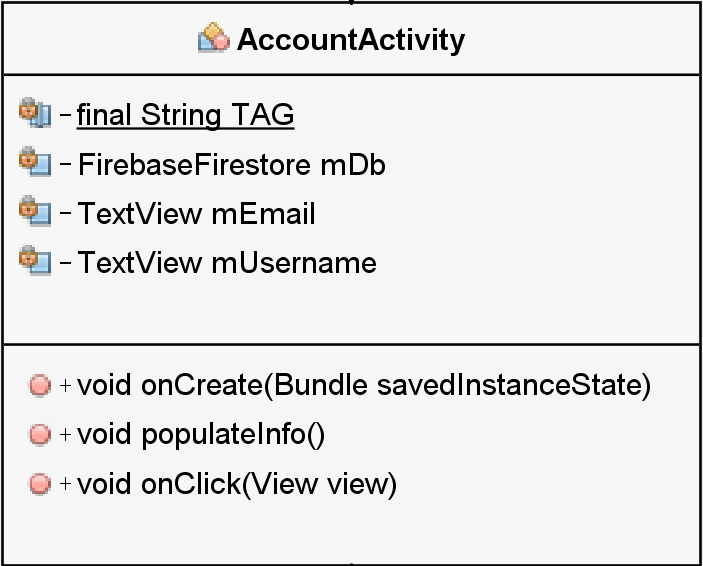


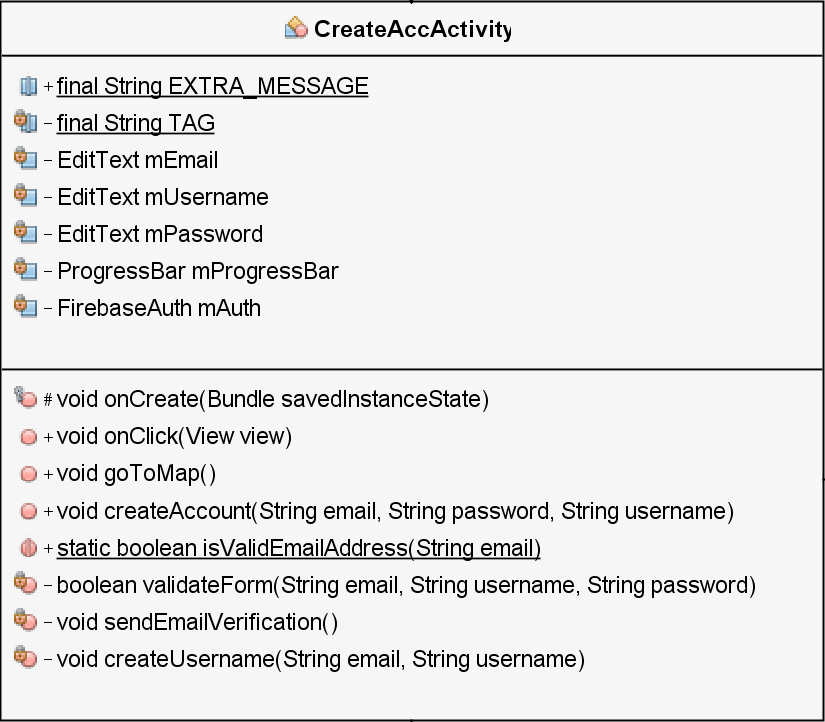
## g) Application Storyboard Illustrating Flow of Control

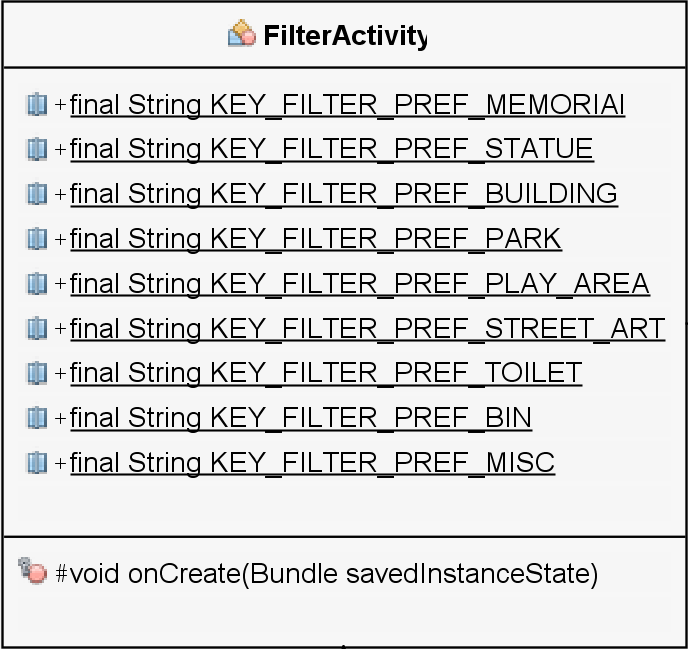


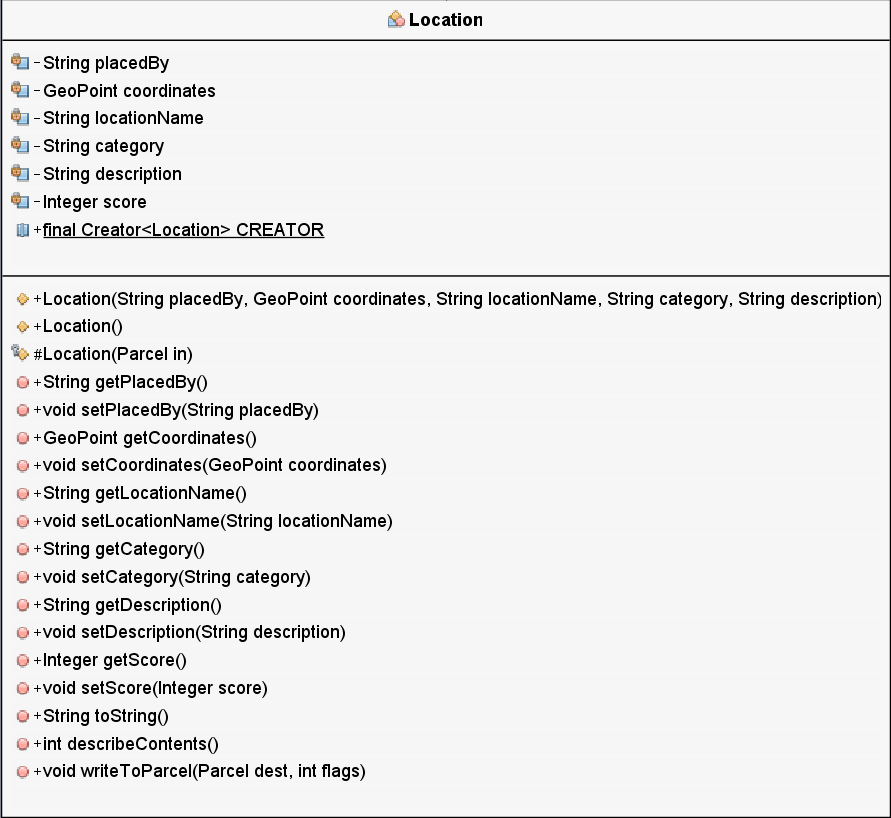
## H) Class Diagram and Class Details

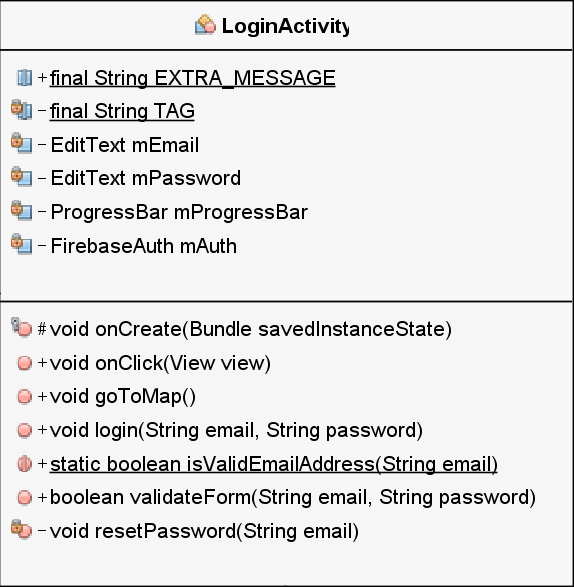
Due to the scale of the diagram, methods had to be removed from classes. Class details will follow the diagram in alphabetical order.

