

Final Project Report

Quartermaster

TU856

BSc in Computer Science

**Ethan Holohan**

**C20322553**

**Colette Kirwan**

School of Computer Science

Technological University, Dublin

**11/04/25**

Abstract

Airsoft is a popular and growing hobby with new people playing trying the hobby every year, however Airsoft is a complex game, with each site having its own sets of rules and games and as a result issues arise from players not fully knowing the rules and site layouts, leading to rule breaking and potential injury.

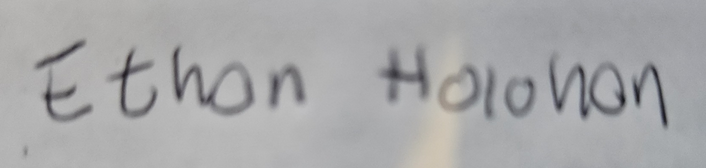
The Goal of this application is to help both newer players and experienced ones better understand the rules of each site and the games played there, as well as help staff enforce rules and maintain safety for players.

The way this app intends to do this is by making sure that all the information that players need for each game is available at all times during the game, by adding in a tracking feature that allows players to see where each other are for gameplay and for staff to see where all players are on the site, to help rule enforcement by using geofencing technology.

Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Ethan Holohan**

**11/11/2024**

Acknowledgements

I would like to thank my project supervisor Colette Kirwan for all the help and stellar guidance they offered on this report.

Id also like to thank the FYP course co-ordinator Jack O’Neill for all his work organising the FYP course this year.

Id also like to thank the Players at the Airsoft site I work with and my boss John for helping with testing and requirements gathering.

Table of Contents

[1. Introduction 7](#_Toc119925914)

[1.1. Project Background 7](#_Toc119925915)

[1.2. Project Description 7](#_Toc119925916)

[1.3. Project Aims and Objectives 7](#_Toc119925917)

[1.4. Project Scope 7](#_Toc119925918)

[1.5. Thesis Roadmap 7](#_Toc119925919)

[2. Literature Review 8](#_Toc119925920)

[2.1. Introduction 8](#_Toc119925921)

[2.2. Alternative Existing Solutions to Your Problem 8](#_Toc119925922)

[2.3. Technologies you’ve researched 8](#_Toc119925923)

[2.4. Other Research you’ve done 8](#_Toc119925924)

[2.5. Existing Final Year Projects 8](#_Toc119925925)

[2.6. Conclusions 8](#_Toc119925926)

[3. System Design 9](#_Toc119925927)

[3.1. Introduction 9](#_Toc119925928)

[3.2. Software Methodology 9](#_Toc119925929)

[3.3. Overview of System 9](#_Toc119925930)

[3.4. Other Sections 9](#_Toc119925931)

[3.5. Conclusions 9](#_Toc119925932)

[5. Prototype Development 11](#_Toc119925938)

[5.1. Introduction 11](#_Toc119925939)

[5.2. Prototype Development 11](#_Toc119925940)

[5.3. Other Sections 11](#_Toc119925941)

[5.4. Conclusions 11](#_Toc119925942)

[6. Issues and Future Work 12](#_Toc119925943)

[6.1. Introduction 12](#_Toc119925944)

[6.2. Issues and Risks 12](#_Toc119925945)

[6.3. Plans and Future Work 12](#_Toc119925946)

[6.3.1. GANTT Chart 12](#_Toc119925947)

[Bibliography 13](#_Toc119925948)

# 1. Introduction

## Project Background

Airsoft is a sport/hobby that is played all around the world, and it is very similar to paintball, with two teams fighting over an objective decided upon by game marshals, whose responsibility it is to make sure players are safe and that they are following the rules of the game. However, this is quite a challenge. Many players don't know the site well, and some sites are extensive. Many players get lost or confused throughout the game, which leads to a bad experience, and since there are not that many marshals compared to the number of players, it is hard to keep track of all players, which can lead to safety issues.  
Additionally, although Airsoft has been around for a while, there has been very little digital integration until recent years.  
  
This app aims to address these issues by giving marshals more tools to monitor the status of the game and its players and improve player experience by easing navigation and providing more information.

## Project Description

Quartermaster is a mobile application designed for Airsoft players and Airsoft Marshals. Its purpose is to give airsoft players more information so they can play the game better with less confusion over rules and to help marshals make sure that players remain safe throughout the game day. It intends to do this using new technology on smartphones to track users on the site and display accurate site maps of each site.

## Project Aims and Objectives

The main aims of the quartermaster project are:

* To enhance players understanding of the game and reduce confusion
* To provide a way of improving team coordination and encouraging team play
* To ensure that players always have a way of communicating with a marshal in an emergency.
* To Provide players with a easy method of tracking their stats.

The way i will know that these goal have been achieved is, Live User testing, multiple field tests will be carried out using the app and if users report that the app achieves its main goals well, then i will know that the goal has been achieved.

The main objectives of Quartermaster are:

* To implement a method of users being able to contact marshals across the site
* To have all the information about a game and the site being easily accessible through the application
* To have a way of tracking players stats through geofencing and GPS tracking.

To ensure that I achieve these objectives, I will use a robust set of requirements gathered from requirements gathering to have a set success criteria for this application.

## Project Scope

Quartermaster's Scope is focused on enhancing the Airsoft experience for both players and marshals, however for this project it is also important to understand what this app doesn't intend to do

In Scope:

* this app is intended to act as a tracking app during an Airsoft event
* this app is intended to help people navigate in a large Airsoft event
* this app is intended to help people understand the rules and objectives of Airsoft
* this app is intended to help enforcement of these rules

Out of Scope:

* this app is not intended to get rid of marshals or their role in the Airsoft experience
* this app is not intended to be used as a tracking app outside of the Airsoft arena

## Thesis Roadmap

In the rest of this report i will go through:

Chapter 2 – Literature Review: This chapter is concerned firstly with the project background and any existing solutions in order to get a good idea of what direction the project should go, then it will go into detail on research done on various technologies and related studies, it also looks at some previous projects for inspiration, and finally how i created my set of requirements for the application.

Chapter 3 – System Design: This chapter is concerned with how the app was designed using the set of requirements to create a good framework of how to create this app.

Chapter 4 – Software Development: This chapter is concerned with the development of the application, firstly in the prototype stage and secondly in the full fledged application, what changes where made etc.

Chapter 5 – Testing and Evaluation: This chapter is concerned with the testing of various stages of development of my application including end user testing.

Chapter 6 – Conclusions and future work: This chapter is concerned with going through the reflections on the project, the work done the work to be done and the challenges and success encountered in development.

# Literature Review

## 2.1. Introduction

In this chapter, we will go through the research that was done to make sure that the app could work and to try to select the best tool for the job for the app, as well as review some other solutions to the problem and see where they failed and succeed, and finally reviewing other students work and seeing what they went with for technology how they designed their app.

## 2.2. Alternative Existing Solutions to My Problem

**Ares Alpha**

Ares Alpha is a pre-existing cross-platform application that allows users to create or join a session created by a user. Then, that user can form teams and assign players to teams and squads within those teams. Users can see where everyone on their squad is located via GPS functionality and can see marks created on the map by their squad leader, and this is to facilitate the co-ordination of Milsim Airsoft games.  
The main strength of this app is its ability to assist team management and its ability to show where squad mates are at all times, and this is due to the complex team management system which helps split up players into squads for the Milsim with squad leaders able to see the names and who is in the team allowing for more effortless co-ordination out in the field. Secondly, its ability to show where squad mates are at all times is another strength, as in Milsim-type games, players get lost quickly due to the large field.  
The main weakness of this app is it is hard to understand and use. When trying to create games on the app, nowhere does it explain how to do it, nor is a desktop required. The UI is also challenging to use, and it is hard to understand what is happening; the app has many features buried under confusing options menus. Another weakness is its unreliability; it only supports specific versions of Android, and as many as half of the players could not connect to the app on the game day due to these issues.

**Airsoft Force Tracking**

Airsoft Force Tracking is a pre-existing application that allows users to create or join a session produced by an admin; then, players can join a team and see where their team members are, put down markers on the map, and see what game is being played.  
The main strength of this app is the ability of the players to interact with the map simply by placing markers on the map to coordinate their team more effectively. Creating new sessions on the app is simple, allowing it to be used easily on a skirmish day. Another strength of the app is the ability to download after the game, the paths, and tracking of players to see where players went during the game; this would be very helpful to the marshal, allowing them to improve game modes via data gained from this.  
  
  
  
  
The main weakness of this app is that it has no safety features. There are no ways for players to contact the marshals in case someone gets injured in the field, which is very important in larger Milsim games where the field is large, and the marshals are spread out over the entire area. This could lead to people being seriously hurt with no contact with a marshal or team-mate, a worst-case scenario in Milsim games.  
I've learned from analysing these previous solutions that while some reasonable solutions and ideas from these apps tend to be missing essential parts of the whole solution or have serious reliability issues encountered often throughout. When developing my solution, I need to remember where these apps went right, how to include them in my project, where they went wrong, and how to avoid these apps' issues.

## 2.3. Technologies I’ve researched

**Native App vs Web app**

When creating this app, there are two routes I could go down: either a native app using Java or Koltin with Android Studio, or I could create a web app using react.js. Both have their advantages over each other.  
A web app is easily multiplatform because web apps run on the web and do not need a local installation. They are very easy to make compatible with a wide range of devices with only a single codebase. Secondly, Web apps have a more straightforward development process.  
A native app has a lot more functionality, with greater access to sensors on the phone, most notably for this project, GPS, and has better performance versus web apps, which is important in my app for preserving battery life over extended periods in Milsim games[1]  
For the reasons listed above, in my project, I will use a native app because my app needs access to accurate GPS to work correctly, and the improved performance will make the battery last longer in the field.

**Java vs Koltin**

Java is a popular programming language for developing native apps on an Android system; it is used on over 3 billion devices and has extensive support on nearly all Android devices; it also has extensive online documentation due to its many years in use.  
Koltin is a newer Android-focused programming language for developing apps using Android Studio. It has many new features compared to Java, such as Null Safety and a lot less boilerplate code. It is compatible with Java, being able to use the same libraries designed for Java, and can run on devices that have Java installed [2]  
Due to these factors, I will use Koltin over Java in my project. Even though I am more familiar with Java overall, the cross-compatibility and the ability to use the same libraries as Java with the additions of new features and less code needed makes me feel like Koltin is the best choice for this project

**No SQL vs SQL**

There are two directions that I can go for a database in this project: using No SQL with Firebase or using a MySQL database with GCP.  
No SQL is a database that stores data differently from standard SQL databases; instead of storing data using rows and tables in a relational model like SQL, it instead uses other ways based on the model: either document, key-value, wide-column, or graph.  
The advantages of No SQL databases are that they are pretty flexible and can deal better with high user loads; there are also different types of No SQL databases, as said before, that can be used individually or together in a multi-model database; this gives No SQL its flexibility[3].  
The advantages of an SQL database are that it is more secure than a regular NO SQL database and is more likely to be used by the government or a secure industry as it meets the criteria of specific standards. They are also better for transactional databases and enterprise resource planning systems.[4]  
Due to these factors, I will be going with a No SQL database with Firebase over an SQL one, mainly due to performance, as my database will be accessed and updated often. However, I must monitor my usage limits and ensure that my app does not use too much data.

