SECURED IN CAMPUS RIDE-SHARING

CHOO HONG YEE

179435

Project report submitted in partial fulfillment of the requirement for the Degree of Bachelor of Engineering (Computer and Communication Systems), Faculty of Engineering, Universiti Putra Malaysia

2018

APPROVAL SHEET

The project report entitled **Secured In Campus Ride Sharing** prepared and submitted by **Choo Hong Yee** in partial fulfilment of the requirement for the Degree of Bachelor of (Computer and Communication Systems) Engineering is hereby accepted.

Approved by:

Puan Siti Mariam Binti Shafie @ Musa,
Supervisor ....
Department of Computer and Communication Systems
Engineering
Faculty of Engineering
Universiti Putra Malaysia

Y. Bhg. Profesor Dr. M. Iqbal Bin Saripan,
Examiner 1 ....
Department of Computer and Communication Systems
Engineering
Faculty of Engineering
Universiti Putra Malaysia

Prof. Madya Dr. Syed Abdul Rahman Al-haddad Bin Syed Mohamed,
Examiner 2 ....
Department of Computer and Communication Systems
Engineering
Faculty of Engineering
Universiti Putra Malaysia

Date Approved: **6 June 2018**

DECLARATION

I hereby declare that the project report is based on my original work except for quotations and citations which have been duly acknowledged. I authorize Universiti Putra Malaysia to lend this thesis to other institutions or individuals for the purpose of scholarly research.

....

CHOO HONG YEE

Date: 6 June 2018

ACKNOWLEDGEMENTS

First and foremost, I would like to express my utmost appreciation and gratitude to my beloved supervisor, Puan Siti Mariam bt. Shafie@Musa for relentlessly giving me guideline, suggestion, encouragement and support in completing my project. She has enlightened me with many useful ideas and information in writing the thesis. A million thanks given to her in making this project successful.

Besides that, I would like to extend my appreciation to all the lecturers and staffs of Department of Computer and Communication Systems for helping me throughout my project. Their commitments to professionalism has provide me a lot of knowledge about the project. Kind comments and constructive suggestions were given which has directly help me managed to complete this project.

Last but not least, I would like to thank my family and friends for giving unconditional love, care, prayers, encouragement and continuing support during my difficult times.

ABSTRACT

Among the ideas of sharing information, ride-sharing is one of the promising growing trends. It is proven that the technology is currently being implemented as a business idea all around the world. The time taken travelling around the campus is high, and it is an issue as public transport are provided around the campus. The issue worsen when it comes to peak hour travel among faculties and colleges. In UPM Serdang Campus there are 3 units of single-seated MyCOMS. A mobile application has been designed to allow students and staffs of UPM Serdang campus to coordinate and use MyCOMS. However, the application is not deployed currently, and several weakness have been found in the application. This project aims to design and develop a secured in campus ride sharing Android application. Other than that, a prediction of station level demand can also be done based on simulation and real time data. Lastly the system design performance is evaluated. The application is built by using Android Studio. Google Maps API is used to obtained location information of both MyCOMS and user. The data obtained such as location of MyCOMS and user details is stored in Firebase. The application that has been developed is able to locate real time location of MyCOMS, place and manages booking of the vehicle. Fuzzy logic is used to predict the station level demand with simulation of data which includes location of station, temperature of the day, time of travel and population of station. In conclusion, the fully workable Android application is developed for MyCOMS is capable of providing convenience and resolving part of the issues in commuting in campus for UPM students and staff.

ABSTRAK

Di antara idea-idea berkongsi maklumat, perkongsian perjalanan merupakan salah satu trend yang berkembang pesat. Ia dapat dibuktikan bahawa teknologi sedang dilaksanakan sebagai idea perniagaan di seluruh dunia. Masa yang diambil adalah tinggi untuk mengunjung fakulti-fakulti di sekitar kampus, sedangkan pengangkutan awam telah pun disediakan. Isu ini semakin teruk sewaktu 'peak hour' di sekitar kampus. Di Kampus UPM Serdang ada 3 unit MyCOMS. Aplikasi mudah alih telah direka untuk membolehkan pelajar dan kakitangan kampus UPM Serdang untuk menggunakan MyCOMS. Walau bagaimanapun, aplikasi mudah alih itu tidak digunakan pada masa ini, dan beberapa kelemahan telah dijumpai dalam aplikasi tersebut. Projek ini bertujuan untuk merekabentuk dan membangunkan aplikasi Android untuk perkongsian perjalanan kereta di dalam kampus. Selain itu, ramalan permintaan peringkat stesen juga boleh dilakukan berdasarkan simulasi dan data masa nyata. Akhir sekali prestasi reka bentuk sistem dinilai. Aplikasi ini dibina dengan menggunakan Android Studio. API Peta Google digunakan untuk mendapatkan maklumat lokasi kedua-dua MyCOMS dan pengguna. Data yang diperoleh seperti lokasi MyCOMS dan butiran pengguna disimpan dalam Firebase. Aplikasi yang telah dibangunkan dapat mencari lokasi masa nyata MyCOMS, tempat dan menguruskan tempahan kenderaan. Logik kabur digunakan untuk meramalkan permintaan tahap stesen dengan simulasi data yang merangkumi lokasi stesen, suhu hari, waktu perjalanan dan populasi stesen. Kesimpulannya, aplikasi Android ini boleh dilaksanakan sepenuhnya untuk MyCOMS kerana ia mampu menyediakan kemudahan dan menyelesaikan sebahagian daripada isu-isu dalam perjalanan di kampus untuk pelajar dan kakitangan UPM.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

API	Application Programming Interface
AVD	Android Virtual Device
COMS	Chotto Odekake Machimade Suisui
ERD	Entity Relationship Diagram
EV	Electric Vehicle
GPS	Global Positioning System
GSM	Global System for Mobile Communications
IDE	Integrated Development Environment
Kyutech	Kyushu Institute of Technology
OS	Operating System
RAM	Random-access memory
SDK	Android Software Development Kit
SMS	Short Message Services
TABJ	Toyota Auto Body Japan
TABM	Toyota Auto Body Malaysia
UPM	Universiti Putra Malaysia
VR	Virtual Reality

CHAPTER 1

INTRODUCTION

1.1 Project Background

With the rise of social media that promotes communication among personnel over the internet has stimulate the innovation of utilizing the power of connectivity among them. Among the ideas of sharing information, ride-sharing is one of the promising growing trends. It is proven that the technology is currently being implemented as a business idea all around the world.

As the emergence of sharing economy gradually arise in this modern society, the ride-sharing community and industry are further developing into different segment, where it is divided into driver and driverless ride-sharing. The examples of ride-sharing with driver are Uber and Grab, and the example for driverless ride-sharing would be bike-sharing such as oBike and MoBike. With these real world implementation of the concept of ride-sharing, the sharing economy is proven successful, as it builds up a brand new wave of investment into technology industry.

There is difference of ride-sharing, which could be further categorized into model that supplies both drivers and vehicles, or model that supplies only transportation itself. The example from industries for the model are stated above. The method in running both system are different. Navigation of the vehicles are different if the supplied vehicle has no driver. For this project, the model that only supplies transportation only and without driver, will be taken in as the main case of investigation.

With the model being stated, certain function must be carried out so that the coordination and scheduling of the usage time by user could be synchronized, and functional at the same time. By arranging the sequence in manual, everything could be done easily but complicated and time consuming at the same time. With that, certain intelligent system must be combined into the model to enhance the automation so that manual operation can be replaced. Artificial technologies are one of the exciting technology that is being developed and paid attention on, so that it is able to be placed into the system in making it more intelligent, as well as automating certain tasks which reduces manpower and saves time in general.

In relation to the case in UPM Serdang campus, MyCOMS is a type of electric vehicle that only have one-seated, which provides difficulty in terms of mobility. When the car is being driven to some other place, it is hard to get the car back to original place, unless there is someone who is responsible in returning the vehicles to the desired location at the start. The daily demand of the electric vehicle differs with stations, as the number of student and staff that has different schedule to go to every places in the campus. Thus, the deployment of the vehicle is crucial, as it is best to place the vehicle at stations which has the most traffic at the start of the day, to ensure that it serves its purpose in providing mobility to students and staffs. With this it is hope that the overcrowded and the over-capacity issue of the number of passenger for transportation currently provided.

By combining both model and the technology specified above, it is hoped that it can be transformed into a better and more intelligent system that would benefit users, and the administration at the same time.

1.2 Problem Statement

The time taken travelling around the campus is high, and it is an issue as public transport are provided around the campus. With ride sharing among students, it is targeted to solve the over-crowded issue with the limited mode of transportation. Full utilization of networks within the students is expected to generate and create a safe environment and community which makes every ride smooth.

Students, staff and every personnel that is in UPM Serdang campus will face traffic problem, when it comes to peak hour travel, and also certain ad-hoc event and program hosted in the campus. One of the most significant event that adds up to the traffic issue would be annual convocation which will last for a week. And weeks before the convocation, ex-students will be returning to the campus in retrieving their graduation robe and to submit documents. This will even add up to the normal traffic volume. This can be solved if there is a platform that can be used in coordinating vehicles by demand. User can expect the availability of vehicle and book it in advance, and use it when they need it the most.

In UPM Serdang Campus there are 3 units of single-seated MyCOMS. Users must be able to utilize it fully, only through sharing of a pool of same network. The availability and whereabouts of the vehicle must be updated almost instantly to allow everyone in the network to view its status, and decide on the usage of it. Certain limitation to the vehicle must be considered, such as battery level, destination of journey, and usage demand. The vehicle will be useless if there is not enough power to run. The vehicle will have limited source of user if destination of a journey is targeted at remote area of the campus. The vehicle must be placed properly at stations where there is higher demand to serve its purpose.

There is an application developed for booking of the MyCOMS. However, the previous application built for booking of MyCOMS is currently not in service, and was not sufficient in servicing the capacity of students and staff in UPM Serdang campus. The application was also reviewed thoroughly from the framework used to the type of database used in supporting the whole application. It was found out that the application was made in Cordova framework and uses relational database as the backbone. In terms of scaling the numbers of concurrent user of the application, certain modifications must be done in order to increase efficiency and reduce breakdown or failure of the application in delivering the objectives.

1.3 Aim and Objectives

The aim of this project is to develop a fully working secured in campus ride-sharing application.

The objectives of this project are:

1. To design and create an EV ride-sharing application in UPM Serdang campus.
2. To predict station level demand of EV.
3. To evaluate the system design performance.

1.4 Project Scope

The project will be focusing on the development of a ride-sharing Android application. The main mode of transportation which is going to be used in this project would be the Toyota Chotto Odekake Machimade Suisui (COMS), which is later renamed as MyCOMS by Universiti Putra Malaysia (UPM), in hopes to promote green technology in the Serdang Campus. The COMS is a single-seated ultra-compact electric vehicle, which carry the meanings of 'smooth, short rides into town". Three COMS are received, and it represents the collaboration between UPM, Toyota Auto Body Malaysia (TABM), Toyota Auto Body Japan (TABJ) and also Kyushu Institute of Technology (Kyutech), Japan. This project will have high similarities of the model of bike-sharing, as the COMS is a single-seated vehicle, and can only be transported whenever there is driver on it.

The proposed mobile application which specially created for mobile device with platform of Android is expected to build to coordinate and monitor the MyCOMS vehicle. It is targeted to provide efficiency for users to manage the booking of the MyCOMS, and also for the administration team to have full time monitoring of the vehicle whenever it is.

1.5 Report Outline

The thesis consists of five chapters, which includes introduction, literature review, methodology, result and discussion, and conclusion.

- Chapter 1 - Introduction to the background of the involved EV which is MyCOMS, and the targeted ride-sharing android application. Problem statements are defined in this chapter. The aims and main objectives of the project are also discussed in the chapter too. Project scope is declared at the end of the chapter.

- Chapter 2 - Literature review of the related research on the ride-sharing industries, including vehicles with driver, and also vehicles that are driverless. The existing ride-sharing application are studied, compared and discussed in this chapter. Main focus of the project which is the ride-sharing android application is discussed in the chapter.

- Chapter 3 - Methodology used in the overall flow, outline and the process of the project. The related materials and equipment such as hardware and software involved: Android devices, Android Studio IDE and MyCOMS are discussed. The architecture of the android system is further explained in this chapter. The emerging idea of developing this ride-sharing application is further discussed in the same section of the chapter.

- Chapter 4 - The result and discussion which shows the screenshot, and also the performance evaluation of the application. The problem faced throughout the application development and project deployment are discussed in this chapter.

- Chapter 5 - Conclusion of the whole project in the last chapter of the thesis. This chapter will includes the application development, the evaluation findings and also the future works possible for the application are discussed in the chapter.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In this chapter, the case study of the operational model of ride-sharing and bike-sharing application are discussed thoroughly. The existing system of the application is studied and discussed further especially on features, limitations, advantages and disadvantages throughout this chapter.

2.2 Ride Sharing

Public transportation plays a significant role in daily lives of Malaysian citizens as it connects each other to the desired destination. Some of the most common form of public transportation such as bus, train, taxi and others, where one of the mode has occupancy lesser than 40% which is taxi in the form of vehicle. One of the reasons is that the service and cost of hiring a taxi is not proportional as customer feel there is unfairness in it [1]. Monopolized and bad behavior of driver are also the reason for the low using rate of taxi. As compared to the other transportation, taxi carries the highest mobility and flexibility, with no tracks needed as compared to trains, and wider roads for busses, it is simply the best mode of transportation for short distance around a certain area.

But since the occupancy rate of taxi is low, and special infrastructure needed for trains and bus, families with higher purchasing power can afford to buy a vehicle for daily travelling purpose. With that the cars on the road increases gradually which leads to high traffic on daily commute. Cars are not fully utilize if it carries only one passenger, and leads to the main reason for creating traffic jam commuting from one place to another. This can be solved by pairing users together with the same destination up to the maximum capacity of the car, the traffic condition could be reduced at least 4 times as compared to the worst case scenario mentioned earlier. By analyzing the field of software development integrating with social media, this is made possible when information is shared on a platform which is to be viewed publicly, and user with high similarity will be known and paired which makes up to solving the case earlier. Crowdsourcing the data would automates the whole process and reduce the effort in exploring potential user in the domain of traffic and transportation [2].

These sharing economy business is built not only on the technology advancement, but also in a sense of "trust" among each other [3]. The sharing of object among strangers solely rely on trust, is the key of success to the service provided, and it can be provided with a little help from technology. Building of platform ease the communication among each other. Tracking of whereabouts and usage rate can be monitored and viewed, in order to prevent unwanted events to happen. User will not be easily affected by words and saying from another user, and instead it will be affected by mainly the service provider's performance and reputation [4].

2.3 Bike Sharing

Since the operational concept of the Android application of MyCOMS is roughly similar to bike sharing system, the case studies done by researchers will be discussed in this section to further differentiates the difference comparing to ride-sharing application. According to Shaheen, bike sharing system is currently one of the transportation system in major cities, in solving congestion problem, as well as the 'last mile problem' which can be further defined as the distance between home and the transit station, which can be too far to walk [5]. Most of the big cities are using bike sharing as the solution for users in commuting between places where transit station is not available. Ricci has stated that bike sharing has a network of bicycles which spread across an area of strategically positioned 'bike sharing station', which can be used by user for point-to-point journey purpose in a period of time [6].

With the reference made, the similarities is shown between bike sharing and MyCOMS application, where both of the application involves deployment of vehicles at stations, and allow users to book and use it for a period of time, for point-to-point travel purposes. However the downside of both application is that the vehicles would require rebalancing every day, as the demand will differs according to the factors based on the stations. The demand will fluctuate based on several aspect such as the weather, time of travel, location of station, population of station and etc., where every element plays important role in affecting the usage level of the application.

In order to rebalance the vehicle deployment, example and reference are taken from bike sharing model. Several research were done in predicting the demand as rebalancing of vehicles is always an issue for the service provider, with various method used. To avoid wastage of manpower in providing mobility in allocating the cars at the desired location, artificial intelligence system is adapted into the system. According to Syed, their study in forecasting bike demand using fuzzy interference mechanism has proved that the un-optimized fuzzy interference system can outperform un-optimized traditional feed forward neural network, and concludes that simple fuzzy interference based prediction is sufficient [7]. Author also uses input parameters such as weather, holidays and number of user. Randomize of input is first done, and the system is run 5 times to ensure that the data obtain is accurate and consistent. This paper shows that the prediction using fuzzy logic is feasible, and is not limited only to using neural network for forecasting purpose.

Other than that, user's trip can be predicted as well according to the research which based on the fuzzy logic based trip generation model [8]. The prediction on user's trip is important as it affect the demand level of each station in a whole. A notification can be sent to user as an alert of vehicle availability if the prediction data matched with the real-time data. A similar research on the user's journey prediction is done as well [9] train predict. It predicts the passenger flow on journey from station to station. From it the traffic management can be done effectively as it helps in providing mobility to the vehicle in reaching stations which have high demand. This paper models the flow with fuzzy logic, adding features of passenger demand to the mode of transportation [10].

With the high number of bicycle in a network of bike sharing system, a group of researcher did simulation optimization, to make sure that resources of bike sharing are shared and used equally [11]. A proper data retrieved from the network can be mined and analyzed for future transportation improvements [12]. Collaboration between modes of transportation will improve the strength and weakness of each other, which brings more benefits to the users. However comparing on the artificial intelligence method that are involved in prediction, several researchers have use different method such as planning algorithm and searching algorithm [13, 14]. Fuzzy logic will be used for this application as the data and number of input which the system has to deal with, is not complicated compared to the large network node in bike sharing system. Simple system for daily prediction require several input is suffice.

2.4 COMS

COMS is a single seated electric vehicle that is designed and manufactured by Toyota. According to the specification obtained from Toyota Auto Body Japan's Corporate Social Responsibility Report 2013, COMS is a fully electric powered vehicle, that uses lead battery which can supply mileage of 50km per charge. COMS is able to run at a maximum speed of 60km per hour, and it has noises which is generated at below 71dB at acceleration, and even lower at 65dB when it is constantly running [15]. It is targeted to serve user that would like to make short trip easier and more convenient. COMS has been deployed in several region of Japan, aiming to reduce carbon emission, as well as navigating through the narrow roads in most area of Japan. Its performance has won itself a Nikkei MJ

Outstanding Award, which further strengthens its identity among community.

Researchers have used COMS in various field, which ranges from entertainment field to safety autonomous featured ride on the road. A virtual reality (VR) system is modeled based on motion car, which in this case COMS was used as the platform [16]. The COMS was deployed as a motion platform, where it provides simulated motion sensation to the VR user with autonomous driving. User will control the car with VR controller and headset, where vision through the headset will feel the motion when the car is synchronized. It is mentioned that COMS is used as it is easy to control and modify, which allow the developer to create more enhancement on the vehicle with the application of the technology.

Another group of researchers have tried to utilize communication between vehicles by programming the vehicles to undergo platooning among each other [17]. By having platooning and communication established among COMS, it is effective to notify users on availability, and enhance security features as the vehicles can communicate with each other. Safe distance are maintained with this features installed, data can be retrieved for analysis on the driver's driving behavior. With the autonomous driving system developed from the past, this feature can be installed on COMS to provide more mobility around the campus [18]. Single-seated MyCOMS has a limitation where it will have limited access when it is being driven to location which is not populated. The vehicle is not able to be used by people from other station, if there is no one to drive it from a remote station. The deployment of vehicle every morning can be scheduled automatically with autonomous driving, in arranging starting location of MyCOMS at a more strategic location.

The battery level can be monitored and tracked as previous researcher has done the research in tracking and managing the electric demand and usage for electric vehicle [19]. With this feature the battery level can be monitored and even predicted daily, in order for the vehicle to function properly throughout the day without having issue of user is stuck in the middle of the journey due to low battery level. A total number of trips can be counted and predicted in advance based on the daily real-time data, and is then charged to prevent the stated issue to happen.

2.5 Android

Android is the software package chosen for developing the MyCOMS system, where it includes several main components, such as operating system (OS), hardware drivers, middleware, and core applications [20]. The preferred programming language for Android was previously Java, and it is now officially supporting Kotlin and Java from an official statement from Google [21].

2.5.1 Android Architecture

The architecture can be divided into 4 levels, which are Application, Application Framework, Libraries and Android Runtime, and Operating System as shown in Figure 2.1.

Level 1 - Application This layer has a set of core application provided from the android develop and support team, which includes maps, email, SMS, calendar, default web browser and others. This layer provides interaction between user and the machine. Customized software application can be used here for interaction between users too. It is the top most layer of the architecture, and can be written in Java or Kotlin programming language.

Level 2 - Application Framework This layer has multiple Activity Views which is helpful in developing and building an application. It includes grids, lists, Buttons, MapView and etc., which can be embedded easily into application with certain lines of code.

Level 3 - Libraries and Android Runtime This layer is the third layer and it has a set of libraries which has most of the functionality in the core libraries of Java programming language.

Level 4- Operating System This layer implies that Android is run on a Linux Kernel, where it uses several services such as memory management, security, network stack and etc. This layer serves as the interaction phase between the hardware and the software in the machine.

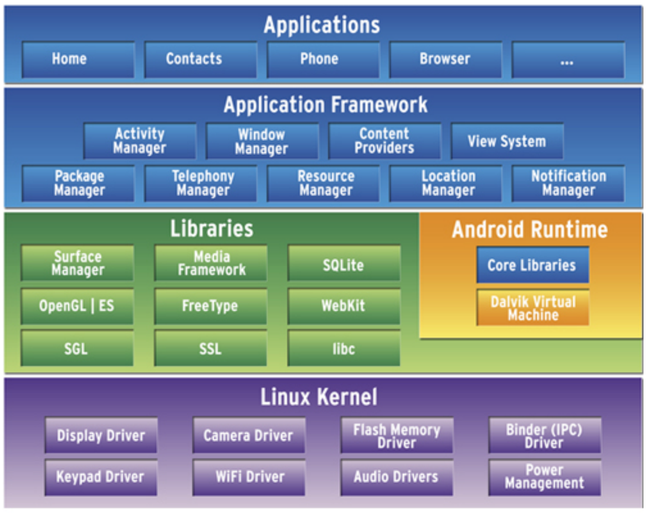


Fig 1. The Android system architecture uses a standard software-stack approach with Linux as its base.