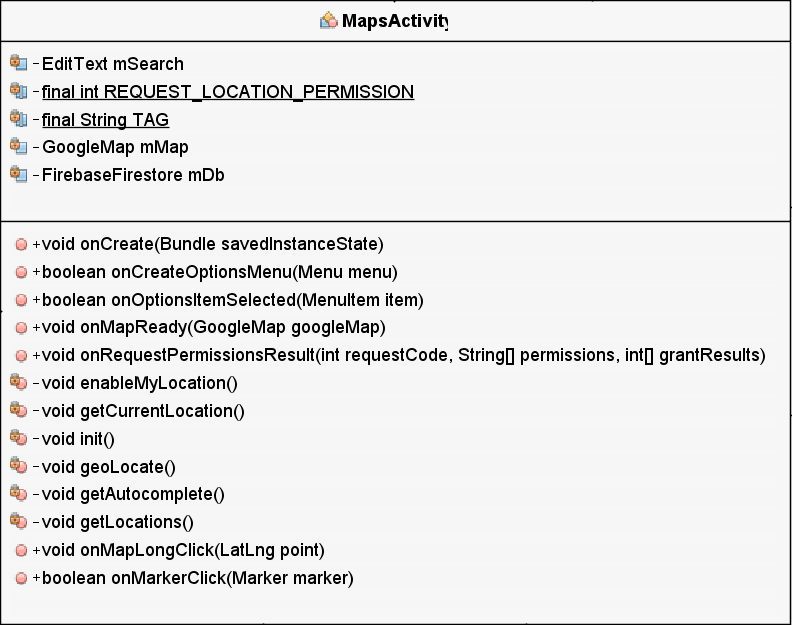


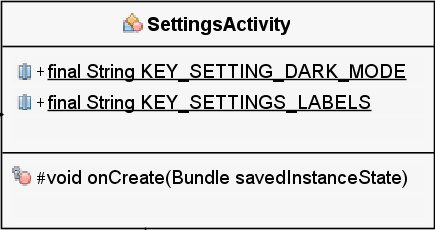


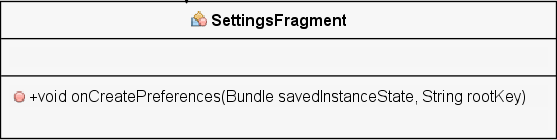
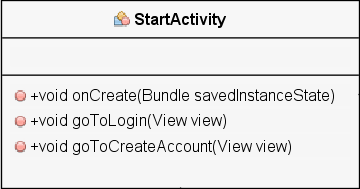


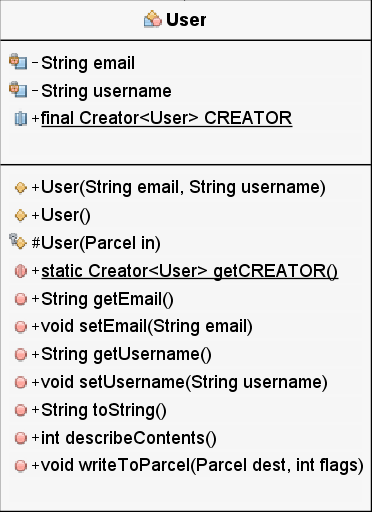
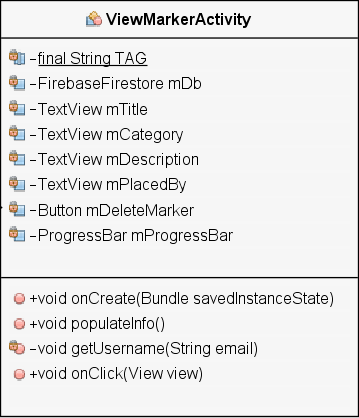
 



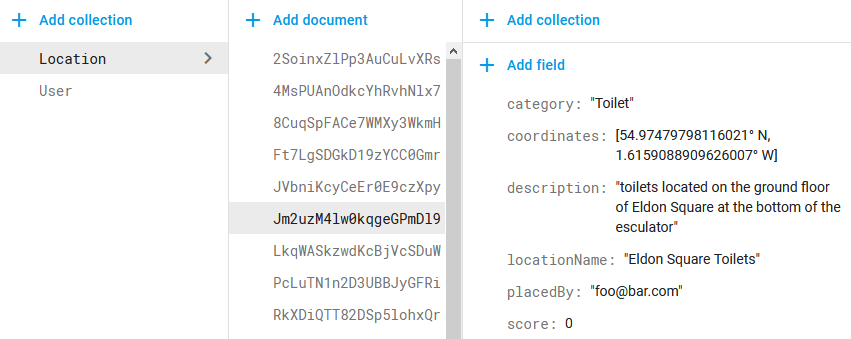


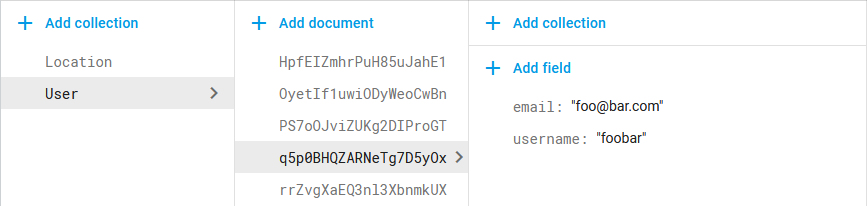
 

## I) Cloud Firestore Structure Screenshot





## J) Testing Plan and Results of Testing

|  |  |
| --- | --- |
| Requirement / Release | Start Activity General Functionality |
| Module Name | StartActivity.java |
| Date Created | 05/01/2019 |
| Updates inc date |  |

|  |  |
| --- | --- |
| **Test Case** | **Notes** |
| N/A | N/A |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| SA-1 | Click the log in button and go to the create account screen | No test case needed  Click log-in button | Application should switch to the log in activity screen | As expected | Pass |
| SA-2 | Click the create account button and go to the create account screen | No test case needed   * + - 1. Click create account button | Application should switch to the create account activity screen | As expected | Pass |
| SA-3 | Click other sections of the screen except the log in and create account buttons to ensure random clicks do cause error | No test case needed  Click areas of screen not including buttons | Nothing | As expected | Pass |

|  |  |
| --- | --- |
| Requirement / Release | Log In Activity General Functionality |
| Module Name | LoginActivity.java |
| Date Created | 06/01/2019 |
| Updates inc date |  |

|  |  |
| --- | --- |
| **Test Case** | **Notes** |
| A1 | User does not already have an account |
| A2 | User already has an account |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| LA-1 | Attempt to log in user who does not have an account | Uses test case A1   1. Enter ‘valid’ email and password combination 2. Click log in button | The user will be informed that the username and password combination do not match, indicating that the user does not exist | Application crashes | Fail |

**Details on LA-1 Failure:**

The error lay with the Firebase User Authentication library, producing the following error message:

“The library com.google.android.gms:play-services-base is being requested by various other libraries at [[15.0.1,15.0.1]], but resolves to 16.0.1. Disable the plugin and check your dependencies tree using ./gradlew :app:dependencies.”

Searching the error online message lead to a Stack Overflow article suggesting that all Android Studio libraries and plug-ins must be updated. However, upon attempting to update Android Studio and relaunching the application the update would not have been applied, and a message would state that an update was still available. It turned out that IOBit Malware Fighter Pro software installed on the PC used was blocking the IDE from updating, and therefore was the reason why the gradle sync was unable to succeed. After closing the IOBit software using Task Manager, and attempting the update again, the IDE was able to successfully update all libraries in the project. The following lines were updated in build.gradle:

implementation 'com.google.firebase:firebase-auth:16.1.0'

implementation 'com.google.firebase:firebase-core:16.0.7'

The error message no longer occurs, and the application can compile and run successfully.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| LA-1.2 | Attempt to log in user who does not have an account | Uses test case A1  1. Enter ‘valid’ email and password combination  2. Click log in button | The user will be informed that the username and password combination do not match, indicating that the user does not exist | Application produces error message and remains at login | Fail |
| LA-2 | Log in user who does have an account | Uses test case A2   1. Enter valid, matching username and password pair 2. Click log in button | The user will be taken to the main application screen | Application crashes | Fail |

**Details on LA-1 Failure:**

**Issue:**

The following error message is taken from the runtime console and details the error:

com.google.firebase.auth.FirebaseAuthException: The given sign-in provider is disabled for this Firebase project. Enable it in the Firebase console, under the sign-in method tab of the Auth section.

**Solution:**

Going to <https://console.firebase.google.com/> and activating the Email/Password sign in provider solved the issue.

**Details on LA-2 Failure:**

**Issue:**

Google Maps activity crashed upon opening the activity, meaning the program could not proceed past the log in. Searching for the issue online suggested checking the file build.gradle, where an error message highlighted: All com.android.support libraries must use the exact same version specification.

**Solution:**

Updating the version number for libraries that were out of date and the addition of three extra libraries solved the issues with the build.gradle file and the addition of a library to the AndroidManifest.xml solved the issues, and allowed the map to function correctly.

Re-test and continuation of testing:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| LA-1  (Attempt 3) | Attempt to log in user who does not have an account | Uses test case A1  1. Enter ‘valid’ email and password combination  2. Click log in button | The user will be informed that the username and password combination do not match, indicating that the user does not exist | As expected | Pass |
| LA-2.2 | Log in user who does have an account | Uses test case A2   1. Enter valid, matching username and password pair 2. Click log in button | The user will be taken to the main application screen | As expected | Pass |
| LA-3 | Log in user using incorrect email | Uses test case A2   1. Enter incorrect email address 2. Enter valid password 3. Click log in button | The user will be notified that their email and password combination is incorrect | As expected | Pass |
| LA-4 | Log in user using incorrect password | Uses test case A2   1. Enter correct email 2. Enter incorrect password 3. Click log in button | The user will be notified that their email and password combination is incorrect | As expected | Pass |
| LA-5 | Click other sections of the screen except the log in and back buttons to ensure random clicks do cause error | No test case needed   1. Click areas of screen not including buttons | Nothing | As expected | Pass |
| LA-6 | Return to previous screen using the back button | No test case needed   1. Click back button in upper left of screen | Application should switch to the start activity | As expected | Pass |
| LA-7 | Reset password | Test case A2   1. Enter email address 2. Click password reset button | A password reset email should be sent to the entered email | As expected | Pass |

|  |  |
| --- | --- |
| Requirement / Release | Login Activity Validation |
| Module Name | LoginActivityTest.java |
| Date Created | 06/01/2019 |
| Notes | Automated Testing using Junit test classes |

