QAU SMART RIDE

Submitted by: Afsheen Altaf

Supervised by: Ms. Ifrah Farukh Khan

Department of Computer Sciences

Quaid-i-Azam University

Islamabad

Session (2014-2018)

ACKNOWLEDGEMENT

All the praises, thanks and acknowledgments are for the creator Allah Almighty, the most Beneficent, the most Merciful, who gave me strength and enabled me to undertake and execute this task. Countless salutations upon the Holy Prophet Hazrat Muhammad (S.A.W), source of knowledge for enlightening with the essence of faith in Allah and guiding the mankind, the true path of life.

First and foremost, I would like to thank everyone who had contributed to the successful completion of this project. I would like to express my gratitude to my Project Supervisor, Ms. Ifrah Farukh Khan for her invaluable advice, guidance and her enormous patience throughout the development of the project. I am sincerely grateful to her for sharing her truthful and illuminating views on a number of issues related to the project.

In addition, I would also like to express my gratitude to my family and friends especially Zobia Jamil who had helped and given me encouragement in my study.

ABSTRACT

QAU Smart Ride works to facilitate drivers, students and employees of the university. Therefore, system has been designed and developed to track and monitor Quaid-e-Azam University buses. It will allow admin to monitor buses and their speed. Admin will be notified if the buses move out of the route or over speed. Admin will be responsible for maintaining data of drivers and allocating routes and stops. Also, admin can send messages to drivers on real time basis. The application also allows the drivers to check the current location of the bus along with stops of the assigned routes. Furthermore, application let students, faculty members and staff of the university to track the location of the buses in real time along with the estimated time of arrival at their stops. Passengers can also set radius for the geo-fencing alerts and can rate drivers. Drivers performance will be calculated on the basis of rating and over-speeding alerts. This system is only for QAU, limited to one organization and in future work, it can be extended to other organization as well.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

UC	Use Case
UCD	Use Case description
SPMP	System Project Management Plan
SRS	Software Requirements Specifications
SDD	Software Design Description
STD	Software Test Documentation
SID	Software Implementation Documentation
SMS	Short Message Service
GPS	Global Positioning System
IDE	Integrated Development Environment
GUI	Graphical User Interface

CHAPTER 1
SOFTWARE PROJECT MANAGEMENT PLAN

1.1 INTRODUCTION

This chapter describes roles, responsibilities, processes and schedules for managing the software development process, in addition to outlining the tools, methods, and procedures to be used.

1.1.1 Project Overview

QAU Smart Ride will be designed to track and monitor Quaid-e-Azam University buses. It will allow admin to monitor buses and their speed. Admin will be notified if the buses move out of the route or over speed. Admin will be responsible for maintaining data of drivers, routes and stops. Also, admin can send messages to drivers on real time basis. The application also allows the drivers to check the current location of the bus along with stops of the assigned routes. Furthermore, application let students, faculty members and staff of the university to track the location of the buses in real time along with the estimated time of arrival at their stops. Passengers can also set radius for the geo-fencing alerts.

1.1.2 Project Deliverables

Deliverable is a tangible output of human effort provided by a developer to a customer. These deliverables are delivered to the project supervisor. The deliverables for the project "QAU Smart Ride" are as follows:

Table 1. 1 Deliverables

1st Phase	SPMP, SRS
2ndPhase	SDD, STD
3rdPhase	SID, Application

1.2 PROJECT ORGANIZATION

This section details the architecture of the project, including: the process model, organizational structure, interfaces, and project responsibilities.

1.2.1 Software Process Model

In this project, waterfall process model will be used because

- Requirements are clear and well-defined.
- The phases proceed in a systematic and sequential approach that begins at the system level and proceed through analysis, design, coding, testing and support.
- Product definition is stable.
- Simple and easy to implement.
- Each phase has review process and well understood milestones.

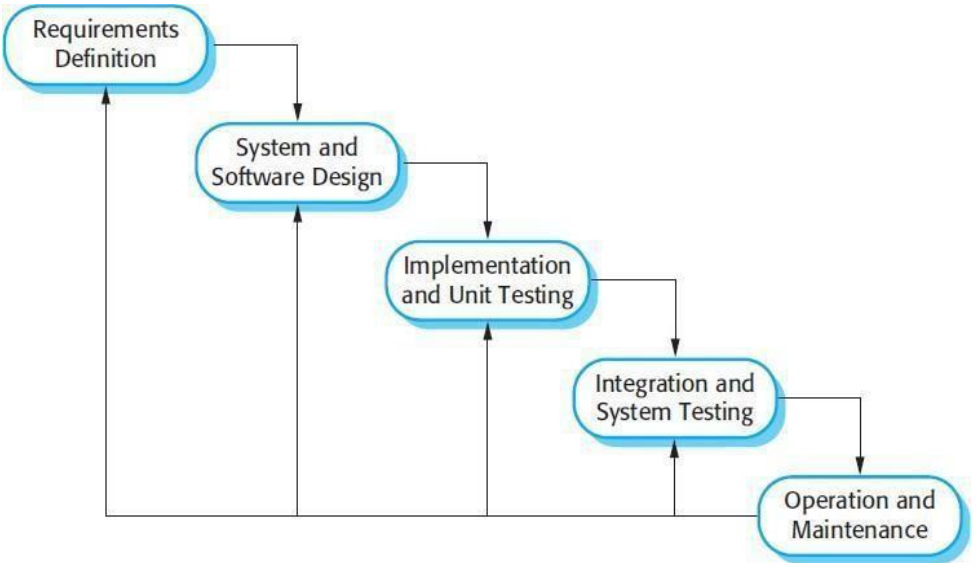


Figure 1.1 Software Process Model

These phases are completed once and there is no chance of repeating and going back to the previous phase. In this project, it is asked to complete each step within the defined time. Therefore, it is chosen.

1.2.2 Roles and Responsibilities

In this Project, I am responsible for developing the Project Plan and responsible for communication, status reporting, developing the software product, and, in general, making sure the project is delivered on schedule and within scope.

The Project Supervisor Ms. Ifrah Farukh Khan assists with major issues, problems, and policy conflicts and removes obstacles.

1.2.3 Tools and Techniques

Following are the tools and techniques used for this project.

Table 1. 2 Tools and Techniques

MS Word 2013	Used for documentation purposes
Draw.io	Used for making diagrams
Java SDK	Required for android studio
Android Studio	Used for coding of client-side application
Notepad++	Used for coding of admin-side application
Mongo DB	Database used for storage purposes.
ProjectLibre 1.7	Used for making project plan

1.3 PROJECT MANAGEMENT PLAN

This section describes how the project will be manage, what are its tasks, deliverables, milestones etc.

1.3.1 Tasks

i. Identify Requirements
ii. Define Use Cases
iii. Develop Analysis Model
iv. Develop SRS
v. Develop Design
vi. Design Input
vii. Develop Algorithm
viii. Software Test Documentation
ix. Software Implementation

1.3.2 Assignments

My project is on individual basis so all the assignments are done by my end. Following are the assignments.

i. Software Requirement Specification
ii. Software Project Management Plan
iii. Software Design Description
iv. Software Test Documentation

1.3.3 Timetable

This section describes the timetable of Project.

