ASHESI

ASHESI UNIVERSITY

VERIFLY: A DRONE-SUPPORTED FIRST AID SYSTEM FOR ASHESI UNIVERSITY

APPLIED PROJECT

B.Sc. Management Information Systems

Verissa Marian Owusu

2019

ASHESI UNIVERSITY

Verifly: A Drone-supported First Aid System for Ashesi University

APPLIED PROJECT

Applied Project submitted to the Department of Computer Science, Ashesi University College in partial fulfilment of the requirements for the award of Bachelor of Science degree in Management Information Systems

Verissa Marian Owusu

April 2019

DECLARATION

I hereby declare that this Applied project is the result of my own original work and that no part of it has been presented for another degree in this university or elsewhere.

Candidate's Signature:

………………………………………………………………………………………

Candidate's Name:

………………………………………………………………………………………

Date:

………………………………………………………………………………………

I hereby declare that preparation and presentation of Applied project were supervised in accordance with the guidelines on supervision of Applied project laid down by Ashesi University College.

Supervisor's Signature:

………………………………………………………………………………………

Supervisor's Name:

………………………………………………………………………………………

Date:

………………………………………………………………………………………

Acknowledgement

I would like to thank God for strength throughout my Ashesi education. I would also like to thank my family for their support throughout this process. I would also like to thank my supervisor, Dr Adjepon- Yamoah, for his guidance and assistance during this entire process. Finally, I would like to thank Albert Cole, Nana Araba Toffah, Josephine Abbey, Victor Hazel, Rahul Srinivas, Emmanel Agamah, Karin Nartey, Hudson Lekunze and Isaac Coffie for all their help in completing this project. I could not have done it without you all.

Abstract

Drones are autonomous vehicles that have been beneficial to humans since their creation. Drones are considered to carry out activities efficiently. In the last ten years, drones are becoming a common sight. Filming for TV and Movies, surveillance, delivery, and hobby flight are some tasks which make use of drones. The potential of drones is slowly, but surely being uncovered and yet we`ve just scratched the surface. Companies like Amazon use drones to make deliveries more efficient.

Drones can make standard services efficient, like in healthcare, public security, etc. This project utilizes drones for deploying emergency healthcare and uses technologies like GPS location, mobile applications, web services, drones, and ground control stations. The system is designed to aid first responders to an incident or injured people. This project also introduces a drone policy to govern the use of drones at Ashesi University.

Table of Contents

Chapter 1: Introduction

1.1 Background context

The World Health Organization reported in 2010 that approximately 5.8 million people die worldwide every year, with injuries. 10% of these deaths and a quarter are a result of road traffic accidents, which mainly occurs in Sub-Saharan Africa. The WHO report also stated that the use of timely and efficient ambulance and emergency care could avert the high mortality rate [1].

Ghana, a developing country in sub-Saharan Africa is a country that has a lot of challenges which can be solved using technology. Thus, in this paper, some applied technological solutions to some common countrywide problems in developed countries will be remodified to fit the African context, particularly Ghana. According to a national road safety report of 2007 indicated that at least six people are killed daily in road traffic accidents (RTAs) [5].

More specifically, the limited number of ambulances in Ghana and sparse road networks limit the ability to provide efficient ambulance and emergency care, through the traditional process of using paramedics and ambulances. In this paper, I propose a solution to this problem which is an alternative means to the conventional method of carrying out emergency care. I carry out a project in the smaller Ashesi University community to test the feasibility of the solution, which also serves as a good sample for a case study since the risk of injuries is also prevalent in this space as injuries can occur anywhere.

1.2 Motivation

The mortality rate in Ghana has been on the rise for some years now. This increase has been attributed to the deaths by road accidents and the late response time of paramedics onto the accident scenes due to either the lack of ambulances for transportation or sparse road networks. This setback hampers on the ability of the Ghanaian health sector to reduce these mortality rates by implementing quick and efficient emergency healthcare.

To narrow the scope of this problem, I decided to focus on Ashesi University's emergency care system and review it based on user experiences and propose a more efficient means to compliment the Natembea health centre's processes. A survey was carried out to find out from members of the community concerning their experience with getting first aid when injured. This survey was carried out mainly to find out if users were satisfied with the current process for obtaining emergency care on the Ashesi University campus. The first aid care procedure on campus currently is such that when a member of the community is injured on campus or anywhere near Ashesi property during the day, he or she must go to the health centre to get treatment. Whereas at night, the on-campus hostel residents are catered to by the Hostel lobby manager and off-campus residents are largely left to fend for themselves. From Figures 1.1, 1.2 and 1.3 below, most students highlighted the challenge to first aid on and off-campus, and also expressed the unanimous need for faster access to first aid.

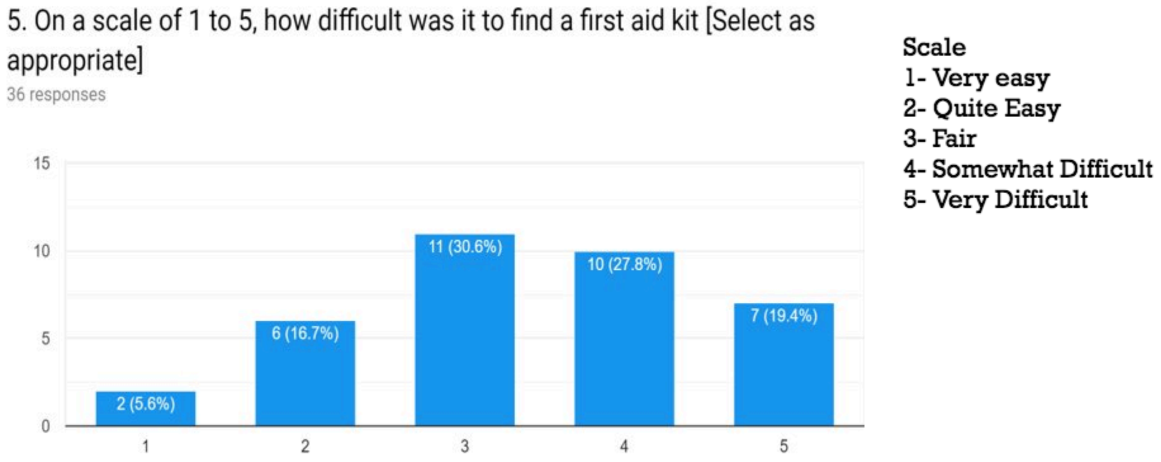
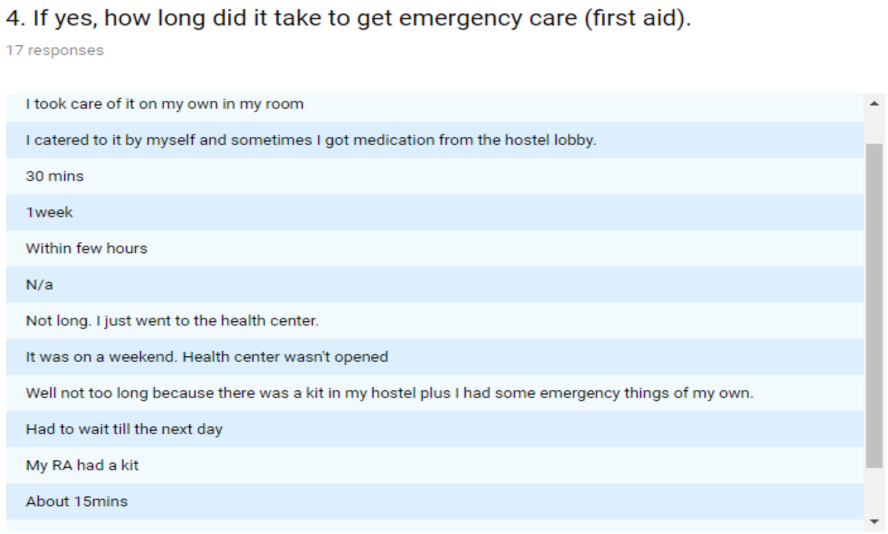


Figure 1.1: Difficulty level of getting first aid kits



4. If yes, how long did it take to get emergency care (first aid).

17 responses

I took care of it on my own in my room

I catered to it by myself and sometimes I got medication from the hostel lobby.

30 mins

1week

Within few hours

N/a

Not long. I just went to the health center.

It was on a weekend. Health center wasn't opened

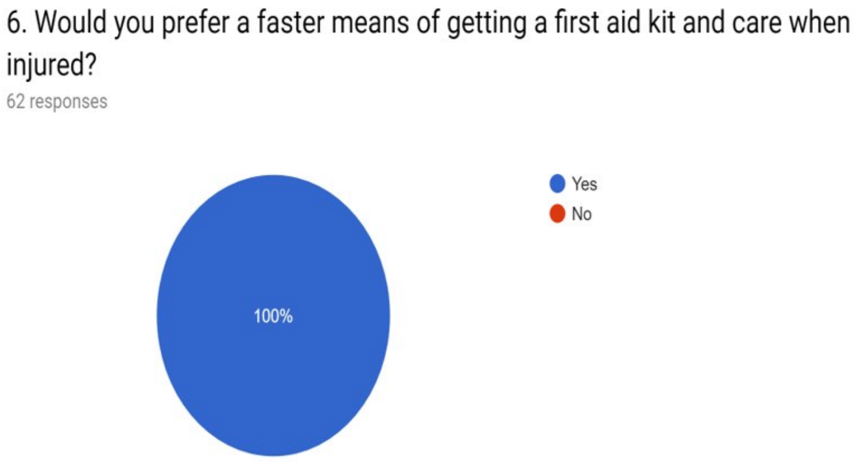
Well not too long because there was a kit in my hostel plus I had some emergency things of my own.

Had to wait till the next day

My RA had a kit

About 15mins

Figure 1.2: Waiting time before getting first aid care



6. Would you prefer a faster means of getting a first aid kit and care when injured?

62 responses

Figure 1.3: Respondents answer

1.3 Related Work

Some unmanned aerial delivery systems have been carried out in countries to help make delivery of goods much more efficient. This section looks at some of these projects which are of importance to this project.

1.3.1 First Aid Drone for Outdoor Sports Activities

The unmanned aerial assistance system proposed in this paper uses drones to assist athletes in cases of minor health problems, like sprains, and abrasions, or in more severe cases, such as, dehydration and heat strokes [2]. Athletes' details are registered into the system, and upon the request of the service to provide care for injuries through an SOS emergency button, then a drone is sent to the registered athlete's location.

1.3.2 The Disaster Project

The European project call for proposals on "Remotely Operated CBRNe Scene Assessment Forensic Examination" incited many submissions, one of which is the "Disaster" project [3]. The "Disaster" project is intended to deal with a wide range of disasters and aid the first responders in their task. Time is of the essence in all types of emergencies, so the project aims to make the responders work both more efficiently, especially concerning time, and safer. The project focuses on a few specific scenarios that have been selected to represent a wide variation on challenges.

1.3.3 Medical Drones Systems for amusement parks

The Medical Drones System is a system implemented in areas such as skiing resorts and amusement parks. These places are usually filled with tourists who are scattered everywhere enjoying their time [4]. As such in the event of an emergency, it is quite challenging to reach the patient or injured person quickly. There is a lot of time involved in locating injured people and not considering the extra time because of long distances and paths. Which could be time-consuming and life-endangering. As a result, the medical delivery system was designed to autonomously locate injured people, which will possibly save effort and time and eventually help to rescue wounded people successfully.

1.4 Proposed Solution

I propose a drone supported first aid system (i.e., Verifly) to assist the Natembea health centre at Ashesi University in delivering emergency care to injured persons as well as provide these wounded persons with a faster means of getting first aid. This proposed system presents mobile applications for users (i.e., the requester and health centre personnel).

The Verifly project will serve as a complementary service to the Natembea Health Centre's existing emergency care practices. The project aims to increase the level of efficiency in delivering first aid care on the university's campus. The solution is a healthcare delivery system which consists of mobile applications and a ground control station for requesting healthcare services and deploying them. The project will also contain a drone policy to ensure the safety of the Ashesi community, i.e., Students, Staff, and Faculty while drones are in the airspace. This system will help reduce emergency care response time and reduce injury complications and, or risks associated with injuries due to delayed healthcare.

Chapter 2: Requirement Specification

2.1 Introduction

The proposed system, Verifly, has two main components, that is the hardware and the software component. The software component is the medium through which a user requests for first aid kit and care assistance and the hardware component is responsible for carrying the first aid kit to the user at a specified location.

2.2 Overall Description

Verifly is a system that readily makes use of a mobile application to engage users in interaction. Users must download the app onto their mobile devices. Upon the first installation and launch, users are asked to sign up, which serves as a form of registration. This activity is done to collect some vital information from users and to assist the Health centre personnel in carrying periodic check-ups on users of the system.

When a user requests for a first aid kit equipped drone on the application, the current location of the user is collected and sent to the ground control station. At the ground control station, the system checks if there is a drone in the air already which has finished a task already and sends the coordinates to that drone to the user's location.

If there is no drone in the air at the moment, an inactive drone at the ground control station is equipped with a first aid kit and given a command to go to the user's location. When the drone gets to the destination, the user takes off the kit and uses it. The user reassembles the first aid kit, and a button on the application is clicked when done to prompt the return of the drone to the ground control station. The overall description is depicted in Figure 2.1 below.

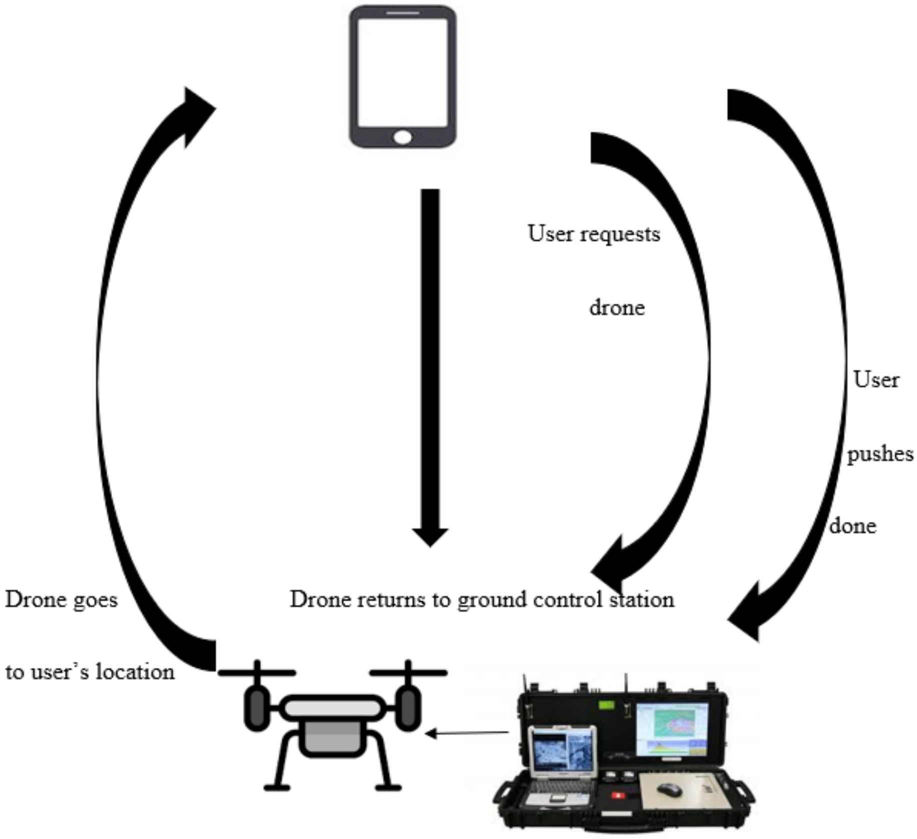


Figure 2.1: Overview of the proposed solution

2.2.1 Hardware component

The hardware component of the system is made up of the drone and the electronic devices needed for the ground control station. The drone is responsible for the transportation of the first aid kit, and the ground control station is for scheduling flights of the drones.

2.2.2 Software component

The software component of the system is made up of a mobile application which lets users request for a first aid kit. The app will also have a functionality that takes the user's GPS coordinates and sends that to the ground control station. The mobile application will also allow administrators (i.e., the health centre personnel) to start the automated flight process to the request locations and also will enable them to create usage reports.

2.2.3 Key features

As part of the survey undertaken, respondents were asked about some features they would like to have on the mobile application, that is from the perspective of someone who would request first aid. Figures 2.2, 2.3, 2.4 and 2.5 depict some responses. Basic critical features of the system were identified to deliver the solution. In addition to this, a survey was carried out to find out any additional features users would like to have on the application.

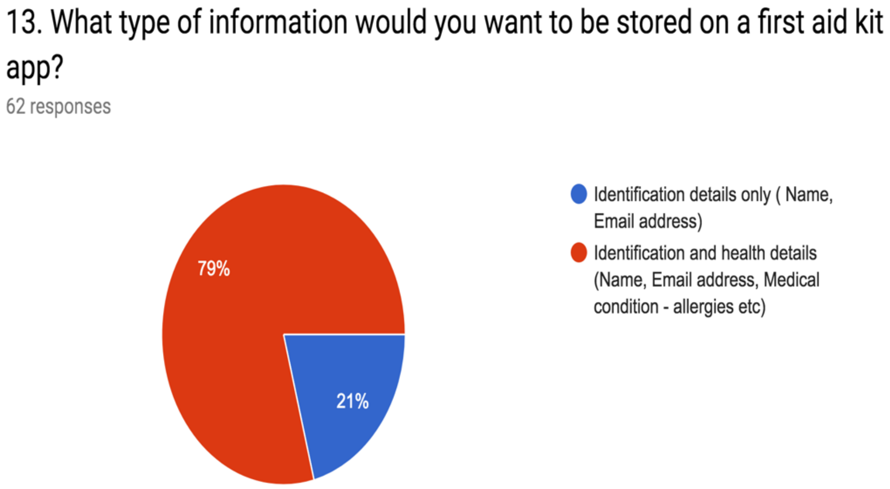


Figure 2.2: Identification details preference

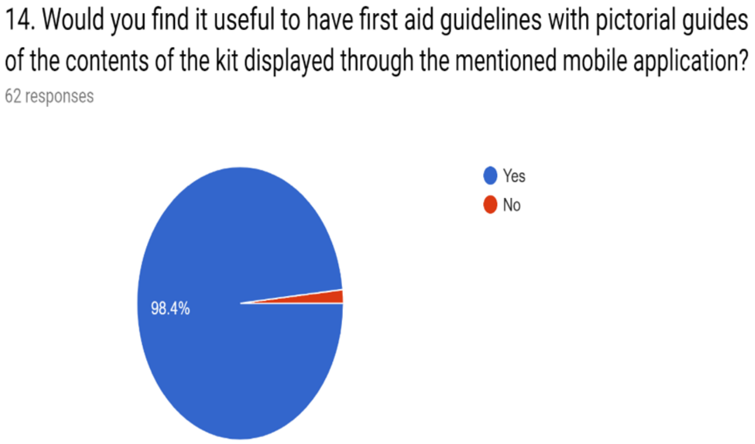
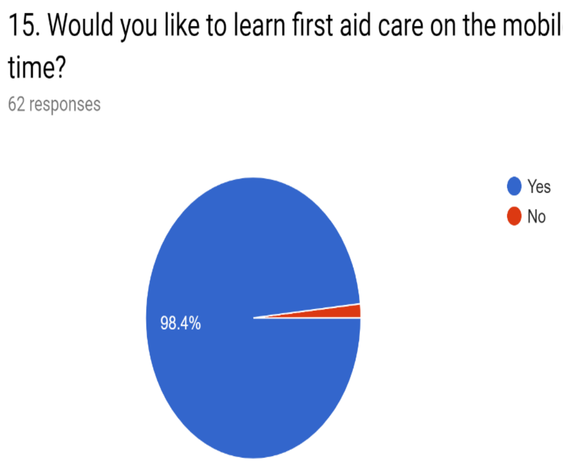


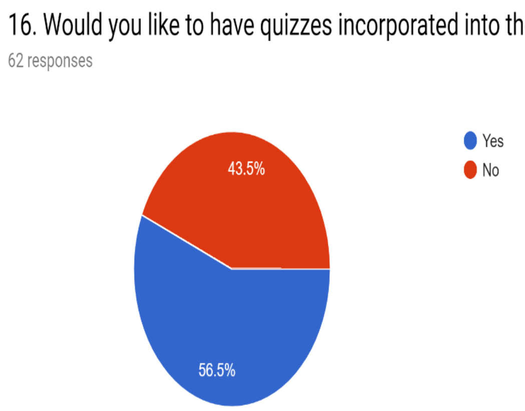
Figure 2.3: Importance of pictorial guides



15. Would you like to learn first aid care on the mobile application at any time?

62 responses

Figure 2.4: Respondents who want to learn first aid care on the application



16. Would you like to have quizzes incorporated into the mobile application

62 responses

Figure 2.5: Respondents who want to take quizzes on the application

The essential features of the system are:

- User registration
- Interaction with server for user authentication
- Request for First aid kit
- Transmission of location information
- Learn first aid
- Take quizzes to test knowledge
- View request locations
- Drone flight to location
- First aid kit and procedure display
- Sending drone back to home location i.e. ground control station
- User reports

2.2.4 User Categories and Characteristics

There are two main user categories of the system. These users can be classified as primary and secondary users.