**Nearby Connections API**

The nearby connections API is an API for Android that allows nearby devices to connect to other nearby devices regardless of internet connection by allowing devices to perform NDP and establish direct offline wireless links to exchange data [5].  
This can be used in my app by allowing devices with outdated data to be able to exchange data with a nearby device that has the updated data, allowing it to receive and send data, allowing phones that are out of connection to be able to still communicate with marshals in an emergency and still receive GPS data for player location [6].

**Maps**

In my app, the Map that tracks and shows other players' locations is vital to the project's functionality. From research, there are a few different libraries that could be used for the project; these are the Google Maps API and the Map box API; both of these APIs offer similar services; another option to consider is just directly using Open Street Maps through a library like OSM droid.  
Map Box is an alternate mapping solution to larger providers such as Google Maps. It provides many customization options and has a slick modern design to its maps; it uses the data from OpenStreetMaps and has a good SDK and API and a generous free tier.  
The first advantage of Map Box is that it offers more customization than Google. It can edit every map layer, which could help add different map styles for accessibility and personal preference in the app. Also, Map Box is cheaper to use, with it only charging once there are 25000+ users and 28,000 web loads per month; Google, however, is $7 per 1,000 loads, which is expensive. [7]  
Google Maps API Is an API offering from Google that allows developers to integrate Google Maps into their products; it is a well-built piece of software with plenty of data and easy integration with Android native apps through its API; it has extensive route-finding technology, being one of the most convenient ways to find and get to a place.[8]  
The advantage of Google Maps is that it has much more map data and features, with satellite images and street view being some of the services that Map lacks. However, this is irrelevant to the project, though having satellite images would be nice.  
Open Street Maps is an open-source mapping project that uses volunteers to map out the world; it is entirely customizable due to its open-source nature and offers itself as an alternative to other more authoritative mapping services[9]  
The advantages of Open Street Map over the rest of the options are, firstly, it is entirely free to use and will not have any overhead costs. It is also highly customizable, with many layers of styles to choose from when making the app; one of the weaknesses of directly using OpenStreetMap is that the OSM droid library, which now is no longer being updated any more, was last updated on August 19, 2024.  
From the research, Open Street Maps will be used as it is a cheaper, more customizable SDK that offers a good library (OSM Droid) of functions for Android; even though it is no longer updated, it has not gotten old enough yet for me to not want to use it.

## 2.4. Other Research

**Cross platform considerations**

For this app, there are two platforms I was considering developing on, either iOS or Android. Both have an almost equal market share in Ireland, and creating an app for both platforms would be too much work for the project's scope, so a platform had to be chosen  
from research. Android became the obvious choice. To develop in iOS, I need to use a VM running macOS or a Mac, which is not ideal for development. Additionally, I have a lot more experience with Android development, so Android was chosen.[10]

**GPS**

GPS, or global positioning system, is one of the major technologies that has enabled many of the phones to be helpful. GPS works by having satellites in the GNSS network continuously broadcasting their location. Once the phone has the location of four of these satellites, it can locate its position anywhere in the world, to a degree of accuracy, typically 5-16 feet, influenced by the phone's surroundings; generally, the more of the visible sky, the better.[11]  
GPS will be a cornerstone of the project. A good amount of the app's functionality will be based around it; one challenge I will need to overcome regarding GPS is the accuracy issue since Airsoft often happens in forests or inside buildings, and accuracy will be affected.

**Geofencing**

Geofencing is a technology that allows GPS-enabled devices to do something when they enter or leave a geo fenced area. These areas can be as large as they want and can be any polygonal shape. Due to GPS not being entirely accurate, it can give many false positives or negatives when near the edges of a geo fenced area.[12]  
Geofencing is used for many things, and it is commonly used in marketing, giving location-based adverts to better appeal to a user and get more value out of their advertisement. [13].  
In the project, It is necessary to implement areas that are "out of bounds" and not in play for either gameplay or safety reasons. Most of the time, newer players will go into these areas due to a lack of experience on the site, so from research, geofencing would be the way to do this.

**GDPR Considerations**

GDPR is a set of rules and regulations implemented by the EU in 2018 that greatly impacted how data was handled worldwide; it is hugely important for the project since the app gathers sensitive information and must abide by these rules.

The data protection principles can be broken up into 7 points in broad strokes.

1. Lawfulness, fairness, and transparency — Processing must be lawful, fair, and transparent to the data subject.
2. Purpose limitation — You must process data for the legitimate purposes specified explicitly to the data subject when you collected it.
3. Data minimization — You should collect and process only as much data as necessary for specified purposes.
4. Accuracy — You must keep personal data accurate and up to date.
5. Storage limitation — You may only store personally identifying data for as long as necessary for the specified purpose.
6. Integrity and confidentiality — Processing must be done in such a way as to ensure appropriate security, integrity, and confidentiality (e.g., by using encryption).
7. Accountability — The data controller is responsible for demonstrating GDPR compliance with these principles.

[14]

The main concerns with the app are data minimization and integrity. The reason is that the app will collect sensitive information like location, and storing identifiable personal data in case of a data breach is a bad idea. Secondly, having good data security is just a good best practice, so sticking to this as a principle will be beneficial. The rest of the principles will also be kept in mind as well.

**Battery Life**

Quartermaster will be using GPS as a core feature of its design. However, GPS and location-based services do have some drawbacks. One of the major ones, especially for this project, is the increased power drain that will limit battery life on the phone that the application is running on; this is a problem as most Airsoft games can last anywhere from a few hours to multiple days, so battery life must be considered.  
In indoor areas, battery consumption from GPS can be increased by as much as 75% from the phone using more battery to find a better signal strength[15]. One way to keep battery life costs down is to use less accurate location-finding methods, such as coarse network location, draining only a tenth of what using fine location may take using GPS[16].  
Hopefully, this will help offset issues of using this app in areas with bad signals; it might be more inaccurate, but having their phone with a workable battery is more important.

**Requirements Gathering**

Requirements gathering is an important stage in the development of any application. Requirements gathering consists of creating requirements by getting user feedback and wants for the potential application.  
  
There are two different types of users this app intends to assist:  
Marshals are the people who create and run the games; their job is to enforce rules and look after the safety of the players.  
Players: the people who participate in and play the game.

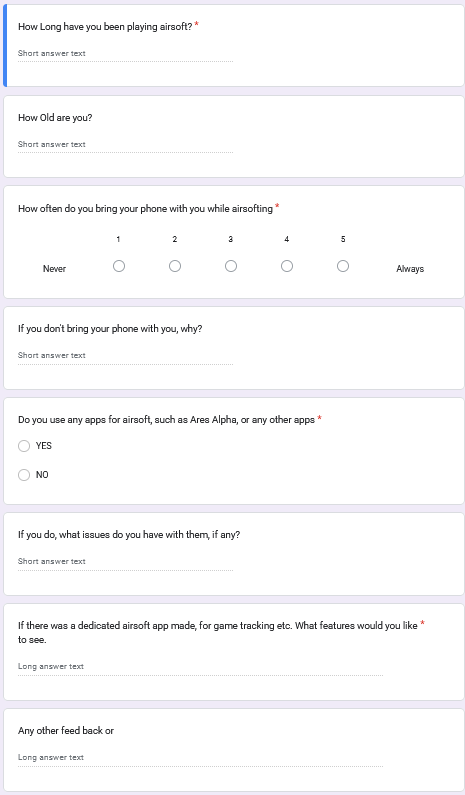
**Interviews**

To get some requirements for the marshals, I drew on my experience from working as one and interviewed fellow marshals to get a good idea of what was needed.  
I asked them questions along the lines of:  
“What would this app need for you to use it?”  
“What problem do you encounter on the field?”  
“What features would you like to see?”  
“What do you think should be avoided?”  
These questions were chosen to get a simple list of requirements and actionable features.  
Moreover, these responses were used to construct some requirements:  
Must-Have: SOS feature in case someone is injured   
Should Haves: Safe zone markers, out-of-bounds areas  
Could Haves: Emergency contact information(phone numbers)  
Avoid Complexity and barriers to entry for players

**Surveys**

To get some requirements for the players, i drew on my own experience of being a player, as well as creating a survey and using the responses to help with creating my requirements.

these questions where chosen as to ascertain weather or not that the app would be viable, via the phone question. And to see if there are any features that players want in this application.

Figure 1: survey

Responses:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| q1 | q2 | q3 | q4 | q5 | q6 | q7 |
| 6 | 15 | 1 | In Case of losing it | N |  | Map of Site |
| 6 | 51 | 5 |  | N | Most apps are bad | Team Mate tracking |
| 13 | 25 | 5 |  | N |  | Sign in numbers |
| 25 | 56 | 1 | Losing it / it gets hit | N |  | Real time updates |
| 14 | 60 | 1 | Dont want contact | N |  | comradely |
| 10 | 58 | 5 |  | N |  | Topics on games terrain etc, a league system |

## 2.5. Existing Final Year Projects

**Anseo!**

**Author: Jonathan Hew**

Anseo is a web app that is designed to be used by lecturers to keep attendance to their lectures in a more convenient way than a paper sign-in sheet; it does this by creating a session and allowing students to join it only if they are physically present within a geo fenced area.  
Jonathan used a three-tier model for the project with a presentation layer, application layer, and data layer; the user opens the web app, which uses react for its function, and then connects using node.js to a PostgreSQL server to store data.  
They also used the Agile development method of feature-driven development for their project; their reason was because of the numerous features of the project; this is similar to the project and the many features I will also have to make, so they might be a good example of what to follow.  
Anseo! It uses technology similar to the app, such as geo-fencing and GPS technology. At the same time, one user hosts a central session that all other users connect to, so some lessons should be taken from this project due to its similarities.

**Nitelite**

**Author: Sean Breen**

Nitelite is a native mobile app designed to help people stay safe in nightlife environments; it intends to do this by creating an app that allows people to keep track of their friends in a busy nightlife environment and, if needed, request urgent help from friends in a dangerous situation.  
They also had a 3-tier system, with a React native front end, a Django back end, and a SQL database.  
They also used an agile development methodology, though they did state that it would be hard to use in a single-person project due to the nature of agile. Sticking to the six key principles would be ideal to ensure future best practices.  
Nitelite interested me because it used Bluetooth beacons to get accurate positions inside a building, which could be helpful for my app.

