CA400 Functional Specification

Project Title:	Protego
Name, Student number:	Emma Ablitt, 17356661 Mikolaj Adamski, 17441622
Supervisor:	Renaat Verbruggen

0 Table of Contents 1

1. Introduction

1.1 Overview

Personal safety is increasingly becoming a major concern in society today. Men and women of all ages are finding themselves in vulnerable situations on a daily basis. Whether it be walking home alone, lost in a new neighbourhood or simply going about day-to-day errands, everyone has experienced some level of stress due to their personal safety being compromised.

Knowing that you're carrying an application in your pocket that will provide help in an emergency, can bring peace of mind and relieve some stress when out and about alone in the world today. This is the main objective of our application, Protego.

The following functional specification will have a highly detailed description of the project, system requirements and the overall design of Protego.

1.2 Scope

Protego is a cross-platform personal safety application.

Built using the React Native framework, this voice activated iOS and Android app will be designed for all to use. Users shall provide Protego with a detailed list of their selected "guardians", including their name and phone number. If a user has activated SOS mode, whether by voice activation or manually through the application, these guardians will be alerted and have access to the user's live location until the user has exited SOS mode. Users themselves will have access to numerous functions while in SOS mode, including having the ability to make a call to the emergency services or being able to take a photograph to send to their guardians. By being able to send a photo of their exact location, a user can inform their guardians of where they are within a building.

To exit SOS mode, a user will have to provide their password to ensure the user is in a safe position and has the ability to exit SOS mode themselves.

The objective of this application is to make a difference in a person's life when they are feeling unsafe and are at risk, by contacting those closest to them and allowing a user immediate contact with emergency services without having to remove their mobile phone from their pocket.

1.3 Glossary

API Application Program Interface is a set of protocols and tools for building software applications. An API specifies how software components should interact.

Backend The backend refers to the part of the application that the user doesn't get to see.

Firebase A google cloud based application development platform which acts as a server for the program.

GUI Graphical User Interface is how users interact with the application using both images and text.

IDE Integrated Development Environment is a software application that provides developers with a range of tools, typically including a code editor and debugger.

Javascript is the language Protego will be developed in.

React Native is an open source javascript framework used to develop iOS and Android applications.

2. General Description

2.1 System Functions

The following is the initial list of functions available in Protego. Details of these functions are expanded in section 3.

- Guardians
- SOS Mode
  - Voice activation/Manual
- Live location
- Password required OK button
- Camera

- Fake call
- Emergency text
- User profile
- Push Notifications

2.2 User Characteristics & Objectives

Protego will be designed with all members of society in mind. Personal safety is an issue that affects everybody and so this application must cater and attract users of all ages and abilities. We aim to make Protego easy to use to accommodate users who are not as skilled in technology. To do this we will develop a simple UI that the user can easily navigate from the moment they enter the app. To aid us with our UI decisions we will refer back to our previous module CA106 (Web Design). From this we will manage to use web safe colours, avoid tricky layouts and limit the amount of text a user comes into contact with. We also plan to implement multiple language support.

For the application to function properly, the user's device needs to be connected to the internet at all times, with location on. The user will need to give location and call permission when they first enter the application.

2.3 Operational Scenarios

To have access to the following application features, users must have successfully downloaded the application onto their mobile device.

New users

- *Launching the application*
  When the user launches the application for the first time, they will be greeted with a registration screen. The user will then be prompted to accept the call and location permissions. Existing users will be sent to the main menu.

Existing users

- *Add Guardian(s)*
  In order to create a safe environment for a user, they must set up their list of guardians. If a user triggers SOS Mode, a guardian will be notified by SMS.

- *Trigger SOS Mode*
  To activate SOS Mode, a user can choose, depending on the situation to do so through voice activation or manually through the app. Once it is triggered, all guardians will receive an alert.

- *Deactivate SOS Mode*
  Once a user feels safe to do so, they can deactivate SOS Mode. This action must be completed within the application to apply additional safety features.

- *User makes a fake call*
  If a user feels unsafe in a situation but does not wish to trigger SOS Mode, or if they are in a position where they feel they are unable to, they can generate a fake call for a swift exit.

- *User edits their profile*
  When a user navigates to their own profile screen, they can change their details. These details include;
    - Add/Remove/Edit guardians
    - Change their safe word
    - Edit their details

2.4 Constraints

Below are the constraints that we have associated with the development of the project.

- **Time Constraints**
  One of the main issues with this project is having it completed before the deadline. Due to lack of experience with some of the implementation tools, we will need time to learn how to efficiently use these tools. We

plan to first implement the basic structure of the application before incorporating more features as we see fit.

