**Personal Security Smartphone Application**

# Final Year Project Report



**TrackMe**



Matthew Finn | 13480362 |

B.Sc. Computer Science & Information Technology

Academic Supervisor: Dr. Desmond Chambers

Statement Of Originality

I declare that this project is my original work except where stated.

Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Table Of Contents

Abstract 7

Acknowledgements 8

1. Introduction 9

1.1 Background 9

1.2 Project Objectives 10

1.3 Context 11

1.3.1 Mobile Application Development 11

1.3.2 Learn Server-Side Scripting Language 11

1.3.3 Database Administration 11

1.3.4 Application Use 11

2. Technology Overview & Selection 12

2.1 Technology Overview 12

2.1.1 Mobile Application Platform 12

Android Platform 12

Advantages of Android Development 13

Disadvantages of Android Development 13

iOS Platform 14

Advantages of iOS Development 14

Disadvantages of iOS Development 14

Platform Selection 14

2.1.2 Application Technologies 15

Location Tracking 15

Considerations 15

Selection 15

Fall Detection 16

Considerations 16

Selection 16

Messaging Service 16

Considerations 16

Selection 16

2.1.3 Web Hosting Service 17

Considerations 17

Selection 17

2.1.4 Server Side Scripting 17

Considerations 17

Selection 17

2.1.5 Database 18

Considerations 18

Selection 18

2.2 Similar Applications 18

2.2.1 bSafe 19

2.2.2 EmergenSee 20

2.2.3 DigitalLife 21

2.2.4 Family Locator 22

2.2.5 SafeTrek 22

2.2.6 Conclusion 23

3. Implementation 24

3.1 Mobile Application 24

3.1.1 Application Module 25

3.1.2 Activity Module 26

Splash 26

StartupActivity 27

MainActivity 28

MyPreferenceActivity 29

3.1.2 Fragment Module 30

HomeFragment 31

TrackUserFragment 33

RegisterFragment 33

LoginFragment 34

AboutFragment 35

HelpFragment 36

ProfileFragment 36

3.1.3 Services Module 37

GPS Service 37

Fall Detection Service 39

3.1.4 Helper Module 40

CloudDBHandler 40

GPSHelper 41

LocalDBHandler 41

MessageHandler 41

SessionManager 42

WidgetHandler 42

3.1.5 UI 43

3.1.6 Miscellaneous 43

Application Permissions 43

Application Dependencies 43

SDK Build numbers 44

3.2 Server-Side Project 44

DBFn 44

DBConfig 45

DBConnect 46

Register 46

Login 47

UpdateLocation 48

GetLocation 48

3.3 Database 49

4. Testing & Evaluation 50

4.1 Testing 50

Testing Of GPS Tracking Algorithm 50

Testing Of Fall Detection Algorithm 51

4.2 User Feedback 51

4.3 Functional Requirements Review 52

Core Requirements 52

Additional Requirements 54

4.4 Comparison With Similar Applications 55

5. Conclusion 56

5.1 Project Outcome 56

5.2 Future Work 57

Profile Editing 57

Data Analytics 57

Password Retrieval/Change 57

References 57

Appendices 60

Table of Figures 60

Abstract

The aim of this project is to create a smartphone application that utilises the unique capabilities of smartphones in order to aid the personal security for vulnerable individuals.

The application was designed to utilise a smartphones built-in GPS and accelerometer to track the users location and detect scenarios in which the user may be in a vulnerable situation. This involves detection of scenarios such as a user physically falling, or veering outside a particular boundary when such behavior is unexpected of them. When possible (i.e. User device has Internet connectivity) a users location data is uploaded to a backend server where other users of the application can query their most recent location and the time at which they were present at that location. The application also has the ability to send an emergency SMS in the case of an emergency such as a fall being detected or a user travelling outside of a pre-defined boundary (i.e. The boundary being a pre-defined distance from their starting position).

The application was designed, developed and deployed on the Android platform using the Android Studio IDE.

The project also incorporates the use of both PHP and MySQL to develop the supporting backend remote database server.

The project meets the primary aims of allowing periodic check-ins from a users smartphone, remote monitoring of smartphone device location, detection of lack of user movement (i.e. the implementation of fall detection) and movement outside certain boundaries.

Acknowledgements

I would like to express sincere gratitude to all that helped me throughout the course of completing this project as without the continued support I would not have completed the project to as high a standard as I have done.

Firstly I would like to thank Dr. Desmond Chambers, my project supervisor, for keeping the project on track. He provided helpful advice, suggestions & guidance, particularly regarding how the application should function, throughout the year and for that I am very grateful.

I would also like to thank my friends and colleagues who participated in the testing process of the application throughout the development process.

And finally I would like the College of Engineering & Informatics, specifically the Discipline of Information Technology.

1. Introduction

# 1.1 Background

Personal security has always been a very important issue as no matter how adequate the personal security measures in place are it is always something that can be enhanced. Security is especially an issue for more vulnerable people such as younger children and the elderly. This is because of hazards such as getting lost, going missing or suffering a fall.

Over the last number of years there has been a staggering amount of missing people reported in Ireland alone. For example in 2013 there were approximately 7700 people reported missing and in 2014 there were approximately 9100 people reported missing. [4]

Injuries due to falls are commonplace especially among the elderly in our population. It is estimated the 1/3 of the population over 65 years old have a fall causing injury each year. [5]

According to statistics gathered by IrishHealth there are approximately 7000 people over 65 are admitted to hospital with injuries relating to a fall each year. [6]

Due to the advancements in smartphone capabilities gives rise to the possibility of enhancing personal security. These advancements allow things such as location tracking, fall detection, SMS transmission without user interaction & boundary detection.

According to figures accumulated in 2015 2.37m people in Ireland own a smartphone with 55% of that 70% owning an android smartphone. [7][8]

The issues with personal security and the advancement in capabilities of smartphones are what inspired the idea behind TrackMe.

TrackMe is an application that provides the functionality of user location tracking, boundary tracking & fall detection. Location information is uploaded to the backend server where it can be analysed and other TrackMe users can track a users latest location via their username. If a fall is detected or if a user passes outside a specified boundary an SMS will be sent to a user defined emergency contact.

# 1.2 Project Objectives

This project involved the creation of a fully functional smartphone application and the application was designed to provide the end user with the following core features:

* Application tracks user location at a user-specified time interval.
* User can query another users latest location update from server.
* User can enable fall detection algorithm.
* User can enable boundary detection from tracking start point.

These core features required the following to also be implemented in the project:

* Allowing the application to upload location data to backend server when possible (i.e. Internet connection is available on user device).
* Allowing the application permission to access devices system services and hardware such as Internet, accelerometer & GPS location hardware.

The goal is to implement these features with a user-friendly interface.

The main purpose of the application is to enhance the personal security of the user by providing means by which to alert others of possible danger to them & also be able to track the users latest location, using their username, if they are deemed to be in any form of unexpected situation or a fall has been detected.

# 1.3 Context

Additional objectives and motivations that I had for undertaking this project are as follows:

## 1.3.1 Mobile Application Development

I have always been interested in developing smartphones applications but prior to undertaking this project I was always under too many time constraints to afford me the opportunity to attempt some mobile app development. I saw this project as a great opportunity to develop an application of my own and gain some knowledge about mobile application development.

## 1.3.2 Learn Server-Side Scripting Language

Throughout the completion of my degree I was never afforded the opportunity to undertake any great amount of development using a server side scripting language such as PHP or Ruby. I used this project as an opportunity to develop my knowledge of PHP by using it to complete server-side scripting.

## 1.3.3 Database Administration

During the completion of my degree I have already gained some experience in the setup and administration of databases, such as MySQL and Microsoft SQL Server, but I wanted to improve on my current level of proficiency by setting up a more complex backend server than I have used in previous projects.

## 1.3.4 Application Use

Having previously been in situations where I would have found an application such as TrackMe useful I believed that this project was interesting to see how I would be able to aid people in scenarios where their personal security was compromised.

2. Technology Overview & Selection

This chapter is a state of the art review of the technologies that are relevant to this project. It includes a brief overview of current technologies available that I have investigated during the completion of this project. This chapter also includes a review of products & services available that offer similar features to this project.

# 2.1 Technology Overview

## 2.1.1 Mobile Application Platform

In this section I will discuss the possible different platform that I could have used to develop TrackMe and the advantages and disadvantages of these platforms.

The platforms that I investigated as part of the project were Android and iOS as the project specification gave the freedom to utilize a platform of my choice. I felt that these were the most appropriate platforms given that they’re the two most popular platforms worldwide.

An example of their popularity is that for the year ending December 2016 there were approximately 1,270.000 Android smartphones sold and in the same timeframe there were approximately 216,000 iOS Devices sold. [9]

### Android Platform

Android is a mobile device operating system built primarily for use with smartphones and tablets. Androids UI is based on emulating real world direct manipulations such as tapping, swiping & dragging on-screen objects in order to manipulate them.

#### Advantages of Android Development

* Significantly larger user base in comparison to any other smartphone operating system meaning that applications developed for Android devices have a much larger potential reach.
* Online documentation for Android development is largely accessible and very thorough.
* Predominantly coded in Java which is the programming language that I am most familiar with.
* Deployment to Android application store, i.e. Google Play Store, is a much simpler & less costly process than deployment of an iOS application to the App Store.
* The Android development platform allows the use of third-party tools and libraries so allows a very broad range of functionality within applications.
* Broad range of possible development environments can be used to develop applications such as Eclipse & Android Studio.

#### Disadvantages of Android Development

* Application development can be more difficult due to the lenient implementation constraints applied to development practices thus leading to the potential for an increased number of bugs.
* There is a large fragmentation of operating system versions meaning that development of features need to be carefully considered as they could potentially prevent a significant portion of the potential market from being able application to run the application on their device.
* Large array of devices to consider for UI compatibility.

### iOS Platform

iOS is a mobile device operating system built exclusively for Apple devices such as the iPhone & the iPad. Similarly to Android the UI is based on the emulations of real world gestures.

#### Advantages of iOS Development

* iOS development features quite rigourous development constraints which lead to an application which can often be to a higher standard in comparison to other operating systems’ application.
* Smaller array of possible devices to develop for meaning that it is easier to develop application that is compatible with the UI of all devices.
* More likely to be compatible with most devices as there is less fragmentation of operating system releases.

#### Disadvantages of iOS Development

* Much smaller potential user base in comparison to the Android platform.
* Development on the iOS platform requires XCode IDE that is only available on Apple branded computers through the iOS App Store.
* The process of deploying an application to the App Store is costly, time consuming and has an expansive level of auditing.

#### Platform Selection

For the development of this application I decided to use the Android platform as it allowed me to access the largest user base and gave more freedom when developing as it allows for the use of many 3rd party libraries during.

## 2.1.2 Application Technologies

In this section I will discuss the possible methods of implementing the key features of the application at a high level. I will also give an insight into my rationale for my technology selection throughout the course of TrackMe’s development.

### Location Tracking

#### Considerations

Implementation could be achieved using GPS location sensors on phone when location services are activated.  The smartphone devices could user it’s network provider to detect a users location when GPS is turned inactive.  A user could be tracked by either sending their location through a network socket to another user or by uploading their location information to a backend server where another user can query their location updates.Throughout my research it became clear that a users address is only available when they have an Internet connection on their device as there is no alternative way to access a Geocoder. I did not consider this to be a major problem as a user can still obtain their latitude and longitude values without needing a connection.

#### Selection

For the development of this application I decided to implement the following:

* A method that would receive the most accurate location data at all times depending on the network and Internet connectivity states of the device. (I.e. Use best available to device at the time)
* Location data would be pushed to the backend server, using an Internet connection. I considered the fact needing an internet connection may be a problem but in recent times having always on Internet connections on smartphone devices has become commonplace so I decided that this would be the most secure and robust way to make user location data available to the necessary individuals.

### Fall Detection

#### Considerations

Fall detection could be implemented in several ways using a smartphone devices accelerometer. Fall detection can be programmed to be either very sensitive or very ignorant to possible falls. Generally speaking a fall is illustrated on a devices’ accelerometer by a sudden spike in gravitational direction. Another consideration that needed to be made was that the phone accelerometer would pick up natural gravitational pull of approximately 9.8m/s at idle.

#### Selection

For the development of a fall detection algorithm I decided that the best approach would be to create an algorithm that was quite sensitive as for the purpose of personal safety I felt that a false positive was a better outcome than a false negative when detecting a fall.

### Messaging Service

#### Considerations

There is a range of possible social media platforms that can be used for messaging such as Viber, Whatsapp and Facebook Messenger applications as well as standard SMS services. For this application the fundamental characteristic I wanted for my messaging platform was that a message could be sent without any user input or interaction. I felt this was a fundamental requirement as the messaging feature of the application would only be used in case of emergency.

#### Selection

I decided to use SMS as the applications’ messaging service as it was the only viable option that could send an emergency message without the need for any user interaction. I felt that this was the best option considering that this functionality would only be used in cases where a possible danger or an emergency has been detected.

## 2.1.3 Web Hosting Service

### Considerations

For Implementation of a web server there were many viable options such as Amazon Web Services, Google Web Server and NUI Galway’s Danu6 Linux server.