The primary users in this case will be all people in the Ashesi community who have the possibility of getting injured or are injured. They will use the system to request for emergency first aid care from their phones to whatever current location they are in.

The secondary users can be the Health centre personnel, who will be responsible for assembling first aid kits. This class of users are also responsible for controlling the flight of the first aid kit equipped drone to the request location. They may also use the system for research purposes to analyse which injury cases occur frequently and the cause of such high occurrences.

2.3 Functional Requirements

The detailed functional requirements are discussed below:

2.3.1 User registration

The user will be notified to register or sign up

Stimulus/response sequences

Step 1: The mobile application is downloaded.

Step 2: When the application is launched for the first time, a registration form is displayed.

Step 3: The form sends information to the database.

User requirements

- The user must fill in the specified information.

System requirements

Validation: The system should be able to detect incorrect inputs and check if any of the primary key details already exists in the database and notify the user.

Input/output

- The input is a valid form entry.
- The output from this is a notification of a successful registration.

2.3.2 Interaction with server for user authentication

On successive launches of the application the user is asked to log in or sign in

Stimulus/response sequences

Step 1: The mobile application is launched.

Step 2: When the application is launched again, the user is asked to sign in via a form.

Step 3: The form sends the information entered to the database.

User requirements

- The user must fill in the required fields.

System requirements

Validation & Query: The system should be able to detect incorrect inputs and search if the inputted information matches any information in the database and notify the user

Input/output

- The input is a valid form entry.
- The output from this is a notification of a successful log in or sign in and a redirection to the home or landing page.

2.3.3 Request for First aid kit

On the home page a button with the caption request is displayed

Stimulus/response sequences

Step 1: A button is displayed on the home page

User requirements

- The user must click on the button.
- After the user clicks on the button, he or she is redirected to a page where a list of injuries is displayed for selection based on what is applicable.

System requirements

Connectivity: The system should be able to connect to mobile devices to be able to receive requests and notify the user.

Input/output

- The inputs are button clicks.

- The output from this is a notification of a request acceptance and a notification of a drone on the way and the display of the user's location on a map.

2.3.4 Transmission of location information

The GPS coordinates from the request phone location is sent to the database and can then be accessed by the Health centre staff.

Stimulus/response sequences

Step 1: After an injury is selected from the list of injuries during the request process, a function runs to get the GPS coordinates of the user.

User requirements

- The user must click on the request button and select an applicable injury.

System requirements

Connectivity: The system should be able to connect to mobile devices to be able to get GPS coordinates.

Input/output

- The input is a button click which then starts a function in the background that accesses the user's location and inputs it into a database.

- The output from this is a notification of a drone on the way.

2.3.5 Learn first aid

On the home page a button with the caption that indicates to the user that this helps to learn first aid is displayed.

Stimulus/response sequences

Step 1: A button is displayed on the home page

User requirements

- The user must click on the button.

System requirements

Connectivity: The system should be able to connect to mobile devices to be able to receive requests and notify the user.

Input/output

- The input is a button click.
- The output from this is a redirection to another page with a list of injuries to start learning the first aid care guidelines for the respective injuries.

2.3.6 Take quizzes to test knowledge

The home page also displays a button with the caption that asks if the user wants to take a quiz to test their knowledge.

Stimulus/response sequences

Step 1: A button is displayed on the home page

User requirements

- The user must click on the button.

System requirements

Connectivity: The system should be able to connect to mobile devices to be able to receive requests and notify the user.

Input/output

- The input is a button click.

- The output from this is a redirection to another page with questions that tests the user’s knowledge of first aid.

2.3.7 View request locations

On the home page view of the administrator (health centre personnel) side of the app, a button with the caption that asks the user if they want to view requests is displayed.

Stimulus/response sequences

Step 1: A button is displayed on the home page

User requirements

- The user must click on the button.

System requirements

Connectivity: The system should be able to connect to mobile devices to be able to receive requests and notify the user.

Input/output

- The input is a button click.

- The output from this is a redirection to another page that displays a map with markers showing where the requests made for a first aid kit equipped drone are.

2.3.8 Drone flight to location

The GPS coordinates is taken from the database and given to the drone to begin automatic flight to the request location.

Stimulus/response sequences

Step 1: The GPS coordinates are taken from the database.

Step 2: A check is made for a drone already in flight or not.

Step 3: The GPS coordinates a sent to an available drone or to one already on a delivery.

User requirements

- The admin user must have clicked on the start flight button.

System requirements

Connectivity: The system should be able to connect to mobile devices to be able to get GPS coordinates from the database and send it to drones.

Input/output

- The input is a button click which starts a function to get GPS coordinates from the database.

- The output is a drone starting flight.

2.3.9 First aid kit and procedure display

On the page a list of injuries is displayed

Stimulus/response sequences

Step 1: A list of common injuries is displayed on the page

Step 2: An injury is selected from the list

Step 3: A pictorial view of the first aid kit and guidelines is displayed on the page

User requirements

- The user selects an injury.

System requirements

Connectivity: The system should be able to connect to the server to be able to receive requests and display requested information.

Input/output

- The input is a button click.

- The output from this is a notification of a request acceptance and a notification of a drone on the way.

2.3.10 Sending drone back to home location i.e. the health centre

When user clicks a done button, the drone is sent back to the ground control station

Stimulus/response sequences

Step 1: A button is displayed on the home page

User requirements

- The user must click on the button.

System requirements

Connectivity: The system should be able to connect to mobile devices to be able to receive requests and notify the drone to return.

Input/output

- The input is a button click.

- The output from this is the start of the drone’s return flight to ground control station.

2.3.11 User reports

On the home page view of the administrator (health centre personnel) side of the app, a button which can be used to create reports.

Stimulus/response sequences

Step 1: A button is displayed on the home page

User requirements

- The user must click on the button.

System requirements

Connectivity: The system should be able to connect to the database

Input/output

- The input is a button click.

- The output from this is charts and graphs indicating the frequency of requests and various injuries.

2.4 Non - Functional Requirements

Performance: The system should be able to perform efficiently even when there are multiple requests and should be able to notify users when there are setbacks or problems with the system.

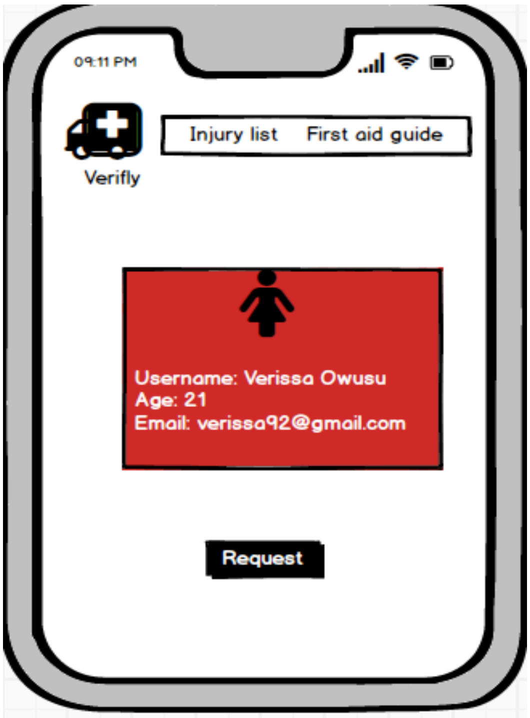
Availability: The system should always also be functioning to ensure 24 hours of access to first aid supplies when injuries occur at any time.

Security: The system should also be secure, to prevent data loss or theft which may cause harm to users. Security measures should also be put in place to prevent unauthorized access to user accounts.

2.5 External Interface Requirements

2.5.1 User Interfaces

The mobile application is designed to enable users' easy access to the Verifly system to request a first aid kit equipped drone. The client side of the application will have the request first aid functionality, first aid care guidelines, standard injury lists, and a finished first aid care functionality. The admin side of the application will show statistics of the app, that is the number of registered users, number of requests and any complaints. A mock-up of the user interface is shown below.



Username: Verissa Owusu
Age: 21
Email: verissa92@gmail.com

Figure 2.6: A mock-up of user interface

2.5.2 Hardware Interfaces

The drone will communicate with the ground control station which will, in turn, interact with the application. The transmission of data to the database is dependent on the bandwidth connection. The TCP/IP protocol is essential in the interaction between the users' device and the web server. The supported mobile phone devices are Android phones.

2.6 Drone Policy

Privacy is a huge concern for people especially when it comes to drones. Audio-visual equipped drones are feared as people believe that they may be used to capture them in compromising situations [14]. As such a drone policy was written to cater to some of the concerns of members of the community. This drone policy gives guidelines concerning the process of seeking permission to operate a drone on campus for research or recreational purposes and other things. The Appendix section of this report contains the proposed drone policy.

Chapter 3: Design and Architecture

3.1 Introduction

The chapter depicts models of the high-level architecture of the first aid drone system. Activity diagrams show the process flow from one activity to the next. The activity diagram in Figure 3.1 below shows the flow of events for a primary user, after registration on first use, or a successful login, a primary user has specific actions available. Such a user can request first aid, learn emergency care or take a quiz to test their knowledge in first aid. When the primary user is injured and wants to ask for first aid, the user first selects an injury from a list and then gets a drone equipped first aid kit sent to his or her location. Once the request for the kit has reached the health centre, the first aid care guidelines for treating the injury is displayed on the application for the user. When the drone gets to the user's location, he or she then removes the kit from beneath the drone and follows the procedure displayed on the mobile application. After treatment has been applied and the first aid kit is reattached to the drone, the user clicks on a finish button to notify the health centre to start a return flight process for the drone. In the case where the user wants to learn first aid for educational purposes, they are prompted to click on a button and then a list of injuries, and the appropriate care for each is displayed. When the user is done, they are returned to the home page of the application. When a user wants to test their knowledge as well, he or she can take a quiz on the app and get their results shown to them as well as the correct answers to questions.

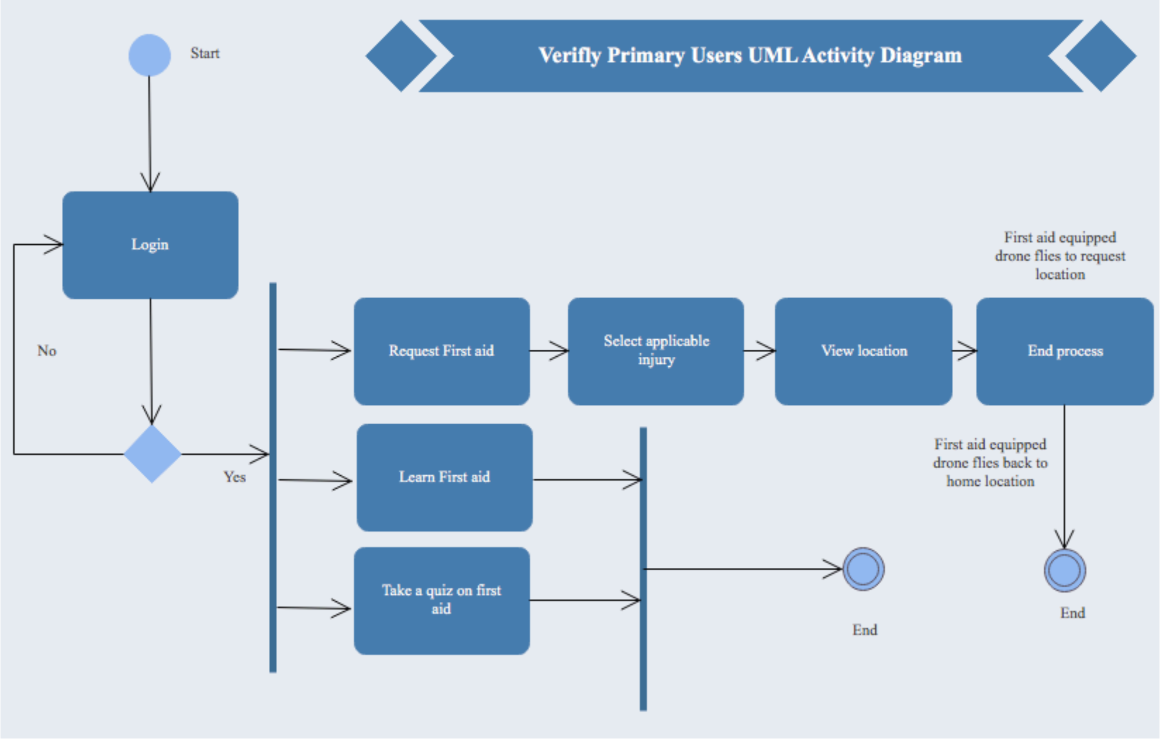

Verifly Primary Users UML Activity Diagram
Start
Login
No
Yes
Request First aid
Select applicable injury
View location
End process
First aid equipped drone flies to request location
Learn First aid
Take a quiz on first aid
First aid equipped drone flies back to home location
End
End

Figure 3.1: Activity diagram showing flow of activities for primary users

Figure 3.2 below shows the flow of activities for the Health centre admin staff. When a member of staff logs into the application, they are provided with options to view requests and create reports. When the user chooses to view requests, a map is displayed showing all the requests for first aid kits that have been sent by any injured user. Then there are buttons displayed to send the first aid- equipped drone to the request location and then a button to return the drone to the Health centre when the user is done using the first aid kit. When staff wants to create reports from the system, they can click on the generate reports button to create graphical representations of the frequency of requests, the occurrence of injury cases and many more.

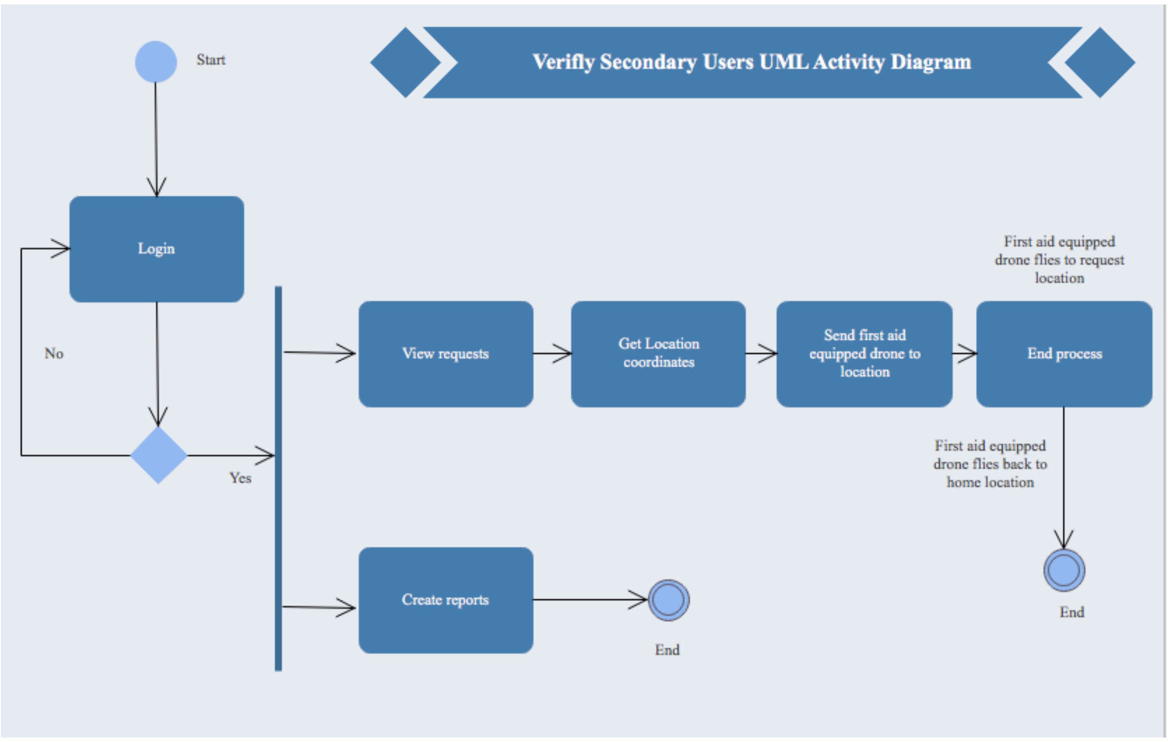

Verifly Secondary Users UML Activity Diagram
Start
Login
No
Yes
View requests
Get Location coordinates
Send first aid equipped drone to location
End process
First aid equipped drone flies to request location
First aid equipped drone flies back to home location
End
Create reports
End

Figure 3.2: Activity diagram showing flow of activities for secondary users

3.2 System Architecture

The system uses the three-tier client-server architecture where the clients, i.e., user classes interact with the system through a mobile application to carry out functions like requesting a first aid kit through the app. The mobile application communicates with the server. The server is on a remote computer; the server helps to carry out the functions. For example, to request a drone the user's location is sent to the server and then sent to the ground control station. All the users interact with the same server to carry out requests they have made to the system.

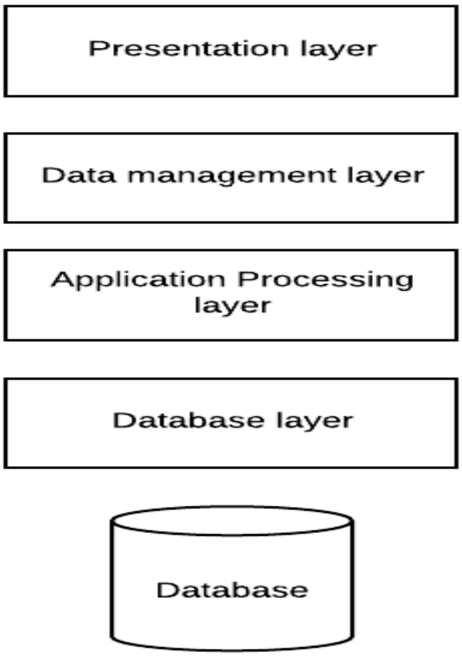


Figure 3.3: Layered architecture for the system

Another architectural pattern that was implemented in the system was the Model-View-Controller pattern. The MVC pattern separates the system into three components that interact with each other. The View component in this system is responsible for presenting the home activities, request activities, quiz activities, etc. to the primary user and the view requests activities and the dashboard to the secondary user. The Controller is responsible for handling user interaction with the views. The Model takes the interactions passed from the controller and checks if it aligns with the business logic of the system; thus, the structure of data and then gets data from the database.

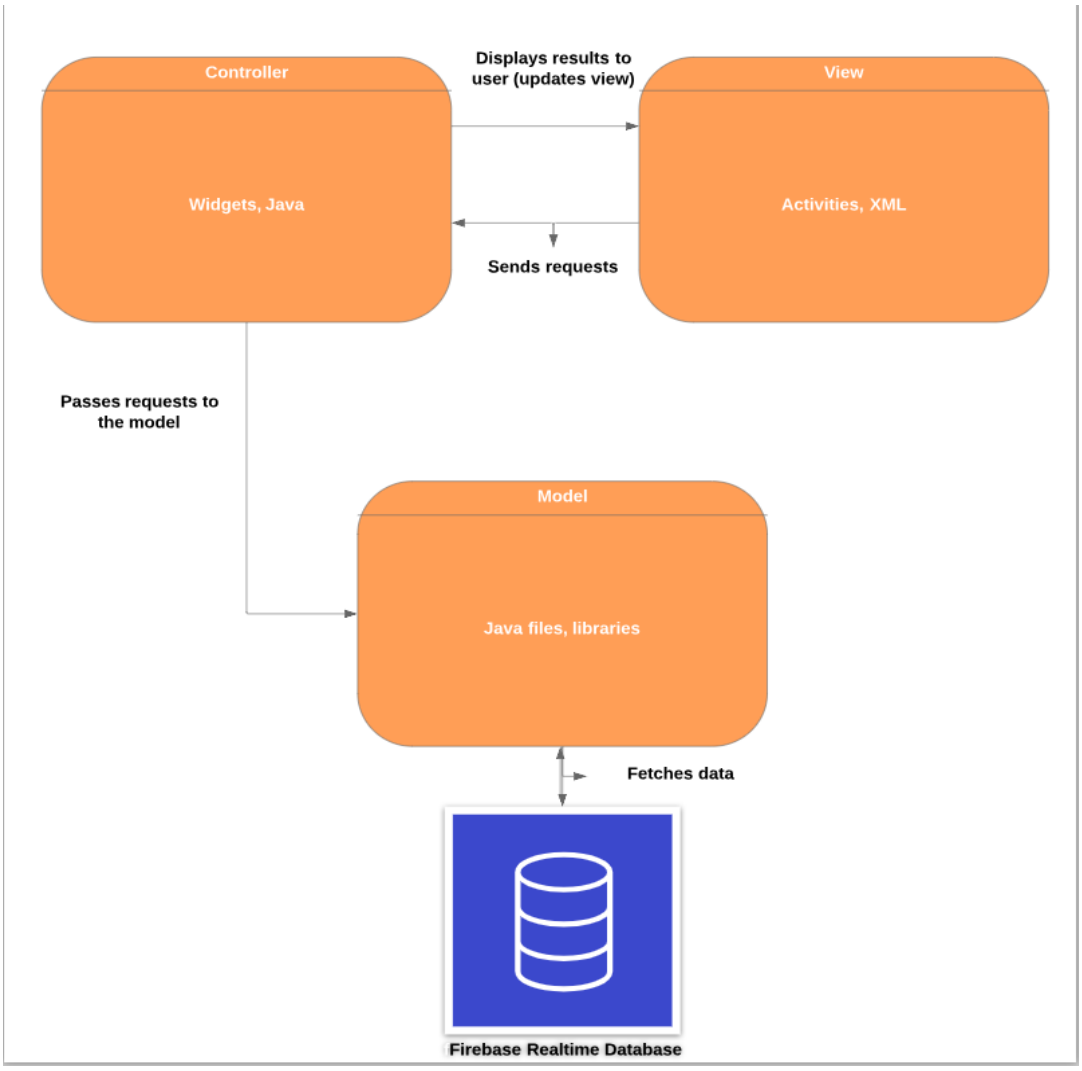


Figure 3.4: MVC architecture of the system

3.3 Design Specification

3.3.1 Use case view

The use case diagram in Figure 3.5 below gives an overview of the parties involved in the first aid system, different users have different actions on the system they make use of and how they interact with the other functions. The use case diagram shows other activities that are included and extended from another action. The log in activity of the primary users includes a verify details actions and extends to a display login error action.

