National College of Ireland
BSc in Computing
2017/2018

Curtis Murphy
x14357576
x14357576@student.ncirl.ie

Mental Health Advisor

Project Report

Table of Contents

Executive Summary

Problem being addressed

Mental health is something that a large section of people has to deal with on a daily basis. Bad mental health ranges in severity and type from anxiety and panic attacks to bipolar disorder and depression.

People suffering from these types of disorders and issues look for ways to cope and deal with them. They look for support from others who understand what they are going through. Unfortunately, not all of them know where to look for help, or even have a clear idea what problem they have. This is what this project hopes to fix.

This project's solution is to provide these people with a mobile application that will help address these problems some people have with finding help. The people will be able to find the right organizations to help them cope with their problems.

Key features of this project will be:

1. This project will consist of two different mobile apps that will work in conjunction with each other. The apps will be: A Doctor's app and a Patient app. The Doctor app will be used to add/edit information that will be displayed on the Patient App.
2. There is a self-checker function in the Patient app. A quiz will be displayed to the app user comprising of questions with multiple answers. The user can then answer the questions and submit these answers to the database.
3. The question and answers for the self-checker function are entered from the Doctor app. There is a functionality that allows for questions and their answers to be entered and added to the quiz via the Firebase Database.
4. There is a functionality that allows for users to find a specific location of a mental health facility/organisation using a Google Maps service. The Patients app has the map side of the functionality, showcasing specified locations on the map using markers.

5. The locations for the Maps function are entered and saved into the database using the Doctor app. The app takes the name of the location and the latitude and longitude entered and stores this information in the Firebase Realtime Database. This information can then be pulled down to be used in the Patient app.
6. The Doctor app has a function that allows for the entering of information regarding certain Mental Health conditions. The user will enter the name of the condition, followed by details on the condition, ways to cope with that condition and any resources or links to help with that.
7. The Patient app will showcase the information about the conditions that will be entered from the Doctor app. The Patient users can find out this information that may help them in their lives.

The technology used to implement this project will be a combination that work well together. The two applications will be built using the Android Studio workspace, which is an ideal application for designing and creating Android mobile apps.

The two apps will be a part of the same Google Firebase project. This will give them access to use the services that Google Firebase provides. The database used to store the information for the apps will be the Firebase Realtime Database. Because they are both apart of the same Firebase project, they will be able to access each other's information. Both apps will also use Google Firebase Authentication. This will allow for both versions of the app to have a Login and Register function.

1 Introduction

1.1 Background

The conception of this project came after seeing the basic scope description for a mental health application. It was the bare bones idea of an application designed around mental health that sparked the idea that would eventually become the finished project.

After deciding on the initial idea of an application centring on mental health, other questions had to be asked. What form will the application take? What functions will the application do for the user? How will any information be stored? These questions and more were asked when starting off with this project. The answer to all most of them was research. So, research was done into various mental health applications and websites to see what an ideal way would be to distribute information and connect with the people of today. Research was also done into what kind of functionality would be useful for mental health. From researching, it was concluded that the best platform to connect with people and distribute information in this day and age is a mobile application.

During the research into various medical applications, two caught my attention – “What’s up?” and “TalkSpace”. What’s up? is a free Android app that utilises certain therapy methods to help combat conditions such as Depression, Stress and others. It does this through its therapy methods and giving advice on how to manage your thoughts. With over 2,800 reviews on the Google Play store, it has a 4.4 rating on the store. TalkSpace is an online therapy tool that allows users to connect with a therapist without having to leave their house. They boast over a 1000 licensed therapist and one of their selling points is that their service can be significantly cheaper than the traditional image of therapy. Doing this research and finding apps like this gave me further insight into how my own project should be structured.

I also came across research papers on the effects of mental health apps on adolescents with mental health conditions. One was a study conducted by

Rebecca Girst, Joanna Porter and Paul Stallard. It was a study of other research publications over an 8 year period that had to do with certain keywords around mental health app, mental health and adolescents. Their conclusion was that there is insufficient evidence supporting the effectiveness of mental health apps on adolescents. They say that more robust studies are desperately needed to find out for sure.

Some members of my family have had trouble seeking help from counsellors/therapists in the past, with one having to wait three weeks to speak to someone. This seemed like an unacceptable time to wait, especially if you consider that a lot can happen in a few days let alone a few weeks. Their mood could spiral, and things could be worse off in a few weeks. This is what gave me the idea to make the project centre around receiving information about mental health conditions and getting feedback from doctors/therapists. Doing this will, hopefully, further interactions between sufferers of mental health conditions and the appropriate people/organisation.

1.2 Aims

The aim of this project is to provide a resource for people with mental health conditions, or those who suspect they might, where they can have access to information that will aid them. They will have access to descriptions of certain mental health conditions, followed by certain first steps that can be taken to help cope with the condition. This is, of course, all just a starting point for help. The next step (which is a lot of the time connected to ways to cope) is to get in contact with people/organisations who can support them. Which is why there will be a section for links to websites and phone numbers that can lead to those kinds of people such as Samaritans.

There will be a Google Maps functionality within the project. This functionality will use markers to pinpoint specific locations on the map. These locations will be the locations of various medical centres and GPs that can help with mental health. The goal will be that users will look at the map and choose one close to their home or in their area to go to or contact for help. They can see the location based on the

various place names on the map and they will be able to find out the name of the place by clicking on the marker. An information window should pop-up with the locations name for further reference.

There will be a quiz portion to the project called “Self-Checker”. A person can have access to a quiz and answer the question within it. They will then submit the answers to the database for storage and reference. The goal is for this quiz to have questions and answers relating to gaging the user’s state of mind/ if there is the possibility they might have certain mental condition or be predisposed to a certain condition.

The project will provide some connectivity between so called “Patients” and “Doctors” through the use of two versions of the app with one being for each group. The Doctor version will be an interface for the user to enter any information that they need to communicate to the user of the Patient version of the app. All of the functions described since now (Descriptions of conditions, Google Maps function and Self-checker quiz) will be available through the Patient Version. The person’s using the Doctor app will be professionals only, who are qualified and understand what information to disseminate to the Patient users.

The application developed for this project will be a scalable, mental health focused mobile app. The app should be responsive and developed to be run on Android devices. Since new mental health issues can be discovered or expanded on, and new support organizations can be founded all the time, scalability will be kept in mind when developing the application.

Hopefully this project will lead to helping other research papers in their task to see if mental health apps are effective. Like the one referenced earlier said, there needs to be more robust research on this topic. So, there is hope that creating apps like these will lead to the more effective research papers on the subject of mental health apps influence on mental health conditions.

While there are other mental health apps out there, the hope is that with this project, and its two version app set up, will provide a better connectivity between a Doctor and Patient through the sharing of information by the Doctor ap to the

Patient app. With both parties involved, it will hopefully lead to better cooperation between them both.

1.3 Technologies

1.3.1 Android Studio

The platform being used to develop the applications will be Android Studio. Studio is an easy to use Code Editor for the development and creation of applications for use on Android devices. There is a handy GUI function that can be used to structure and design the layout of the different pages in the apps. Android Studio also has a good compatibility with the other technologies being used in this project.

There will be two versions of this app, and both of them will be developed and designed on Android Studio.

1.3.2 Firebase

As stated, Android Studio is easily compatible with the other technologies, including this one. Google Firebase is a cloud-based service that provides a multitude of services, a couple of which are utilised in this project.

1.3.2.1 Realtime Database

The Realtime Database function in Firebase is used throughout both apps. The data stored within the database is done so in a JSON format instead of being store in tables. This is because Firebase uses a NoSQL type of database that is maintained by Google.

The information stored is the like of the locations for the Google Maps function, the names and information for the various conditions, and the questions and answers for the quiz. This information is all taken and entered in from the Doctor version of the app.

1.3.2.2 Firebase Authentication

The Firebase Authentication is used by both versions of the app. This functionality allows for the saving and continual storage of login credentials This allows for both

apps to have functional Login and Register functionality. The apps can take in an email address and a password through the Register activity and store it on Firebase. This information can then be used in the Login Activity.

1.3.3 Google Maps

This project uses the Google Maps API for the map functionality. This involves creating an API key and inserting it into the project. The marked locations are stored using the Realtime Database and are then displayed on the map.

1.4 Structure

Introduction

The Introduction chapter is where the basic idea for the project is introduced. The basic aims of the project will be outlined in a section of this chapter. The background of the project will also have its own section in the introduction along with another section describing the technologies involved in the project.

System

The System chapter will hold the bulk of the report. It will include a section holding the functional requirements of the project, including the use case diagram and the use cases for each of the requirements. Other sections included in this chapter are the data requirements section, where the requirements for storing the projects data are outlined, the user requirements section, where the requirements needed for the optimal user experience with the project are laid out and the environmental requirements section where any requirements about the user's environment when using either of the applications are specified. There is also a section detailing the usability requirements needed for the applications to run at their best. All of these sections will be held in a larger section called requirements.

After the Requirements section, the next section in the system chapter, is the Design and Architecture section. This section will include two diagrams that

showcases the architecture of each application's system. It will also include a written section describing the diagrams.

The Implementation section will eventually include snippets and descriptions of code and classes from both applications.

The GUI Layout section will be where the screenshots of both application's GUI will be displayed. Each page's screenshot will have a description underneath it, outlining what each part of the screenshot does or is for.

The Testing section will outline the testing plan for testing the two applications. It will also outline any testing tools that are needed to test the Mental Health Advisor.

The Customer Testing section of the System Chapter will outline any testing having to do with customers and/or users. Although I won't be testing people so there won't be much in that section.

The Evaluation section will outline how the system was evaluated and what the results from that evaluation were.

Conclusion

The Conclusion chapter will outline the advantages and disadvantages of the project. It will also detail any of the opportunities and limitations that the project has.

Further Development/Research

This chapter will be for an analysis of where the project could lead if given more resources and research.

References

The References chapter will be for the citing of any outside resources used and looked up for this project. Any outside resources used will be put in here.

Appendix

This chapter will include any sections or documents that are not part of the main technical report but have to be included.

The Project Proposal section will be where the project proposal for this project will be included.

The Monthly Journals section is where the reflective journals for each month will be included. These journals will outline what was done for the project for that month. At the end of the project there should be one for each month.

There will be a section with a User Manual with steps on how to do each function in the two app.

2 System

2.1 Requirements.

2.1.1 Functional requirements

There are two apps that make up this project. They each have their own functionalities

1. Both versions of the app have the same Login functionality. User enters in an email address and a password to access the rest of the app
2. If they both have Login functionality then they both have the same Register functionality. Enter the email and password you want for the app and submit it to the database to be save for future logins.

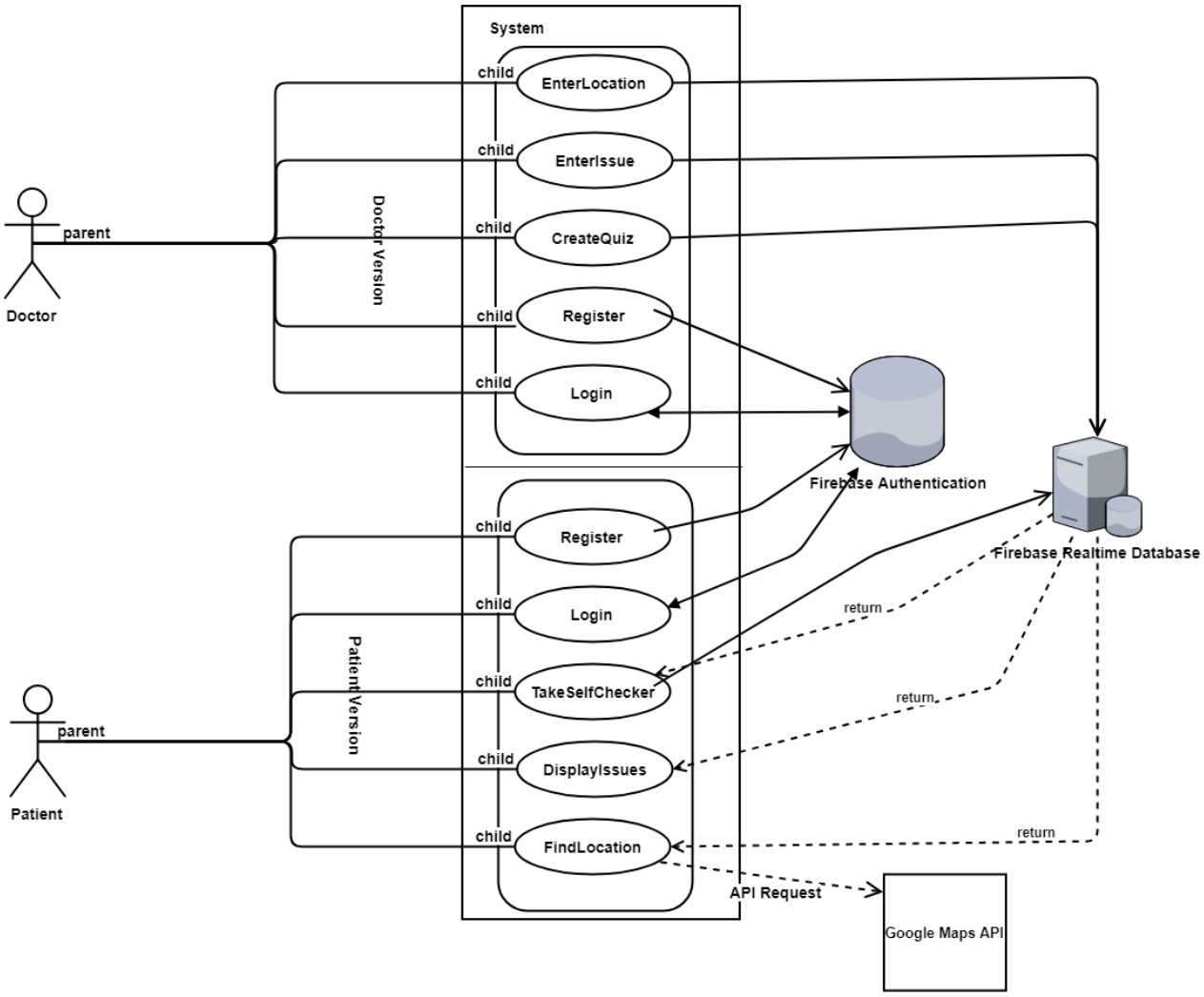
2.1.1.1 Mental Health Advisor – Doctor Version