Figure 2.1: Android System Architecture

2.5.2 Android Application Development

Android application will be built using Android Software Development Kit (SDK) and Android Studio Integrated Development Environment (IDE). The IDE has all the tools required, and has support from the official Android developing team on plugin and software updates.

SDK provides software packages, software framework, sample code and application software development tools. It has Application Programming Interface (API) and tools provided for development of application for Android platform with the use of Java and Kotlin programming language. Basic tools such as debugger, libraries, emulator, documentation and etc. are included in the SDK

2.6 Google Map

Google has released the API, which is developed to allow interaction between google services with internet content into web or mobile application. It acts as a black box, in receiving input of user, and process it, and is able to return the query with a Street view scene. Researcher have use the combination of Google Maps together with GSM modem, in providing tracking and locating services of vehicle [22]. Usage of smartphone which has both of the function, will allow the tracking and locating vehicle for monitoring purpose more convenient. According to this paper, the vehicle fleet management system can be polished with the tracking using GPS[23]. A software is created for the tracking purposes as data is retrieved from the real-time update of the system. The system can be even be more advance with the implantation of GPS system on a System-On-Chip, and built in into the vehicle [24]. The same tracking method is also applied in this project, and it is fully deployed to evaluate the system performance[25].

2.7 Review of Similar Application

2.7.1 oBike

oBike originates from Singapore, provides the security features between user and the bicycle, by having the user to scan QR code via the oBike app or enter the bike ID found on the bike. The application also allow bike booking, and is reserved under the user's name for at most 10 mins. The sign up process requires phone number, and requires the user to allow permission on location so that nearest oBike can be found on the map via the oBike app. Payment can be done via Credit and Debit cards, and more payment option will be available later. It also has the Bluetooth technology to allow the users in locating the bicycle for a higher accuracy, and to identify the bicycle if there is a lot of them.

2.7.2 MoBike

Mobike that originates from China, has a lot similarities in terms of application as compared to oBike. The difference between these two companies is that the location of deployment is different, and the infrastructure of the bicycle is different as well. The application allow user to book for 15 minutes, and is able to view the final location of bicycle when the trip is ended.

2.8 Summary

As outlined in the previous section, there are several application that can be referred to, and the features in it such as QR code scanning, Bluetooth location detecting and GPS location identifying can be adapted into MyCOMS application. The overall architecture of the developing structure of Android is explained and outlined to have more insight on the functionalities of the components. Several studies on the ride sharing and bike sharing system is done, and is relatable to MyCOMS application as it requires sharing of vehicles and time of usage optimized between desired users. Therefore, a hybrid version of ride-sharing which is MyCOMS application is needed in order to tackle the targeted vehicle, and improve the overall deliveries to the users.

CHAPTER 3

METHODOLOGY

3.1 Introduction

The main ideas to be carried out, development process and cycle, hardware and software needed are discussed in this chapter. This project is targeted to create a ride-sharing application in Android, which is suitable for staffs and students in UPM Serdang campus, in travelling in between faculties and major location point using MyCOMS. This application is developed using Android OS, and is installed in an Android smartphone or tablet. The project development cycle is represented graphically in figure 3.1, as it shows the overall flow of the project is developed.

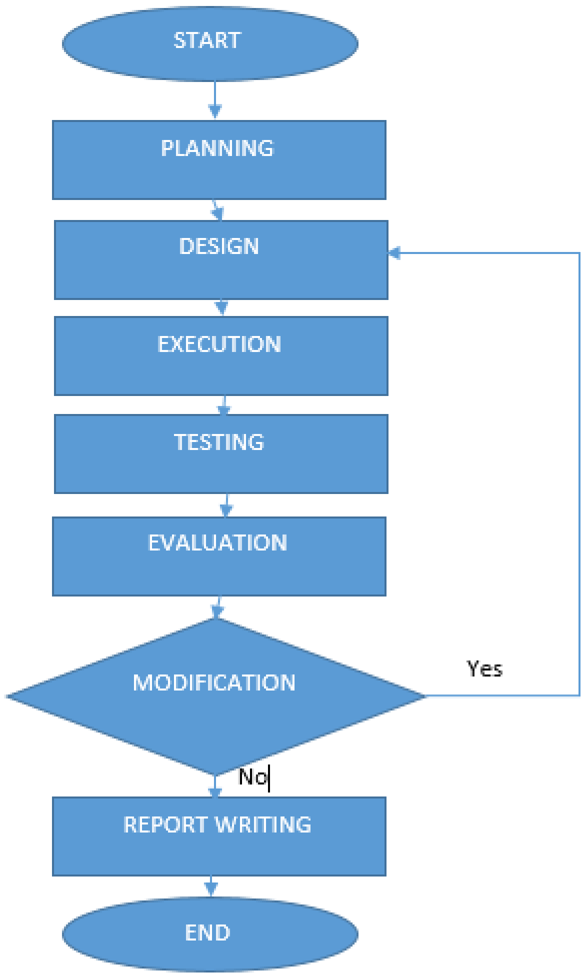


Figure 3.1: The flowchart of the proposed project

The first stage of the project is to carry out planning. It is carefully plan and a scope of work is detailed out in order to efficiently handle the project. Problem statement, aim and objectives are identified so that goal is clearly defined, and the project will be stay focus on track through the process of development. Timeline and Gantt chart is drawn in order to track and plan the project progress in order to complete it within the given time period.

After the goals are set, design is carried out together with the teams that is involved with the MyCOMS application development project. Rounds of discussion and decision have been made in order to achieve the targeted functionality of the application. Based on that, the software and hardware requirement are discussed and decided. Choices of programming language and IDE is done, which is Java as the main language of development, on Android Studio, which to deploy the Android application on an Android compatible devices.

During project implementation, the application will be installed on devices, and real time data must be captured in order to evaluate the system performance. Testing and debugging is done to ensure that minimal bug activity is supervised. If there is no modification to the project, it will advance to the last stage where it is discussion of result and conclusion.

3.2 Software and Hardware Development Requirement

The software required for this project is Android Studio and Adobe Experience Design. The following section explains the roles of having these software for the development for the project.

3.2.1 Software Requirement

3.2.1.1 Android Studio

Android Studio is an open source IDE and free software which is released by Google, mainly targeted for developers which aims to create application for Android platform. It has pre-installed library, sample code, tools and etc. required in creating an application. It has workspace and extendible plug-ins available to allow customization for Java application development. The Android studio version used in this project is Android Studio 3.1.2, JRE: 1.8.0_152-release-1024-b02 amd64, with SDK version of 27 and Gradle Build 3.1.2.

3.2.1.2 Adobe Experience Design (Adobe XD)

Adobe XD is a user experience design software developed by Adobe Systems. It is a pay to use software, and offers 7 days of free trial. It allow users to do design and customization on the user interface of an application, and allow the interface to be modelled into an application with limited function of interface. It supports vector design and wire framing, and allow users in creating simple interactive click-through prototypes.

3.2.1.3 Android Virtual Device (AVD)

AVD is a built-in emulator by Android Studio, which allow users to create a mock Android device without having it in real life. Customizable RAM, OS and pixel setting can be done to simulate the created application before deploying it on the real-time devices. It is used in testing and debugging phase of the project as the codes must be tested out frequently to check the functionality, and several model of Android devices can be tested in order to know the flaw of the software before deploying it on market.

3.2.2 Hardware Requirement

For the project, computer and smartphone is a must as it is the basic tools needed in developing the application. MyCOMS electric vehicle will be involved once the application is ready to deploy, in order to get real time data to evaluate the system performance.

A system block diagram is shown as in Figure 3.2 in order to visualize the relationship between each component. It is illustrate as below.

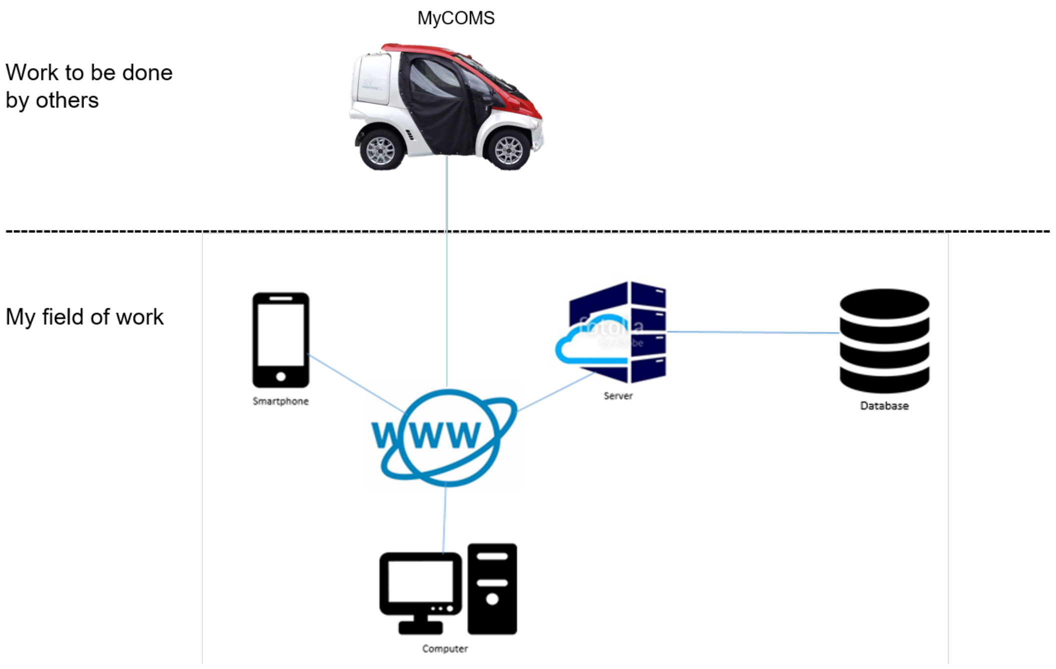


Figure 3.2: System block diagram of the project.

The application will be linked to computer and database through internet and server. User will be using the application, while the administrator is able to view the activity through the server, and analyses the data through the database. This step is important as it allow the administrator to evaluate the system performance.

3.3 Application Design and Development

For the design of the application, a flow of required function is detailed, with the targeted group of users. The flow is represented in Figure 3.3.

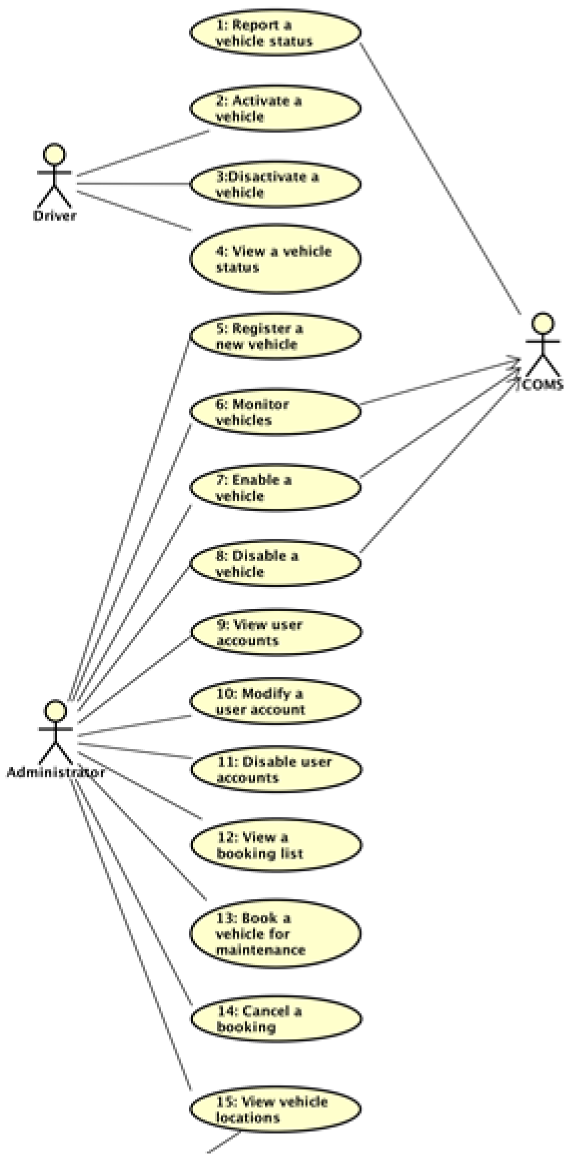


Figure 3.3: Use case diagram for the application.

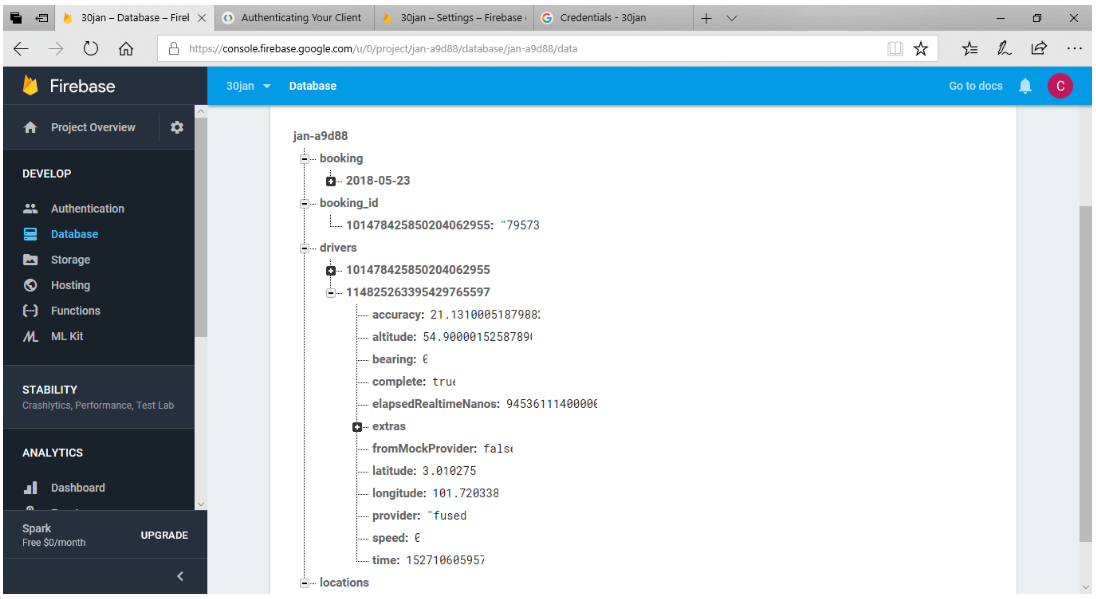
Firstly, the main screen of the application will prompt user input of identify, such as matric number and password for security purposes. New sign up user will have to register themselves, while registered user have to log in just once, unless the user log out after every session. The application will have selection where user is able to book the vehicle and check the availability with real-time synchronization of database. Once the booking is done, user will be able to start the application as a navigation application, where it will route between the starting and the ending point of the trip. System will alert the user if the route is detected to be out of course from the targeted location. Abnormal activity of the application by user will be recorded and uploaded to the database for reference purposes.

3.3.1 Firebase Database

Firebase is an online database that is released by Google, to allow more control of Android application. This database allow android application to store and synchronize data in real time. Firebase allow encryption of 2048-bit SSL for the protection of the data. It has custom authentication and access control for developers to suit their application development specification.

Firebase can be used in cross platform development, such as iOS, Android and also Web development. User will need to sign up an account in Firebase, the API can be easily embedded into the application and to create connection to the server. Firebase saves time as it does not require extra scripting on server-side.

Figure 3.4 shows the relationship between the elements in the database.



```
jan-a9d88
 ⊟ booking
    ⊞ 2018-05-23
 ⊟ booking_id
    └ 101478425850204062955: "79573
 ⊟ drivers
    ⊞ 101478425850204062955
    ⊟ 114825263395429765597
       ├ accuracy: 21.1310005187988
       ├ altitude: 54.900001525878
       ├ bearing: 0
       ├ complete: true
       ├ elapsedRealtimeNanos: 9453611140000
       ⊞ extras
       ├ fromMockProvider: false
       ├ latitude: 3.010275
       ├ longitude: 101.720338
       ├ provider: "fused
       ├ speed: 0
       └ time: 152710605957
 ⊟ locations
```

Figure 3.4: Relationship between elements in database.

3.4 Application Design and Development

For the design of the application, a block diagram for the application overview is shown in figure 3.5, where it shows the overall activity and functions to be carried out in the application.

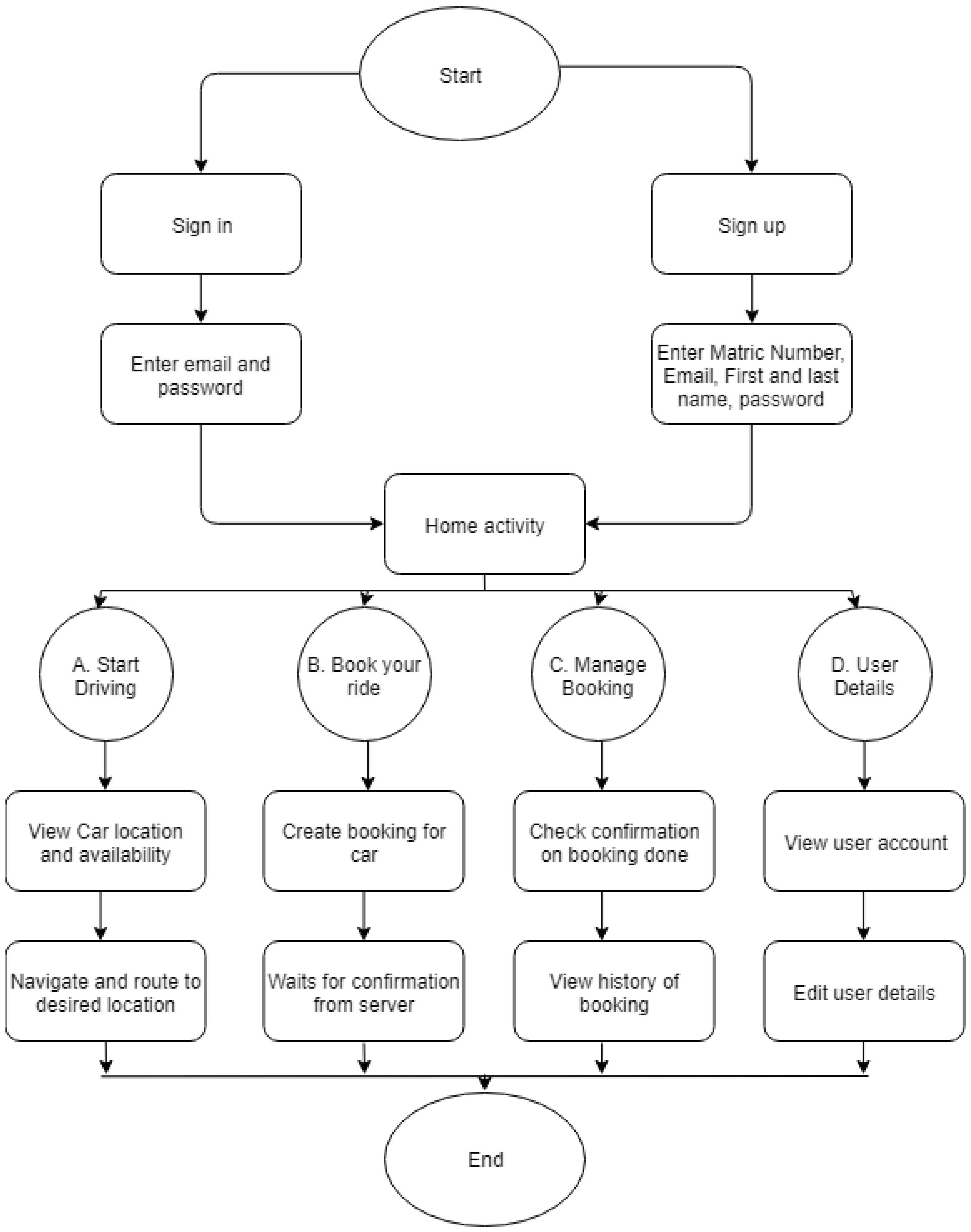


Figure 3.5: The flowchart of the system

User will be prompted to register or log in to the system at the main interface of the application. Once the user's identity is confirmed which is a student or staff of UPM, the user will be lead to the home interface where all the functions are located. User will be able to choose between viewing the location of the vehicle, booking the vehicle, viewing and managing the status of the booked vehicle, and also view user details. User can choose to log out from the application, or stay in the session until next launch of the application.

User can see the real time location of the vehicles so that they can know when and where to book the available vehicles. User can also delete and authenticate the booked vehicle by going to the managing booking activity.

3.5 Testing and Debugging

AVD emulator plays an important role for this stage of development. It allows the application development process into easier and more convenient. Customization of Android devices can be done so that developer can predict and observe the errors that might happen if the application is deployed real-time.

For final stage of the development, application is tested on several Android devices such as Samsung J7, Redmi Note 4, and emulated device in terms of resolution.

3.6 System Requirement

3.6.1 Functional Requirement

System is defined by a group of functions which is required to run. A system requirement consists of functional and non-functional requirements which has constraints to the design and implementation.

Table 3.1 shows the functional requirements, labelled by ID.

Table 3.1: Functional Requirements.

ID	Description	Explaination
FR1	Users must be able to sign up.	The user must be able to sign up with relevant matric number, name, email and password.
FR2	Users must be able to sign in.	The user must be able to log in using matric number and password.
FR3	Users must be able to log out from the system.	The user must be able to log out from the session.
FR4	Users must be able to change their profile details.	The user must be able to change profile details such as password, email, name and matric number (if there is no duplicate).
FR5	Users must be able to book a ride.	User must be able to view availability of ride and book it.
FR6	Users must be able to cancel a ride.	User must be able to remove the booking from the ride list.
FR7	Users must be able to launch the driving mode of the ride.	User must be able to start the tracking and monitoring features of the application.
FR8	Users must be able to end the driving mode of the ride.	User must be able to end the tracking and monitoring session.
FR9	Users must be able to view the location of vehicle.	User must be able to see the current whereabouts of the vehicle to decide on booking.
FR10	System must be able to generate error message.	System must be able to recognize faulty process throughout the launching of application.