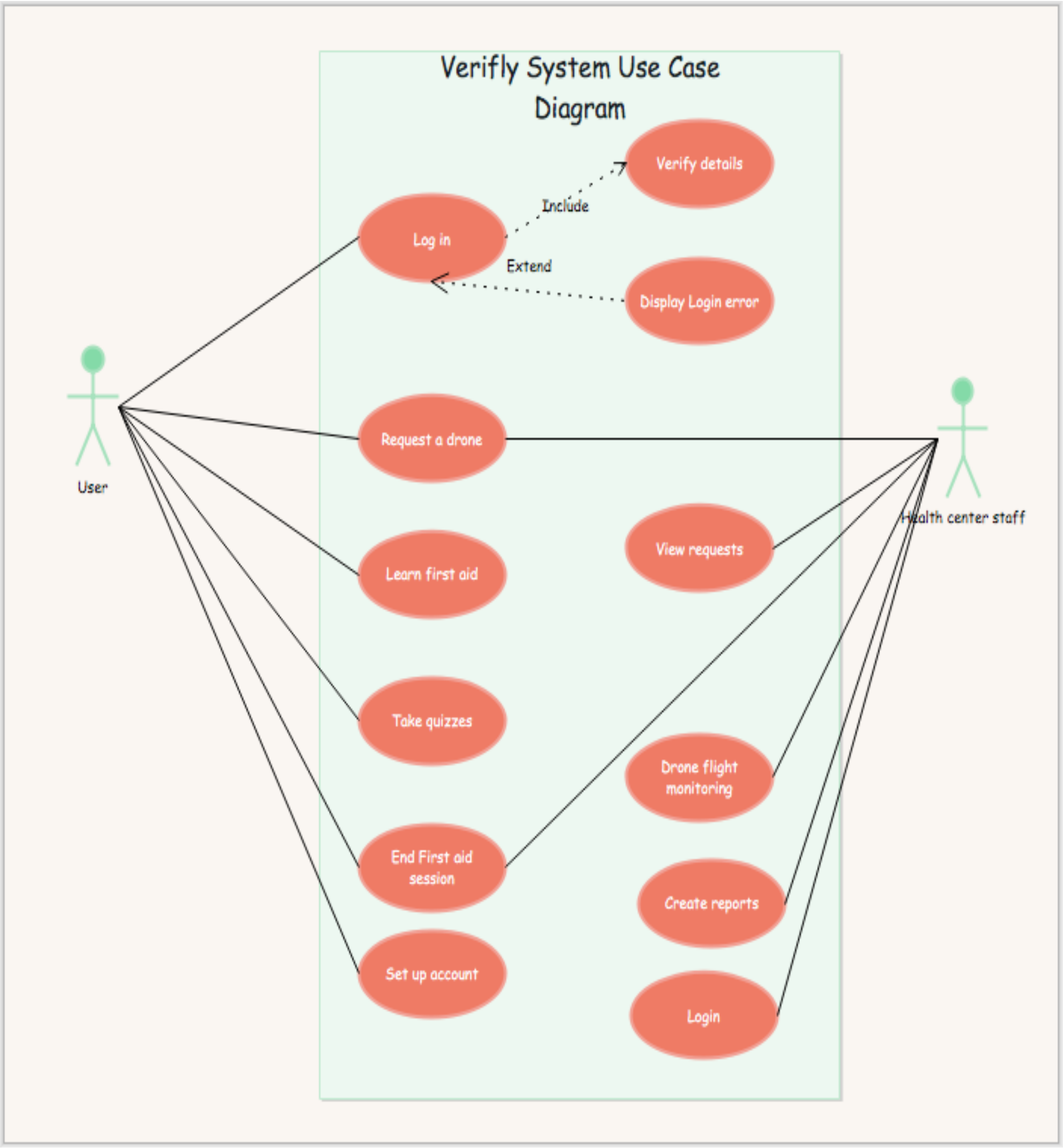


Figure 3.5: Use case diagram for the system

3.3.2 Database architecture

The system will need a database that will contain tables regarding user details, location details, admin (Health centre personnel) credentials. The user table will have details about users which will have data inserted into it mainly when users register; and will also be used for authentication purposes when a user logs in to the application.

The drone table will contain details about the drone, such as the drone id, capacity and other information to help track the number of drones available as well as functioning correctly.

The location details should contain information about the users and their location. Information will be sent to the table when the user activates the function needed to do so.

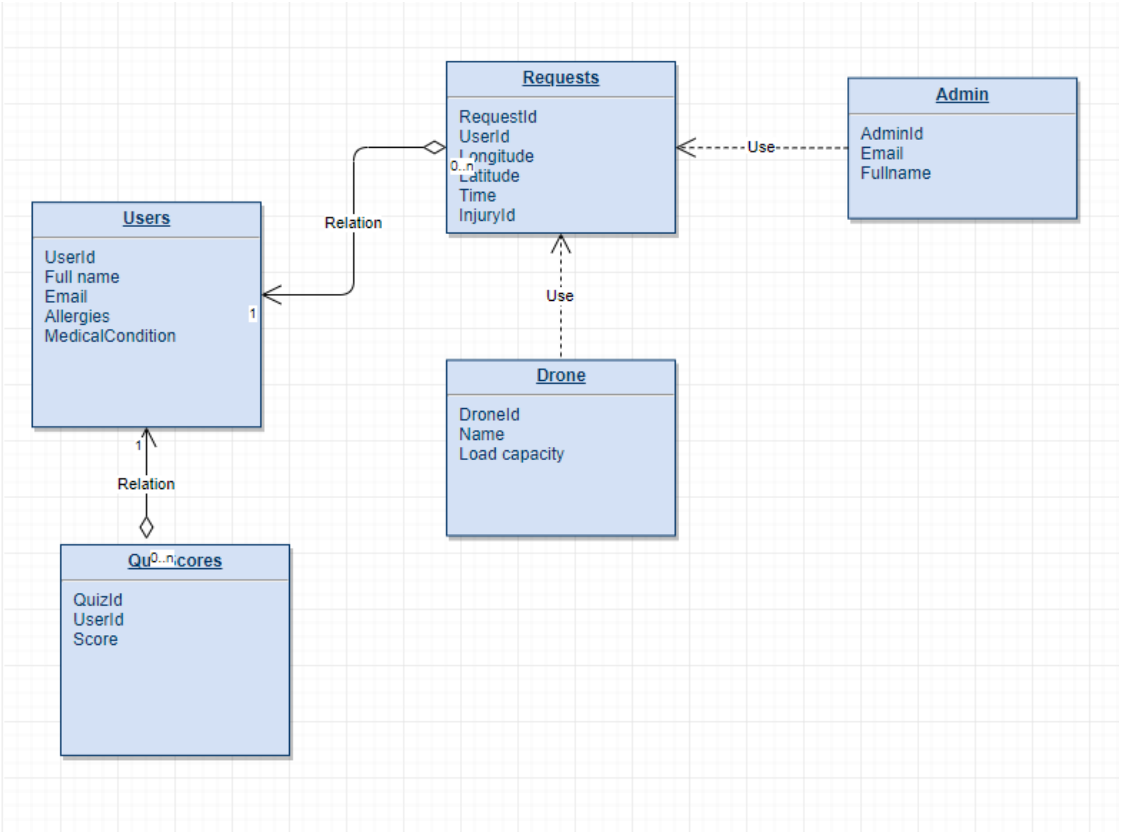


Figure 3.6: Database architecture diagram

3.3.3 Logical view

The logical view of the system describes the entire functionality. Sequence diagrams below depicts models of some interactions between user classes and the system. The first sequence diagram models the interaction from the registration to the request of a first aid drone on the primary user's side and the second sequence diagram models the interaction from the login to the start of flight on the secondary user's side.

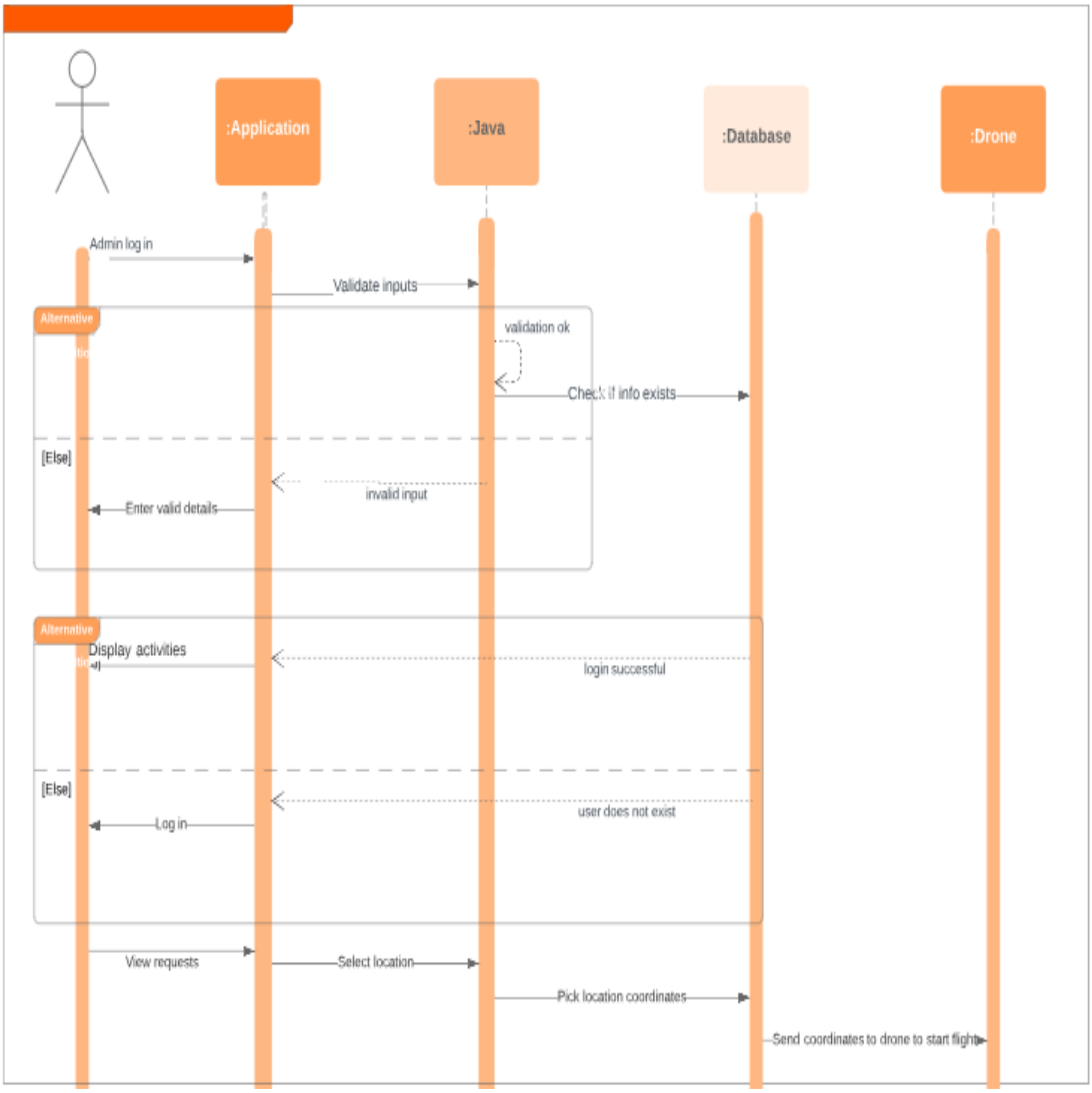


Figure 3.7: Sequence diagram for registration and request first aid activities

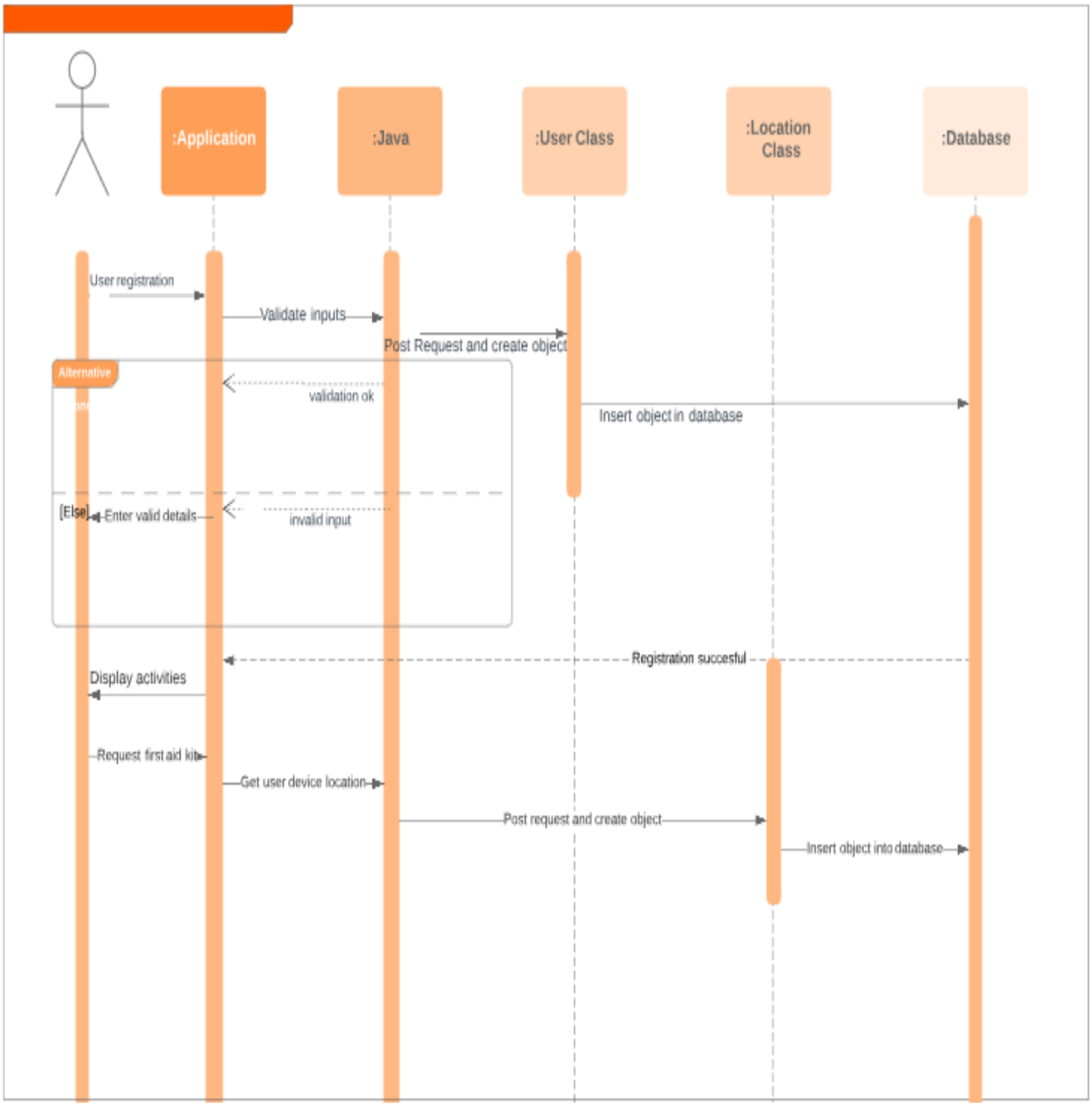


Figure 3.8: Sequence diagram for login and flight control activities

Summary

This chapter depicted the system's design and architecture specifications. It also contains illustrations of the use case diagrams, logical view, activity diagrams and database designs.

Chapter 4: Implementation

4.1 Introduction

The Verifly system comprises of Android mobile applications for primary users and secondary users. Primary users are categorized as any member of the Ashesi University committee and the Ashesi University Natembea Health Centre personnel as secondary users. Prime users can request for a first aid kit through the application and get the first aid guidelines displayed through the app. Primary users also have the option to learn first aid as well as take quizzes to test their knowledge. Secondary users have functionalities such as viewing the location from which requests are sent, create reports to find out the currency of injuries and automate the first aid kit equipped drone's flight to the user.

The DJI Phantom 4 is the drone used in this system. However, the drone was not designed to deliver first aid kits. As such a first aid kit must be strapped to it before the flight by a secondary user as depicted in Figure 4.1.

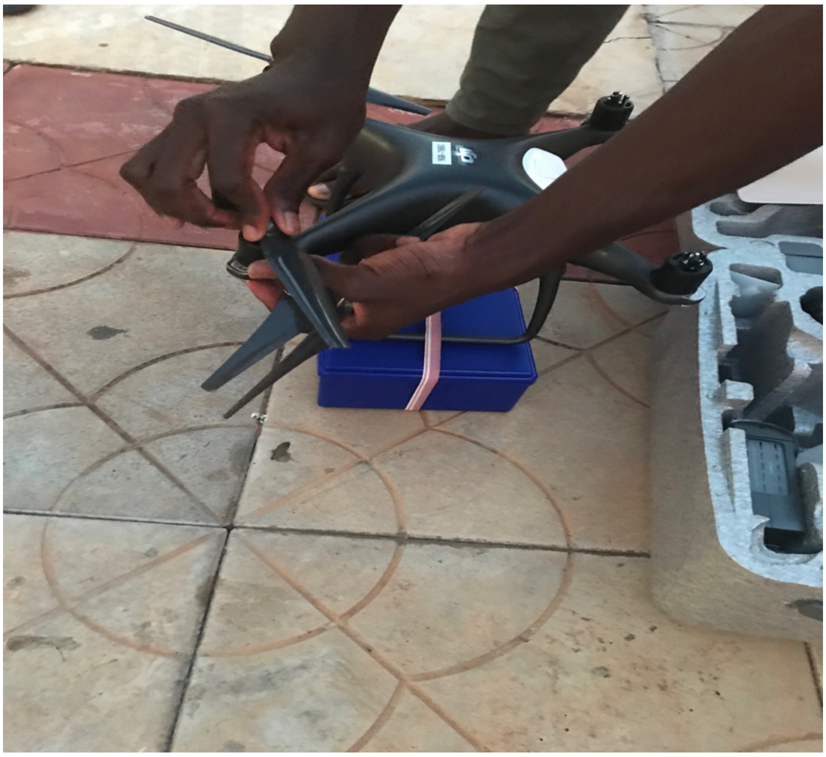


Figure 4.1: A picture showing the DJI Phantom 4 with a first aid kit attached

4.2 Key technologies

A mobile application that requires registration upon the first launch and then the user's details are entered into a database. The app will then display necessary user information and options to request for first aid, which sends the GPS coordinates of the person seeking to the ground control station- at the Natembea Health Centre, for emergency care assistance. The drone will have a first aid kit which will contain medical supplies. On arrival, a list of injuries options as well as pictorial guidelines is displayed on the app to help the patient or a bystander administer proper first aid care. The primary user can also learn first aid steps and take a quiz. Secondary users can use the second mobile application to view requests and then initiate the first aid kit equipped drone to the wounded person, and also create reports. A mobile application is an interface for users to interact with the system. One of the reasons for using a mobile app rather than a website is because the DJI Phantom 4 drone does not have a Web SDK to enable the implementation of automated flight control.