	Name	Duration	Start	Finish
1	**Requirement Analysis of Case-Stud**	**50 days**	**9/25/17 8:00 AM**	**12/1/17 5:00 PM**
2	**Identify Requirements**	**22 days**	**9/25/17 8:00 AM**	**10/24/17 5:00 PM**
3	Review case-study	7 days	9/25/17 8:00 AM	10/3/17 5:00 PM
4	Define requirements	15 days	10/3/17 8:00 AM	10/23/17 5:00 PM
5	Meet stakeholders	1 day	10/23/17 8:00 AM	10/23/17 5:00 PM
6	Review requirements	1 day	10/24/17 8:00 AM	10/24/17 5:00 PM
7	**Define Use Cases**	**8 days**	**10/24/17 8:00 AM**	**11/2/17 5:00 PM**
8	Write Use cases	7 days	10/24/17 8:00 AM	11/1/17 5:00 PM
9	Draw Use case diagram	1 day	11/1/17 8:00 AM	11/1/17 5:00 PM
10	Review Requirements and use cases	1 day	11/2/17 8:00 AM	11/2/17 5:00 PM
11	**Develop Analysis Model**	**8 days**	**11/2/17 8:00 AM**	**11/13/17 5:00 PM**
12	Develop Domain model	3 days	11/2/17 8:00 AM	11/6/17 5:00 PM
13	Develop System Sequence Diagrams	2 days	11/8/17 8:00 AM	11/9/17 5:00 PM
14	Review Diagrams	1 day	11/11/17 8:00 AM	11/13/17 5:00 PM
15	**Develop SRS**	**15 days**	**11/13/17 8:00 AM**	**12/1/17 5:00 PM**
16	Identify Functional and Non-function.	2 days	11/13/17 8:00 AM	11/14/17 5:00 PM
17	Identify Entities and their Relationship	2 days	11/15/17 8:00 AM	11/16/17 5:00 PM
18	Develop ERD	1 day	11/17/17 8:00 AM	11/17/17 5:00 PM
19	Review Requirements	3 days	11/17/17 8:00 AM	11/21/17 5:00 PM
20	Finalize SRS	10 days	11/17/17 8:00 AM	11/30/17 5:00 PM
21	Requirement Analysis Done	1 day	12/1/17 8:00 AM	12/1/17 5:00 PM
22	**Design Phase**	**30 days**	**12/2/17 8:00 AM**	**1/12/18 5:00 PM**
23	**Develop Achitectural Design**	**7 days**	**12/2/17 8:00 AM**	**12/12/17 5:00 PM**
24	Define Modules	2 days	12/2/17 8:00 AM	12/5/17 5:00 PM
25	Define Interfaces	2 days	12/5/17 8:00 AM	12/6/17 5:00 PM
26	Define Methods and their parameters	3 days	12/7/17 8:00 AM	12/11/17 5:00 PM
27	Review architectural design	1 day	12/12/17 8:00 AM	12/12/17 5:00 PM
28	**Develop Detailed Design**	**22 days**	**12/12/17 8:00 AM**	**1/10/18 5:00 PM**

	Name	Duration	Start	Finish
29	Make sequence diagrams	7 days	12/12/17 8:00 AM	12/20/17 5:00 PM
30	Make Class diagram	2 days	12/20/17 8:00 AM	12/21/17 5:00 PM
31	Write Algorithms	15 days	12/21/17 8:00 AM	1/10/18 5:00 PM
32	Review diagrams and algorithms	1 day	1/10/18 8:00 AM	1/10/18 5:00 PM
33	**Evaluate Design**	**3 days**	**1/10/18 8:00 AM**	**1/12/18 5:00 PM**
34	Validate Design	2 days	1/10/18 8:00 AM	1/11/18 5:00 PM
35	Verify Design	1 day	1/11/18 8:00 AM	1/11/18 5:00 PM
36	Refine Design	1 day	1/11/18 8:00 AM	1/11/18 5:00 PM
37	Design Phase Completed	1 day	1/12/18 8:00 AM	1/12/18 5:00 PM
38	**Develop Software Test Documenta.**	**9 days**	**1/12/18 8:00 AM**	**1/24/18 5:00 PM**
39	Develop Test plan	3 days	1/12/18 8:00 AM	1/16/18 5:00 PM
40	Develop Test Cases	7 days	1/16/18 8:00 AM	1/24/18 5:00 PM
41	Review Test Documentation	1 day	1/24/18 8:00 AM	1/24/18 5:00 PM
42	**Implementation**	**103 days?**	**2/5/18 8:00 AM**	**6/27/18 5:00 PM**
43	Setup Development Environment	8 days?	2/5/18 8:00 AM	2/14/18 5:00 PM
44	Start front end coding	25 days	2/15/18 8:00 AM	3/21/18 5:00 PM
45	Start Backend coding	66 days	3/21/18 8:00 AM	6/20/18 5:00 PM
46	Review code	6 days	6/20/18 8:00 AM	6/27/18 5:00 PM
47	Implementation Document	1 day	6/27/18 8:00 AM	6/27/18 5:00 PM
48	Integration	2 days	6/27/18 8:00 AM	6/28/18 5:00 PM
49	Testing	3 days	6/28/18 8:00 AM	7/2/18 5:00 PM

Figure 1. 2 Project Timetable

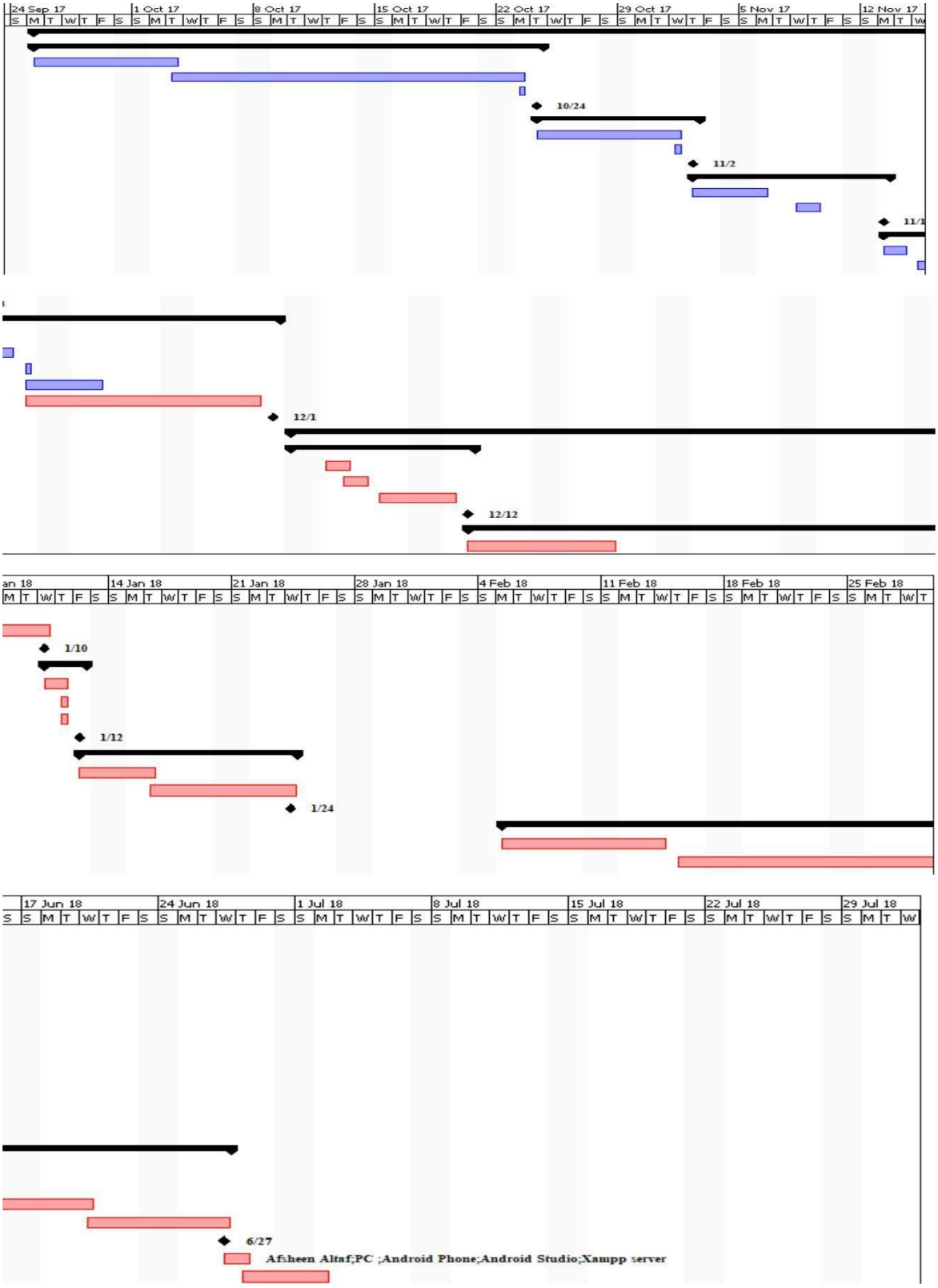


Figure 1. 3 Project Gantt Chart

CHAPTER 2
SOFTWARE REQUIREMENTS SPECIFICATION

2.1 INTRODUCTION

This chapter aims at defining the overall Software Requirements for “QAU Smart Ride”. Efforts have been made to define the requirements accurately. The final product will be having features mentioned in this document.