**Conclusions**

From these projects I have examined above, I have learned some important lessons; firstly, the agile development method is quite popular, and for good reason: it leads to better outcomes for the project overall. Secondly, both projects gave me a good idea of what a finished project looks like and my requirements to achieve a similar result to the above projects.

## 2.6. Conclusions

The conclusion of this literature review conducted for my final year project is, firstly, the research into different tech away from the stuff I would usually use led to some fascinating and beneficial insight into some of the tech I am going to use for this app and made me change my mind on confident choices I had already made up in my head about the development of this app.  
Secondly, there are some existing solutions from my examination of previous solutions. However, both of those examined have flaws I also need to overcome and strengths I should try to put into my project, and it proved that what I wanted to do was feasible.  
Thirdly, reading how previous students approached similar problems showed me some examples of what was expected in terms of complexity from this project and ideas on how to move forward with it.

**Requirements**

From what I have learned, I have built a set of requirements using the ideas gathered from above and using some requirements gathering from the expected user base using interviews and a survey.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Req ID | Requirement Desc | Category | Priority | Source | Criteria | Notes |
| UR-001 | Realtime Tracking of players and teammates | Functional | High | Survey | Tracking works well | Completed in Prototype |
| UR-002 | Stats Tracking of players | Functional | Medium | Survey | Be able to track your win loss and other stats |  |
| UR-003 | Accurate Map of site | Functional | High | Survey | Have the ability to create an accurate map of the site |  |
| UR-004 | Reliability of functionality | Non-functional | High | Interview | Have all the functions work reliably | This is a common issue with other apps |
| UR-004.1 | Tracking Accuracy | Non-functional | High | Interview | Have tracking be accurate to 5-10 feet |  |
| UR-004.2 | Session Reliability | Non-functional | High | Interview | Have session be easy to join and not close randomly |  |
| UR-005 | Safety SOS features | Functional | High | Interview | Have a way for players to contact marshals in case of an emergency |  |
| UR-006 | Be able to tell how many people are at the site/ are booked in remotely | Functional | Low | Survey | Have the ability to see different sites and how many people are planning to be there | Would require a lot of work and a whole site system, do if time permits but otherwise ignore |

# 3. System Design

## 3.1. Introduction

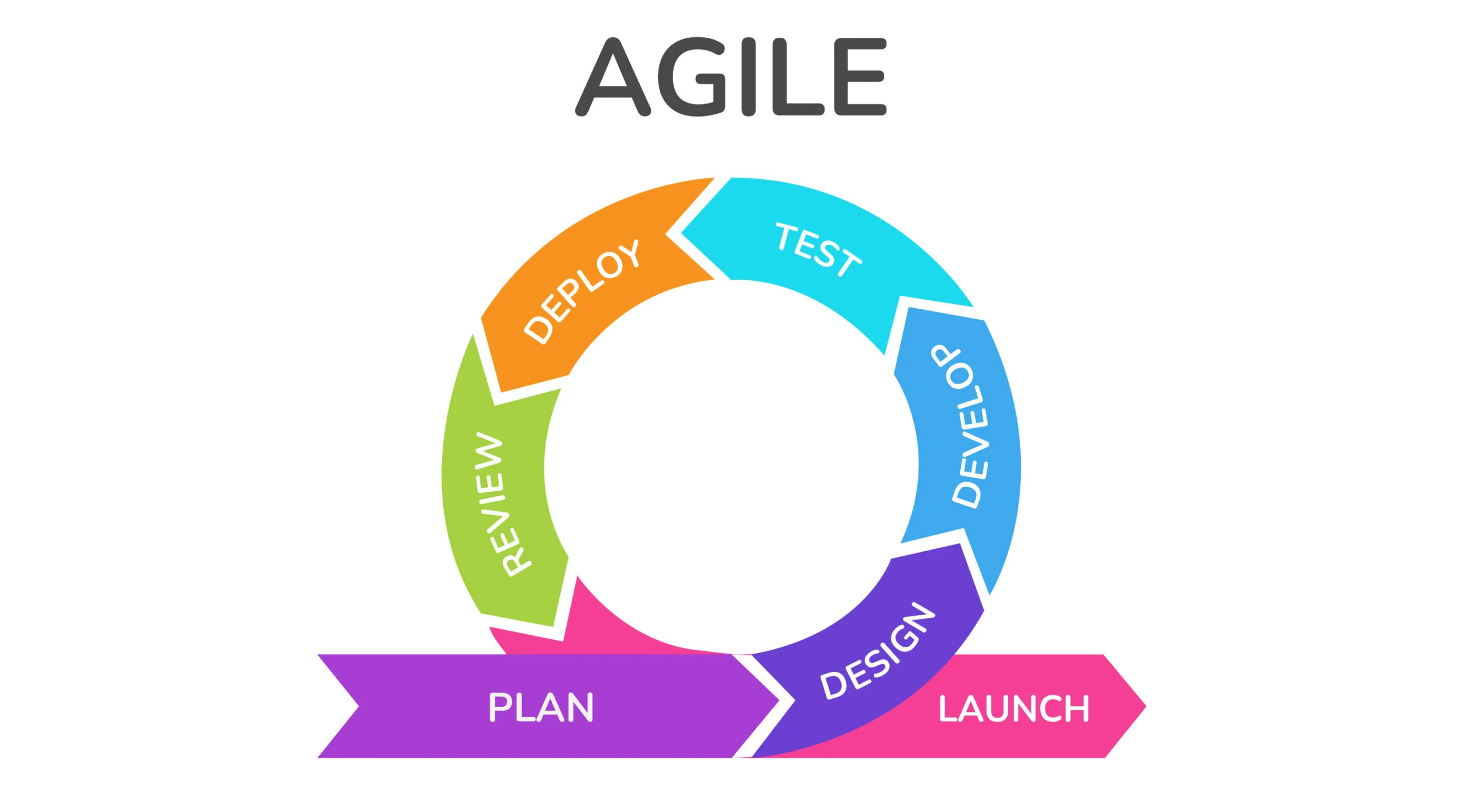
System design is a vital part of the development process, as it sets guidelines and the design for the project in place to ensure that the project is done to a good standard and in a reasonable time frame. In this chapter, we will review some software methodologies and the one chosen for the project. We will review the system's overview, examining use cases, activity diagrams, and the database design.

## 3.2. Software Methodology

In my research i found 2 software methodologies that i thought where suitable for this project

**Agile development methodology**

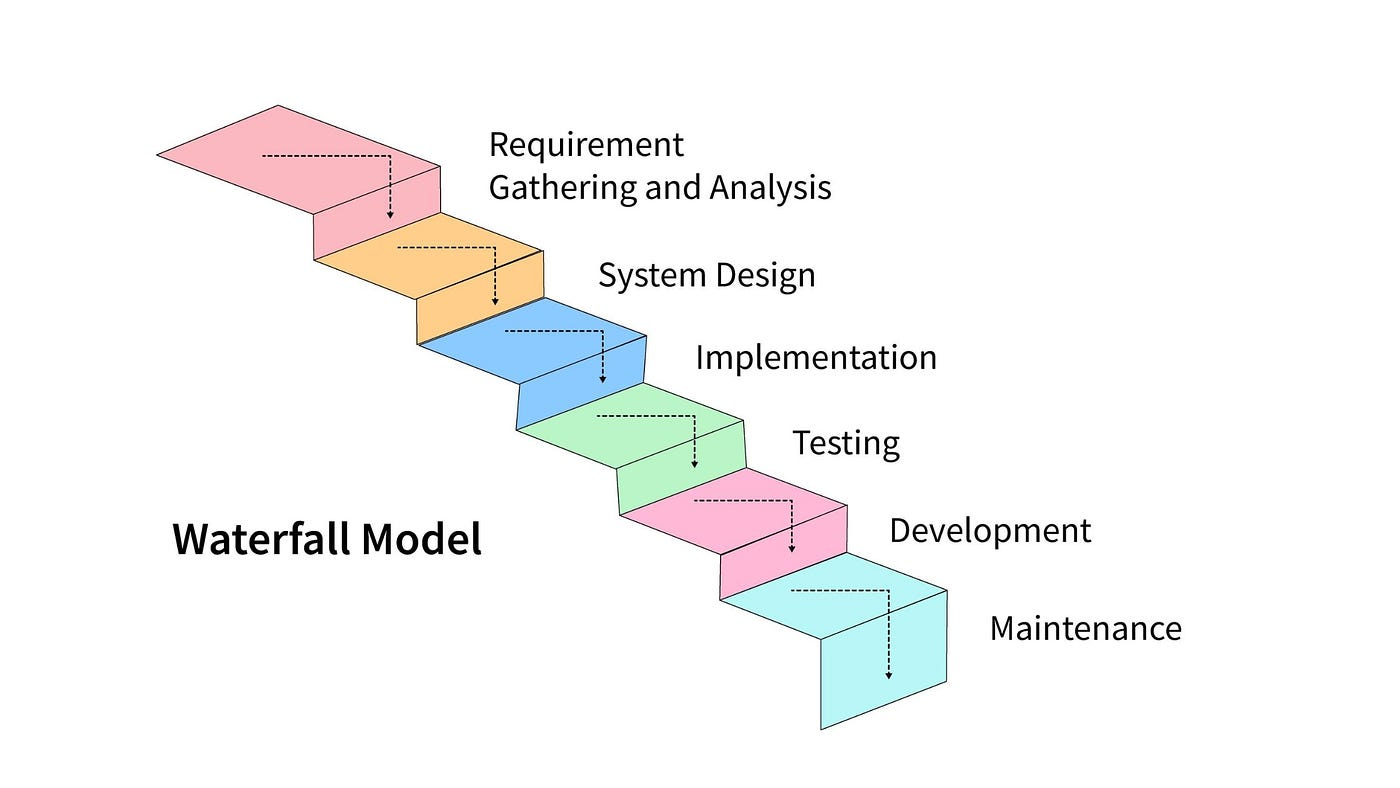
The agile methodology is a methodology framework that is based on breaking down the project into multiple stages and iterating on the development cycle, and it is designed to be reactive and be able to change based on changing requirements and take in shareholder or project owner feedback and incorporate it into the project with minimal issues, there are a few ways to implement an agile approach, examples include scrum and kanban[17][18]

Figure 2: example of a agile development cycle

This approach appeals to me due to my circumstances around this project, i am a solo developer for this so a lot of the collaborative benefits of agile will be lost however since i work at an Airsoft site it puts me in a good position to receive feedback on each iteration of the project.