### Selection

I decided to use NUI Galway’s DANU6 Linux web server as it already had all the technologies I would possibly need to complete my project.

Danu6 is a Gentoo Linux cluster that runs:

* Apache 2.4 and MySQL 5.5
* PHP 5.5
* Java 1.7
* Python 2.7 & 3.3,
* PhpMyAdmin PHP 5.5

## 2.1.4 Server Side Scripting

### Considerations

For this part of the project I considered server-side scripting language such as Ruby and PHP. Both of these languages very easily are able to accomplish what I needed from my server interaction.

### Selection

I decided to use PHP as it would easily accomplish what I needed and it looked like Ruby had a much steeper learning curve. PHP also had an abundance of online information and tutorials that would be very useful if I was to run into problems.

## 2.1.5 Database

### Considerations

For the implementation of a database to store user profile and location details I considered several viable options such as Microsoft SQL Server, MySQL and MongoDB. All of these above options can be setup on NUI Galway’s Danu6 server and offer you a network accessible database with full user authorisation and control.

### Selection

I decided to use a MySQL server as I had previous experience in administering a MySQL database and it offered all the functionality that I needed from a database.

# 2.2 Similar Applications

During the development of any project it is important to identify if similar products already exist or whether your project offers something unique as then there is the potential to obtain a patent for your idea.

In my research of applications that provide similar functionality to the proposed features of TrackMe I have found several applications. These include BSafe, Emergensee. DigitalLife, Family Locator and SafeTrek.

All these applications are Android based personal security applications. Some share common features with TrackMe but ultimately they all have the same goal of augmenting the personal security of individuals.

The features of each of these applications are outlined below.

## 2.2.1 bSafe

One of the applications I found with similar features to TrackMe was bSafe. [3]

bSafe offers the following features:

* Social Personal Safety Network
* Location Sharing
* Location Tracking
* Location Check-In
* Check-In Timer
* Fake Call Triggers
* Guardian Alert Button



Figure 2-2‑ bSafe Application

## 2.2.2 EmergenSee

Another application I found with similar features to that of TrackMe was EmergenSee. [13]

EmergenSee offers the following features:

* Live streaming of video & audio
* GPS location data
* Preset timer for check-in
* Text & email distress notifications
* Incident recording capability
* 24/7 monitoring
* Real-time precautionary escort
* Pre-set safety contacts



Figure 2-2‑2 EmergenSee Application

## 2.2.3 DigitalLife

Personal security application DigitalLife is another app operating in the domain of personal security. [12]

It offers the following features:

* Send messages to frequent contacts from within the application.
* Share location information with Monitoring Center Professionals in emergency situations.
* Use a countdown timer in uncomfortable situations that will send an alert to the monitoring center when the time expires (e.g.: walking to the car alone late at night).

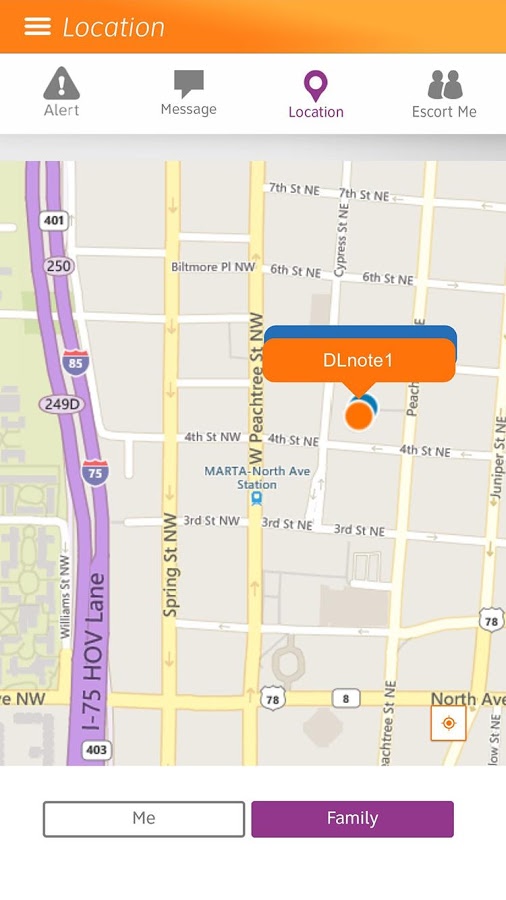


Figure 2‑2‑3 DigitalLife Application

## 2.2.4 Family Locator

Another personal security Android application I discovered during my research was Family Locator. [14]

It offers the following features:

* Create your own groups, called “Circles,” of loved ones, friends, teammates etc.
* View real-time location of members on private map
* Receive real-time alerts when “Circle” members leave or arrive a destination.



Figure 2-2‑4 Family Locator Application

## 2.2.5 SafeTrek

SafeTrek is a personal security application that offers users the ability to make a passive connection to emergency services if they feel they are in danger. This is achieved with minimal user interaction and serves as the only functionality involved in the application. [15]

## 2.2.6 Conclusion

It is clear from my research that there isn’t any patentable functionality present in the TrackMe application as any key purpose of the application has already been implemented in other personal security applications.

Of the products that I have researched during the development of TrackMe it seems that Emergensee, BSafe and Digital Life all contain similar features to that of TrackMe.

*BSafe* offers location sharing and tracking similar to TrackMe. It also offers guardian alert but rather than it being in the case of detection of a possible emergency situation it offers this functionality through the means of a button press. Emergensee offers similar features to TrackMe such as remote location monitoring, pre-set safety contacts & SMS distress messages.

DigitalLife also shares some common ground with TrackMe as it has the ability to send SMS distress messages and it implements remote location monitoring. It also implements a countdown timer before sending a distress signal in a similar way to TrackMe.

FamilyLocator offers similar features to that of TrackMe such as location tracking and remote monitoring of others’ locations but seems to operate in a more specialised manner with the introduction of groups.

SafeTrek operates in the personal security domain but there are no similar features present that are comparable to *TrackMe*.

3. Implementation

# 3.1 Mobile Application

This section will detail the implementation of the mobile application that was developed using Android Studio IDE. The application was developed in a modularized way so that each module served a particular purpose.

The application has been built with “Minimum SDK Version 15” and “Target SDK Version 24”. The table below shows that this means the application will be compatible with android platform version 4.0.3 up to version 7.0. [1]

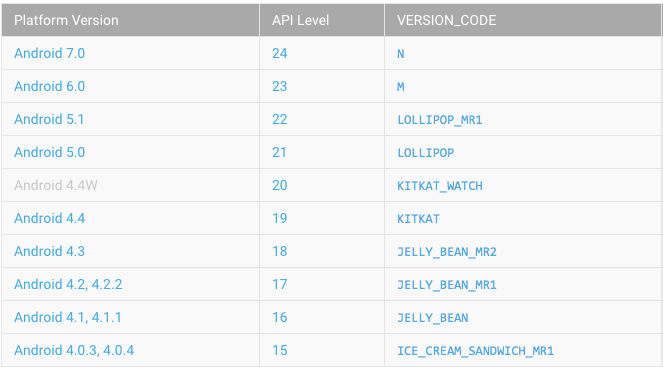


Figure 3‑1‑ Android Platform Versions

## 3.1.1 Application Module

Macintosh HD:Users:matthew:Desktop:Screenshots:Screen Shot 2017-03-25 at 22.10.01.png

Figure 3-1-2 Application Module

In order for the application to be able to interact with the backend server I used Android’s Volley Library. Volley is an HTTP library that makes networking for Android apps easier and most importantly, faster. [10]

The application module sets up the core Volley objects, such as the Volley request queue, and URLS needed to store user location.

The Volley request queue is a handler for request objects. It manages worker threads for running the network operations, reading from and writing to the cache and parsing responses. [11]

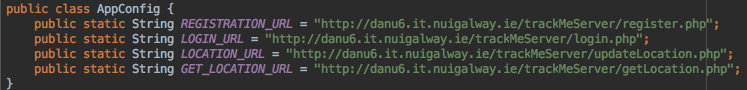


Figure 3‑1-3 AppConfig.java

## 3.1.2 Activity Module

## Macintosh HD:Users:matthew:Desktop:Screenshots:Screen Shot 2017-03-25 at 23.42.02.png

Figure 3-1-4 Activity Module

The activity module of the application was built to handle fragment transaction & the application navigation drawers. The application navigation drawer allows the user to access the application features such as login, registration, home screen & preferences screen.

### Splash

The Splash activity implements the applications’ splash screen. A splash screen is a screen usually seen upon app start up that displays branding for the application. It is implemented in order to give the application time to start up.

The splash screen implemented for TrackMe is displayed for 5 seconds.

This class queries if the user device has location services turned on and if not it asks the user to enable location services as they’re vital to the functionality of this application. If Location Services are enabled this class proceeds to check is a user login session is active using the *SessionManager* (Detailed later). If there is a login session active the *MainActivity* is initiated and if not the *StartupActivity* is initiated.

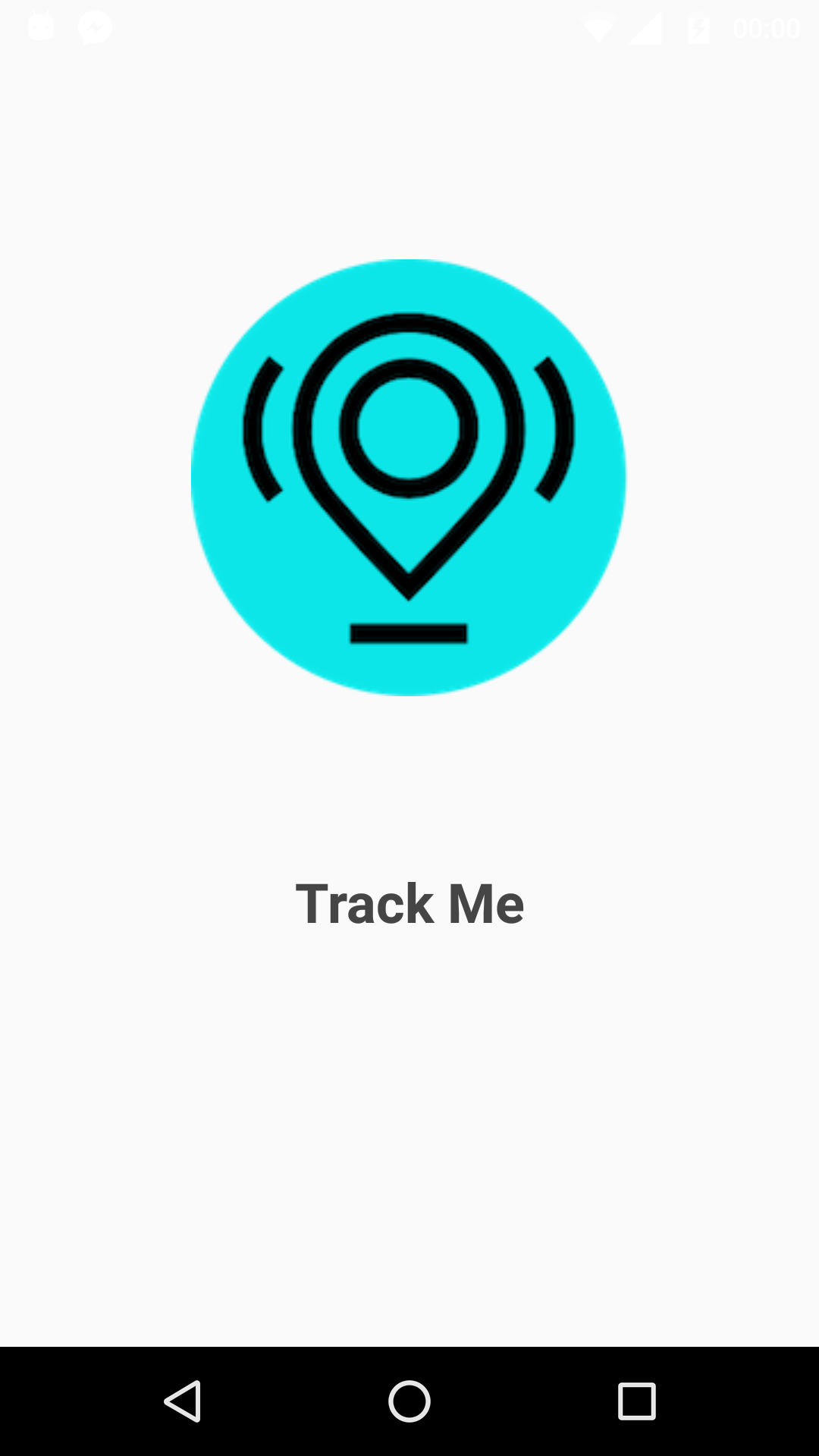
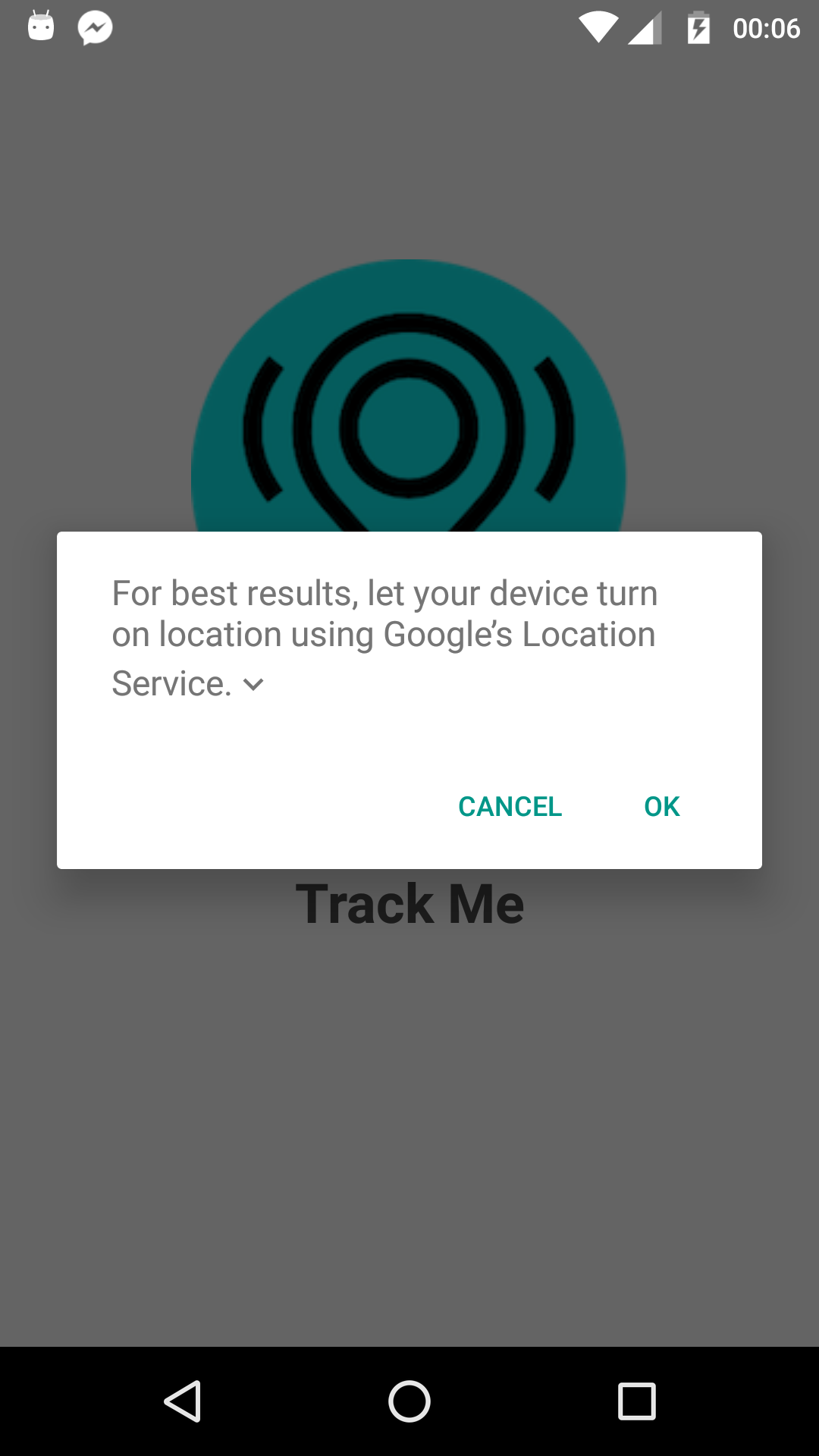
 