3.6.2 Non-Functional Requirement

A non-functional requirement of a system will define what a system will do, but not the way the system will do it. Several parameters such as system performance requirement, design constraint and software quality attributes are one of the example of non-functional requirement.

The system's related non-functional requirements are shown in table 3.2.

Table 3.2: Non-Functional Requirements

ID	Description	Explanation
NFR1	The graphic user interface should be user friendly, and maintain a style for all screens.	The buttons, menus and layout should maintain the same design for all screens so that user will not be confuse with different design of buttons.
NFR2	Notification message should be display clear and concise to the user.	System will display message in pop up window in informing user.
NFR3	The application must not be affected by bugs, and will inform user on wrong steps taken using the application.	System must be checked in possible of bugs, and only be released when no harmful bugs are found.
NFR4	The application will have fast response time.	The system should provide operation and react to instruction very fast.

3.7 Station Level Demand Prediction

3.7.1 Design Flowchart/Diagram And Processes

Fuzzy logic is used in predicting the station level demand, and a basic flow of work is needed in ensuring the process of data is smooth with the use of fuzzy logic. Figure 3.6 shows the flowchart of fuzzy logic. To use the fuzzy logic, it must at least have the basic of several input variables, and is then fuzzified with the input membership function. Fuzzy rules are constructed to create interference with the fuzzy membership function, and together with the output membership function. Defuzzification is then done to obtain the center of gravity for the output, in our case which is the demand level.

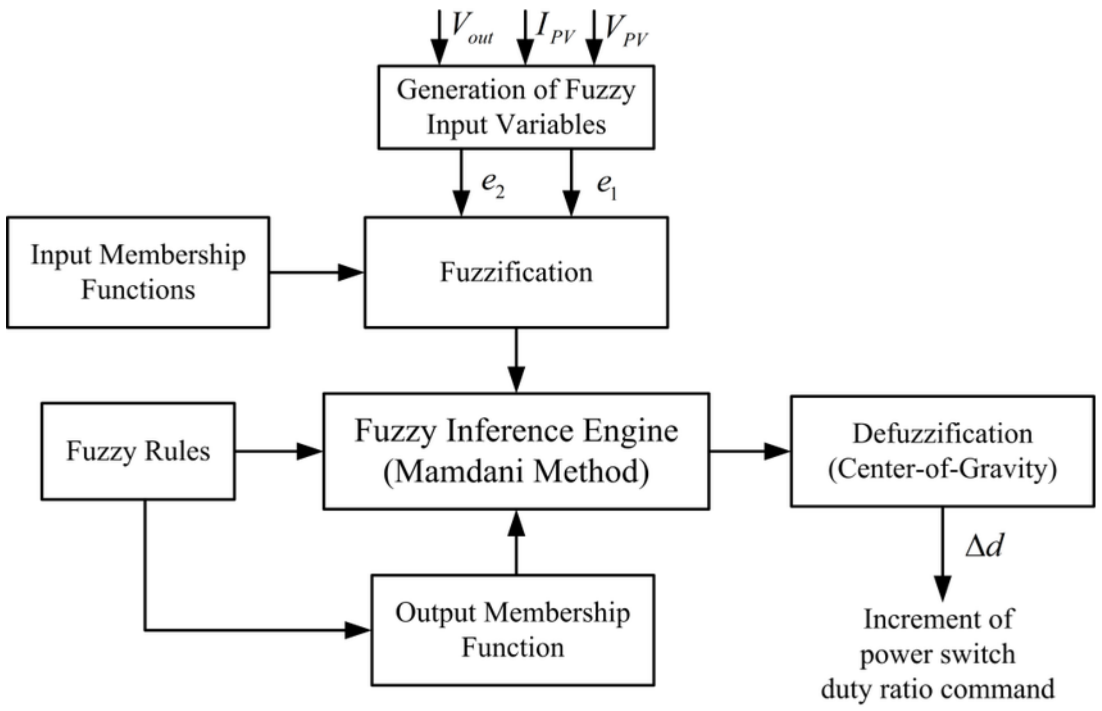

V_{out}
I_{PV}
V_{PV}
Generation of Fuzzy Input Variables
e_2
e_1
Input Membership Functions
Fuzzification
Fuzzy Rules
Fuzzy Inference Engine (Mamdani Method)
Defuzzification (Center-of-Gravity)
Δd
Output Membership Function
Increment of power switch duty ratio command

Figure 3.6: Fuzzy Logic flowchart

To complete the steps, parameters and inputs are then determined based on the case studies and literature review. It must be done carefully so that the output obtained will have better accuracy and consistency. The fuzzy rules must be done precisely according to the input and output so that the decision making will match and par with the expected outcome set earlier. A flowchart is constructed so that the flow of the whole process can be supervised and referred to, to ensure that the steps taken will generate the best outcome after the process of fuzzy logic of the input data. Python 3.6 is used in processing the data.

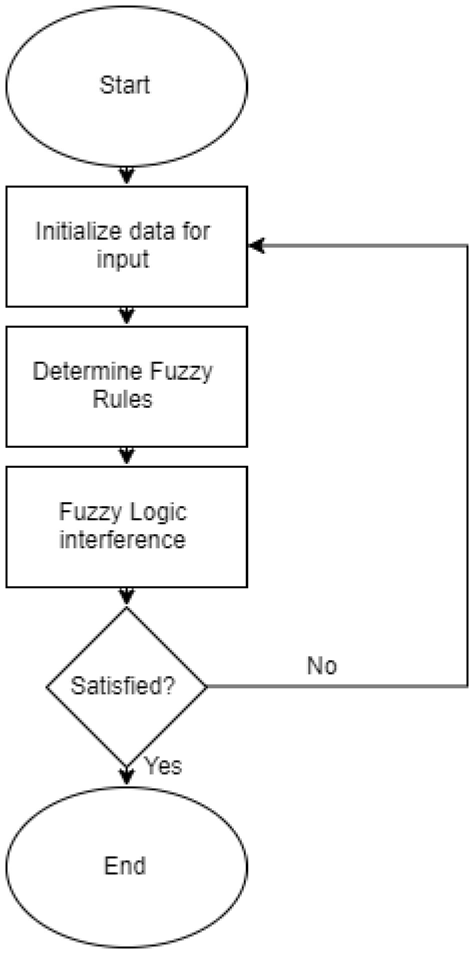


Figure 3.7: The flowchart of the process.

Figure 3.7 shows the overall flowchart of the process. It is expected that the whole process will not require hardware connectivity as the data will be generated randomly. The whole project will be done using Python and SciKit Fuzzy Logic library as the input are easily process using these two medium.

3.7.2 Pseudocode

A simple set of pseudo code is used to demonstrate the flow of the program with fuzzy logic. There is two cases involved in determining the demand level, where both of them are explained at the following section.

Case 1 (fixed input variables to find out demand level):

Begin Initialize input variable with desired data

```
Calculate demand level Print demand level
End
Case 2 (fixed demand level to get values of input variable):
Begin

Initialize demand level

Loop


Random generate input variables


Compare demand level with the pre-defined demand level

While calculated demand level >= pre-defined demand level

Print input variables

Print generated demand level
End
```

3.7.3 Dataset

As there is no real application at the simulation of the fuzzy logic prediction, random data is generated as input in order for the fuzzy logic interference to occur. It is to simulate user's impression and it is hope that the random generation will have the result that has high accuracy consistency as compared to real time data. The collection of data must be done daily once there is a steady application launched to the targeted user. The daily searching of the vehicles is retrieved and processed again to find out the demand for each location. The real time data must be tested to make sure that the searches is reaching the expectation of the artificial intelligent processes.

3.7.3.1 Input

The input are divided into 4 categories, which are:

- Location of station
- Population of station
- Time of travel
- Temperature of the day

The inputs is then further categories in order to plot the membership function. The parameters are pre-defined.

- Location:

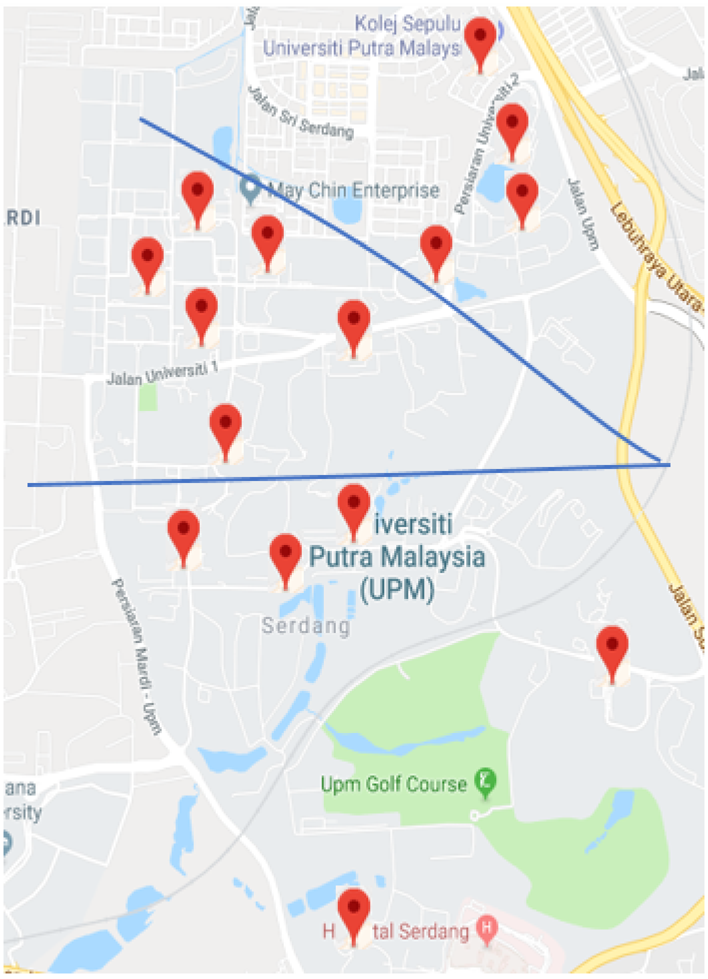


Figure 3.8: The stations based on location

The station locations are plotted on the map with 15 predicted locations as shown in Figure 3.8. It is based on the faculties and the colleges location which has the higher population compared to every other location in the campus. The region is separated with blue lines, into northern, central and southern region. Membership function is plotted as shown in figure 3.9.

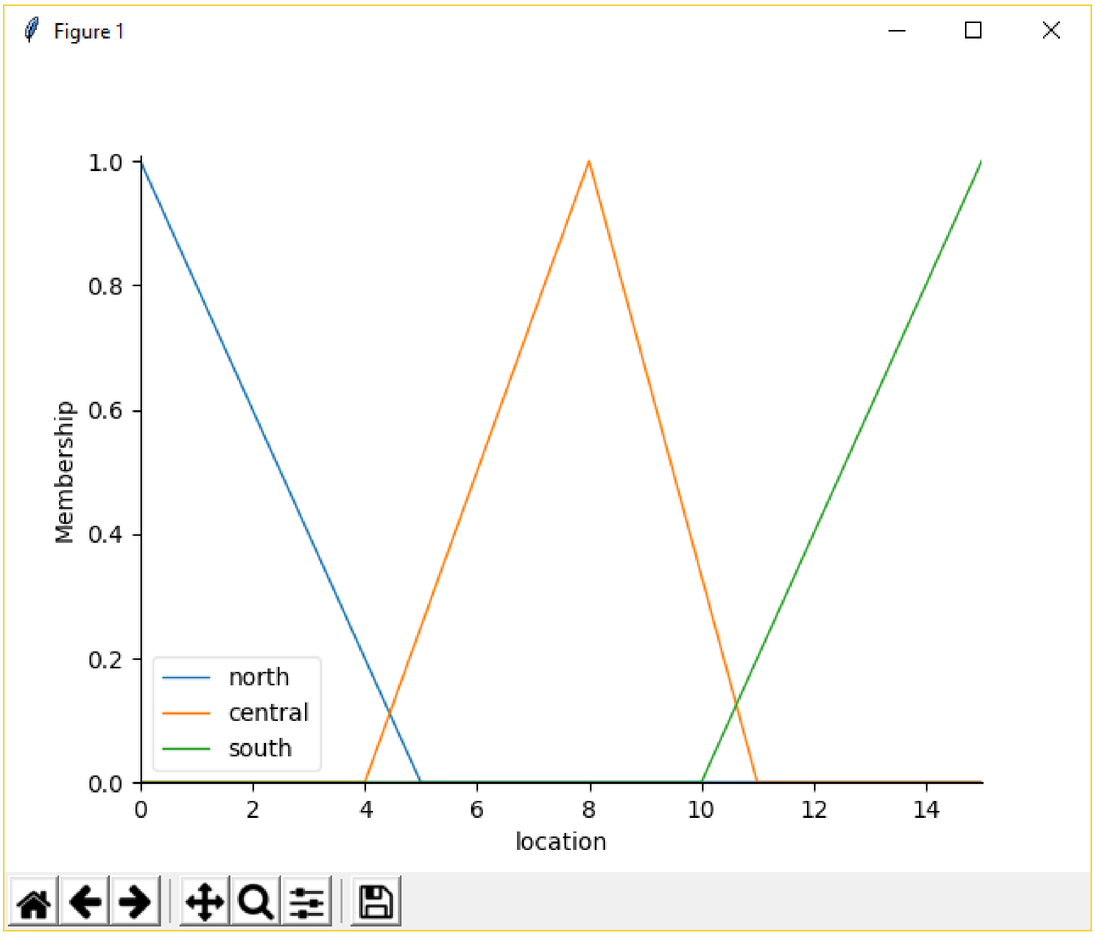


Figure 3.9: Membership function of location.

- Population of station:

  The population of the station is predicted to have a sample size of 50 people per station. The number is then randomly generated for each stations, to find out the demand level. The membership function as shown in figure 3.10.

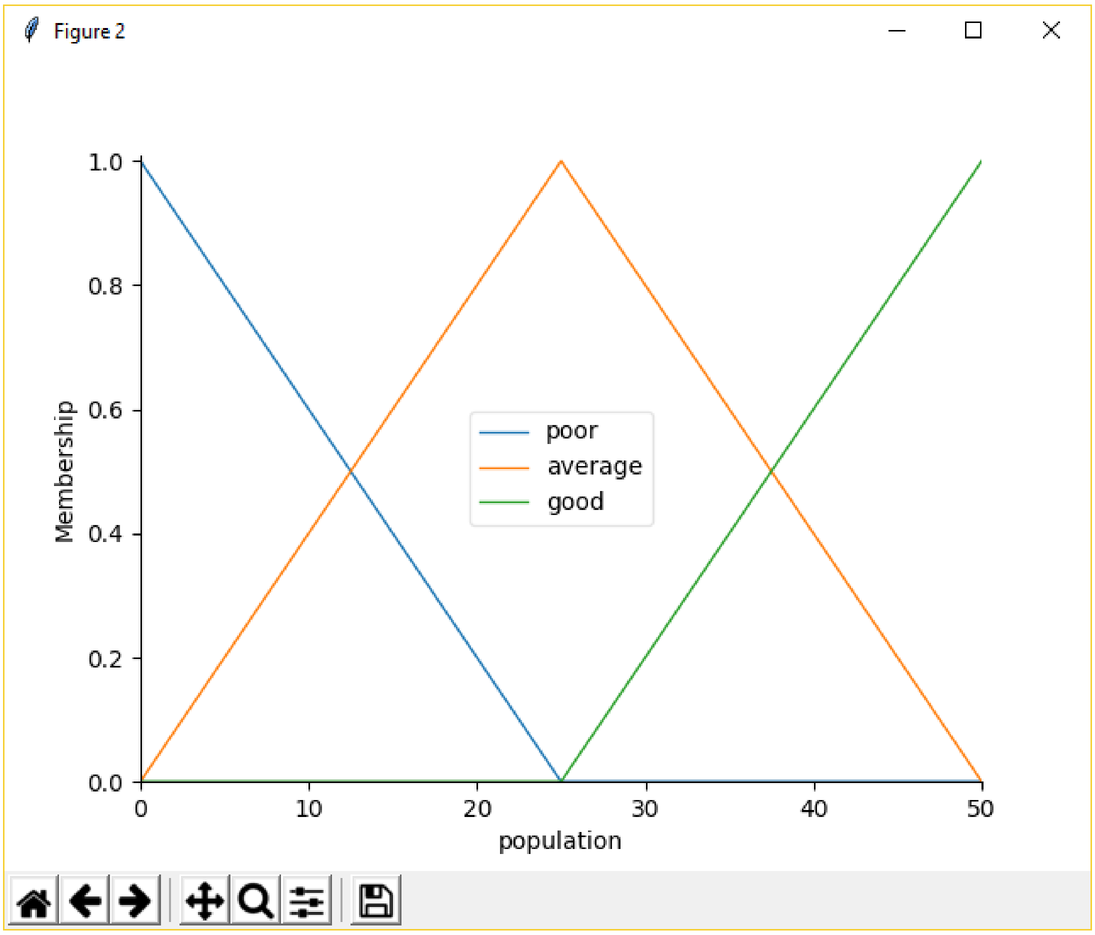


Figure 3.10: Membership function of population.

- Time of travel

  To ease the plotting of the membership function, the range of parameters used will be the full clock hour, which is from midnight until midnight (24 hours), and is further divided into 3 region for membership function plotting. The membership function as shown in figure 3.11.

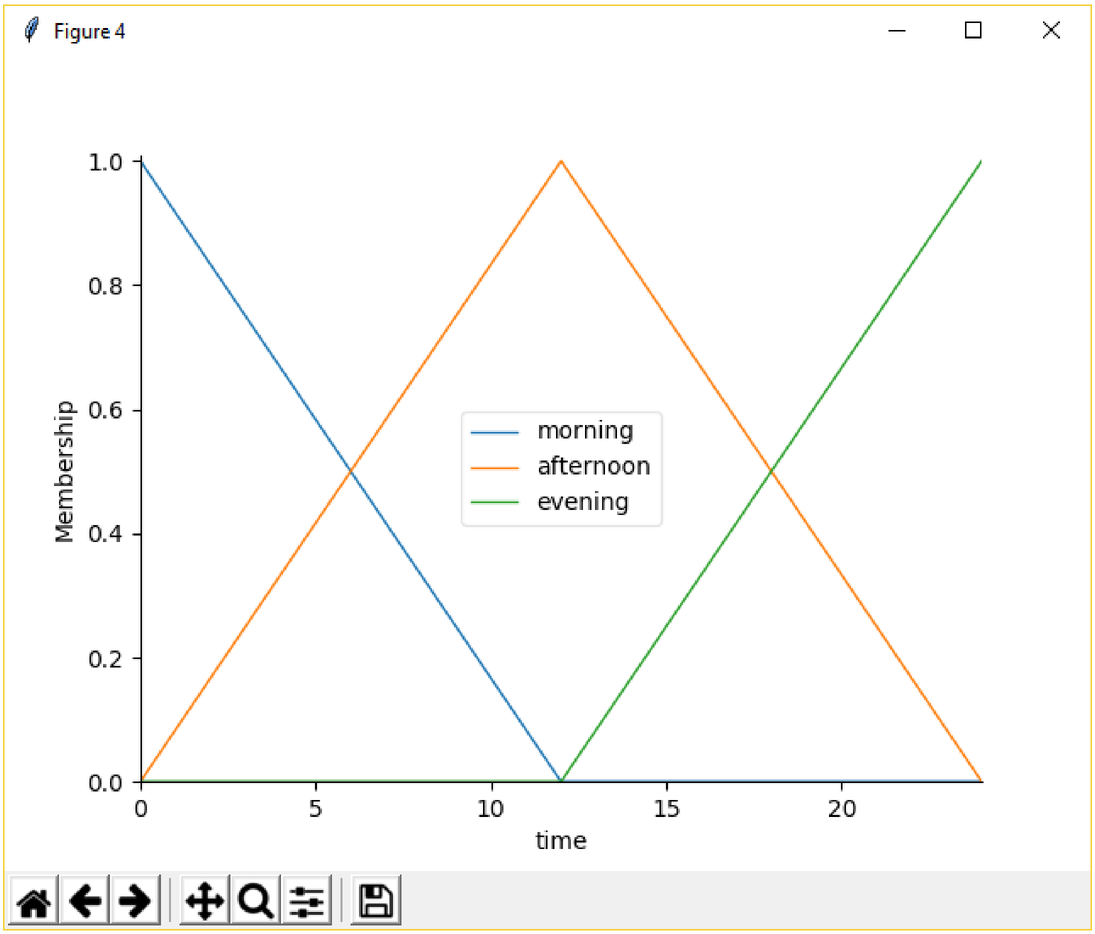


Figure 3.11: Membership function of time of travel.

- Temperature of the day

  The temperature of the day will have the parameters in between of 0 to 40℃. Although in Malaysia there is no temperature that below 16℃, the parameters are set to ease the plotting of membership function. The membership function as shown in figure 3.12.