**Waterfall development methodology**

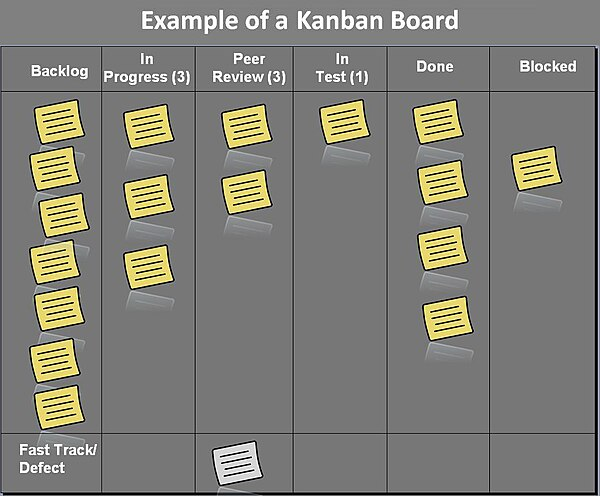
The waterfall development methodology is a linear sequential life cycle model, where planning, designing, development, and launching are handled in sequential order; due to this highly structured approach, the waterfall methodology is highly inflexible and does not react well to changing requirements.

Figure 3: an example of a waterfall development cycle

However, since this project is a solo-created project, its highly structured nature, and easy-to-use nature might be beneficial to the project, and its focus on getting the project done rather than re-iterating over and over could help get the project done quicker[19]

**Kanban Methodology**

The Kanban method is a framework for implementing agile software development by visualizing all work to be done and limiting the work in progress. It does this by breaking down the work to be done into individual items and then organizing them on a board and sorting them by the categories “to-do,” “Progress,” “in Review,” “Finished,” and “Blocked” This is all done to minimize work wasted and ensure that projects are finished on time, instead than using fixed sprints a more continuous flow to development with each item being worked on to completion then moved onto the next stage.

Figure 4: a example of a kanban board

**The Chosen Method: Agile Kanban**

In the end i decided that using an agile Kanban method would be good for this project, i feel that with the ability to be able to regularly test my app on a weekly basis that an agile method would produce good results, as i will be able get feedback on each sprint and actually use that feedback to improve the project.

The main reason i am going to be using the kanban method is because it is very good at visualisation, with each task being represented on a physical board as a card, it will help me keep track of what work i need to do and what work i have already done. Additionally a way to visualise the progress i will be making will help me stay motivated and productive for this project.

## 3.3. Overview of System

**Overview**

Quartermaster will follow a simple 2 tier architecture with the front end native application communicating directly with the back end server, the reason this architecture has been chosen is due to its simple and quick to make nature additional the additional complexity of an N tier architecture is not required for the scope of this project

**Front End**

The Front end of the Quartermaster application will be an android native app that uses Kotlin, a new object oriented programming language, designed to be interoperable with Java and Java Libraries., The applications UI will be created using Jetpack Compose, which is the recommended toolkit for making UI in native android apps as it is significantly easier to use and allows for far more interactivity within the UI.

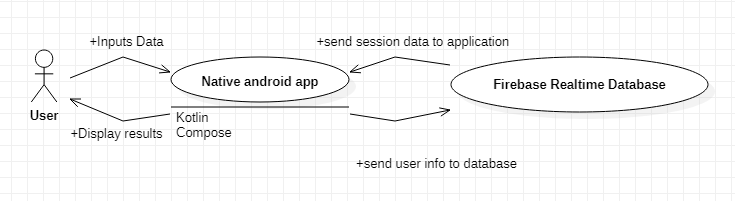
OSMdroid will be used to create any maps screens with the OpenStreetMaps map set, and will include an editor allowing people to create their own maps for the application.

**Back end**

The Back end of the application will use firebase to host the data for the application on a real-time database, firebase was chosen as it has an affordable pricing model, and it is designed to be used with mobile applications similar to this project.

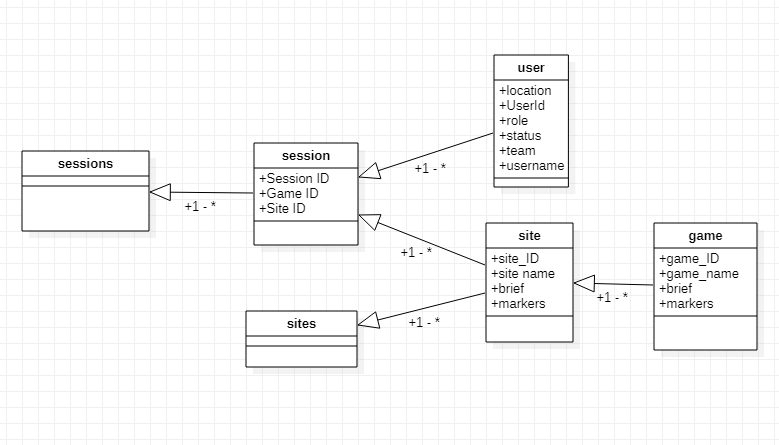
**Real-time database**

Real-time database is going to be the data storage solution used, it is a NoSql Database and it stores its data in JSON format. its being used as it is able to be updated and changed very quickly, which is essential for this projects functionality, in addition its integration with other google cloud tools like cloud functions allows for additional functionality. In addition since we are using firebase, all the hosting and setup is handled by Firebase itself, saving time.

Figure 5: basic diagram of planned architecture

**Database Design**

for my database i need to have a few different linked objects such a users and sessions and sites and games etc, in order to do this in a planned fashion a database diagram was created as a visual representation of what the final database will look like.

Purpose of each object:

**Sessions**

Sessions is simply a holder for the session object to make it easier to access without causing issues later.

**Session**

Session contains the data that is related to each individual session, these objects are created when a user wants to create a session and is always created with atleast one user.

**User**

User is each individual user in a session, these are contained within the session and contain all the info needed to display the user correctly.

**Site**

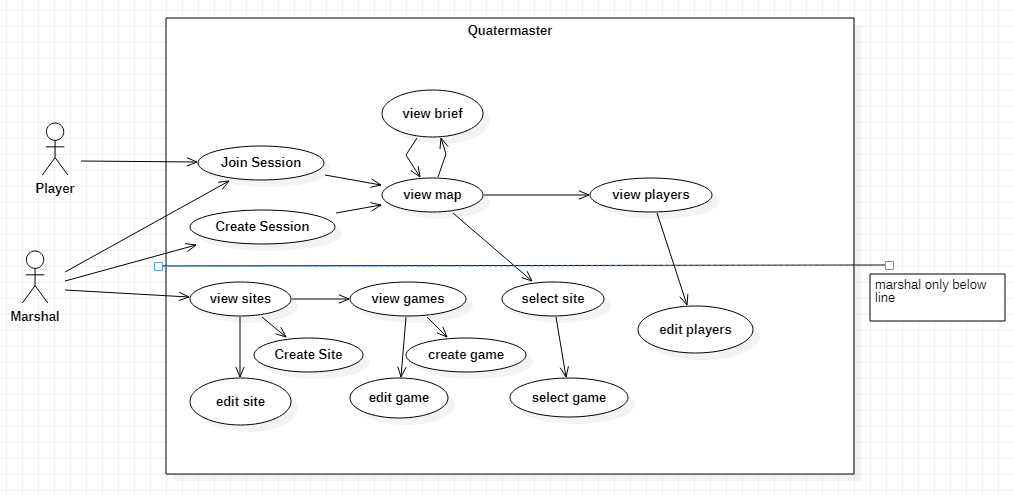
Site is an object used to contain the info about each site, it contains info like the safety brief and the site name ,game objects are always in one of these objects, markers contains info about map objects that will be drawn by the app.

**Game**

Game contains info about each individual game, such as brief and game markers.

**Use Cases**

in my app there a 2 different types of User Players and Marshals, players only need of the app is to be able to join sessions , marshals act as the admins they create the sessions the games etc. and they also need to be able to join other marshals sessions as there can be more than one marshal out there at once, below diagram depicts a high level diagram of the uses case of the app, and following that a more in depth explanation of each

Figure 6: Use case

**Use Case Diagrams**

In this part I will elaborate on the use cases set out previously

|  |  |  |
| --- | --- | --- |
| **UC-01** | **Create Session** | |
| **Dependencies** |  | |
| **Description** | The app will behave in this way when the create session option is selected | |
| **Preconditions** | The user has opened the app and isn’t already In a session | |
| **Ordinary Sequence** | **Step** | **Action** |
| **1** | User selects the Create session button |
| **2** | System connects to the firebase server |
| **3** | System creates a new session with a unique ID |
| **4** | System creates a new user in that session and moves the user to the Map screen |
| **Postcondition** | The user is in a session and can have other users join it | |
| **Exceptions** | **Step** | **Action** |
| **2** | if the system cannot connect throw an exception |
| **Comments** |  | |

|  |  |  |
| --- | --- | --- |
| **UC-02** | **Join Session** | |
| **Dependencies** |  | |
| **Description** | The app will behave this way when a user attempts to join a session | |
| **Preconditions** | The User has opened the app and isn’t already in a session, a session exists | |
| **Ordinary Sequence** | **Step** | **Action** |
| **1** | User inputs a session code |
| **2** | User presses the join session button |
| **3** | System connects to the firebase |
| **4** | System checks to see if there is a session that exists with that code |
| **5** | If the session exists it adds them to it and puts them in the map screen |
| **Postcondition** | The user is in a session | |
| **Exceptions** | **Step** | **Action** |
| **3** | Cannot connect, throw an error |
| **5** | Session doesn’t exist throw an error |
| **Comments** |  | |

|  |  |  |
| --- | --- | --- |
| **UC-03** | **Create Game mode** | |
| **Dependencies** | **UC-05** | |
| **Description** | The app will behave this way when the user creates a gamemode. | |
| **Preconditions** | The user has opened the app and has navigated to the site they want to create the game mode in | |
| **Ordinary Sequence** | **Step** | **Action** |
| **1** | User Presses Create Gamemode Button |
| **2** | System opens dialogue that requests input for name |
| **3** | User inputs a valid name |
| **4** | System connects to firebase and creates a new game in the current site |
| **Postcondition** | Gamemode is created and user can now edit it | |
| **Exceptions** | **Step** | **Action** |
| **3** | User inputs invalid name, refuse input and try again |
| **4** | System cannot connect, throw error |
| **Comments** |  | |

|  |  |  |
| --- | --- | --- |
| **UC-04** | **Call SOS** | |
| **Dependencies** | **UC-01/UC-02** | |
| **Description** | The app will behaves this way when the user selects to send a SOS signal. | |
| **Preconditions** | User is in a session | |
| **Ordinary Sequence** | **Step** | **Action** |
| **1** | User needs help, either emergency or not and presses the SOS button |
| **2** | System connects to firebase and updates their status to SOS |
| **3** | FCM sends out notification to all admins |
| **4** | Users location is marked on all admins maps |
| **Postcondition** | User is marked on map and admin have notification informing them | |
| **Exceptions** | **Step** | **Action** |
| **2** | System cannot connect to firebase throw error |
| **Comments** |  | |