4.2.1 Mobile Application

The Android Operating System is open source and free. The developers use the Application Framework for Android as depicted in Figure 4.2. The Java programming language is used in creating Android Applications. Android SDK tools help to compile the data and the resource files and other images into one single archive package with a '.apk.' extension. This apk file can be used to install the application with just a click on Android Devices [6].

The mobile applications in this project will be developed using Android Studio. The Android Studio IDE enables user interfaces to be designed using the drag and drop method as well as programmatically using XML. Functions and classes are programmed in Java. Dependencies like the Google services will be implemented to enable the use of Google maps and connection to the Firebase database to facilitate authentication and read and write operations. The implementation of the DJI mobile SDK for Android is done using the android studio application.

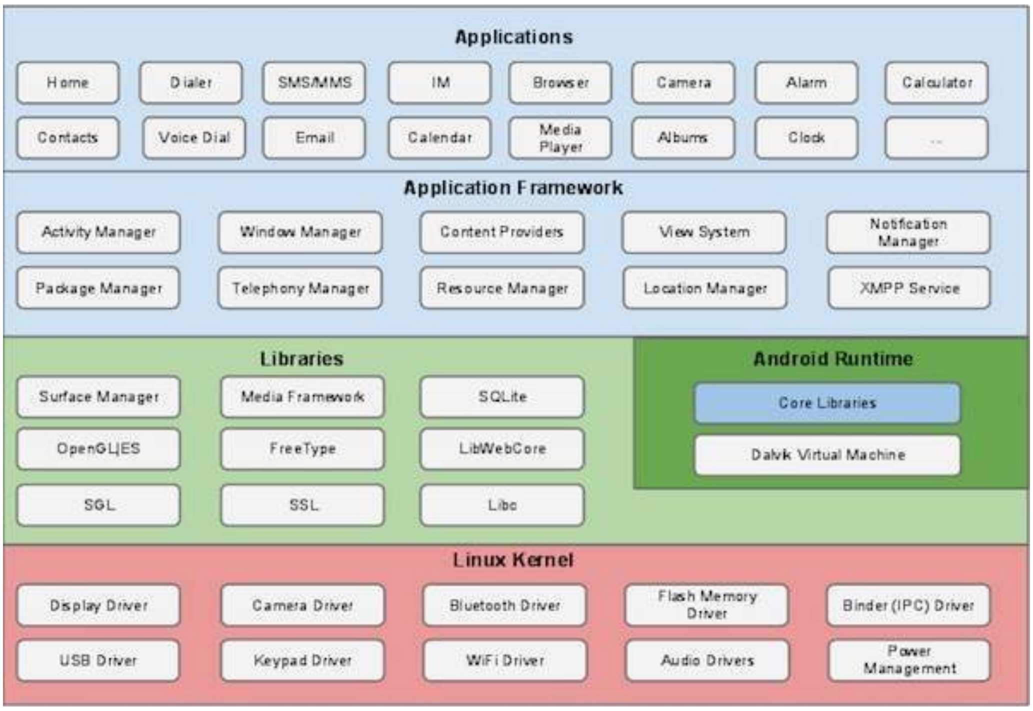


Figure 4.2: Architecture of the Android Operating System

4.2.2 Global Position System (GPS) Technology

The global position system (GPS) is a global navigation satellite system that comprises between 24 and 32 medium earth orbit satellites to form a constellation that surrounds the Earth planet. A GPS receiver needs at least three satellite signals to compute its two-dimensional position (latitude and longitude) or at least four satellites signals to compute its three-dimensional location (latitude, longitude, and altitude) [2]. The GPS signal will be sent from the phone application to a server, using GPS as a navigation system, the GCS (Ground Control Station) will read the message from the server, and fly the drone on the defined path until they reach the destination.

4.2.3 Ground Control Station

The ground control station is responsible for the coordination and control of all the drones. Therefore, a microcomputer is required to process all the data and manage all the communications. The ground control station will be responsible for handling drone queues and communicating patients' locations from the applications with the drones. The ground control station is the Natembea Health centre.

4.2.4 Google Maps API

To be able to display the user's location on the map as well as display request locations, it is essential to install the Google Play services SDK in Android studio. This feature requires an API key. The fast way to get one is through the Android studio application. This process first requires that a new Maps activity is created in the project files. Android studio creates a google_maps_api.xml file for you. Open the document, and in the comment section, you will see a link, copy this link and paste it in your browser. Follow the steps to create your API key. Copy the API key and paste it in the <string> element in the google_maps_api.xml file.

4.2.5 Google Firebase

Firebase is a platform that supports both mobile and web app development. Firebase is essential to developers as it provides a plethora of tools and services to help develop high-quality apps, grow user base and earn more profit [7]. Firebase also has cloud storage features and a NoSQL database feature, that serves as a bonus for protecting against SQL injections.

4.2.6 DJI Mobile SDK for Android

The DJI Mobile SDK is a software development kit designed to give developers access to the capability of DJI's aircraft and other products. The SDK simplifies the application development process by taking care of lower level functionality such as flight stabilization, signal transmission, and communication. This way, developers do not necessarily require a background in robotics or embedded systems to make use of drone capabilities and can instead focus on the application the DJI product is integrated into [8]. Figure 4.3 shows the DJI Mobile SDK architecture and Figure 4.4 shows some classes of the DJI Mission component.

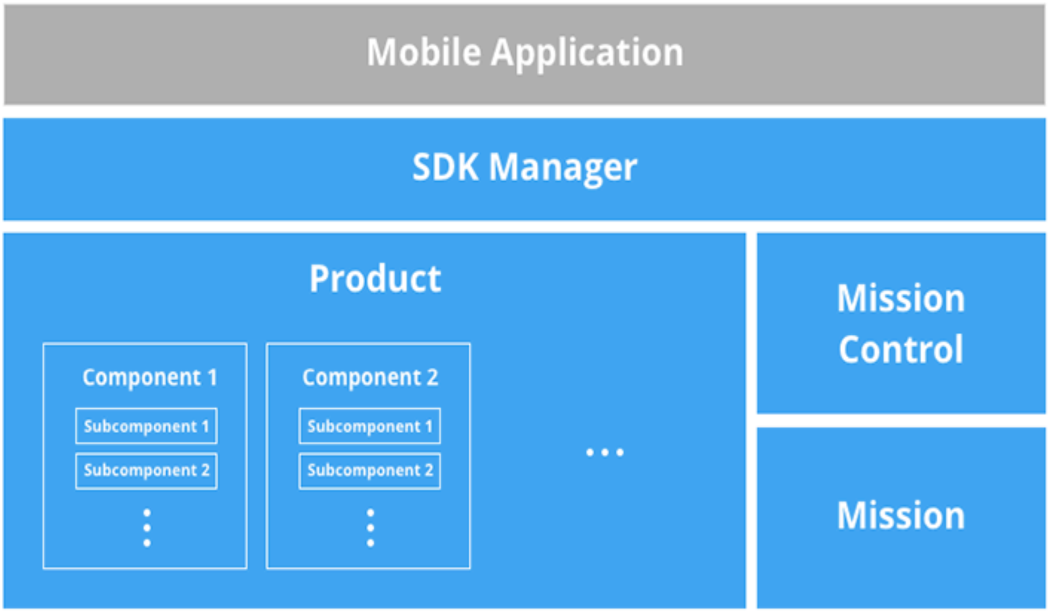


Figure 4.3: DJI Mobile SDK Architecture

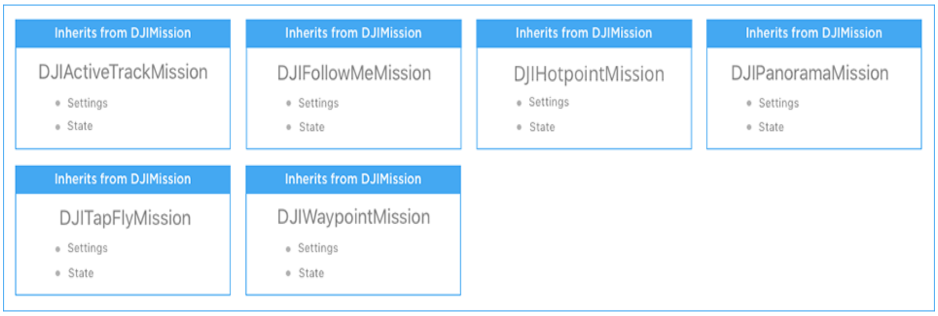


Figure 4.4: DJIMission components

4.2.7 Drone

A drone for civilian use is a propulsion aircraft, unmanned, reusable, operated by remote control or/and autonomously (autopilot). The drone is an essential component of this solution as it presents an alternative means of deploying emergency care. For this system, a DJI Phantom 4 drone is a suitable drone with a payload capacity of 1.02lbs, and a first aid kit is about 0.85kg and has auto take-off and auto return home features with GPS technology capabilities which makes controlling easy. It has an application which enables monitoring or camera operation and makes it more accessible. It also has a flight range of 3.1 miles, which is roughly 5 kilometres with a speed of 40 mph. The drone also comes equipped with a unique feature that allows it to avoid obstacles which might be interrupting the device in its flight.

4.3 Android mobile application

The mobile application was built on the Android operating system. This decision is because android is open source, has a more significant developer and community reach and reduced cost of development and higher success ratio. The Android mobile application was built using the Android Studio software. Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA [9]. Android Studio also enables easier integration with Google services. Thus, making it the optimal software for the mobile application development for easy integration with Firebase and Google Maps services implemented in the application.

4.3.1 Android Application for Primary Users

4.3.1.1 Getting user's location

When an injured user wants to get first aid care a button on the home page is clicked to start the request process. A screenshot of the home page is depicted in Figure 4.5. Once the user clicks the button, the device location is fetched using the Fused Location Provider which is used to retrieve the device's last known location. The fused location provider is one of the location API services available in Google Play services. The fused location provider manages the underlying location technology and provides a simple API so that you can specify requirements at a high level. It also optimizes the device's use of battery power [10]. The getLastLocation() method is used to access the device location. This method returns the best most recent position currently available.

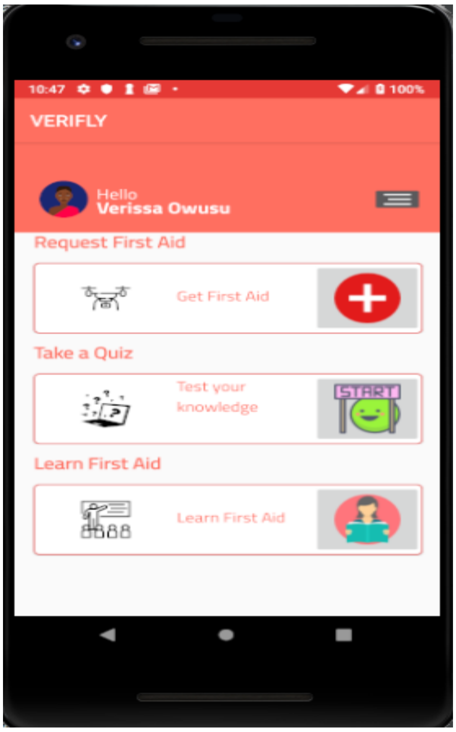


Figure 4.5: Home page (primary users)

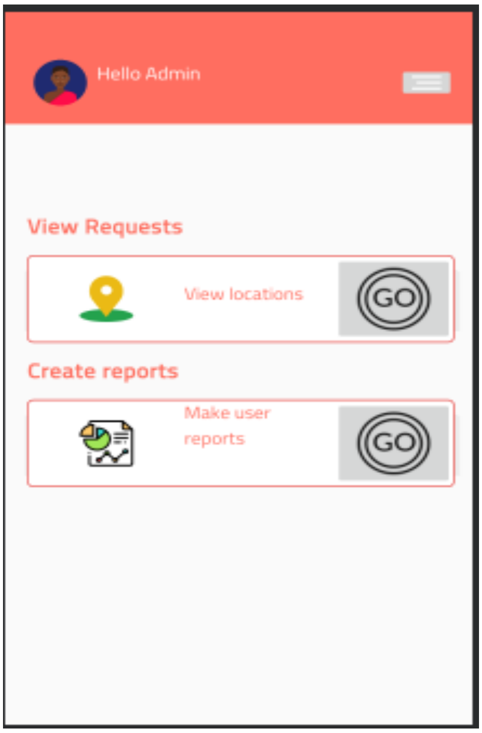


Figure 4.6: Home page (secondary users)

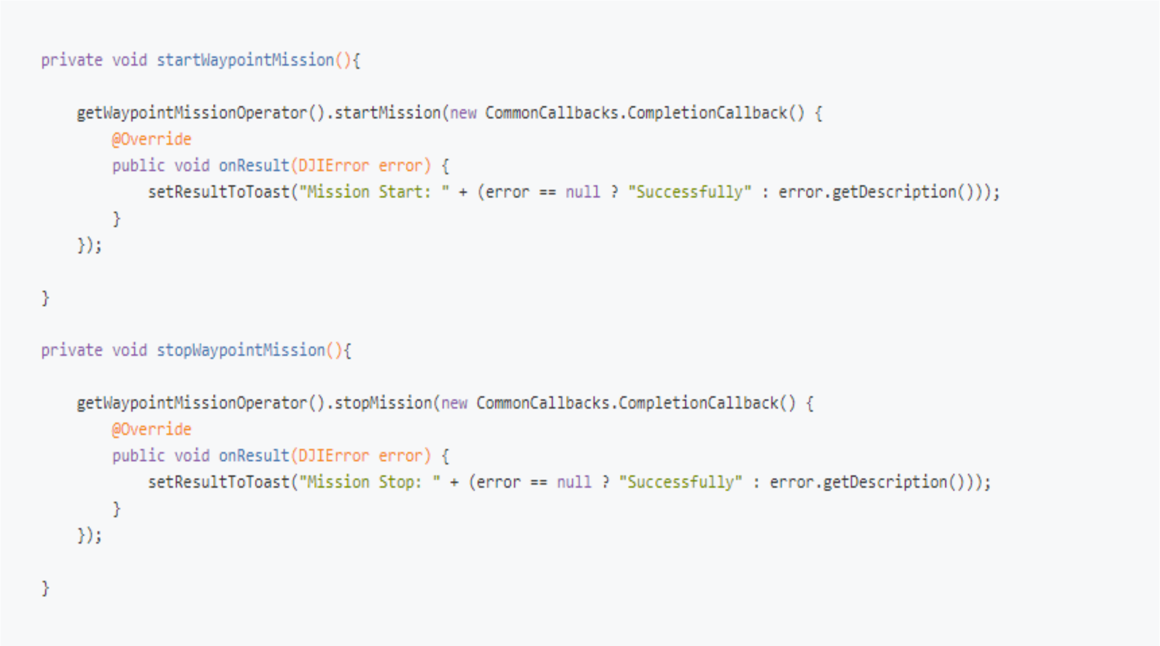
4.3.2 Android Application for Secondary Users

4.3.2.1 View request location

Health centre staff can view the requests made for a first aid kit and view these requests on the map. The requests, when made by injured users, is entered into the database. Thus, when health centre personnel choose to view request locations, they are retrieved from the Real-time database and then displayed on the map using the Google maps API. This visual representation enables quick decisions to be made and view the closest and furthest request locations.

4.3.2.2 Set waypoints and start flight

A waypoint mission is a series of pre-defined places (waypoints) the aircraft will fly. Waypoint mission can be used to automate the flight control of the DJI Phantom 4 drone. The health centre staff can select request locations they would like to send the first aid equipped drone. The location coordinates are then passed into the Waypoint Mission component of the DJI Mobile SDK. The health centre personnel can then click on a button to start the mission. The start of the mission is implemented by using the startMission() methods and ended using the stopMission() method. Figure 4.7 shows the implementation of these methods to enable the automated flight control process [11].



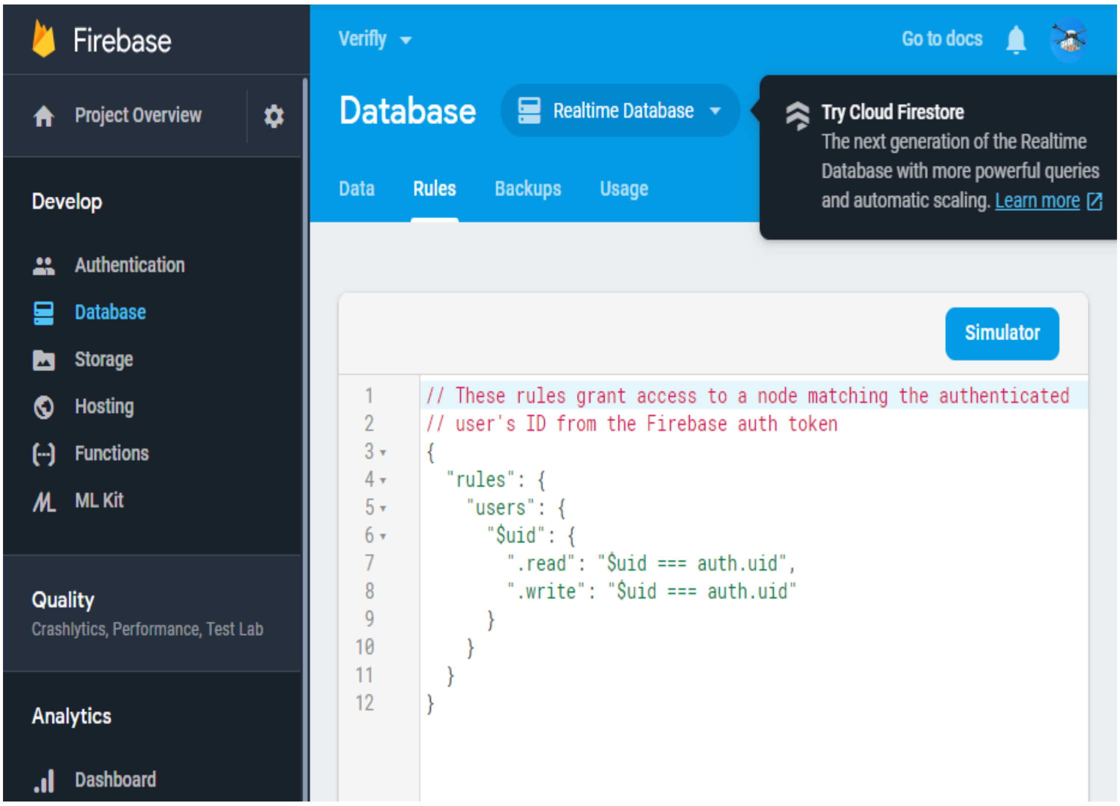
```
private void startWaypointMission(){

    getWaypointMissionOperator().startMission(new CommonCallbacks.CompletionCallback() {
        @Override
        public void onResult(DJIError error) {
            setResultToToast("Mission Start: " + (error == null ? "Successfully" : error.getDescription()));
        }
    });

}

private void stopWaypointMission(){

    getWaypointMissionOperator().stopMission(new CommonCallbacks.CompletionCallback() {
        @Override
        public void onResult(DJIError error) {
            setResultToToast("Mission Stop: " + (error == null ? "Successfully" : error.getDescription()));
        }
    });

}
```

Figure 4.7: An implementation of the Waypoint Mission component of the SDK

4.3.2.3 Create Reports

This feature allows the Health centre personnel to draw statistical analysis on some data received from the primary users. The android Graph view plotting library is implemented in this feature to create pie charts and line graphs to generate reports containing information such as the occurrence of injuries and the locations of requests. The Graph View library is an open source graph plotting library used to create flexible and beautiful diagrams [12] programmatically. Information from these reports can inform decisions for health talks organized by the department and other choices.