1. The Doctor app provides an interface to enter in questions and answers that will be save in the Firebase database. These questions and answers will be used in the Patient app.
2. The user will also be able to enter information on various mental health conditions and push it to the Realtime Database. This information will be used by the Patient app.
3. The user will have the ability to enter in the name, latitude and longitude into the Database through the use of an interface provided. These locations show up on the Patient app

2.1.1.2 Mental Health Advisor – Patient Version

1. The Patient app will have the ability to display to the user information, that was entered from the Doctor app into the Database, about various mental health conditions.
2. The user will be able to find locations of various mental health facilities in Ireland through the use of a Google Maps function that displays locations stored in the Database by the Doctor App.
3. The user will be able to take a self-checker quiz and submit the answers to the Realtime Database. The questions and answers for the quiz will be created from the Doctor app.

2.1.2 Use Case Diagram



2.1.3 Requirement 1 Login to App

2.1.3.1 Description & Priority

Users can login to either version of the app with an account they had previously created with that app. The user can logout of the app when they like. Doing so, should return the user to the login page.

2.1.3.2 Use Case

Login to App

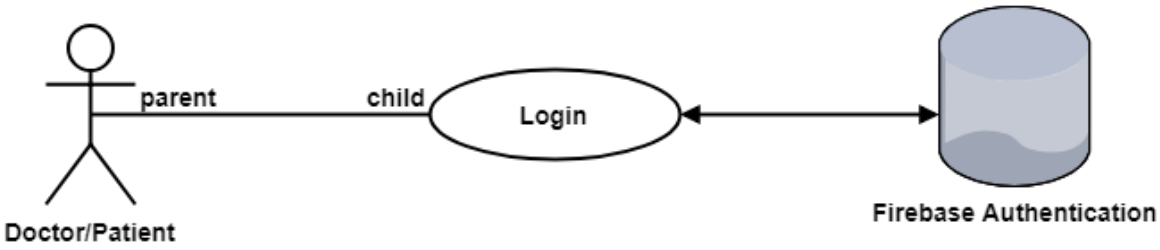
Scope

The scope of this use case is for users to login to the Mental Health Advisor (either version)

Description

This use case describes the logging in to the either version of the application by a user

Use Case Diagram



Flow Description

Precondition

The app has been opened on the phone and the user is registered with the app.

Activation

This use case starts when the application is launched.

Main flow

1. The User launches the application.
2. The User enters their login details. (See A1 and E1)
3. The User presses the login button.
4. The system verifies the user's details
5. The system displays the home page to the user.
6. The use case ends

Alternate flow

A1: Register with App

1. The User clicks the link to redirect to the register page.
2. The User enters what they want their login details to be.

3. The User presses the register button
4. The system saves the login information in the database
5. The use case continues at position 4 of the main flow

Exceptional flow

E1: Closes App

1. The User closes the application.
2. The use case continues at position 6 of the main flow

Termination

The user closes the app.

Post condition

The system goes into a wait state

2.1.4 Requirement 2 Register to App

2.1.4.1 Description & Priority

New users can register to either of the apps with an email address of theirs and their own password. After registering with the app, the user can log into the app when they like.

2.1.4.2 Use Case

Register to App

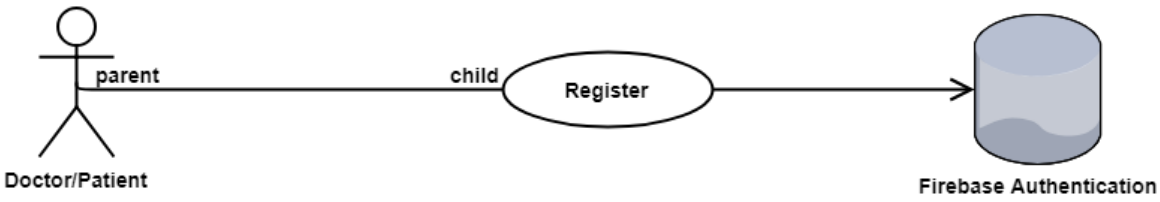
Scope

The scope of this use case is for users to register with either versions of the Mental Health Advisor.

Description

This use case describes the registering with the app by the user.

Use Case Diagram



Flow Description

Precondition

The app has been opened on the phone and the user is not currently registered with the app.

Activation

This use case starts when either of the applications is launched.

Main flow

1. The User launches the application.
2. The User clicks the link to the register page. (See A1 and E1)
3. The User enters the email address and password they want as their login details.
4. The User presses the register button.
5. The system verifies the user's details and stores them in the database.
6. The system displays the home page to the user.
7. The use case ends

Alternate flow

A1: Login with App

1. The User enters credentials already stored in the database into the login page fields displayed on the apps opening page.
2. The User presses the login button
3. The system verifies the user's details
4. The use case continues at position 6 of the main flow

Exceptional flow

E1: Closes App

1. The User closes the application.

2. The use case continues at position 6 of the main flow

Termination

The user closes the user.

Post condition

The system goes into a wait state

2.1.5 Requirement 3 Find Locations

2.1.5.1 Description & Priority

The Find Locations requirement is an essential part of the entire system. This requirement is what lets the users find the location of any medical centers or other types of locations in their area that deal with mental health. This requirement is only available in the Patient version of the app.

2.1.5.2 Use Case

Find Locations

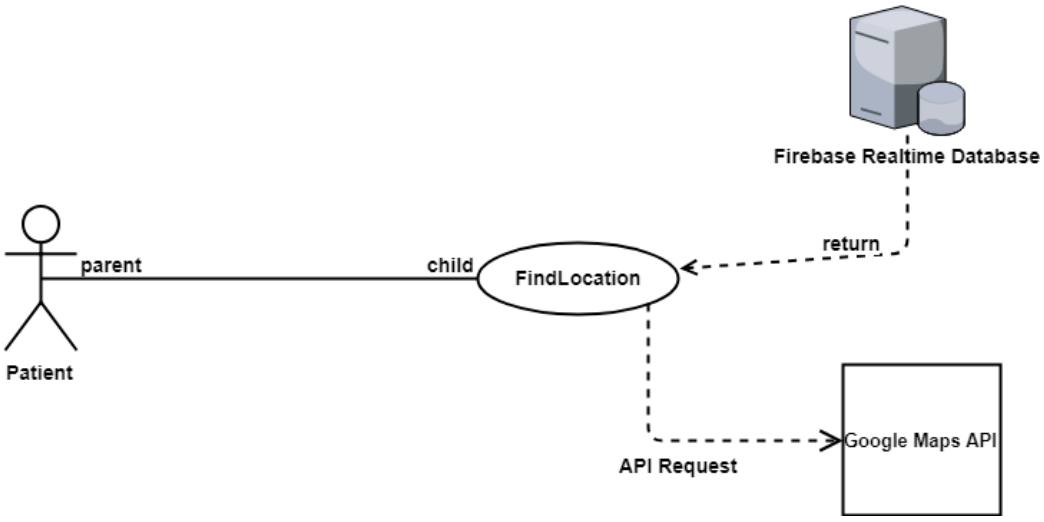
Scope

The scope of this use case is for patient users to find the locations of medical centres in their area.

Description

This use case describes the finding of a location by a user.

Use Case Diagram



Flow Description

Precondition

The Patient app has been opened and the user has logged in.

Activation

This use case starts when a User is looking for medical centres in their area.

Main flow

7. The User logs into the application.
8. The User navigates to the Find Locations page. (See E1)
9. The system displays a Google Map with relevant locations marked on it.
10. The User chooses a location on the Map that is in their area. (See A1)
11. The system displays a pop-up that includes the location information
12. The User reads this location information
13. The use case ends.

Alternate flow

A1: Find Different Location

1. The User chooses a different marked location on the map
2. The use case continues at position 11 of the main flow

Exceptional flow

E1: Choose Different Option

1. The User navigates towards a different page in the app
2. The use case continues at position 13 of the main flow

Termination

The User is finished looking for information on the organizations.

Post condition

The system goes into a wait state

2.1.6 Requirement 4 Display Issues

2.1.6.1 Description & Priority

The Display Issues requirement displays information about various mental health conditions. The information includes basic description, simple ways to cope and links/resources to contact people. This requirement is only available on the Patient version of the app. All the information comes from the Doctor version of the app.

2.1.6.2 Use Case

Display Issues

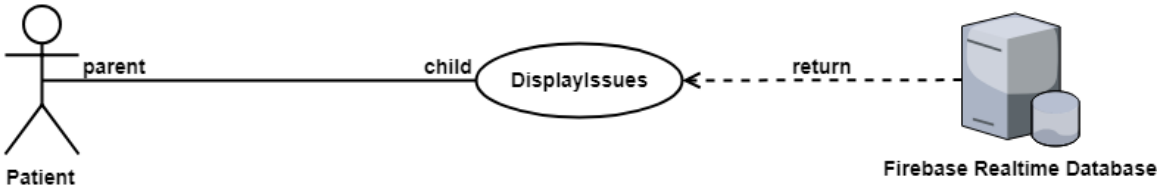
Scope

The scope of this use case is for users to find information specific mental health conditions stored in the Database.

Description

This use case describes the displaying of information on mental health conditions by the application to the user.

Use Case Diagram



Flow Description

Precondition

The app has been opened on the phone screen.

Activation

This use case starts when a User has logged into the app

Main flow

1. The User presses the button called Mental Health Conditions. (See E1)

2. The system navigates to the page in question.
3. The system displays information on the mental health conditions stored in the database.
4. The User chooses one condition to read in detail.
5. The use case ends.

Alternate flow

A1: Find Different Location

1. The User chooses a different condition to read in detail.
2. The use case continues at position 5 of the main flow

Exceptional flow

E1: Choose Different Option

1. The User navigates towards a different page in the app
2. The use case continues at position 5 of the main flow

Termination

The application is closed by the user.

Post condition

The system goes into a wait state

2.1.7 Requirement 5 Take Self-Checker

2.1.7.1 Description & Priority

The Take Self-Checker requirement is a high priority of the app. This requirement is for the Patient app only. A quiz is displayed to the user with several questions and answers. The user answers the questions and submits them to the database.

2.1.7.2 Use Case

Take Self-Checker

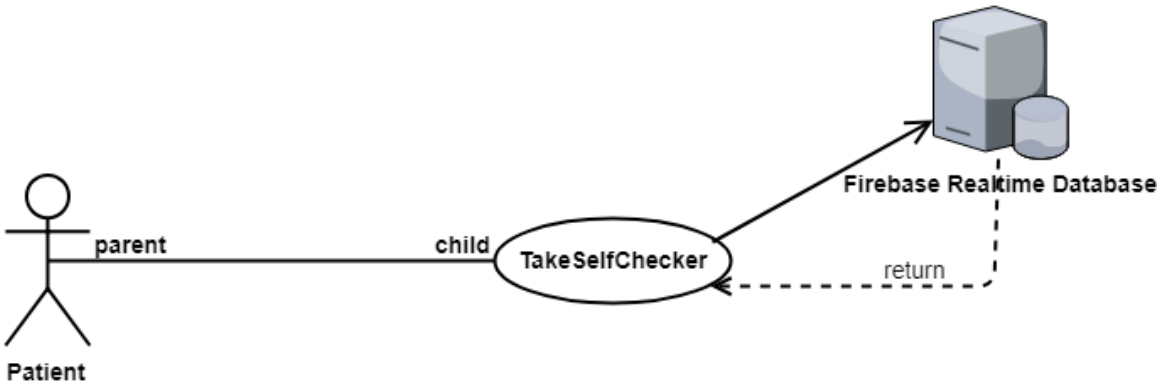
Scope

The scope of this use case is for users to answer the questions in a quiz and submit those answers to the database.

Description

This use case describes the answering of questions in a quiz and the submitting of those answers to the Database.

Use Case Diagram



Flow Description

Precondition

The app has been turned on.