2.1.1 Purpose

The purpose of this SRS document is to provide a detailed overview of software product, its parameters and goals. It will illustrate the purpose and complete declaration for the development of system. This document describes the project’s intended audience, all the functional as well as non-functional requirements. It also defines how the client, developer and audience see the product and its functionality.

2.1.2 Scope

There are three types of users

- Driver
- Passenger (Student/Employees)
- Admin

QAU Smart Ride is a combination of web as well as android application where the admin will be using the web application whereas driver and passenger will be using android application. The system will allow admin, drivers and passengers to track the location of buses. Admin will be notified when driver exceeds the speed limit or moves out of the route. System will also admin to manage drivers, routes and stops and can send SMS to drivers on real-time basis. Drivers will be able to view their assigned routes and can send panic alert to admin. System will allow passengers to track the location of bus of particular route along with estimated time of arrival. Passengers will be able to receive the geo-fencing alerts if they enable alerts.

2.1.3 Overview

The remaining sections of this chapter provide a general description, including characteristics of the users of this project, the product’s hardware, and the functional and data requirements of the product. General description of the project is discussed in section 2 of this document. Section 3 gives the functional requirements, data requirements and constraints and assumptions made while designing the software. Section 3 also gives the specific requirements of the product. Section 3 also discusses the external interface requirements and gives detailed description of functional requirements.

2.2 OVERALL DESCRIPTION

This section will give an overview of the whole system. The system will be explained in its context to show how the system interacts with other systems and introduce the basic functionality of it. It will also describe what type of stakeholders that will use the system and what functionality is available for each type. At last, the constraints and assumptions for the system will be presented.

2.2.1 Product Perspective

QAU Smart Ride is a standalone system and is intended for use on the Android platform and web. The scope of the project encompasses both server- and client-side functionalities, so both aspects are covered in detail within this document. Below is a diagram of the QAU Smart Ride application which illustrates the interactions between the server and client applications.

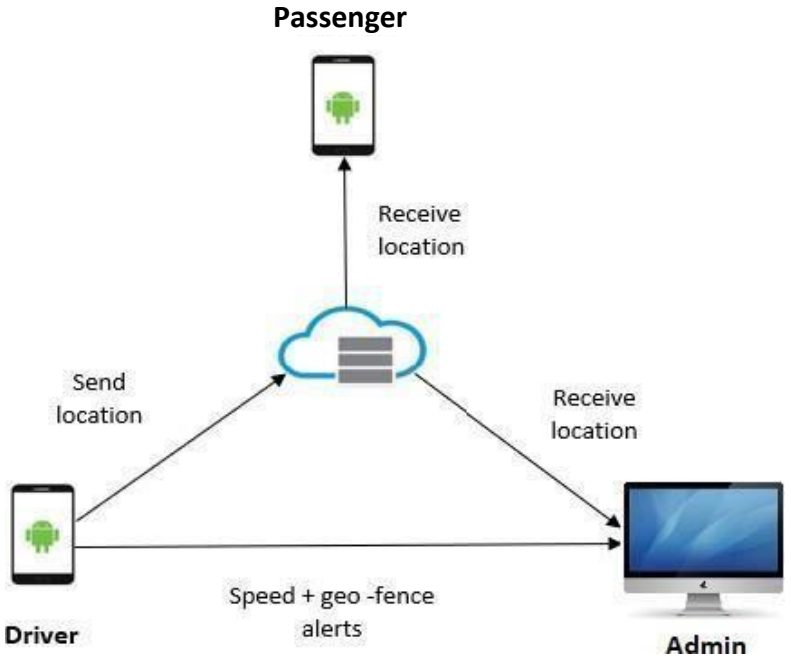


Figure 2. 1 System Block diagram

2.2.2 User Characteristics

There are three types of users involved: Admin, Driver and Passenger.

- Users must have technical expertise to use smartphone.
- Users may have to be trained for using the application.

2.2.3 General Constraints

The constraints are mentioned below:

- Application must be installed in the phone.
- Only registered drivers and passengers can use the application.
- The system must be user-friendly.

2.2.4 Assumptions and Dependencies

- System is dependent on access to the GPS and Internet.
- The users should know the English language, as the interface will be provided in English.

2.3 EXTERNAL INTERFACE REQUIREMENTS

2.3.1 User interfaces

QAU Smart Ride should be designed for ease of use, providing help instructions, and appropriate error messages for user inputs. The QAU Smart Ride makes sure at every point, that the user spends most of the time using the application rather than figuring out how to use it. The background will be light in color with dark colored font to enhance the contrast and visibility. Controls which allow the user to interact with the application will be clear and imply their functionality within the application.

The home screen offers a menu with a list of functions that the application performs. The user can select one of the options on the menu, and is taken to the respective screen. Every screen displays the menu on the top. The user can click on any one of the options and is taken to the screen of their choice.

2.3.2 Hardware interfaces

This application will work on android phones and tablets. Since the application must run connected to internet and need a location data, Android devices must have a GPS and internet connection to run this application.

2.3.3 Software interfaces

PLATFORM	:	Android SDK framework
ANDROID EMULATOR	:	SDK version 2.2 or higher
TECHNOLOGIES USED	:	Java, Xml, MEAN Stack, HTML
DATABASE	:	Mongo DB

2.3.4 Communications interfaces

All communication between driver, admin and passengers will be performed over the internet, so QAU Smart Ride will use the HTTP protocol.

2.4 SOFTWARE PRODUCT FEATURES

Summary of major functions that the system should perform are listed below.

Register account: This function will allow admin to register account of drivers.

Login account: This function will allow driver/admin/passenger to login account.

Track location: This function will allow admin/driver/passenger to check the current location of the bus.

Allocate Routes: This function will allow admin to allocate routes to drivers.

Send over speeding alert: This function will send notification to the admin via SMS when bus exceeds the certain speed.

Send Geo-fencing alert: This function will send notification to the admin when buses will enter or exit the university premises.

Send Panic alert: This function will send push notification to the Admin Panel.

Send message: Allow admin to send instant messages to drivers.

2.5 USE CASE MODEL

QAU Smart Ride has many features. Each user panel has its own features. Some features are provided for every kind of user but some features are provided for only particular type of user and only these users have access to these features.
QAU Smart Ride has the following use cases:

ADMIN

1. Login
2. Logout
3. Register account
4. Update account
5. Delete account
6. Send Message
7. Track buses location
8. Allocate resources
9. Add stop
10. View stop
11. View route

DRIVER

1 Login
2 Logout
3 Track bus location
4 View message
5 Track current location
6 Send panic alert

PASSENGER

1. Login
2. Logout
3. Track bus location
4. Set geo-fencing alert

SPECIAL REQUIREMENTS

1. Android based smartphone.
2. GPS and Internet connection is available.
3. Text must be visible to the users.

STAKEHOLDERS AND INTERESTS

Admin:

Admin wants to keep track of all buses and receives alert if the buses exceeds the speed limit and buses enter or leave the university premises.

Driver:

Driver wants to know the current location of the bus. Also, the routes assigned to him along with stops and can send panic alert to admin.

Passenger:

Passenger wants to track the buses and be able to get estimated time of arrival of buses arriving at their stops.