|  |  |
| --- | --- |
| **Test Case** | **Notes** |
| A1 | Uses normal data |
| A2 | Uses boundary data |
| A3 | Uses erroneous data |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Data | Expected | Actual | Pass/Fail |
| LAT-1 | Valid email passes | Test case A1  “foo@bar.com” | Valid | Valid | Pass |
| LAT-2 | Valid email passes | Test case A2  “foobar123@foobar123.co.uk” | Valid | Valid | Pass |
| LAT-3 | Invalid email fails | Test case A3  null | Fails | Passes | Fail |
| LAT-4 | Invalid email fails | Test case A3  “foobar” | Fails | Fails | Pass |
| LAT-5 | Invalid email fails | Test case A3  "foobar.com" | Fails | Fails | Pass |
| LAT-6 | Invalid email fails | Test case A3  “@” | Fails | Fails | Pass |
| LAT-7 | Invalid email fails | Test case A3  “@..” | Fails | Fails | Pass |
| LAT-8 | Invalid email fails | Test case A3  “@.com” | Fails | Fails | Pass |
| LAT-9 | Invalid email fails | Test case A3  “foobar123@foobar123” | Fails | Passes | Fail |
| LAT-10 | Valid email and valid password pass | Test case A1  “foo@bar.com”  "@[Pa55w0rd!]" | Passes | Passes | Pass |
| LAT-11 | Valid email and valid password pass | Test case A1/A2  “foo@bar.com”  “@@@@@@” | Passes | Passes | Pass |
| LAT-12 | Valid email and valid password pass | Test case A1/A2  “foo@bar.com”  “123456” | Passes | Passes | Pass |
| LAT-13 | Valid email and valid password pass | Test case A1/A2  “foo@bar.com”  “passwo” | Passes | Passes | Pass |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| LAT-14 | Valid email and valid password pass | Test case A1/A2  “foo@bar.com”  “PASSWO” | Passes | Passes | Pass |
| LAT-15 | Valid email and valid password pass | Test case A1/A2  “foo@bar.com”  “12345678912345678912” | Passes | Passes | Pass |
| LAT-16 | Valid email and valid password pass | Test case A1/A2  “foo@bar.com”  "\uD83E\uDD8A \uD83E\uDD9D \uD83D\uDC3B \uD83D\uDC3C" | Passes | Passes | Pass |
| LAT-17 | Valid email and invalid password fails | Test Case A3  “foo@bar.com”  null | Fails | Passes | Fail |
| LAT-18 | Valid email and invalid password fails | Teat Case A3  “foo@bar.com”  “” | Fails | Fails | Pass |
| LAT-19 | Valid email and invalid password fails | *Test Case A3*  “foo@bar.com”  “ ” | Fails | Fails | Pass |
| LAT-20 | Valid email and invalid password fails | *Test Case A3*  *“foo@bar.com”*  *“12345”* | Fails | Fails | Pass |
| LAT-21 | Valid email and invalid password fails | *Test Case A3*  *“foo@bar.com”*  *"123456789123456789123"* | Fails | Fails | Pass |
| LAT-22 | Invalid email and valid password fails | *Test Case A3*  *Null*  *"@[Pa55w0rd!]"* | Fails | Passes | Fail |
| LAT-23 | Invalid email and valid password fails | *Test Case A3*  *“”*  *"@[Pa55w0rd!]"* | Fails | Fails | Pass |
| LAT-24 | Invalid email and valid password fails | *Test Case A3*  *“foobar”*  *"@[Pa55w0rd!]"* | Fails | Fails | Pass |
| LAT-25 | Invalid email and valid password fails | *Test Case A3*  *“foobar.com”*  *"@[Pa55w0rd!]"* | Fails | Fails | Pass |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| LAT-26 | Invalid email and invalid password fails | *Test Case A3*  *null*  *null* | Fails | Passes | Fail |
| LAT-27 | Invalid email and invalid password fails | *Test Case A3*  *“”*  *“”* | Fails | Fails | Pass |
| LAT-28 | Invalid email and invalid password fails | *Test Case A3*  *“foobar”*  *“pa55w”* | Fails | Fails | Pass |

**Details on LAT-3 /LAT-17/LAT-22/LAT-26 Failure:**

**Issue:**

Inputs were not able to handle null values and produce null value exceptions.

**Solution:**

In addition to checking if a string is empty, additional checks to were added to detect null values.

**Details on LAT-9 Failure**

An email that should be invalid will return valid, however this error lies with the JavaMail API and unable to be fixed as a result.

Re-testing of failed tests

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Data | Expected | Actual | Pass/Fail |
| LAT-3 | Invalid email fails | Test case A3  null | Fails | Fails | Pass |
| LAT-17 | Valid email and invalid password fails | Test Case A3  “foo@bar.com”  null | Fails | Fails | Pass |
| LAT-22 | Invalid email and valid password fails | *Test Case A3*  *Null*  *"@[Pa55w0rd!]"* | Fails | Fails | Pass |
| LAT-26 | Invalid email and invalid password fails | *Test Case A3*  *null*  *null* | Fails | Fails | Pass |

|  |  |
| --- | --- |
| Requirement / Release | Create Account Activity General Functionality |
| Module Name | CreateAccountActivity.java |
| Date Created | 09/01/2019 |

|  |  |
| --- | --- |
| **Test Case** | **Notes** |
| A1 | User does not have existing account |
| A2 | User has existing account |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| CAA-1 | Create account for user | Uses test case A1   1. Enter valid email address 2. Enter valid username 3. Enter valid password 4. Click create account | Account should be added to the authentication database and username added to Firestore document - application should switch to the map activity | Account added to database and application moves to map activity, but username is added to database without indicating which user it belongs to | Fail |
| CAA-2 | Create account for user | Uses test case A2   1. Enter valid email address (which has previously been used to create an account successfully) 2. Enter valid username 3. Enter valid password 4. Click create account | Account will not be created, and user will be informed that authentication has failed and that their email may already be in user | As expected | Pass |
| CAA-3 | Verification email is send to entered address | Uses test case A2   1. Enter valid email, username and password 2. Click create account 3. Check email inbox for entered address | Verification email will be sent to the entered email address, and a message will be displayed to inform the user this has occured | As expected | Pass |
| CAA-4 | Click other sections of the screen except buttons to ensure random clicks do cause error | No test case needed   1. Click areas of screen not including buttons | Nothing | As expected | Pass |
| CAA-5 | Return to previous screen using the back button | No test case needed   1. Click back button in upper left of screen | Application should switch to the start activity | As expected | Pass |

|  |  |
| --- | --- |
| Requirement / Release | Create Account Activity Validation |
| Module Name | CreateAccountActivityTest.java |
| Date Created | 11/01/2019 |
| Notes | Automated Testing using Junit test classes |

|  |  |
| --- | --- |
| **Test Case** | **Notes** |
| A1 | Uses normal data |
| A2 | Uses boundary data |
| A3 | Uses erroneous data |
| A4 | Input contains emoji (Non-traditional data type) |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Data | Expected | Actual | Pass/Fail |
| CAAT-1 | Valid email passes | Test case A1  “foo@bar.com” | Valid | Valid | Pass |
| CAAT-2 | Valid email passes | Test case A2  “foobar123@foobar123.co.uk” | Valid | Valid | Pass |
| CAAT-3 | Invalid email fails | Test case A3  null | Fails | Passes | Fail |
| CAAT-4 | Invalid email fails | Test case A3  “foobar” | Fails | Fails | Pass |
| CAAT-5 | Invalid email fails | Test case A3  "foobar.com" | Fails | Fails | Pass |
| CAAT-6 | Invalid email fails | Test case A3  “@” | Fails | Fails | Pass |
| CAAT-7 | Invalid email fails | Test case A3  “@..” | Fails | Fails | Pass |
| CAAT-8 | Invalid email fails | Test case A3  “@.com” | Fails | Fails | Pass |
| CAAT-9 | Invalid email fails | Test case A3  “a@.com” | Fails | Fails | Pass |
| CAAT-10 | Invalid email fails | Test case A3  “@b.com” | Fails | Fails | Pass |
| CAAT-11 | Invalid email fails | Test case A3  “foobar123@foobar123” | Fails | Passes | Fail |
| CAAT-12 | Valid Email, valid username and valid password passes | Test case A1  "foo@bar.com"  "@U53RN4M3!"  "@[Pa55w0rd!]" | Passes | Passes | Pass |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| CAAT-13 | Valid Email, valid username and valid password passes | Test case A1  "foo@bar.com"  “a"  "pass” | Passes | Fails | Fail |

**Details on CAAT-13 Failure:**

**Issue:** When attempting to create account authentication failed and returned a runtime error as follows:

W/createAccActivity: createUserWithEmailAndPassword:Failure

com.google.firebase.auth.FirebaseAuthWeakPasswordException: The given password is invalid. [Password should be at least 6 characters]

**Solution**: Validation was added to the password field to ensure that the length of the user input is at least 6 characters long but no longer than 20 characters. Validation was also added to the username field, so all usernames must be between 4 and 20 characters long.