Activation

This use case starts when a User is finished logging into the app.

Main flow

1. The User presses the button labelled “Self-Checker”. (See E1)
2. The system displays the page with the quiz information that is stored in the Realtime Database.
3. The User selects the answers that best suit them within the questions of the quiz. (See A1)
4. The User presses the submit button
5. The system submits the answers to the database where they are stored.
6. The use case ends.

Alternate flow

A1: Choose different Answers.

1. The User chooses different answers to the questions in the quiz.
2. The use case continues at position 4 of the main flow

Exceptional flow

E1: Choose Different Option

1. The User navigates towards a different page in the app
2. The use case continues at position 2 of the main flow

Termination

The system presents a message saying the answers were saved to the database.

Post condition

The system displays the page with the quiz information on it.

2.1.8 Requirement 6 Enter Issues

2.1.8.1 Description & Priority

The Enter Issues requirement is only found on the Doctor app. This is the interface through which the information on selected mental health conditions is entered into the Realtime Database. This information is then used within the Patient version of the app.

2.1.8.2 Use Case

Enter Issues

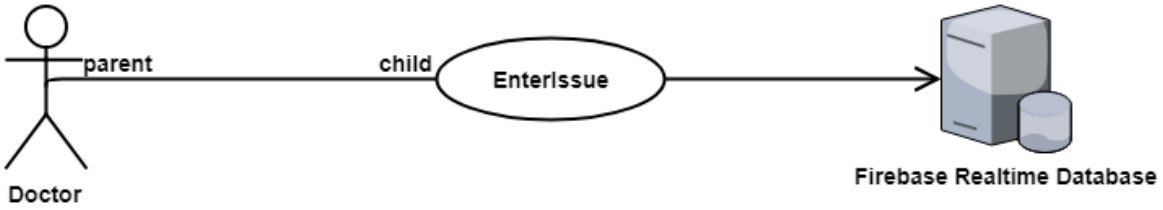
Scope

The scope of this use case is for users to enter in information on various mental health conditions and save them to the database.

Description

This use case describes the entering of mental health condition information and the saving of said information into the Realtime Database.

Use Case Diagram



Flow Description

Precondition

The app has been turned on.

Activation

This use case starts when a User has logged into the app.

Main flow

1. The User presses the Create Issue button (See E1)
2. The system displays the interface for entering in the issue information
3. The User enters in the information for a mental health condition. (See A1)
4. The User presses the submit button once they have entered all the information they want to.
5. The system submits the information to the database where it is saved.
6. The use case ends.

Alternate flow

A1: Enter Different Information

1. The User enters in different information for a different mental health condition.
2. The use case continues at position 4 of the main flow

Exceptional flow

E1: Choose Different Option

1. The User navigates towards a different page in the app
2. The use case continues at position 2 of the main flow

Termination

The system presents the Mental Health problem that the self-checker determined matched the user's answers, to that user

Post condition

The system goes into a wait state

2.1.9 Requirement 7 Create Quiz

2.1.9.1 Description & Priority

The Create Quiz function is in the Doctor version of the app. It is where the quiz that the Patient app displays is created. The user of the Doctor app can enter in the question and the answers they want and add it to the Realtime Database where it'll be used for the Patient app.

2.1.9.2 Use Case

Create Quiz

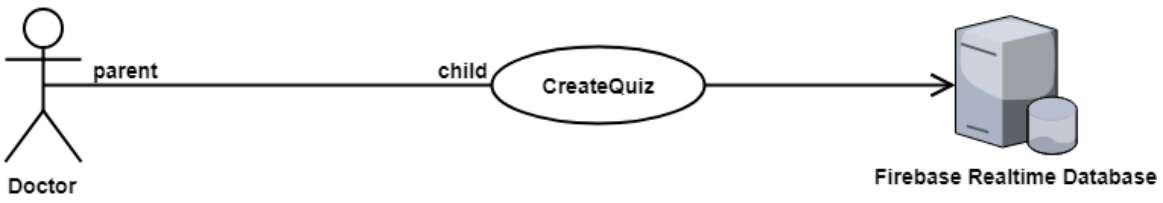
Scope

The scope of this use case is for users to enter in questions and answers that they wish to be a part of a quiz for the Patient app.

Description

This use case describes the entering of questions and answers for a quiz and the saving of said information into the Database.

Use Case Diagram



Flow Description

Precondition

The app has been turned on.

Activation

This use case starts when a User has logged into the app.

Main flow

1. The User presses the Create Quiz button (See E1)
2. The system displays the interface for entering in the question information
3. The User enters in the question information they want to add to the quiz.(See A1)
4. The User presses the submit button once they have entered all the question information they can.
5. The system submits the information to the database where it is saved.
6. The User continues to enter questions and answers until they are finished creating the quiz.
7. The use case ends.

Alternate flow

A1: Enter Different Question

1. The User enters in a different question to be added to the quiz via the Database.
2. The use case continues at position 4 of the main flow

Exceptional flow

E1: Choose Different Option

1. The User navigates towards a different page in the app
2. The use case continues at position 2 of the main flow

Termination

The system presents a clear interface for another question to be added.

Post condition

The system goes into a wait state

2.1.10 Requirement 8 Enter Location

2.1.10.1 Description & Priority

The Enter Location requirement is the entering of location information for specific medical centers into an interface on the Doctor app only. This information is then saved to the Realtime Database. It is then used by the Patient app to mark those locations on a Google Map Activity.

Enter Location

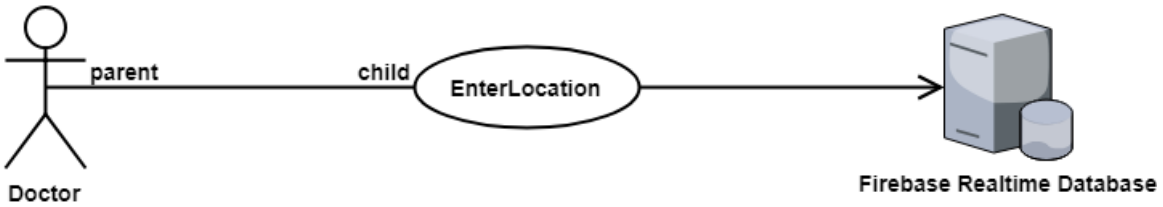
Scope

The scope of this use case is for users to enter in location information of medical centres which is the save to the database for future use.

Description

This use case describes the entering of location information into the interface on the Doctor app and from there saving it to the Realtime Database for later use.

Use Case Diagram



Flow Description

Precondition

The app has been turned on.

Activation

This use case starts when a User has logged into the app.

Main flow

1. The User presses the Create Location button (See E1)
2. The system displays the interface for entering in the location information.
3. The User enters in the location information for a specific medical centre or organisation. (See A1)
4. The User presses the submit button once they have entered all the location information needed.
5. The system submits the information to the database where it is saved.
6. The use case ends.

Alternate flow

A1: Enter Different Location

1. The User enters in different location information for a different medical centre or organisation.
2. The use case continues at position 4 of the main flow

Exceptional flow

E1: Choose Different Option

1. The User navigates towards a different page in the app
2. The use case continues at position 2 of the main flow

Termination

The system presents the user with an empty Enter Location interface in case they want to enter in another location.

Post condition

The system goes into a wait state

2.2 Data requirements

In order to store the data needed for this project, a proper database is needed. The cloud-based data storage service, Google Firebase, will be used as a backend solution for the storage of the applications data. The Firebase database will have to store data coming in from the Doctor version of the app. It will then have to be able to send that data to the Patient version for users of that app to see and use.

The database being used for the Mental Health Advisor apps is a NoSQL type database that uses cloud-storage. All data stored within this database is stored as a JSON format within JSON objects.

There needs to be a service to register and store account information for logging into the apps. Again, Google Firebase provides this. The service is called Firebase Authentication and it stores the account information needed for logging into the apps (email address, passwords).

2.3 User requirements

Usability

The apps must be easy to use. The navigation between all of the activities for each app should be obvious and not hidden away on the application page. The design should be intuitive. This ease of use is to make things easier for users in general but also for users that haven't had much experience with mobile smartphones and technology in general.

Security

Some of the data in the project's database (organizations' locations,) is public knowledge. But the results from the users taking the self-checker quiz are sensitive information. Thankfully, Google Firebase very secure and will keep the users' data stored safely and protected. It's managed by Google which is one of the biggest tech giants in the world. Their security is great.

Availability

The users will need the appropriate hardware to run either of these applications. In the case of this project, that hardware is an Android smartphone. Both applications are being developed in the same Android environment (Android Studio), so an Android device is needed. Hopefully, any future versions of this project can be expanded upon to include other platforms such as the iPhone's iOS.

2.4 Environmental requirements

Internet

An internet connection is essential for both versions of the app, especially if you want to run it at its fullest. The apps won't have access to the information on the Database, which is especially bad for the Patient app as that uses most of the Database information. There will still be navigational functionality without internet (unless you're logged out) but other than that, there is very little.

2.5 Usability requirements

Mobile users these days cannot tolerate lag or delays in their mobile applications. Too much lag or delays and a mobile user may drop that application and move on to one of its competitors.

Navigation

The menus for this app will be easy to use and consistent throughout the entire app. The user will be able to go back to the home page from any of the screens in the app. Android smartphones have built in back button, unlike iPhones.

There should be no scrolling by the necessary by the user to use the menu. All menu options should be visible to the user with no scrolling features included. The buttons that lead to the other functionalities will be front and centre on the Homepage screen on both apps.

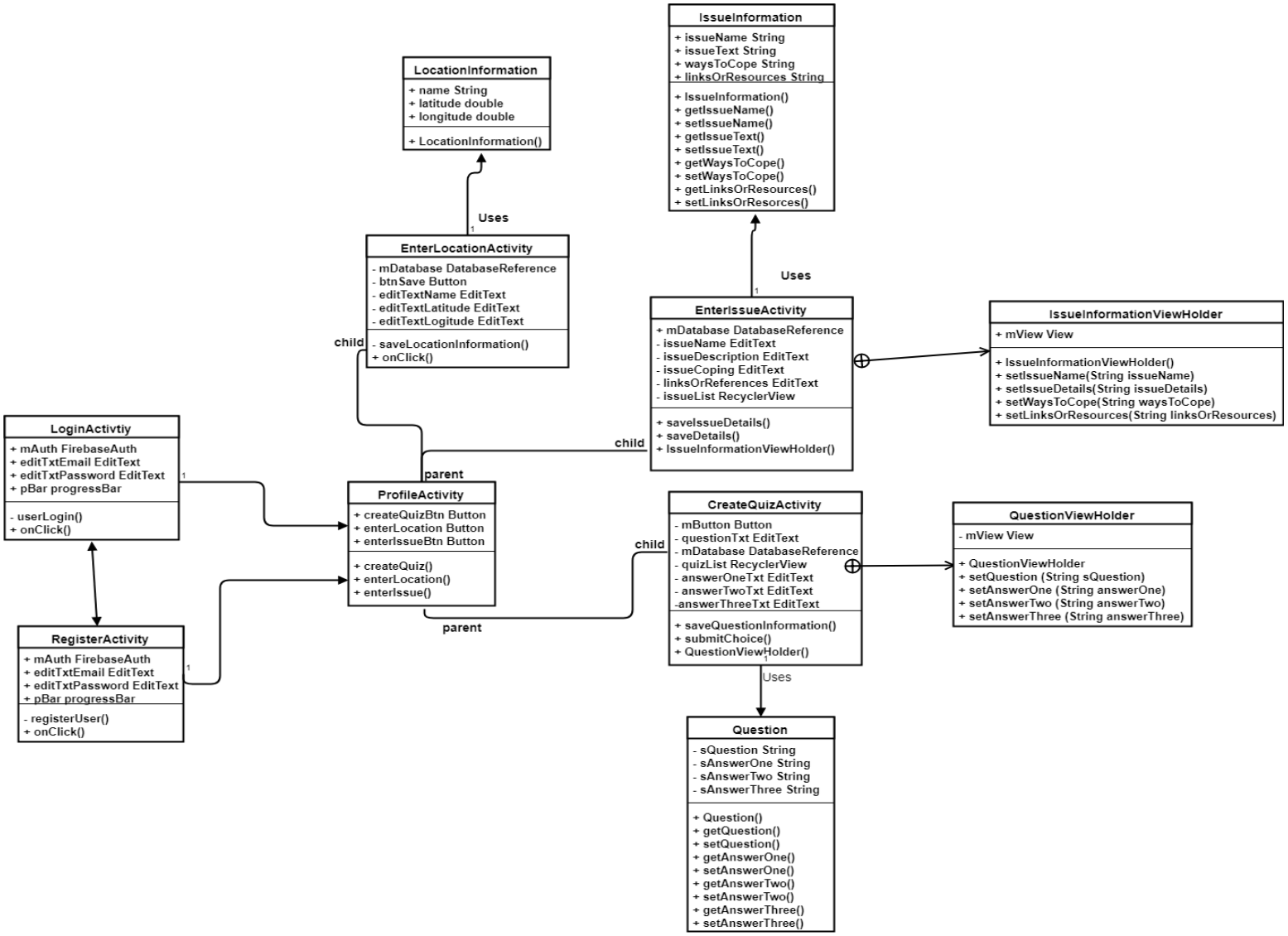
2.6 Design and Architecture

2.6.1 Architecture Design

These applications will be developed in an Android platform and be connected to a cloud-based backend database as a part of Google Firebase. The user's android device will be able to receive some of the data in the database, especially if they are using the Patient version of the app, and also be able to send their own information to be stored in the database, especially if they are using the Doctor version of the app