Figure 3-1-5 Splash Screen Figure 3‑1-6 Location Services Request

### StartupActivity

The *StartupActivity* class hosts the Login & Registration fragments as well providing access to the About & Help Fragments. This activity also hosts the startup navigation drawer which allows the user to navigate between the available fragments. This activity uses the local database handler class to query if a database for TrackMe exists on the user device. If there is an existing database it is assumed that the user has already registered an account so the user is navigated to the login page, if not the user is navigated to the registration page.

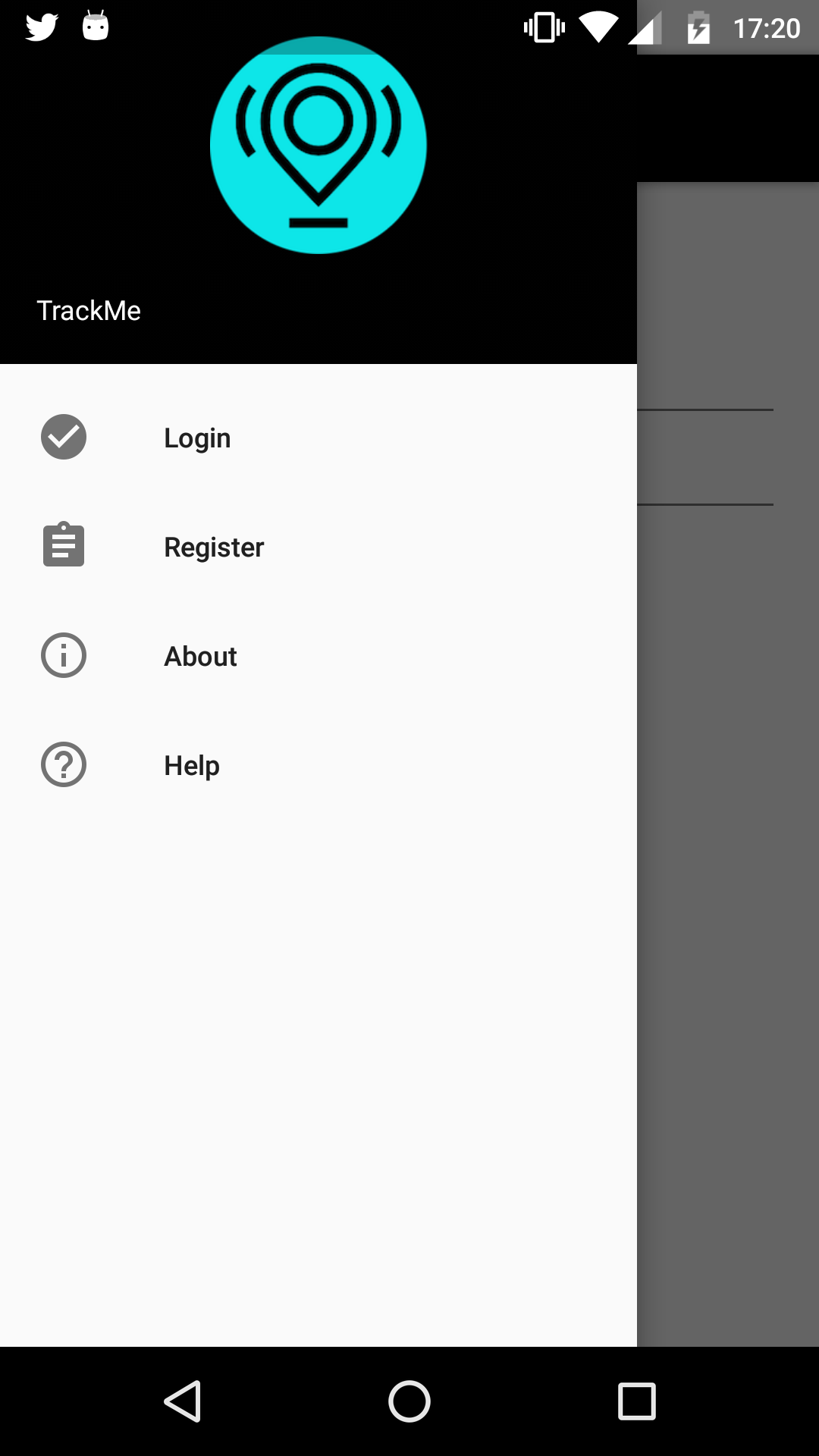


Figure 3‑1-7 StartupActivity Navigation Drawer

### MainActivity

The *MainActivity* class hosts the applications main navigation drawer as well as handling the navigation between all of the fragments that control the main features of the app.

The main activity also handles the notification that is created when the application detects a possible fall. This activity receives, through the implementation of a broadcast receiver, the countdown timer information from a broadcast sent from the fall detection service. If the stop button on the notification is not clicked before the time runs out a text is sent to the users emergency contact letting them know that the user may just have fallen.

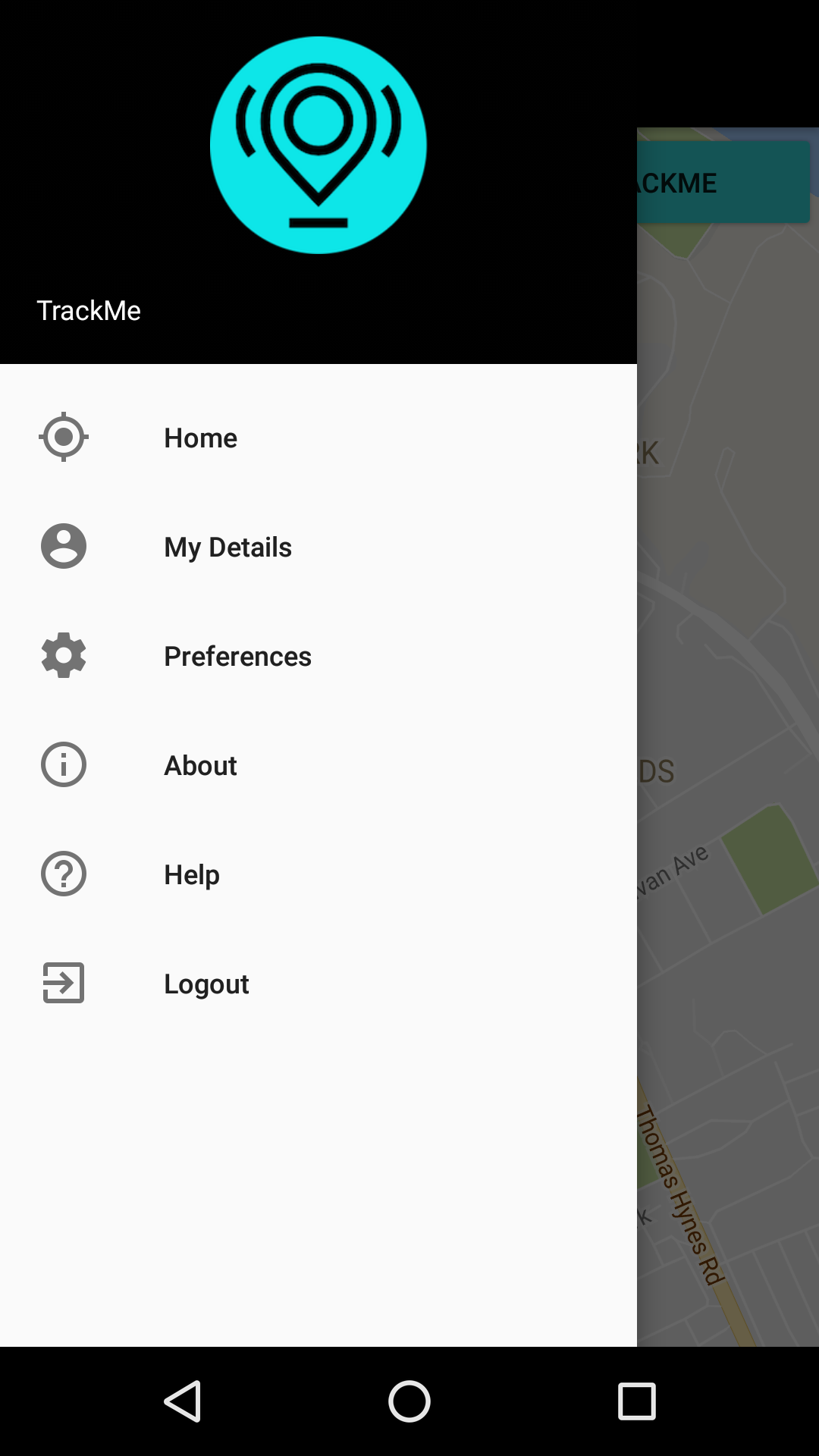


Figure 3‑1-8 MainActivity Navigation Drawer



Figure 3‑1-9 Fall Detection Notification

### MyPreferenceActivity

This activity hosts the application preferences fragment where the user can specify their preferences & settings for the following:

* Fall Detection Enable
* GPS Boundary Existing
* Emergency Contact Phone Number
* Notification Timer
* Location Update Frequency

The preference fragment automatically implements handling of shared preferences so when a user changes a preference it is automatically changed in the applications *SharedPreference* file. This class automatically loads shared preferences based on the user profile type (i.e. Default, Teenager, Adult or Elderly).