- **Firebase**
  As we are using the free version of Firebase, our database size is limited. This affects the number of users we can store, including test cases.

- **Internet Access**
  Due to our application taking advantage of location services, if there was a loss of internet connection, it would no longer be possible to use these services. Loss of internet access whilst in SOS Mode would result in guardians only having access to the users last known location before their internet connection was lost.

3. Functional Requirements

3.1 Registration

- **Description**
  After launching the application for the first time, the user will be prompted to register. They will need to enter a username, password and their phone number. The username will be used to inform guardians who are in need of help. The password will be used to deactivate SOS mode when needed. Once the user has completed registration, they will be redirected to the application's home screen.
- **Criticality**
  Without the user registering and providing their details, the application would not be able to run it's features. It also enforces that each user can only be registered once.
- **Technical Issues**
  We will be using Firebase for this part of the application. We will take advantage of both the Firebase Database and Authentication ADK. If there was an issue with either of these, there would be an issue in registering the user.
- **Dependencies**

The user must have successfully downloaded the application to their mobile device.

3.2 Add Guardian

- **Description**
  One of the main features of the application is creating a list of guardians. When adding a guardian, a user will be prompted to provide a name and phone number. These details will be used when contacting said guardian, if needs be. Users can choose up to 5 guardians.
- **Criticality**
  Without a list of guardians, there will be no alerts sent out if the user was to trigger SOS Mode, rendering SOS Mode almost useless.
- **Technical Issues**
  Users must provide the correct details for their guardians. This feature also makes use of the Firebase Database.
- **Dependencies**
  The user must be registered.

3.3 User Profile

- **Description**
  A user can edit certain details from the application. They can manage their guardians, which includes; adding a new guardian, editing an existing guardians details or removing a guardian from their list. A user can also edit their keyword for triggering SOS Mode. If a user wishes to change their username, password or phone number they can do so within their profile.
- **Criticality**
  These details are incredibly important and must be entered correctly.
- **Technical Issues**
  Changes must be properly made within the database.
- **Dependencies**
  Users must be previously registered.

3.4 Fake Call

- **Description**
  If a user wishes to generate a fake call, they must do so through the application. Once the user taps to start a fake call, their phone will play a ringtone. When they tap the screen again, their phone will show a default screen appearing as if a user is on a phone call. The user can then tap the screen again to end the fake call.
- **Criticality**
  Users can make use of this feature in situations they wish to leave quickly, such as being harassed by a stranger.
- **Technical Issues**
  At the moment we plan on using react-native-sound to play the ringtone.
- **Dependencies**
  Users must be registered with the application.

3.5 Enter SOS Mode

- **Description**
  SOS Mode is available for users who feel their personal safety is threatened and wish to inform those close to them that they feel unsafe. Once SOS Mode is triggered, an SMS will be sent to the user's list of guardians to alert them of this. Guardians will also be able, if they also have Protego downloaded, to watch the user's live location while SOS Mode is activated. User's will have the ability to take a picture and have it sent to all of their guardians. This can aid users who are inside a building to show their guardians exactly where they are. Emergency call features are also available.
- **Criticality**
  SOS Mode is the main function of the application. When a user feels unsafe it is critical that it, and all of its features, work to a high standard.
- **Technical Issues**
  SOS Mode makes use of location services, sms and various react native libraries. If a user was to lose their internet connection or their phone service, these features would not be available.
- **Dependencies**
  Users must be registered. User's must have an internet connection. User's must have given their location permission.

3.5.1 Voice Activated

- **Description**
  Users can trigger SOS Mode by saying their keyword aloud. This is useful when a user has their mobile phone in their pocket but wishes to quickly notify their guardians of their whereabouts.
- **Criticality**
  It is highly important that this feature works correctly, given the circumstances that it will be used under are potentially high risk.
- **Technical Issues**
  We are initially looking to use react-native-voice library but may use a different library.
- **Dependencies**
  Users must be registered with the application.

3.5.2 Manually

- **Description**
  If the user wishes to activate SOS Mode but doesn't feel safe using their keyword, they are able to trigger it through the application. Once they launch the app, they can press the SOS button and SOS Mode will be triggered the same as if it was voice activated.
- **Criticality**
  Critically important these features work as to not prevent help in an emergency.
- **Technical Issues**
  All features must work the same as if SOS Mode was triggered by voice.
- **Dependencies**
  Users must have successfully registered.