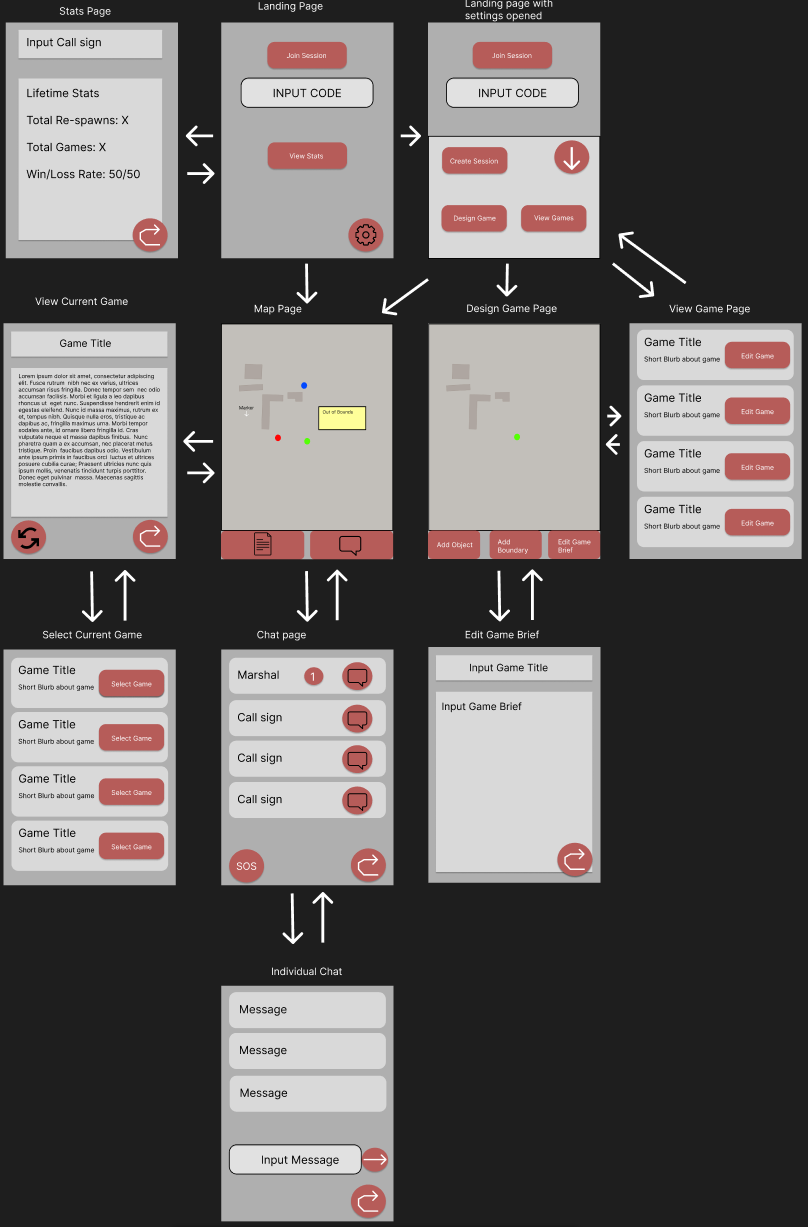
|  |  |  |
| --- | --- | --- |
| **UC-05** | **Create Site** | |
| **Dependencies** |  | |
| **Description** | How App will behave when create site is invoked | |
| **Preconditions** | User has navigated to site list page | |
| **Ordinary Sequence** | **Step** | **Action** |
| **1** | User presses the create site button |
| **2** | System opens dialogue that requests input for site name |
| **3** | User inputs a valid site name |
| **4** | System connects to firebase and creates a new site |
| **Postcondition** | Site is created and is able to be accessed | |
| **Exceptions** | **Step** | **Action** |
| **3** | User inputs invalid site name, refuse input and try again |
| **4** | System cannot connect, throw error |
| **Comments** |  | |

**UI Design**

One of the most important aspects of the app is UI, since Quartermaster is going to be used in high stress situations, its important that the UI is well designed and is easy to use

**Prototype UI design**

During the prototyping stage of development, a few designs where tried out for the UI, below in figure 5 you can see the first mock-up of a possible final UI design, however this was not implemented for the prototype in time, and it was scrapped after some features where dropped and others added.

Figure 7: UI Mockup of Pages in App

**Final UI design**

in order for quartermaster to reach its full potential a good simple to UI was necessary, and the previous mock-up prototype UI was to cluttered and complex to fulfil that vision so when going into the developing the full application, more UI design and testing was done in-order to land on an optimal solution, in this mock-up, the UI was simplified and labels where added to each of the navigation options, a lighter grey colour was chosen as it is easier on the eyes in most light conditions.

In this design more simplicity was emphasised in order to make the design easier to navigate, also a landscape view will also have to be made due to the fact that alot of users use chest mounted phone holders while playing airsoft so a landscape is a must.

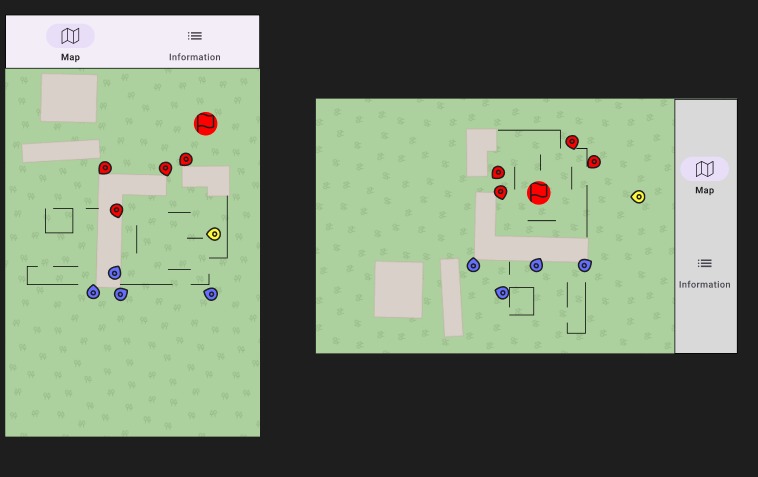
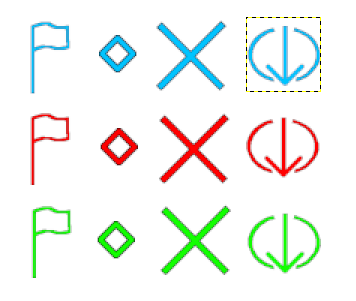
Figure 8: UI mockup for Map Page

Figure 9: example phone holder

**Map Icons**

it was decided that map icons and user icons must be as easy to spot and identity as possible, when designing the icons the NATO joint military Symbology was used as a base, due to the fact its designed to be easy to identify on a map, and is on theme for the purpose of the app.

Figure 10: icon set created

## 3.5. Other Sections

**Balancing battery life with location updates**

Due to Quartermasters intended design is that the app is to be used throughout the course of a game-day (4-24 hours) battery life is very important, however a lot of the technologies i use in this app tend to consume a lot of battery, especially since i want the location tracking and geofencing to be running while the app is in the background or the app is closed.

One way i will try to be managing this is to only have location be updated every 15 seconds to minimize the amount of battery that is used by the application, this will save battery at the cost of live information.[20]

Another way i can reduce battery consumption is via lowering the accuracy of the GPS as GPS will take up more power to get a better signal in bad connection areas, however i am loath to do this as i feel that it would reduce the usefulness of my app.

## 3.6. Conclusions

From the design work covered in this chapter the possible methodologies for this project where covered and then a chosen methodology, in this case Agile Kanban was chosen due to its visualisation of progress and the flexibility of the system, secondly the design work for Quartermaster was covered, including architecture design, database design and UI design, as well as the creation of use cases which will be fulfilled during development.

# 4. Project Development

## 4.1. Introduction

For my prototype i wanted to try get the main functionality of the app down, in my mind the main thing that everything needed to be done first to even start working on anything else was being able to join and create sessions and then actually being able to see the location of other users within that session.

So with that as my goal i began prototype development

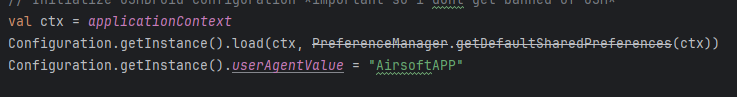
## 4.2. Prototype Development

**Preliminary learning**

to start off with since i haven't done android development in a while and i was using a new language and UI design method with Koltin and jetpack compose, i had to read a little documentation and what the syntax was of both languages and how they are intended to be used, once that was done i then moved onto trying to get the map view with OSMDroid working

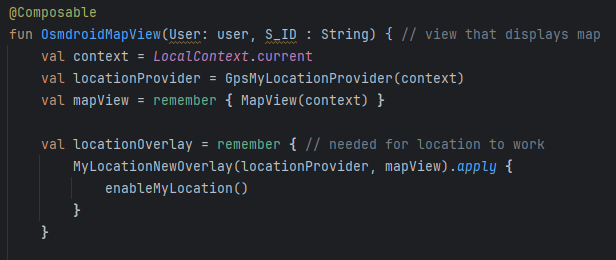
**Setting Up The Map View**

So when i started on the map i had some issues with setting up OSM droid, mainly because the documentation for it was not great and it was all using XML instead of what i was doing, so it took a bit of effort to setup, but in the end i got it running.

Figure 11: setting up the config for OSMDroid

This is setting up the config so when i get information off the OSM servers i dont get banned as and default apps with this not setup can get your app banned from the service.

In order for the current location to be marked on the map an location overlay had to be added to the map view, overlays are how OSM droid can display additional information over the map.