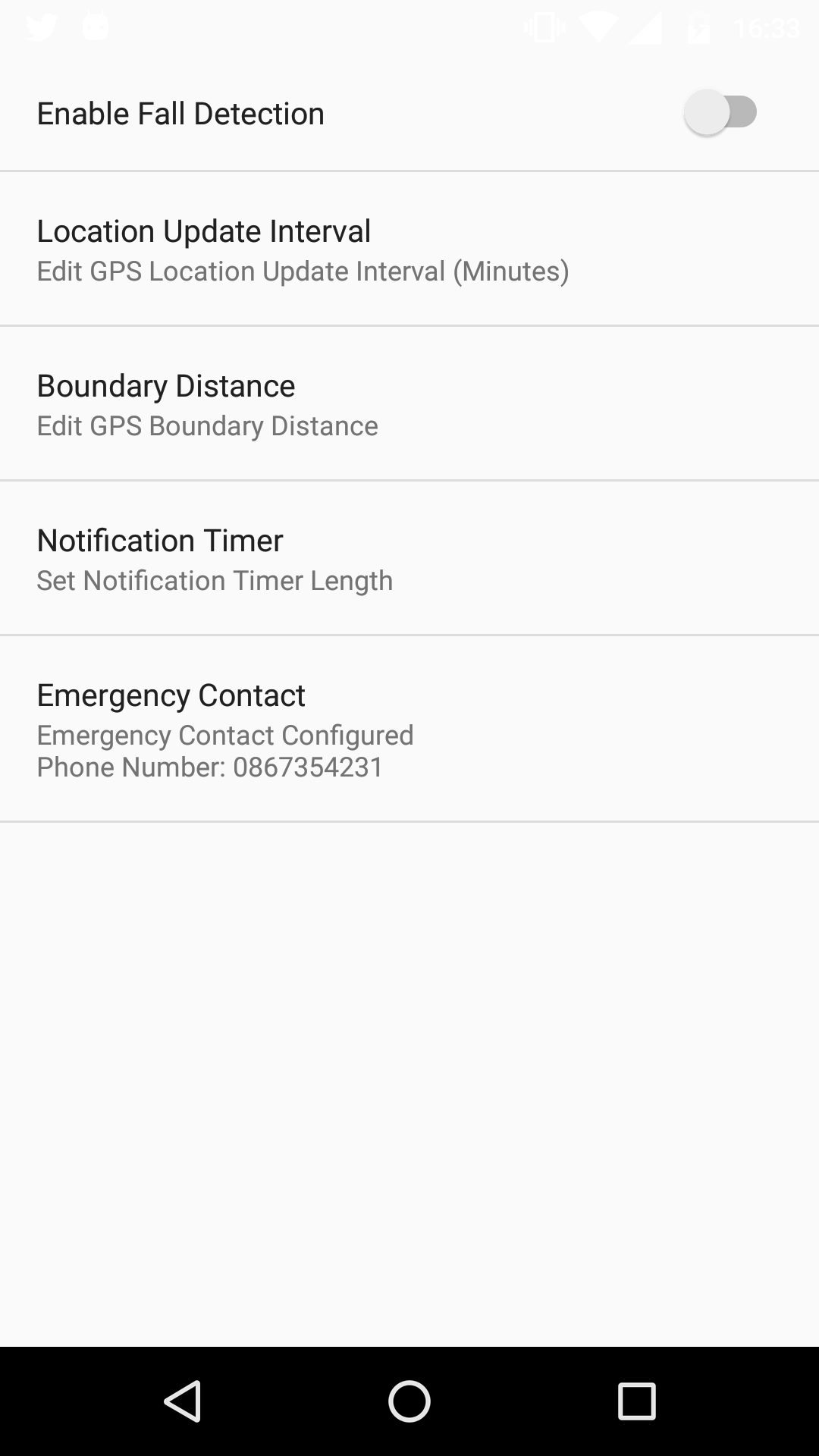


Figure 3‑1-10 Preferences

## 3.1.2 Fragment Module

## Macintosh HD:Users:matthew:Desktop:Screenshots:Screen Shot 2017-03-29 at 16.00.15.png

Figure 3-1-11 Fragment Module

The fragment module of the application consists of all of the fragment classes that formulate the main UI features of the application. These fragments and their interactions are managed by the applications activities.

### HomeFragment

The applications *HomeFragment* class is the first fragment loaded upon successful user login. This class queries the permission status of the necessary device features that the application needs access to. On the occasion that the user has not yet granted the necessary permissions to the application this activity will request the user to grant permissions.

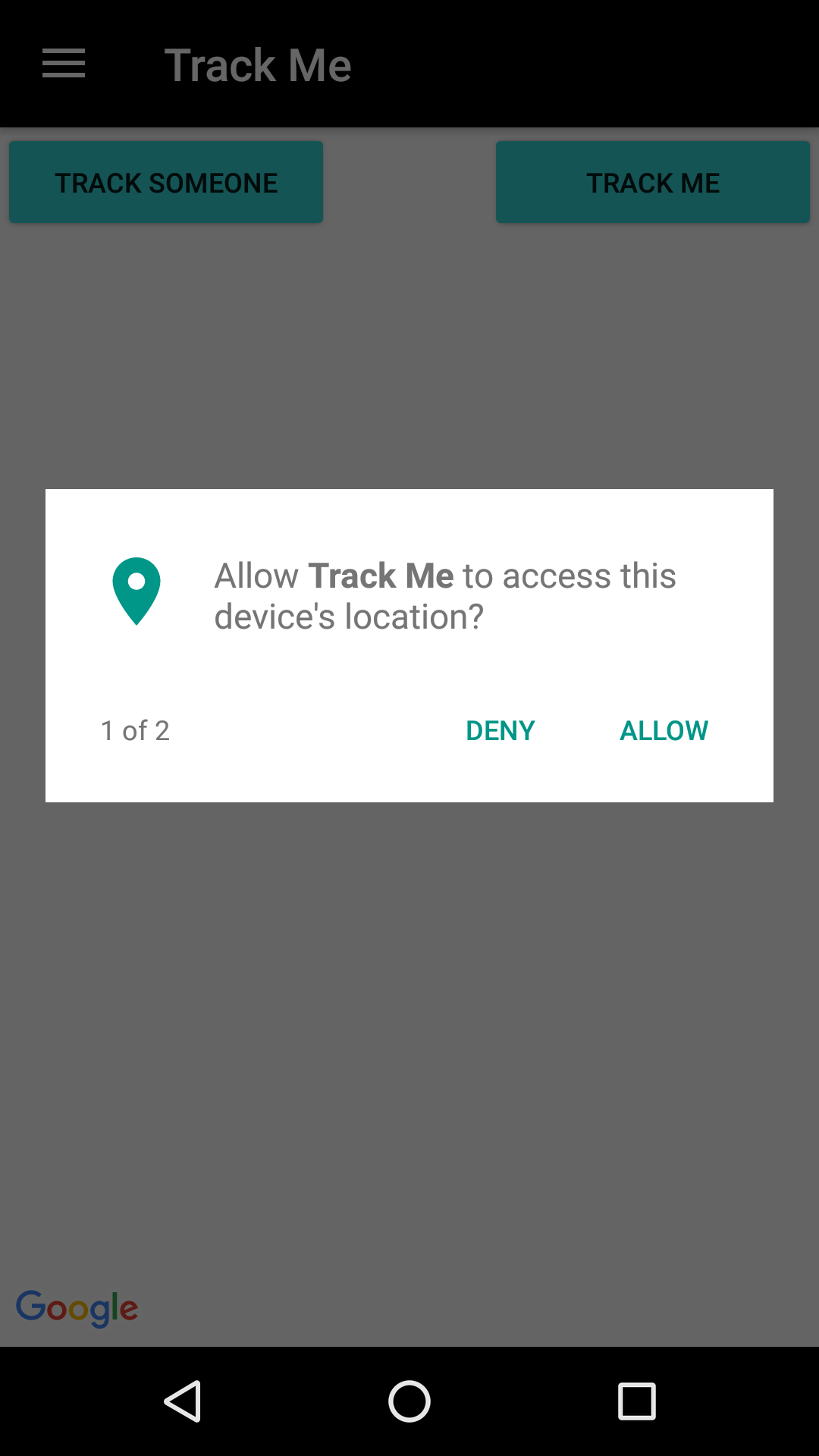
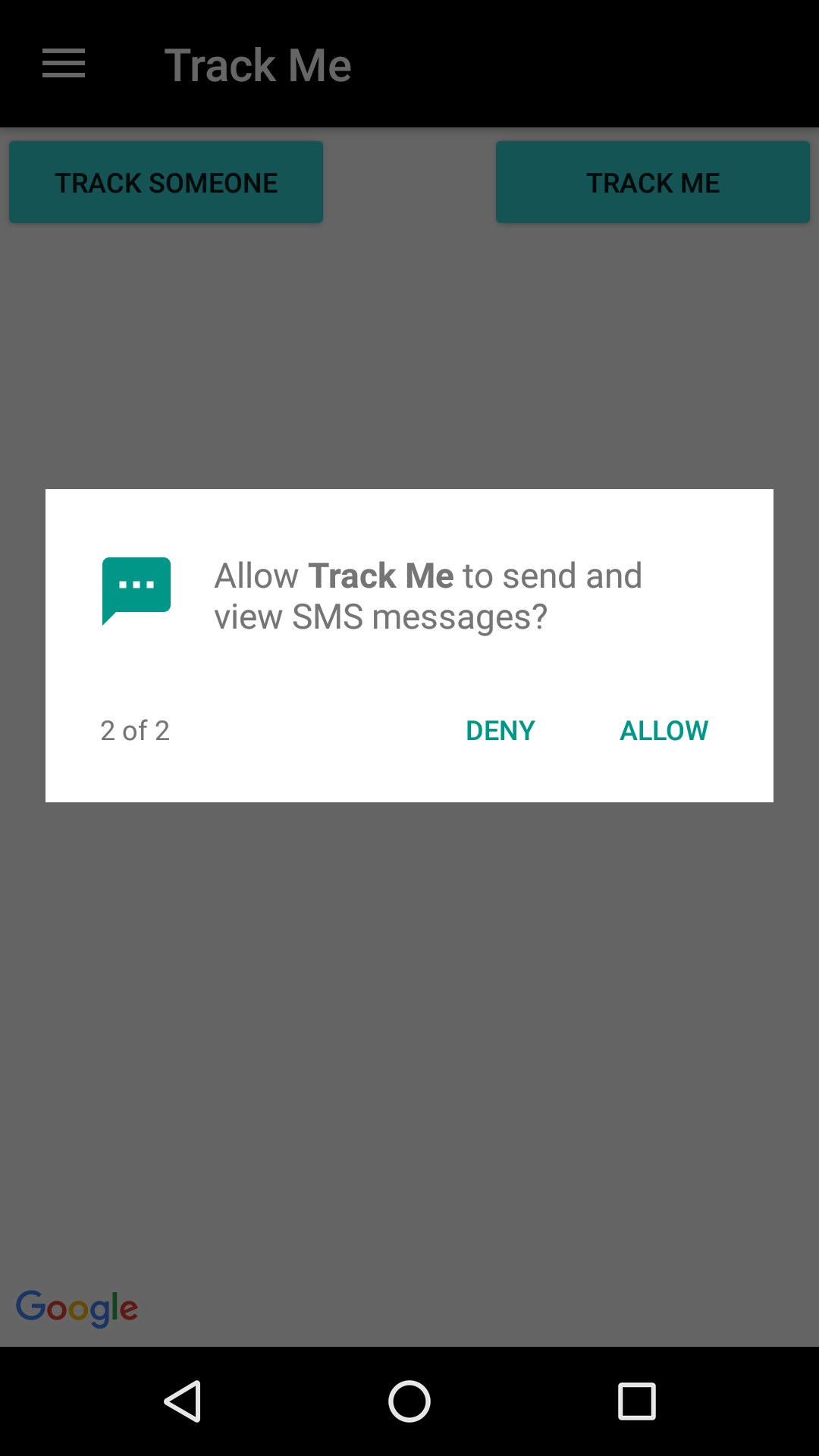
 

Figure 3‑1-12 Location Permission Figure 3‑1-13 SMS Permission

This fragment hosts a Google Maps API *SupportMapFragment* that displays the current user location. If the GPS tracking service is running it will broadcast an intent that is received by this fragment and the UI is updated to display a marker at the latest user location.



Figure 3‑1-14 HomeFragment

This fragment provides buttons that allow the user to start/stop the GPS tracking service, the service that tracks their location, and to allow the user to track another users latest location update to the cloud server by entering their username.