Re-test and continuation of testing:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Data | Expected | Actual | Pass/Fail |
| CAAT-13  (altered and retested) | Valid Email, valid username and invalid password fails | Test case A3  "foo@bar.com"  “a"  "pass” | Fails | Passes | Fail |
| CAAT-14 | Valid Email, valid username and valid password passes | Test case A1  "foo@bar.com"  "User1998"  "@@@@@@" | Passes | Passes | Pass |
| CAAT-15 | Valid Email, valid username and valid password passes | Test case A1  "foo@bar.com"  "User \uD83E"  "123456" | Passes | Passes | Pass |
| CAAT-16 | Valid Email, valid username and valid password passes | Test case A1  "foo123@bar123.com"  "user1234567891234567"  "passwo" | Passes | Passes | Pass |
| CAAT-17 | Valid Email, valid username and valid password passes | Test case A1  "foo123@bar123.com"  "1234”  "passwo" | Passes | Passes | Pass |
| CAAT-18 | Valid Email, valid username and valid password passes (both username and password contain emoji) | Test case A1  "foo@bar.com",  "\uDD8A \uD83E\uDD9D \uD83D\uDC3B\uD83D  \uDC3C"  "\uDD8A \uD83E\uDD9D \uD83D\uDC3B\uD83D  \uDC3C" | Passes | Passes | Pass |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| CAAT-19 | Valid Email, valid username and valid password passes (includes quotation marks that may cause error) | Test case A1  "foo123@bar123.com"  "\"\'username"  "passwo\"" | Passes | Passes | Pass |
| CAAT-20 | Invalid email, valid username, valid password fails | Test case A3  null  "user1998"  "@[Pa55w0rd!]" | Fails | Passes | Fail |
| CAAT-21 | Invalid email, valid username, valid password fails | Test case A3  ""  "user1998"  "@[Pa55w0rd!]" | Fails | Fails | Pass |
| CAAT-22 | Invalid email, valid username, valid password fails | Test case A3  "@.com"  "user1998"  "@[Pa55w0rd!]" | Fails | Fails | Pass |
| CAAT-23 | Valid email, invalid username, valid password fails | Test case A3  "foo@bar.com"  null  "@[Pa55w0rd!]" | Fails | Passes | Fail |
| CAAT-24 | Valid email, invalid username, valid password fails | Test case A3  "foo@bar.com"  ""  "@[Pa55w0rd!]" | Fails | Pass | Fail |
| CAAT-25 | Valid email, invalid username, valid password fails | Test case A3  "foo@bar.com"  "use"  "@[Pa55w0rd!]" | Fails | Passes | Fail |
| CAAT-26 | Valid email, invalid username, valid password fails | Test case A3  "foo@bar.com",  "user12345678912345678"  "@[Pa55w0rd!]" | Fails | Passes | Fail |
| CAAT-27 | Valid email, valid username, invalid password fails | Test case A3  "foo@bar.com"  "@U53RN4M3!"  null | Fails | Passes | Fail |
| CAAT-28 | Valid email, valid username, invalid password fails | Test case A3  "foo@bar.com"  "@U53RN4M3!",  "" | Fails | Fails | Pass |
| CAAT-29 | Valid email, valid username, invalid password fails | Test case A3  "foo@bar.com"  "@U53RN4M3!",  "passw" | Fails | Passes | Fail |
| CAAT-30 | Valid email, valid username, invalid password fails | Test case A3  "foo@bar.com"  "@U53RN4M3!",  "123456789123456789123" | Fails | Passes | Fail |
| CAAT-31 | Invalid email, invalid username, invalid password fails | Test case A3  null  null  null | Fails | Passes | Fail |
| CAAT-32 | Invalid email, invalid username, invalid password fails | Test case A3  ""  ""  "" | Fails | Passes | Fail |

**Details on CAAT-3/CAAT-20/CAAT-23/CAAT-27/CAAT-31 Failure:**

**Issue:**

Inputs were not able to handle null values and produce null value exceptions.

**Solution:**

In addition to checking if a string is empty, additional checks to were added to detect null values.

**Details on CAAT-11 Failure:**

**Issue:**

An email that should be invalid will return valid, however this error lies with the JavaMail API and unable to be fixed as a result.

**Details on CAAT-13 CAAT-25/CAAT-29/CAAT-30 Failure:**

**Issue:**

A line of code was missing from the section of the validation method which checks the password is between the minimum and maximum permitted lengths. The missing line was to return that the validation had failed, in its absence it was instead returned that the validation had succeeded

**Details on CAAT-24/CAAT-32 Failure:**

**Issue:**

A line of code was missing from the section of the validation method which checks if the username is null or empty. The missing line was to return that the validation had failed, in its absence it was instead returned that the validation had succeeded

**Solution:**

Missing line was added to validation code.

Re-testing:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Data | Expected | Actual | Pass/Fail |
| CAAT-3.2 | Invalid email fails | Test case A3  null | Fails | Fail | Pass |
| CAAT-13.3 | Valid Email, valid username and invalid password fails | Test case A3  "foo@bar.com"  “a"  "pass” | Fails | Fail | Pass |
| CAAT-20.2 | Invalid email, valid username, valid password fails | Test case A3  null  "user1998"  "@[Pa55w0rd!]" | Fails | Fail | Pass |
| CAAT-23.2 | Valid email, invalid username, valid password fails | Test case A3  "foo@bar.com"  null  "@[Pa55w0rd!]" | Fails | Fail | Pass |
| CAAT-24.2 | Valid email, invalid username, valid password fails | Test case A3  "foo@bar.com"  ""  "@[Pa55w0rd!]" | Fails | Fails | Pass |
| CAAT-25.2 | Valid email, invalid username, valid password fails | Test case A3  "foo@bar.com"  "use"  "@[Pa55w0rd!]" | Fails | Fails | Pass |
| CAAT-27.2 | Valid email, valid username, invalid password fails | Test case A3  "foo@bar.com"  "@U53RN4M3!"  null | Fails | Fail | Pass |
| CAAT-29.2 | Valid email, valid username, invalid password fails | Test case A3  "foo@bar.com"  "@U53RN4M3!",  "passw" | Fails | Fail | Pass |
| CAAT-30.2 | Valid email, valid username, invalid password fails | Test case A3  "foo@bar.com"  "@U53RN4M3!",  "123456789123456789123" | Fails | Fail | Pass |
| CAAT-31.2 | Invalid email, invalid username, invalid password fails | Test case A3  null  null  null | Fails | Fail | Pass |

|  |  |
| --- | --- |
| Requirement / Release | Change password functionality |
| Module Name | ChangePasswordActivity.java |
| Date Created | 14/01/2019 |

|  |  |
| --- | --- |
| **Test Case** | **Notes** |
| N/A | N/A |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| CPA-1 | Update user password | 1. Enter correct current password 2. Enter a new password, different to the current one 3. Click change password | User password should be updated in database | As expected | Pass |
| CPA-2 | Update user password | 1. Enter incorrect current password 2. Enter a new password that differs to current password 3. Click change password | Error message should inform user that current password is incorrect | As expected however progress bar appears and spins continuously | Fail |
| CPA-3 | Prevent change of password where current password is equal to new password | 1. Enter correct current password 2. Enter identical current password into the new password field 3. Click change password | Password should not be changed | Password is changed to be identical to current password | Fail |
| CPA-4 | Ensure random clicks do not cause error | 1. Click all sections of the touchscreen except those containing buttons | Nothing | Nothing | Pass |
| CPA-5 | Go back to previous screen | 1. Click back button in upper left of screen | Application should go back to the account activity screen | As expected | Pass |

**Details on CPA-2 failure:**

**Issue:**

Progress bar would appear and continuously spin, due to a missing line of code to set it to be invisible when validation returns a false result.

**Solution:**

Added missing code.

**Details on CPA-3 Failure:**

**Issue:**

The system allows for users to enter their current password as their new password, which would lead to a meaningless transfer of data between the application and the Firebase database – while this will not produce an error it is an ineffective use of computational resources and power for both the mobile device and the server. To ensure that the database is only used when needed, to prevent unnecessary strain on the server, this should be prevented.

**Solution:**

An additional check was added to the input validation to ensure the passwords are not identical before attempting to make a connection with the Firebase server.

Re-test:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| CPA-3 | Prevent change of password where current password is equal to new password | 1. Enter correct current password 2. Enter identical current password into the new password field 3. Click change password | Error message should inform user new password cannot equal current password – password should not be changed | As expected | Pass |

|  |  |
| --- | --- |
| Requirement / Release | Change password validation |
| Module Name | ChangePasswordActivityTest.java |
| Date Created | 15/01/2019 |
| Notes | Automated Testing using Junit test classes |

|  |  |
| --- | --- |
| **Test Case** | **Notes** |
| A1 | Uses normal data |
| A2 | Uses boundary data |
| A3 | Uses erroneous data |
| A4 | Input contains emoji (Non-traditional data type) |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Data | Expected | Actual | Pass/Fail |
| CPAT-1 | Valid current password and valid new password passes | Test case A1  "@PA55w0rd!", "@PA55w0rd!2" | Passes | Passes | Pass |
| CPAT-2 | Valid current password and valid new password passes | Test case A1/A2  "passw1”,  "passw2" | Passes | Passes | Pass |
| CPAT-3 | Valid current password and valid new password passes | Test case A1/A4  "\uDD8A \uD83E\uDD9D \uD83D\uDC3B\uD83D\uDC3C",  "\uDD8A \uD83E\uDD9D \uD83D\uDC3B \uD83D" | Passes | Passes | Pass |
| CPAT-4 | Valid current password and valid new password passes (inputs contain quotations which may cause error) | Test case A1  "\"\"password\"\"", "\"\"password2\"\"" | Passes | Passes | Pass |
| CPAT-5 | Invalid current password and valid new password fails | Test case A3  null,  "@PA55w0rd!2" | Fails | Fail | Pass |
| CPAT-6 | Invalid current password and valid new password fails | Test case A3  "",  "@PA55w0rd!2" | Fails | Fail | Pass |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| CPAT-7 | Invalid current password and valid new password fails | Test case A3  "12345",  "@PA55w0rd!2" | Fails | Pass | Fails |
| CPAT-8 | Invalid current password and valid new password fails | Test case A3  "123456789123456789123", "@PA55w0rd!2" | Fails | Pass | Fails |
| CPAT-9 | Valid current password and invalid new password fails | Test case A3  "@PA55w0rd!",  null | Fails | Fail | Pass |
| CPAT-10 | Valid current password and invalid new password fails | Test case A3  "@PA55w0rd!",  "" | Fails | Fail | Pass |
| CPAT-11 | Valid current password and invalid new password fails | Test case A3  "@PA55w0rd!",  "12345" | Fails | Pass | Fails |
| CPAT-12 | Valid current password and invalid new password fails | Test case A3  "@PA55w0rd!",  "123456789123456789123" | Fails | Pass | Fails |
| CPAT-13 | Invalid current password and invalid new password fails | Test case A3  null,  "" | Fails | Fail | Pass |
| CPAT-14 | Invalid current password and invalid new password fails | Test case A3  "",  null | Fails | Fail | Pass |
| CPAT-15 | Invalid current password and invalid new password fails | Test case A3  "12345",  "12345" | Fails | Pass | Fails |
| CPAT-16 | Invalid current password and invalid new password fails | Test case A3  "123456789123456789123",  "123456789123456789123" | Fails | Pass | Fails |
| CPAT-17 | New password does not equal current password | Test Case A1  "@PA55w0rd!",  "@PA55w0rd!" | Fails | Fail | Pass |

**Details on CPAT-7/CPAT-8/CPAT-11/ CPAT-12/CPAT-15/CPAT-16 Failure:**

**Issue:**

A missing return statement in the section of the validation code responsible for ensuring password length is between six and twenty was missing, as a result passwords outside of this range would falsely be returned as valid.