2.5.1 Use Case Diagram

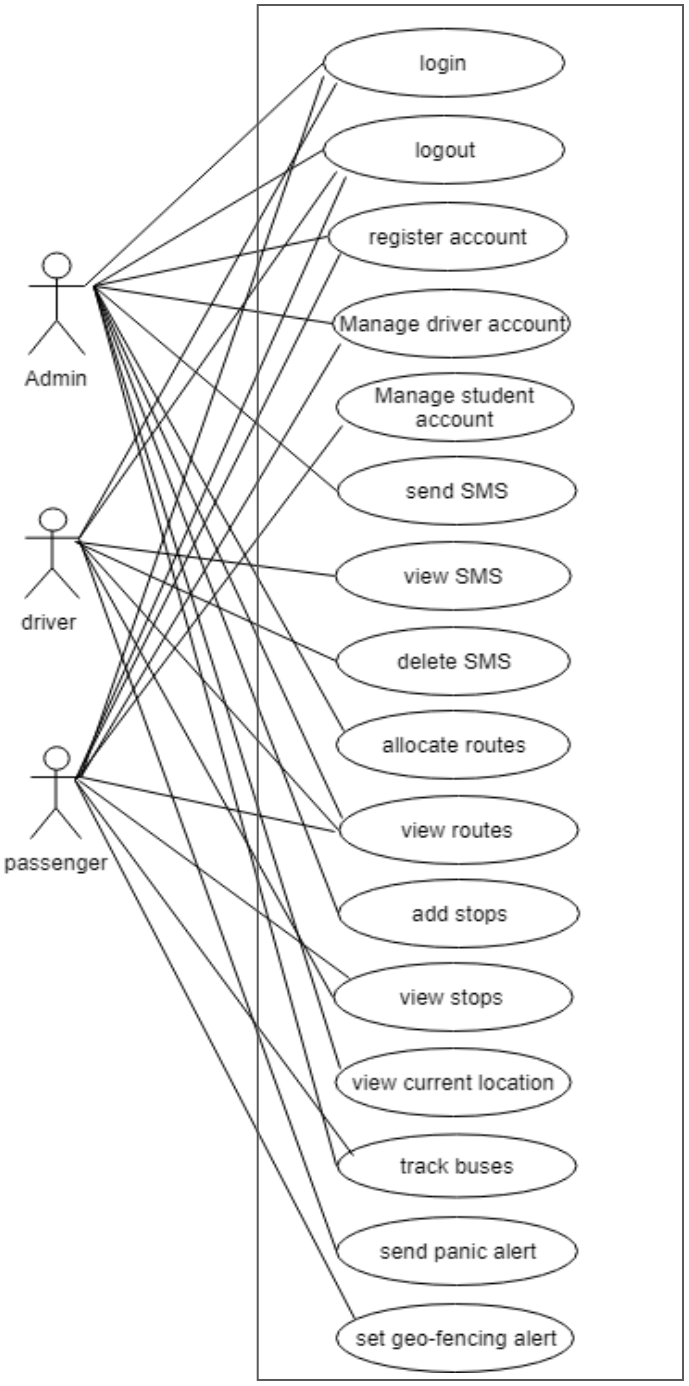


Figure 2. 2 Use case Diagram

2.5.2 Use Cases Description

The full description of use cases is given below:

Table 2. 1 UCD for Login

UC-1: Login	
Primary actor	User (Admin, Driver, Passenger)
Pre-condition	User must have an account.
Post-condition	User has logged in successfully.
Main success scenario	1. The System prompts the user for username and password or register new account. 2. The user enters username and password. 3. User presses login button. 4. The system validates the entered information, making sure that the entered username and password is valid for one user account in the system, and that the required password is entered for the entered username. 5. The User is returned to the home page as logged in user.
Alternative scenario	a. Internet is not available. b. GPS is disabled c. Android OS is crashed. 1a. User enters wrong credentials. System prompts the user to enter correct information. User re-renter's information. 2a. user submits information without filling all required fields. System prompts the user to fill all required fields
Special requirements	Android phone or tablet
Technology	None
Frequency	Many times, a day.

Table 2. 2 UCD for Logout

UC-2: Logout	
Primary actor	User (Admin, Driver, Passenger)
Pre-condition	User must be logged in.
Post-condition	User has been logged out successfully.
Main success scenario	1. User presses logout button. 2. System displays login screen.
Alternative scenario	a. Internet is not available. b. GPS is disabled c. Android OS is crashed.
Special requirements	Android phone or tablet
Technology	None
Frequency	Many times, a day.

Table 2. 3 UCD for Register account

UC-3: Register account	
Primary actor	User (Admin, passenger)
Pre-condition	User must be logged in.
Post-condition	New driver's account has been created successfully
Main success scenario	1. User presses the create account button. 2. System shows "Register User" model. 3. User inputs name, contact number, address, username and password. 4. User presses Submit button. 5. System displays successful registration message.
Alternative scenario	a) Internet is not available. b) GPS is disabled c) Android OS is crashed. 4a) User submits information without filling all required fields. System prompts the user to fill all required fields
Special requirements	Android phone or tablet
Technology	None
Frequency	Many times, a day.

Table 2. 4 UCD for Update account

UC-4: Update Account	
Primary actor	User (Admin, passenger)
Pre-condition	User must be logged in.
Post-condition	User updated the account successfully
Main success scenario	1. User selects the user category. 2. System shows list of users of selected category. 3. User selects the user to edit profile. 4. The System displays the user profile. 5. User updates the profile and click submit button. 6. System displays account's updated message
Alternative scenario	a. Internet is not available. b. GPS is disabled c. Android OS is crashed. 5a. user submits information without filling all required fields. System prompts the user to fill all required fields
Special requirements	Android phone or tablet
Technology	None
Frequency	Many times, a day.

Table 2. 5 UCD for Delete account

UC-5: Delete Account	
Primary actor	User (Admin, passenger)
Pre-condition	User must be logged in.
Post-condition	User deleted the account successfully
Main success scenario	1. User selects the user category. 2. System shows list of users of selected category.
	3. User selects the user to delete profile. 4. The System displays the user profile. 5. User clicks submit button. 6. System prompts user for confirmation. 7. User confirms the operation. 8. System displays account's updated message.
Alternative scenario	a. Internet is not available. b. GPS is disabled c. Android OS is crashed. 7a. User not confirms the operation
Special requirements	Android phone or tablet
Technology	None
Frequency	Many times, a day.

Table 2. 6 UCD for Send Message

UC-6: Send Message	
Primary actor	Admin
Pre-condition	Admin must be logged in.
Post-condition	Admin has sent message successfully.
Main success scenario	1. Admin presses the message button. 2. System shows compose message model. 3. Admin selects the recipient 4. Admin inputs the message. 5. Admin presses “Send Message” button. 6. System displays message of send message.
Alternative scenario	2a. Admin sends message without choosing recipient. System prompts user to fill all required fields.
Special requirements	Android phone or tablet
Technology	None
Frequency	Many times, a day.

Table 2. 7 UCD for view message

UC-7: View Message	
Primary actor	Driver
Pre-condition	Driver must be logged in.
Post-condition	Driver has viewed message successfully.
Main success scenario	1. User inputs message information in search bar. 2. System shows Search/Filter message display on screen.
Alternative scenario	a. Internet is not available. b. GPS is disabled c. Android OS is crashed. 2a. User inputs incorrect information. System does not display any result.
Special requirements	Android phone or tablet
Technology	None
Frequency	Many times, a day.

Table 2. 8 UCD for tracking bus location

UC-8: Track bus location	
Primary actor	User (Admin, Driver, Passenger)
Pre-condition	User must be logged in.
Post-condition	User has tracked buses successfully.
Main success scenario	User select the tracking option. System displays the location on the map. a) If the user is driver, system displays the current location of bus on the map. b) If the user is passenger, system displays the time to reach bus stop, location and route. c) If the user is admin, system displays the location of all buses on the map.
Alternative scenario	a. Internet is not available. b. GPS is disabled c. Android OS is crashed.
Special requirements	Android phone or tablet
Technology	None
Frequency	Many times, a day.

Table 2. 9 UCD for Delete message

UC-9: Delete Message	
Primary actor	User (Admin, Driver)
Pre-condition	User must be logged in.
Post-condition	User has deleted message successfully.
Main success scenario	1. User presses “delete” button. 2. System prompts user for confirmation. 3. User confirms the deletion operation. 4. System displays the deleted message.
Alternative scenario	a. Internet is not available. b. GPS is disabled c. Android OS is crashed. 3a. User not confirms the operation.
Special requirements	Android phone or tablet
Technology	None
Frequency	Many times, a day.

Table 2. 10 UCD for Allocating routes

UC-10: Allocate routes	
Primary actor	Admin
Pre-condition	Admin must be logged in.
Post-condition	Routes have been successfully allocated to the drivers.
Main success scenario	1. Admin selects add route option. 2. System prompts for route name, driver name, bus no, start and end location. 3. Admin enters the required information. 4. Admin presses “ok” button.
Alternative scenario	2a. Admin submit without filling al fields. System prompts user to fill all required fields.
Special requirements	Android phone or tablet
Technology	None
Frequency	Many times, a day.