3.6 Live location

- **Description**
  When SOS Mode is triggered, the user's guardians are able to access the user's live location. If the guardian wishes to find the user, they can keep

track of where they are. Or should the guardian wish to involve authorities, they can inform them of the user's location. This feature can also simply be for the user to feel safe travelling alone, knowing that a loved one is watching over them

- **Criticality**
  Highly important that this is implemented correctly for both guardians and users.
- **Technical Issues**
  We plan to implement this using react-native-maps as well as the realtime database provided by Firebase in order to keep updating the current location of the user.
- **Dependencies**
  Users must have registered correctly and accepted all the related permissions. Guardians must also have registered successfully to be able to access the user's location.

3.7 Exit SOS Mode

- **Description**
  If a user wishes to deactivate SOS Mode when they no longer feel at risk, they must do so within the application. Users must press the 'I'm OK' button within the app. When they do so they'll be prompted to enter the password they created at registration, or has since changed.
  By requiring authentication to release the safety features, we feel it is an added layer of protection for the user as their phone can't be stolen and location turned off. It also requires the user to be in a headspace where they can remember their password, meaning they feel safe again. When SOS Mode is deactivated, guardians will once again receive a notification by SMS.
- **Criticality**
  Important that the user can turn off SOS Mode when they feel safe.
- **Technical Issues**
  The application must be able to turn off all features activated by users whilst in SOS Mode. Users must have working internet connection to send the notification to their guardians that they are now safe.
- **Dependencies**

Users must be registered and triggered SOS Mode. Stable internet connection is required to complete efficiently.

3.8 Emergency text

- **Description**
  If a user feels in danger and wishes to involve the authorities, they will be able to avail of the 112 emergency text services. It is not always possible for a person in danger to speak over the phone, they could be hiding or perhaps would like to be subtle about contacting emergency services.
- **Criticality**
  Very high importance as the user may be in a dangerous situation and need immediate attention. Exact location can be sent in the text message, quickening up the process.
- **Technical Issues**
  Relies on the react-native-geolocation library to attach user's location to the SMS
- **Dependencies**
  User must be successfully registered with the application in order to pass their information to the 112 services. Correct permissions must be given for location also.

3.9 Camera

- **Description**
  Whilst in SOS Mode, users will be able to capture an image and send it their list of guardians. The idea behind this is if a user is in a building, the map will show a guardian which building they user is in, but not their exact location within the building. A user sending a picture of what surrounds them solves this issue.
- **Criticality**
  If a user is trapped in a position that they can not be found without their exact location being known, it is crucial this feature works correctly.
- **Technical Issues**
  We plan on using the react-native-camera library for this feature.
- **Dependencies**

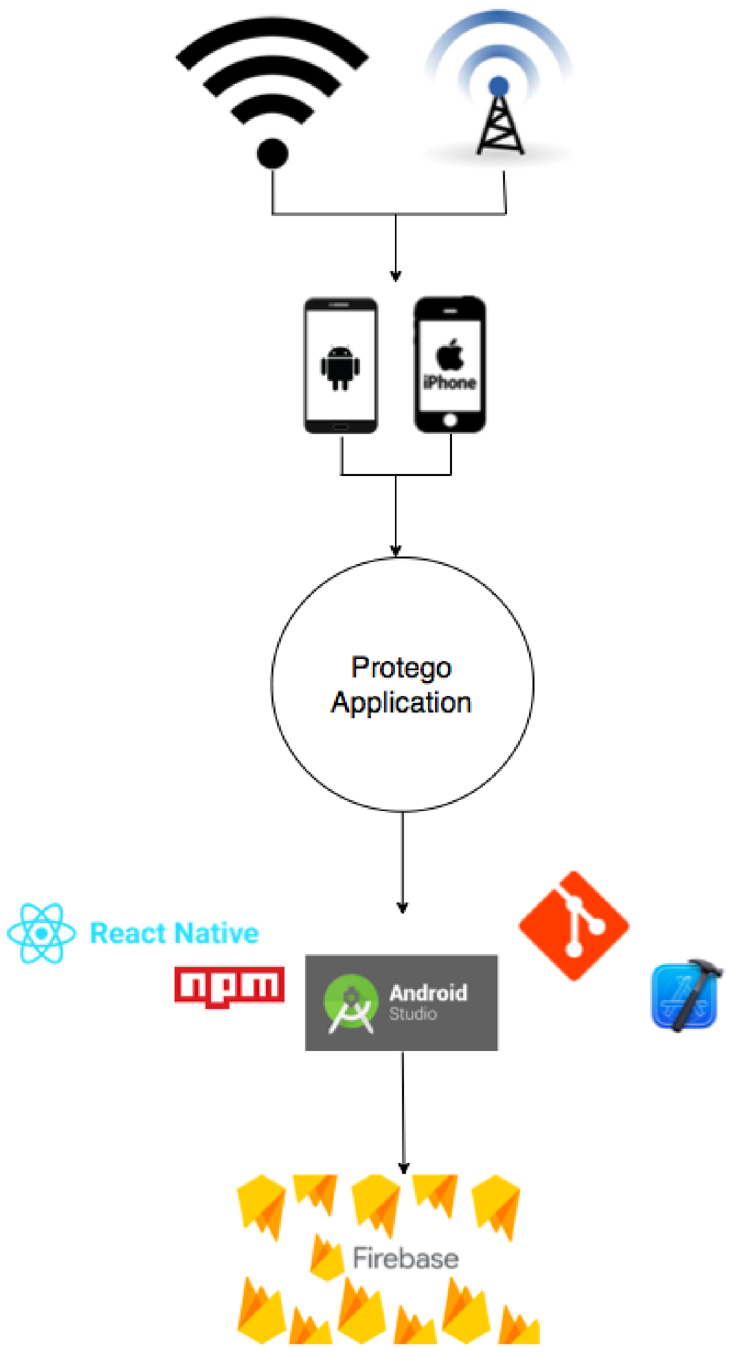
A user must first be registered correctly. The user’s device must also have a working camera, and they must have given the correct permissions.