2.6.2 Class Diagram

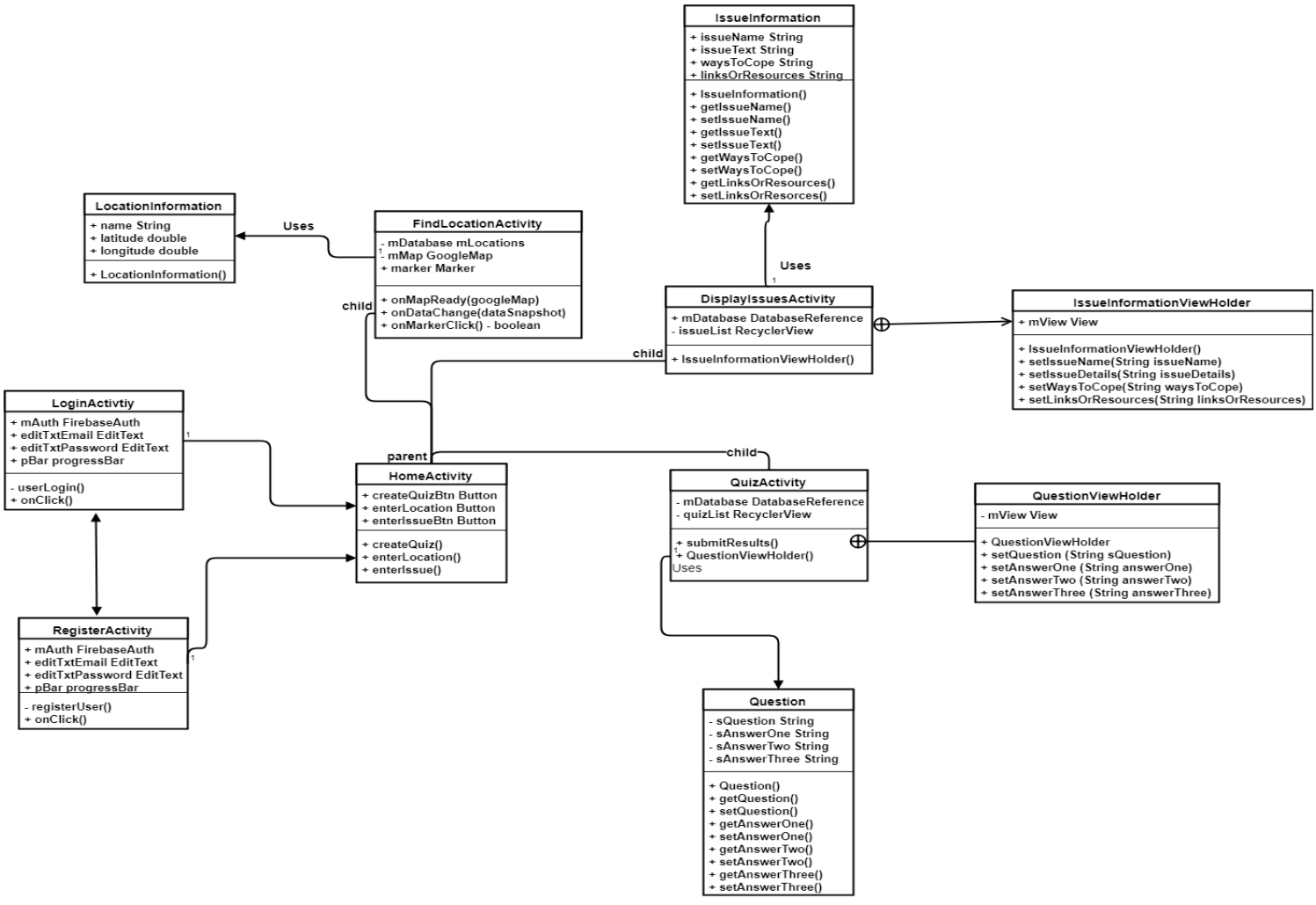
2.6.2.1 Mental Health Advisor – Doctor Version



The class diagram above is an outline of the class architecture of the Doctor version of the Mental Health Advisor. The class diagram shows how the system classes and methods are connected. The system starts with logging in or registering with the application. This involves entering in a valid email address and a password. After that, you're directed to the ProfileActivity page. Once there, the architecture of this system branches off further. The ProfileActivity class contains methods that direct a user to three other class activities. There is the CreateQuizActivity, which includes a subclass called QuestionViewHolder, the EnterIssueActivtiy which also includes a subclass, this time called IssueInformationViewHolder. The third class activity the ProfileActvity class leads to is the EnterLocationActivity which does not contain a subclass.

All three of these classes use information that is set in other classes. The CreateQuizActivity uses the Question class and its methods, the EnterLocationActivity uses the LocationInformation class and its methods, and the EnterIssueActivity utilizes the IssueInformation class and all of its methods.

2.6.2.2 Mental Health Advisor – Patient Version



This is the class architecture for the classes in the Patient version of the Mental Health Advisor. The class diagram shows how the system classes and methods are connected. The system starts with logging in or registering with the application. This involves entering in a valid email address and a password. After that you are directed to the HomeActivity class page. From here, the architecture branches out more to include other classes.

The HomeActivtiy has methods that direct users to three different class Activities. The classes are the QuizActivity, FindLocationActivity and DisplayIssuesActivity. Both of the QuizActivity and the DisplayIssuesActivity have subclasses, with QuizActivity's subclass being QuestionViewHolder and DisplayIssuesActivity's subclass being IssueInformationViewHolder. All three of the classes take information from other class files. QuizActivity takes from the Question class, DisplayIssuesActivity takes from the IssueInformation class and the FindLocationActivity takes from the LocationInformation class.

2.7 Implementation

When implementing this project, it was best to decide the type of application that would be best for use. There are three types of applications for mobile:

1. ***Web Apps***

Web Apps are essentially just responsively designed websites that, because they are designed like that, they can work quite well on mobile devices. The usual tools used to build such apps are HTML and CSS with perhaps some JavaScript included too.

They generally have low device memory due to the fact that they are still a website-based application. Most data would be stored on remote servers and accessed via the internet. This does mean, however, that a bad internet connection will lead to very poor performance and possibly the inability to access data. Another of the downsides of these apps is that there is very little API support available for them.

2. ***Hybrid Apps***

Hybrid apps are created using tools and technologies that are best-suited for multi-platform. This would include the likes of HTML. CSS and JavaScript, the same technologies used or Web Apps. This is because, when it gets right down to it, hybrid apps are usually just Web Apps that have been placed within a native wrapper.

Hybrid apps are generally quick to develop and very easy to do so too. Also, unlike Web Apps, Hybrids have a larger range of APIs to choose from, like gyroscope or geolocation.

Those advantages are great but, there are downsides to these apps too. While they are quick to develop, the speediness doesn't always carry over to performance and their optimization is bad too. They generally pale in comparison to the native apps.

3. ***Native Apps***

These are mobile apps that have been created for use on a singular operating system and will therefore only work on devices with this system. So, if an application was made for Android devices, odds are it'll only work for devices with that operating system.

These apps tend to have a higher performance quality than the other types and also usually has the use of a native UI for use with a device. Like Hybrid apps, there is a large range of APIs available to use in these apps. Generally speaking, these types of apps are usually for sale in online stores connected to the operating system it was made for (Google Play Store for Android, Apple Store for iOS).

A downside for these apps however, is cost. There is a usually a larger cost for creating these types than the other types. This is due to the fact that each operating system you want your app to work on means you have to create duplicate apps that work for those specific operating systems.

For this project, after researching the types of apps there are, it was concluded that the best app for this project is Native apps. Native apps have higher performance than the other two types of apps and provides a better user experience because of it. A hybrid app has poor speed and optimisation compared to a native app and a Web app is dependent on a strong internet connection to work at optimal efficiency. Adding up all of the pros and cons of each led me to believe that this project would benefit the best from being in the form of a native app, specifically it will be native to Android.

2.7.1 Android Studio

With the type of app decided, the environment within which it would be developed was needed. For that, Android Studio was chosen. Android Studio is a code editor that provides the tools needed to create applications for every Android device out there. It has tools such as the Visual Layout Editor, which allows for you to visually see the layout of a specific page that you are working on. It also allows for you to drag, place, resize and otherwise interact with the individual parts of a page's layout. For example, you can drag a Text View containing the title of the page and place it wherever you want and resize it however you want, all without have to type out the instructions.

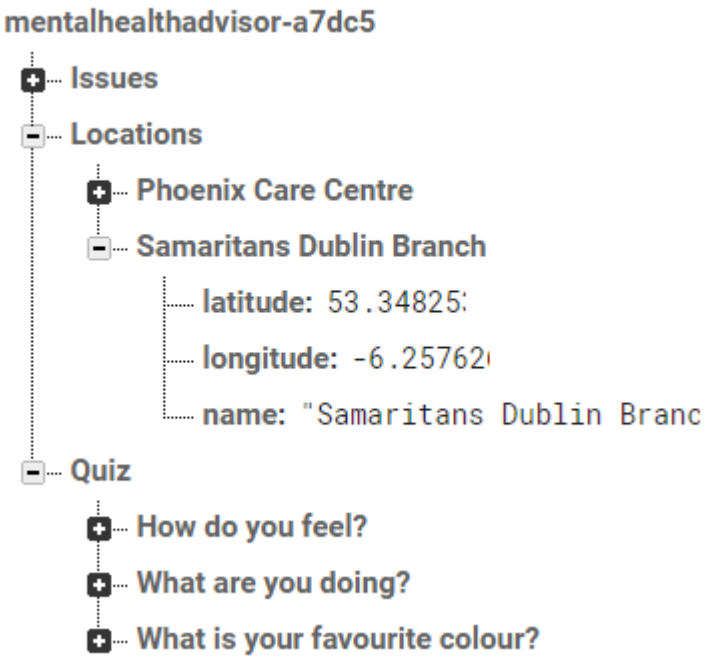
The different code files are also grouped correctly, with all the gradle files being grouped together, the Java Activity files are grouped together, and all of the layout xml files are grouped together. For reference sake this is handy.

2.7.2 Firebase Realtime Database

For the storage of the projects data, a Google Firebase project was created called "MentalHealthAdvisor" and both versions of the app were added to it. With both added, they have access to the services of were available and it is with the Firebase Realtime Database that both apps data is stored.

The apps have functionalities for making quizzes, showing specified locations on a map and displaying general information about mental health conditions. All this

information is store in this database in a JSON format, for use in the apps when needed.



This is the layout that the information is stored in the Realtime Database. The information is stored in a master Parent called mentalhealthadvisor and each child of this parent subsequently has their own children nodes. This leads to the nested way with which the data is stored. Those main children of the master Parent node are the ones that are called and referenced in the code. This tells the app which tree of information to call data from or it tells the app which child to add data to.

```
private DatabaseReference mDatabase;
```

This line uses the DatabaseReference object type to create an object called mDatabase This object is then made equal to the instance of a child within the Firebase Realtime Database for this project.

```
mDatabase = FirebaseDatabase.getInstance().getReference().child("Quiz");
```

The object, mDatabase, will then be used to either add more data to the specified child, or call information down from that child.

2.7.3 Firebase Authentication

Firebase also provided a service that is used to allow login and register information to be stored for future use. The service is called Firebase Authentication and for this project, it allows the storing of email and password information within its authentication database through the use of a Register function. This information can then be used within a Login function to access the app that it is included in.

```
mAuth.createUserWithEmailAndPassword(email, password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
    @Override
    public void onComplete(@NonNull Task<AuthResult> task) {
        pBar.setVisibility(View.GONE);
        if(task.isSuccessful()) {
            Intent intent = new Intent ( packageContext: RegisterActivity.this, ProfileActivity.class);
            intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
            startActivity(intent);
        }
        else {
            if(task.getException() instanceof FirebaseAuthUserCollisionException) {
                Toast.makeText(getApplicationContext(), text: "You are already registered", Toast.LENGTH_SHORT).show();
            }
            else {
                Toast.makeText(getApplicationContext(), task.getException().getMessage(), Toast.LENGTH_SHORT).show();
            }
        }
    }
});
```

The Register function uses the method, “createUserWithEmailAndPassword()” to add new email and passwords to the firebase authentication database. The object, mAuth, is equal to an instance of the FirebaseAuth.

```
mAuth.signInWithEmailAndPassword(emailAddress, password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
    @Override
    public void onComplete(@NonNull Task<AuthResult> task) {
        pBar.setVisibility(View.GONE);
        if(task.isSuccessful()) {
            Intent intent = new Intent ( packageContext: LoginActivity.this, ProfileActivity.class);
            intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
            startActivity(intent);
        }
        else {
            Toast.makeText(getApplicationContext(), task.getException().getMessage(), Toast.LENGTH_SHORT).show();
        }
    }
});
```

The Login function uses the method “signInWithEmailAndPassword()” to compare the entered email and password to any inside the authentication database. If it does match both email and password, then it finishes logging in and takes the user to a specified Activity.