Table 2. 11 UCD for viewing routes

UC-11: View routes	
Primary actor	User (Admin, driver, passenger)
Pre-condition	User must be logged in.
Post-condition	User has viewed the routes
Main success scenario	1. User selects view route option. 2. System displays the route on the map.
Alternative scenario	a. Internet is not available b. GPS is disabled.
Special requirements	Android phone or tablet
Technology	None
Frequency	Many times, a day.

Table 2. 12 UCD for Add stops

UC-12: Add stops	
Primary actor	Admin
Pre-condition	Admin must be logged in.
Post-condition	Stops have been successfully added to the route.
Main success scenario	1. Admin selects route option. 2. System displays all routes. 3. Admin select a particular route to add stops. 4. System displays the map. 5. Admin add markers on the map.
Alternative scenario	a. Internet is not available. b. GPS is disabled. 3a. Admin selects wrong route.
Special requirements	Android phone or tablet
Technology	None
Frequency	Many times, a day.

Table 2. 13 UCD for view stops

UC-13: View stops	
Primary actor	User (Admin, driver, passenger)
Pre-condition	User must be logged in.
Post-condition	User has viewed the routes
Main success scenario	1. User selects view route option. 2. System displays the route on the map along with all stops.
Alternative scenario	a. Internet is not available b. GPS is disabled.
Special requirements	Android phone or tablet
Technology	None
Frequency	Many times, a day.

Table 2. 14 UCD for sending panic alert

UC-14: send panic alert	
Primary actor	Driver
Pre-condition	Driver must be logged in.
Post-condition	Alert has sent successfully.
Main success scenario	1. Driver clicks the panic alert button. 2. App displays the alert sent message.
Alternative scenario	a. Internet is not available b. GPS is disabled.
Special requirements	Android phone or tablet
Technology	None
Frequency	Many times, a day.

Table 2. 15 UCD for setting Geo-fencing alert

UC-15: Set Geo-fencing alert	
Primary actor	Passenger
Pre-condition	Passenger must be logged in.
Post-condition	Geo-fencing alert has set successfully.
Main success scenario	1. Passenger searches and selects the stop. 2. Passenger selects the radius and submits ok. 3. App displays the alert set message.
Alternative scenario	a. Internet is not available b. GPS is disabled. 1a. Passenger selects the wrong stop.
Special requirements	Android phone or tablet
Technology	None
Frequency	Many times, a day.

2.6 SYSTEM SEQUENCE DIAGRAMS

A system sequence diagram is an interaction diagram that shows how objects operate with one another and in what order.

SSD for login

The scenario in figure 2.3 is about the login in which user enter the login credentials and then press login button and in response home screen will appear.

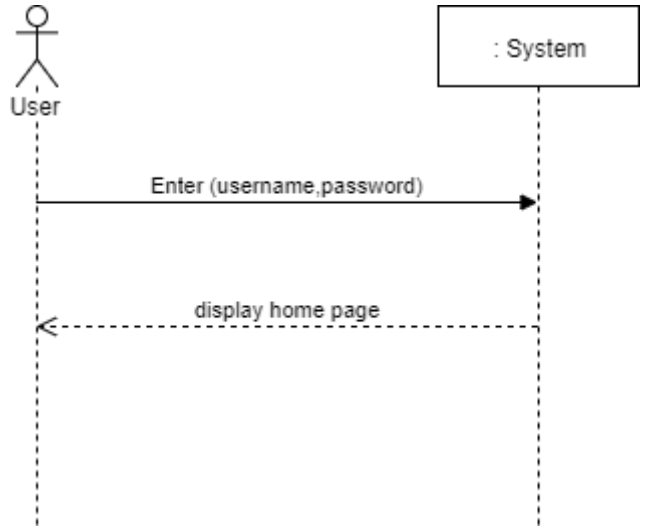


Figure 2. 3 SSD for login

SSD for logout

The scenario in figure 2.4 is about the logout in which user press logout button and in response login screen will appear.

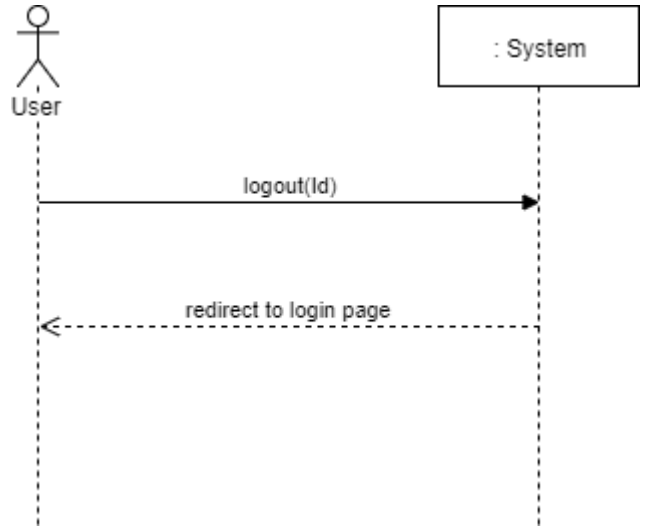


Figure 2. 4 SSD for logout

SSD for Register account

The scenario in Figure 2.5 is about the register account in which admin create accounts of drivers.

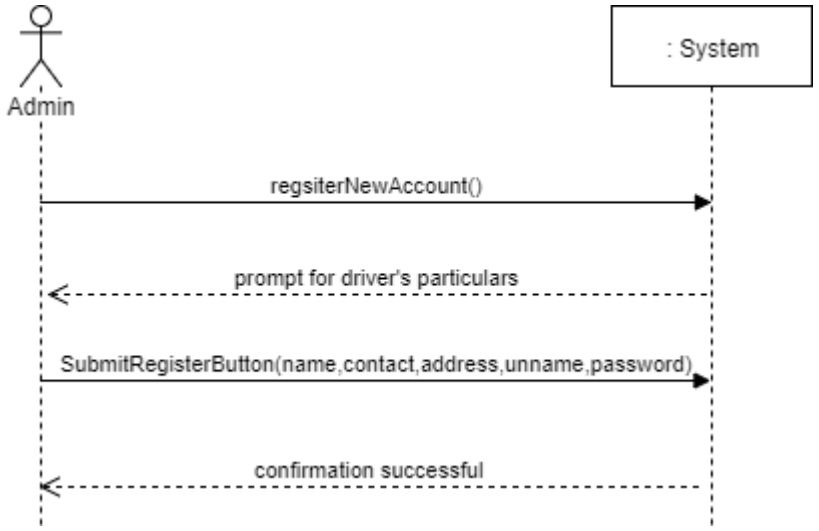


Figure 2. 5 SSD for register account

SSD for Update account

The scenario in Figure 2.6 is about the update account in which user update accounts.

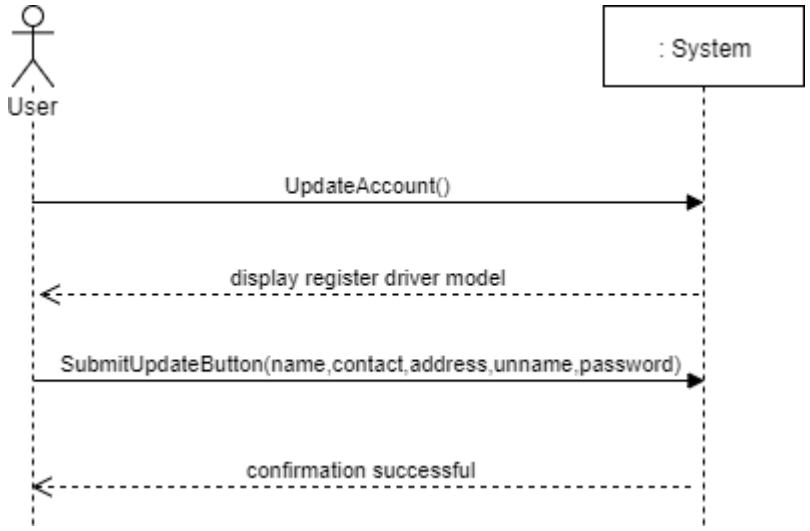


Figure 2. 6 SSD for update account

SSD for Delete account

The Scenario in Figure 2.7 is about the update account in which user delete accounts.