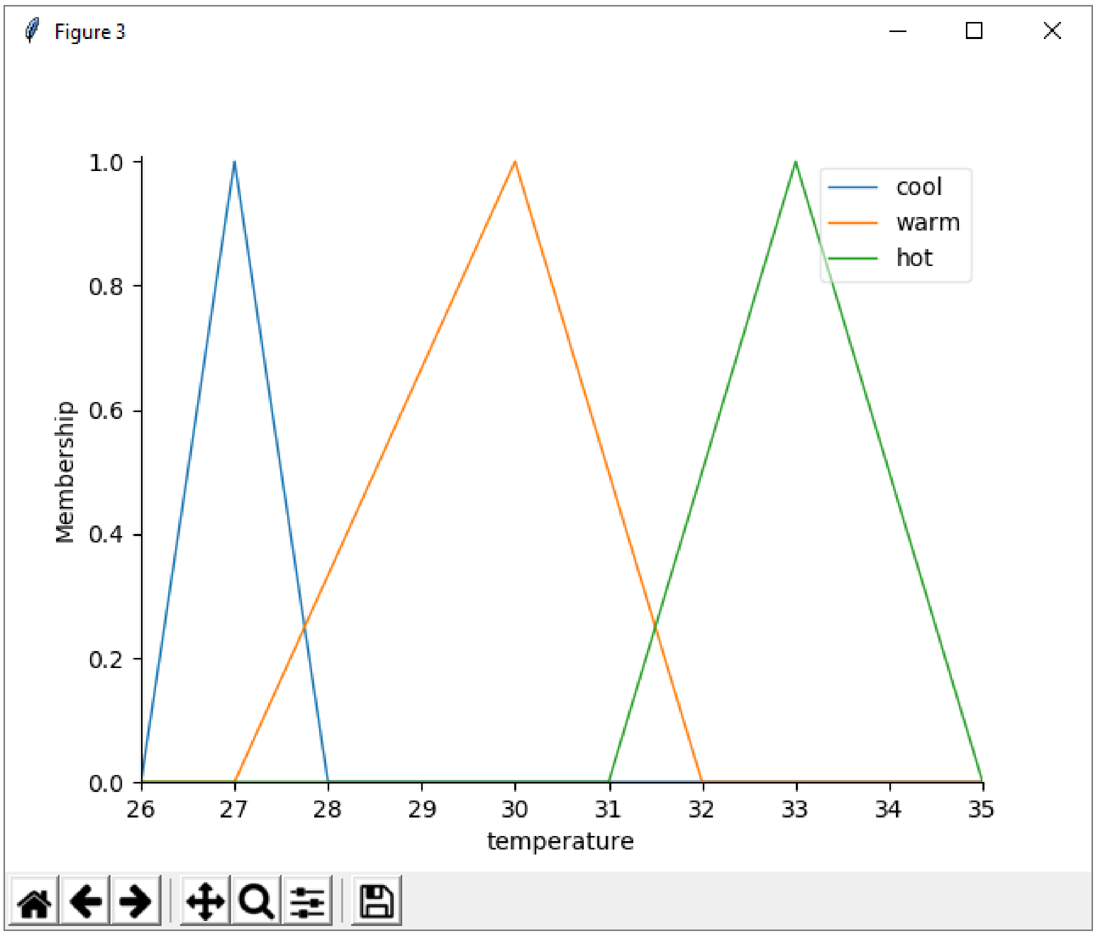


Figure 3.12: Membership function of temperature

3.7.3.2 Output

The output of the fuzzy logic is set as the Station Level demand, with the scale of 0 to 10. The membership function is plotted as shown in figure 3.13.

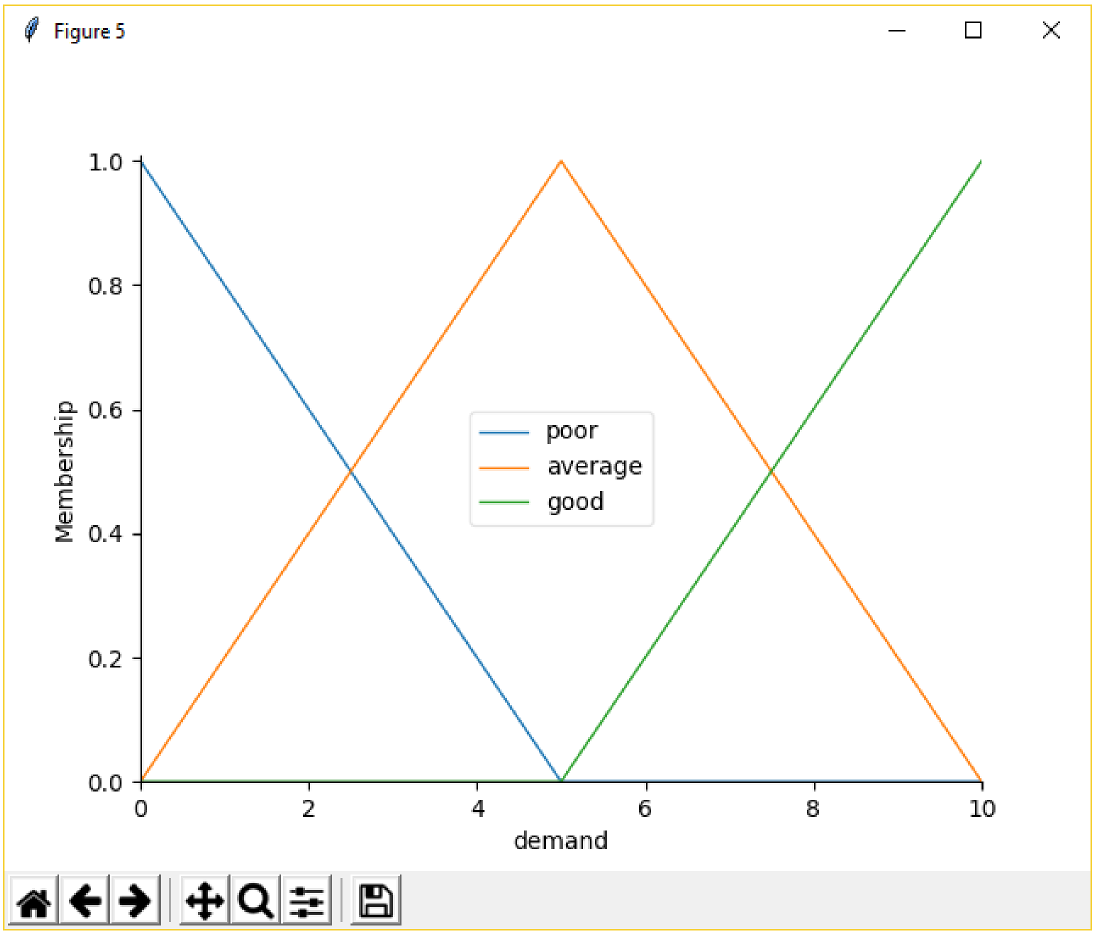


Figure 3.13: : Membership function of demand level

3.7.3.3 Fuzzy Rules

There is 3 cases of fuzzy rules to be set, to make sure that the demand level calculated will have high accuracy and consistency of data.

- IF location is North OR population is Good OR temperature is Cool OR time is in Morning,
  - then demand will be Good
- IF location is South OR population is Average OR temperature is Warm OR time is in Afternoon,
  - then demand will be Good

- IF location is Central OR population is Average OR temperature is Warm OR time is in Afternoon,
  - then demand will be Average
- IF population is Poor OR temperature is Hot OR time is in Evening,
  - then demand will be Poor

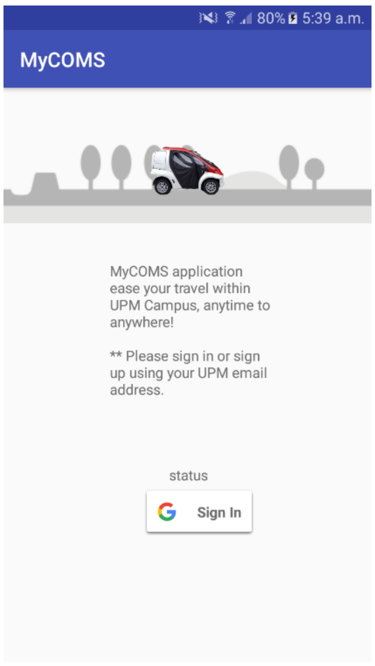
CHAPTER 4

RESULTS AND DISCUSSION

4.1 Introduction

This chapter presents the result of the user interface of the developed application. The features and performance testing of the application is discussed in this chapter too. In addition, the problems and obstacles faced during the application development are discussed and reviewed in this chapter. This ride sharing application will enable user to book EV rides to travel within UPM Serdang campus. Other basic features such as managing and editing the booking, viewing the user and EV's location are also included.

4.2 Main Screen



MyCOMS application ease your travel within UPM Campus, anytime to anywhere!

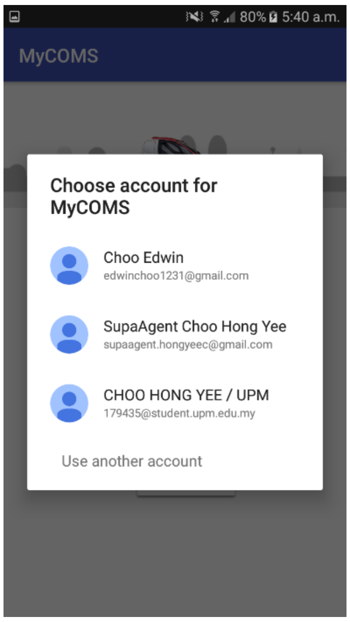
** Please sign in or sign up using your UPM email address.

Figure 4.1: Main Screen of the application.

Main screen is the first screen shown when user activate or runs the application by clicking onto the app. In the main screen, the name of the app is shown on the action bar, as shown in figure 4.1.

4.3 User Authentication

There is only a standardized button for Google account sign in and sign up for new users. User is suggested to sign in and register account using UPM email address only, so that the information about the user which related to UPM can be obtained easily. Figure 4.2 shows the action to select email address for registration and figure 3 shows authentication process, and figure 4.4 shows the interaction with database.



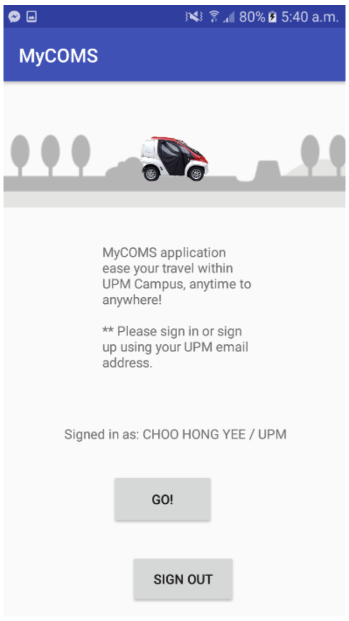
Choose account for MyCOMS

Choo Edwin
edwinchoo1231@gmail.com

SupaAgent Choo Hong Yee
supaagent.hongyeec@gmail.com

CHOO HONG YEE / UPM
179435@student.upm.edu.my

Figure 4.2: Log in or registration of user.

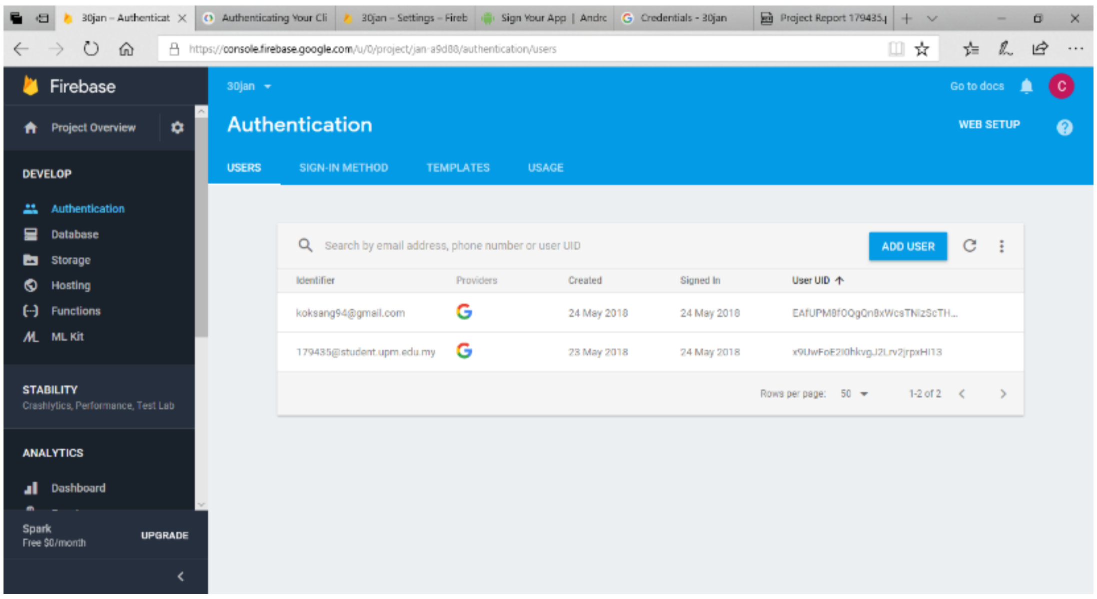


MyCOMS application ease your travel within UPM Campus, anytime to anywhere!

** Please sign in or sign up using your UPM email address.

Signed in as: CHOO HONG YEE / UPM

Figure 4.3: Display of user details on main screen.



Identifier	Providers	Created	Signed In	User UID ↑
koksang94@gmail.com	G	24 May 2018	24 May 2018	EAfUPM8fOQgQn8xWcsTNizScTH...
179435@student.upm.edu.my	G	23 May 2018	24 May 2018	x9UwFoE2I0hkvgJ2Lrv2jrpxHI13

Rows per page: 50 1-2 of 2

Figure 4.4: User details captured in database.

4.4 Home Screen

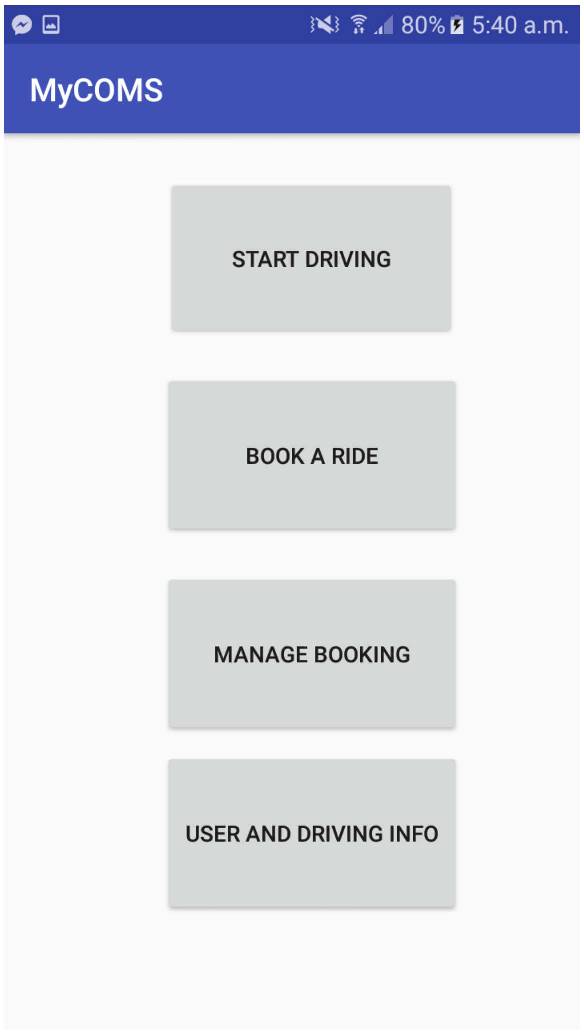


Figure 4.5: Home screen of the app

Once user completed user authentication process by having entering credentials, user will be brought to home screen as shown in figure 4.5. The home screen is the main interface after the user verification, as it acts as the central for navigation of functions of the app. There are 4 buttons on the screen such as “Start Driving”, “Book a Ride”, “Manage Booking” and “User and Driving Info”.

4.4.1 Start Driving

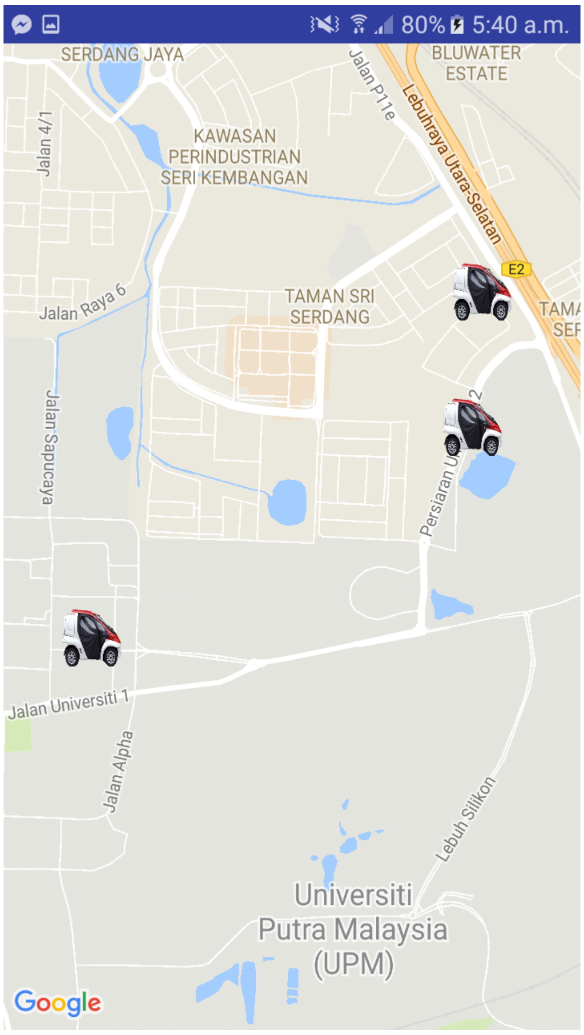


Figure 4.6: Marker shows real time location of MyCOMS

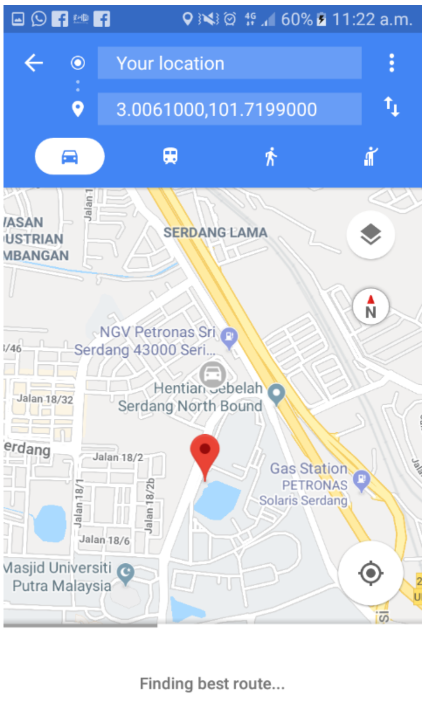

60% 11:22 a.m.
Your location
3.0061000,101.7199000
SERDANG LAMA
NGV Petronas Sri Serdang 43000 Seri...
Hentian Sebelah Serdang North Bound
Gas Station PETRONAS Solaris Serdang
Jalan 18/2
Jalan 18/32
Jalan 18/6
Masjid Universiti Putra Malaysia
Finding best route...

Figure 4.7: Google Maps navigation calculating best route.

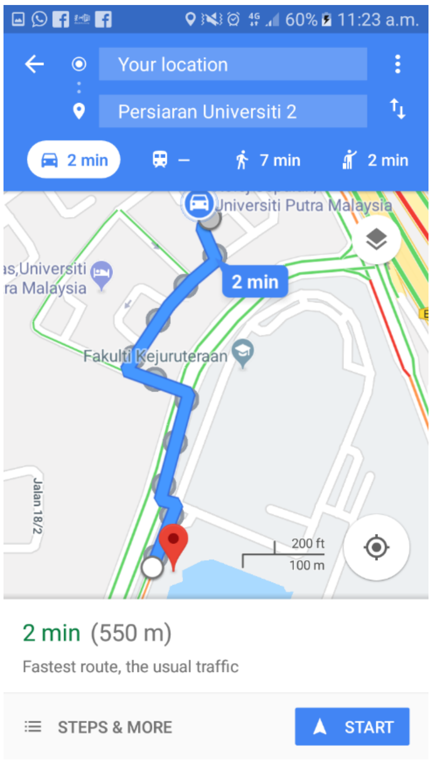


Figure 4.8: Generated navigation by Google Maps.

User will be directed to google maps where to show the location of the available EVs as shown in figure 4.6. User can press the button on the screen of the maps to navigate to the exact location of the EV by using Google Maps navigation as shown in figure 4.7 and figure 4.8.