2.7.4 FirebaseUI

FirebaseUI is a collection of libraries that is grouped by the API that they are for. The ones used for this project are the library for FirebaseUI Database and the library for FirebaseUI Auth.

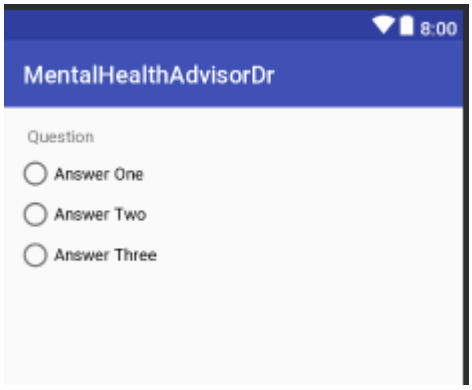
The FirebaseUI Database makes it easier for the data from the Realtime Database to be bound to the UI of the apps in the project. This essentially allows me to set up a layout of Text Views and other widgets with holder text inside them. Then, with FirebaseUI, that holder text can be replaced with the information within the Realtime Database.

2.7.4.1 Data Model

```
public class Question {
    private String sQuestion;
    private String sAnswerOne;
    private String sAnswerTwo;
    private String sAnswerThree;

    public Question(){

    }

    public Question(String sQuestion, String sAnswerOne, String sAnswerTwo, String sAnswerThree) {
        this.sQuestion = sQuestion;
        this.sAnswerOne = sAnswerOne;
        this.sAnswerTwo = sAnswerTwo;
        this.sAnswerThree = sAnswerThree;
    }

    public String getQuestion() { return sQuestion; }

    public void setQuestion(String sQuestion) { this.sQuestion = sQuestion; }

    public String getAnswerOne() { return sAnswerOne; }

    public void setAnswerOne(String sAnswerOne) { this.sAnswerOne = sAnswerOne; }

    public String getAnswerTwo() { return sAnswerTwo; }

    public void setAnswerTwo(String sAnswerTwo) { this.sAnswerTwo = sAnswerTwo; }

    public String getAnswerThree() { return sAnswerThree; }

    public void setAnswerThree(String sAnswerThree) { this.sAnswerThree = sAnswerThree; }
}
```

This is the Data Model for the creation of a question in a quiz. There are EditText fields in a layout page that, when words are entered into it and a submit button is pressed, stores that entered information into the Realtime Database. Because of this Data Model, and more code in the specific Activity, that information entered, gets mapped and matched to the different objects in this model. That information is then what replaces the holder text in the question_layout file widgets.

2.7.4.2 Question_layout



This is the question layout design. There is a TextView with holder text, “Question”, and a RadioGroup widget with three radio Buttons with their own holder text. This text is replaced by the information stored in the Database. This layout itself is then used, with the new information, and displayed inside a RecyclerView.

2.7.4.3 RecyclerView

RecyclerView is an alternative to something like a ListView. It is used to display the set question_layout a multitude of times, with each iteration stacked on top of one another. Each question_layout added to the Recycler view, should have its holder data replaced by different child data inside the Database based on the parameters for the data

```
mDatabase = FirebaseDatabase.getInstance().getReference().child("Quiz");//create
quizList = (RecyclerView) findViewById(R.id.quizRec);
quizList.setHasFixedSize(true);
LinearLayoutManager linearLayoutManager = new LinearLayoutManager( context: this);
linearLayoutManager.setStackFromEnd(false);
quizList.setLayoutManager(linearLayoutManager);
```

mDatabase is a DatabaseReference object that gets an instance of the child reference called “Quiz”. The quizList object is a RecyclerView one and is equal to the id of the RecyclerView. LinearLayoutManager is making sure that the way the objects entering the RecyclerView are stacked, is done properly.

```
public void saveQuestionInformation () {
    final String questionValue = editText.getText().toString().trim();
    final String answerOneValue = answerOneTxt.getText().toString().trim();
    final String answerTwoValue = answerTwoTxt.getText().toString().trim();
    final String answerThreeValue = answerThreeTxt.getText().toString().trim();
    Question question = new Question(questionValue, answerOneValue, answerTwoValue, answerThreeValue);

    if ((!TextUtils.isEmpty(questionValue )) && (!TextUtils.isEmpty(answerOneValue))) {
        final DatabaseReference newQuestion = mDatabase.push();

        mDatabase.child(questionValue).setValue(question);
    }

}
```

This method contains the code for pushing any information entered into the specified EditTextFields, up to the Database. The .push() method is what is used to save the information and it is put at the end of the mDatabase object which is

equal to the instance of the “Quiz” child in the Database. Therefore, the information will be pushed and stored within that child. The information will also, as stated, replace the holder text in the question_layout, and that layout will be added to the RecyclerView.

2.7.4.4 Query

```
Query query = FirebaseDatabase.getInstance()
        .getReference()
        .child("Quiz")
        .limitToLast(50);



FirebaseRecyclerOptions<Question> options =
        new FirebaseRecyclerOptions.Builder<Question>()
                .setQuery(query, Question.class)
                .build();
```

The query above is equal to the instance of the child “Quiz”, just like the mDatabase object. But unlike that one, query is only equal to a certain amount of the data stored in the “Quiz” child inside the database. It is limited to 50 of the database entries but only the last 50 entered. This means that only the last 50 data entries can be showcased within the RecyclerView. The second part of that snippet is the building of FirebaseRecyclerOptions. This is in order to configure to the adapter

2.7.4.5 RecyclerAdapter

```
FirebaseRecyclerAdapter FBRA = new FirebaseRecyclerAdapter<Question, QuestionViewHolder>(options) {

    @Override
    public QuestionViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(parent.getContext())
                .inflate(R.layout.question_layout, parent,  attachToRoot: false);

        return new QuestionViewHolder(view);
    }
    @Override
    protected void onBindViewHolder(QuestionViewHolder holder, int position, Question model) {
        holder.setQuestion(model.getQuestion());
        holder.setAnswerOne(model.getAnswerOne());
        holder.setAnswerTwo(model.getAnswerTwo());
        holder.setAnswerThree(model.getAnswerThree());
    }
};
```

The FirebaseRecyclerAdapter is for is what binds the query to the RecyclerView. So, after the RecyclerOptions is built, the above snippet happens. A RecyclerAdapter object is created called FBRA. There is a custom class included here called QuestionViewHolder. This makes it so that a View object called view is equal to a LayoutInflater of the question_layout xml file. The onBindViewHolder() then binds the question_layout objects to the objects in the Question class Data Model. The code to get the objects in the question_layout page is at the bottom of this page of code in subclass called QuestionViewHolder().

2.7.4.6 QuestionViewHolder

```
private static class QuestionViewHolder extends RecyclerView.ViewHolder {
    View mView;
    public QuestionViewHolder(View itemView) {
        super(itemView);
        mView = itemView;
    }
    public void setQuestion(String sQuestion) {
        TextView question_content = (TextView) mView.findViewById(R.id.questionTxt);
        question_content.setText(sQuestion);
    }
   public void setAnswerOne(String answerOne) {
        RadioButton answerOne_content = (RadioButton) mView.findViewById(R.id.answerOne);
        answerOne_content.setText(answerOne);
    }

    public void setAnswerTwo(String answerTwo) {
        RadioButton answerTwo_content = (RadioButton) mView.findViewById(R.id.answerTwo);
        answerTwo_content.setText(answerTwo);
    }

    public void setAnswerThree(String answerThree) {
        RadioButton answerThree_content = (RadioButton) mView.findViewById(R.id.answerThree);
        answerThree_content.setText(answerThree);
    }
}
```

In this subclass, there are multiple methods. These methods create objects based on what is in the question_layout file (One TextView, Three RadioButtons). Each one of those objects is then equal to a corresponding object in the question_layout file. This information is then used by the RecyclerAdapter to bind the query to the Data Model.

The examples I have given here relate to just the Quiz maker functionality. But the methods used in the code (RecyclerView, FirebaseUI, Data Model) are all used for the EnterIssues functionality. That has a similar function where information is entered into the Database and displayed in a RecyclerView by replacing holder text in a layout xml file.

All of this code is from the Doctor version of the app, where the entering of information is done and the displaying of information. For the Patient version of the app, there is just a display of the information. To do this you use the same code above but none of the EditText .push() code.

2.7.5 Google Maps

There is a functionality that allows for the display of certain specified locations. The location information is entered from the Doctor version of the app. This information is then displayed on the MapActivity in the Patient version of the app.

```
private DatabaseReference mDatabase;
```

A DatabaseReference object is created called mDatabase.

```
mDatabase= FirebaseDatabase.getInstance().getReference().child("Locations");
```

This object is then equal to an instance of the child “Locations” inside the Firebase Database

```
private void saveLocationInformation(){
    String name =editTextName.getText().toString().trim();
    double latitude= Double.parseDouble(editTextLatitude.getText().toString().trim());
    double longitude= Double.parseDouble(editTextLongitude.getText().toString().trim());
    LocationInformation locationInformation =new LocationInformation(name,latitude,longitude);
    mDatabase.child(name).setValue(locationInformation);
    Toast.makeText( context: this, text: "Saved to Database",Toast.LENGTH_LONG).show();
}
```

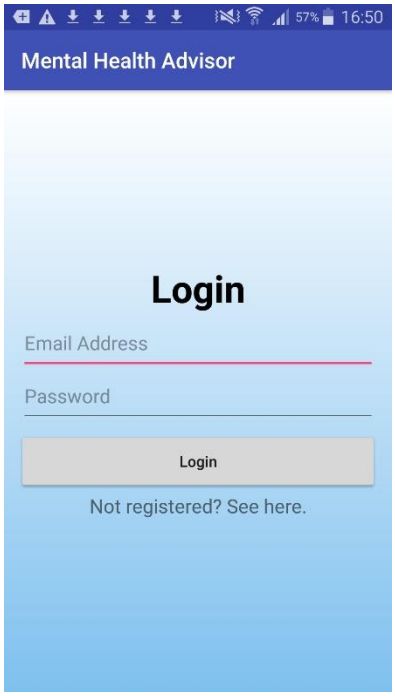
This method contains the code to save the location information to the Realtime Database. Instead of the .push() method to save the information, the .setValue() method is used, with the value being the object called locationInformation which contains all information entered into the EditText fields located in a specified layout

xml file. The child created in the database to contain these values is equal to the value entered into the EditText field that equals to the object name.

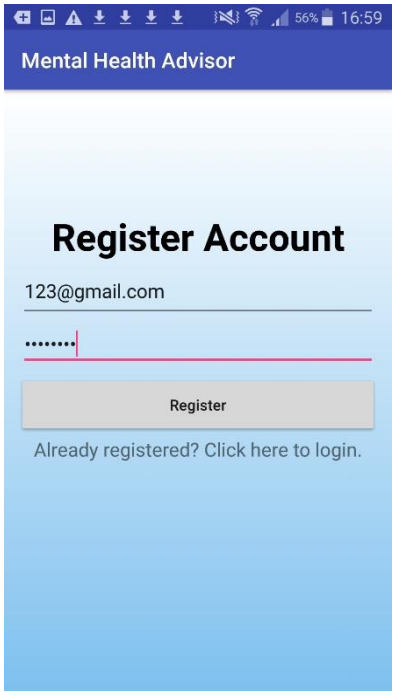
```
private LatLngBounds IRELAND = new LatLngBounds(
        new LatLng( v: 50.999929, v1: -10.854492), new LatLng( v: 55.354135, v1: -5.339355));

@Override
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;
    googleMap.setOnMarkerClickListener(this);
    googleMap.setMapType(GoogleMap.MAP_TYPE_NORMAL);
    mMap.moveCamera(CameraUpdateFactory.newLatLngBounds(IRELAND,  R: 0));
    mLocations.addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            for (DataSnapshot s : dataSnapshot.getChildren()){
                LocationInformation localInfo = s.getValue(LocationInformation.class);
                LatLng location=new LatLng(localInfo.latitude,localInfo.longitude);
                mMap.addMarker(new MarkerOptions().position(location).title(localInfo.name)).setIcon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_CYAN));
            }
        }
```

This is from the Patient app. The top two lines of code are to make sure the map centres on Ireland and not the entire world. The method that follows, “onMapReady()” is the method where the locations within the Database stored there from the Doctor app, are placed on the map. The object, “mLocations” is the DatabaseReference object for this class. It’s equal an instance of the “Locations” child in Firebase. Therefore, the locations are marked using marker symbols set to the colour hue of cyan.