3.10 Help Manual

- **Description**
  For new users or users that haven’t used the application in a while there will be a help manual provided on the application.
- **Criticality**
  Very important as if the application is working correctly, the user has all the correct permissions and a working connection but they are unable to navigate the application, it may not help in a scary situation. We plan to implement a very simple application to navigate but for those who are not familiar with technology, a help manual could go a long way.
- **Technical Issues**
  No technical issues can be foreseen at this moment.
- **Dependencies**
  A user must be registered correctly.

4. System Architecture

4.1 System Architecture Diagram

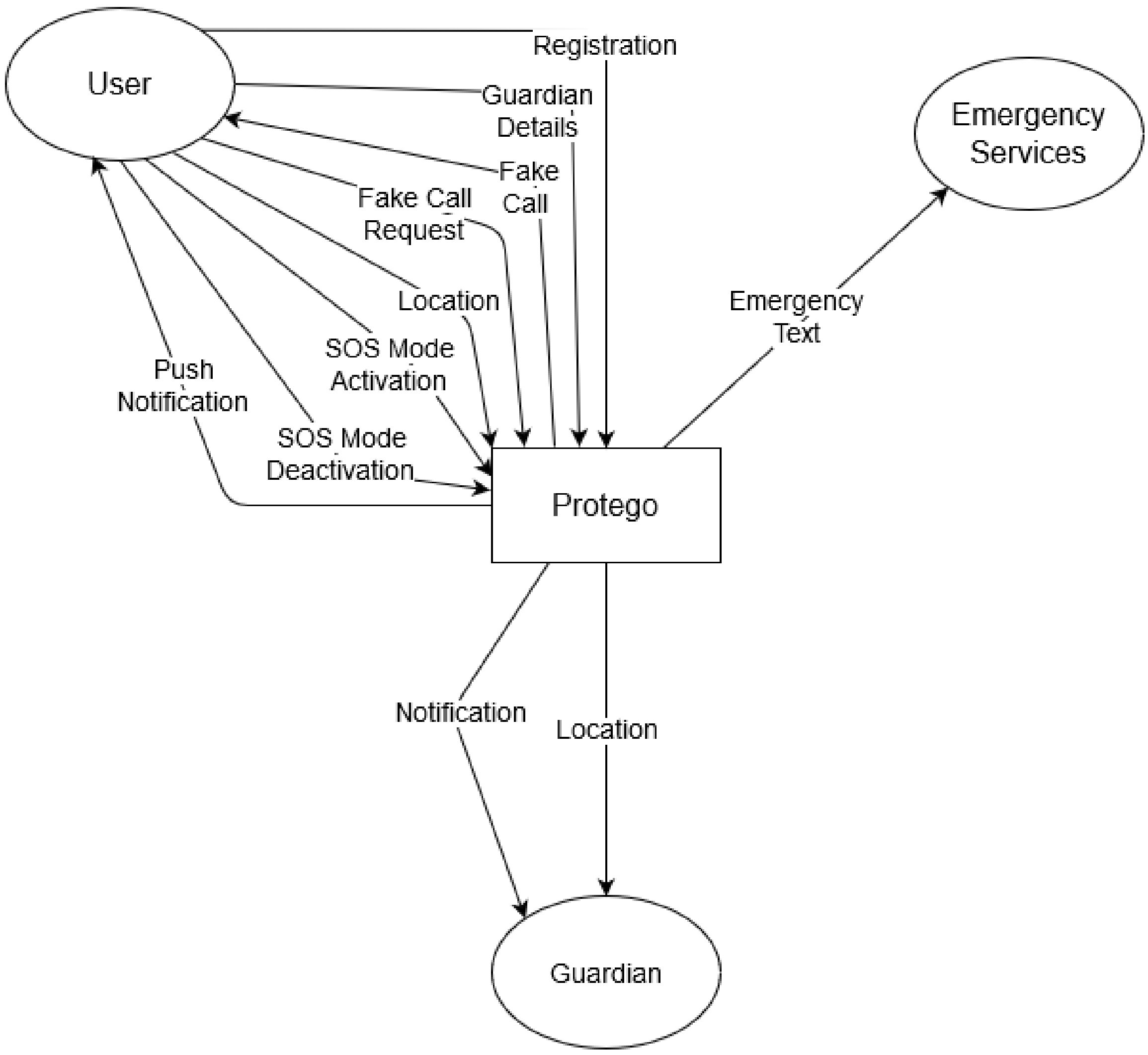