4.3.3 Google Firebase Realtime Database



Try Cloud Firestore
The next generation of the Realtime Database with more powerful queries and automatic scaling. Learn more

```
// These rules grant access to a node matching the authenticated
// user's ID from the Firebase auth token
{
  "rules": {
    "users": {
      "$uid": {
        ".read": "$uid === auth.uid",
        ".write": "$uid === auth.uid"
      }
    }
  }
}
```

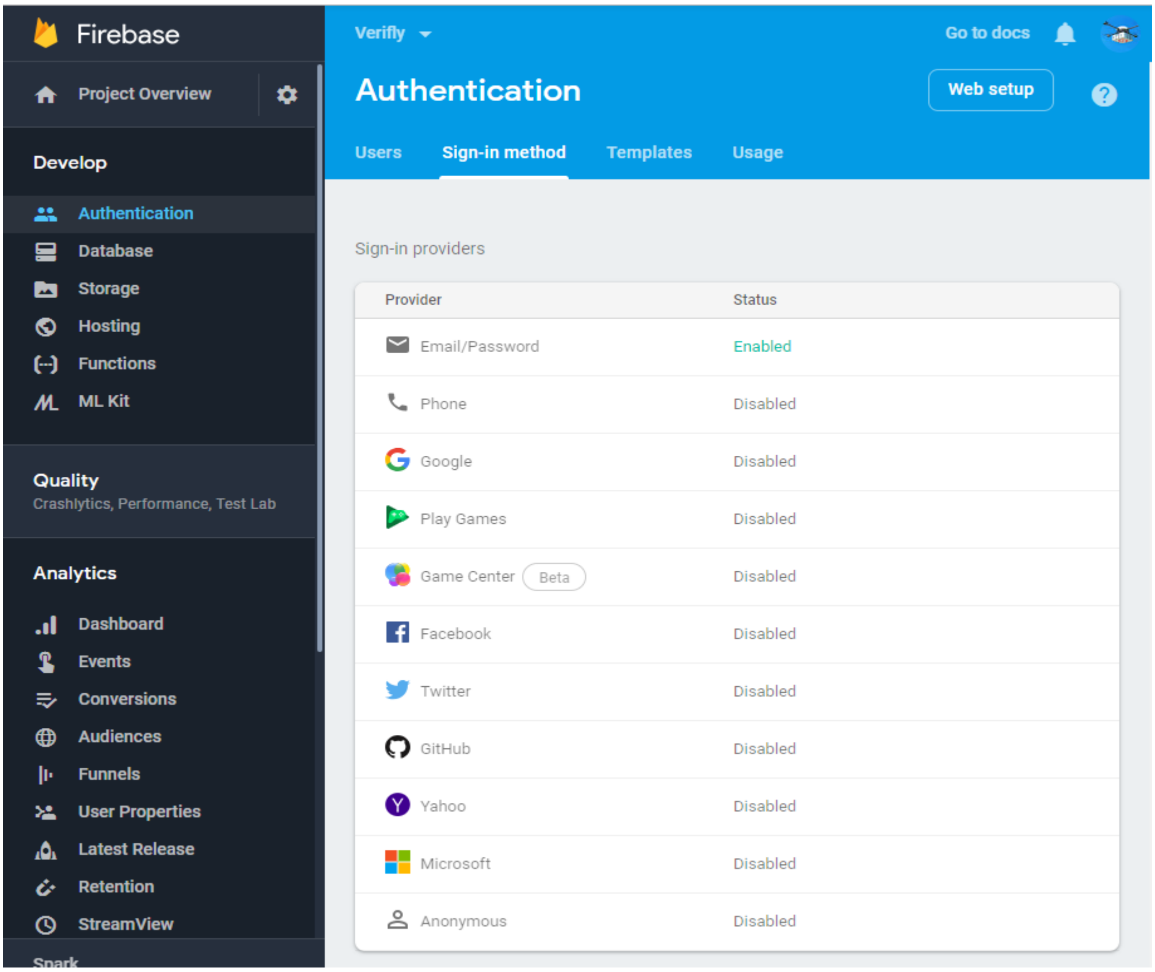
Figure 4.8: Database rules

The Real-time Database is a NoSQL database. Instead of common HTTP requests, the Firebase Real-time Database uses data synchronization—every time data changes, any connected device receives that update within milliseconds. Provide collaborative and immersive experiences without thinking about the networking code. The Firebase Real-time Database can be accessed directly without the need for an application server from a mobile device or web browser. Firebase Real-time Database Security Rules are expression-based rules executed when data is read or written to ensure security and data validation. Figure 4.8

shows that the database rules have been set to allow only authenticated users to read and write data to the Firebase Real-time Database and Cloud Storage.

4.3.4 Authentication

Firebase Authentication is a product of Google Firebase that contributes to ensuring security on the mobile application. Firebase authentication ensures secure sign in for users. Google, Facebook, and Twitter are some sign-in providers that enable authentication. Figure 4.9 shows that the email and password sign-in provider is enabled and implemented for this project. Firebase authentication also provides settings to restrict multiple account creations; thus one account per email address. Firebase also allows email templates, an example shown in Figure 4.10, created for sending notifications for password resets and email verification.

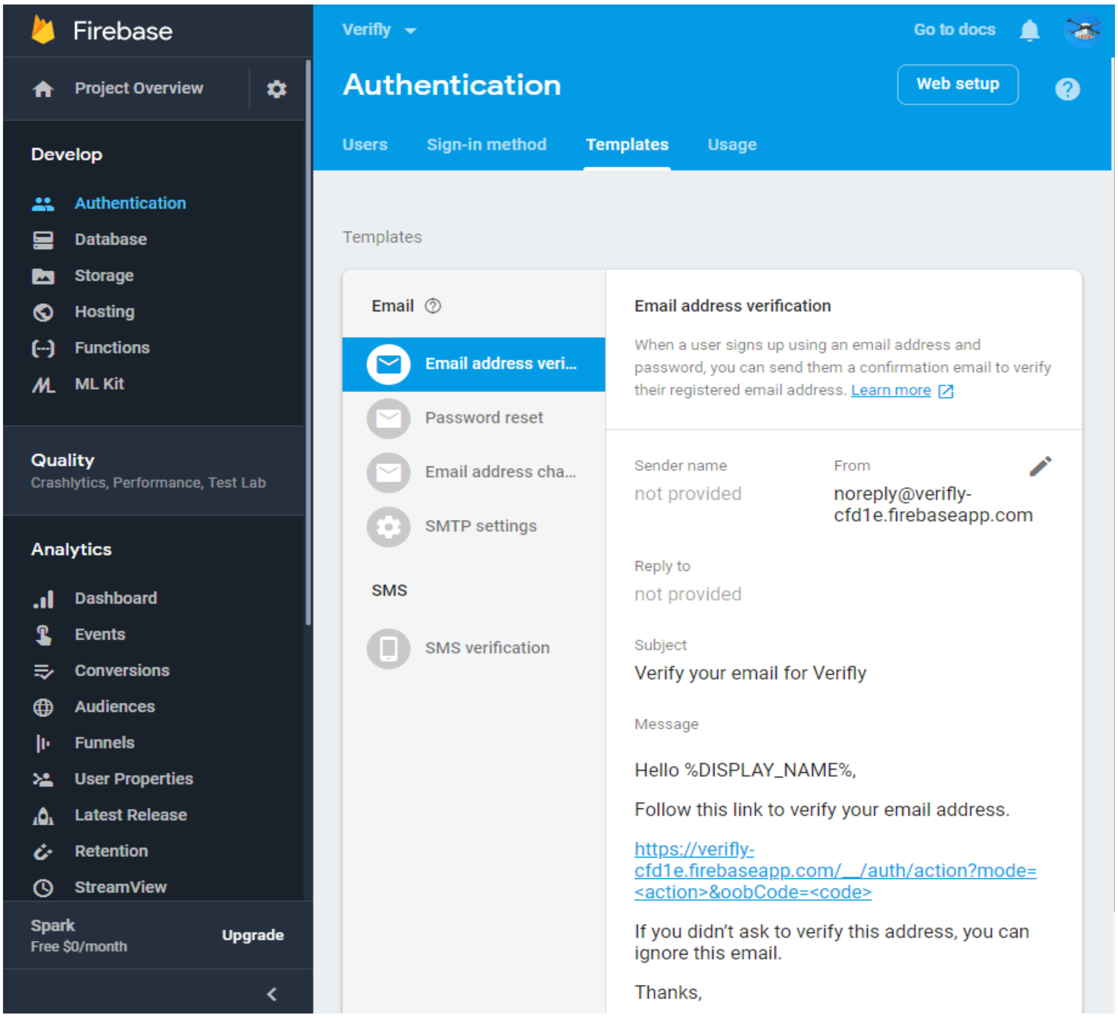


Authentication

Sign-in providers

Provider	Status
Email/Password	Enabled
Phone	Disabled
Google	Disabled
Play Games	Disabled
Game Center Beta	Disabled
Facebook	Disabled
Twitter	Disabled
GitHub	Disabled
Yahoo	Disabled
Microsoft	Disabled
Anonymous	Disabled

Figure 4.9: Screenshot of authentication providers



Email address verification

When a user signs up using an email address and password, you can send them a confirmation email to verify their registered email address. Learn more

Sender name
not provided

From
noreply@verifly-cfd1e.firebaseapp.com

Reply to
not provided

Subject
Verify your email for Verifly

Message

Hello %DISPLAY_NAME%,

Follow this link to verify your email address.

https://verifly-cfd1e.firebaseapp.com/__/auth/action?mode=<action>&oobCode=<code>

If you didn't ask to verify this address, you can ignore this email.

Thanks,

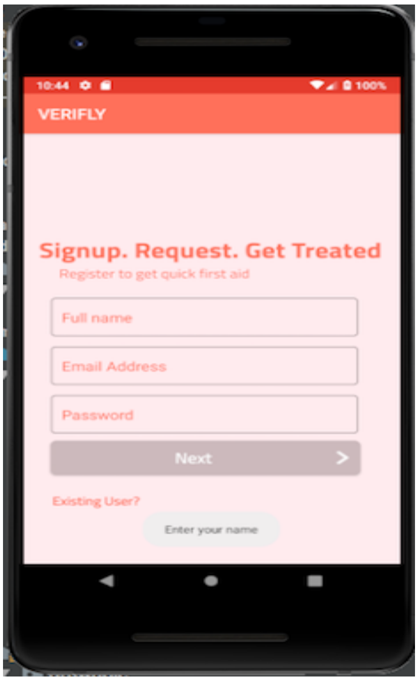
Figure 4.10: Screenshot of e-mail template

Chapter 5: Testing and Results

To test the system and its level of satisfying the functional requirements of the system. It was decided that the test should be done according to the key features that were identified. Thus, the system was tested in parts according to the functional requirements.

5.1 User registration testing

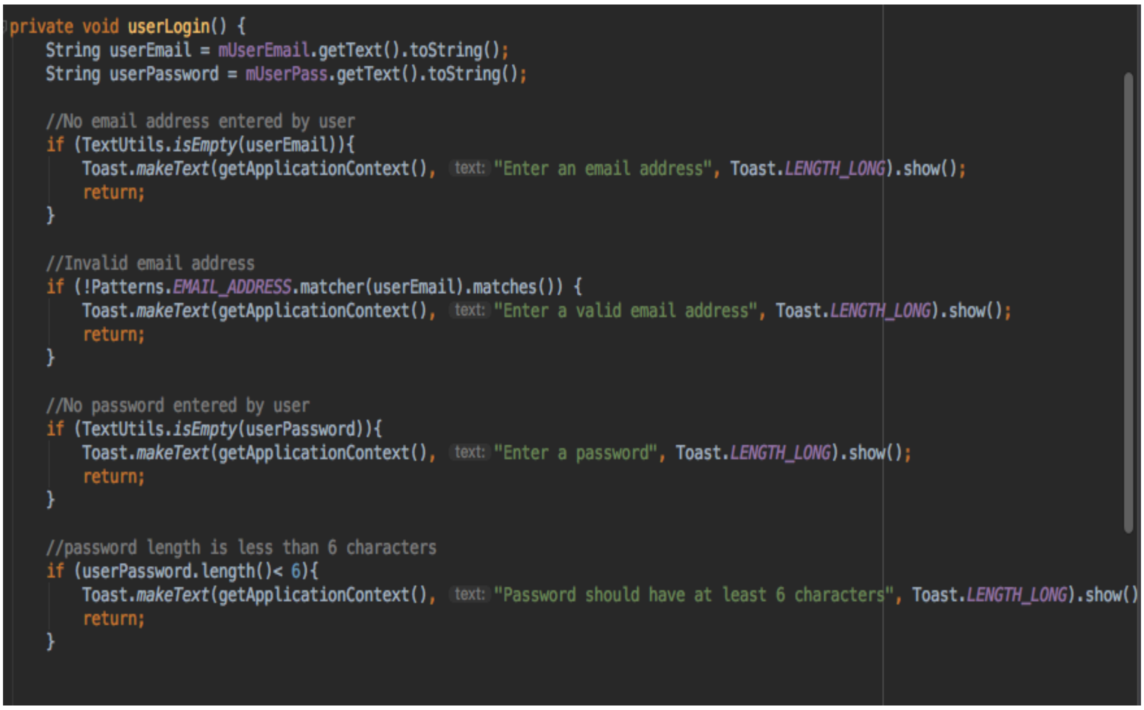
Valid inputs	**Result**
All fields are filled with valid input	Notification of successful registration and redirection to home page
Invalid input	**Result**
No input for any field	Alert of invalid input and redirection to registration page as displayed in Figure 5.1


Signup. Request. Get Treated

Register to get quick first aid

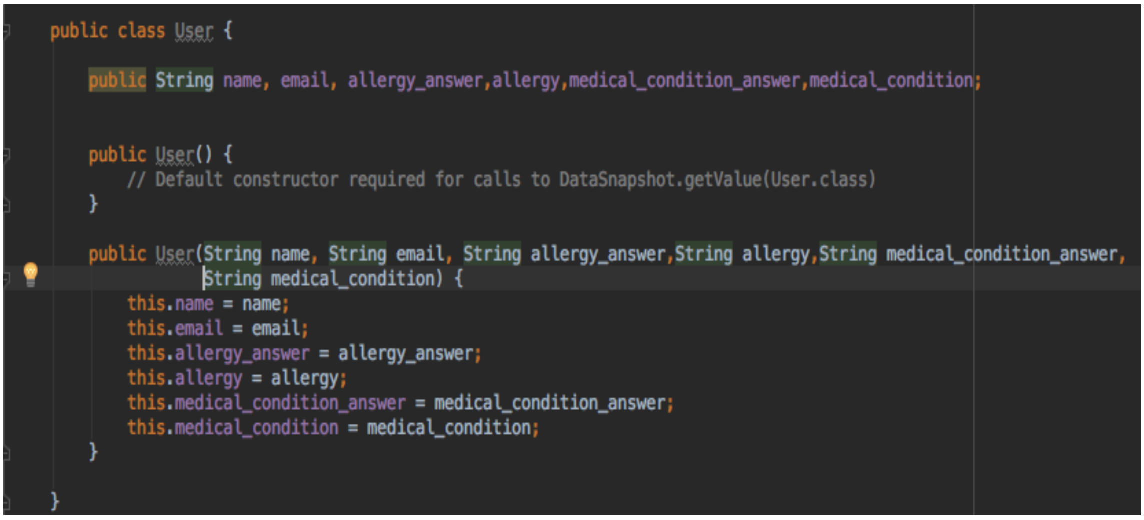
Figure 5.1: Alert on invalid input

The functions in Figure 5.2 is run to check the validity of the inputs a user enters into each field. The User class is also shown in Figure 5.3 below. The entry into the database and authentication is depicted in Figure 5.4 and 5.5 below.



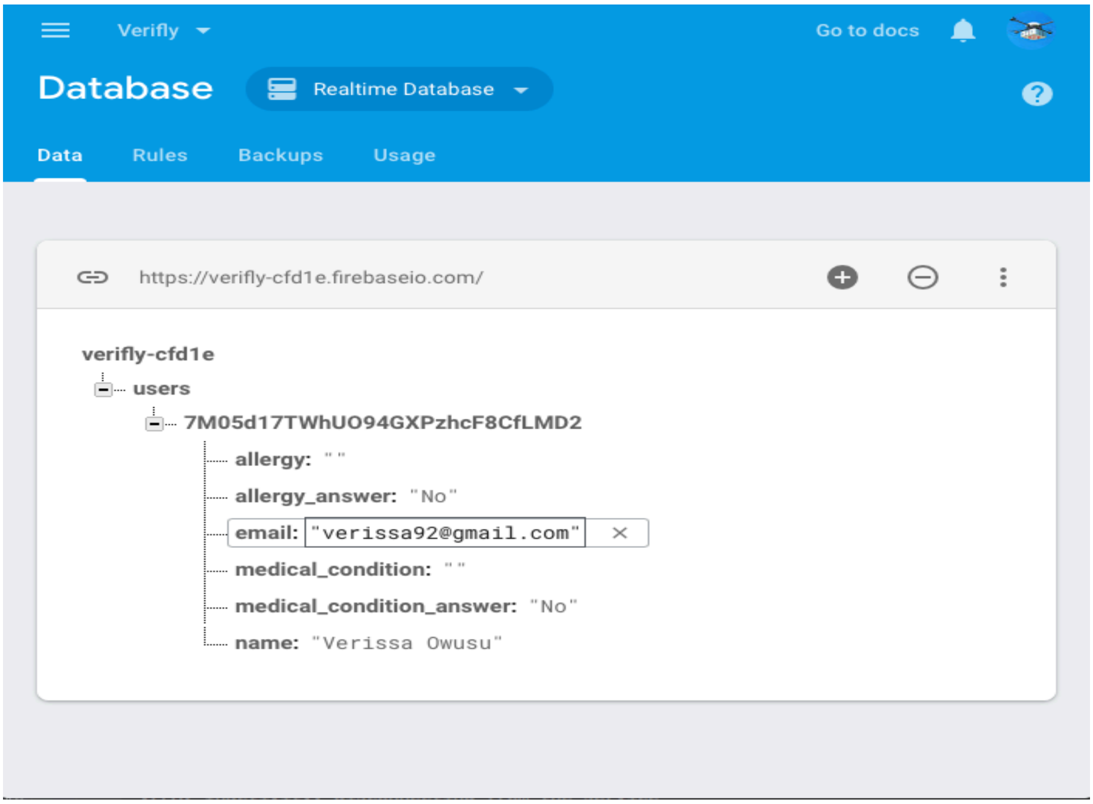
```
private void userLogin() {
    String userEmail = mUserEmail.getText().toString();
    String userPassword = mUserPass.getText().toString();

    //No email address entered by user
    if (TextUtils.isEmpty(userEmail)){
        Toast.makeText(getApplicationContext(),  text: "Enter an email address", Toast.LENGTH_LONG).show();
        return;
    }

    //Invalid email address
    if (!Patterns.EMAIL_ADDRESS.matcher(userEmail).matches()) {
        Toast.makeText(getApplicationContext(),  text: "Enter a valid email address", Toast.LENGTH_LONG).show();
        return;
    }

    //No password entered by user
    if (TextUtils.isEmpty(userPassword)){
        Toast.makeText(getApplicationContext(),  text: "Enter a password", Toast.LENGTH_LONG).show();
        return;
    }

    //password length is less than 6 characters
    if (userPassword.length()< 6){
        Toast.makeText(getApplicationContext(),  text: "Password should have at least 6 characters", Toast.LENGTH_LONG).show()
        return;
    }
```

Figure 5.2: Validation code



```
public class User {

    public String name, email, allergy_answer,allergy,medical_condition_answer,medical_condition;


    public User() {
        // Default constructor required for calls to DataSnapshot.getValue(User.class)
    }

    public User(String name, String email, String allergy_answer,String allergy,String medical_condition_answer,
                String medical_condition) {
        this.name = name;
        this.email = email;
        this.allergy_answer = allergy_answer;
        this.allergy = allergy;
        this.medical_condition_answer = medical_condition_answer;
        this.medical_condition = medical_condition;
    }

}
```

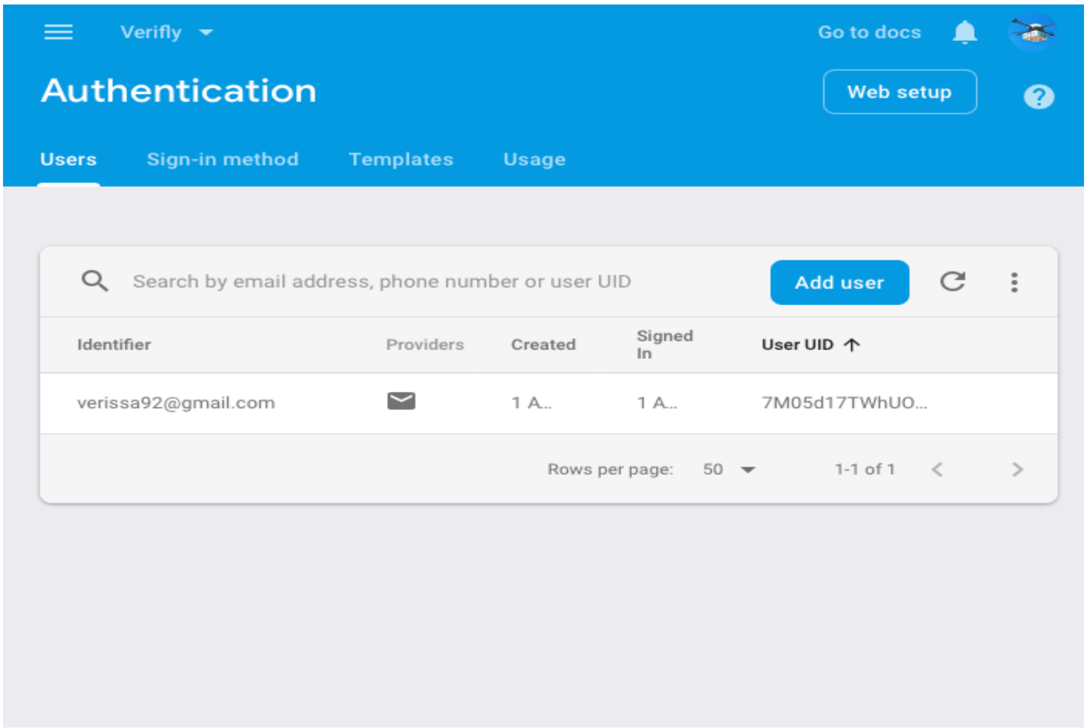
Figure 5.3: User class



https://verifly-cfd1e.firebaseio.com/

```
verifly-cfd1e
  users
    7M05d17TWhUO94GXPzhcF8CfLMD2
      allergy: ""
      allergy_answer: "No"
      email: "verissa92@gmail.com"
      medical_condition: ""
      medical_condition_answer: "No"
      name: "Verissa Owusu"
```