2.8 Graphical User Interface (GUI) Layout

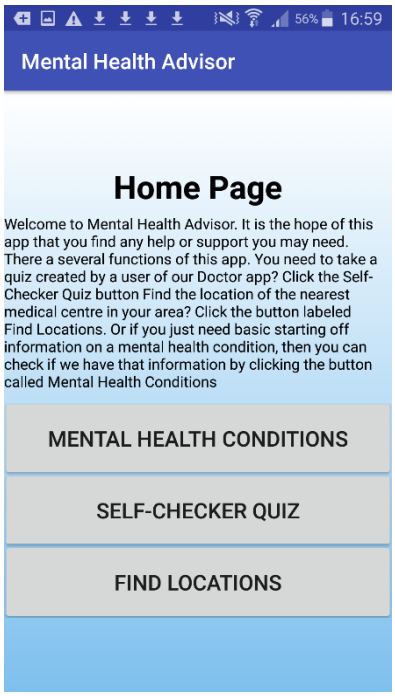


The above screenshot shows Login page for this project. This page is the same for both version of the app. The users will need to first register a username and a password before they can use this functionality. The user enters in their email and password. After checking they are in the database, the user is logged into the app.

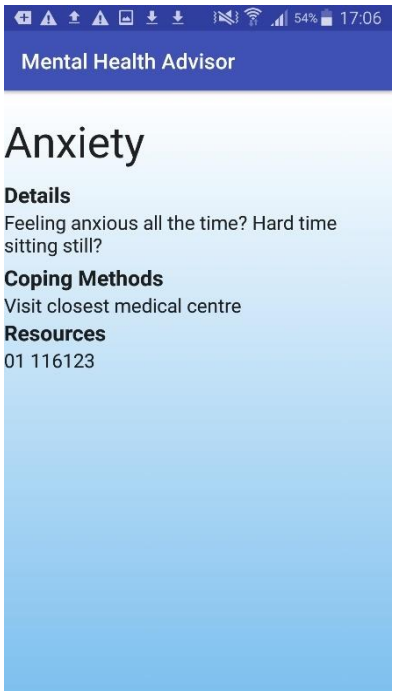


This is the Register page. This page is also the same for both versions of the app. The user enters in the email and password they want to use for the app. It is then submitted to the database where it is store for future login purposes.

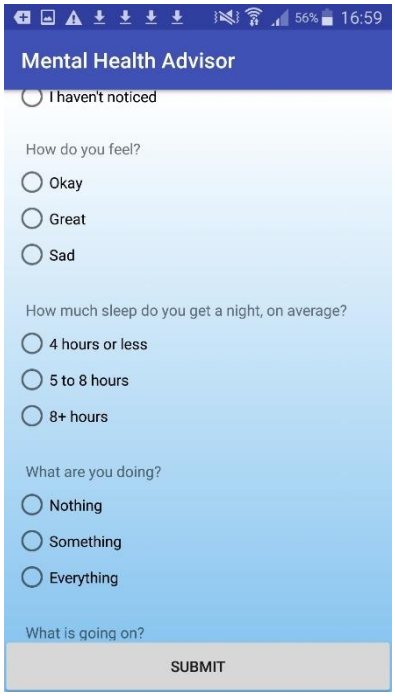
2.8.1 Mental Health App – Patient Version



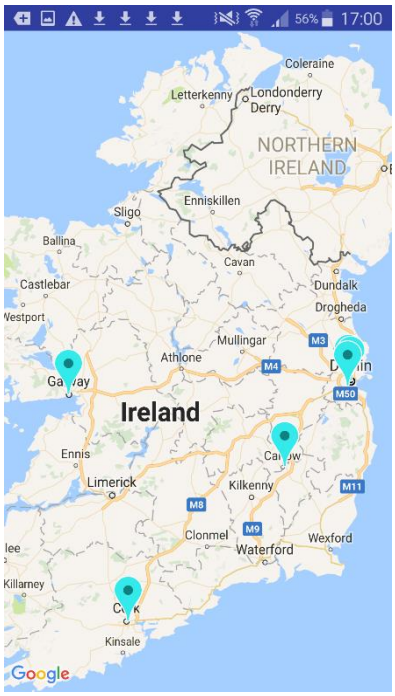
The screenshot above this is of the Homepage for the Patient version of this app. There is a brief description of what the app has in it. There are three buttons. Each button goes towards a different functionality. Mental Health Conditions goes to a page with a list of conditions, Self-Checker Quiz goes to the page with the quiz that users can take, and Find Locations goes to the Google Maps functionality and displays locations on that map.



This displays the list of mental health conditions and information on each. Every time information on a mental health condition is added to the Database via the Doctor app, it is added to the list on this page. Users can find out basic details about conditions here and resources on how to find out more.

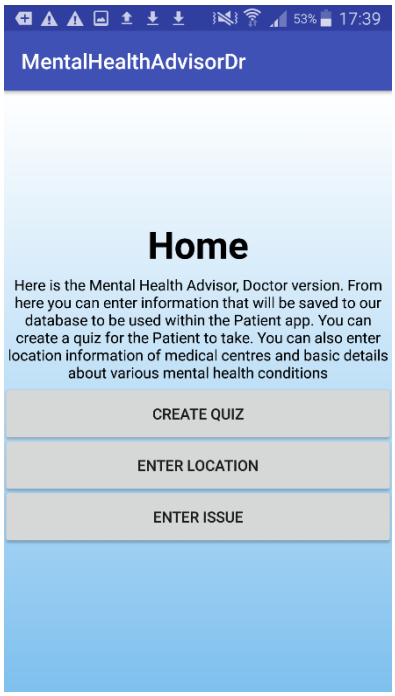


This displays the Self-Checker Quiz functionality. The questions that are entered into the database from the Doctor app are pulled from the Database and displayed within the list on this page. There is a submit button at the bottom that is supposed to submit the answers chosen to the database when the functionality is fully working.

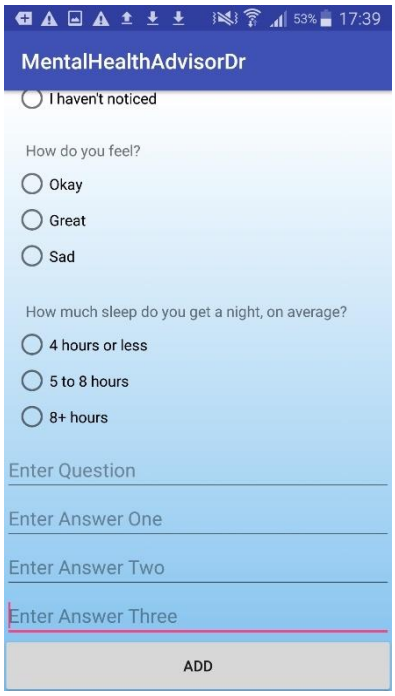


This screenshot above displays the Find locations functionality. This is a Google Maps functionality that highlights certain locations by placing markers on them. These locations are stored in the Database when someone enters in that information from the Doctor app. It is then pulled down from the Database by this app for the purpose of this functionality.

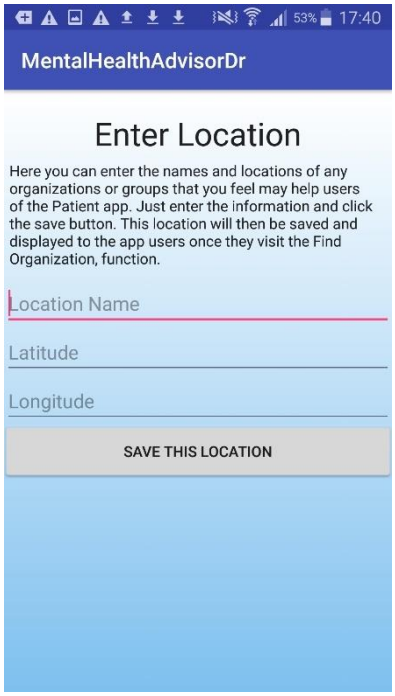
2.8.2 Mental Health Advisor – Doctor Version



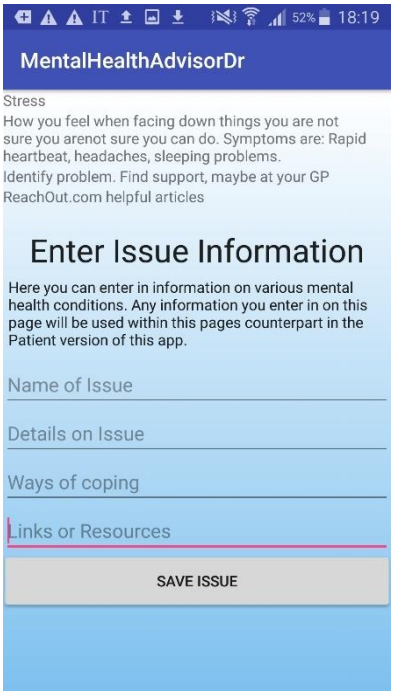
This is the homepage of the Doctor Version of the app. There is a brief introduction to the app explaining the functionalities that make up the app. There are three buttons. Create Quiz takes you to the page where you can add question information to the database for use in the Patient app. Enter Location takes you to the page where you enter location information for use with the Google Maps functionality. Enter Issue will go to the page where you enter in the information on specific mental health conditions.



This page is the Create Quiz page. There are Edit Text fields to type in and a button to add it to the Database and to add it to the quiz. You enter in the question and its answers before hitting Add. There is a preview of the quiz shown above the Edit Text fields.



This above screenshot is the Enter Location page. There is a brief description of what happens on this page. You enter in the place name along with the longitude and latitude into the Edit Text fields provided. You press the save button and it is stored within the Database for future use.



This page is the Enter Issue Information page. You enter in the name of a condition, a brief description of it, ways to cope with it and any resources for it in to the provide Edit Text fields. You then click the Save Issue button to save it to the database for later use.

This displays the general look of the Self Checker page. The page will display all of the questions in the test. At the end of the page will be a submit button. After answering all of the questions, the user will press the submit button and the results will be saved to the database. The app will redirect the user to an appropriate mental health problem based on their answers

2.9 Testing

In order to test this project, several versions of Android mobile devices were used. There was a variety in order to test that the two apps being developed

would work on them and not be limited to only working on one kind of Android Device.

1. The main device used for testing in this project is a Samsung Galaxy S5. It is a smart phone with an Android version of 6.0.1. As this is the main device used for running the apps, they are both tested on it constantly. The apps are run on the device multiple times a day, in order to test any new or modified functionality. As such the functionalities of the app work fully on this device.
2. The second device that the apps were tested on was the Nexus 5x. The app was tested on the Emulator version of the device. The apps appearance appeared as they should on this devices screen, with no cut off elements at the side or unnecessary stretching.

For this project, the problems with internet connectivity were also tested. A big part of this app is done using an online cloud database. Therefore, it is necessary to determine what would happen if either of the apps lost their connection to the internet and with that, connection to the database.

Mental Health Advisor – Doctor Version

The effects of no internet activity were tested an all functions within the Doctor app. Firstly, from the login screen, if you lose connectivity before you can log in, you are unable to access the app until you find an internet connection. This is the same for the Register function.

If you are already logged in when you lose connection, you can still access all the pages from the Homepage. While you can access the pages, any information taken from the database will not appear on the pages.

With no internet, you can still enter information into the text fields and submit it. But the information will not appear on the Patient app or in the database at that time. However, if you then reconnect to the internet after submitting information when there was no internet, that information will suddenly be added to the database.

No connection to the internet affects the Patient app more so than the Doctor app. This is because the Patient app pulls down more of its content from the Database than the Doctor app.

Like the Doctor app, if you are not logged in already before you lose internet connection, you cannot interact with the apps functions. The same goes for the Register function.

If you lose your connection while on the Homepage, then every time you got to one of the apps functions, none of the information from the Database will be there. But if you lose connection while on the page of one of those functions, the Database information displayed there will stay until the page is refreshed.

2.10 Customer testing

There was no User testing done for this app. It was thought that the subject matter would be too sensitive to include real people in the testing process as it is impossible to fully know a person's state of mind beforehand.

2.11 Evaluation

This system was evaluated by testing every functionality on several machines. The submission of data, the display of data on the screen and the ease of use on different devices was noted within the testing. It was concluded that the functionality of both apps worked fine on other devices. There didn't seem to be any hindrance by it. All functions worked perfectly. The EnterLocation functionality was able to submit information to the Database from all the devices and so were the other functionalities in the Doctor app. Not only that, but the Patient app was able to receive data from the Database on every device tested, with no hassle detected.

There were also no display problems that were noticed. No sentences were cut off by the screen, or background not covering the whole screen due to being too small for it. The text size stayed consistent, with no noticeable changes at all in

the words on the screen the same can be said for the input boxes in the Doctor app. They didn’t fluctuate in size and stayed consistent throughout.

3 Conclusions

From doing this project, I have learned a lot. There are skills that I have now that I didn't have at the beginning of the project. I was a novice at Android Studios at the beginning of this project, having never really used it outside of one or two instances. After the project, while I'm definitely no expert on it, I am much more confident in my skills with Android Studio and Mobile App Development than ever before. I have actually grown to like developing apps with this program and would love to do more of it in the future.