4.4.2 Book A Ride

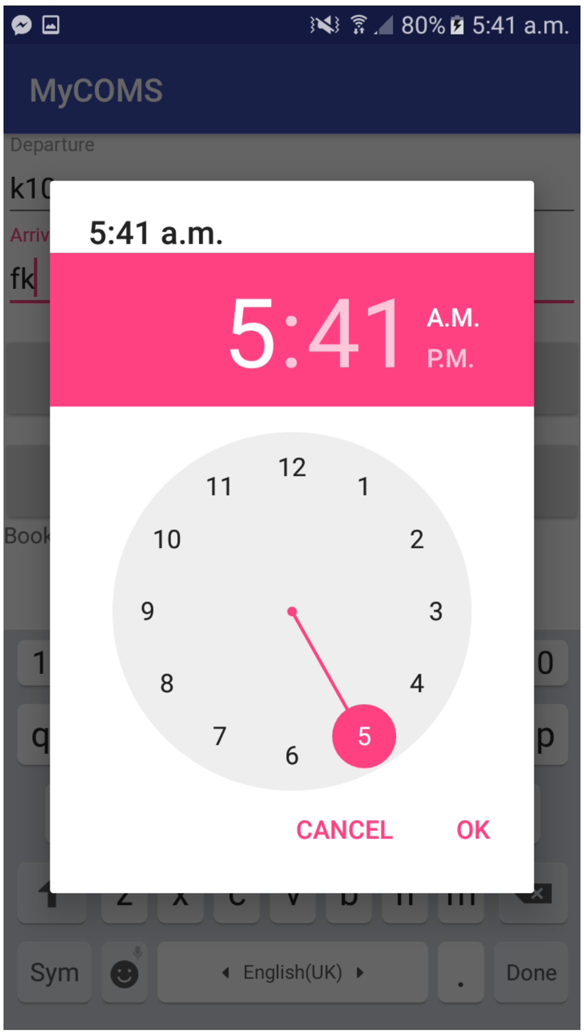
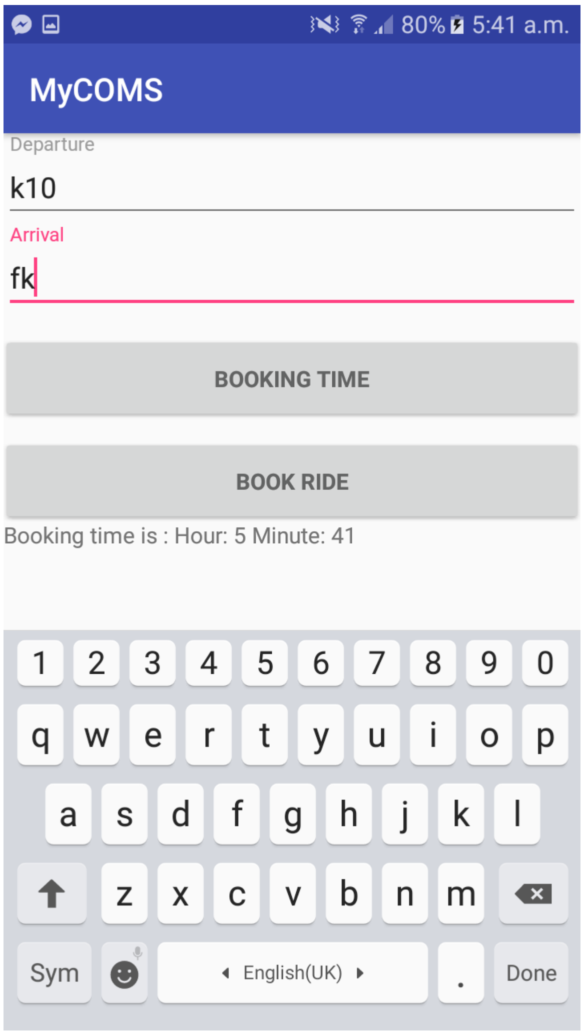


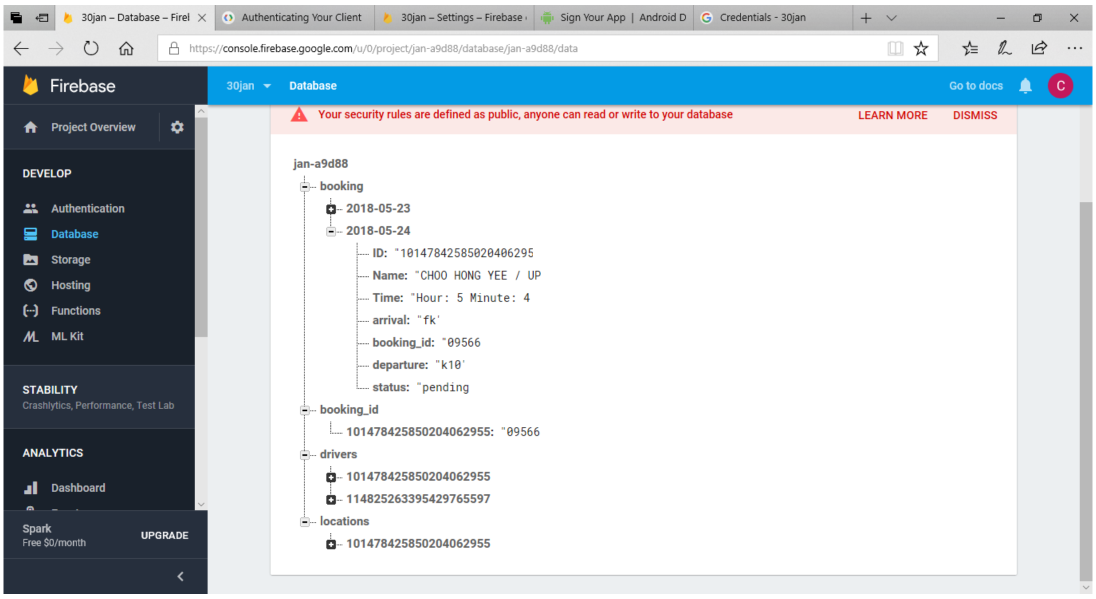
Figure 4.9: Time picker for user to book timing for MyCOMS.



Departure
k10
Arrival
fk

Booking time is : Hour: 5 Minute: 41

Figure 4.10: Display of booking detail in booking screen.



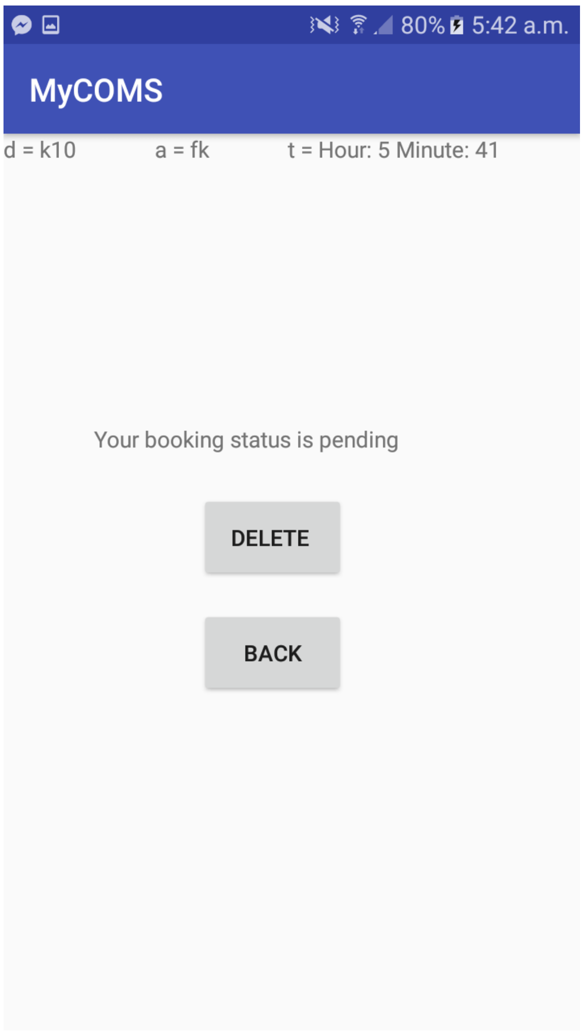
Your security rules are defined as public, anyone can read or write to your database

```
jan-a9d88
 booking
   2018-05-23
   2018-05-24
     ID: "10147842585020406295
     Name: "CHOO HONG YEE / UP
     Time: "Hour: 5 Minute: 4
     arrival: "fk'
     booking_id: "09566
     departure: "k10'
     status: "pending
 booking_id
   10147842585020406295̶5: "09566
 drivers
   10147842585020406295̶5
   114825263395429765597
 locations
   10147842585020406295̶5
```

Figure 4.11: Booking details captured in database.

Figure 4.9 and figure 4.10 shows the screen for book a ride button. User will be able to book the EV's slot using this button. User will be prompt to enter departure and arrival location, including the desired booking time. All the data will be recorded in database as shown in figure 4.11, and can be view by admin for monitoring purpose.

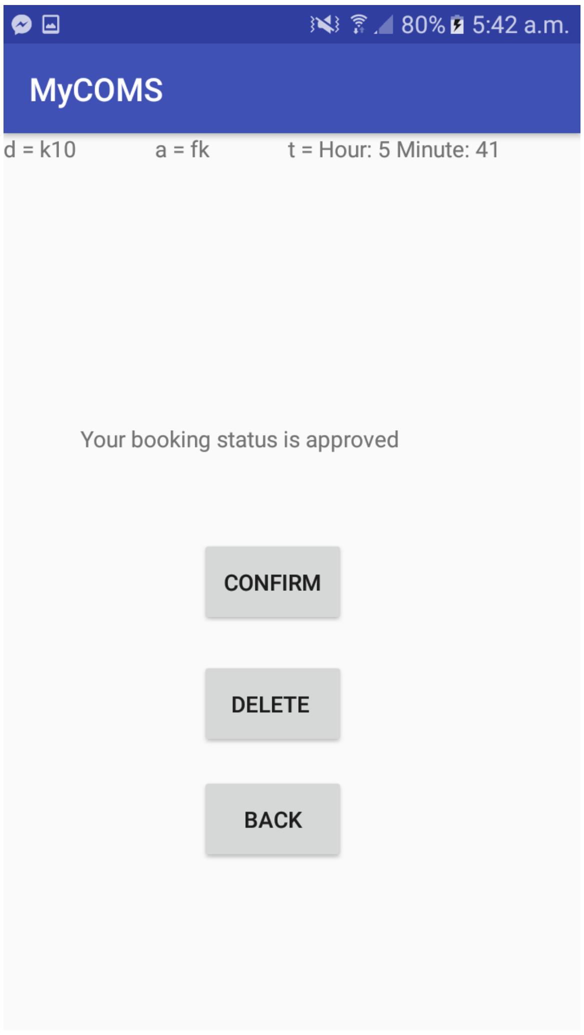
4.4.3 Manage Booking



d = k10 a = fk t = Hour: 5 Minute: 41

Your booking status is pending

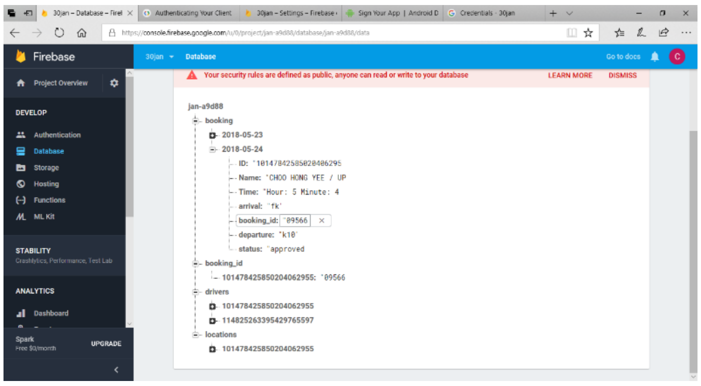
Figure 4.12: Manage booking screen shows status and details of booking.



d = k10 a = fk t = Hour: 5 Minute: 41

Your booking status is approved

Figure 4.13: Confirm button appears after booking is confirmed by admin.



Your security rules are defined as public, anyone can read or write to your database

```
jan-a9d88
  booking
    2018-05-23
    2018-05-24
      ID: "10147842585020406295
      Name: "CHOO HONG YEE / UP
      Time: "Hour: 5 Minute: 4
      arrival: "fk"
      booking_id: "09566
      departure: "k10"
      status: "approved
  booking_id
    101478425850204062955: "09566
  drivers
    101478425850204062955
    114825263395429765597
  locations
    101478425850204062955
```

Figure 4.14: Details captured in database.

Figure 4.12 shows the manage booking screen of the application. It allows user to view the bookings made earlier, and views the status of the booking. Once admin approve the booking, a confirmation button will appear for user to acknowledge the confirmation as shown in figure 4.13. The record of the acknowledgment and approval of rides are recorded in the database as shown in figure 4.14.

4.4.4 User and Driving Info

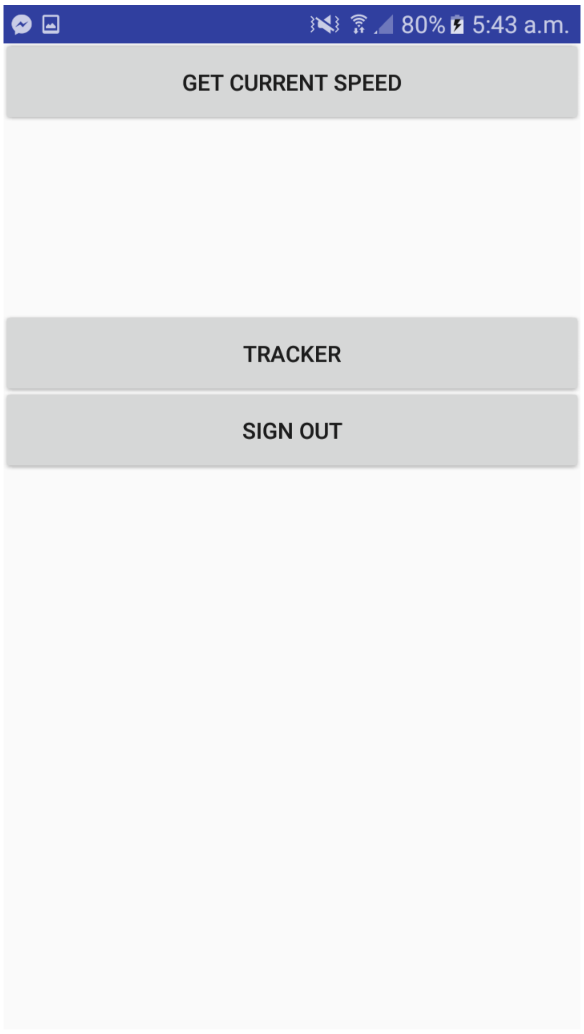


Figure 4.15: Buttons shown in user and driving info screen.

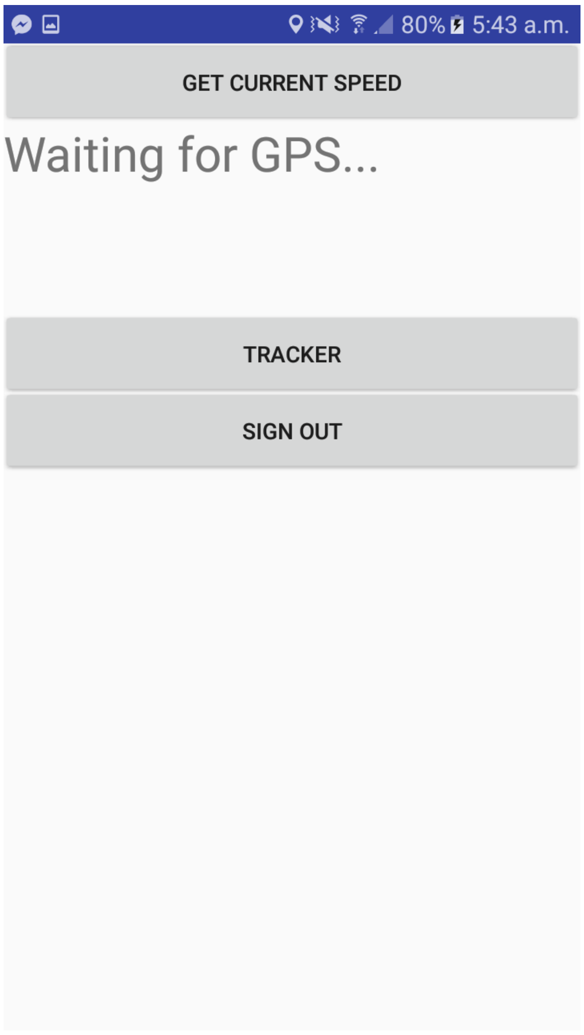
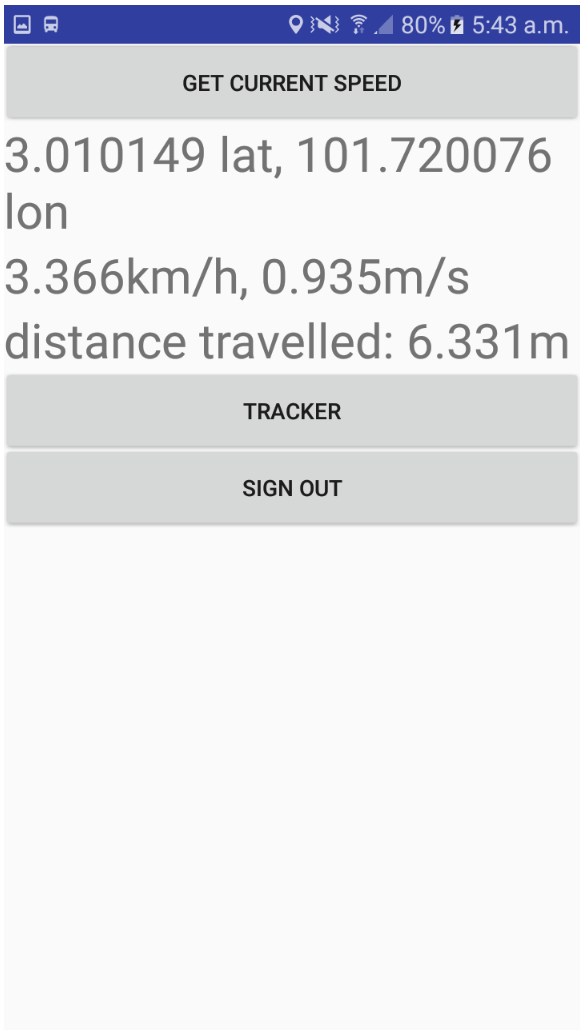
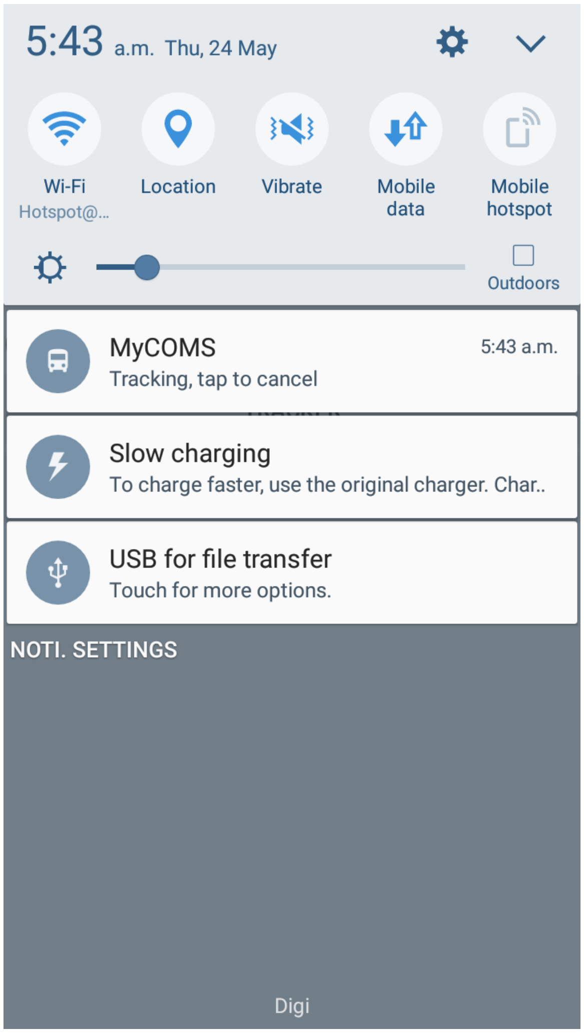


Figure 4.16: Application waiting for details of coordinate from GPS.



3.010149 lat, 101.720076 lon
3.366km/h, 0.935m/s
distance travelled: 6.331m

Figure 4.17: User's coordinate and speed is captured using GPS.

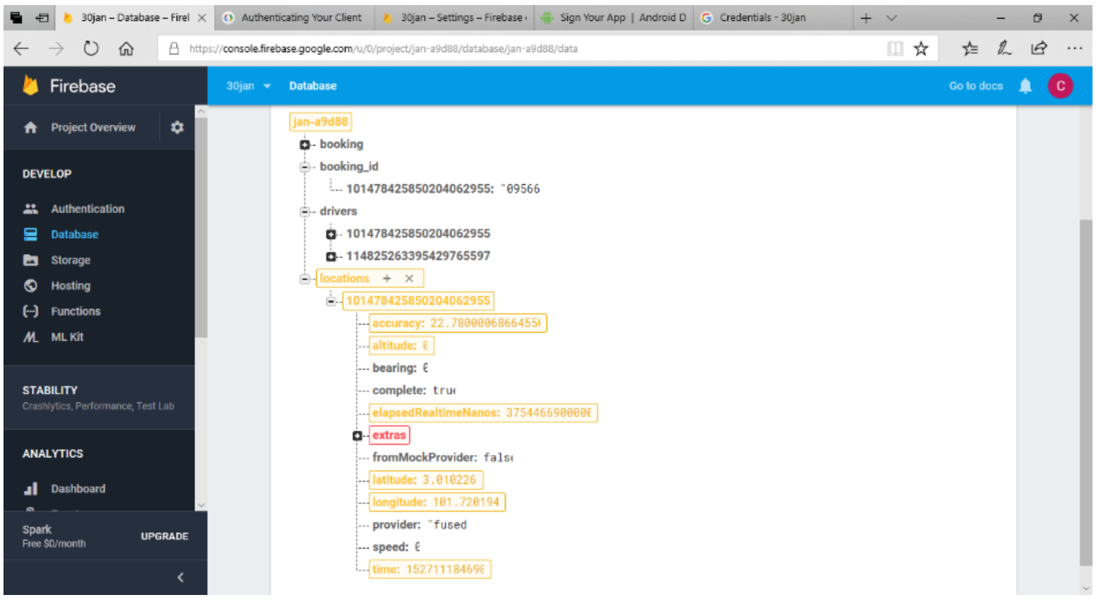


MyCOMS 5:43 a.m.
Tracking, tap to cancel

Slow charging
To charge faster, use the original charger. Char..

USB for file transfer
Touch for more options.

Figure 4.18: MyCOMS application sending GPS details to database.



https://console.firebase.google.com/u/0/project/jan-a9d88/database/jan-a9d88/data

```
jan-a9d88
  booking
  booking_id
    101478425850204062955: "09566
  drivers
    101478425850204062955
    114825263395429765597
  locations
    101478425850204062955
      accuracy: 22.7800006866455
      altitude: 0
      bearing: 0
      complete: true
      elapsedRealtimeNanos: 375446690000
      extras
      fromMockProvider: false
      latitude: 3.010226
      longitude: 101.720194
      provider: "fused
      speed: 0
      time: 152711184690
```

Figure 4.19: Details captured by database.

Figure 4.15 shows the interface of driving info screen. When user does not need navigation by google maps, user can access this button to find out the location in GPS-coordinates, speed and also the mileage travelled when the app is active as shown in figure 4.16 and figure 4.17. User will be prompt to activate the GPS function in the phone to ensure that the data is captured as shown in figure 4.18. The data retrieved from this activity will be tabled into database for administrative purpose as shown in figure 4.19. Admin can refer to the database to track the whereabouts of the user, to ensure that user does not go out of bounds of UPM Serdang campus. Algorithm is imposed to act as geo-fencing to notify user whenever user is using the app outside the campus area.