Figure 5.4: User object entry into the database



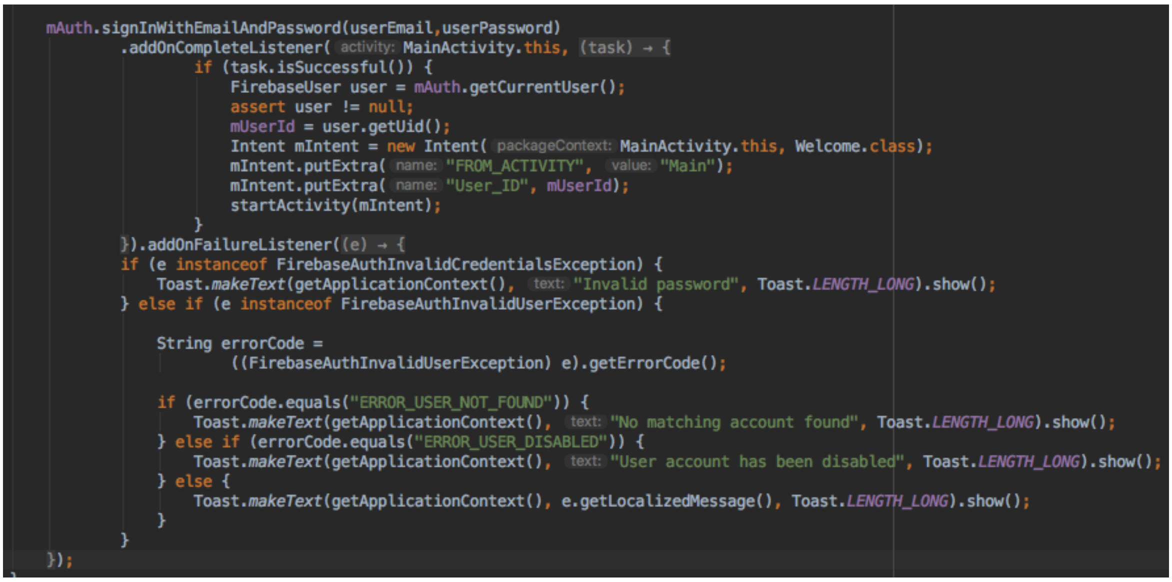
Identifier	Providers	Created	Signed In	User UID ↑
verissa92@gmail.com	✉	1 A...	1 A...	7M05d17TWhUO...

Rows per page: 50 1-1 of 1

Figure 5.5: Authentication details to enable log in in the future

5.2 Interaction with server for user authentication testing

Valid input	**Result**
User details does not exist in the database	User details are entered into the database
Invalid input	**Result**
An alert is given to the user notifying them on the problem, i.e. account already exists	Redirection to the home page to start registration process

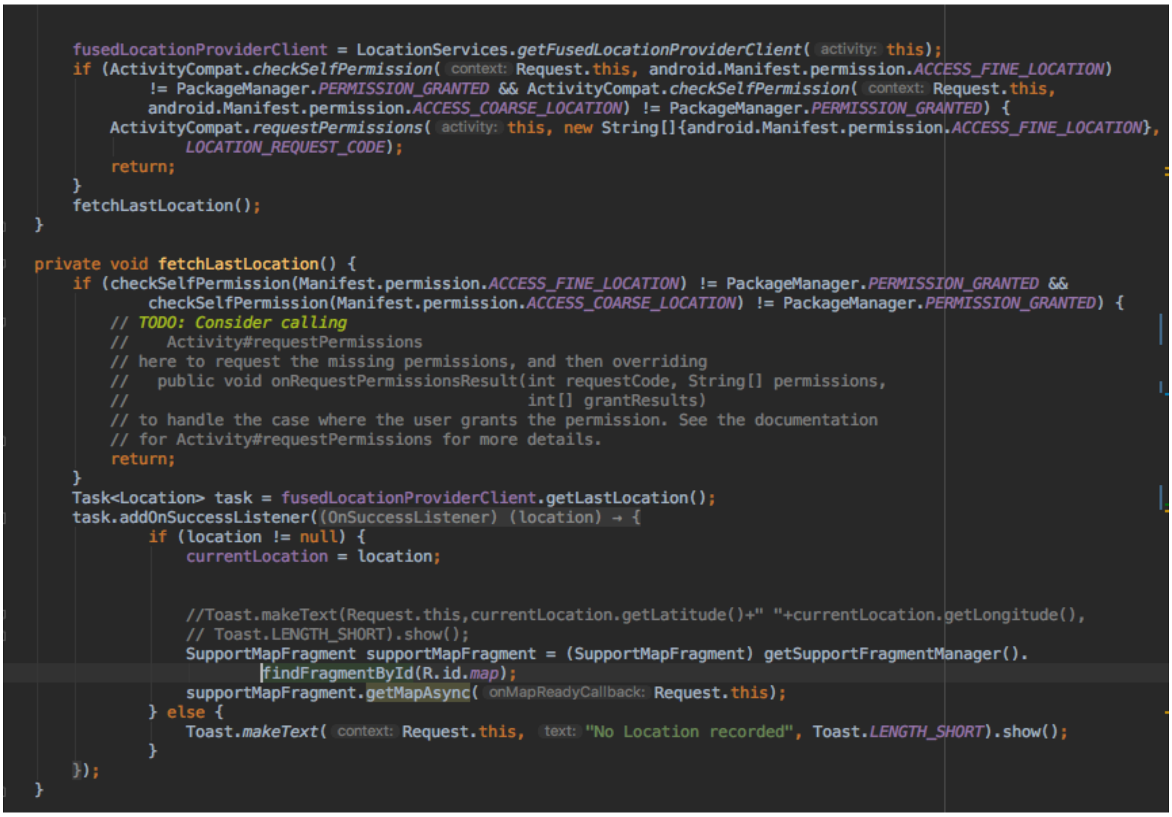


```
mAuth.signInWithEmailAndPassword(userEmail,userPassword)
        .addOnCompleteListener( activity: MainActivity.this, (task) → {
                if (task.isSuccessful()) {
                    FirebaseUser user = mAuth.getCurrentUser();
                    assert user != null;
                    mUserId = user.getUid();
                    Intent mIntent = new Intent( packageContext: MainActivity.this, Welcome.class);
                    mIntent.putExtra( name: "FROM_ACTIVITY",  value: "Main");
                    mIntent.putExtra( name: "User_ID", mUserId);
                    startActivity(mIntent);
                }
        }).addOnFailureListener((e) → {
        if (e instanceof FirebaseAuthInvalidCredentialsException) {
            Toast.makeText(getApplicationContext(),  text: "Invalid password", Toast.LENGTH_LONG).show();
        } else if (e instanceof FirebaseAuthInvalidUserException) {

            String errorCode =
                    ((FirebaseAuthInvalidUserException) e).getErrorCode();

            if (errorCode.equals("ERROR_USER_NOT_FOUND")) {
                Toast.makeText(getApplicationContext(),  text: "No matching account found", Toast.LENGTH_LONG).show();
            } else if (errorCode.equals("ERROR_USER_DISABLED")) {
                Toast.makeText(getApplicationContext(),  text: "User account has been disabled", Toast.LENGTH_LONG).show();
            } else {
                Toast.makeText(getApplicationContext(), e.getLocalizedMessage(), Toast.LENGTH_LONG).show();
            }
        }
});
```

Figure 5.6: Code for authentication

5.3 Request for First aid kit testing

The request for first aid kit function first checks to see if location permissions have been enabled by the user. If they have not an alert is given to the user to enable location access by the application. Thus, a user who has not enabled location access for the application will not be able to request for a first aid kit. The code to get the device location is depicted in Figure 5.7. Figure 5.8 shows a marker illustrating the location of the emulator device.



```
        fusedLocationProviderClient = LocationServices.getFusedLocationProviderClient( activity: this);
        if (ActivityCompat.checkSelfPermission( context: Request.this, android.Manifest.permission.ACCESS_FINE_LOCATION)
                != PackageManager.PERMISSION_GRANTED && ActivityCompat.checkSelfPermission( context: Request.this,
                android.Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions( activity: this, new String[]{android.Manifest.permission.ACCESS_FINE_LOCATION},
                    LOCATION_REQUEST_CODE);
            return;
        }
        fetchLastLocation();
    }

    private void fetchLastLocation() {
        if (checkSelfPermission(Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED &&
                checkSelfPermission(Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
            // TODO: Consider calling
            //    Activity#requestPermissions
            // here to request the missing permissions, and then overriding
            //   public void onRequestPermissionsResult(int requestCode, String[] permissions,
            //                                          int[] grantResults)
            // to handle the case where the user grants the permission. See the documentation
            // for Activity#requestPermissions for more details.
            return;
        }
        Task<Location> task = fusedLocationProviderClient.getLastLocation();
        task.addOnSuccessListener((OnSuccessListener) (location) → {
                if (location != null) {
                    currentLocation = location;


                    //Toast.makeText(Request.this,currentLocation.getLatitude()+" "+currentLocation.getLongitude(),
                    // Toast.LENGTH_SHORT).show();
                    SupportMapFragment supportMapFragment = (SupportMapFragment) getSupportFragmentManager().
                            findFragmentById(R.id.map);
                    supportMapFragment.getMapAsync( onMapReadyCallback: Request.this);
                } else {
                    Toast.makeText( context: Request.this,  text: "No Location recorded", Toast.LENGTH_SHORT).show();
                }
        });
    }
```

Figure 5.7: Code to fetch user location

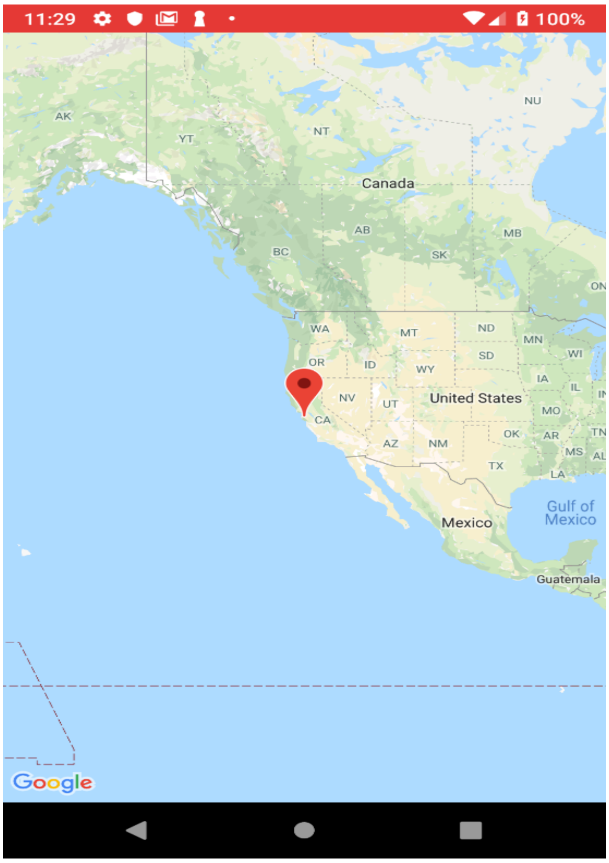
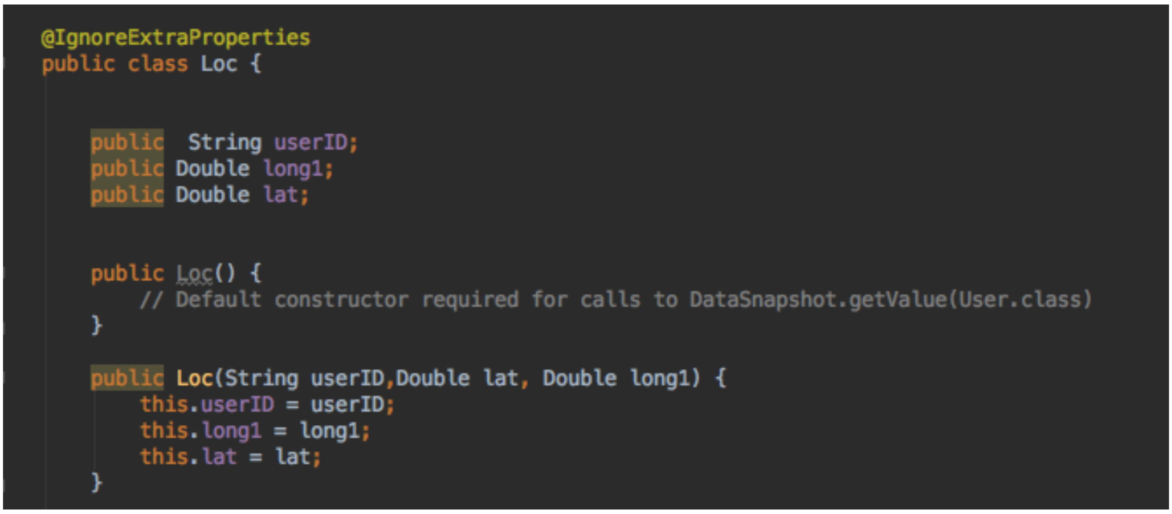


Figure 5.8: Display of emulator location coordinates

5.4 Transmission of location information testing

The GPS location coordinates of the device used to request is obtained and is used to create a Location object which is then inserted into the database. The location object consists of the primary user's id, and double values for the longitude and latitude values. As such if the information received from the request is not valid, the user will be redirected to the home page and will not have a successful redirection to the map page and finally to the injury selection page. The details of the Location class is depicted in Figure 5.9 below.

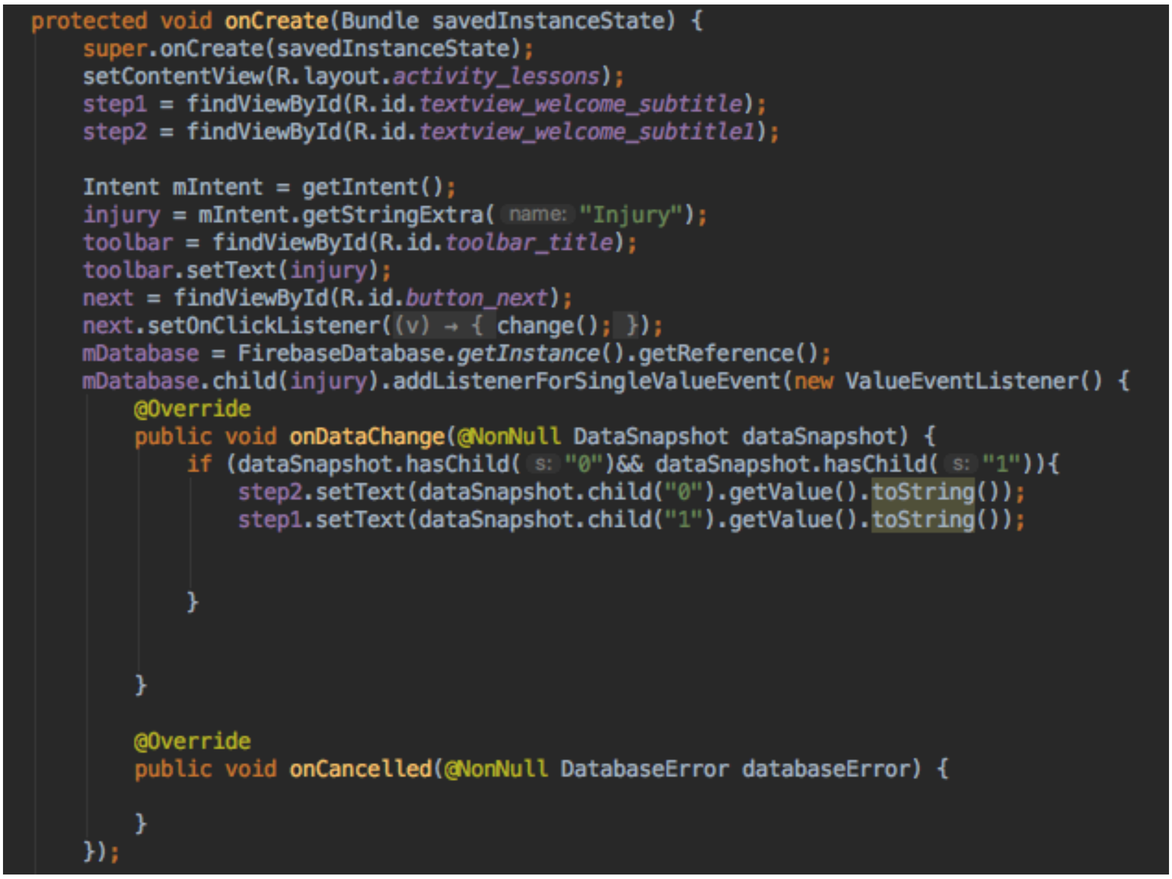


```
@IgnoreExtraProperties
public class Loc {


    public  String userID;
    public Double long1;
    public Double lat;


    public Loc() {
        // Default constructor required for calls to DataSnapshot.getValue(User.class)
    }

    public Loc(String userID,Double lat, Double long1) {
        this.userID = userID;
        this.long1 = long1;
        this.lat = lat;
    }
```

Figure 5.9: Location class

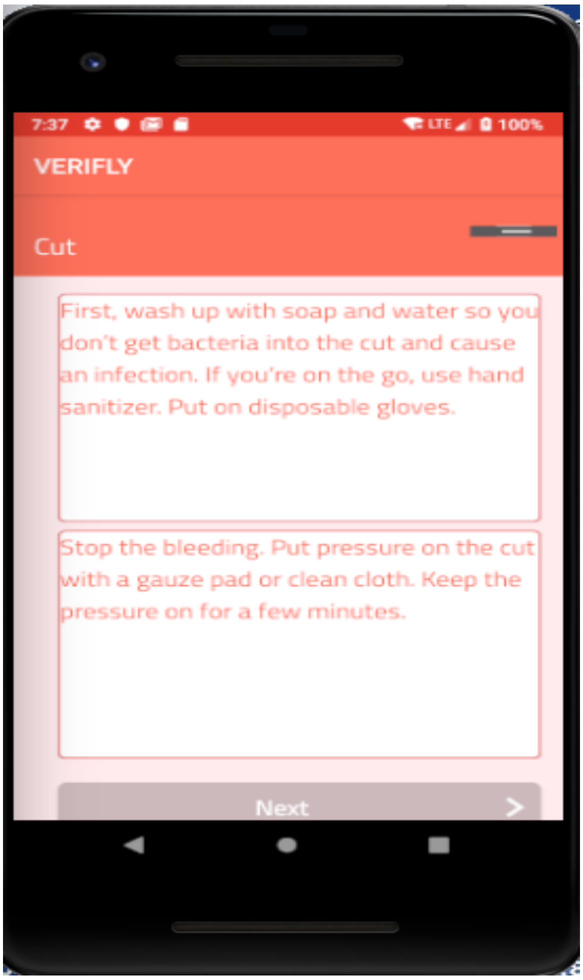
5.5 Learn first aid testing

The valid input needed to start the learn first aid process is a button click. If the user does not click on the specified button the function will not be started. On a successful button click, the user is redirected to a page with a list of common injuries that he or she can learn the first aid care guidelines for the respective injury types. This is only valid if the user has successfully logged into the application. The code and results for this feature is displayed in Figure 5.10 and 5.11 respectively.



```java
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_lessons);
    step1 = findViewById(R.id.textview_welcome_subtitle);
    step2 = findViewById(R.id.textview_welcome_subtitle1);

    Intent mIntent = getIntent();
    injury = mIntent.getStringExtra( name: "Injury");
    toolbar = findViewById(R.id.toolbar_title);
    toolbar.setText(injury);
    next = findViewById(R.id.button_next);
    next.setOnClickListener((v) → { change(); });
    mDatabase = FirebaseDatabase.getInstance().getReference();
    mDatabase.child(injury).addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
            if (dataSnapshot.hasChild( s: "0")&& dataSnapshot.hasChild( s: "1")){
                step2.setText(dataSnapshot.child("0").getValue().toString());
                step1.setText(dataSnapshot.child("1").getValue().toString());


            }


        }

        @Override
        public void onCancelled(@NonNull DatabaseError databaseError) {

        }
    });
```