Figure 12: OSMDroid Map view

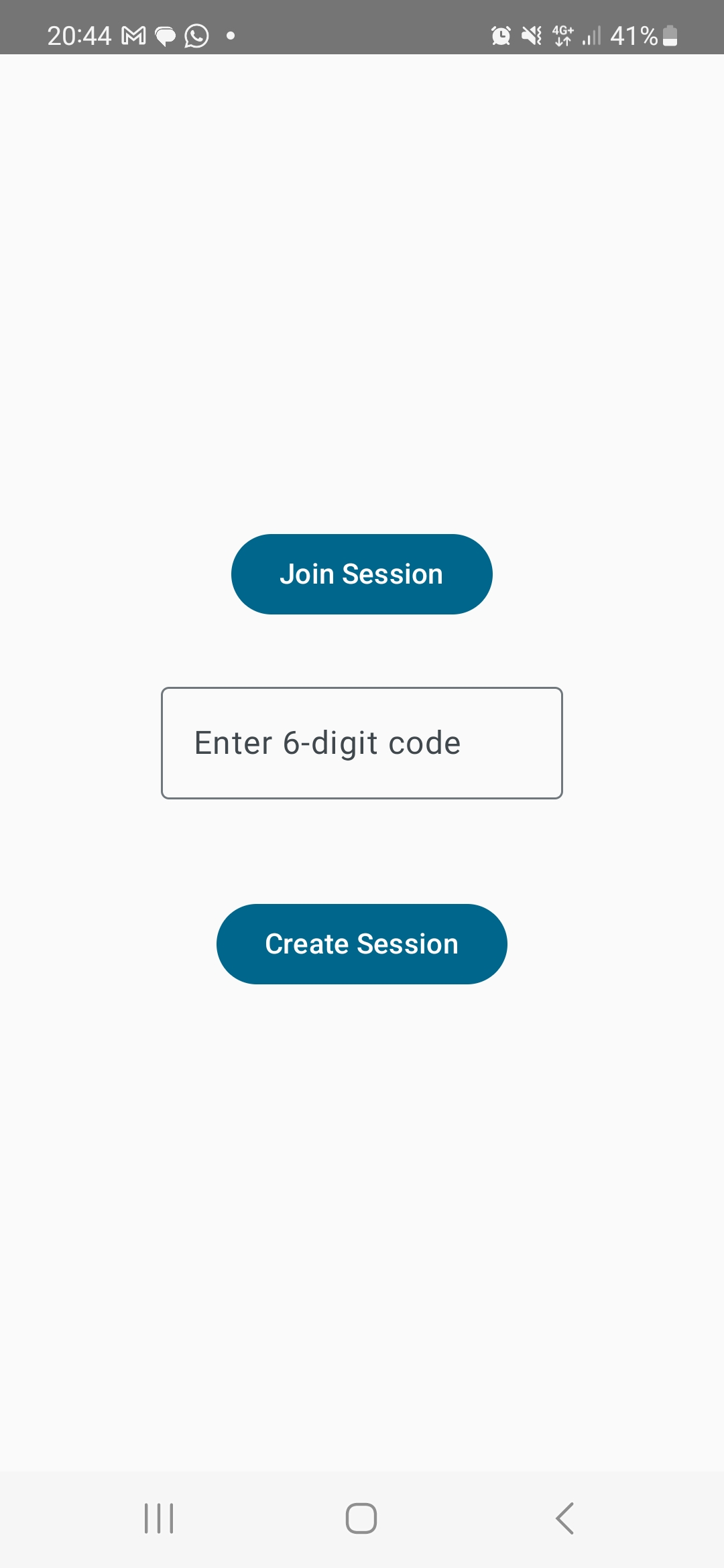
**Setting up Joining / Creating Sessions**

Before i could start coding the creating and joining a session, i had to create a firebase realtime database to host the data, thankfully the process of doing this is very easy due to the easy to use starting guides and the good documentation, once that was setup, all i had to do was setup my app to use firebase by adding the dependencies and setting up the con fig file, the i had to decide on a structure for my data, i ended up using this as my document structure for the database

Figure 13: Database document example

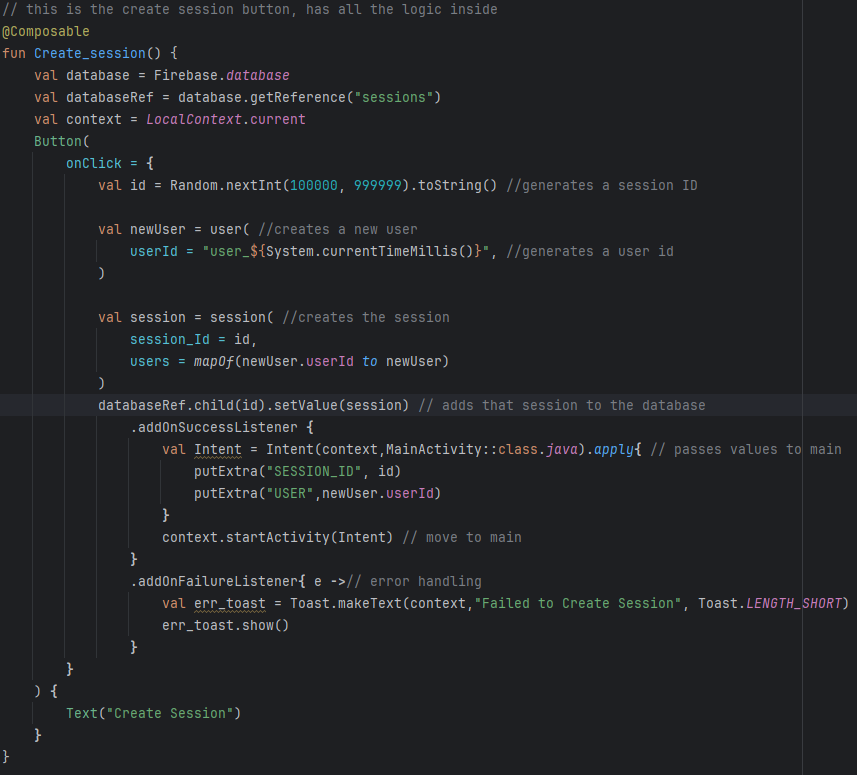
for the prototype i kept the structure simple only adding what i really needed in, in the future i will need to add more fields for stuff like the SOS and messaging features but until then this fulfils my needs.

Then i moved onto the harder part, implementing first creating a session and then joining a session, firstly i created a new activity and set it as the new startup page, this became landing\_page.kt, then i added the create session and join session button with a text input field for a basic UI.

Figure 14: prototype UI for landing page

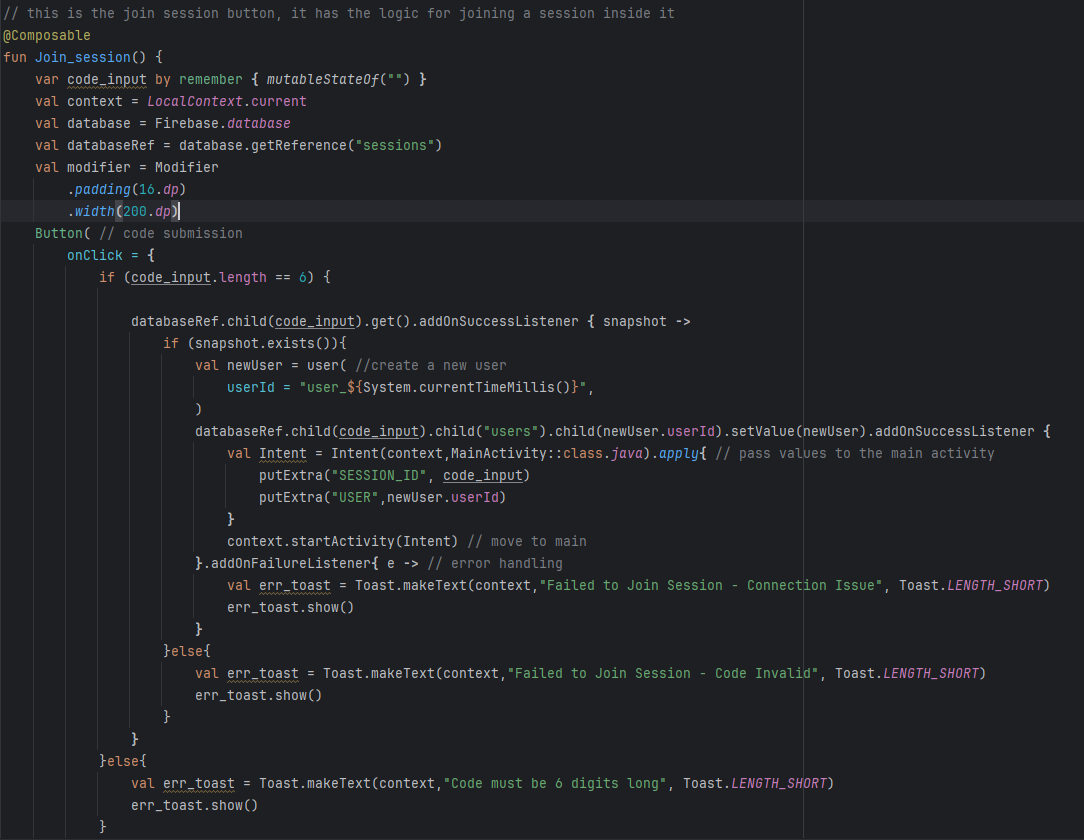
Next i had setup how sessions where created, i created the User and session classes to contain the information about users and the current session, they have the same parameters as the database document, this is in an attempt to ensure that the database functions correctly, as i had issues in development with documents being created outside of the “sessions” index due to bugs.

so the way creating a session works is, firstly it creates a user, using the phones current time in milliseconds, hardly an id that would get repeated and cause issues, then it creates the session object with a random id, it then connects to the database and on a successful connection and update, it will pass the session and user id to the main activity for the map.

Figure 15: code for creating a session

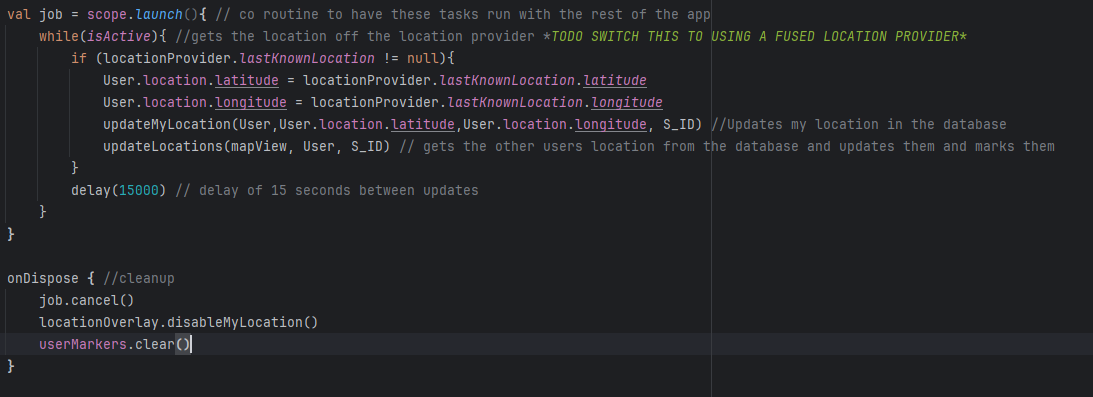
For joining session its a bit more complicated as we have to check the database for the session first before and then add the user to that session.

I implemented joining a session by firstly when the input button is pressed it checks to see if there is a 6 digit code in the code\_input field, if there is one it will check the database by getting any session with the same code as the one inputted, if there is one it then creates a user similarly to create session and then adds that user to the session in the database before passing the user and the session id to the main activity.