I've learned a great deal about Google Firebase and how to push and pull data to a cloud-based database in general. Again, this was something I had never used before but was not hard to pick up. This is another

3.1 Advantages

The data is stored within a Realtime database. As the name suggests, the data on the screen can be updated in Realtime. This is a great feature as today's mindset is all about fulfilling your request as soon as possible so updating in real time plays right into that mindset. Firebase Authentication is another secure service used in both of these apps for storing login information. It is managed by Google who are one of the largest tech giants in the world. They are more than capable of keeping track of data and keeping it secure.

3.2 Disadvantages

Not all of the functionalities were completed during the 9 month timeframe of work. The quiz maker can only make one quiz at a time and, while users of the Patient app can answer the questions in the quiz, they can't submit them to the database. Also, the login credentials work regardless of what app they are used on. As long as they match any credentials in the Firebase Authentication database, you can log into either of the apps.

3.3 Opportunities

There is the chance to expand the project and create apps for other mobile operating systems like iOS for the Apple devices. Expanding to different types of devices would increase the number of potential users for each app.

Firebase has some upgrade features to use if you pay for them. If I had some start-up money I could expand and use those upgrades.

3.4 Limits

As stated, only one Quiz can be created at a time. That functionality isn't finished yet. So, you are limited to just one quiz. Also, the apps will not work fully unless there is a strong internet connection. Without one, there is very limited functionality.

A limit right now is that the apps only work on Android devices. They are only native to Android and therefore are restricted from working on the likes of iPhones and the like. This can be fixed in the future, however.

Overall, throughout this project I have come out of it with more knowledge than before and a finished product, even if not all of it is finished. The limits could be worked on in the future and the disadvantages are partly tied to those as well.

4 Further development or research

With more resources, this project could hopefully be completed to the standard it deserves. With more time and better access to code resources, some of the functionalities in this project that are lacking or not finished could potentially be fixed and made functional.

When that is done, there could be time to expand the projects scope to include other mobile operating systems. There could be versions of the apps for iPhone users to use. This would expand the market of people willing to use the apps.

More research could be done into mental health conditions, or the effect of mental health applications on mental health itself. This would vastly help the information functionality of the apps as the more accurate the data in the app, the more appealing it will seem.

There is a possibility to expand the scope of the project functionalities themselves with more resources. Make improvements on the existing functions is one thing but adding more advanced functionality could be a possibility if given the chance with more time and resources at my disposal.

5 References

GitHub. (n.d.). *FirebaseUI for Android — UI Bindings for Firebase*. [online] Available at: https://github.com/firebase/FirebaseUI-Android [Accessed 15 Oct. 2017].

Grist, R., Porter, J. and Stallard, P. (2017). *Mental Health Mobile Apps for Preadolescents and Adolescents: A Systematic Review*. [online] Available at: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5465380/ [Accessed 11 May 2018].

sourcemaking.com. (n.d.). *Use Case Diagrams*. [online] Available at: https://sourcemaking.com/uml/modeling-business-systems/external-view/use-case-diagrams [Accessed 10 May 2018].

yourmentalhealth.ie. (n.d.). *Mental Health Problems*. [online] Available at: http://www.yourmentalhealth.ie/about-mental-health/common-problems/mental-health-problems/ [Accessed 10 Oct. 2017].

6 Appendix

6.1 User Manual

6.1.1 Mental Health Advisor – Doctor Version

6.1.1.1 Register with app

- Download app from where it is stored
- Open app on your Android Mobile Device and after a few seconds, a Login screen should appear.
- To go to the Register page, click the link provided on the Login page
- Enter in the email address and password you want to use for this app into the fields provided.
- Click the Register button and your information will be saved to the Database for future login.
- The app will also log you in as part of registering you and you will be taken to the Homepage.

6.1.1.2 Login to app

- Open app on your Android Mobile Device
- After a few seconds, the Login page will appear
- Enter in the email address and password you used when registering with the app, into the given text fields.
- Click the login button once your information has been entered.
- Once the app has verified that your login information matches any that is stored within the system, you are taken to the home page.

6.1.1.3 Create Quiz

- Once logged into the app you are presented with three buttons that take you to different functionalities of the app.
- Press the Create Quiz button to set questions for a quiz
- The app takes you to the Create Quiz page

- Enter the question you want to add to a quiz, along with the answers.
- Once all the information you want is entered, press the Submit button
- This information you have entered is displayed on the screen in front of you within a question format.
- Keep adding questions until you are satisfied with the quiz

6.1.1.4 Enter Location

- Back on the Homepage, press the button called Enter Location.
- The Enter Location page will be displayed with the text fields needed to enter your information into.
- Enter in the name of a place you wish to add the location of to the database, along with the location it is found at. (You will need to know the longitude and latitude of a location in order to enter it into the database.)
- Once the information is entered, press the Submit button.
- The information is saved to the database for later use.

6.1.1.5 Enter Issue Information

- From the Homepage, click the Enter Issue button.
- You will be taken to the Enter Issue page
- From this page enter in the information for a specific mental health condition (name, description, coping tools and any links or resources to help with that.)
- Once the information is all entered, press the Submit button.
- The information is stored into the database.

6.1.2 Mental Health Disorder – Patient Version

6.1.2.1 Register with app

- Download app from where it is stored
- Open app on your Android Mobile Device and after a few seconds, a Login screen should appear.
- To go to the Register page, click the link provided on the Login page

- Enter in the email address and password you want to use for this app into the fields provided.
- Click the Register button and your information will be saved to the Database for future login.
- The app will also log you in as part of registering you and you will be taken to the Homepage.

6.1.2.2 Login to app

- Open app on your Android Mobile Device
- After a few seconds, the Login page will appear
- Enter in the email address and password you used when registering with the app, into the given text fields.
- Click the login button once your information has been entered.
- Once the app has verified that your login information matches any that is stored within the system, you are taken to the home page.

6.1.2.3 Take Self-Checker Quiz

- From the Homepage, click the button called Take Self-Checker.
- You are taken to the Self-checker page where a number of questions and answers will be displayed, along with a submit button
- Select the answers that best suit you based on the question. (One answer per question)
- After you press the Submit button, your answers should be added to the Database.

6.1.2.4 Find Location

- From the Homepage, click the button called “Find Locations”
- You are taken to a page displaying a Google Map of Ireland with markers shown on it.
- Choose the marker most convenient for you and press it.
- The name of the place is show to you via an info window.
- Now you know the name of a medical center and its location on a map.

6.1.2.5 Display Issues

- From the Homepage, click the button labeled Mental Health Conditions.
- You are taken to a page that displays the mental health condition information held within the Database.
- Browse through the information to see if the one you are looking for is there.
- If you find it, there is a brief description of the condition as well as the beginnings of how to cope with it and links or other resources to contact others for more information

6.2 Project Proposal

6.2.1 Objectives

The goal of this project is to provide a way for those with mental health problems, or those who suspect they may have one, to, first, self-check themselves to make sure they may have a problem. If it turns out they do in fact have a mental health problem, they can then seek help from various organizations. Also, basic information on how to manage the problem will be provided for them. In order to achieve all of this, the project will be in the form of an Android mobile app.

Users will be able to register a username and password. They can then use these to login to the app. This will add more security to the app seeing it will be dealing with mental health issues and they can be very private for most people.

The application will provide users with a way to search for mental health support organizations and groups in their local area. Users will then be supplied with basic information on that organization: location, what they do, how large they are. The interface though which the users shall look up the locations will be some sort of Google Map with locations stored in a database.

The users of this application will be able to leave comments on the information pages of specific organizations. These comments will hopefully detail their experiences with the said organization and provide future users with more insight into the organization when they themselves are looking into them

The target users of my app will, of course as stated above, people with suspected or already known mental health disorders/problems. Seeing as it will be an Android app, it will be for those with Android smart phones.

(Max. 1 Page)

6.2.2 Background

Start of Idea

Initially, I had come up with a different idea for a project, but that idea didn't pan out. Then, after looking through a list of project ideas, an idea for a mental health stuck out from the rest. There are many people out there who suffer from Mental Health problems and don't know how to seek help or deal with their problem. This application would be another tool in helping those people find the help and support they need.

Research

After confirming that this would be the basis of my app, I started doing research into it. I research various Mental Health problems such as, depression, panic attacks, anxiety, Bipolar disorder, and just plain stress. I found information on the types of symptoms for each disorder/problem, and tips and ways in which they can deal or overcome the problems. A website I found called yourmentalhealth.ie had a lot of this information. It had descriptions of several disorders/problems and their symptoms. They have a movement called #littlethings, and on their website, is a list of posters with slogans. One of the sayings, "Problems feel smaller when you share them", was what gave me the idea for one of the main functionalities for the project. This website was a good starting off point in my research.

Another website found for information was the website for Mental Health Ireland. This website gave further insight into various mental health issues and their symptoms. The website also goes into detail on the five ways to wellbeing, which are, Connect, Be active, Take notice, Keep learning and Give. These are five things that can be done in order for those with mental health issues, or even just people in general, to function well and feel good about themselves.

For this project, research was done into a database to store the Mental Health Advisor's data. The database chosen was the Google Firebase cloud database. Through researching, it was discovered that Firebase is a cloud-based database system that can be used for an Android platform. It is a NoSQL database that stores its information in a JSON format. Because the information is in a JSON format, there is no need for the information to be stored in rows and columns of tables. The information is stored in JSON objects instead.

6.2.3 Technical Approach

For the project I am going to need a way to store data inputted from the user, as well as information that will be displayed to the users. Various different cloud storage services have been looked into and the one that was decided on is Google Firebase.

Google Firebase was chosen for several reasons, one of the main ones being that, it looks like integrating it into Android Studio seems to be easy enough to do. Firebase also seems to have a sizable community of people to help support it. I have never used Firebase before so there will be some learning for me to do while the project goes on and develops.

This project will also involve providing the users the locations of various mental health support agencies and organizations in their local area. For the application, it has been decided that the best way to display this information would be through an interactive map showcasing the locations. The best service to use to provide this type of functionality seems to be Google Maps API service. This API will provide the application with the ability to leave markers at the location data for the specified organization. Also, the Google Maps API can be used in conjunction with a database. It can receive location data from a database and mark that location on the map. This relationship between the Map API and the database will be crucial to some of the applications functionality.

First, before starting to code the application, wireframes for the basic layout of the application's pages will be created. After the wireframes are created, Android

Studio will be the coding environment used to create a basic UI for the app. This will be the basic prototype that can be presented at the midpoint presentation

6.2.4 Technical Details

IDE and Language

This application will be built using Android Studio 2.3.1 and will be developed solely for an android platform. The language I will be using to develop this application is the Java programming language.

3rd party libraries

One of the 3rd part libraries that will be used is Google Firebase. Firebase is a tool that will handle any of the data storage that is needed for the application.

Version Control

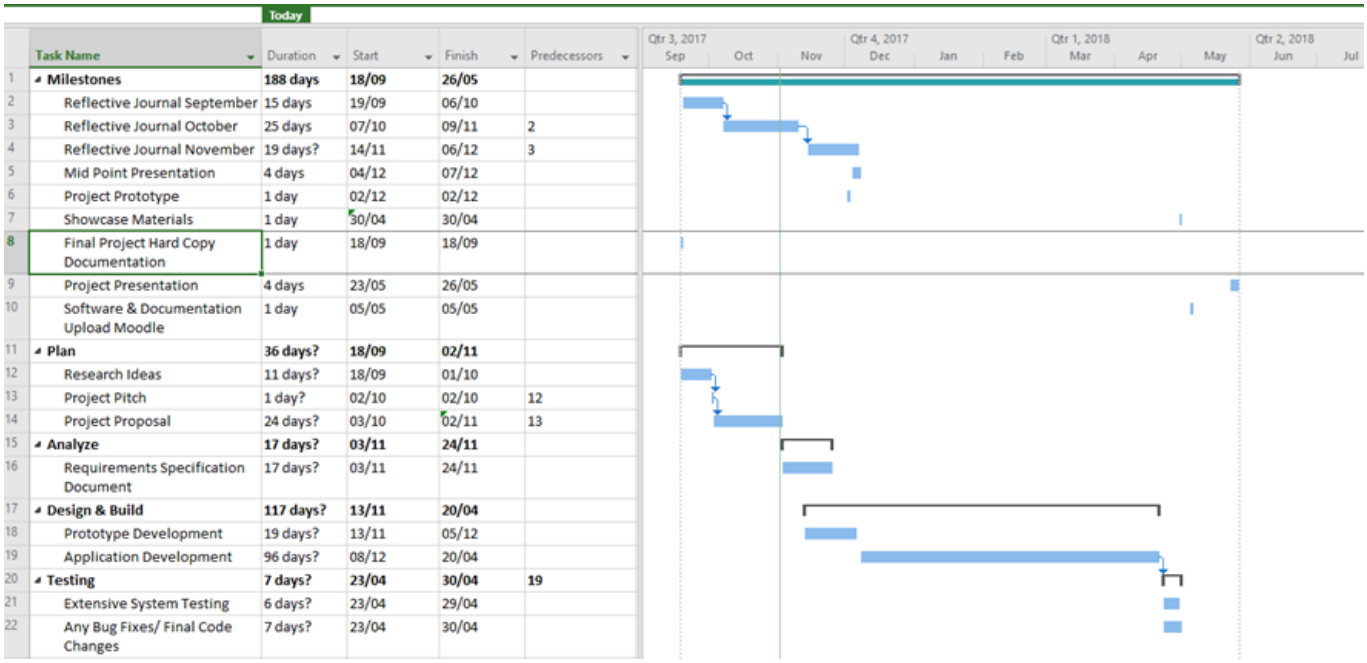
For version control while developing this application, I will be using GIT. The code for the application will be stored in a remote repository.