Figure 5.10: Code to display lessons



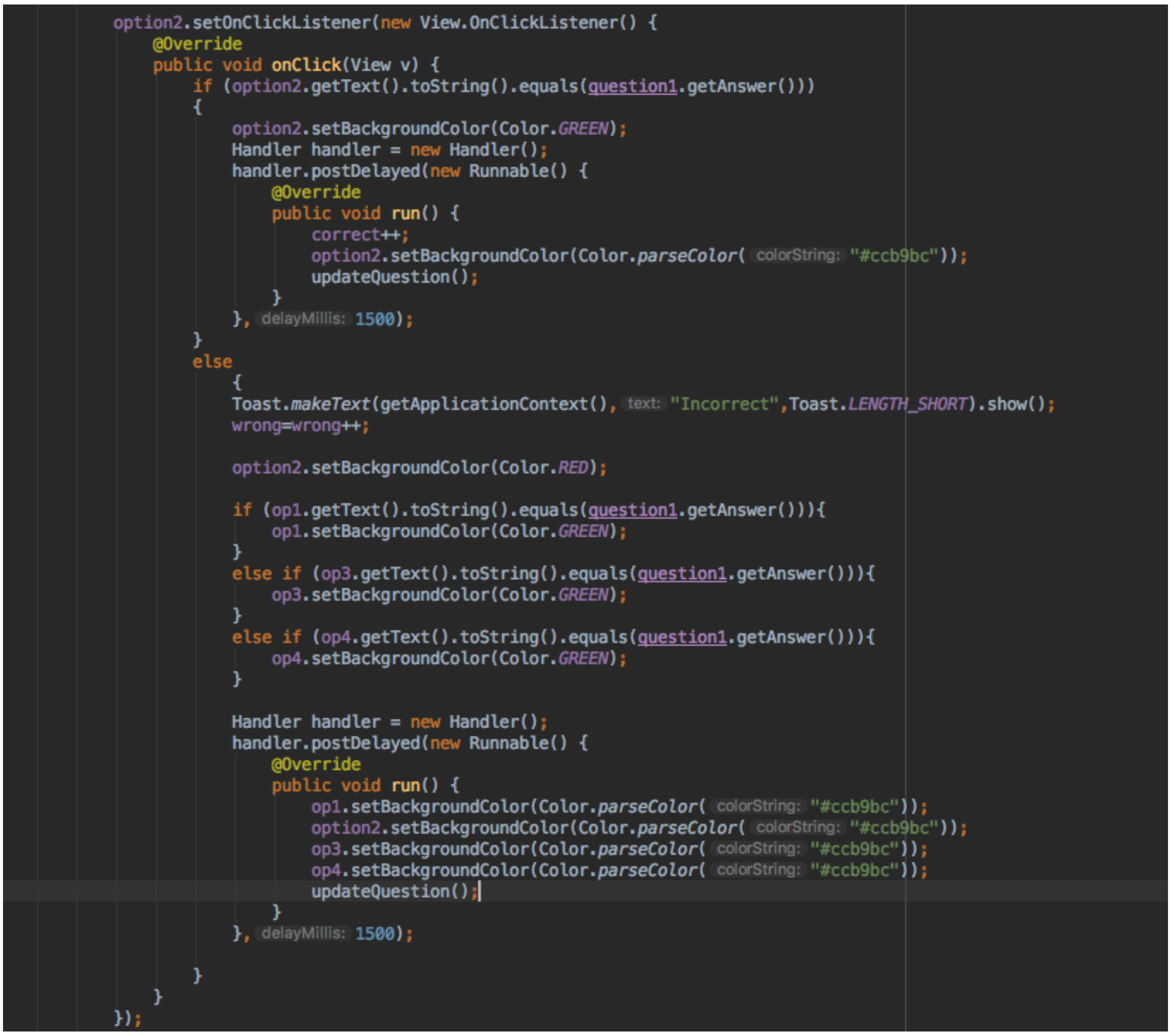
First, wash up with soap and water so you don't get bacteria into the cut and cause an infection. If you're on the go, use hand sanitizer. Put on disposable gloves.

Stop the bleeding. Put pressure on the cut with a gauze pad or clean cloth. Keep the pressure on for a few minutes.

Figure 5.11: First aid lessons display

5.6 Take quizzes to test knowledge testing

A button click is the needed valid input to start the quiz to test a user's knowledge on first aid. If the user does not click on the specified button the function will not be started. On a successful button click, the user is redirected to an activity that starts the quiz process. This is only valid if the user has successfully logged into the application. The user's score is taken with the user id and is inserted into the scores table in the database. Figures 5.12 and 5.13 show the code and display for the quiz feature.

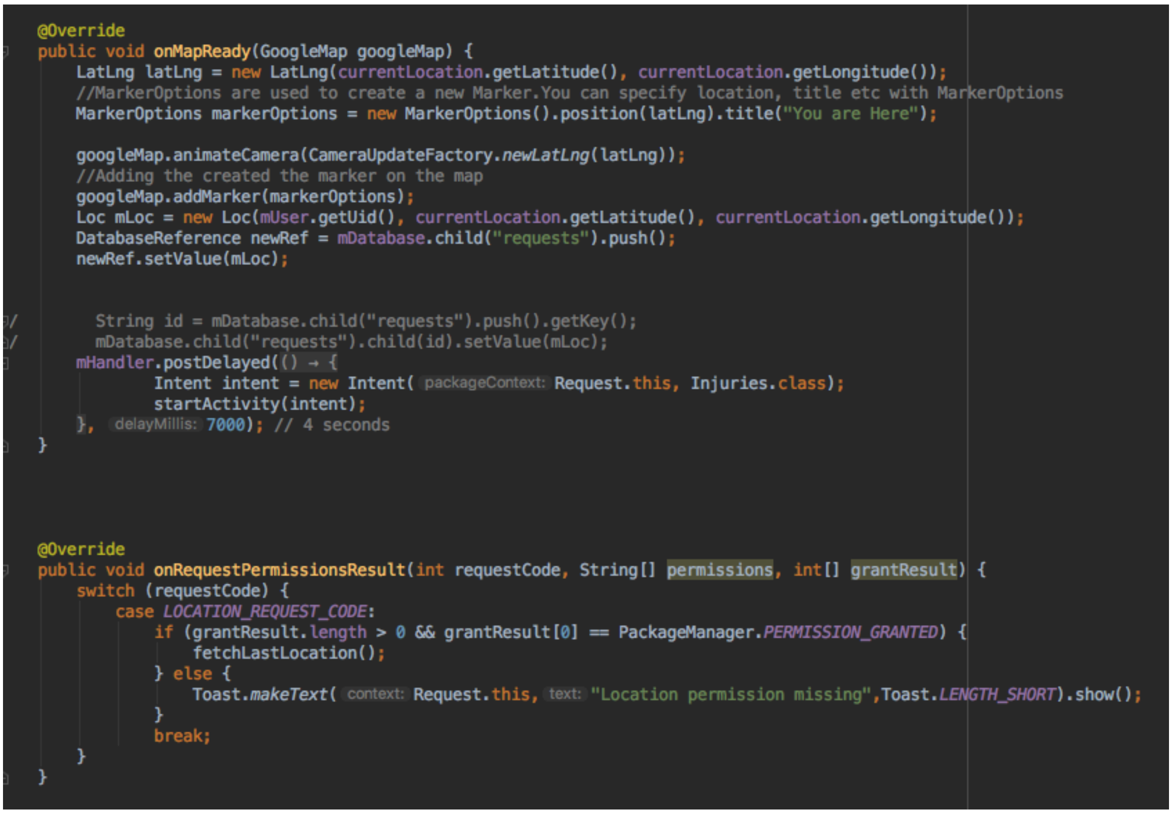


```
option2.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (option2.getText().toString().equals(question1.getAnswer()))
        {
            option2.setBackgroundColor(Color.GREEN);
            Handler handler = new Handler();
            handler.postDelayed(new Runnable() {
                @Override
                public void run() {
                    correct++;
                    option2.setBackgroundColor(Color.parseColor( colorString: "#ccb9bc"));
                    updateQuestion();
                }
            }, delayMillis: 1500);
        }
        else
            {
            Toast.makeText(getApplicationContext(), text: "Incorrect",Toast.LENGTH_SHORT).show();
            wrong=wrong++;

            option2.setBackgroundColor(Color.RED);

            if (op1.getText().toString().equals(question1.getAnswer())){
                op1.setBackgroundColor(Color.GREEN);
            }
            else if (op3.getText().toString().equals(question1.getAnswer())){
                op3.setBackgroundColor(Color.GREEN);
            }
            else if (op4.getText().toString().equals(question1.getAnswer())){
                op4.setBackgroundColor(Color.GREEN);
            }

            Handler handler = new Handler();
            handler.postDelayed(new Runnable() {
                @Override
                public void run() {
                    op1.setBackgroundColor(Color.parseColor( colorString: "#ccb9bc"));
                    option2.setBackgroundColor(Color.parseColor( colorString: "#ccb9bc"));
                    op3.setBackgroundColor(Color.parseColor( colorString: "#ccb9bc"));
                    op4.setBackgroundColor(Color.parseColor( colorString: "#ccb9bc"));
                    updateQuestion();
                }
            }, delayMillis: 1500);

        }
    }
});
```

Figure 5.12: Code to check selected option

5.7 View requests location testing

The administrator has to log in first for their details to be validated and authenticated before they can access the view a request's location. In order for the map to be displayed showing the locations of requests the administrator has to click on the view button to start the function. The function then allows the authenticated user to retrieve details from the location table in the database and then display them on the map. The map display is done through a function depicted in Figure 5.14 below.

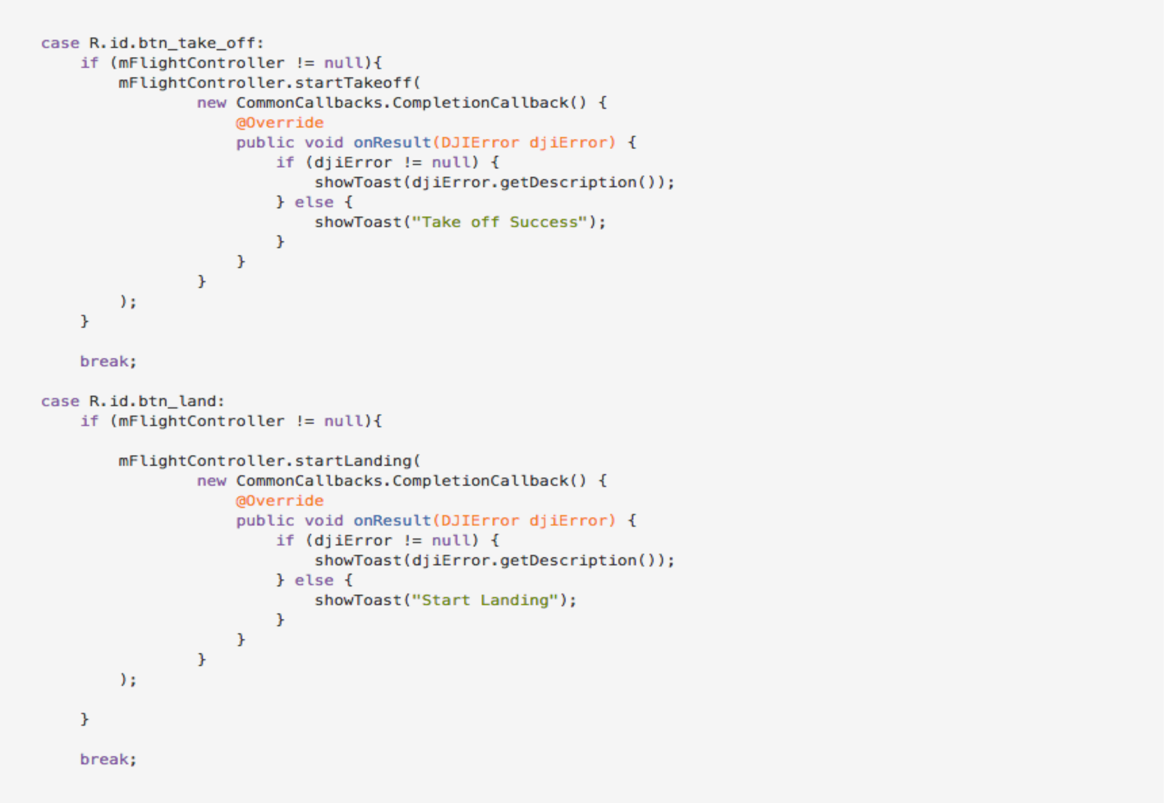


```
    @Override
    public void onMapReady(GoogleMap googleMap) {
        LatLng latLng = new LatLng(currentLocation.getLatitude(), currentLocation.getLongitude());
        //MarkerOptions are used to create a new Marker.You can specify location, title etc with MarkerOptions
        MarkerOptions markerOptions = new MarkerOptions().position(latLng).title("You are Here");

        googleMap.animateCamera(CameraUpdateFactory.newLatLng(latLng));
        //Adding the created the marker on the map
        googleMap.addMarker(markerOptions);
        Loc mLoc = new Loc(mUser.getUid(), currentLocation.getLatitude(), currentLocation.getLongitude());
        DatabaseReference newRef = mDatabase.child("requests").push();
        newRef.setValue(mLoc);


/          String id = mDatabase.child("requests").push().getKey();
/          mDatabase.child("requests").child(id).setValue(mLoc);
        mHandler.postDelayed(() → {
                Intent intent = new Intent( packageContext: Request.this, Injuries.class);
                startActivity(intent);
        },  delayMillis: 7000); // 4 seconds
    }



    @Override
    public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResult) {
        switch (requestCode) {
            case LOCATION_REQUEST_CODE:
                if (grantResult.length > 0 && grantResult[0] == PackageManager.PERMISSION_GRANTED) {
                    fetchLastLocation();
                } else {
                    Toast.makeText( context: Request.this,  text: "Location permission missing",Toast.LENGTH_SHORT).show();
                }
                break;
        }
    }
```

Figure 5.14: Function to display location on the map

5.8 Drone flight to location testing

For this component of the system to be implemented, the administrator has to first of successfully view request locations thus ensuring that the location coordinates in the database have been accessed. Once the user has been able to view requests' locations then he or she can click on the start flight button, which will send the location coordinates of the first request to the drone to start the automatic flight control process. After the flight has been started the database entry is removed and inserted into an archives table.



```
case R.id.btn_take_off:
    if (mFlightController != null){
        mFlightController.startTakeoff(
                new CommonCallbacks.CompletionCallback() {
                    @Override
                    public void onResult(DJIError djiError) {
                        if (djiError != null) {
                            showToast(djiError.getDescription());
                        } else {
                            showToast("Take off Success");
                        }
                    }
                }
        );
    }

    break;

case R.id.btn_land:
    if (mFlightController != null){

        mFlightController.startLanding(
                new CommonCallbacks.CompletionCallback() {
                    @Override
                    public void onResult(DJIError djiError) {
                        if (djiError != null) {
                            showToast(djiError.getDescription());
                        } else {
                            showToast("Start Landing");
                        }
                    }
                }
        );

    }

    break;
```

Figure 5.15: Code to control take-off and landing [15]

5.9 First aid kit and procedure display testing

Once the request for a first aid kit has gone through and the user has been successfully alerted that a drone is on the way. The user can then select the applicable injury for his or her current situation, an example is shown in Figure 5.16 below. When this is done a pictorial description of the kit together with guidelines for administering first aid care is displayed. If the user does not start the request process, they will not be able to view the kit and guidelines. The guidelines are depicted as shown in Figures 5.17 and 5.18, the code is found in Figure 5.19.

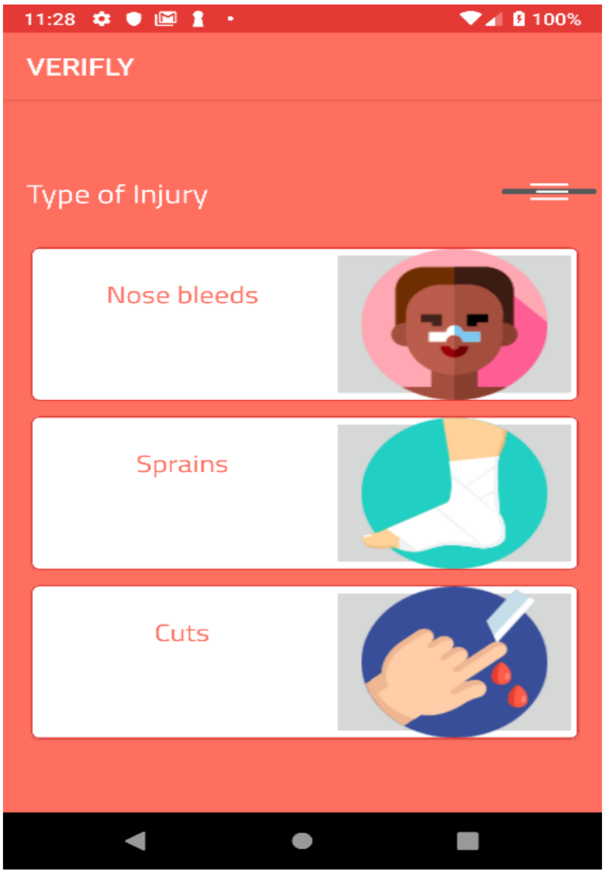
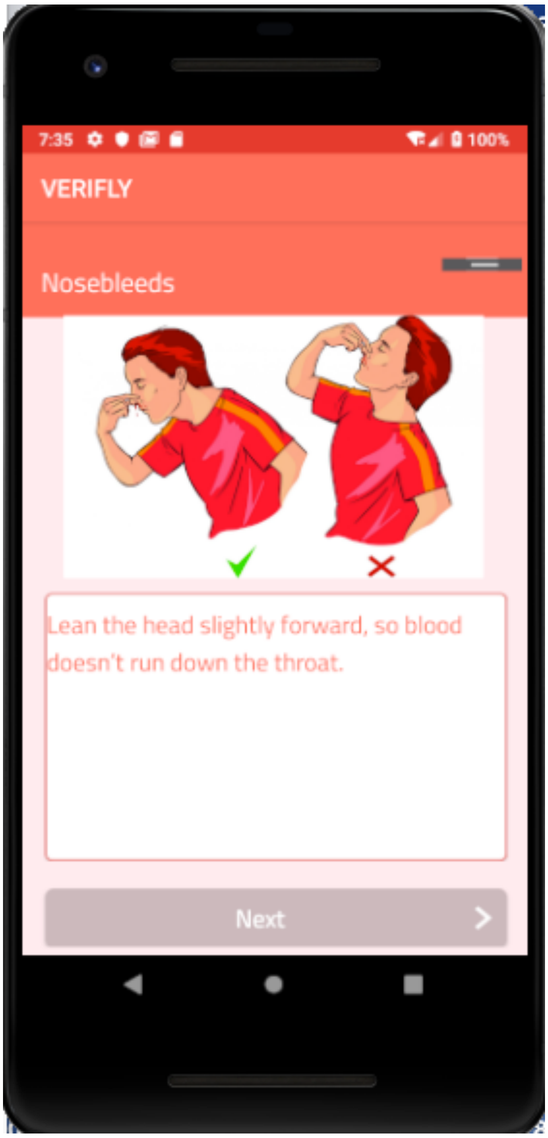
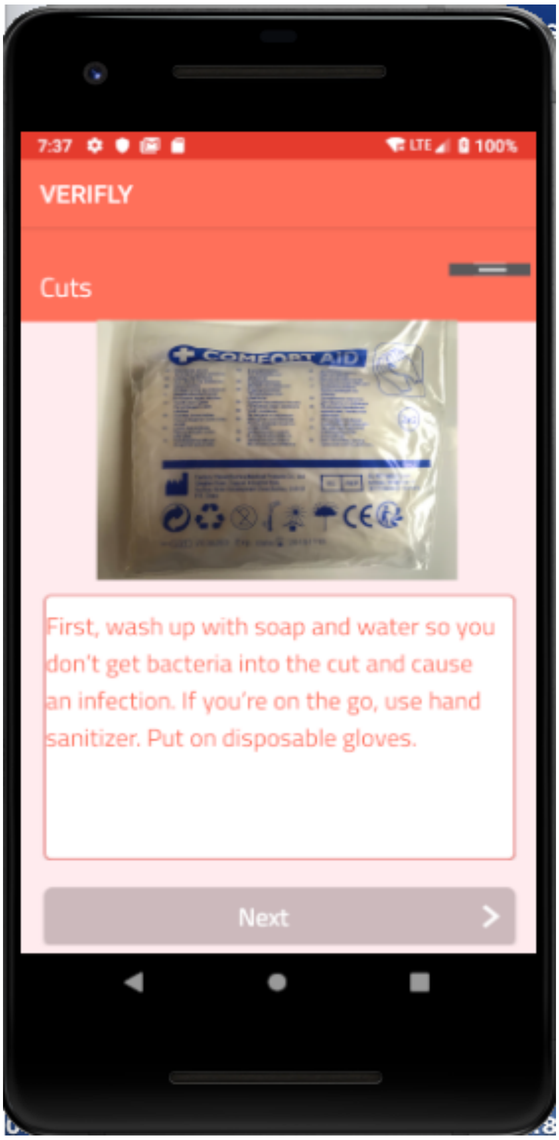