4.5 Station Level Demand Prediction

4.5.1 Case 1

The range of input as below:

- location = 11 (central region)
- population = 37
- temperature = 33°C
- time = 8pm

The defuzzified result obtained is shown in a graph:

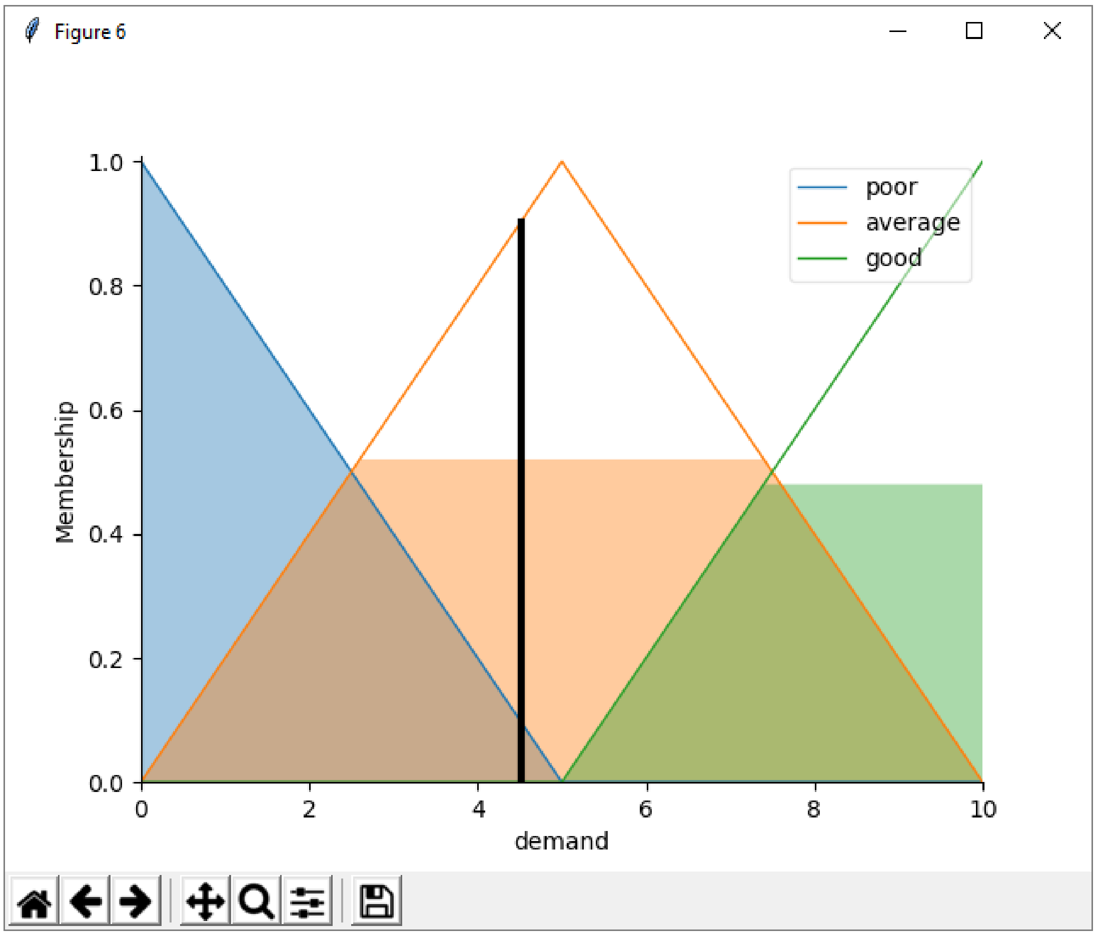


Figure 4.20: Defuzzified result of the output.

The figure 4.20 shows with the input as above, it will have a demand level of 4.5, which signifies that it has an average demand for the station.

4.5.2 Case 2

The demand level is set at 7.5, which act as a barrier to find out the best condition of all inputs, and the result is shown in figure 4.21.

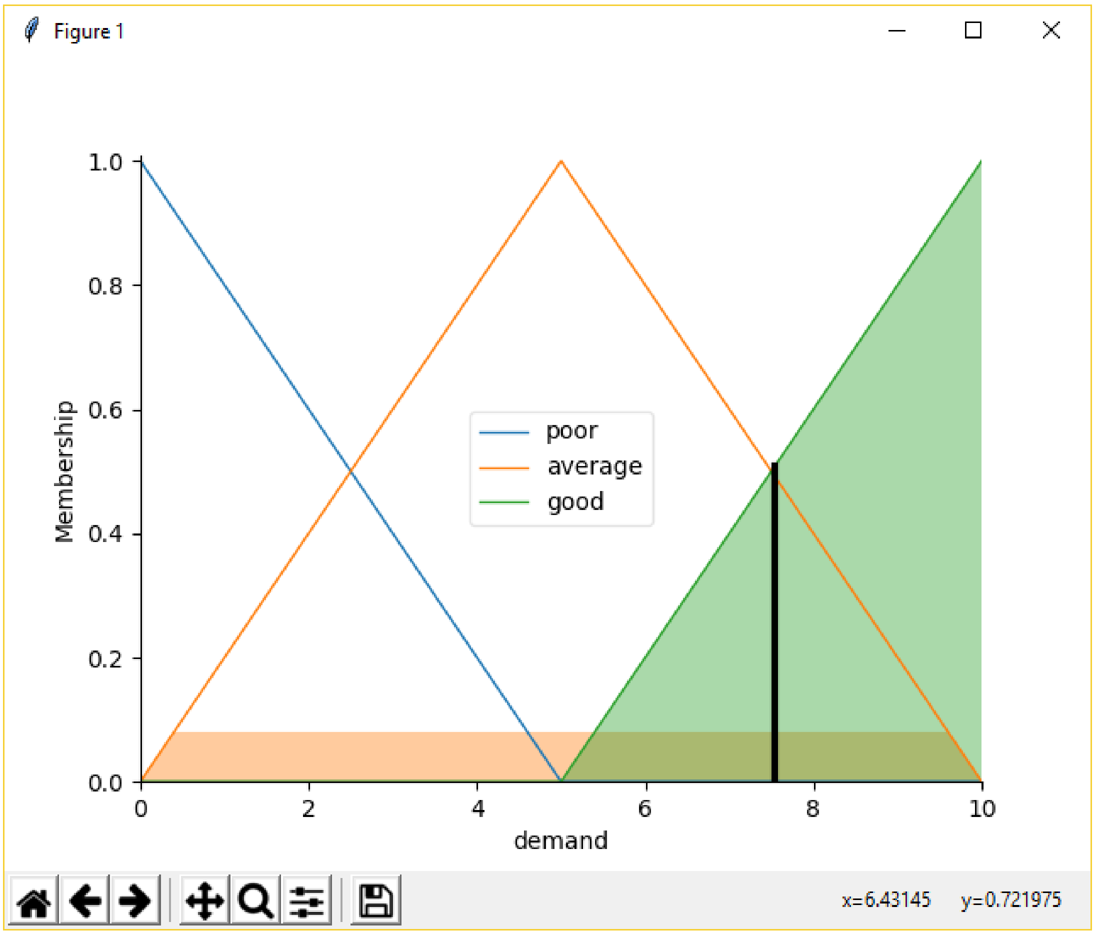


Figure 4.21: Defuzzified result of the output after loops.

After the run of the loops for the 2384th time, the system has able to generate the randomize input of:

Location	: 12 (Central Region)
Population	: 48 (out of 50 sample size)
Temperature of the day	: 35 (high region)
Time of travel	: 0 (morning region)

It is observed that the readings does not seems logical, as Malaysia at midnight 12am is not recommended to stay out in campus. It is obvious that the system will require a real-time dataset in order for it to improve. The other ways for improvement is that the system must improves its fuzzy rule in making a more detailed and precise rule which fits the system better to the real-time dataset.

4.6 Performance Testing

The MyCOMS app is tested in different type of Android devices, which are Samsung Galaxy J7, Redmi Note 4 and Asus Zenfone 5. All the features in the application are working well. However there is slight difference in the accuracy of the GPS coordinate as different devices has different sensitivity of GPS and different modules are used. User will have to wait for a few moment for the GPS sensor to calibrate the accuracy of the coordinate.

Smart phone's OS and the version of the google play services will affect the deployment of the application, as certain functions and user interface of the application is based on the SDK version of 27, which some of it might not be applicable to the previous versions. User's smartphone must be updated in terms of OS and the google play services in order for the application to run normally.

4.7 Summarize

The GPS coordinate and speed obtained from the driving info component is the latitude and longitude of the user, and estimated speed of the user based on the calculation from built-in GPS module. The location coordinate is updated constantly in database repeatedly to maintain latest information of the user and EV. The location is tracked real-time and is displayed on the map.

4.8 Limitation

This EV booking app is restricted to Android based smart phone only. This application will work when there is connection to the internet. The navigation, user info collection and also booking managing and selection requires internet access to send the data to the database. User will need internet connection to retrieve the constant update of the location of the EV before creating booking request. The location of the user can only be estimated due to the sensitivity of the GPS module from the Android mobile phone.

CHAPTER 5

CONCLUSION

5.1 Conclusion

In conclusion, an Android based single-seated vehicle booking and ride sharing system is developed and presented in a form of Android mobile application, MyCOMS. MyCOMS application is developed with real time viewing and booking of vehicle features. With all the features developed in the application, users are able to interact and experience high quality of user interface as well as minimal delays in synchronizing data with the database. Moreover, this MyCOMS application is able to increase the efficiency among students and staff of UPM travelling in between faculties and colleges with interactions made through the application.

5.2 Future Work

The MyCOMS application still has a big room for improvements. More advance features such as two-factor authentication, voice and sms for emergency purpose can be studied and developed in the future. Moreover, the user interface of the MyCOMS application can be improved by having proper icons customized specially for the application.

A website for admin should be developed in the future to ease admin in managing multiple users' account with ease. With good design and layout, the website would reduce the trouble for admin to search through the database to look for certain details. The application for the vehicle can be developed and synchronized so that the details of the vehicles such as battery level, speed and location will be automatically updated to the database.

More advanced features and better graphical user interface should be developed and designed in the future.

REFERENCES

[1] F. R. Amirul and D. Hands, "The taxi service review: Malaysia context," *Mediterranean Journal of Social Sciences*, vol. 7, no. 4, p. 559, 2016.

[2] D. Ulloa, P. Saleiro, R. J. Rossetti, and E. R. Silva, "Mining social media for open innovation in transportation systems," in *Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on.* IEEE, 2016, pp. 169–174.

[3] C. MacDonald, "Uber is built on trust [ethics opinion]," *IEEE Technology and Society Magazine*, vol. 35, no. 2, pp. 38–39, 2016.

[4] P. W. Handayani *et al.*, "Analysis on effects of brand community on brand loyalty in the social media: A case study of an online transportation (uber)," in *Advanced Computer Science and Information Systems (ICACSIS), 2016 International Conference on.* IEEE, 2016, pp. 239–244.

[5] S. Shaheen, S. Guzman, and H. Zhang, "Bikesharing in europe, the americas, and asia: past, present, and future," *Transportation Research Record: Journal of the Transportation Research Board*, no. 2143, pp. 159–167, 2010.

[6] M. Ricci, "Bike sharing: A review of evidence on impacts and processes of implementation and operation," *Research in Transportation Business & Management*, vol. 15, pp. 28–38, 2015.

[7] S. M. Salaken, M. A. Hosen, A. Khosravi, and S. Nahavandi, "Forecasting bike sharing demand using fuzzy inference mechanism," in *International Conference on Neural Information Processing.* Springer, 2015, pp. 567–574.

[8] M. E. S. Pulugurta and K.Ravinder, "Prediction of future trips using fuzzy logic based trip generation model," in *CSIR-Central Road Research Institute*, 2012.

[9] A. B. Alvarez, V. J. S. Urrutia, R. C. George, and F. J. C. Poyo, “Passenger’s flow for a train’s coach and dwelling time using fuzzy logic,” in *Bio-inspired Intelligence (IWOBI), 2014 International Work Conference on.* IEEE, 2014, pp. 30–36.

[10] A. Berbey, R. Galan, J. Sanz Bobi, and R. Caballero, “A fuzzy logic approach to modelling the passengers¡¯ flow and dwelling time,” *WIT Trans Built Environ*, vol. 128, pp. 359–369, 2012.

[11] N. Jian, D. Freund, H. M. Wiberg, and S. G. Henderson, “Simulation optimization for a large-scale bike-sharing system,” in *Winter Simulation Conference (WSC), 2016.* IEEE, 2016, pp. 602–613.

[12] O. O¡¯brien, J. Cheshire, and M. Batty, “Mining bicycle sharing data for generating insights into sustainable transport systems,” *Journal of Transport Geography*, vol. 34, pp. 262–273, 2014.

[13] Z. Li, J. Zhang, J. Gan, P. Lu, and F. Lin, “Large-scale trip planning for bike-sharing systems,” in *2017 IEEE 14th International Conference on Mobile Ad Hoc and Sensor Systems (MASS).* IEEE, 2017, pp. 328–332.

[14] H. Xu and J. Ying, “An improved grasp for the bike-sharing rebalancing problem,” in *Smart Grid and Electrical Automation (ICSGEA), 2017 International Conference on.* IEEE, 2017, pp. 324–328.

[15] T. A. B. Japan, “Corporate social responsibility report 2013,” in *http://www.toyota-body.co.jp/english/csr/2013.html*, 2017.

[16] R. Kodama, M. Koge, S. Taguchi, and H. Kajimoto, “Coms-vr: Mobile virtual reality entertainment system using electric car and head-mounted display,” in *3D User Interfaces (3DUI), 2017 IEEE Symposium on.* IEEE, 2017, pp. 130–133.

[17] N. Wu, D. Ai, H. Ogai, and S. Tateno, "Vehicle to vehicle communication and platooning for sev coms by wireless sensor network," in *SICE Annual Conference (SICE), 2014 Proceedings of the.* IEEE, 2014, pp. 566–571.

[18] N. Wu, Y. Lou, N. Sun, W. Wang, and H. Ogai, "Automatic driving system by small electric vehicle for elderly person," in *SICE Annual Conference (SICE), 2012 Proceedings of.* IEEE, 2012, pp. 232–235.

[19] G. Hill, P. Blythe, and V. Suresh, "Tracking and managing real world electric vehicle power usage and supply," 2012.

[20] C. Walls, "Get started with android," in *https://www.ecnmag.com/article/2011/07/get-started-android*, 2017.

[21] M. Cleron, "Android announces support for kotlin," in *https://android-developers.googleblog.com/2017/05/android-announces-support-for-kotlin.html*, 2017.

[22] M. R. A. Fuad and M. Drieberg, "Remote vehicle tracking system using gsm modem and google map," in *Sustainable Utilization and Development in Engineering and Technology (CSUDET), 2013 IEEE Conference on.* IEEE, 2013, pp. 15–19.

[23] S. Pethakar, N. Srivastava, and S. Suryawanshi, "Gps and gsm based vehicle tracing and employee security system," *International Journal of Computer Applications*, vol. 62, no. 6, 2013.

[24] A. I. Yaqzan, I. W. Damaj, and R. N. Zantout, "Gps-based vehicle tracking system-on-chip," in *Proceedings of the world Congress on Engineering*, vol. 1, 2008, pp. 2–4.

[25] M. N. Z. Juhari and H. Mansor, "Iium bus on campus monitoring system," in *Computer and Communication Engineering (ICCCE), 2016 International Conference on.* IEEE, 2016, pp. 138–143.

APPENDIX

MainActivity.java

```
package com.example.edwin.fyp;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.View;
import android.view.Window;
import android.widget.Button;
import android.widget.TextView;
import com.google.android.gms.auth.api.signin.GoogleSignIn
    ;
import com.google.android.gms.auth.api.signin.
    GoogleSignInAccount;
import com.google.android.gms.auth.api.signin.
    GoogleSignInClient;
import com.google.android.gms.auth.api.signin.
    GoogleSignInOptions;
import com.google.android.gms.common.SignInButton;
import com.google.android.gms.common.api.ApiException;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthCredential;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.auth.GoogleAuthProvider;
import com.q42.android.scrollingimageview.
    ScrollingImageView;

public class MainActivity extends AppCompatActivity {
  private static final String TAG = MainActivity.class.
      getSimpleName();
  private FirebaseAuth mAuth; private static final int
      RC_SIGN_IN = 9001;
  TextView mStatusTextView,mDetailTextView;
  Button mgoButton;
  GoogleSignInClient mGoogleSignInClient;
  String personName,personGivenName,personFamilyName,
      personEmail,personId;
  Uri personPhoto;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    this.requestWindowFeature(Window.FEATURE_NO_TITLE);
    super.onCreate(savedInstanceState); setContentView(R.
        layout.activity_main);
    // scrolling for image 'van' ScrollingImageView
        scrollingBackground = findViewById(R.id.
        scrolling_background); scrollingBackground.start();
    //to stop use below code
    //scrollingBackground.stop();
    mAuth = FirebaseAuth.getInstance();
    mStatusTextView = findViewById(R.id.status);
    mgoButton = findViewById(R.id.go_to_home);
    GoogleSignInOptions gso = new GoogleSignInOptions.
        Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
```

```
            .requestIdToken(getString(R.string.
                default_web_client_id))
            .requestEmail()
            .build();
        mGoogleSignInClient = GoogleSignIn.getClient(this, gso
            );
        SignInButton signInButton = findViewById(R.id.
            sign_in_button);
        signInButton.setSize(SignInButton.SIZE_STANDARD);
        findViewById(R.id.sign_in_button).setOnClickListener(
            signinOnclickListener);
        findViewById(R.id.sign_out_and_disconnect).
            setOnClickListener(signoutOnclickListener); }

    private View.OnClickListener signinOnclickListener = new
        View.OnClickListener(){
        @Override
        public void onClick(View v) {
            switch (v.getId()) {
            case R.id.sign_in_button: signIn();
            break;// ... } } };

    private View.OnClickListener signoutOnclickListener =
        new View.OnClickListener(){
        @Override
        public void onClick(View v) {
            switch (v.getId()) {
            case R.id.sign_out_and_disconnect: signOut();
            break; // ... }
        Intent homeIntent = new Intent(Intent.ACTION_MAIN);
        homeIntent.addCategory( Intent.CATEGORY_HOME );
        homeIntent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        startActivity(homeIntent); } };
    @Override
    public void onStart() {
      super.onStart();
      // Check if user is signed in (non-null) and update UI
          accordingly.
      FirebaseUser currentUser = mAuth.getCurrentUser();
      updateUIFire(currentUser);
      GoogleSignInAccount account = GoogleSignIn.
         getLastSignedInAccount(this);
      if (account != null) {
      personName = account.getDisplayName();
      personGivenName = account.getGivenName();
      personFamilyName = account.getFamilyName();
      personEmail = account.getEmail();
      personId = account.getId();
      personPhoto = account.getPhotoUrl(); }
      updateUI(account); }
    private void signIn() {
      Intent signInIntent = mGoogleSignInClient.
         getSignInIntent();
      startActivityForResult(signInIntent, RC_SIGN_IN); }
    @Override
    public void onActivityResult(int requestCode, int
       resultCode, Intent data) {
      super.onActivityResult(requestCode, resultCode, data);
      // Result returned from launching the Intent from
          GoogleSignInClient.getSignInIntent(...);
      if (requestCode == RC_SIGN_IN) {
        // The Task returned from this call is always
            completed, no need to attach
        // a listener.
        Task<GoogleSignInAccount> task = GoogleSignIn.
            getSignedInAccountFromIntent(data);
        handleSignInResult(task); } }
    private void handleSignInResult(Task<GoogleSignInAccount
        > completedTask) {
      try { GoogleSignInAccount account = completedTask.
         getResult(ApiException.class);
        firebaseAuthWithGoogle(account);
        // Signed in successfully, show authenticated UI.
            updateUI(account); }
      catch (ApiException e) {
        // The ApiException status code indicates the
            detailed failure reason.
        // Please refer to the GoogleSignInStatusCodes class
             reference for more information.
        Log.w(TAG, "signInResult:failed code=" + e.
            getStatusCode());
        updateUI(null); } }
    private void updateUI(GoogleSignInAccount account) {
      if (account != null) {
        mStatusTextView.setText(getString(R.string.
            signed_in_fmt, account.getDisplayName()));
        findViewById(R.id.sign_in_button).setVisibility(View
            .GONE);
        findViewById(R.id.go_to_home).setVisibility(View.
            VISIBLE);
        findViewById(R.id.sign_out_and_disconnect).
            setVisibility(View.VISIBLE); }
      else {
        findViewById(R.id.sign_in_button).setVisibility(View
            .VISIBLE);
        findViewById(R.id.go_to_home).setVisibility(View.
            GONE);
        findViewById(R.id.sign_out_and_disconnect).
            setVisibility(View.GONE); } }
    private void updateUIFire(FirebaseUser user) {
      if (user != null) {
        mStatusTextView.setText(getString(R.string.
            signed_in_fmt, user.getEmail()));
        findViewById(R.id.sign_in_button).setVisibility(View
            .GONE);
        findViewById(R.id.go_to_home).setVisibility(View.
            VISIBLE);
        findViewById(R.id.sign_out_and_disconnect).
            setVisibility(View.VISIBLE); }
      else {
        findViewById(R.id.sign_in_button).setVisibility(View
            .VISIBLE);
        findViewById(R.id.go_to_home).setVisibility(View.
            GONE);
        findViewById(R.id.sign_out_and_disconnect).
            setVisibility(View.GONE); } }
```

```
private void firebaseAuthWithGoogle(GoogleSignInAccount
    acct) {
  Log.d(TAG, "firebaseAuthWithGoogle:" + acct.getId());
  AuthCredential credential = GoogleAuthProvider.
      getCredential(acct.getIdToken(), null);
  mAuth.signInWithCredential(credential).
      addOnCompleteListener(this, new OnCompleteListener<
      AuthResult>() {
    @Override
    public void onComplete(@NonNull Task<AuthResult>
        task) {
      if (task.isSuccessful()) {
        // Sign in success, update UI with the signed-in
            user's information
        Log.d(TAG, "signInWithCredential:success");
        FirebaseUser user = mAuth.getCurrentUser();
        updateUIFire(user); }
      else {
        // If sign in fails, display a message to the
            user.
        Log.w(TAG, "signInWithCredential:failure", task.
            getException());
        updateUIFire(null); }
        // ... } }); }
private void signOut() {
  mGoogleSignInClient.signOut().addOnCompleteListener(
      this, new OnCompleteListener<Void>() {
  @Override
    public void onComplete(@NonNull Task<Void> task) {
        // ... mAuth.signOut(); } }); }
  public void onGoHome(View view) {
    Intent intent = new Intent(this, HomeActivity.class);
        startActivity(intent); }
}
```