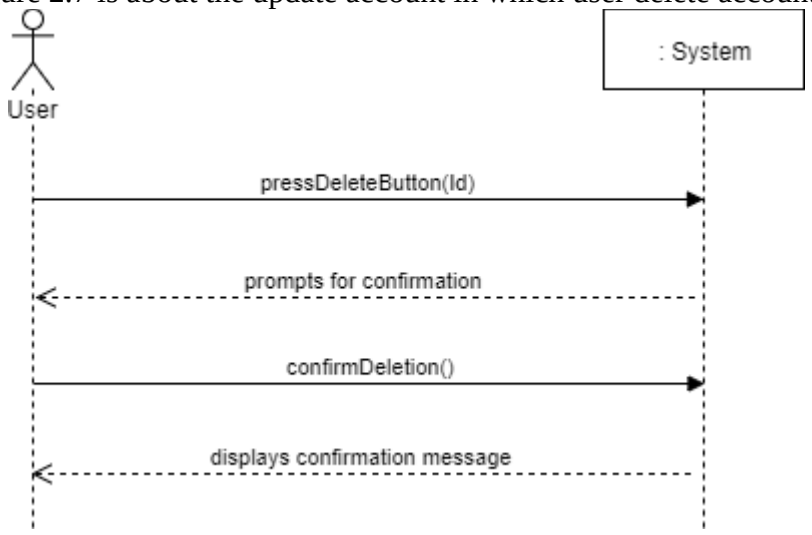


Figure 2. 7 SSD for delete account

SSD for Register Bus

The scenario in Figure 2.8 is about the register/add bus in which admin register new buses.

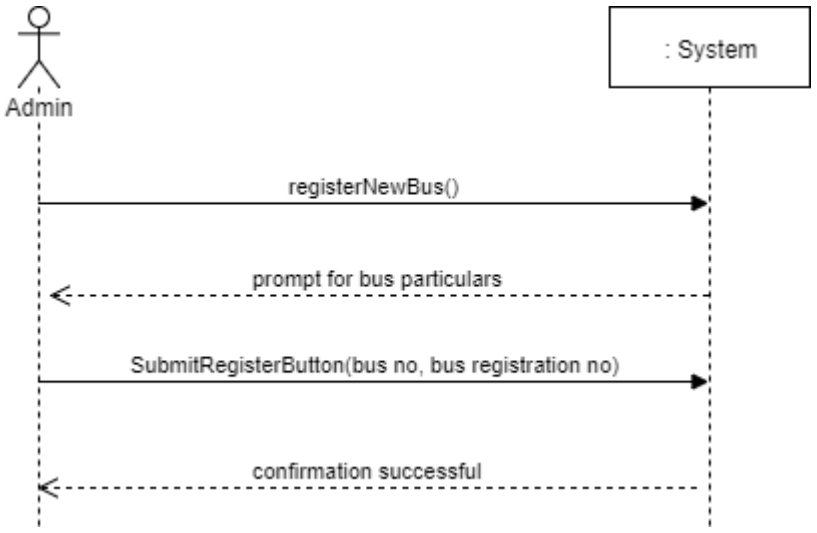


Figure 2. 8 SSD for register bus

SSD for Search Driver

The Scenario in Figure 2.9 is about the search/filter driver in which user searched/filtered driver.

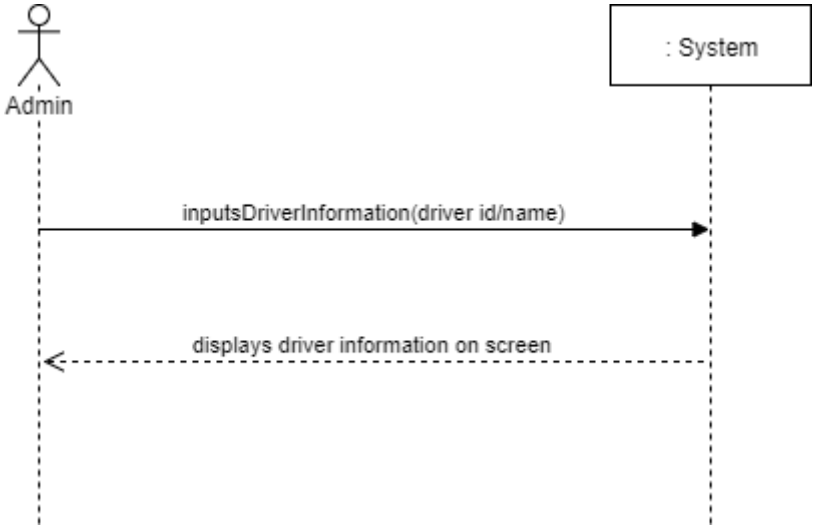


Figure 2. 9 SSD for search driver

SSD for Search Bus

The scenario in Figure 2.10 is about the search/filter bus in which user searched/filtered bus.

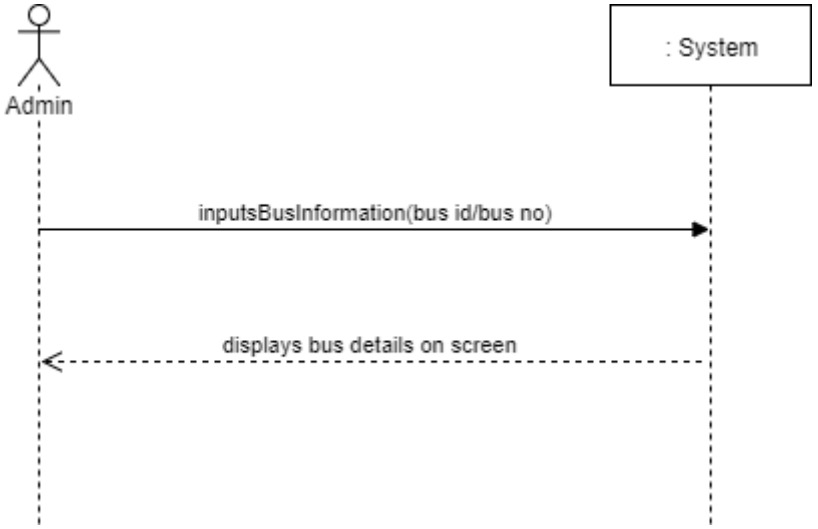


Figure 2. 10 SSD for search bus

SSD for Send message

The scenario in Figure 2.11 is about the send message in which admin send messages to drivers.

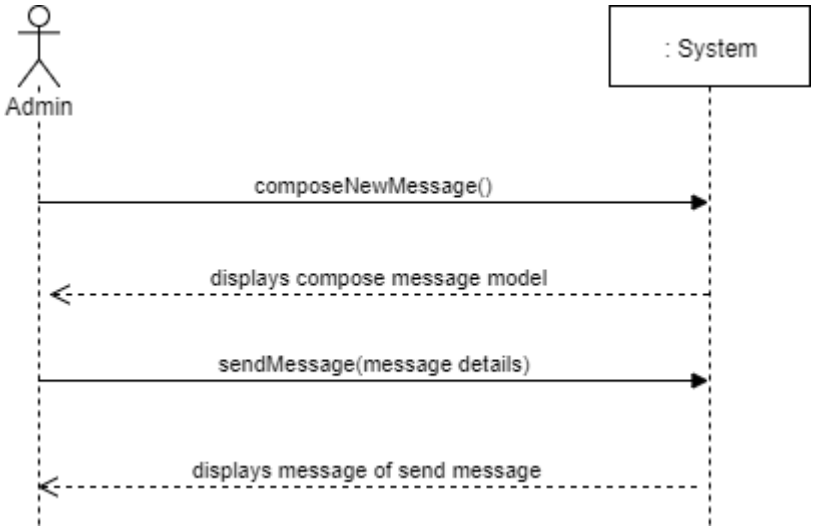


Figure 2. 11 SSD for send message

SSD for View message

The scenario in Figure 2.12 is about the view message in which user views details of message.