The above diagram represents the structure and relationships of Protego.
The user, on either an Android or Apple device, has a stable internet connection and a solid network connection. They can then download and run the Protego application.

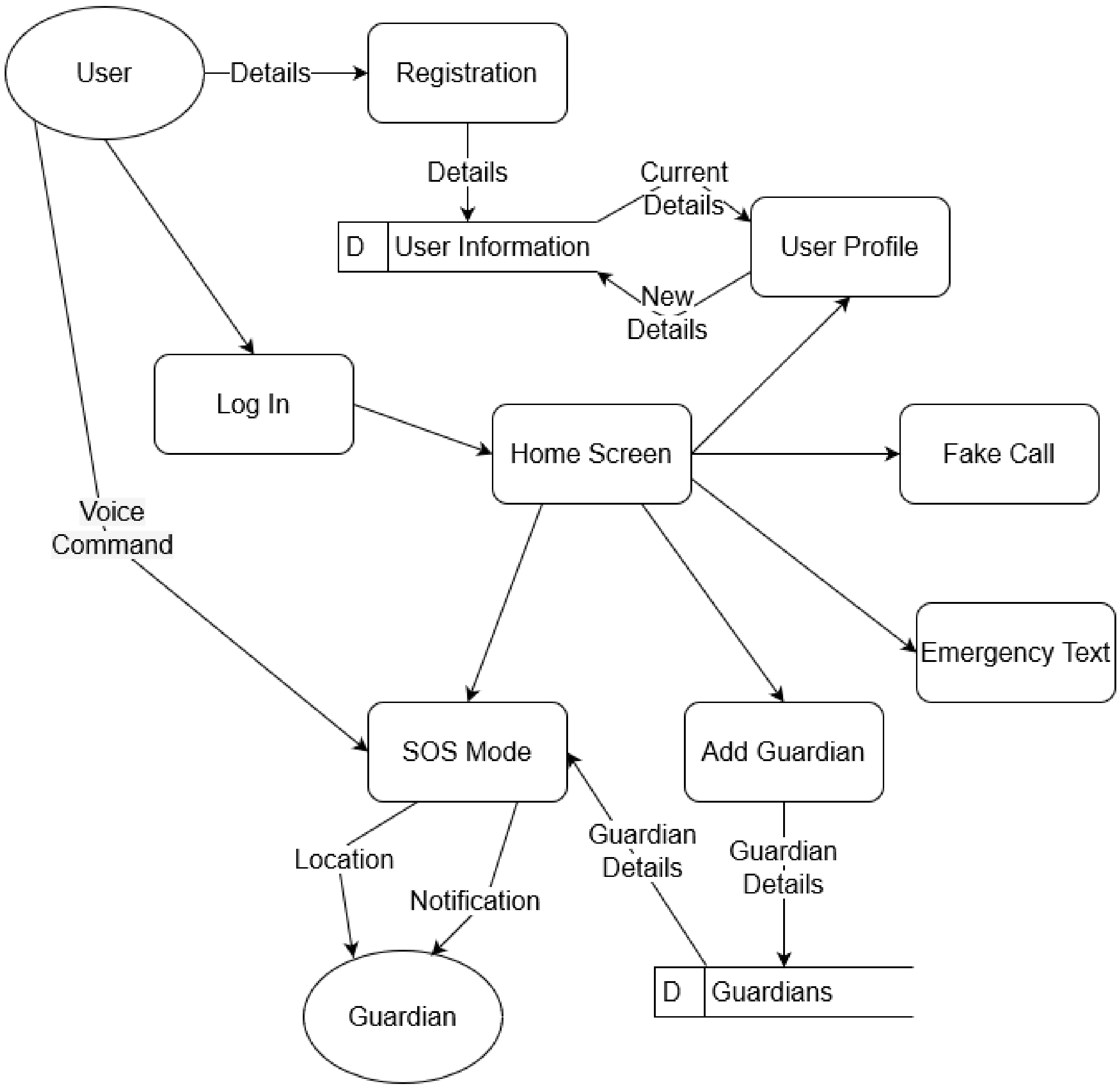
Protego then makes use of various software applications including React Native, npm, Android Studio, Javascript, Git and XCode. Our codebase will be in Git, while the iOS IDE will be XCode. We will be making use of various React Native libraries throughout the development. The application will be connecting with Firebase to retrieve and store information in the database as well as other features including authentication and real time information. Firebase will also act as the host for the application.