HomeActivity.java

```
package com.example.edwin.fyp;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle; import android.view.View;
public class HomeActivity extends AppCompatActivity {
    @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_home);      }
    public void onStartDriving(View view) {
    Intent intent = new Intent(this, MapsActivity.class);
    startActivity(intent);      }
    public void onBookARide(View view) {
    Intent intent = new Intent(this, BookRideActivity.
        class);
    startActivity(intent);      }
    public void onViewBooking(View view) {
    Intent intent = new Intent(this, ViewBookingActivity.
        class);
    startActivity(intent);      }
    public void onDrivingInfo(View view) {
    Intent intent = new Intent(this, DrivingInfoActivity.
        class);
    startActivity(intent);     } }
```

MapsActivity.java

```
package com.example.edwin.fyp;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.Bundle;
import android.support.v4.app.FragmentActivity;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.
    BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
public class MapsActivity extends FragmentActivity
    implements OnMapReadyCallback {
private GoogleMap mMap;
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_maps);
// Obtain the SupportMapFragment and get notified when the
    map is ready to be used.
SupportMapFragment mapFragment = (SupportMapFragment)
    getSupportFragmentManager()
.findFragmentById(R.id.map);
mapFragment.getMapAsync(this);
}
```

```
/**
* Manipulates the map once available.
* This callback is triggered when the map is ready to be
    used.
* This is where we can add markers or lines, add listeners
     or move the camera. In this case,
* we just add a marker near Sydney, Australia.
* If Google Play services is not installed on the device,
    the user will be prompted to install
* it inside the SupportMapFragment. This method will only
    be triggered once the user has
* installed Google Play services and returned to the app.
*/
@Override
public void onMapReady(GoogleMap googleMap) {
float zoomLevel = 16.0f;
mMap = googleMap;
LatLng C1 = new LatLng(3.0061, 101.7199);
mMap.addMarker(new MarkerOptions()
.title("C1")
.position(C1).icon(BitmapDescriptorFactory.fromBitmap(
    resizeMapIcons("mycoms",75,75))));
mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(C1,
    zoomLevel));
LatLng C2 = new LatLng(3.0061, 101.7199);
mMap.addMarker(new MarkerOptions()
.title("C2")
.position(C2).icon(BitmapDescriptorFactory.fromBitmap(
    resizeMapIcons("mycoms",75,75))));
mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(C2,
    zoomLevel));
LatLng C3 = new LatLng(2.999652, 101.708264);
mMap.addMarker(new MarkerOptions()
.title("C3")
.position(C3).icon(BitmapDescriptorFactory.fromBitmap(
    resizeMapIcons("mycoms",75,75))));
mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(C3,
    zoomLevel));
LatLng C4 = new LatLng(3.0102375, 101.7202033);
mMap.addMarker(new MarkerOptions()
.title("C4")
.position(C4).icon(BitmapDescriptorFactory.fromBitmap(
    resizeMapIcons("mycoms",75,75))));
mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(C4,
    zoomLevel));
}
public Bitmap resizeMapIcons(String iconName, int width,
    int height){
Bitmap imageBitmap = BitmapFactory.decodeResource(
    getResources(),getResources().getIdentifier(iconName, "
    drawable", getPackageName()));
Bitmap resizedBitmap = Bitmap.createScaledBitmap(
    imageBitmap, width, height, false);
return resizedBitmap;
}
}
```

BookRideActivity.java

```
package com.example.edwin.fyp;
```

```
import android.app.DialogFragment;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import com.google.android.gms.auth.api.signin.GoogleSignIn
    ;
import com.google.android.gms.auth.api.signin.
    GoogleSignInAccount;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import java.security.SecureRandom;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Locale;
public class BookRideActivity extends AppCompatActivity {
TextView show_booking_time;
Button timeBtn, mbook_ride_button;
EditText marrival, mdeparture;
String ArrivalHolder, DepartureHolder;
String personName,personGivenName,personFamilyName,
    personEmail,personId;
```

```
Uri personPhoto;
String path;
DatabaseReference mRootRef = FirebaseDatabase.getInstance
    ().getReference();
DatabaseReference mConditionRef;
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_book_ride);
mbook_ride_button = findViewById(R.id.book_ride_button);
marrival = findViewById(R.id.arrival);
mdeparture = findViewById(R.id.departure);
SecureRandom random = new SecureRandom();
int num = random.nextInt(100000);
final String formatted_id = String.format("%05d", num);
GoogleSignInAccount account = GoogleSignIn.
    getLastSignedInAccount(this);
if (account != null) {
personName = account.getDisplayName();
personGivenName = account.getGivenName();
personFamilyName = account.getFamilyName();
personEmail = account.getEmail();
personId = account.getId();
personPhoto = account.getPhotoUrl();
}
String date = new SimpleDateFormat("yyyy-MM-dd", Locale.
    getDefault()).format(new Date());
path = "booking/"+date+"/"+personId;
mbook_ride_button.setOnClickListener(new View.
    OnClickListener(){
@Override
public void onClick(View view){
GetDataFromEditText();
mConditionRef = mRootRef.child(path+"/booking_id");
mConditionRef.setValue(formatted_id);
mConditionRef = mRootRef.child("booking_id/"+personId);
mConditionRef.setValue(formatted_id);
mConditionRef = mRootRef.child(path+"/Name");
mConditionRef.setValue(personName);
String a_details = ArrivalHolder;
mConditionRef = mRootRef.child(path+"/arrival");
mConditionRef.setValue(a_details);
String d_details = DepartureHolder;
mConditionRef = mRootRef.child(path+"/departure");
mConditionRef.setValue(d_details);
mConditionRef = mRootRef.child(path+"/status");
mConditionRef.setValue("pending");
Toast.makeText(BookRideActivity.this,"Data Inserted
    Successfully into Firebase Database", Toast.LENGTH_LONG
    ).show();
startActivity(new Intent(BookRideActivity.this,
    HomeActivity.class));
}
});
}
@Override
public boolean onCreateOptionsMenu(Menu menu) {
// Inflate the menu; this adds items to the action bar if
```

```
    it is present.
//getMenuInflater().inflate(R.menu.menu_main, menu);
return false;
}
@Override
public boolean onOptionsItemSelected(MenuItem item) {
// Handle action bar item clicks here. The action bar will
// automatically handle clicks on the Home/Up button, so
    long
// as you specify a parent activity in AndroidManifest.xml
    .
int id = item.getItemId();
//noinspection SimplifiableIfStatement
if (id == R.id.action_settings) {
return true;
}
return super.onOptionsItemSelected(item);
}
public void showTimePickerDialog(View v){
DialogFragment newFragment = new TimePickerFragment();
newFragment.show(getFragmentManager(),"timePicker");
}
public void GetDataFromEditText(){
ArrivalHolder = marrival.getText().toString().trim();
DepartureHolder = mdeparture.getText().toString().trim();
}
}
```

TimePickerFragment.java

```
package com.example.edwin.fyp;
```

```java

import android.app.Dialog;
import android.app.DialogFragment;
import android.app.TimePickerDialog;
import android.os.Bundle;
import android.text.format.DateFormat;
import android.widget.Button;
import android.widget.TextView;
import android.widget.TimePicker;

import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.Date;
import java.util.Locale;

public class TimePickerFragment extends DialogFragment
    implements TimePickerDialog.OnTimeSetListener{
String show_booking_time_on_app;
Button timeBtn;
OnTimePickedListener mCallback;
DatabaseReference mRootRef = FirebaseDatabase.getInstance
    ().getReference();
DatabaseReference mConditionRef = mRootRef.child("booking/
    time");

public interface OnTimePickedListener {
```

```
public void onTimePicked(int hour, int minute);
}

@Override
public Dialog onCreateDialog(Bundle savedInstanceState) {
final Calendar c = Calendar.getInstance();
int hour = c.get(Calendar.HOUR_OF_DAY);
int minute = c.get(Calendar.MINUTE);

return new TimePickerDialog(getActivity(),this,hour,minute
    , DateFormat.is24HourFormat(getActivity()));
}

@Override
public void onTimeSet(TimePicker view, int hourOfDay, int
    minute) {
TextView tv1= getActivity().findViewById(R.id.
    show_booking_time);
tv1.setText(getString(R.string.booking_time_is, "Hour: " +
     view.getCurrentHour() + " Minute: " + view.
    getCurrentMinute()));
show_booking_time_on_app = "Hour: " + view.getCurrentHour
    () + " Minute: " + view.getCurrentMinute();
String date = new SimpleDateFormat("yyyy-MM-dd", Locale.
    getDefault()).format(new Date());
String path = "booking/"+date;
mConditionRef = mRootRef.child(path+"/Time");
mConditionRef.setValue(show_booking_time_on_app);
}
```

```
}
```

ViewBookingActivity.java

```
package com.example.edwin.fyp;

import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.auth.api.signin.GoogleSignIn
    ;
import com.google.android.gms.auth.api.signin.
    GoogleSignInAccount;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.Query;

import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Locale;

public class ViewBookingActivity extends AppCompatActivity
    {
```

```

DatabaseReference mRootRef = FirebaseDatabase.getInstance
    ().getReference();
DatabaseReference mDepartureRef,mArrivalRef,
    mshow_booking_timeRef,mStatusRef,mDeleteRef; //=
    mRootRef.child("condition");

TextView mDeparture, mArrival, mshow_booking_time,
    mstatus_view;
Button mConfirm,mDelete;
String personName,personGivenName,personFamilyName,
    personEmail,personId;
Uri personPhoto;
String path;

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_view_booking);
final String date = new SimpleDateFormat("yyyy-MM-dd",
    Locale.getDefault()).format(new Date());
path = "booking/"+date;
mDeparture = findViewById(R.id.departure);
mArrival = findViewById(R.id.arrival);
mshow_booking_time = findViewById(R.id.show_booking_time);
mstatus_view = findViewById(R.id.status_view);
mConfirm = findViewById(R.id.confirm);
mDelete = findViewById(R.id.delete_book);

```

```
GoogleSignInAccount account = GoogleSignIn.
    getLastSignedInAccount(this);
if (account != null) {
personName = account.getDisplayName();
personGivenName = account.getGivenName();
personFamilyName = account.getFamilyName();
personEmail = account.getEmail();
personId = account.getId();
personPhoto = account.getPhotoUrl();
}

mConfirm.setOnClickListener(new View.OnClickListener(){
@Override
public void onClick(View view){
Toast.makeText(ViewBookingActivity.this,"You have
    confirmed your ride. Enjoy your drive!", Toast.
    LENGTH_LONG).show();
startActivity(new Intent(ViewBookingActivity.this,
    HomeActivity.class));
}
});

mDelete.setOnClickListener(new View.OnClickListener(){
@Override
public void onClick(View view){
mDeleteRef =mRootRef.child("booking/date");
mDeleteRef.removeValue();

Toast.makeText(ViewBookingActivity.this,"You have deleted
    your ride. Please book again!!", Toast.LENGTH_LONG).
    show();
startActivity(new Intent(ViewBookingActivity.this,
    HomeActivity.class));
}
});
}

@Override
protected void onStart() {
super.onStart();
Query mIDRef = mRootRef.child(path+"/ID").equalTo(personId
    );
mIDRef.addListenerForSingleValueEvent(new com.google.
    firebase.database.ValueEventListener() {
@Override
public void onDataChange(DataSnapshot dataSnapshot) {
//String text = dataSnapshot.getValue(String.class);

mDepartureRef = mRootRef.child(path+"/departure");
mDepartureRef.addValueEventListener(new com.google.
    firebase.database.ValueEventListener() {
@Override
public void onDataChange(DataSnapshot dataSnapshot) {
String text = dataSnapshot.getValue(String.class);
mDeparture.setText(getString(R.string.departure_view,text)
    );
}

```

```
@Override
public void onCancelled(DatabaseError databaseError) {
}
});

mArrivalRef = mRootRef.child(path+"/arrival");
mArrivalRef.addValueEventListener(new com.google.firebase.
    database.ValueEventListener() {
@Override
public void onDataChange(DataSnapshot dataSnapshot) {
String text = dataSnapshot.getValue(String.class);
mArrival.setText(getString(R.string.arrival_view,text));
}

@Override
public void onCancelled(DatabaseError databaseError) {
}
});

mshow_booking_timeRef = mRootRef.child(path+"/Time");
mshow_booking_timeRef.addValueEventListener(new com.google
    .firebase.database.ValueEventListener() {
@Override
public void onDataChange(DataSnapshot dataSnapshot) {
String text = dataSnapshot.getValue(String.class);
mshow_booking_time.setText(getString(R.string.
    booking_time_view,text));
}

```

```
@Override
public void onCancelled(DatabaseError databaseError) {
}
});

mStatusRef = mRootRef.child(path+"/status");
mStatusRef.addValueEventListener(new com.google.firebase.
    database.ValueEventListener() {
@Override
public void onDataChange(DataSnapshot dataSnapshot) {
String text = dataSnapshot.getValue(String.class);
mstatus_view.setText(getString(R.string.booking_status,
    text));
if (text.equals("approved")){
findViewById(R.id.confirm).setVisibility(View.VISIBLE);
}
if (text.equals("pending")){
findViewById(R.id.confirm).setVisibility(View.GONE);
}
}

@Override
public void onCancelled(DatabaseError databaseError) {
}
});
}

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

```java
}
});
}

public void onBack(View view) {
Intent intent = new Intent(this, HomeActivity.class);
startActivity(intent);
}
}
```

DrivingInfoActivity.java

```java
package com.example.edwin.fyp;

import android.app.Activity;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.Location;
import android.location.LocationManager;
import android.os.Bundle;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.view.View;
import android.widget.TextView;

import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

import java.math.BigDecimal;

public class DrivingInfoActivity extends Activity
    implements GPSCallback {
private GPSManager gpsManager = null;
private double speed = 0.0,longitude = 0.0,latitude = 0.0;
Boolean isGPSEnabled=false;
LocationManager locationManager;
double currentSpeed,kmphSpeed,newlatitude, newlongitude,
    msSpeed, distance_travelled = 0.0;
TextView txtview,txtview2,txtview3;

// Write a message to the database
FirebaseDatabase database = FirebaseDatabase.getInstance()
    ;
DatabaseReference mRootRef = database.getReference();
DatabaseReference mLocationRef = mRootRef.child("Location"
    );

@Override
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_driving_info);
txtview= findViewById(R.id.info);
txtview2 = findViewById(R.id.info2);
txtview3 = findViewById(R.id.info3);
try {
if (ContextCompat.checkSelfPermission(
    getApplicationContext(), android.Manifest.permission.
    ACCESS_FINE_LOCATION) != PackageManager.
    PERMISSION_GRANTED ) {
ActivityCompat.requestPermissions(this, new String[]{
    android.Manifest.permission.ACCESS_FINE_LOCATION}, 101)
    ;
}
} catch (Exception e){
e.printStackTrace();
}
}

@Override
protected void onStart() {
super.onStart();

ValueEventListener valueEventListener = mLocationRef.
    addValueEventListener(new ValueEventListener() {
@Override
public void onDataChange(DataSnapshot dataSnapshot) {
//Double text = dataSnapshot.getValue(newlatitude);
}

@Override
public void onCancelled(DatabaseError databaseError) {
}
});
}

public void getCurrentSpeed(View view){
```

```
txtview.setText(getString(R.string.info));
locationManager = (LocationManager)getSystemService(
    LOCATION_SERVICE);
gpsManager = new GPSManager(DrivingInfoActivity.this);
isGPSEnabled = locationManager.isProviderEnabled(
    LocationManager.GPS_PROVIDER);
if(isGPSEnabled) {
gpsManager.startListening(getApplicationContext());
gpsManager.setGPSCallback(this);
} else {
gpsManager.showSettingsAlert();
}
}

@Override
public void onGPSUpdate(Location location) {
speed = location.getSpeed();
latitude = location.getLatitude();
longitude = location.getLongitude();
currentSpeed = round(speed, 3, BigDecimal.ROUND_HALF_UP);
kmphSpeed = round((currentSpeed * 3.6), 3, BigDecimal.
    ROUND_HALF_UP);
//txtview.setText(kmphSpeed+"km/h");
newlatitude = round(latitude, 6, BigDecimal.ROUND_HALF_UP)
    ;
newlongitude = round(longitude, 6, BigDecimal.
    ROUND_HALF_UP);
msSpeed = kmphSpeed * 1000 / 3600;
msSpeed = round(msSpeed,3,BigDecimal.ROUND_HALF_UP);
```

```
distance_travelled += msSpeed;
distance_travelled = round(distance_travelled ,3 ,BigDecimal
    .ROUND_HALF_UP);
txtview.setText(newlatitude + " lat , " + newlongitude + "
    lon");
txtview2.setText(kmphSpeed + "km/h, " + msSpeed + "m/s");
txtview3.setText("distance travelled: "+distance_travelled
    +"m");
}

@Override
protected void onDestroy() {
gpsManager.stopListening();
gpsManager.setGPSCallback(null);
gpsManager = null;
super.onDestroy();
}

public static double round(double unrounded, int precision
    , int roundingMode) {
BigDecimal bd = new BigDecimal(unrounded);
BigDecimal rounded = bd.setScale(precision , roundingMode);
return rounded.doubleValue();
}

public void onTrackerActivity(View view) {
Intent intent = new Intent(this, TrackerActivity.class);
startActivity(intent);
}
```

```
}
```

GPSManager.java

```
package com.example.edwin.fyp;

import android.Manifest;
import android.app.Activity;
import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.Criteria;
import android.location.GpsSatellite;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.provider.Settings;
import android.support.v4.app.ActivityCompat;
import android.util.Log;

import java.util.List;

public class GPSManager implements android.location.
    GpsStatus.Listener{
    private static final int gpsMinTime = 500;
    private static final int gpsMinDistance = 0;

    private static LocationManager locationManager = null;
    private static LocationListener locationListener = null;
    private static GPSCallback gpsCallback = null;
    Context mcontext;
    public GPSManager(Context context) {
        mcontext=context;
        GPSManager.locationListener = new LocationListener() {
            @Override
            public void onLocationChanged(final Location location) {
                if (GPSManager.gpsCallback != null) {
                    GPSManager.gpsCallback.onGPSUpdate(location);
                }
            }

            @Override
            public void onProviderDisabled(final String provider) {
            }

            @Override
            public void onProviderEnabled(final String provider) {
            }

            @Override
            public void onStatusChanged(final String
                provider, final int status, final Bundle
                extras) {
            }
        };
    }
    public void showSettingsAlert(){
        AlertDialog.Builder alertDialog = new AlertDialog.
            Builder(mcontext);
        // Setting Dialog Title
        alertDialog.setTitle("GPS is settings");
        // Setting Dialog Message
        alertDialog.setMessage("GPS is not enabled. Do you
             want to go to settings menu?");
        // On pressing Settings button
        alertDialog.setPositiveButton("Settings", new
            DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog,int
                 which) {
                Intent intent = new Intent(Settings.
                    ACTION_LOCATION_SOURCE_SETTINGS);
                mcontext.startActivity(intent);
            }
        });
        // on pressing cancel button
        alertDialog.setNegativeButton("Cancel", new
            DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog,
                int which) {
                dialog.cancel();
                }
            });
            // Showing Alert Message
            alertDialog.show();
        }
        public GPSCallback getGPSCallback()
        {
            return GPSManager.gpsCallback;
        }

        public void setGPSCallback(final GPSCallback
           gpsCallback) {
            GPSManager.gpsCallback = gpsCallback;
        }

        public void startListening(final Context context) {
            if (GPSManager.locationManager == null) {
                GPSManager.locationManager = (LocationManager)
                        context.getSystemService(Context.
                      LOCATION_SERVICE);
            }

            final Criteria criteria = new Criteria();

            criteria.setAccuracy(Criteria.ACCURACY_FINE);
            criteria.setSpeedRequired(true);
            criteria.setAltitudeRequired(false);
            criteria.setBearingRequired(false);
            criteria.setCostAllowed(true);
```

```java
        criteria.setPowerRequirement(Criteria.POWER_LOW);

        final String bestProvider = GPSManager.locationManager.getBestProvider(criteria, true);

        if (ActivityCompat.checkSelfPermission(context, android.Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED && ActivityCompat.checkSelfPermission(context, android.Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
          ActivityCompat.requestPermissions((Activity) context, new String[]{android.Manifest.permission.ACCESS_FINE_LOCATION, android.Manifest.permission.ACCESS_COARSE_LOCATION}, 101);
        }
        if (bestProvider != null && bestProvider.length() > 0) {
          GPSManager.locationManager.requestLocationUpdates(bestProvider, GPSManager.gpsMinTime,
                  GPSManager.gpsMinDistance, GPSManager.locationListener);
        }
        else {
          final List<String> providers = GPSManager.
```

```
                locationManager.getProviders(true);
            for (final String provider : providers)
            {
                GPSManager.locationManager.
                    requestLocationUpdates(provider,
                    GPSManager.gpsMinTime,
                        GPSManager.gpsMinDistance,
                            GPSManager.locationListener);
            }
        }
    }
    public void stopListening() {
        try
        {
            if (GPSManager.locationManager != null &&
                GPSManager.locationListener != null) {
                GPSManager.locationManager.removeUpdates(
                    GPSManager.locationListener);
            }
            GPSManager.locationManager = null;
        }
        catch (final Exception ex) {
            ex.printStackTrace();
        }
    }

    public void onGpsStatusChanged(int event) {
        int Satellites = 0;
        int SatellitesInFix = 0;
        if (ActivityCompat.checkSelfPermission(mcontext,
            android.Manifest.permission.
            ACCESS_FINE_LOCATION) != PackageManager.
            PERMISSION_GRANTED && ActivityCompat.
            checkSelfPermission(mcontext, android.Manifest.
            permission.ACCESS_COARSE_LOCATION) !=
            PackageManager.PERMISSION_GRANTED) {
          ActivityCompat.requestPermissions((Activity)
              mcontext, new String[]{android.Manifest.
              permission.ACCESS_FINE_LOCATION, Manifest.
              permission.ACCESS_COARSE_LOCATION}, 101);
        }
        int timetofix = locationManager.getGpsStatus(null)
            .getTimeToFirstFix();
        Log.i("GPs", "Time to first fix = "+String.valueOf
            (timetofix));
        for (GpsSatellite sat : locationManager.
            getGpsStatus(null).getSatellites()) {
          if(sat.usedInFix()) {
              SatellitesInFix++;
          }
          Satellites++;
        }
        Log.i("GPS", String.valueOf(Satellites) + " Used
            In Last Fix ("+SatellitesInFix+")");
    }
}
```

GPSCallBack.java