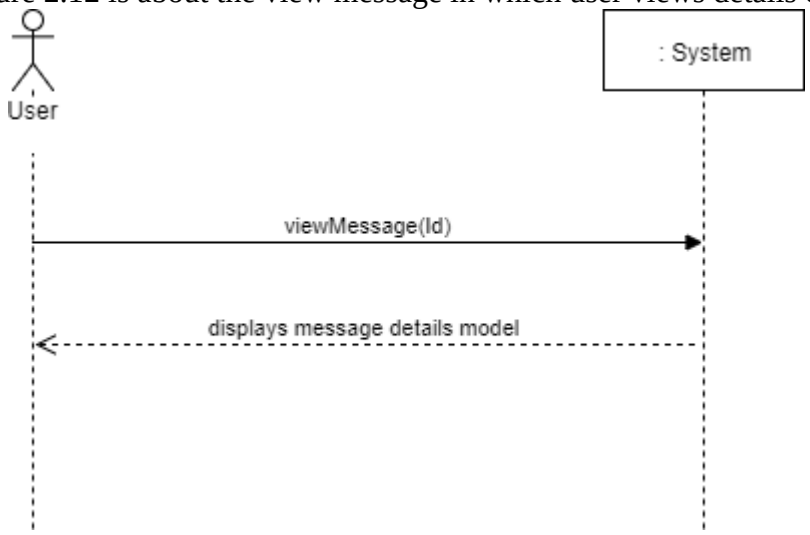


Figure 2. 12 SSD for view message

SSD for Delete Message:

The scenario in Figure 2.13 is about the delete message in which user delete particular message.

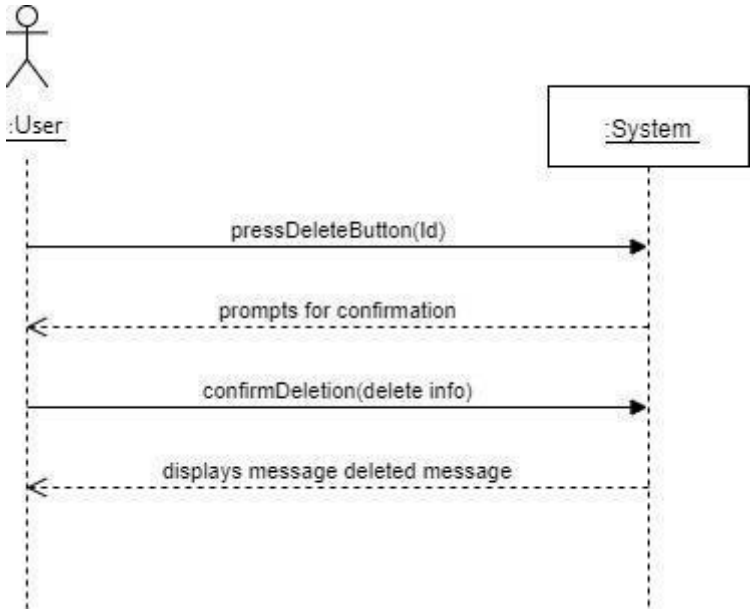


Figure 2. 13 SSD for delete message

SSD for Track Bus:

The scenario in Figure 2.14 is about the track bus location in which user tracks location of the bus.

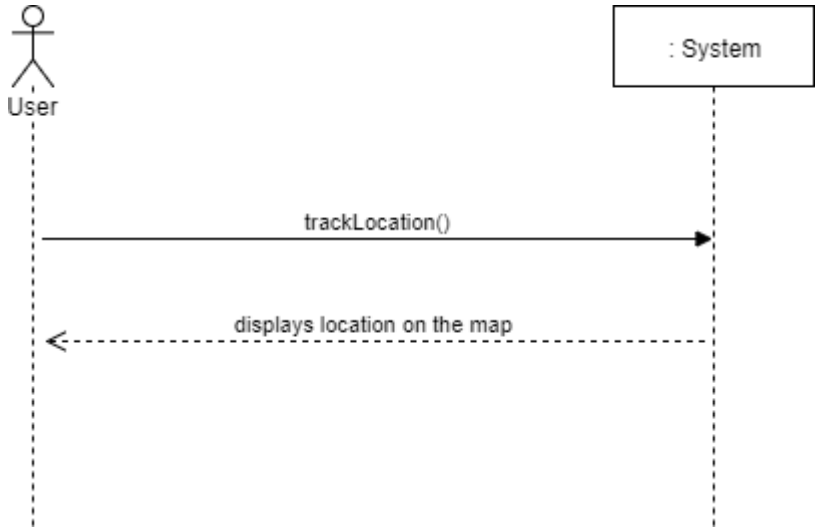


Figure 2. 14 SSD for track location

2.7 ACTIVITY DIAGRAMS

Activity diagram is UML behavior diagram which shows flow of control or object flow with emphasis on the sequence and conditions of the flow.