6.2.5 Evaluation

There will be rigorous testing done using my main Android device. The testing will continuous as the applications features are being added. On top of the main device, I will also be using the inbuilt emulator that Android Studio has to test the application on numerous different Android devices of varying sizes.

6.3 Project Plan

6.3.1 Gantt Chart



	Task Name	Duration	Start	Finish	Predecessors
1	**Milestones**	**188 days**	**18/09**	**26/05**	
2	Reflective Journal September	15 days	19/09	06/10	
3	Reflective Journal October	25 days	07/10	09/11	2
4	Reflective Journal November	19 days?	14/11	06/12	3
5	Mid Point Presentation	4 days	04/12	07/12	
6	Project Prototype	1 day	02/12	02/12	
7	Showcase Materials	1 day	30/04	30/04	
8	Final Project Hard Copy Documentation	1 day	18/09	18/09	
9	Project Presentation	4 days	23/05	26/05	
10	Software & Documentation Upload Moodle	1 day	05/05	05/05	
11	**Plan**	**36 days?**	**18/09**	**02/11**	
12	Research Ideas	11 days?	18/09	01/10	
13	Project Pitch	1 day?	02/10	02/10	12
14	Project Proposal	24 days?	03/10	02/11	13
15	**Analyze**	**17 days?**	**03/11**	**24/11**	
16	Requirements Specification Document	17 days?	03/11	24/11	
17	**Design & Build**	**117 days?**	**13/11**	**20/04**	
18	Prototype Development	19 days?	13/11	05/12	
19	Application Development	96 days?	08/12	20/04	
20	**Testing**	**7 days?**	**23/04**	**30/04**	**19**
21	Extensive System Testing	6 days?	23/04	29/04	
22	Any Bug Fixes/ Final Code Changes	7 days?	23/04	30/04	

This Gantt Chart highlights the ideal timeline for all the tasks in the project. The chart is broken into sections. The first section, Milestones, is for all of the big deliverables of the project. The Plan section is for the activities that will go towards the actual planning of the project. The Analyse section is for the activities that will further plan and devise the functions and parts of the project. The Design and Build section is for the activities that will go towards the creation of the project's application. The Testing section holds the testing activities for the project.

Reflective Journals

The reflective journal tasks include creating a journal for each one detailing everything that was done towards the project during that month.

Midpoint Presentation

The Midpoint presentation task is where the project is presented to lecturers for feedback.

Project Prototype

The Project Prototype task is where a prototype of the project is created for the midpoint presentation.

Showcase Materials

The showcase material task is a milestone that involves gathering the materials needed for the project showcase at the end of the project timeline.

Project Presentation

The Project presentation milestone is when the completed project will be presented in full.

Software & Documentation Upload

This milestone is where the finished project code and documentation will be uploaded.

Research Ideas

This task is where the initial ides for the project were researched. Each idea for the direction of the project was researched and the best one was chosen.

Project Pitch

The project pitch is where the initial project idea is pitched to a panel of lecturers. They suggest changes and ideas for the project.

Project Proposal

The Project Proposal is the initial document created at the start of the project. It will be the foundation for the technical report.

Requirement Specification

The Requirement Specification is the second major document that will be done for the project. It will detail the functional and non-functional requirements of the project.

Prototype Development

The prototype development task is an estimate towards the amount of time it will take to develop the prototype app for the project.

Application Development

The application development task is an estimate for how long will be needed to get the end application fully functioning for the project.

Extensive System Testing

This task is where the system will be rigorously tested for bugs and problems. The time is an estimate of how long it may take.

Bug Fixes/Final Code Changes

This task is where any bugs that were found in the system testing will be fixed. Also, any last minute code changes will be made during this task.

6.4 Monthly Journals

6.4.1 Reflective Journal Month 1

Student name: **Curtis Murphy**

Programme: **BSc in Computing**

Month: **September**

September 18th

I've done research into a mobile app idea. The user would be able to choose a location they are going to on a Google Map in the application. They would then choose a person in their contacts and once the user reached the location they chose; the message would be sent to the selected person. The message would say something like "I have arrived safely". This is so users won't have to worry about sending word when they get to a place, or in case they forget to send a message.

September 19th

I have researched my idea a bit more. I looked up other applications that do something similar. I found an Android app called Buzzer that seems to do a lot of what I was researching for my idea. It also has one or two ideas that I didn't think

of for my idea. I came across other similar apps too. Some of which hadn't been kept up to date and maintenance.

September 20th

I spoke with Dominic about my idea today. He said it was a bit too simple for a final year project in 4th year. After looking over the project again I kind of agree with him. He suggested maybe finding some way to build on the idea, to make it more complex. After my talk with him I decided to start researching ways to make my idea more complex

September 22nd

I spent the rest of the week researching ways to make my idea more complex. One idea I came up with was to add a tracking element to the app. Having it so that the you can track your progress on your route to your location or so that the person you want to send the message to can track your progress. I found some links on how to do geotracking. I'll look into the idea more over the weekend.

September 25th - September 27th

I researched my original idea more over the weekend. But while I was doing that I decided to also research more IoT related project ideas. I researched specifically IoT projects having to do with a Raspberry Pi seeing as we were given Raspberry Pi's this week. I looked into different Pi sensors, like motion, sound and temperature, and how they can be used for projects.

September 28th - September 30th

I've put my original idea to the side for now and focused more on researching more IoT related projects. One thing that interested me was a home alarm system using a Raspberry Pi. The past few days, I've done some research into using a special camera made for Raspberry Pi to monitor a home, detect motion and sound then send images from the Raspberry Pi camera. I could use an infrared sensor to detect the motion. I'll research the idea a bit more before my Project Pitch.

October 2nd

I had my Project Pitch today. I was rejected. I went in with my motion sensor idea and was told it was too simple. Dominic said it was not complex enough to be a 4th year end of year project. He also said it wasn't an easy project to improve upon with more complexity. I'm disappointed that my project was rejected but am glad for the feedback. I may look into making my idea a bit more complex, but I am going to go back to looking for other ideas to do for my project.

6.4.2 Reflective Journal Month 2

Student name: **Curtis Murphy**

Programme: **BSc in Computing**

Month: **October**

October 3rd – 8th

For the rest of the week after the Project Pitch, I researched other project ideas, including the ones on the project list provided on Moodle. I narrowed the possible ideas down to two, a portal to help with the process of assigning project ideas to students, and an application to help those with Mental Health problems. I have a few ideas for both projects, but I have decided to go with the project assigning portal project. I will email Dominic about taking this idea, as he was the one who submitted it to the list.

October 9th

I emailed Dominic about using his project assigning portal idea. He got back to later the same day and unfortunately, another student had already taken that idea as their project. Thankfully I had the mental health application as a backup project. I have replied to Dominic's email asking about doing the Mental Health idea and am waiting for his reply.

October 10th

Dominic got back to me about the mental health idea. He said I was clear to use it. So now I can go ahead with the project and create an application to help those

with Mental Health Problems. I was thinking of doing the project as a mobile application for Android devices. I'll continue looking into that as an option.

October 11th – 15th

I have started putting more research into my project now that I have a solid idea. This includes doing my project proposal. Also, I got assigned my Project Supervisor for this Final Year Project. My supervisor is Anu Sahni. Anu seems like a great supervisor for me. She's my lecturer for Mobile Application development and I've been thinking of doing this project in the form of an Android mobile application. So, she may be able to help me and advise me on what is best to use for the project. I have sent her an email to set up our first meeting and am now just waiting for her reply.

October 16th- 18th

Anu replied to my email, and we have decided on meeting next Thursday after our Mobile Application Development lecture. I have now started getting ready for that meeting. I have continued working on my Project proposal and researching what technologies I may need for the project. I have been looking using Google Maps in my app to display different Mental Health organizations.

October 19th

I had my first meeting with Anu. It went well. I told her my idea and she liked it. She suggested I look into TripAdvisor for my Google Maps functionality. TripAdvisor has it so that its users can leave comments about the places marked on the Google Maps. Anu suggested looking into that for my app and I agree it sounds like a good idea. I have download the TripAdvisor app and am becoming familiar with its Google Maps functions

October 20th- 25th

I have done more work on my Project Proposal. I have also done more research into what other functionality I can add to the application. I have looked into including a login/register feature to the application for privacy. I looked into other Mental

Health apps that are out there. I also did more research into using the web service Firebase, as a Realtime database for the app.

October 26th – Nov 3rd

I have continued my work on my Project Proposal. Had my second meeting with Anu. She suggested I look into several mental health quizzes online. The goal is to create a self-checker test for the finished application. I researched several tests online to see the types of questions that are asked. After researching them, I have started to create a basic layout of what the app will eventually look like. After that, I have started creating the self-checker page and the Mental Health information pages for the app using Android Studio.

Nov 6th – Nov 10th

I finished a basic outline of what the finished could look like. I also did more research into exactly how to integrate a Firebase real-time database into an Android Studio's project. I met with Anu again. We talked more about the functionality of my app. She suggested I look into if there are any Mental Health API's out there that I could implement into my app. She also told me what aspects of the app I should focus on for the midpoint presentation. She said I should focus on creating the self-checker for one mental health problem to start with, as a proof of concept. I agree with her suggestion.

6.4.3 Reflective Journal Month 3

Student name: **Curtis Murphy**

Programme: **BSc in Computing**

Month: **November**

Nov 11th to Nov 29th

Met with Anu a bit but unfortunately couldn't meet with her the week before my Mid-point presentation. As the Mid-point is coming up soon, I have been focusing on getting the basic UI of my app done. I have also been focusing on the document and will continue to do so up until the upload.

I hope to have all of the pages of my Prototype app linked together for the Mid-Point. I worked on getting the layout of the pages done for this Presentation and left the large functionality for afterword. I'll start back on that after the Mid-Point.

Because Anu is away, I will have Simon Caton as an examiner for this Presentation. Never met him. Disappointed that there will be no one there that knows my project, but it can't be helped. Just have to hope for the best

Nov 30th – Dec 15th

Worked mainly on the document up until the mid-point. Did a bit of work on the prototype but it was mainly UI related.

Had my Mid-Point Presentation. It went well. Simon gave me some good insight into my project. I definitely need to restructure how I develop it. He gave me some advice on testing too that will be helpful

6.4.4 Reflective Journal Month 4

Student name: **Curtis Murphy**

Programme: **BSc in Computing**

Month: **January**

January 14thto January 24th

I've done some work over December and the first part of January but not much. After the Mid-Point I've been focusing on finishing my other projects and then was studying for my final exams. Now that the exams are coming to an end soon I'll have more time to work on this project again.

January 25th – February 1st

Now that the exams are out of the way I have more time for this project. I've started redesigning the functionalities of the project. They need to be reworked. I've been thinking of some sort of secondary interface for entering in information into the database to be displayed in the application.

Anu's back. Meeting is scheduled with her. Hopefully she can give me more insight into how I should restructure my project.

6.4.5 Reflective Journal Month 5

Student name: **Curtis Murphy**

Programme: **BSc in Computing**

Month: **February**

February 2nd to February 16th

Spoke with Anu about my project. Has been decided that my project will now be two separate apps that work together. This is great as I was looking to use a secondary interface to enter in the information for the app. I can just use the second app for that.

February 17th to February 28th

After talking with Anu about the two apps idea I developed it some more. I think that one app will be for Patients, and one will be for Doctors. Ultimately the Doctor app will mostly involve with the entering and submitting of data to the Database, for use in the Patient app. The Patient app will mostly display the information with some intractability from the Map function and the Quiz

Started working on the Quiz maker part of the project. I know of a useful tool called FirebaseUI. It may be the thing for this project.

Did most of my work on the Quiz maker with some smaller work on the layout of the app. I started laying out the paths to different pages on both of the apps and the general look was thought of.

6.4.6 Reflective Journal Month 5

Student name: **Curtis Murphy**

Programme: **BSc in Computing**

Month: **March**

March 1st to March 17th

My main work has been done on the quiz Maker. The information goes to the database but doesn't show up on the screen. I've talked to Anu about it but haven't made much headway.

Tried different names, moving code around, googling it multiple times but no success. I'll have to move on and do other sections of the project. Don't want to neglect other parts.

March 18th to March 31st

Worked on other sections of the project. The Google Maps functionality wouldn't work. Turns out had to change API key. The Quiz maker still not working properly. Still trying to find out how to fix it. Been talking to Anu every couple of days. Still no luck fixing it.

Found out how to fix it. A unique key was being made when the information was added to the database. It was blocking the pathway for calling the data. Information shows up on screen now.

Nearly got the other parts of the project up to the same quality. Just need to keep working on it up until the end. Can't forget about the document though. Need to start doing that again.

6.5 Other Material Used

Any other reference material used in the project for example evaluation surveys etc.