```
package com.example.edwin.fyp;
```

```

import android.location.Location;

public interface GPSCallback {
    void onGPSUpdate(Location location);
}
```

TrackerActivity.java

```
package com.example.edwin.fyp;

import android.Manifest;
import android.app.Activity;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.LocationManager;
import android.os.Bundle;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.widget.Toast;

public class TrackerActivity extends Activity {

    private static final int PERMISSIONS_REQUEST = 1;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // Check GPS is enabled
        LocationManager lm = (LocationManager)
```

```
            getSystemService(LOCATION_SERVICE);
        if (!lm.isProviderEnabled(LocationManager.
            GPS_PROVIDER)) {
            Toast.makeText(this, "Please enable location
                services", Toast.LENGTH_SHORT).show();
            finish();
        }

        // Check location permission is granted - if it is
            , start
        // the service, otherwise request the permission
        int permission = ContextCompat.checkSelfPermission
            (this,
                Manifest.permission.ACCESS_FINE_LOCATION);
        if (permission == PackageManager.
            PERMISSION_GRANTED) {
            startTrackerService();
        } else {
            ActivityCompat.requestPermissions(this,
                    new String[]{Manifest.permission.
                        ACCESS_FINE_LOCATION},
                    PERMISSIONS_REQUEST);
        }
    }

    private void startTrackerService() {
        startService(new Intent(this, TrackerService.class
            ));
        finish();
    }

    @Override
    public void onRequestPermissionsResult(int requestCode
        , String[] permissions, int[]
            grantResults) {
        if (requestCode == PERMISSIONS_REQUEST &&
            grantResults.length == 1
                && grantResults[0] == PackageManager.
                    PERMISSION_GRANTED) {
            // Start the service when the permission is
                granted
            startTrackerService();
        } else {
            finish();
        }
    }
}
```

TrackerService.java

```
package com.example.edwin.fyp;

import android.Manifest;
import android.app.PendingIntent;
import android.app.Service;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.content.pm.PackageManager;
import android.location.Location;
import android.net.Uri;
import android.os.IBinder;
import android.support.annotation.NonNull;
import android.support.v4.app.NotificationCompat;
import android.support.v4.content.ContextCompat;
import android.util.Log;

import com.google.android.gms.auth.api.signin.GoogleSignIn
    ;
import com.google.android.gms.auth.api.signin.
    GoogleSignInAccount;
import com.google.android.gms.location.
    FusedLocationProviderClient;
import com.google.android.gms.location.LocationCallback;
import com.google.android.gms.location.LocationRequest;
import com.google.android.gms.location.LocationResult;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthCredential;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.auth.GoogleAuthProvider;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

public class TrackerService extends Service {

    private static final String TAG = TrackerService.class
        .getSimpleName();
    String personName,personGivenName,personFamilyName,
        personEmail,personId;
    Uri personPhoto;
    private FirebaseAuth mAuth;


    @Override
    public IBinder onBind(Intent intent) {return null;}

    @Override
    public void onCreate() {
        super.onCreate();
        mAuth = FirebaseAuth.getInstance();
        GoogleSignInAccount account = GoogleSignIn.
            getLastSignedInAccount(this);
        if (account != null) {
            personName = account.getDisplayName();
            personGivenName = account.getGivenName();
            personFamilyName = account.getFamilyName();
            personEmail = account.getEmail();
            personId = account.getId();
            personPhoto = account.getPhotoUrl();
        }
        buildNotification();
        firebaseAuthWithGoogle();
        //loginToFirebase();
    }

    private void buildNotification() {
        String stop = "stop";
        registerReceiver(stopReceiver, new IntentFilter(
            stop));
        PendingIntent broadcastIntent = PendingIntent.
            getBroadcast(
                this, 0, new Intent(stop), PendingIntent.
                    FLAG_UPDATE_CURRENT);
        // Create the persistent notification
        NotificationCompat.Builder builder = new
            NotificationCompat.Builder(this)
                .setContentTitle(getString(R.string.
                    app_name))
                .setContentText(getString(R.string.
                    notification_text))
                .setOngoing(true)
                .setContentIntent(broadcastIntent)
                .setSmallIcon(R.drawable.ic_tracker);
        startForeground(1, builder.build());
    }

    protected BroadcastReceiver stopReceiver = new
        BroadcastReceiver() {
        @Override
        public void onReceive(Context context, Intent
            intent) {
            Log.d(TAG, "received stop broadcast");
```

```
                // Stop the service when the notification is
                    tapped
                unregisterReceiver(stopReceiver);
                stopSelf();
            }
        };
        private void firebaseAuthWithGoogle() {
            GoogleSignInAccount acct = GoogleSignIn.
                getLastSignedInAccount(this);
            Log.d(TAG, "firebaseAuthWithGoogle:" + acct.getId
                ());

            AuthCredential credential = GoogleAuthProvider.
                getCredential(acct.getIdToken(), null);
            mAuth.signInWithCredential(credential)
                    .addOnCompleteListener(new
                        OnCompleteListener<AuthResult>() {
                        @Override
                        public void onComplete(@NonNull Task<
                            AuthResult> task) {
                            if (task.isSuccessful()) {
                                // Sign in success, update UI
                                    with the signed-in user's
                                    information
                                Log.d(TAG, "
                                    signInWithCredential:
                                    success");
                                FirebaseUser user = mAuth.
                                    getCurrentUser();
                                ////updateUIFire(user);
                                requestLocationUpdates();
                            } else {
                                // If sign in fails, display
                                    a message to the user.
                                Log.w(TAG, "
                                    signInWithCredential:
                                    failure", task.
                                    getException());
                                //updateUIFire(null);
                            }

                            // ...
                        }
                    });
        }

        private void requestLocationUpdates() {
            LocationRequest request = new LocationRequest();
            request.setInterval(10000);
            request.setFastestInterval(5000);
            request.setPriority(LocationRequest.
                        PRIORITY_HIGH_ACCURACY);
            FusedLocationProviderClient client =
                        LocationServices.getFusedLocationProviderClient
                        (this);
            final String path = getString(R.string.
                        firebase_path) + "/" + personId;//getString(R.
                        string.transport_id);
        int permission = ContextCompat.checkSelfPermission
            (this,
                Manifest.permission.ACCESS_FINE_LOCATION);
        if (permission == PackageManager.
            PERMISSION_GRANTED) {
            // Request location updates and when an update
                is
            // received, store the location in Firebase
            client.requestLocationUpdates(request, new
                LocationCallback() {
                @Override
                public void onLocationResult(
                    LocationResult locationResult) {
                    DatabaseReference ref =
                        FirebaseDatabase.getInstance().
                        getReference(path);
                    Location location = locationResult.
                        getLastLocation();
                    if (location != null) {
                        Log.d(TAG, "location update " +
                            location);
                        ref.setValue(location);
                    }
                }
            }, null);
        }
    }

    public int onStartCommand(Intent intent, int flags,
        int startId) {

        personName = intent.getStringExtra("
            EXTRA_DISPLAY_NAME");

        //do something

        return START_STICKY;

    }
}
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/
    res/android"
    package="com.example.edwin.fyp">

    <!-- To auto-complete the email text field in the
        login form with the user's emails -->
    <uses-permission android:name="android.permission.
        GET_ACCOUNTS" />
    <uses-permission android:name="android.permission.
        READ_PROFILE" />
    <uses-permission android:name="android.permission.
        READ_CONTACTS" />

    <!--
        The ACCESS_COARSE/FINE_LOCATION permissions are
            not required to use
        Google Maps Android API v2, but you must specify
            either coarse or fine
        location permissions for the 'MyLocation'
            functionality.
    -->
    <uses-permission android:name="android.permission.
        ACCESS_FINE_LOCATION" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.
                    action.MAIN" />

                <category android:name="android.intent.
                    category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".LoginActivitySample"
            android:label="@string/
                title_activity_login_sample" />
        <activity android:name=".SignUpActivity" />
        <activity android:name=".HomeActivity" />
        <!--
             The API key for Google Maps-based APIs is defined as a string resource.
             (See the file "res/values/google_maps_api.xml").
             Note that the API key is linked to the encryption key used to sign the APK.
             You need a different API key for each encryption key, including the release key that is used to
             sign the APK for publishing.
             You can define the keys for the debug and release targets in src/debug/ and src/release/.
        -->
        <meta-data
            android:name="com.google.android.geo.API_KEY"
            android:value="@string/google_maps_key" />

        <activity
            android:name=".MapsActivity"
            android:label="@string/title_activity_maps" />
        <activity android:name=".BookRideActivity" />
        <activity android:name=".ViewBookingActivity" />
        <activity android:name=".DrivingInfoActivity" />
        <activity android:name=".TrackerActivity" />

        <service
```

```
            android:name=".TrackerService"
            android:enabled="true"
            android:exported="true"></service>
    </application>

</manifest>
```

strings.xml

```
<resources>
    <string name="app_name">MyCOMS</string>
    <string name="sign_in">Sign In</string>
    <string name="sign_up">Sign Up</string>
    <string name="lorem_ipsum">MyCOMS application ease
        your travel within UPM Campus, anytime to anywhere!
          "\n\n"** Please sign in or sign up using your UPM
            email address.</string>


    <string name="title_activity_login">Sign in</string>
    <string name="title_activity_login_sample">Sign in</
        string>

    <!-- Strings related to login -->
    <string name="prompt_email">Email</string>
    <string name="prompt_password">Password (optional)</
        string>
    <string name="action_sign_in">Sign in</string>
    <string name="action_sign_in_short">Sign in</string>
    <string name="error_invalid_email">This email address
        is invalid</string>
    <string name="error_invalid_password">This password is
        too short</string>
    <string name="error_incorrect_password">This password
        is incorrect</string>
    <string name="error_field_required">This field is
        required</string>
    <string name="permission_rationale">"Contacts
        permissions are needed for providing email
         completions."
    </string>
    <string name="action_sign_up">Sign Up</string>
    <string name="action_sign_up_short">Sign Up</string>
    <string name="confirm_password">Confirm Password</
        string>
    <string name="matrik">Matrik Number</string>
    <string name="first_name">First Name</string>
    <string name="last_name">Last Name</string>
    <string name="start_driving">Start Driving</string>
    <string name="book_a_ride">Book A Ride</string>
    <string name="title_activity_navigation">
        NavigationActivity</string>

    <string name="navigation_drawer_open">Open navigation
        drawer</string>
    <string name="navigation_drawer_close">Close
        navigation drawer</string>

    <string name="action_settings">Settings</string>
    <string name="title_activity_maps">Map</string>
    <string name="departure">Departure</string>
    <string name="arrival">Arrival</string>
    <string name="book_ride">Book Ride</string>
    <string name="Manage_booking">MANAGE BOOKING</string>
    <string name="booking_time">Booking Time</string>
    <string name="booking_time_is">Booking time is : %s</
        string>
    <string name="user_details">USER DETAILS</string>
    <string name="get_current_speed">Get Current Speed</
        string>
    <string name="info">Waiting for GPS...</string>
    <string name="signed_out">Sign Out</string>
    <string name="go_to_home">Go!</string>
    <string name="signed_in_fmt">Signed in as: %s</string>
    <string name="firebase_status_fmt">Firebase User: %s</
        string>

    <string name="notification_text">Tracking, tap to
        cancel</string>
    <string name="transport_id" translatable="false">123</
        string>
    <string name="firebase_path" translatable="false">
        locations</string>
    <string name="firebase_email" translatable="false">
        test@example.com</string>
    <string name="firebase_password" translatable="false">
        password</string>
    <string name="current_hour">00</string>
    <string name="current_minute">00</string>
    <string name="booking_status">Your booking status is %
        s</string>
    <string name="departure_view">d = %s</string>
    <string name="arrival_view">a = %s</string>
    <string name="booking_time_view">t = %s</string>
</resources>
```

Activity_book_ride.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/
    apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.edwin.fyp.BookRideActivity"
        >

    <ProgressBar
        android:id="@+id/book_ride_progress"
        style="?android:attr/progressBarStyleLarge"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="8dp"
        android:visibility="gone" />

    <ScrollView
        android:id="@+id/book_ride_Form"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <LinearLayout
            android:id="@+id/book_ride_location_form"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">

            <android.support.design.widget.TextInputLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content">

                <EditText
                    android:id="@+id/departure"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:hint="@string/departure"
                    android:inputType="textPersonName"
                    android:maxLines="1"
                    android:singleLine="true" />

            </android.support.design.widget.
               TextInputLayout>

            <android.support.design.widget.TextInputLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content">

                <EditText
                    android:id="@+id/arrival"
```

```
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:hint="@string/arrival"
                    android:inputType="textPersonName"
                    android:maxLines="1"
                    android:singleLine="true" />


            </android.support.design.widget.
                TextInputLayout>

            <Button
                android:id="@+id/booking_time"
                style="?android:textAppearanceSmall"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="16dp"
                android:text="@string/booking_time"
                android:textStyle="bold"
                android:onClick="showTimePickerDialog"/>


            <Button
                android:id="@+id/book_ride_button"
                style="?android:textAppearanceSmall"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="16dp"
                android:text="@string/book_ride"
```

```
                    android:textStyle="bold" />

                <TextView
                    android:id="@+id/show_booking_time"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:text="@string/booking_time_is" />
            </LinearLayout>
        </ScrollView>

</LinearLayout>
```

Activity_driving_info.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/
    apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    >

    <Button
        android:id="@+id/getCurrentSpeed"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:onClick="getCurrentSpeed"
        android:text="@string/get_current_speed" />

    <TextView
        android:id="@+id/info"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="30sp" />

    <TextView
        android:id="@+id/info2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="30sp" />

    <TextView
        android:id="@+id/info3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="30sp" />

    <Button
        android:id="@+id/tracker"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:onClick="onTrackerActivity"
        android:text="tracker" />

    <Button
        android:id="@+id/sign_out_and_disconnect"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Sign out" />

```

```
</LinearLayout>
```

activity_home.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android
    ="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.edwin.fyp.HomeActivity">

    <Button
        android:id="@+id/start_driving"
        android:layout_width="178dp"
        android:layout_height="94dp"
        android:layout_marginEnd="103dp"
        android:layout_marginStart="103dp"
        android:layout_marginTop="31dp"
        android:onClick="onStartDriving"
        android:text="@string/start_driving"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/book_a_ride"
        android:layout_width="183dp"
        android:layout_height="96dp"
```

```
            android:layout_marginEnd="100dp"
            android:layout_marginStart="101dp"
            android:layout_marginTop="28dp"
            android:text="@string/book_a_ride"
            android:onClick="onBookARide"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.0"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toBottomOf="@+id/
                start_driving"
            android:layout_marginRight="100dp"
            android:layout_marginLeft="101dp" />

      <Button
       android:id="@+id/manage_booking"
       android:layout_width="183dp"
       android:layout_height="96dp"
       android:layout_marginEnd="100dp"
       android:layout_marginLeft="101dp"
       android:layout_marginRight="100dp"
       android:layout_marginStart="101dp"
       android:layout_marginTop="28dp"
       android:onClick="onViewBooking"
       android:text="@string/Manage_booking"
       app:layout_constraintEnd_toEndOf="parent"
       app:layout_constraintHorizontal_bias="0.0"
       app:layout_constraintStart_toStartOf="parent"
       app:layout_constraintTop_toBottomOf="@+id/book_a_ride"
            />
```

```

    <Button
        android:id="@+id/user_details"
        android:layout_width="183dp"
        android:layout_height="96dp"
        android:layout_marginEnd="100dp"
        android:layout_marginLeft="101dp"
        android:layout_marginRight="100dp"
        android:layout_marginStart="101dp"
        android:layout_marginTop="16dp"
        android:onClick="onDrivingInfo"
        android:text="User and Driving Info"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/
            manage_booking" />

</android.support.constraint.ConstraintLayout>
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android
    ="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:scrolling_image_view="http://schemas.android.com
        /apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.edwin.fyp.MainActivity">


    <FrameLayout
        android:id="@+id/frameLayout2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true"
        android:layout_marginTop="56dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <com.q42.android.scrollingimageview.
            ScrollingImageView
            android:id="@+id/scrolling_background"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            scrolling_image_view:speed="1dp"
            scrolling_image_view:src="@drawable/
                scrolling_background" />

        <com.q42.android.scrollingimageview.
            ScrollingImageView
            android:id="@+id/scrolling_foreground"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            scrolling_image_view:speed="2.5dp"
            scrolling_image_view:src="@drawable/
                scrolling_foreground" />
    </FrameLayout>

    <ImageView
        android:id="@+id/Van"
        android:layout_width="0dp"
        android:layout_height="49dp"
        android:layout_centerInParent="true"
        android:layout_marginTop="56dp"
        android:scaleType="centerInside"
        android:src="@drawable/mycoms"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <com.google.android.gms.common.SignInButton
        android:id="@+id/sign_in_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginEnd="136dp"
        android:layout_marginStart="116dp"
        android:layout_marginTop="80dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toEndOf="@+id/guideline
            "
        app:layout_constraintTop_toBottomOf="@+id/
            introduction">

    </com.google.android.gms.common.SignInButton>

    <Button
        android:id="@+id/go_to_home"
        android:layout_width="105dp"
        android:layout_height="wrap_content"
        android:layout_marginEnd="140dp"
        android:layout_marginStart="118dp"
        android:onClick="onGoHome"
        android:text="@string/go_to_home"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toEndOf="@+id/guideline
            "
        app:layout_constraintTop_toBottomOf="@+id/
            introduction"
        app:layout_constraintVertical_bias="0.515" />

    <TextView
        android:id="@+id/status"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="9dp"
        android:layout_marginEnd="160dp"
        android:layout_marginStart="142dp"
        android:layout_marginTop="25dp"
        android:text="status"
```

```
            app:layout_constraintBottom_toTopOf="@+id/
                go_to_home"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toEndOf="@+id/guideline
                "
            app:layout_constraintTop_toBottomOf="@+id/
                introduction" />

        <TextView
            android:id="@+id/introduction"
            android:layout_width="176dp"
            android:layout_height="140dp"
            android:layout_marginEnd="104dp"
            android:layout_marginStart="104dp"
            android:layout_marginTop="64dp"
            android:text="@string/lorem_ipsum"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.0"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toBottomOf="@+id/Van" />

        <android.support.constraint.Guideline
            android:id="@+id/guideline"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:orientation="vertical"
            app:layout_constraintGuide_begin="20dp" />

        <Button
```

```
            android:id="@+id/sign_out_and_disconnect"
            android:layout_width="108dp"
            android:layout_height="46dp"
            android:layout_marginBottom="16dp"
            android:layout_marginEnd="149dp"
            android:layout_marginStart="118dp"
            android:layout_marginTop="12dp"
            android:text="Sign Out"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.083"
            app:layout_constraintStart_toEndOf="@+id/guideline
                "
            app:layout_constraintTop_toBottomOf="@+id/
                sign_in_button" />


</android.support.constraint.ConstraintLayout>
```

activity_maps.xml

```
<fragment xmlns:android="http://schemas.android.com/apk/
    res/android"
    xmlns:map="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/map"
    android:name="com.google.android.gms.maps.
        SupportMapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.edwin.fyp.MapsActivity" />
```

activity_tracker.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android
    ="http://schemas.android.com/apk/res/android"
      xmlns:app="http://schemas.android.com/apk/res-auto"
      xmlns:tools="http://schemas.android.com/tools"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      tools:context=".TrackerActivity">

</android.support.constraint.ConstraintLayout>
```

activity_view_booking.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android
    ="http://schemas.android.com/apk/res/android"
      xmlns:app="http://schemas.android.com/apk/res-auto"
      xmlns:tools="http://schemas.android.com/tools"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      tools:context=".ViewBookingActivity">

      <LinearLayout
            android:id="@+id/linearLayout2"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            tools:layout_editor_absoluteX="167dp"
            tools:layout_editor_absoluteY="270dp">


```

```
        <TextView
            android:id="@+id/departure"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="@string/departure_view" />

        <TextView
            android:id="@+id/arrival"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="@string/arrival_view" />

        <TextView
            android:id="@+id/show_booking_time"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="@string/booking_time_view" />


    </LinearLayout>

    <TextView
        android:id="@+id/status_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="360dp"
        android:layout_marginEnd="113dp"
        android:layout_marginStart="113dp"
        android:layout_marginTop="113dp"
        android:layout_weight="1"
        android:text="@string/booking_status"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/
            linearLayout2"
        app:layout_constraintVertical_bias="1.0" />

    <Button
        android:id="@+id/confirm"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginEnd="148dp"
        android:layout_marginTop="56dp"
        android:layout_weight="1"
        android:text="Confirm"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/
            status_view" />

    <Button
        android:id="@+id/back_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
            android:layout_marginEnd="148dp"
            android:layout_marginTop="100dp"
            android:onClick="onBack"
            android:text="Back"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintTop_toBottomOf="@+id/confirm"
                />

        <Button
            android:id="@+id/delete_book"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginBottom="24dp"
            android:layout_marginEnd="148dp"
            android:layout_marginTop="28dp"
            android:text="Delete "
            app:layout_constraintBottom_toTopOf="@+id/
                back_view"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintTop_toBottomOf="@+id/confirm"
            app:layout_constraintVertical_bias="1.0" />

</android.support.constraint.ConstraintLayout>
```