**Solution:**

Added missing line of code.

Re-test:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Data | Expected | Actual | Pass/Fail |
| CPAT-7.2 | Invalid current password and valid new password fails | Test case A3  "12345",  "@PA55w0rd!2" | Fail | Fail | Pass |
| CPAT-8.2 | Invalid current password and valid new password fails | Test case A3  "123456789123456789123", "@PA55w0rd!2" | Fail | Fail | Pass |
| CPAT-11.2 | Valid current password and invalid new password fails | Test case A3  "@PA55w0rd!",  "12345" | Fail | Fail | Pass |
| CPAT-12.2 | Valid current password and invalid new password fails | Test case A3  "@PA55w0rd!",  "123456789123456789123" | Fail | Fail | Pass |
| CPAT-15.2 | Invalid current password and invalid new password fails | Test case A3  "12345",  "12345" | Fail | Fail | Pass |
| CPAT-16.2 | Invalid current password and invalid new password fails | Test case A3  "123456789123456789123",  "123456789123456789123" | Fail | Fail | Pass |

|  |  |
| --- | --- |
| Requirement / Release | New Marker Activity General Functionality |
| Module Name | NewMarkerActivity.java |
| Date Created | 23/01/2019 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| NMA-1 | Populate spinner with category string | 1. Click spinner 2. Ensure all relevant categories are present | All categories should be present | Category ‘Misc’ is missing from spinner | Fail |
| NMA-2 | Add marker to database with correct data | 1. Enter valid marker details 2. Click the add marker button | Marker will be added to database with all correct details | Marker is added to database but username of the user who placed marker is missing | Fail |
| NMA-3 | Ensure random clicks do not cause error | 1. Click all sections of the touchscreen except those containing buttons | Nothing | Nothing | Pass |
| NMA-4 | Go back to previous screen | 1. Click back button in upper left of screen | Application should go back to the map activity screen | As expected | Pass |

**Details on NMA-1 Failure:**

**Issue:**

The code that handles the miscellaneous category is missing from the spinner.

**Solution:**

Missing code was added to array containing the category strings.

**Details on NMA-2 failure:**

**Issue:**

After placing a marker, the marker will not show the user who placed it. Upon using logs to check the value returned by:

FirebaseAuth.*getInstance*().getCurrentUser()

It was observed that the resulting string was not the email address, as was initially presumed, but instead an assortment of letters and numbers.

**The solution:**

The get current user code was amended to:

FirebaseAuth.getInstance().getCurrentUser().getEmail()

This code specifically gets the email address of the current user.

However, despite the correct email now being returned and used to locate the corresponding username successfully, as shown in the snippet from the runtime log shown below, the username would still not be correctly added to the database.

D/NewMarkerActivity: getUsername: current user: mike\_buglass@hotmail.co.uk

D/NewMarkerActivity: createMarker:Inserting

D/TAG: Document Id: PS7oOJviZUKg2DIProGT

D/NewMarkerActivity: getUsername: Found: michael

However, it was not possible to return this username from the on-complete listener method in which it was found. As a workaround, a new text view was added to the act\_newmarker.xml file to store username located by the method, as writing to components is possible from the on complete listener. This value can then be read by other methods allowing the value to be used when adding the marker information to the database.

Re-test:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| NMA-1.2 | Populate spinner with category string | 1. Click spinner 2. Ensure all relevant categories are present | All categories should be present | As expected | Pass |
| NMA-2.2 | Add marker to database with correct data | 1. Enter valid marker details 2. Click the add marker button | Marker will be added to database with all correct details | As expected | Pass |

|  |  |
| --- | --- |
| Requirement / Release | New Marker Activity Validation |
| Module Name | NweMarkerActivityTest.java |
| Date Created | 28/01/2019 |
| Notes | Automated Testing using Junit test classes |

|  |  |
| --- | --- |
| **Test Case** | **Notes** |
| A1 | Uses normal data |
| A2 | Uses boundary data |
| A3 | Uses erroneous data |
| A4 | Input contains emoji (Non-traditional data type) |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Data | Expected | Actual | Pass/Fail |
| NMAT-1 | Valid title and valid description pass | Test case A1  "Example Title",  "Example Description" | Passes | Passes | Pass |
| NMAT-2 | Valid title and valid description pass | Test case A1  "!\"£$%^&\*()-=#'@`", "!\"£$%^&\*()-=#'@`" | Passes | Passes | Pass |
| NMAT-3 | Valid title and valid description pass | Test case A1/A4  "\uD83E\uDD9D ", "\uD83E\uDD9D " | Passes | Passes | Pass |
| NMAT-4 | Invalid title and valid description fail | Test Case A3  null,  "Example Description" | Fail | Fail | Pass |
| NMAT-5 | Invalid title and valid description fail | Test Case A3  "",  "Example Description" | Fails | Pass | Fails |
| NMAT-6 | Valid title and invalid description fail | Test Case A3  "Example Title",  null | Fail | Fail | Pass |
| NMAT-7 | Valid title and invalid description fail | Test Case A3  "Example Title",  "" | Fails | Pass | Fails |
| NMAT-8 | Invalid title and invalid description fail | Test Case A3  null,  null | Fail | Fail | Pass |
| NMAT-9 | Invalid title and invalid description fail | Test Case A3  "",  "" | Fails | Pass | Fails |

**Details on NMAT-5/NMAT-7/NMAT-9 failure:**

**Issue:**

The empty string test produces a failure due to a known compatibility issue with the jUnit and TextUtils libraries which causes false negatives in the validation method. "We are aware that the default behaviour is problematic when using classes like Log or TextUtils and will evaluate possible solutions in future releases." - Tools.android.com. (2019). *Unit testing support - Android Studio Project Site*. [online] Available at: http://tools.android.com/tech-docs/unit-testing-support#TOC-Method-...-not-mocked.- [Accessed 28 Jan. 2019].

Re-test [Performed 10/02/2019]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Data | Expected | Actual | Pass/Fail |
| NMAT-5 | Invalid title and valid description fail | Test Case A3  "",  "Example Description" | Fails | Fails | Pass |
| NMAT-7 | Valid title and invalid description fail | Test Case A3  "Example Title",  "" | Fails | Fails | Pass |
| NMAT-9 | Invalid title and invalid description fail | Test Case A3  "",  "" | Fails | Fails | Pass |

|  |  |
| --- | --- |
| Requirement / Release | Maps Activity Map Component |
| Module Name | MapsActivity.java |
| Date Created | 03/01/2019 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| MAC-1 | Map displays correctly | 1. Open map activity and observe the map | Map will display correctly | Map gives API key error | Fail |
| MAC-1.2 | Map displays correctly | 1. Open map activity and observe the map | Map will display correctly | Map API not active in console error | Fail |
| MAC-1.3 | Map displays correctly | 1. Open map activity and observe the map | Map will display correctly | As expected | Pass |

|  |  |
| --- | --- |
| Requirement / Release | Maps Activity General Functionality |
| Module Name | MapsActivity.java |
| Date Created | 05/02/2019 |

|  |  |
| --- | --- |
| **Test Case** | **Notes** |
| A1 | Map is on a view other than normal |
| A2 | Map is on a view other than hybrid |
| A3 | Map is on a view other than satellite |
| A4 | Map is on a view other than terrain |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| MA-1 | Use back button to go back to login | 1. Click back button in upper left of screen | Application screen will switch to Login Activity | As Expected | Pass |
| MA-2 | Filter button functionality | 1. Click filter button | Filter button will appear on toolbar at top of screen and take user to filter activity | Filter button does not appear | Fail |
| MA-3 | Check that map starts at the user’s current location | 1. Open maps activity 2. Observe default location presented by map and compare to actual location of device | Map should start focused on the user’s current location | Map does not focus on user location | Fail |
| MA-4 | Menu button appears with correct | 1. Click menu button 2. Check all values are correct | Menu will show all options | As expected | Pass |
| MA-5 | Go to account screen | 1. Click menu button 2. Click ‘Account’ option | Application will go to the account screen | As expected | Pass |
| MA-6 | Go to setting screen | 1. Click menu button 2. Click ‘Settings’ option | Application will go to the setting screen | As expected | Pass |
| MA-7 | Switch map to normal view | Test case A1   1. Click meu button 2. Click ‘Normal’ option | Map will switch to normal view | As expected | Pass |
| MA-8 | Switch map to hybrid view | Test case A1   1. Click meu button 2. Click ‘Hybrid’ option | Map will switch to hybrid view | As expected | Pass |
| MA-9 | Switch map to satellite view | Test case A1   1. Click meu button 2. Click ‘Satellite’ option | Map will switch to satellite view | As expected | Pass |
| MA-10 | Switch map to terrain view | Test case A1   1. Click meu button 2. Click ‘Terrain’ option | Map will switch to terrain view | As expected | Pass |
| MA-11 | Markers appear on map | 1. Locate markers on map 2. Compare to marker in database to ensure all are present | All markers should appear on map | As expected | Pass |
| MA-12 | View marker details | 1. Locate marker on map 2. Click marker | Application should switch to view marker activity | Application crashed | Fail |
| MA-13 | Clicking the ‘geo-locate’ button focuses on the user’s current location | 1. Click ‘geo-locate’ button in top right of map 2. Observe location change on map | Application should move the map view to focus on the devices current GPS coordinates | As expected | Pass |
| MA-14 | Zoom in and out of map using button | 1. Click the plus and minus buttons on the bottom right of the map 2. Observe change of scale of map | Application should zoom in when plus button pressed; zoom out when minus button pressed | As expected | Pass |

**Details on MA-2 failure:**

**Issue:**

The filter button would not appear on the toolbar and no fix was able to be found for this issue.