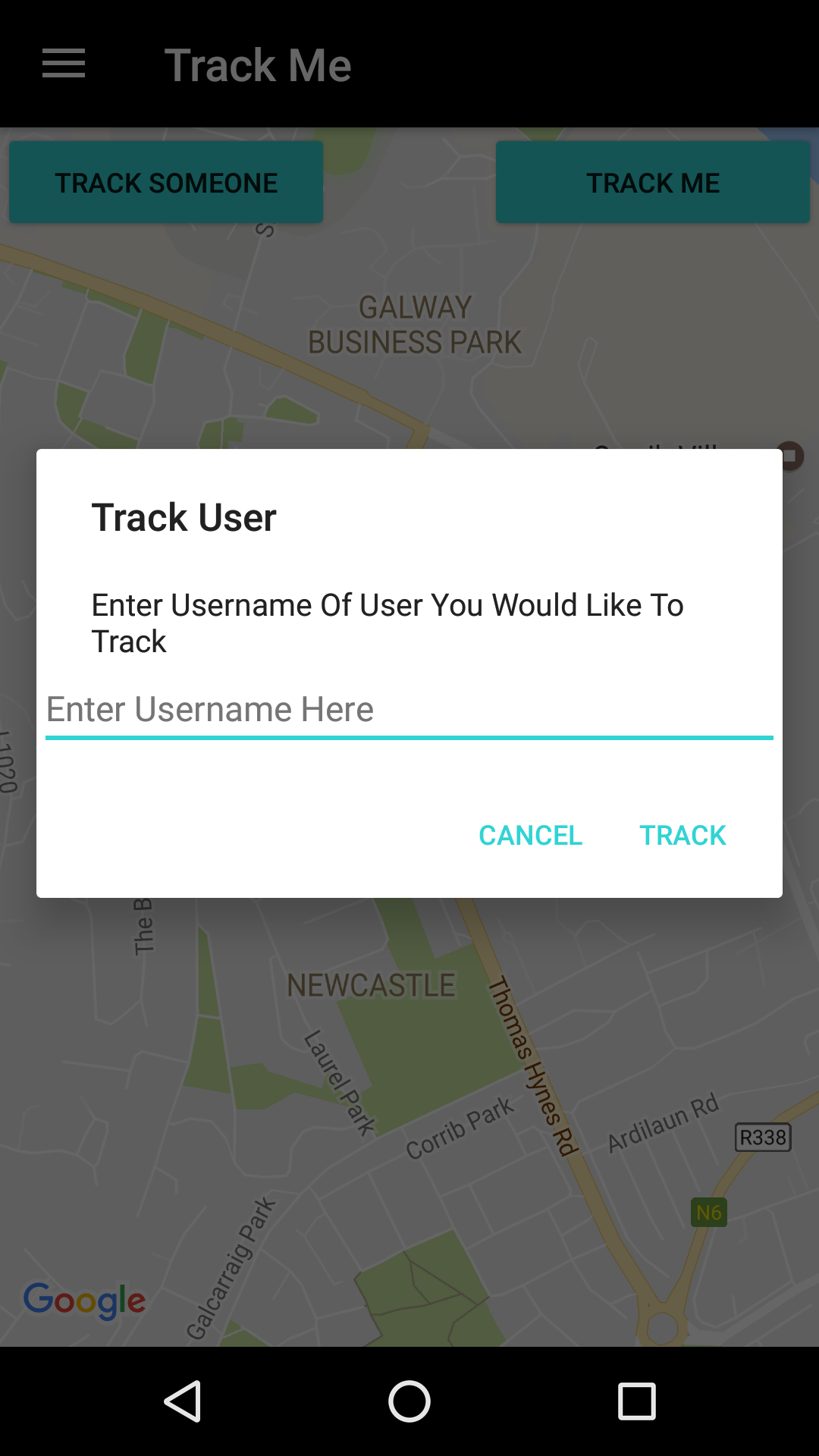


Figure 3‑1-15 Tracking A User

### TrackUserFragment

Upon the successful request for the latest location update of a user in the *HomeFragment* the *TrackUserFragment* is opened and it displays the users latest location and the time at which that location update was posted to the backend server. The location update is retrieved using the Cloud Database Handler class.



Figure 3‑1-16 Remotely Viewing User Location

### RegisterFragment

The *RegisterFragment* class allows the user to create a TrackMe account and post their details to the TrackMe backend server.

This fragment allows the user to enter the following details:

* Name
* Email
* Username
* Phone Number
* Password + Confirmation
* Profile Type

The fragment implements a Volley post request and posts the users details server side database interaction application that in turn posts the data to the TrackMe backend server. This fragment runs error some checking making sure that the fundamental profile details are entered by the user (i.e. Name, Email, Username, Password). If the user has not entered all mandatory details the application will prompt the user to enter the missing information. [1]



Figure 3‑1-17 Register Fragment UI

### LoginFragment

The *LoginFragment* class allows the user to log in to their TrackMe account. This fragment uses a volley post request to post the login details entered by the user to the server side database interaction application.

Since the application requires Internet access to function at it’s highest capacity & connect to the backend cloud database this fragment will only allow a user to login if there’s an Internet connection available on their device. [1]

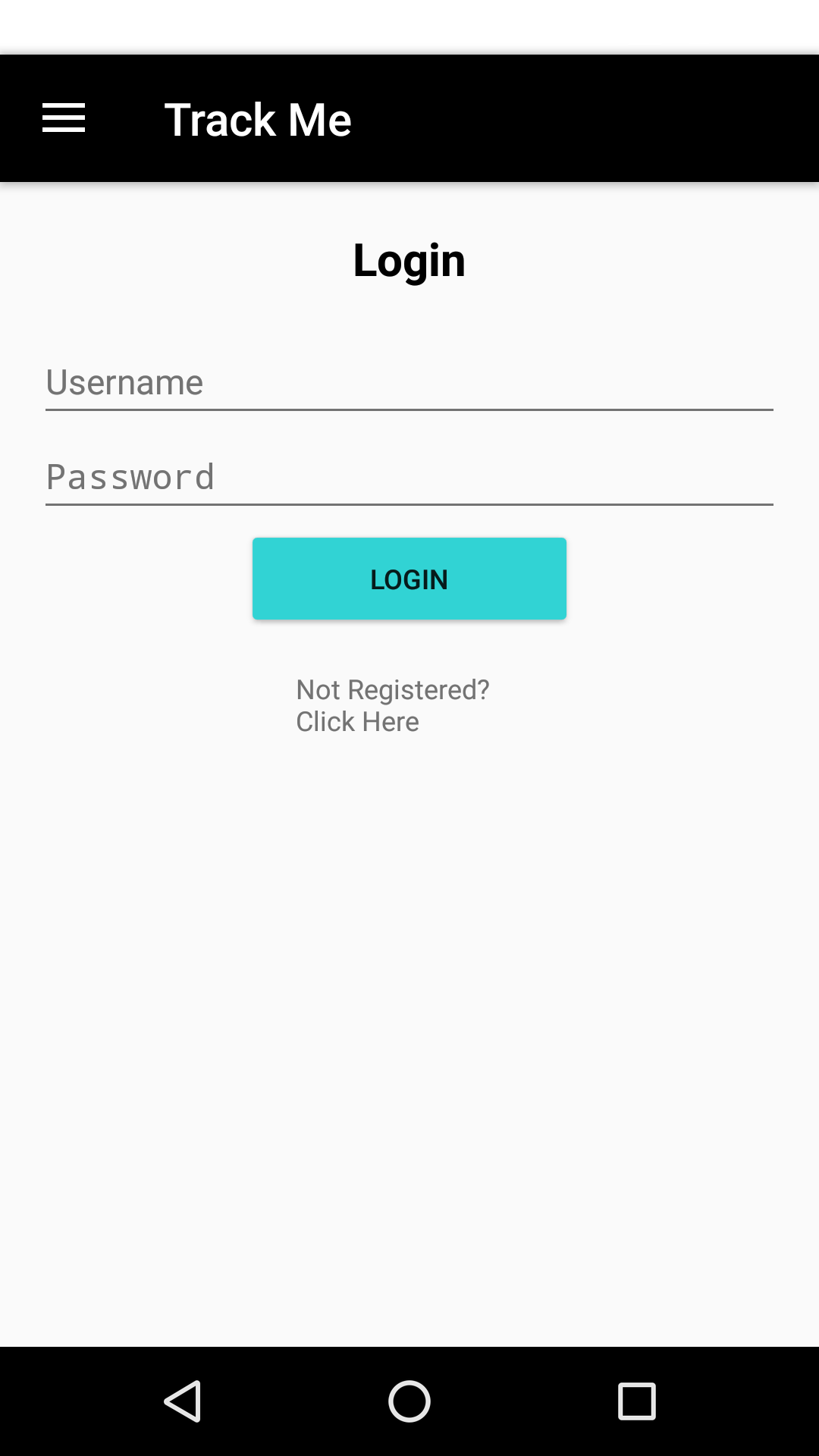


Figure 3‑1-18 Login Fragment UI

### AboutFragment

This fragment displays some information about the application to the user. It provides information such as the goals of the application and the applications’ features.

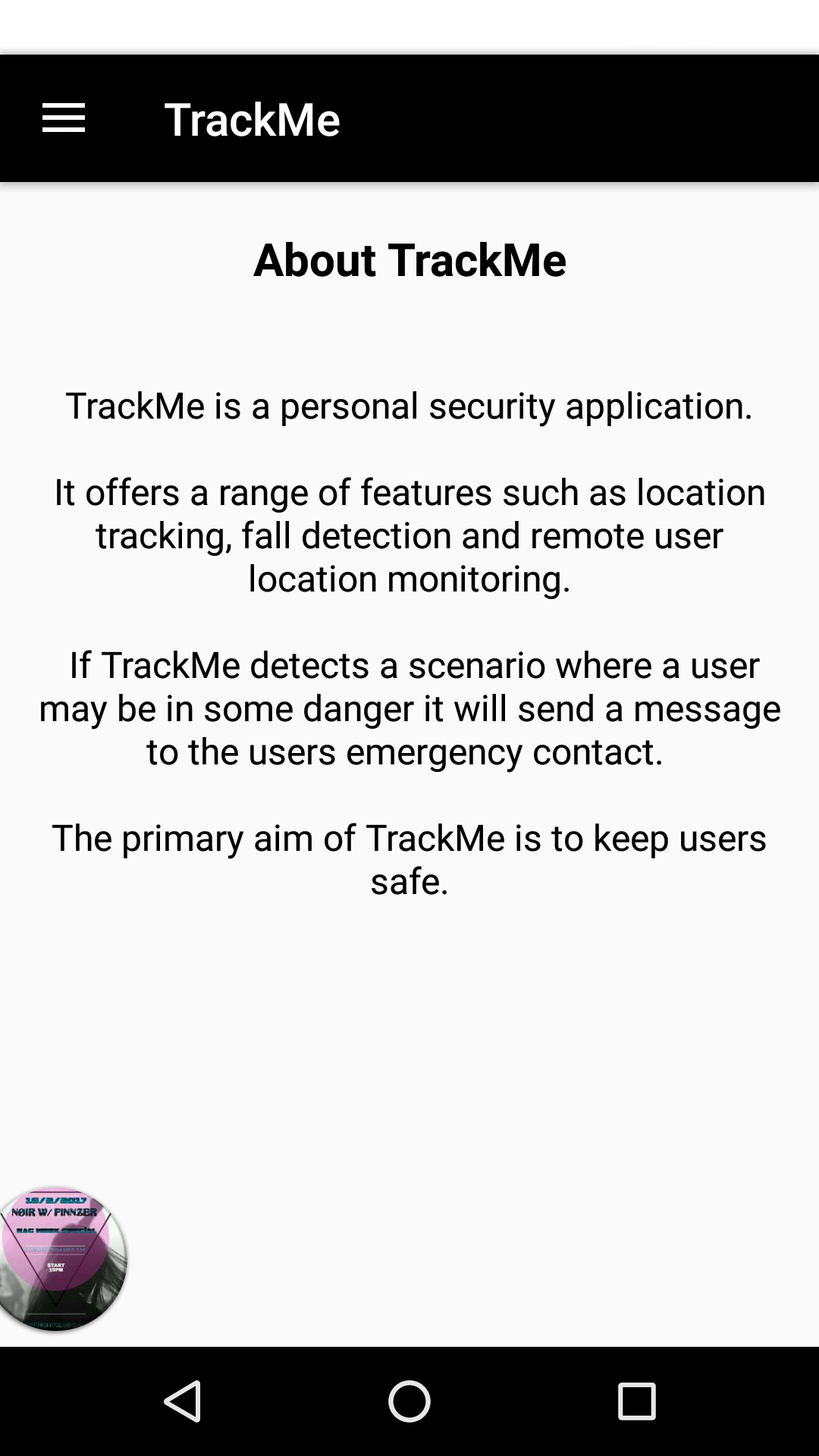


Figure 3‑1-19 AboutFragment

### HelpFragment

This fragment displays some information about the application settings.

It also offers configuration guidelines for these user settings.