5. High-Level Design

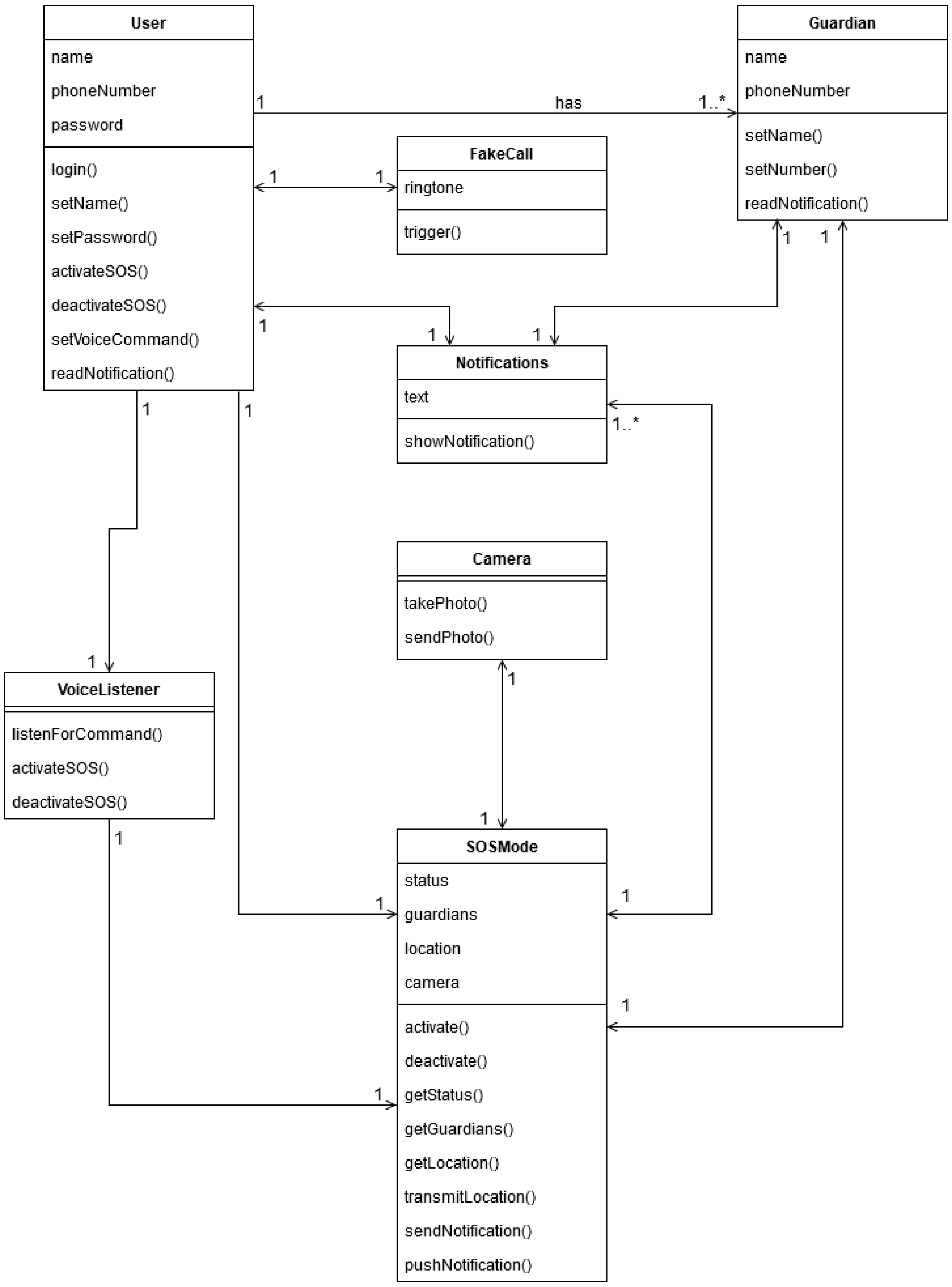
5.1 Context Diagram



5.2 Data Flow Diagram

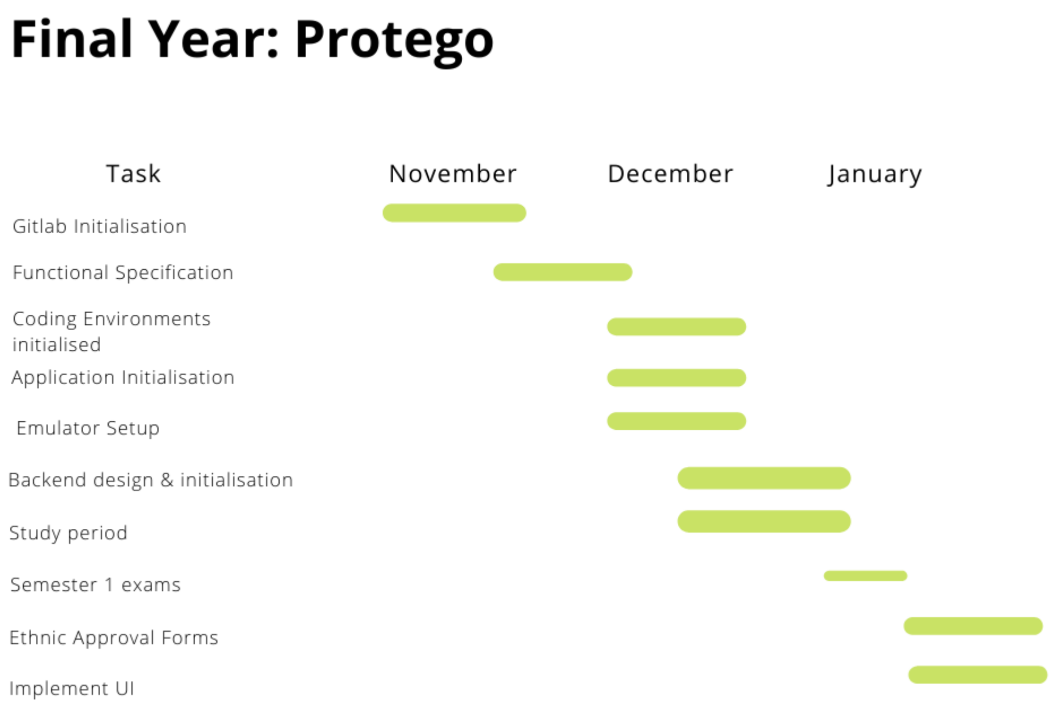


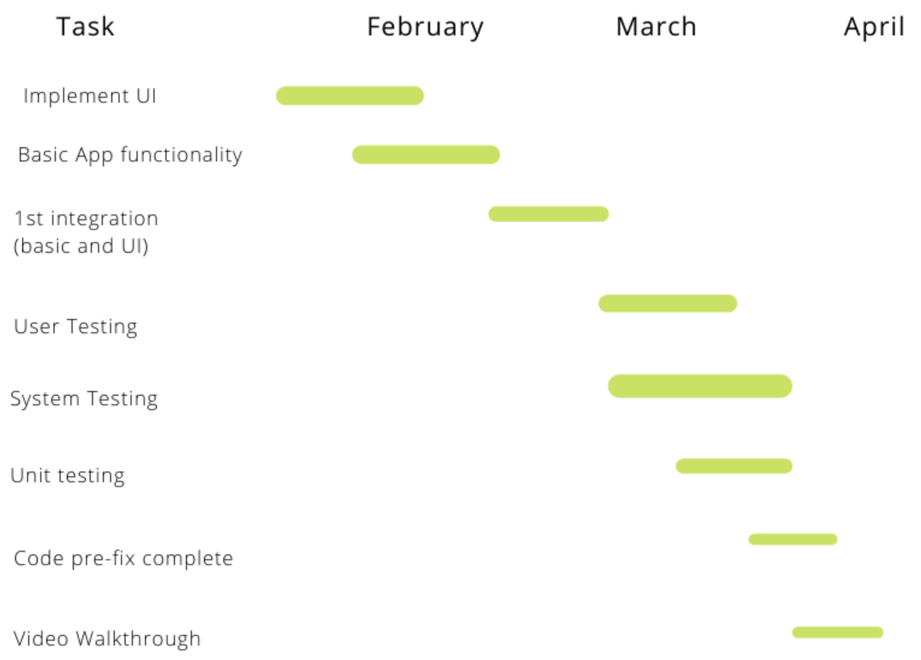
5.3 Class Diagram

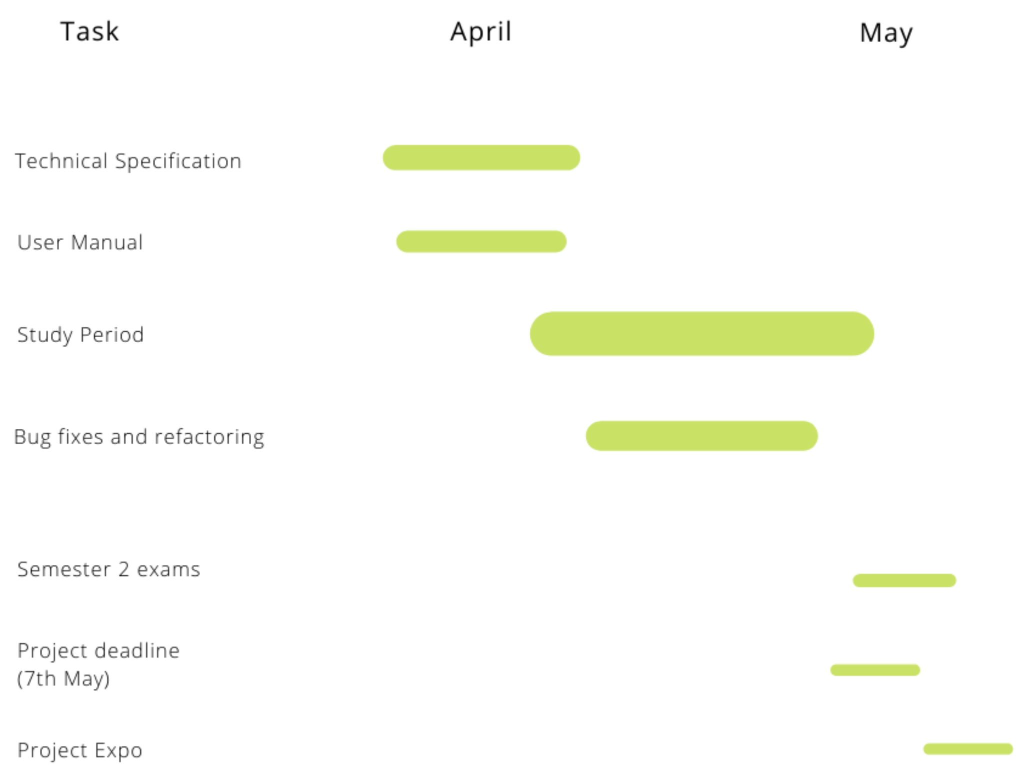


6. Preliminary Schedule

6.1 Gantt Chart







6.2 Task Timetable

Component	***Start Date***	***End Date***	***Duration(days)***
Coding Environments setup	5/12/2020	12/12/2020	7
Application Initialisation	7/12/2020	14/12/2020	7
Emulator Setup	7/12/2020	14/12/2020	7
Backend Design & Initialisation	15/12/2020	21/12/2020	6
Study Period	22/12/2020	4/1/2021	13
Semester 1 Exams	5/1/2021	9/1/2021	4
Ethnic Approval Forms	16/1/2021	30/1/2021	14
Implement UI	16/1/2021	1/2/2021	16

Basic Functionality	2/2/2021	28/2/2021	26
1st Integration	1/3/2021	7/3/2021	6
User Testing	8/3/2021	14/3/2021	6
System Testing	15/3/2021	21/3/2021	6
Unit Testing	22/3/2021	26/3/2021	4
Code prefix complete	2/4/2021	2/4/2021	-
Video Walkthrough	5/4/2021	7/4/2021	3
Technical Specification	8/4/2021	14/4/2021	6
User Manual	15/4/2021	18/4/2021	3
Study period	19/4/2021	3/5/2021	14
Bug fixes & Refactoring	19/4/2021	6/5/2021	17
Semester 2 Exams	4/5/2021	17/5/2021	13
Project Deadline	7/5/2021	7/5/2021	-
Project Expo	17/5/2021	?	-