**Solution:**

Rather than being a separate button, the filter button was instead added to the drop down list of the menu.

**Details on MA-3 failure:**

**Issue:**

Code used for moving camera to user’s location is deprecated and no longer functions as intended.

**Solution:** The code was updated to use newer methods.

**Details on MA-12 Failure:**

**Issue:**

Upon opening view marker activity, the following error message would be produced by the runtime console:

java.lang.NullPointerException: Attempt to invoke virtual method 'android.view.Window$Callback android.view.Window.getCallback()' on a null object reference

This issue was caused by declaring variables before initialising all components.

**Solution:**

Declerations were moved in the OnCreate method.

Re-testing:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| MA-2 (Altered and retested) | Filter button functionality | 1. Click menu button 2. Select ‘Filter’ option | Filter button will appear in the menu, when clicked takes user to filter activity | As expected | Pass |
| MA-3 | Check that map starts at the user’s current location | 1. Open maps activity 2. Observe default location presented by map and compare to actual location of device | Map should start focused on the user’s current location | As expected | Pass |
| MA-12 | View marker details | 1. Locate marker on map 2. Click marker | Application should switch to view marker activity | As expected | Pass |

|  |  |
| --- | --- |
| Requirement / Release | Maps Activity Search Functionality |
| Module Name | MapsActivity.java |
| Date Created | 12/02/2019 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| MAS-1 | Search-bar autocomplete | 1. Enter location into search bar 2. Observe autocomplete suggestion | When typing, relevant autocomplete suggestions should appear | No suggestions appear | Fail |
| MAS-2 | Camera moves to searched location | 1. Enter location into search bar 2. Click suggestion or click enter | Camera should move to location entered | Camera generally moves to location entered unless input is too vague | Pass |

**Details on MAS-1 failure:**

**Issue:**

The method used to implement the autocomplete using Google Places API has been deprecated by Google as of 29th January 2019, as the Google Places for Android API has been discontinued.

**Solution:**

The new Google Places API, which succeeds the ‘for Android’ version, was implemented and a similar method was recreated using the powered by google search bar.

Re-test:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| MAS-1.2 | Search-bar autocomplete | 1. Enter location into search bar 2. Observe autocomplete suggestion | When typing, relevant autocomplete suggestions should appear | Powered by google search bar appears, immediately encounters an error and closes. | Fail |

**Details on MAS-1.2 failure:**

**Issue:**

The method used does not work with the new API and causes the powered by Google search bar to crash – no alternative implementation can be found at the time of development as learning materials have not been updated to match the relatively new API.

**Solution:**

Rather than rely on an API to populate text fields, a list of major global cities with populations greater than 15,000 was acquired from the website: <https://datahub.io/core/world-cities>.

The acquired data is licensed under the creative common attribution licence and allows the data to be adapted for use in different applications including commercial, and distributed in any medium or format providing credit is given to geonames ([www.geonames.org/](http://www.geonames.org/)) as the original source of this information. While no credit is formally required, Alexandre Bonnasseau, aka ‘Lexman’ and the Open Knowledge Foundation (https://okfn.org) are also responsible, in part, for the provision of this data.

All data except from city names was removed from the world-cities.csv file and the data was then moved into the file cities.txt. This file is then read by MapsActivity.java into a list of strings, which is then added to the auto complete text view as autocomplete results.

Re-test:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| MAS-1.3 | Search-bar autocomplete | 1. Enter location into search bar 2. Observe autocomplete suggestion | When typing, relevant autocomplete suggestions should appear | Suggestions do not show | Fail |

**Details on MAS-1.3 failure:**

**Issue:**

Code responsible for reading text from the text file does no work as expected – file is not read.

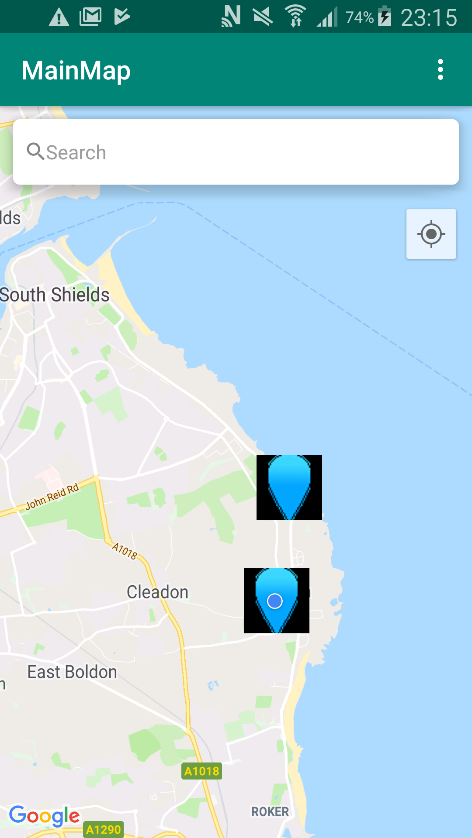
**Solution:**

Rather than using a file reader and file system methods to read text, a buffered reader will be used instead.

Re-test:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | | Testing Steps | Expected | Actual | Pass/Fail |
| MAS-1.4 | Search-bar autocomplete | | 1. Enter location into search bar 2. Observe autocomplete suggestion | When typing, relevant autocomplete suggestions should appear | As expected | Pass |
| Requirement / Release | | Maps Activity Marker Colour Functionality | | | | |
| Module Name | | MapsActivity.java | | | | |
| Date Created | | 19/02/2019 | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| MAMC-1 | Display custom bitmap image in place of default marker | 1. Observe how the bitmap image appears on map | Bitmap image should appear at the same location as the default marker at a reasonable size. | Marker does not point to the correct location and does not appear correctly | Fail |

**Details on MAMC-1 Failure:**

**Issue:**

Custom markers require the Bitmap extension type, which generally does not support the alpha channel in most software. Even when producing a version of the marker that does use an alpha channel to hide the background, using this resource in Android Studio will produce the black background regardless.

Markers also do not stem from the marked location, as the coordinates of the marked location will be the centre of the square image, which leads to less precise location marking.

**The solution:**

Instead of using custom images to change the colour of markers, code was implemented to change the colour of the default Google Maps API marker to different colours to indicate category.

This means marker style will be tied to the API, meaning that the most up to date style should always be used allowing the app to keep modern appearance with less maintenance.

Markers also stem from the relevant coordinates making the marking more precise.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| MAMC-1.2 (Altered and retested) | Markers display in different colours depending on category type | 1. Observe colour of marker on map and compare to expected colour for each category | Markers should show in the correct colour | As expected | Pass |

Re-test:

|  |  |
| --- | --- |
| Requirement / Release | Maps Activity Add Marker Functionality |
| Module Name | MapsActivity.java |
| Date Created | 27/02/2019 |

|  |  |
| --- | --- |
| **Test Case** | **Notes** |
| A1 | ‘New marker’ marker not placed |
| A2 | ‘New marker’ marker already placed |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| MAAM-1 | Holding down on section of map to place marker | Test case A1   1. Long press on the map | A marker should be placed at the pressed location | As expected | Pass |
| MAAM-2 | Holding down on section of map to place marker | Test case A2   1. Long press on the map | Old marker should vanish and new one should be placed at pressed location | As expected | Pass |
| MAAM-3 | Clicking placed marker to add to map | Test case A1   1. Long press on the map 2. Click the purple marker that appears | Application should switch to the new marker activity screen | As expected | Pass |

|  |  |
| --- | --- |
| Requirement / Release | Filter Activity Functionality |
| Module Name | FilterActivity.java |
| Date Created | 29/02/2019 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| FA-1 | Filter buttons appear for each category | 1. Go to filter activity 2. Observe and compare visible buttons to expected categories | All categories should be present | The ‘Misc’ category is not present | Fail |
| FA-2 | Filters should change which markers appear on the map | 1. Select a random combination of filters 2. Go back to map activity using back button 3. Observe changes in the visible markers | All markers that fall under selected categories should appear, those that fall under categories not selected should not appear | Markers belonging to the ‘Misc’ category do not appear | Fail |
| FA-3 | Click other sections of the screen except buttons to ensure random clicks do cause error | 1. Click areas of screen not including buttons | Nothing | As expected | Pass |
| FA-4 | Return to previous screen using the back button | 1. Click back button in upper left of screen | Application should switch to the maps activity | As expected | Pass |

**Details on FA-1/FA-2 Failure:**

**Issue:**

Code responsible for adding the filter switch for the ‘Misc’ category to the filter activity was missing from the file filter\_preferences.xml. The lack of this filter leads to the markers falling under the category to not be added to the map.