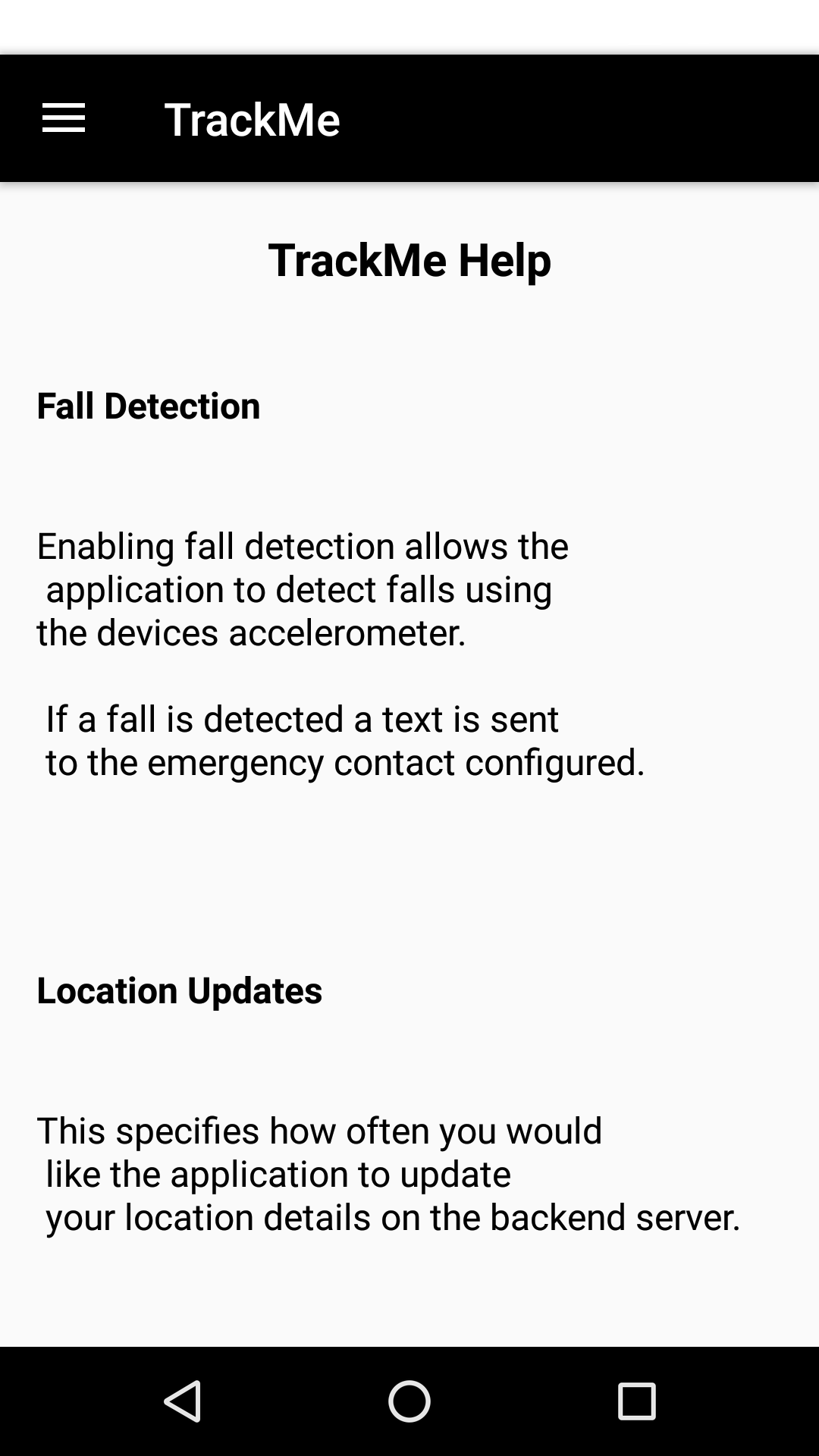


Figure 3‑1-20 HelpFragment

### ProfileFragment

This fragment displays user profile details. These details include items such as a users phone number, email, username, first name and surname.

It has a private inner class that acts as an adapter to display the *hashmap* entries in a *ListView* object. This adapter takes in a *hashmap* as a parameter and puts its’ entries into the *ListView* object specified in the fragments layout xml file.

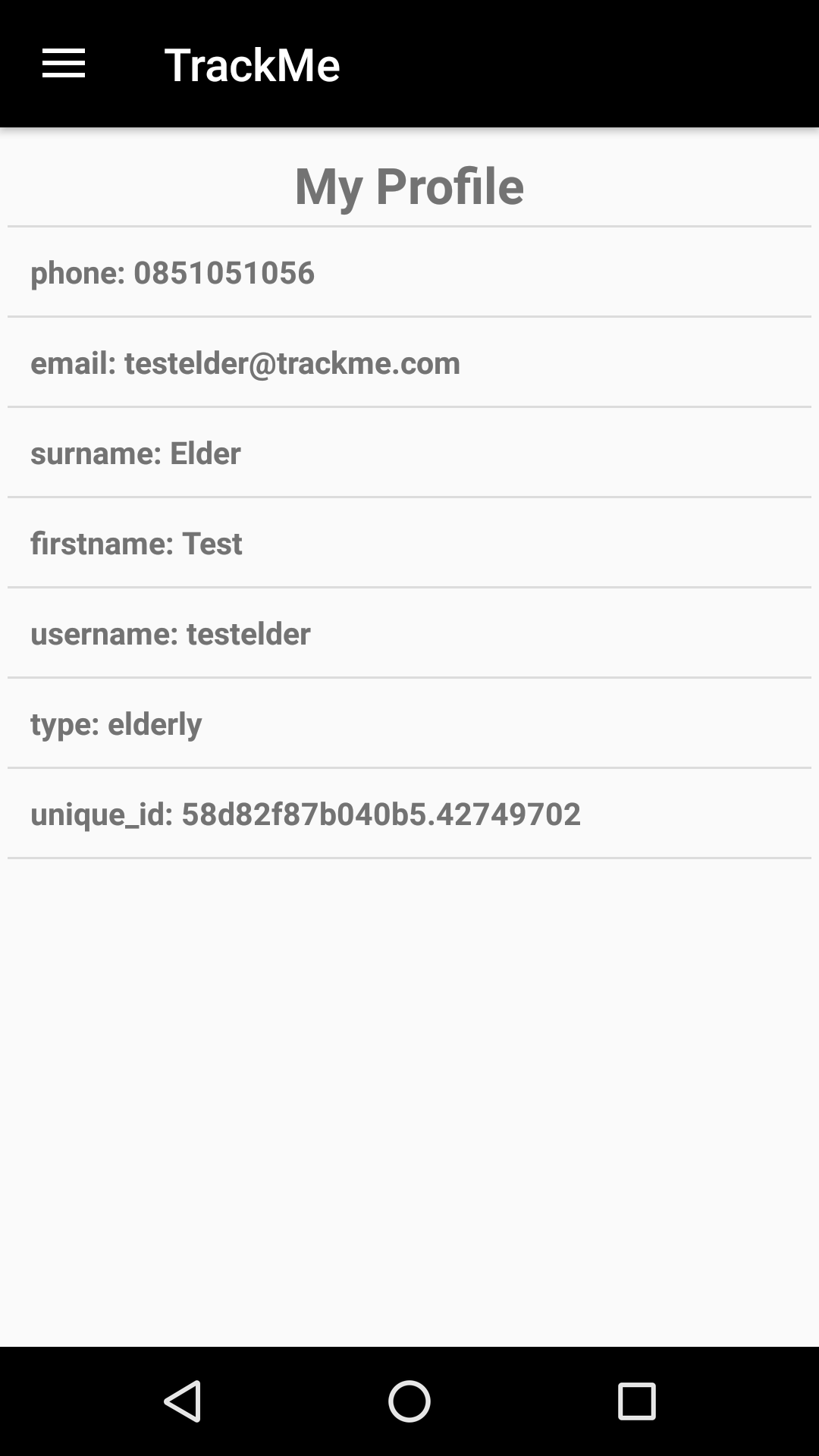


Figure 3-1-21 ProfileFragment

## 3.1.3 Services Module

The services module contains the Service classes that provide the applications fundamental features i.e. GPS Location Tracking and Fall Detection.

## Macintosh HD:Users:matthew:Desktop:Screenshots:Screen Shot 2017-03-25 at 22.19.27.png

Figure 3‑1-22 Services Module

### GPS Service

The GPS service implements an Android Location listener. It is configurable to query location at a user specified interval with the default interval being 60 seconds. The GPS service is set up so that it uses the most accurate location data available at all times.

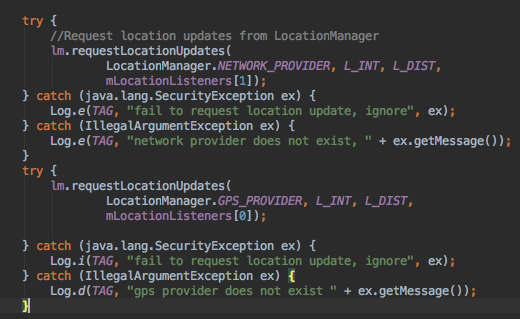


Figure 3‑1-23 GPS Location Service

When this service is started a *LatLng* object is initialized so that the user starting location is saved. This initial location object is used to check if the user has travelled outside the specified boundary from their starting point.

When a location update is obtained this service updates both the local SQLite database and the MySQL cloud database (if the user device is connected to the Internet) with the users latest location details

Upon each location update the algorithm runs verifications to check whether the user has travelled outside of a specified boundary distance from their starting location. If the user has travelled outside the specified boundary then an SMS is sent to their emergency contact with their latest location details.

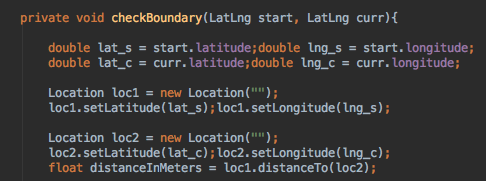


Figure 3‑1-24 GPS Boundary Check

### Fall Detection Service

The fall detection service is a fundamental application feature. This class runs a background servicer that implements a sensor event listener that is used to process readings from the device accelerometer.

Each time a sensor reading is processed this service computes the reading using Euclidean vector mathematics to see if the characteristics of the reading are deemed to be a possible fall. [16]

The criteria specified for a fall are that the accelerometer value had spiked above 25m/s or below 1m/s bearing in mind that at a standstill the computed reading would be approximately 9.8m/s accounting for natural gravitational pull. If these criteria are met then fall detection service deems the readings to constitute a possible fall. A timer is then started and an intent containing the timer’s readings is broadcast. If the timer reaches 0 without being cancelled by the user (through the main activity notification created) then an SMS is sent to the users emergency contact letting them know that a possible fall may have occurred.

The timers default value is set to 30 seconds but is user configurable through the application preferences.

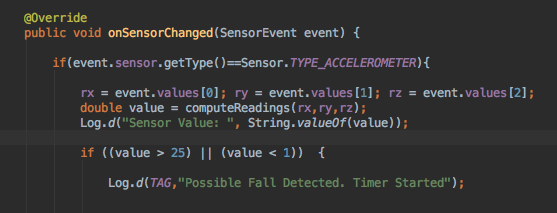


Figure 3‑1-25 Fall Detection

## 3.1.4 Helper Module

The Helper module provides utilities to enhance the functionality of other classes in the application.

## Macintosh HD:Users:matthew:Desktop:Screenshots:Screen Shot 2017-03-25 at 22.19.19.png

Figure 3-1-26 Helper Module

### CloudDBHandler

The *CloudDBHandler* class handles user location data HTTP GET and POST requests to the Cloud database. This includes posting of users location data to the cloud database as well as querying a users latest location update. To achieve this functionality this class uses Android Volley objects to manage GET and POST requests.

### GPSHelper

The *GPSHelper* class provides utilities to help augment the GPS location tracking functionality. These utilities include retrieval of address Strings using a Geocoder (If an internet connection is available on the device). This class also implements a *LocationListener* so that the users location can be retrieved upon app startup

### LocalDBHandler

The *LocalDBHandler* class provides utilities to manage the local devices SQLite database. Upon the initial application install and startup this class creates the local SQLite database and initializes the ‘user’ and ‘location’ tables within this database.

The ‘user’ table contains user profile details for any user that logs into the TrackMe application on that device. These details include name, username, email, profile type, ID and unique ID.

The ‘location’ table contains user location details that are generated when running the applications GPS tracking service. These details include id, username, latitude value, longitude value, unique ID and a timestamp relating to when the location data was obtained.

The devices manages the size of the user ‘location’ table using an SQL trigger that limits the size of the table to 50 rows so that the impact of the table on available device storage is limited.

This class also handles database upgrade so when the application is upgraded any existing database is deleted and a new database containing empty ‘user’ and ‘location’ tables is created.

### MessageHandler

The *MessageHandler* class handles the transmission of SMS messages to the users specified emergency contact. The message to be sent is taken in as an argument as the message content is specific to the scenario detected by the application (i.e. Messages sent for fall detection differ to messages sent for boundary detection).

### SessionManager

The *SessionManager* class handles the creation and termination of user login sessions. This involves the setting of shared preferences for the logged in user (i.e. Setting their profile details as the shared preferences) and the deletion of these shared preferences from device storage upon a user logging out. [2]

This class contains getter and setter methods for all possible shared preferences.

### WidgetHandler

This class defines the functionality of the TrackMe application widget from which the user can start or stop the GPS tracking service as well as being able to start the application or opening the existing instance of the application if it has already been started.