Figure 16: code for joining a session

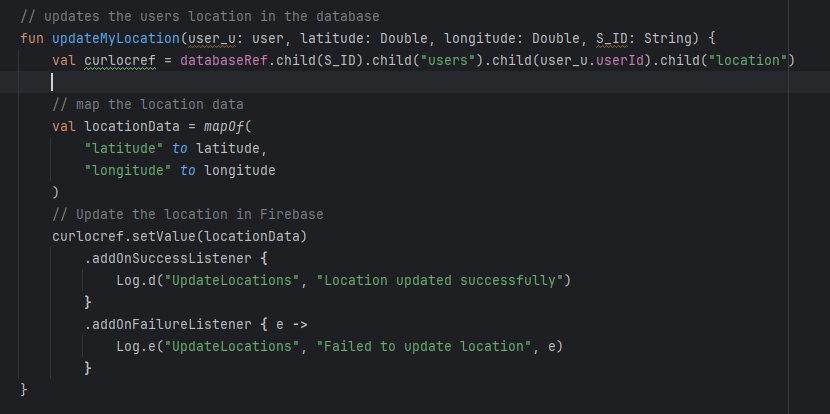
**Tracking the Other Users in the Session and uploading my location to the database**

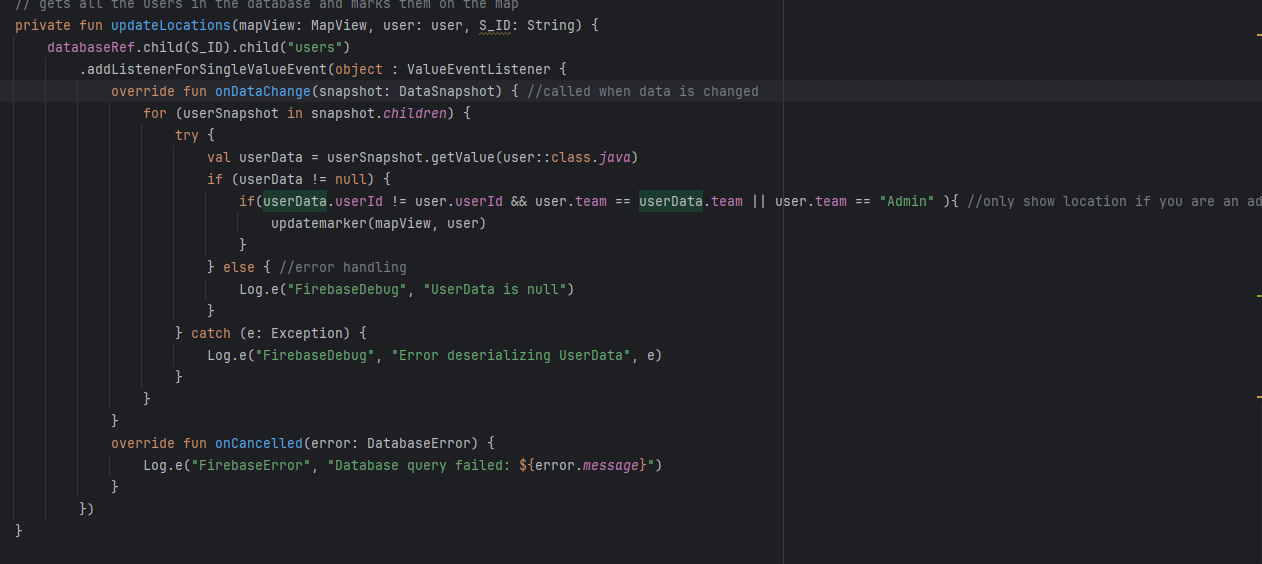
next step in getting my prototype done now that we are connected to the database and able to join a session is to now get all the other users locations from the database and then display them on the map using markers. To-do this without disrupting the rest of the code the use of a co-routine was needed, inside that co-routine it will both update the local list of other users and my current location on the database.

Figure 17: co-routine code

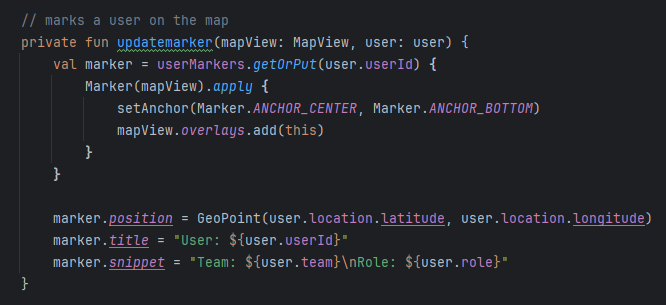
Next i got the update my location function working, i started by just extracting the current location as longitude and latitude and mapping it to a value, i did this because in the database the location is nested inside of each user in the location value, which contains both longitude and latitude, then its as simple as uploading that value to the database.

Next step was to implement the updateLocations function, this functions goal was to retrieve a list of users from the database and then mark on the map where each user was, it does this by getting every user in the session, then converting the raw data back into the user class and marking down every user that is either on your team, or if you are an admin marking everyone.

Figure 18: updateMyLocation code

Figure 19: updatelocations code

Finally last step now, all that was left was to implement marking down on the map where each player was this was done by using the OSMdroid marker function which allow me to put markers where ever i want on the map as long as i have the long and lat of where.

Figure 20: code of update markers

## 4.3. Other Sections

**Resolved Issues**

one major issue i had in developing the prototype was my inexperience with using koltin and jetpack compose, a lot of the bugs i encountered throughout the development of the prototype was mainly down to me not really knowing how to format jetpack compose and koltin, and it was difficult to implement the UI even a basic one when you don't really have a good preview, i know jetpack has the @preview feature that you can use to build previews but android studio just wasn't having it i guess and wouldn't let me load a preview, the way i resolved this issue was just through brute force, i am going to have to get good at using these tools if i want i successful project, so just actually applying them in the prototype has given me a better understanding of them, and to solve the preview issue anytime i made changes to the UI i would run it on my phone to make sure it looked the way i wanted it

Another major issue i was having at the tail end of development was issues with accessing the database properly, for almost an entire day i couldn't figure out why when it was updating the user location in the updatemylocation function it would instead of adding the location to the user it would just put it at the top of the session, i was baffled by this, until i realised that i had made an error in how i did the curlocref to get the right place to put the data and after i changed that it resolved itself

Figure 21: the line in question

**Unresolved Issues**

there are unfortunately defiantly some issues with the prototype, firstly the UI for the map isnt great both the users current location and the markers for other players are the default ones at the moment which doesn't look good and the UI for the landing page is as bare bones as possible, secondly there is no way to change your team or role at the moment and that feature that was planned for the prototype was cut due to time constraints

## 4.4. Conclusions

In conclusion i feel that while my prototype is a good starting point for my project as it develops the core feature of the app, and while it was unfortunate that i didn't have enough time to get all the features i wanted in the end, it has helped me achieve a greater understanding of the technologies i am going to use and will greatly help with the next stage of development.

# 5. Issues and Future Work

## 5.1. Introduction

In the writing of this interim report and the development of the prototype numerous issues where encountered, additionally due to the nature of the report much work on developing Quartermaster into a fully fledged FYP needs to be done, in this chapter i will go though the issues and Risks encountered, and talk about how i will try to solve them in the future, then i will talk about my plans and future work that needs to be done to finish the app.

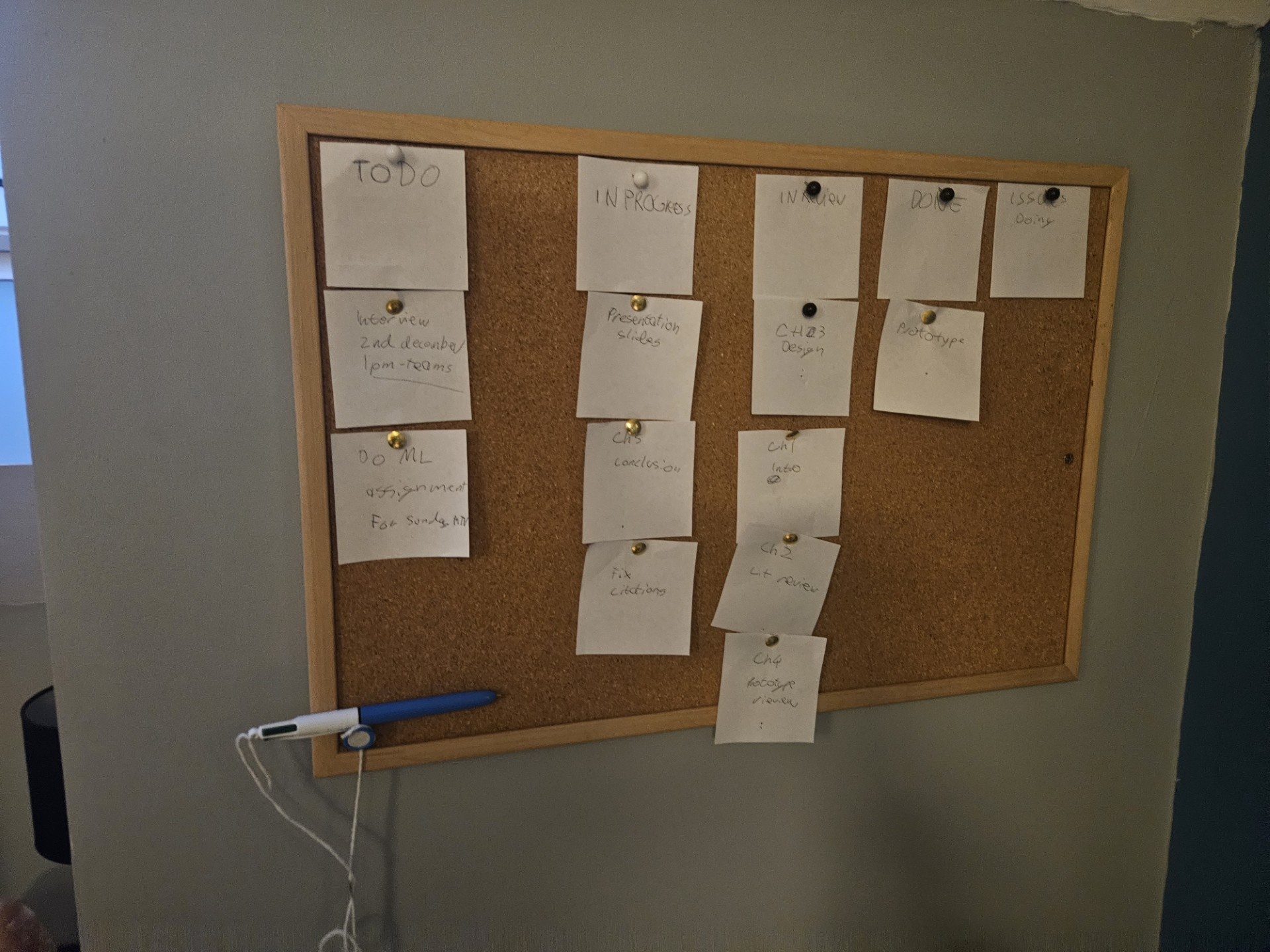
## 5.2. Issues and Risks

**Issue #1: Issues with OSMdroid documentation**

one issue i had was most of the OSMdroid documentation was designed to be used with java and XML not Koltin and Jetpack compose, this caused numerous issues early on when i was trying to get to grips with using OSMdroid in my prototype development, i resolved this issue by learning how jetpack worked properly and then using that to create the map view.