**Solution:**  
Missing code was added to the filter\_preferences.xml file.

Re-testing:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| FA-1.2 | Filter buttons appear for each category | 1. Go to filter activity 2. Observe and compare visible buttons to expected categories | All categories should be present | As expected | Pass |
| FA-2.2 | Filters should change which markers appear on the map | 1. Select a random combination of filters 2. Go back to map activity using back button 3. Observe changes in the visible markers | All markers that fall under selected categories should appear, those that fall under categories not selected should not appear | Markers belonging to the ‘Misc’ category do not appear | Fail |

**Details on FA-2.2 Failure:**

**Issue:**

A spelling error in the implementation of the filter switch for the ‘Misc’ category meant the program would always assume the switch was false and the markers would not be added to the map.

**Solution:**

The mis spelled code was amended.

Re-test:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| FA-2.3 | Filters should change which markers appear on the map | 1. Select a random combination of filters 2. Go back to map activity using back button 3. Observe changes in the visible markers | All markers that fall under selected categories should appear, those that fall under categories not selected should not appear | As expected | Pass |

|  |  |
| --- | --- |
| Requirement / Release | View Marker Activity Functionality |
| Module Name | ViewMarkerActivity.java |
| Date Created | 11/03/2019 |

|  |  |
| --- | --- |
| **Test Case** | **Notes** |
| A1 | Marker placed by current user |
| A2 | Marker placed by different user |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| VMA-1 | Marker details are correct | Test case N/A   1. Compare marker details to those in database | Marker screen should contain correct details | As expected | Pass |
| VMA-2 | Delete marker appears for user who placed marker | Test case A1   1. Compare user who placed marker to logged in user 2. Check if button is visible | Button should not be visible | As expected | Pass |
| VMA-3 | Delete marker doesn’t show for users who did not place marker | Test case A2   1. Compare user who placed marker to logged in user 2. Check if button is visible | Button should be visible | As expected | Pass |
| VMA-4 | Delete marker | Test case A2   1. Click delete marker button 2. Check database to ensure marker was deleted | Marker should be removed from database | As expected | Pass |
| VMA-5 | Click other sections of the screen except buttons to ensure random clicks do cause error | 1. Click areas of screen not including buttons | Nothing | As expected | Pass |
| VMA-6 | Return to previous screen using the back button | 1. Click back button in upper left of screen | Application should switch to the map activity | As expected | Pass |

|  |  |
| --- | --- |
| Requirement / Release | Settings Activity Functionality |
| Module Name | SettingsActivity.java |
| Date Created | 15/03/2019 |

|  |  |
| --- | --- |
| **Test Case** | **Notes** |
| A1 | Dark mode active |
| A2 | Dark mode not active |
| A3 | Labels active |
| A4 | Labels not active |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| SET-1 | Turn on dark mode | Test case A2   1. Click the dark mode setting into the on position 2. Return to map | Map should now be in dark mode | As expected | Pass |
| SET-2 | Turn off dark mode | Test case A1   1. Click the dark mode setting into the off position 2. Return to map | Map should be in the regular colour scheme | Map still in dark mode | Fail |
| SET-3 | Turn off labels | Test case A1/A3   1. Click the labels setting into the off position 2. Return to map | Map should be dark and have no labels | As expected | Pass |
| SET-4 | Turn off labels | Test case A2/A3   1. Click the labels setting into the off position 2. Return to map | Map should be normal colours and no labels | As expected | Pass |
| SET-5 | Turn on labels | Test case A1/A4   1. Click the labels setting into the off position 2. Return to map | Map should be dark and have labels | As expected | Pass |
| SET-6 | Turn on labels | Test case A2/A4   1. Click the labels setting into the off position 2. Return to map | Map should be normal colours with labels | Labels do not reappear | Fail |
| SET-7 | Click other sections of the screen except buttons to ensure random clicks do cause error | 1. Click areas of screen not including buttons | Nothing | As expected | Pass |
| SET-8 | Return to previous screen using the back button | 1. Click back button in upper left of screen | Application should switch to the map activity | As expected | Pass |

**Details on SET-2/SET-6 failure:**

**Issue:**

When instead of returning to default, in some cases the map would remain the same. This was caused by the map not resetting in some instances.

**Solution:**

A fourth json file containing code for the default view was added and is called to reset the map when necessary.

Re-testing:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| SET-2.2 | Turn off dark mode | Test case A1   1. Click the dark mode setting into the off position 2. Return to map | Map should be in the regular colour scheme | Map still in darkmode | Fail |
| SET-6.2 | Turn on labels | Test case A2/A4   1. Click the labels setting into the off position 2. Return to map | Map should be normal colours with labels | Labels do not reappear | Fail |

**Details on SET-2.2/SET-6.2 failure:**

**Issue:**

Logical error with the if statements was the cause of issue, where the or operator was used instead of the and operator. This was the actual cause of previous attempts of these tests failing.

**Solution:**

Deciding logic was changed to use the correct operators.

Re-testing:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| SET-2.3 | Turn off dark mode | Test case A1   1. Click the dark mode setting into the off position 2. Return to map | Map should be in the regular colour scheme | As expected | Pass |
| SET-6.3 | Turn on labels | Test case A2/A4   1. Click the labels setting into the off position 2. Return to map | Map should be normal colours with labels | As expected | Pass |

|  |  |
| --- | --- |
| Requirement / Release | Account Activity Functionality |
| Module Name | AccountActivity.java |
| Date Created | 18/03/2019 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| AA-1 | View user details | 1. Compare shown details to those stored in database | Database should show correct details | As expected | Pass |
| AA-2 | GUI colour scheme | 1. Observe colour scheme of activity | Colours should match colour scheme | Change password button is the wrong colour | Fail |
| AA-3 | Change password button | 1. Click the change password button | Application should change to change password activity | As expected | Pass |
| AA-4 | Click other sections of the screen except buttons to ensure random clicks do cause error | 1. Click areas of screen not including buttons | Nothing | As expected | Pass |
| AA-5 | Return to previous screen using the back button | 1. Click back button in upper left of screen | Application should switch to the map activity | As expected | Pass |

**Details on AA-2 failure:**

**Issue:**  Code responsible for changing button colour is missing.

**Solution:** Missing code was added.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Num. | Purpose | Testing Steps | Expected | Actual | Pass/Fail |
| AA-2.2 | GUI colour scheme | 1. Observe colour scheme of activity | Colours should match colour scheme | As expected | Pass |

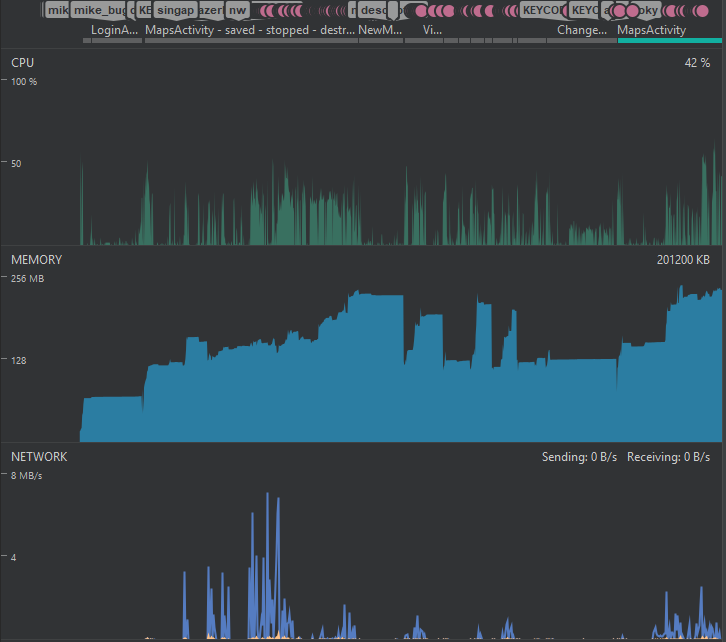
|  |  |
| --- | --- |
| Requirement / Release | XML File validation |
| Date Created | 20/03/2019 |
| Notes | Validation of files using W3 Schools |

|  |  |  |
| --- | --- | --- |
| **Test Num.** | **File** | **Pass/Fail** |
| XVAL-1 | act\_account.xml | Pass |
| XVAL-2 | act\_changepassword.xml | Pass |
| XVAL-3 | act\_createaccount.xml | Pass |
| XVAL-4 | act\_login.xml | Pass |
| XVAL-5 | act\_newmarker.xml | Pass |
| XVAL-6 | act\_start.xml | Pass |
| XVAL-7 | act\_viewmarker.xml | Pass |
| XVAL-8 | act\_maps.xml | Pass |
| XVAL-9 | map\_options.xml | Pass |
| XVAL-10 | colours.xml | Pass |
| XVAL-11 | string.xml | Pass |
| XVAL-12 | styles.xml | Pass |
| XVAL-13 | filter\_preferences.xml | Pass |
| XVAL-14 | settings\_preferences.xml | Pass |
| XVAL-15 | AndroidManifest.xml | Pass |
| XVAL-16 | google\_maps\_api.xml | Pass |

|  |  |
| --- | --- |
| Requirement / Release | JSON File validation |
| Date Created | 20/03/2019 |
| Notes | Validation of files using jsonlint.com |

|  |  |  |
| --- | --- | --- |
| **Test-Num.** | **File** | **Pass/Fail** |
| JVAL-1 | darkmode\_json.json | Pass |
| JVAL-2 | darkmodenolabels\_jason.json | Pass |
| JVAL-3 | normal\_jason.json | Pass |
| JVAL-4 | normalnolabels\_json.json | Pass |
| JVAL-5 | google-services.json | Pass |

|  |  |
| --- | --- |
| Requirement / Release | Application performance test |
| Date Created | 21/03/2019 |
| Notes | Performance test using CPU profiler in Android Studio |

****

The above graph shows performance statistics for a five-minute test of the application performing various activities including log in, adding markers, viewing markers, using the search function and changing settings.