Figure 5.16: List of injuries



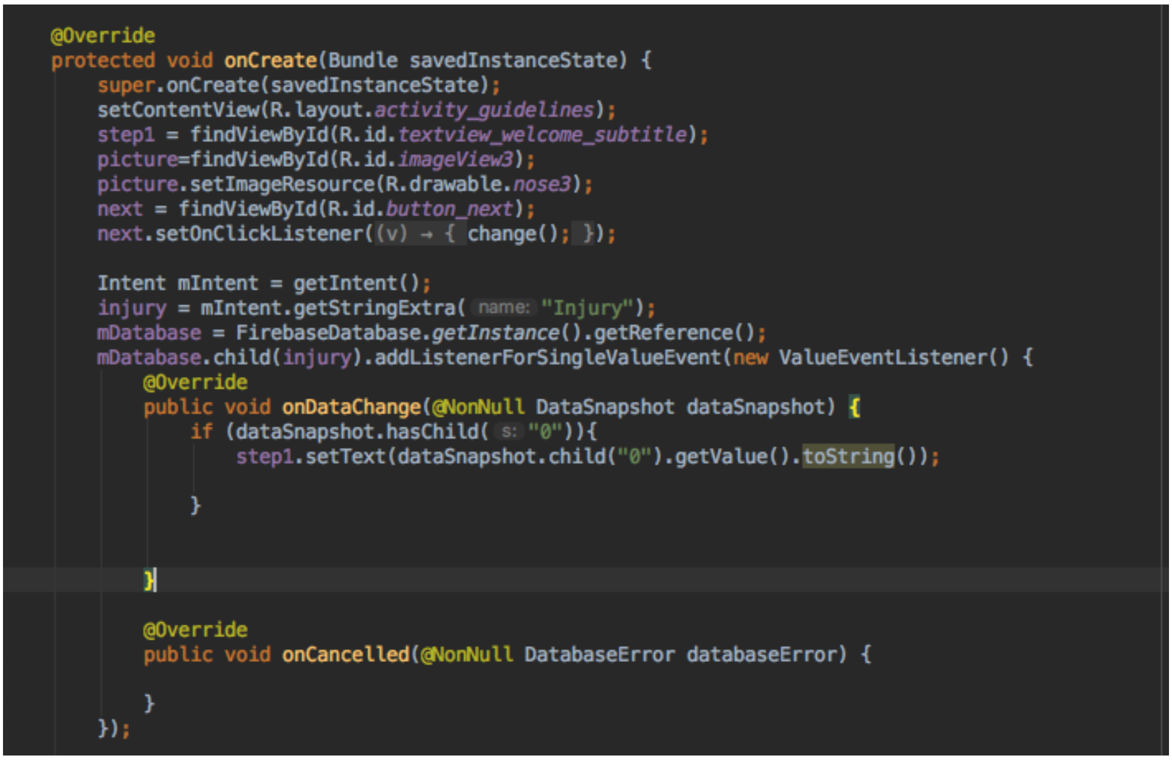
Lean the head slightly forward, so blood doesn't run down the throat.



First, wash up with soap and water so you don't get bacteria into the cut and cause an infection. If you're on the go, use hand sanitizer. Put on disposable gloves.

Figure 5.17: Guidelines for nosebleeds

Figure 5.18: Guidelines for cuts



```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_guidelines);
    step1 = findViewById(R.id.textview_welcome_subtitle);
    picture=findViewById(R.id.imageView3);
    picture.setImageResource(R.drawable.nose3);
    next = findViewById(R.id.button_next);
    next.setOnClickListener((v) -> { change(); });

    Intent mIntent = getIntent();
    injury = mIntent.getStringExtra( name: "Injury");
    mDatabase = FirebaseDatabase.getInstance().getReference();
    mDatabase.child(injury).addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
            if (dataSnapshot.hasChild( s: "0")){
                step1.setText(dataSnapshot.child("0").getValue().toString());

            }


        }

        @Override
        public void onCancelled(@NonNull DatabaseError databaseError) {

        }
    });
```

Figure 5.19: Code for fetching guidelines from the database

5.10 Sending drone back to home location testing

When the user is done administering first aid care he or she attaches the kit to the drone and clicks on a finished button. The finished button starts a return flight to the health centre by calling the return to home function. If the user does not click on the button the drone will not return to the health centre. In order to test this, feature the simulator for the DJI Phantom 4 drone, DJI Assistant 2 for Phantom, was used. The drone has to be connected to the simulator with a USB.

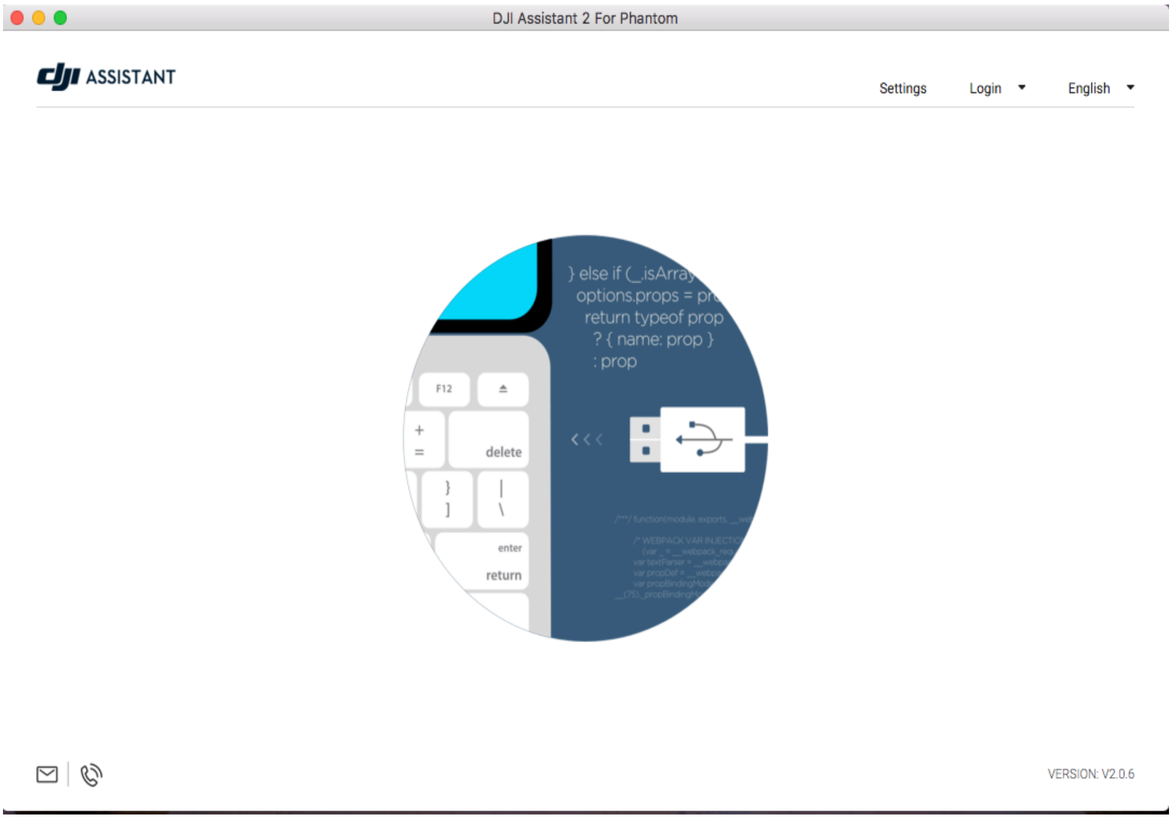
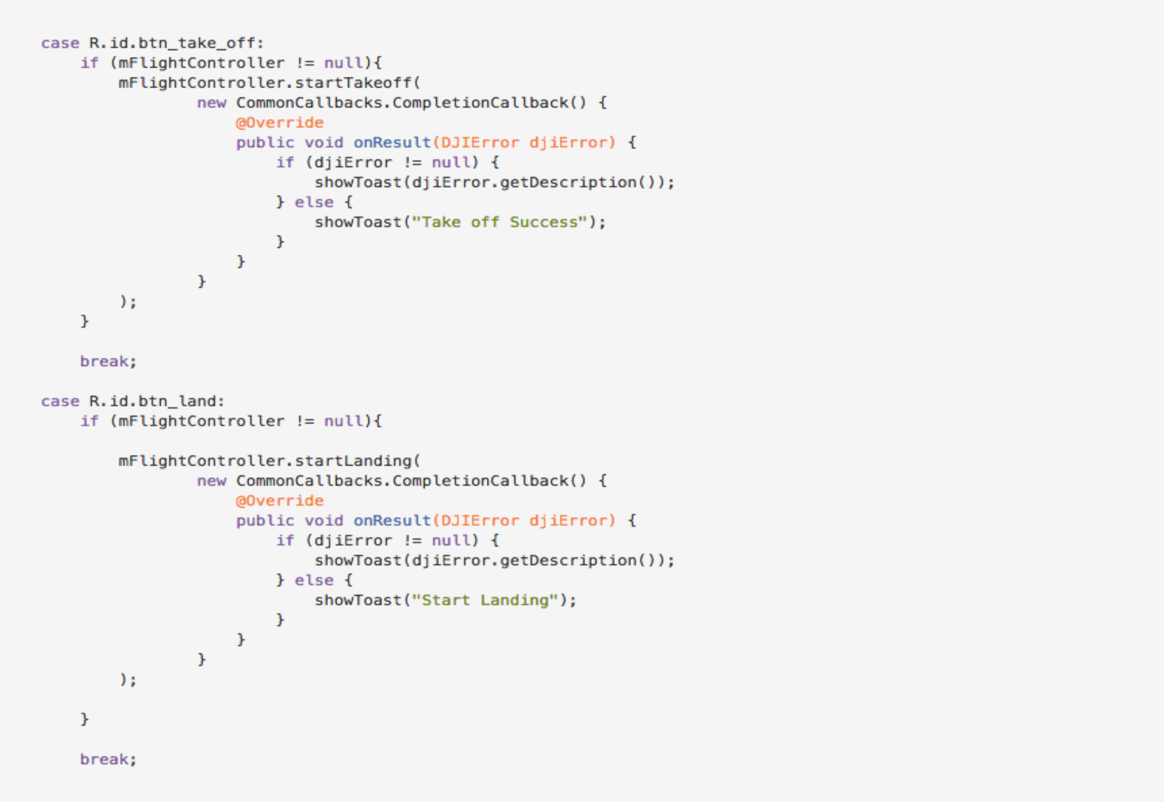


Figure 5.14: DJI Assistant 2 for Phantom



```java
case R.id.btn_take_off:
    if (mFlightController != null){
        mFlightController.startTakeoff(
                new CommonCallbacks.CompletionCallback() {
                    @Override
                    public void onResult(DJIError djiError) {
                        if (djiError != null) {
                            showToast(djiError.getDescription());
                        } else {
                            showToast("Take off Success");
                        }
                    }
                }
        );
    }

    break;

case R.id.btn_land:
    if (mFlightController != null){

        mFlightController.startLanding(
                new CommonCallbacks.CompletionCallback() {
                    @Override
                    public void onResult(DJIError djiError) {
                        if (djiError != null) {
                            showToast(djiError.getDescription());
                        } else {
                            showToast("Start Landing");
                        }
                    }
                }
        );

    }

    break;
```

Figure 5.15: Code to control take-off and landing

5.11 User reports testing

An authenticated administrator has the options to create user reports pertaining to the system. The option to create reports is displayed on the home page where the function is enabled by a button click. If the administrator has not been authenticated, he or she cannot create user reports.

Chapter 6: Conclusion and Recommendations

6.1 Conclusion

This project sought to develop a more efficient means of administering emergency care to people who get injured on the Ashesi University campus. To build an efficient system, the requirements elicitation process showed that the system should enable users get a first aid kit quickly. Thus, the user waiting time will be reduced. To do this a system – Verifly, was implemented. This system allows injured users to request for a first aid kit through a mobile application rather than walking to the health centre with whatever injury they have. The first aid kit is attached to a drone and then sent to the request location.

The relevance of this project can be measured by measuring the time duration spent getting emergency care through the traditional methods and using the Verifly system. This test will show the efficiency of the system in reducing the user's wait time in getting first aid.

6.2 Limitation

Since the system is dependent on the interaction between the mobile application, database and the drone. As such in the case where the Health centre staff have their phones switched off, they will not be able to start the flight of the drone to the user's location. Also, the mobile application only caters for Android smartphone users and as such does not cater to iOS and Windows smartphone users. Another limitation of the system is its dependence on network connectivity as such, the user must always be connected to the internet to be able to use the system effectively [13].

Due to time constraints as well, there were some limitations to the testing process. As such there was no provision for user testing and acceptance testing to be able to get feedback from the members of the community, to make changes to the system in order to make it much more efficient. Also, since the drone did not belong to me, I could not get to test the automated flight control feature to ensure that it worked perfectly or figure out any flaws in that feature. The drone is also needs to be connected to the laptop with a Universal Serial Bus (USB) in order to be tested using the simulator, which was also not possible because of the unavailability of the drone.

6.3 Recommendation

This section discusses some improvements that can be made on the system to further increase efficiency.

- Addition of tracking features. Currently on the system, users who request for the first aid drone cannot know exactly where the drone is at any moment till it gets to their location. Users are only notified of the fact that the drone is on its way to them. To combat this feeling of uncertainty, a tracking feature can be implemented to allow users graphically track the movement of the drone to their location. Such a feature could probably be implemented using the Google Maps API to display both the user's location and that of the drone and showing the amount of time left for arrival and showing the path the drone must travel to get to the user.
- Set location. When an injured user requests for a first aid kit, a function is run to get the user's location and then displayed on a map. There could be situations where the user's location coordinates may not be entirely accurate. Some locations may also have architectural features that may limit the drone's access to the user. An additional feature could be implemented to allow the user to move the marker that is shown on the map to their exact location if they think there has been an error. Thus, when the marker is moved, the new coordinates is what is picked and sent into the database. This will improve the accuracy level of deliveries and make the system more efficient.
- Automate start of drone flight. The administrator who is part of the staff at the health centre initiates the start of the drone's flight by signing into the application and then

clicks on a button to start the process for initiating the start of flight. Thus, an administrator is given alert notifications when a request comes into the system so he or she can start the drone's flight process. This means that in a situation where the administrator's phone is switched off or has low network connectivity, there will be no communication to start the flight process of the drone. As such a feature should be implemented that will automatically start the drone's flight process as soon as the request is entered into the database.

- Videos demonstrating first aid care should be incorporated into the application to ensure that users of the application are able to follow meticulously in order to ensure that the right processes are employed. Another check could be an added in person check-up, where the health centre personnel follows after the drone to the injured user's location to ensure that they have been treated properly.

References

[1] WHO | Injuries and violence: the facts. *WHO*. Retrieved April 1, 2019 from https://www.who.int/violence_injury_prevention/key_facts/en/

[2] Gustavo C. Bravo, Diego M. Parra, Luis Mendes, and A. Manuel de Jesus Pereira. 2016. First aid drone for outdoor sports activities. In *2016 1st International Conference on Technology and Innovation in Sports, Health and Wellbeing (TISHW)*, 1–5. DOI:https://doi.org/10.1109/TISHW.2016.7847781

[3] Olaf Hallan Graven, Jon-Vegard S∅rli, Joakim Bj∅rk, Dag Andreas Hals Samuelsen, and Jan Dyre Bjerknes. 2017. Managing disasters-rapid deployment of sensor network from drones: Providing first responders with vital information. In *2017 2nd International Conference on Control and Robotics Engineering (ICCRE)*, 184–188. DOI:https://doi.org/10.1109/ICCRE.2017.7935067

[4] Ahmed Bitar, Aliaa Jamal, Hesham Sultan, Nour Alkandari, and Mohammed El-Abd. 2017. Medical Drones System for Amusement Parks. In *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*, 19–20. DOI:https://doi.org/10.1109/AICCSA.2017.62

[5] Auditor Generals Department, Ghana. Performance Audit of the Auditor General on Road Safety in Ghana. www.ghaudit.org/gas/site/reports/download-report/20

[6] 2016. Structure of an Android Operating System | edu CBA. *EDUCBA*. Retrieved April 1, 2019 from https://www.educba.com/structure-of-an-android-operating-system/

[7] GeekyAnts. 2017. Introduction to Firebase🔥. *Hacker Noon*. Retrieved April 1, 2019 from https://hackernoon.com/introduction-to-firebase-218a23186cd7

[8] Mobile SDK Introduction - DJI Mobile SDK Documentation. Retrieved April 1, 2019 from https://developer.dji.com/mobile-sdk/documentation/introduction/mobile_sdk_introduction.html

[9] Meet Android Studio. *Android Developers*. Retrieved April 1, 2019 from https://developer.android.com/studio/intro

[10] Optimize location for battery. *Android Developers*. Retrieved April 1, 2019 from https://developer.android.com/guide/topics/location/battery

[11] Missions - DJI Mobile SDK Documentation. Retrieved April 1, 2019 from https://developer.dji.com/mobile-sdk/documentation/introduction/component-guide-missions.html

[12] Android Graph View plotting library. Retrieved April 1, 2019 from http://www.android-graphview.org/

[13] Manav Singhal and Shukla. Implementation of Location based Services in Android using GPS and Web Services. *International Journal of Computer Science Issues*. Retrieved April 1, 2019 from http://www.ijcsi.org/articles/Implementation-of-location-based-services-in-android-using-gps-and-web-services.php

[14] Judith Odili Uchidiuno, Justin Manweiler, and Justin D. Weisz. 2018. Privacy and Fear in the Drone Era: Preserving Privacy Expectations Through Technology. In *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems* (CHI EA '18), LBW505:1–LBW505:6. DOI:https://doi.org/10.1145/3170427.3188457

[15] DJI Simulator Tutorial. Retrieved April 23, 2019 from https://developer.dji.com/mobile-sdk/documentation/androidtutorials/SimulatorDemo.html

Appendix

ASHESI

ASHESI UNIVERSITY DRONE POLICY

The purpose of this drone policy is to define the requirements and guidelines for the use of drones on property owned or controlled by Ashesi University. This policy seeks to clearly define rules to be followed by persons who wish to use drones on the Ashesi University campus so as to ensure public safety, privacy and security is respected.

DEFINITIONS

Drone: an aircraft that has no direct human intervention from within the aircraft to control flight. The aircraft is rather controlled from a remote station by an operator.

Operator: a person responsible for controlling the flight of a drone.

Restricted areas: these are areas on the Ashesi University campus that has been identified as off limits to the use of drones due to security or privacy measures.

Visual Line of Sight: the operator of the drone must be able to clearly see the drone while in flight and as such guide its path to avoid collisions.

SCOPE

This policy is applicable to Ashesi University students, Ashesi University staff and faculty and visitors to the campus and the use of drones on the campus premises.

RESPONSIBILITIES

As an operator of a drone on the Ashesi University campus you are responsible for:

I. Requesting permission from the appropriate University authorities before operating a drone on the university campus. All requests for research purposes should be accompanied with a project description to the CSIS department coordinator in order to obtain permission and access to the university's owned drones.

II. Inexperienced people who wish to operate a drone on campus should do so under the guidance or supervision of someone with some experience.

III. Visual line of sight must be maintained with drones at all times.

IV. Do not fly drones higher than 400 feet vertically.

V. Drones may not fly within a 30-meter (98 feet) radius of buildings or vehicles without prior express permission.

VI. Operating drones during campus events or where crowds are gathered is not prohibited but permission must be sought from the people to be recorded and the event organizers.

VII. Drones may not be operated in restricted areas such as washroom facilities on the campus and hostel facilities where members of the community will have a reasonable expectation of privacy is met.

VIII. The use of Ashesi University owned drones for recreational purposes is prohibited. However personal drones can be used once permission has been granted.