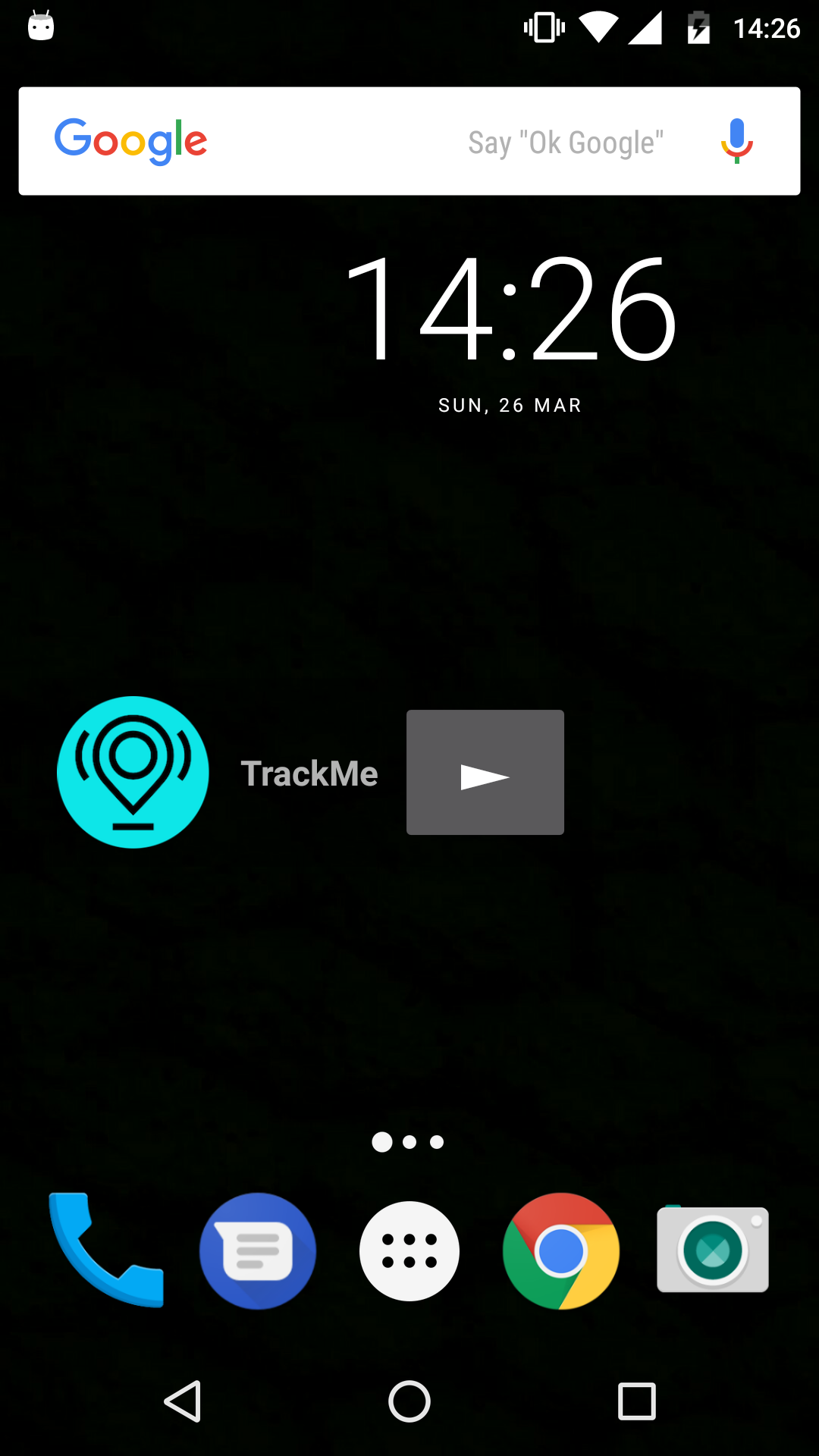


Figure 3‑1-27 TrackMe Widget

## 3.1.5 UI

The applications UI was designed to be user-friendly with ease of use being the primary objective behind the design features. The application allows navigation between pages by utilizing navigation drawers and the Android OS’s native back button. The applications UI theme is a simple theme that uses black, white and turquoise throughout.

## 3.1.6 Miscellaneous

### Application Permissions

This is an overview of the permissions the application requires to function in its’ fullest form:

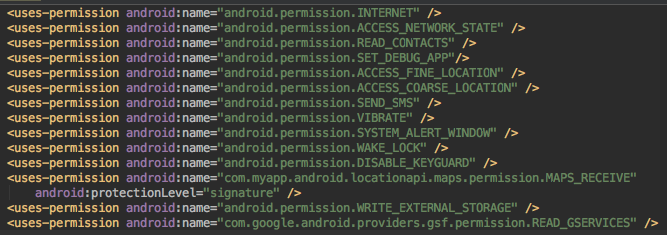


Figure 3‑1-28 Application Permissions

### Application Dependencies

This shows an overview of the libraries that the application is using in its’ implementation:

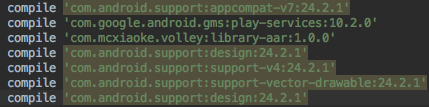


Figure 3‑1-29 Application Dependencies

### SDK Build numbers

This shows the SDK’s that the application is built for ranging from SDK version 15 (Android version 4.0.3) to SDK version 24 (Android version 7.0).

# Macintosh HD:Users:matthew:Desktop:Screenshots:Screen Shot 2017-03-26 at 17.48.13.png

Figure 3‑1-30 Application SDK’s

# 3.2 Server-Side Project

The *TrackMeServer* PHP project, which is hosted on NUI Galway’s Danu6 server, handles the mobile applications interactions with the MySQL cloud database.

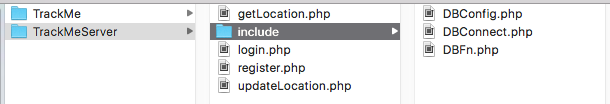


Figure 3‑2‑ Server-Side Project Layout

### DBFn

The *DBFn* class contains all the database functions needed by the mobile application to interact with the cloud database.

This class contains all the SQL queries needed by the application for user registration, login, location updates & location queries.

It provides error checking for user detail duplication such as username, email and password. If any of these database column entries already exists in the user details table, in the cloud database, when a new user is registering an account this class echoes a JSON response to the user device informing them of the duplication details.

This class provides functionality to insert user details into the cloud database.

Macintosh HD:Users:matthew:Desktop:Screenshots:Screen Shot 2017-03-26 at 15.46.40.png

Figure 3‑2‑ Saving User Details

Prior to user details being posted to the cloud database the users password is encrypted using a hash/salt mechanism. This provides a more secure user login implementation meaning that even if the raw database were to be compromised user details would be unobtainable. [1]

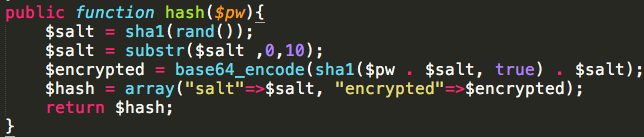


Figure 3‑2‑3 Hash Function

User location details can be posted to the server using this class. This class provides functionality that posts location details to 2 different tables in the cloud database.

One table contains all location updates for every user whereas the other contains one entry for each user that has posted location updates. This single entry per user corresponds to the users latest location update that has been posted to the cloud database.

### DBConfig

This class defines the parameters needed to make a connection to the cloud database. The parameters are the following:

* Database Host
* Database User
* User Password
* Database Name

### DBConnect

This class contains one method that uses the database details configured in the *DBConfig* class to connect to the cloud database.

If the connection is successful this class returns the connection object.

If not, this class echoes response saying that a database connection was unable to be established.

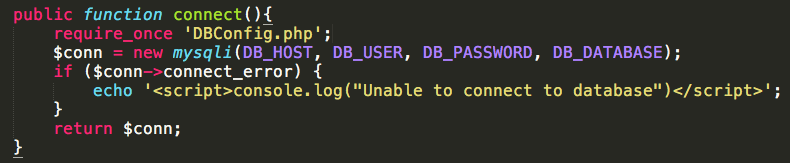


Figure 3‑2‑ Database Connection

### Register

This class handles user registration HTTP POST requests.

It runs some error checking to see if a user is attempting to register using an already occupied email address, username or phone number.

If there isn’t any duplication in the request details then a *DBFn* object method is called to save the user details.

Upon successful registration of user details this class returns a JSON Object containing the users details and if user registration is unsuccessful a JSON object containing some error information is returned to the user device.

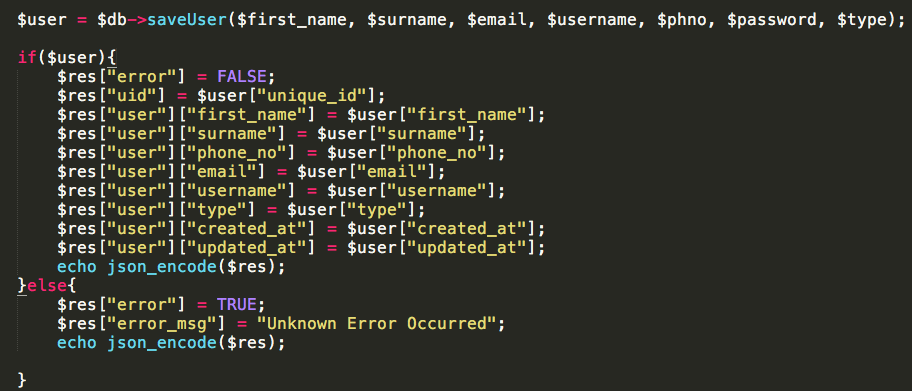


Figure 3‑2‑ Registration

### Login

This class handlers user login HTTP POST requests received from the mobile application. If verifies the user details entered using a *DBFn* object method. If the details are deemed to be correct this class returns the user details in a JSON response object.

If the user enters incorrect details an error is flagged and JSON response containing an error message is sent back to the user device containing the error details.

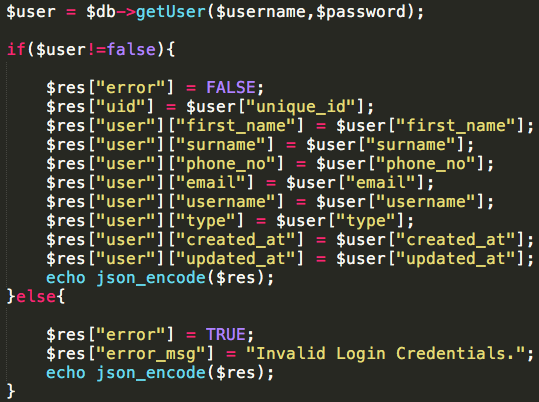


Figure 3-2‑ Login

### UpdateLocation

The *UpdateLocation* class handles user location updates received from the mobile application. It receives these HTTP POST request containing the users own details along with their location details. It proceeds to call the *saveUserLocation* method in the *DBFn* class (functionality of which detailed above) to store the information in the cloud database.

### GetLocation

The *GetLocation* class handles user location queries.

It receives POST requests from the mobile application with the username to query the latest location data for.

This class then proceeds to call the *getUserLocation* method in the *DBFn* class (functionality of which detailed above)

Upon successful retrieval of a users location data the method returns the location details in a JSON response

# 3.3 Database

This section details the schemas used in the MySQL cloud database.

This database consists of 3 tables used by the mobile application:

* *userDetails*
* *usersLocations*
* *latestUserLocation*



3‑3-1 Cloud Database Schemas

The *userDetails* table uses *id* (an auto incrementing integer value) as it’s primary key with *unique\_id*, *email, username and phone\_no* being indexes in the table.

The *usersLocations* table uses *timestamp* as it’s primary key with *unique\_id* being an index in the table.

The *latestUserLocation* table uses the *unique\_id* field as it’s primary key with *username* acting as its index.

4. Testing & Evaluation

# 4.1 Testing

This section gives an overview of the testing completed throughout the course of the applications development. The majority of the testing was carried out on a Motorola G4 Plus smartphone device running Android version 6.0.1 (API 23). Other devices used to test the application were an LG G4, a OnePlus Three, a Sony Xperia Z3 and the Android Studio IDE’s inbuilt emulator. All devices used to test were running Android versions between 4.0.3 and 7.0.

## Testing Of GPS Tracking Algorithm

To test the GPS location algorithms’ accuracy I started the tracking algorithms in multiple known locations where the address was known to me so I could assess whether or not the location address obtained from the Geocoder was accurate.

I found the location data received to be very accurate as around 80% of the time the exact address location was obtained (accuracy to the detail of the exact house number) and the rest of the time the location data obtained was very close to the real location (Usually one house number, or the equivalent, away from the actual location).

During the testing process of the GPS boundary algorithm I used various different boundary settings. Each time I started the GPS tracking algorithm when I was in the same location, I would travel slightly beyond the approximate distance of the boundary variable I had set and then test whether or not an ‘emergency’ message was sent to the emergency contact that I had defined. The boundary tracking always proved to be very accurate with the only real issue arising being the permissions needed to send a background SMS, Once this was rectified the boundary tracking and SMS sending worked flawlessly.

The reliability of the algorithm was tested by running several instances of the application, with the GPS Tracking service running, over a long period of time. It was observed whether ach of these instances tested location data was uploaded to the cloud database as expected. This never proved to be an issue.