Peak memory usage is 202.4MB and occurs when browsing the map, likely due to the large amount of data and visuals used.

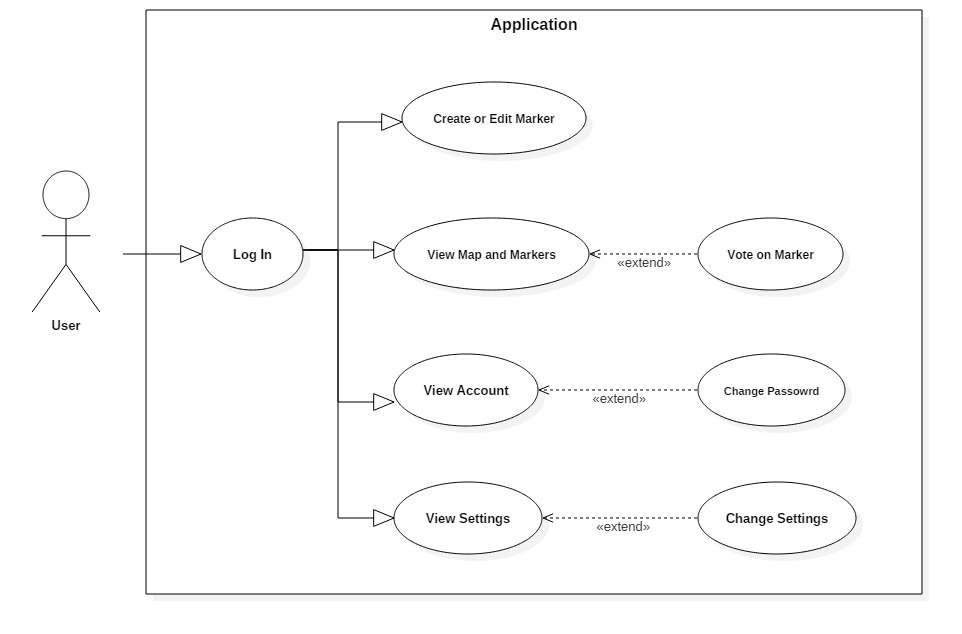
Peak incoming network usage was 4.8MB/s when browsing the map.

Peak outgoing network usage was 0.2MB/s when browsing the map.

Peak CPU Usage was 64.9% when browsing the map.

From this information is can be deduced that the task which requires the most computational resources is the is the maps activity, specifically the Google map component.

## K) Use Case Diagram and Descriptions



**Figure 2: Use Case Diagram for Proposed Application**

The above diagram visually outlines the key use-cases that will be present throughout the system. For reasons of security and the prevention of malicious activity, all functionalities require the user to create an account and sign in as a prerequisite which serves as a deterrent.

Use Case: Place Marker

Use Case Summary:

Actor(s): User

Prerequisite:

1. User logs in to the application

Prime Scenario:

1. Add marker button is pressed (Alternative: **User edits existing marker**)
2. User will be prompted to select where, on the map, they wish to add the marker
3. User will click desired location (Exception: **User wishes to cancel the placement of marker**)
4. UI will change to a screen containing several labelled text boxes, indicating what must be entered each. (Exception: **User decides not to place the marker**)
5. Information about the marker is entered by the user, including name, category, and description (Exception: **User leaves an input field blank or uses invalid characters**)
6. Data is confirmed, and the marker is added to the database
7. The user is returned to the map UI with their new marker now visible

Alternative Scenario:

**User edits existing marker:** (the user will not be able to edit or delete markers placed by another user, only ones they have placed themselves.)

1. The user will locate and press a marker they have made
2. The UI will display un-editable text boxes containing information about the marker
3. User will click ‘Edit Marker’ button when viewing a marker that they have created (Alternative: **User wishes to delete marker**)
4. Text boxes displaying marker information will become editable
5. The user will amend any information they wish to change
6. The user will click the ‘Submit’ button after editing their marker (Exception: **User leaves an input field blank or uses invalid characters**)
7. Information will update in the database and the text boxes will become un-editable

**User wishes to delete marker**:

1. The user will click the ‘Delete Marker’ button on the marker information screen
2. A confirmation message will appear asking if the user is sure they wish to delete their marker
3. The user will click yes (Alternative: **User decides not to delete** **marker**)
4. The marker will be removed from the database
5. The user will be returned to the default map UI screen, with their marker now gone

**User Decides not to delete marker**

1. The user will click no when prompted to delete their marker
2. The user will be returned to the marker information screen

Exceptional Scenario:

**User leaves an input blank or enters an invalid character**:

1. A message will appear to inform the user that all fields must be filled in and that invalid characters may be present
2. The user can dismiss this message and return to account creation
3. The user will now amend input and try again

**User wishes to cancel the placement of marker:**

1. User presses cancel button in the top left of the screen
2. User is returned to the default map UI

**User decides not to place the marker:**

1. User presses the back button in the top left of the screen
2. User is returned to default map UI

Use Case: View Map/Markers

Actor(s): User

Prerequisite:

1. User logs in to application

Prime Scenario:

1. User will press marker of interest on the map (Exception: **No markers of interest are present in the user’s area**)
2. Application will display text boxes that show information about the marker in a pop-up window on the screen, which can be closed when the user is done reading (Alternative: **User votes on marker)**
3. The user will be returned to the default map UI

Alternative Scenario:

**User votes on marker:** If the user deems a marker to be helpful, or unhelpful, they can vote on the marker to contribute to its score, which can be seen by other users and contributes to a user’s overall score. (Exception: **The user created the marker**)

1. User clicks on either the up arrow or down arrow, to increase or decrease the score of the marker
2. The score will be updated in the database accordingly, and the application will display the new total score

Exceptional Scenario:

**No markers of interest are present in the user’s area:** If there are no locations of interest in the user’s vicinity they may make use of the search feature to find points of interest in another location. Alternatively, they could add their own markers to the area.

**The user created the marker**: If the marker being viewed was created by the user, they will not have the option to vote, however will still be able to see the marker’s score. They will also be able to see an edit button, allowing the user to change details about the marker, and a delete button, if they wish to remove the marker from the map. Only the creator of the marker will have these options.

Use Case: View Account

Actor(s): User

Prerequisite:

1. User logs in to application

Prime Scenario:

1. User logs in
2. User is taken to the map UI screen.
3. User clicks the hamburger menu button on the main screen, which will display the menu and its options as well as showing the user’s score at the top of the menu
4. The user will select Account from the list of options
5. The user will be taken to the account UI screen, which will display the current values for username, and email address and an option to change their password (no password value will be displayed for security reasons)
6. The user will update any relevant personal information by pressing the edit button in the top right of the screen then clicking on the text box typing in the new value before applying changes (Alternative: **User wants to change password**) (Alternative: **User wishes to log out**) (Exceptions: **No information needs updating**)
7. The system will update this information in the database and refresh to display this new data on the account screen (Exception: **The user may close the app after updating information**)
8. The user will use the back button in the top right to return to the menu
9. The user will click the map on the side of the menu to return to the map UI screen

Alternative Scenario:

**User wants to change their password**:

* + - 1. User clicks ‘Change Password’ text
      2. New UI will open showing two textboxes, one for current password and one for new password
      3. The user will enter their current password, and the password they wish to replace it with
      4. User will click to submit password (Exception: **Incorrect current password entered**) (Exception: **Field left blank**)
      5. The database will update password and return the user to the menu screen

**User wishes to log out**

User clicks log out button

Confirmation message appears asking user if they are sure they wish to log out

User clicks yes (Exception: **User clicks no**)

The user is logged out and is returned to the log in screen

Exceptional Scenario:

**No information needs updating**:

1. User clicks the back button
2. The system will return the user to the menu

**The user may close the app after updating information**: After submitting the new information the user may choose to close the app immediately after using their phone’s controls. This does not require the user to leave the account screen in the application, but data will be lost if not submitted first.

**Incorrect current password**:

1. A message will appear informing the user they have entered their current password incorrectly
2. The user can dismiss this message
3. The user is returned to the change password screen and can amend their current password

**Field left blank or invalid input**

1. A message will appear informing the user they not entered information into a text field, or that their input is invalid
2. The user can dismiss this message
3. The user is returned to the change password screen and can amend their input data

**User clicks no**

1. User clicks no, so that that they are not logged out
2. User is returned to account UI screen

Use Case: View Settings

Actor(s): User

Prerequisite:

1. User logs in to application

Prime Scenario:

1. User logs in
2. User is taken to the map UI screen.
3. User clicks the hamburger menu button on the main screen, which will display the menu UI
4. The user will select ‘Settings’ from the list of options
5. The settings UI will appear
6. The user will change any relevant settings by changing values in relevant fields (May change between toggle buttons and drop downs depending on the setting) (Exceptions: **No settings need updating**)
7. The application will change accordingly to the new settings chosen, and the settings UI will display these new changes (Exception: **The user may close the app after updating information**)
8. The user will return to the menu using the back button

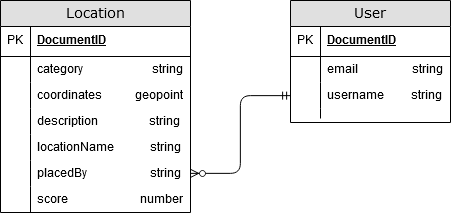
Exceptional Scenario:

**No Settings needs updating:**

1. User clicks the back button
2. The system will return the user to the menu

**The user may close the app after updating information**: After changing the desired settings the user may choose to close the app immediately after using their phone’s controls. This does not require the user to leave the account screen in the application. Data will not be lost when changed in settings, as the settings will apply on change.

## L) Entity Relationship Diagram



## M) Mock Advert Concept

The below image is a mock-up of where banner adverts would appear in the application, to be shown to volunteers for feedback.