**Issue #2: Issues with time management**

another issue i had when writing this report was time management i was spending to much time on certain aspects and not enough on others, in some cases i would forget one part of the project and then do no work on it for a week, to fix this issue i started using a kanban board to help visualise the work i needed to do and the progress i was making on the FYP, this approach really helped it motivated me to do the work and since i have started using it my productiveness has gone up considerably as i am able to visualise my progress on the FYP and keep focus.

Figure 22: my kanban board for the Interim Report

**Issue #3: Issue with report writing**

another issue that i encountered was my reporting writing, i am not great at writing reports and it shows sometimes, this was brought up to me by my FYP supervisor during the writing of the project proposal, to try and resolve this issue i have contacted learning support and i am going to get some help with getting my report writing to a higher standard.

**Risk #1: Risk of feature creep**

with this app and its features there is a large risk of feature creep resulting in me getting new ideas to add stuff as i work on the app, to try address this, i will use the MoSCoW prioritization method, getting the main features done first and then any new ideas will get added on to-do later, not before the core is done.

**Risk #2: Risk of Burnout**

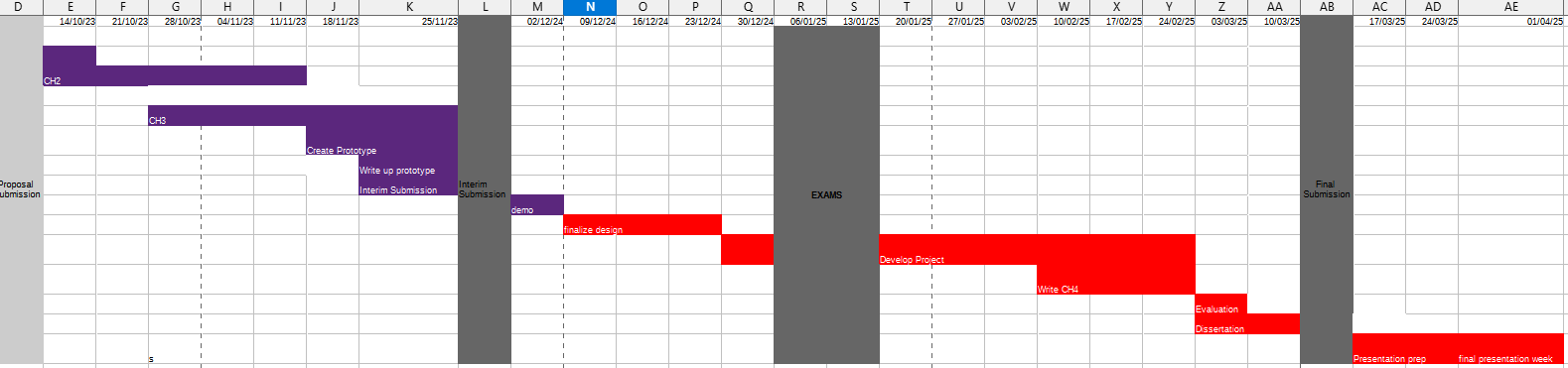
with the amount of work that has to be done there is a large risk of me burning out with this project, which is not to say that i wont get it done, but that i will lose motivation and not put my full energy into this project, i have felt this already while just writing this report due to the poor time management issue from earlier, to try address this i will try to spread the work out more over time, so i have enough time to take breaks in-between sprints on this project to hopefully avoid burnout.

## 5.3. Plans and Future Work

My plans after the interim report is submitted and the demo is done is to immediately get some more design work done, while I’m pretty happy with the structure of the front end of the project more work needs to be done on the back-end to get it up to a good standard of work, additionally i might change over from using a firebase database to one on postgres or mongoDB, the reason behind this is that firebase does have a limit on its free spark tier, and i go over that i will have to pay, however i could set up my own server using one of the aforementioned software and a spare computer i have lying around, this will save me money and also show that i am a capable 4th year student.

### 5.3.1. GANTT Chart

To manage my time till the final submission i have created a gantt chart to show how i will manage time over the coming weeks.



# Bibliography

[1] ‘Web Apps vs. Native Apps vs. Hybrid Apps - Difference Between Types of Web and Mobile Applications - AWS’, Amazon Web Services, Inc. Accessed: Nov. 13, 2024. [Online]. Available: <https://aws.amazon.com/compare/the-difference-between-web-apps-native-apps-and-hybrid-apps/>

[2]‘Kotlin vs Java: the 12 differences you should know’. Accessed: Nov. 13, 2024. [Online]. Available: <https://www.imaginarycloud.com/blog/kotlin-vs-java>

[3] ‘What Is NoSQL? NoSQL Databases Explained’, MongoDB. Accessed: Nov. 16, 2024. [Online]. Available: <https://www.mongodb.com/resources/basics/databases/nosql-explained>

[4]‘NoSQL Vs SQL Databases’, MongoDB. Accessed: Nov. 16, 2024. [Online]. Available: <https://www.mongodb.com/resources/basics/databases/nosql-explained/nosql-vs-sql>

[5] T. Lagos Jenschke, M. Dias de Amorim, and S. Fdida, ‘Nearby connections strategies: Features, usage, and empirical performance evaluation’, *Internet of Things*, vol. 23, p. 100895, Oct. 2023, doi: [10.1016/j.iot.2023.100895](https://doi.org/10.1016/j.iot.2023.100895).

[6] ‘Overview | Nearby Connections’, Google for Developers. Accessed: Nov. 16, 2024. [Online]. Available: <https://developers.google.com/nearby/connections/overview>

[7] ‘Mapbox vs Google Maps — What are the differences?’, SoftKraft. Accessed: Nov. 18, 2024. [Online]. Available: <https://www.softkraft.co/mapbox-vs-google-maps/>

[8]H. Mehta, P. Kanani, and P. Lande, ‘Google Maps’, *IJCA*, vol. 178, no. 8, pp. 41–46, May 2019, doi: [10.5120/ijca2019918791](https://doi.org/10.5120/ijca2019918791).

[9] J. Vargas, S. Srivastava, D. Tuia, and A. Falcao, ‘OpenStreetMap: Challenges and Opportunities in Machine Learning and Remote Sensing’, *IEEE Geosci. Remote Sens. Mag.*, vol. 9, no. 1, pp. 184–199, Mar. 2021, doi: [10.1109/MGRS.2020.2994107](https://doi.org/10.1109/MGRS.2020.2994107).

[10] ‘Android vs iOS Development: Which Should I Learn First?’ Accessed: Nov. 17, 2024. [Online]. Available: <https://www.upwork.com/resources/android-vs-ios-which-should-i-learn-first>

[11] X. Li, X. Zhang, X. Ren, M. Fritsche, J. Wickert, and H. Schuh, ‘Precise positioning with current multi-constellation Global Navigation Satellite Systems: GPS, GLONASS, Galileo and BeiDou’, *Sci Rep*, vol. 5, no. 1, p. 8328, Feb. 2015, doi: [10.1038/srep08328](https://doi.org/10.1038/srep08328).

[12] Z. Ozdemir and B. Tugrul, ‘Geofencing on the Real-Time GPS Tracking System and Improving GPS Accuracy with Moving Average, Kalman Filter and Logistic Regression Analysis’, in *2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, Ankara, Turkey: IEEE, Oct. 2019, pp. 1–6. doi: [10.1109/ISMSIT.2019.8932766](https://doi.org/10.1109/ISMSIT.2019.8932766).

[13] M. B. Garcia, ‘Location-based marketing using mobile geofencing: lessons learned from a user-centred application development research’, *International Journal of Technology Marketing*, vol. 17, no. 1, pp. 1–29, Jan. 2023, doi: [10.1504/IJTMKT.2023.127322](https://doi.org/10.1504/IJTMKT.2023.127322).

[14] ‘What is GDPR, the EU’s new data protection law?’, GDPR.eu. Accessed: Nov. 28, 2024. [Online]. Available: https://gdpr.eu/what-is-gdpr/

[15]

L. A. Tawalbeh, A. Basalamah, R. Mehmood, and H. Tawalbeh, ‘Greener and Smarter Phones for Future Cities: Characterizing the Impact of GPS Signal Strength on Power Consumption’, *IEEE Access*, vol. 4, pp. 858–868, 2016, doi: [10.1109/ACCESS.2016.2532745](https://doi.org/10.1109/ACCESS.2016.2532745).

[16] J. Sharkey, ‘Coding for Life--Battery Life, That Is’. http://files.blogjava.net/jicheng687/W\_0300\_CodingforLife-BatteryLifeThatIs.pdf

[17] ‘What is Agile? | Agile 101 | Agile Alliance’. Accessed: Nov. 26, 2024. [Online]. Available: <https://www.agilealliance.org/agile101/>

[18] ‘https://www.idpublications.org/wp-content/uploads/2015/05/Agile-Software-Development-Methodology.pdf’. Accessed: Nov. 26, 2024. [Online]. Available: <https://www.idpublications.org/wp-content/uploads/2015/05/Agile-Software-Development-Methodology.pdf>

[19] ‘https://www.researchgate.net/profile/Udesh-S-Senarath/publication/353324450\_Waterfall\_Methodology\_Prototyping\_and\_Agile\_Development/links/60f41f71fb568a7098b9d035/Waterfall-Methodology-Prototyping-and-Agile-Development.pdf’. Accessed: Nov. 26, 2024. [Online]. Available: <https://www.researchgate.net/profile/Udesh-S-Senarath/publication/353324450_Waterfall_Methodology_Prototyping_and_Agile_Development/links/60f41f71fb568a7098b9d035/Waterfall-Methodology-Prototyping-and-Agile-Development.pdf>

[20] A. T. de Aquino, J. A. L. Barboza Júnior, N. de Araújo Moreira, and P. Peixoto Praça, ‘Impacts of GPS module on energy consumption and machine-learning based battery lifetime estimation’, 2023, doi: [10.48545/ADVANCE2023-FULLPAPERS-3\_3](https://doi.org/10.48545/ADVANCE2023-FULLPAPERS-3_3).