## Testing Of Fall Detection Algorithm

The fall detection algorithm was tested and tuned over the period of a number of weeks. The application was reconfigured and tuned several times with different characteristics defined that would indicate a possible fall. Each time it was reconfigured I simulated a number of accelerometer spikes that I would have either classified as falls or not. Throughout the course of this testing I was able to identify the most accurate parameters for fall detection so that the algorithm would be slightly oversensitive. I decided to implement the algorithm in this way as personal security is the fundamental aim of this application and because of that I would rather detect a false positive rather than a false negative.

# 4.2 User Feedback

Throughout the course of the development process I deployed the application to a number of my friends and family’s Android devices.

The application was deployed to user devices in various stages of development to test a range of the applications features such as fall detection, GPS tracking as well as UI interactions and compatibility with various devices.

The versions of the application deployed to user devices ranged from early version with very little background functionality, so to allow users to test the application UI and login/registration features of the application, to later versions of the application deployed to user devices to allow them to test the accuracy of the fall detection and GPS tracking.

User feedback proved to be a very effective method of discovering application functionality based bugs and changes that needed to be implemented. This was due to the fact new users of the application often carried out variating sequences of interaction with the app that I had not anticipated.

This led to the discovery of several bugs in the applications underlying functionality.

Some bugs identified by users are as follows:

1. Email duplication upon registration let to error and inability for application to continue.
2. Opening of application without location services turned on led to application crash.
3. Emergency contact configuration was not saved upon re-opening of the application.
4. No functionality to check for duplication of username of phone number was present in earlier versions of the application.

I also adopted user feedback as my main source of UI testing as I felt that errors and bad practice in the applications UI would be very easily picked up by first time users due to their impartial and newly-formulated perspective of the application. This method helped identify several application UI bugs that I had not previously observed as well as suggestions to improve that applications flow and ease of use.

This feedback led to the following changes in the application:

1. Change to using navigation drawers to navigate through application.
2. Placement of *TrackMe* and *Track Someone* button at the top of the *HomeFragment* UI.
3. The decision to include *HelpFragment* and *AboutFragment* in the application so users would have access to guidelines for application configurations as well as being able to see a summary of the features the application offered.

Overall I found user feedback to be a very useful and productive method of application testing.

I found it useful for several reasons such as:

1. New users offered a fresh perspective on the application.
2. New users oftentimes carried out interactions of the application that I had not considered of previously used leading to the discovery of bugs or the opportunity to add new features to the application.
3. New users often used the application with device settings configurations that I had not considered or tested leading to the discovery of bugs.

# 4.3 Functional Requirements Review

This section reviews the functional requirements set out in the original Software Requirements Specification for the project. It details whether or not the requirements were completed and if not the reasoning or rationale behind the decision or constraint.

## Core Requirements

***FR1 - User Registration***

*Complete As Defined.*

The application allows for users to register an account for the TrackMe application.

***FR2 - User Profile***

*Complete As Defined.*

The application allows the user to configure profile details and choose from various default profile types.

***FR3 - Location Tracking***

*Complete As Defined.*

The application has the ability to track a users location at all times given that there is either network or internet connection available on the user device.

***FR4 - TrackMe (Request To Be Tracked)***

*Incomplete As Defined.*

The application sends an SMS to the users defined emergency contact if a scenario is detected where the user might be in danger. This SMS contains information such as their username and latest location address. The username can be used to see a visual representation of the latest location of the user.

***FR5 - Fall Detection***

*Complete As Defined.*

The application has the ability to detect a user fall.

This service can also be enabled or disabled depending on user preference.

***FR6 - Messaging Settings***

*Incomplete as Defined*

The application has been defined so that it uses SMS as it’s messaging service as it is the only means of sending a message that doesn’t require any user interaction which was a fundamental requirement of the applications messaging feature.

***FR7 - Contacts***

*Incomplete As Defined.*

The application allows the selection of an emergency contact that is sent an SMS upon the detection an *emergency* situation.

***FR8 - Help Menu***

*Complete As Defined*

The user is given the ability to access guidelines for configuring the application.

***FR9 - Push Notifications***

*Complete As Defined*

The ability to show push notifications on the Android OS home screen in the case of a fall being detected.

## Additional Requirements

***FR10 - Social Media Account Integration***

*Incomplete as defined*

Due to time constraints and a change in direction this wasn’t seen as a fundamental requirement as social media accounts wouldn’t be used as a messaging service in the application.

***FR11 - DB Management Web Service***

*Incomplete As Defined*

Development of a web service to administer the DB was not undertaken due to time constraints and requirement prioritization.

***FR12 - Real Time Data Analytics***

*Incomplete As Defined*

Due to time constraints I was unable to complete the implementation of a backend web service that ran data analytics on the user location data uploaded to the backend server.

# 4.4 Comparison With Similar Applications

The TrackMe performs admirably in terms of available features in comparison to the other personal security applications researched as part of this project.

The feature matrix below, that includes similar applications, highlights that TrackMe provides many of the same fundamental security features that are provided by professional subscription based services. TrackMe also has the added ability of fall detection that gives it another dimension in comparison to other similar products.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Feature/Application** | **TrackMe** | **Emergensee** | **BSafe** | **FamilyLocator** | **DigitalLife** | **SafeTrek** |
| **Remote Monitoring** | ✔ | ✔ | ✔ | ✔ | ✔ |  |
| **Fall Detection** | ✔ |  |  |  |  |  |
| **Emergency Contacts** | ✔ | ✔ | ✔ | ✔ | ✔ |  |
| **Background SMS Transmission** | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| **Backend Server Enhancement** | ✔ |  |  |  | ✔ |  |
| **Audio & Video Streaming** |  | ✔ |  |  |  |  |
| **Fake Call Triggers** |  |  | ✔ |  |  |  |
| **Location Boundary Specification** | ✔ |  |  | ✔ |  |  |
| **Alert Countdown Timer** | ✔ | ✔ | ✔ |  | ✔ |  |

Figure 4‑4‑ Comparison Matrix

5. Conclusion

# 5.1 Project Outcome

This project meets the primary goals of developing a smartphone application that is used to aid the personal security of vulnerable adults and children. The application enables periodic device check-in, remote monitoring of device location, detection of lack of movement (through the fall detection algorithm) and has the ability to detect user movement outside specified boundaries depending on the requirements of the individual being monitored. The smartphone application in enhanced by a backend server that serves that purposes of making data available to other relevant users of the application that can query the data.

Overall I am happy with the experience I have gained in Android Development that enhanced my ability to built smartphone application as well as enhancing my competency in Java programming.

The completion of the project also gave my good experience in using PHP and learning about the fundamentals database administration.

I believe that the skills I’ve learned due to competing this project will be of great benefit to me in my future career.

# 5.2 Future Work

The smartphone application that I have developed has met its main primary goals but I have identified some areas in which I could improve or enhance the implementation of the project as a whole.

## Profile Editing

I would like to implement a means for users to edit their profile details.

This would involve making changes to both the mobile application and PHP database interaction project. Addition of this feature would make the application more well rounded and professional.

## Data Analytics

Another feature I would like to implement is the harnessing of all of the dormant user location data stored in the MySQL database. If statistical data was generated about this information there could be a lot of data discovered such as possible ‘dangerous areas’ or areas in which falls are more likely to occur

## Password Retrieval/Change

A feature that I think would greatly benefit the usability as well as the longevity of the mobile application would be a means for the user to retrieve a forgotten password using their email and/or username. I think this feature would add a lot of value to the application as it would increase the user retention.

References

[1] androidhive. 2017. Android Login and Registration with PHP, MySQL and SQLite. [ONLINE] Available at: <http://www.androidhive.info/2012/01/android-login-and-registration-with-php-mysql-and-sqlite/>.

[2] androidhive. 2017. Android User Session Management using Shared Preferences. [ONLINE] Available at: <http://www.androidhive.info/2012/08/android-session-management-using-shared-preferences/>.

[3] bSafe. 2017. bSafe You - The End Of Worry. [ONLINE] Available at: [http://getbsafe.com](http://getbsafe.com/).

[4] The Journal. 2017. Ireland’s missing people: The numbers behind the

heartbreak. [ONLINE] Available at: <http://www.thejournal.ie/missing-persons-ireland-statistics-facts-numbers-2015-2450180-Nov2015/>. [Accessed 23 March 2017].

[5] Ireland's Leading Falls Screening, Prevention & Monitoring Service. 2017. Ireland's Leading Falls Screening, Prevention & Monitoring Service. [ONLINE] Available at: [http://www.falls.ie](http://www.falls.ie/).

[6] Thousands of elderly injured in falls . 2017. Thousands of elderly injured in falls . [ONLINE] Available at: <http://www.irishhealth.com/article.html?id=18209>.

[7] Silicon Republic. 2017. Ireland is an Android haven as iOS suffers drop. [ONLINE] Available at: <https://www.siliconrepublic.com/gear/ireland-android-v-ios-market-smartphones>.

[8] Silicon Republic. 2017. 70pc of Irish population now owns a smartphone (infographic). [ONLINE] Available at: <https://www.siliconrepublic.com/comms/tech-nation-irish-population-smartphone>.

[9] Statista. 2017. Smartphones industry: Statistics & Facts | Statista. [ONLINE] Available at: <https://www.statista.com/topics/840/smartphones/>.

[10] Transmitting Network Data Using Volley | Android Developers . 2017. Transmitting Network Data Using Volley | Android Developers . [ONLINE] Available at: <https://developer.android.com/training/volley/index.html>.

[11] Sending a Simple Request | Android Developers . 2017. Sending a Simple Request | Android Developers . [ONLINE] Available at: <https://developer.android.com/training/volley/simple.html>.

[12] Digital Life Personal Security - Android Apps on Google Play. 2017. Digital Life Personal Security - Android Apps on Google Play. [ONLINE] Available at: <https://play.google.com/store/apps/details?id=com.att.digitallife.personalsecurity&hl=en>.

[13] EmergenSee - Personal Safety - Android Apps on Google Play. 2017. EmergenSee - Personal Safety - Android Apps on Google Play. [ONLINE] Available at: <https://play.google.com/store/apps/details?id=com.emergensee&hl=en>.

[14] Family Locator - GPS Tracker Android Apps on Google Play. 2017. Family Locator - GPS Tracker Android Apps on Google Play. [ONLINE] Available at: <https://play.google.com/store/apps/details?id=com.life360.android.safetymapd&referrer=utm_source%3DAppszoom%26utm_medium%3DAppszoom%26utm_campaign%3DAppszoom>.

[15] SafeTrek - Hold Until Safe  - Android Apps on Google Play. 2017. SafeTrek - Hold Until Safe  - Android Apps on Google Play. [ONLINE] Available at: <https://play.google.com/store/apps/details?id=com.safetrekapp.safetrek&hl=en>

[16] . 2017. Android Open Source - Hardware accelerometer Fall-detection-in-Android. [ONLINE] Available at: <http://www.java2s.com/Open-Source/Android_Free_Code/Hardware/accelerometer/BharadwajS_Fall_detection_in_Android.htm>.

Appendices

# Table of Figures

Figure 2-2‑1 bSafe Application 19

Figure 2-2‑2 EmergenSee Application 21

Figure 2-2‑3 DigitalLife Application 21

Figure 2-2‑4 Family Locator Application 22

Figure 3‑1‑1 Android Platform Versions 24

Figure 3‑1‑2 Application Module 25

Figure 3‑1‑3 AppConfig.java 25

Figure 3‑1‑4 Activity Module 26

Figure 3‑1‑5 Splash Screen 27

Figure 3‑1‑6 Location Services Request 27

Figure 3‑1‑7 StartupActivity Navigation Drawer 27

Figure 3‑1‑8 MainActivity Navigation Drawer 28

Figure 3‑1‑9 Fall Detection Notification 29

Figure 3‑1‑10 Preferences 30

Figure 3‑1‑11 Fragment Module 30

Figure 3‑1‑12 Location Permission 31

Figure 3‑1‑13 SMS Permission 31

Figure 3‑1‑14 HomeFragment 32

Figure 3‑1‑15 Tracking A User 32

Figure 3‑1‑16 Remotely Viewing User Location 33

Figure 3‑1‑17 Register Fragment UI 34

Figure 3‑1‑18 Login Fragment UI 35

Figure 3‑1‑19 AboutFragment 35

Figure 3‑1‑20 HelpFragment 36

Figure 3‑1‑21 ProfileFragment 37

Figure 3‑1‑22 Services Module 37

Figure 3‑1‑23 GPS Location Service 38

Figure 3‑1‑24 GPS Boundary Check 39

Figure 3‑1‑25 Fall Detection 40

Figure 3‑1‑26 Helper Module 40

Figure 3‑1‑27 TrackMe Widget 42

Figure 3‑1‑28 Application Permissions 43

Figure 3‑1‑29 Application Dependencies 43

Figure 3‑1‑30 Application SDK's 44

Figure 3‑2‑1 Server-Side Project Layout 44

Figure 3‑2‑2 Saving User Details 45

Figure 3‑2‑3 Hash Function 45

Figure 3‑2‑4 Registration 47

Figure 3‑2‑5 Login 47

Figure 3‑3‑1 Cloud Database Schemas 49

Figure 4‑4‑1 Comparison Matrix 56