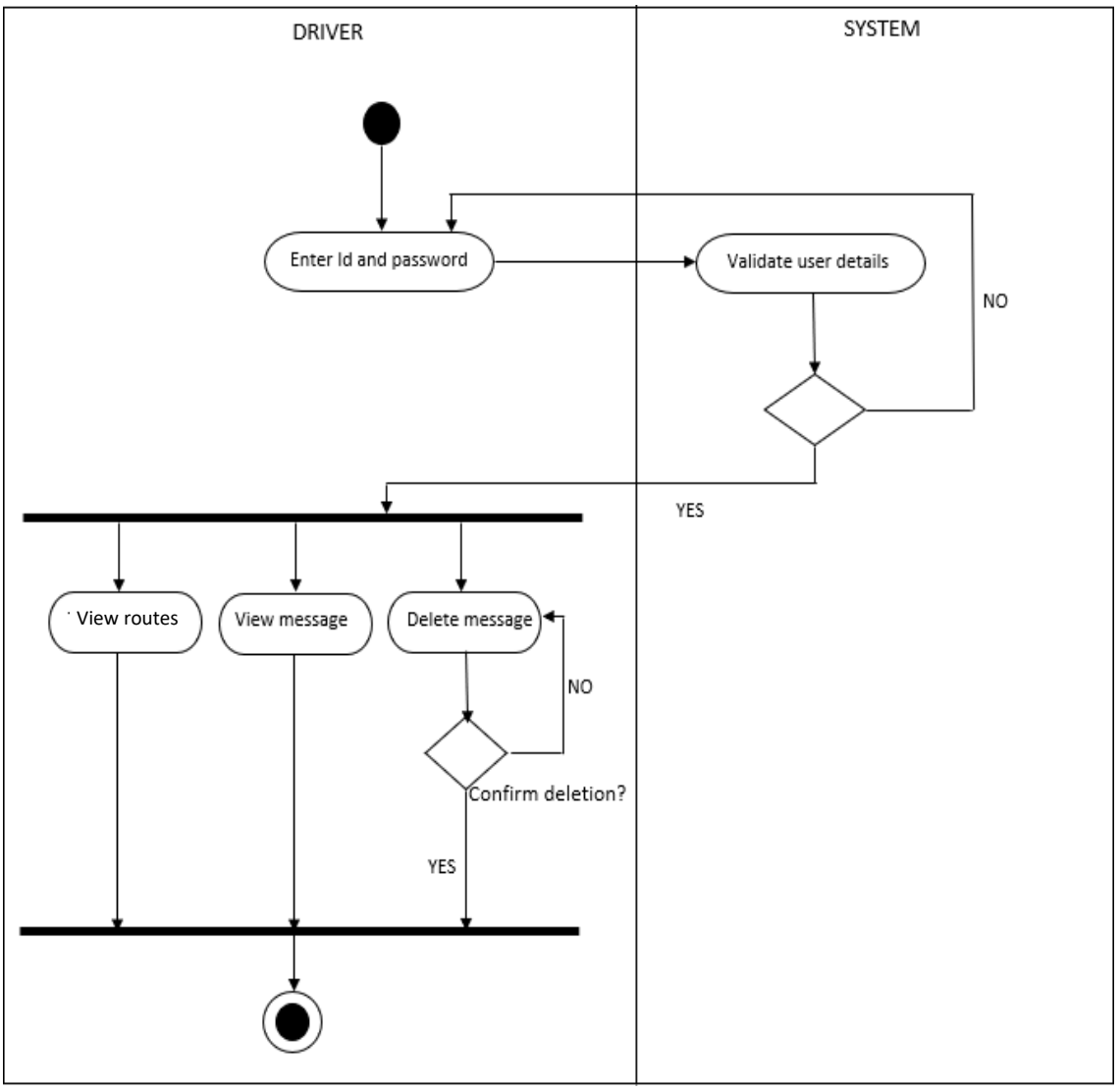


Figure 2. 15 Activity diagram for driver

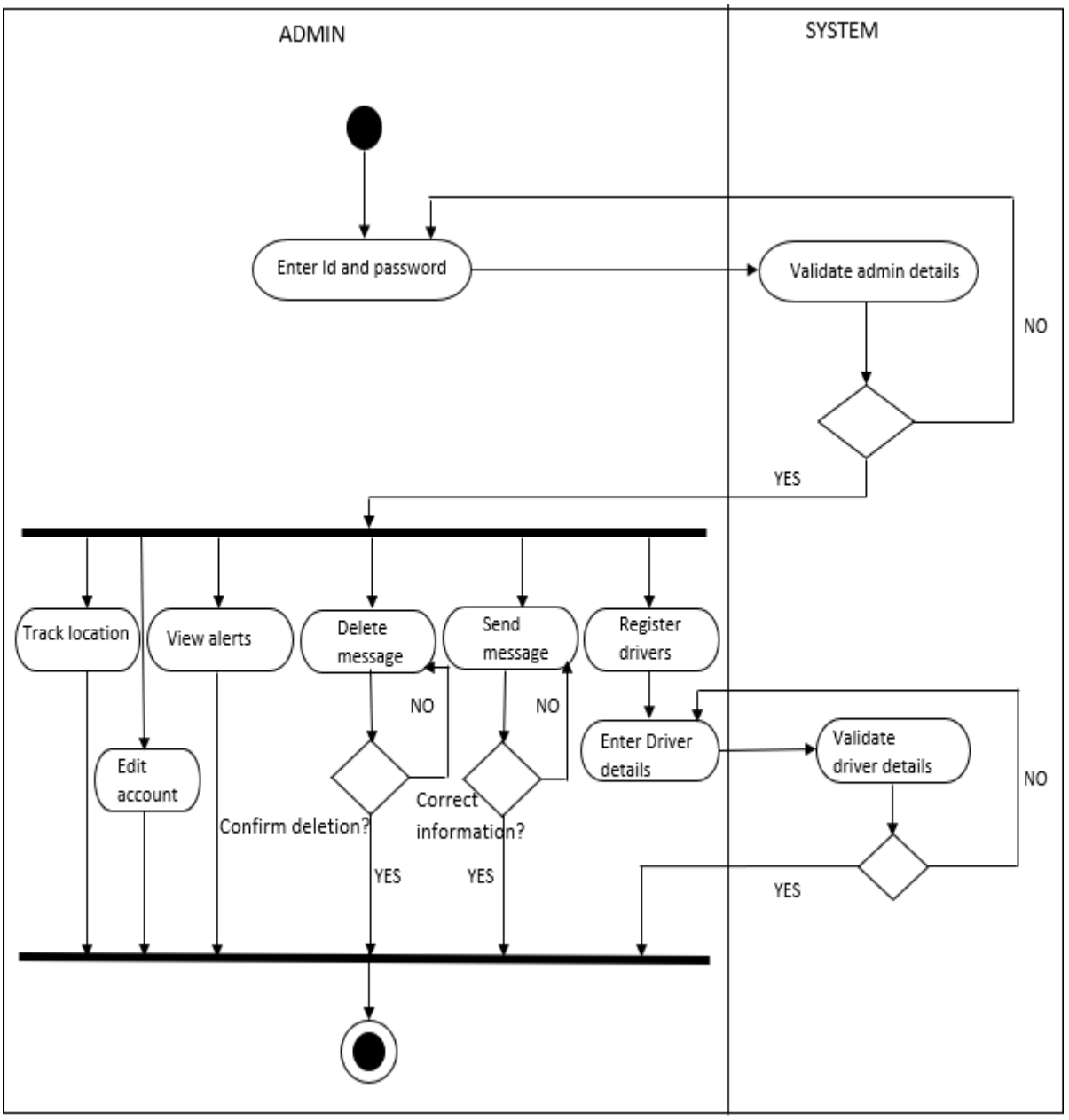


Figure 2. 16 Activity diagram for Admin

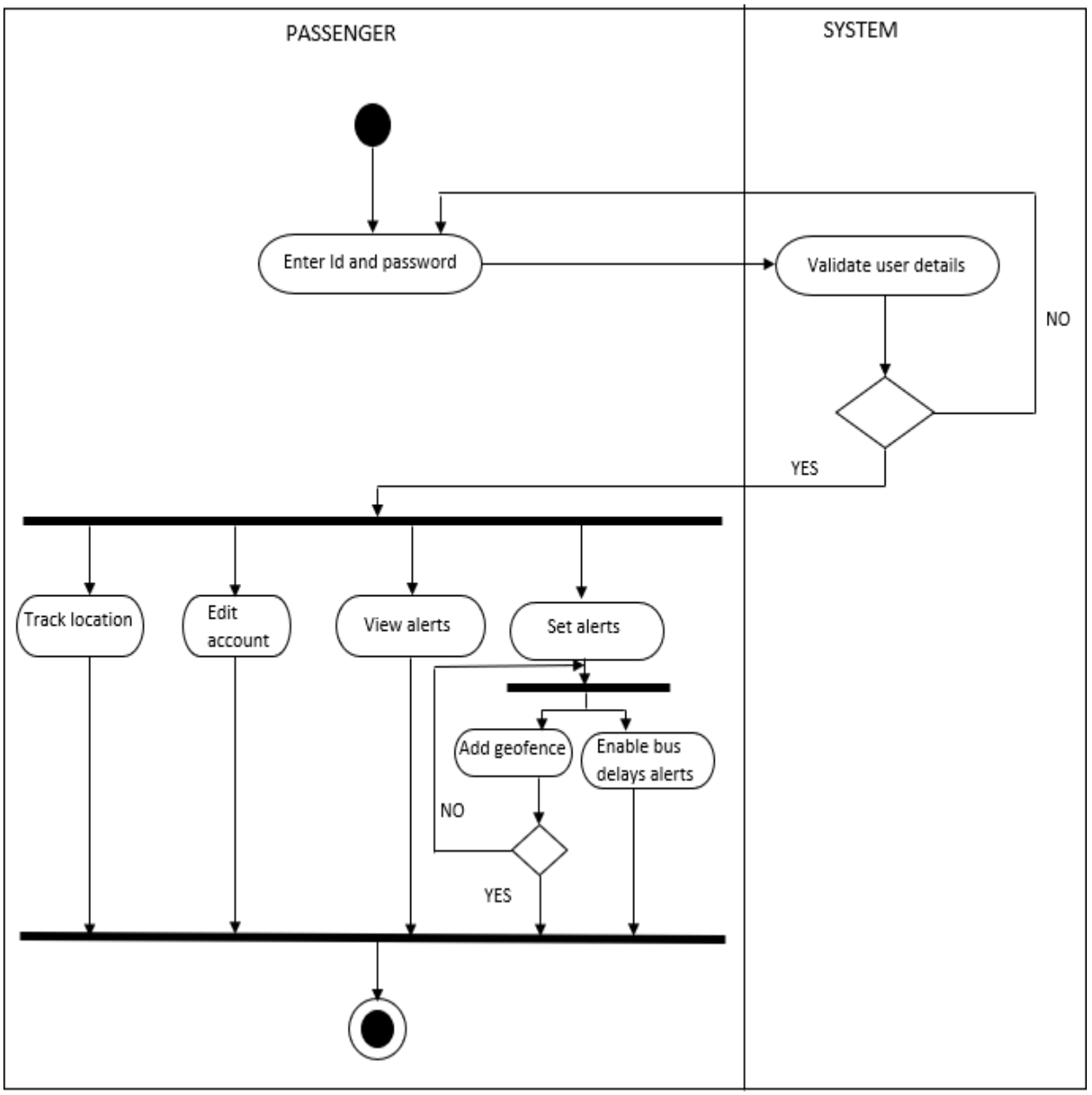


Figure 2. 17 Activity diagram for passenger

2.8 DOMAIN MODEL

Domain Modeling is a way to describe and model real world entities and the relationships between them, which collectively describe the problem domain space.

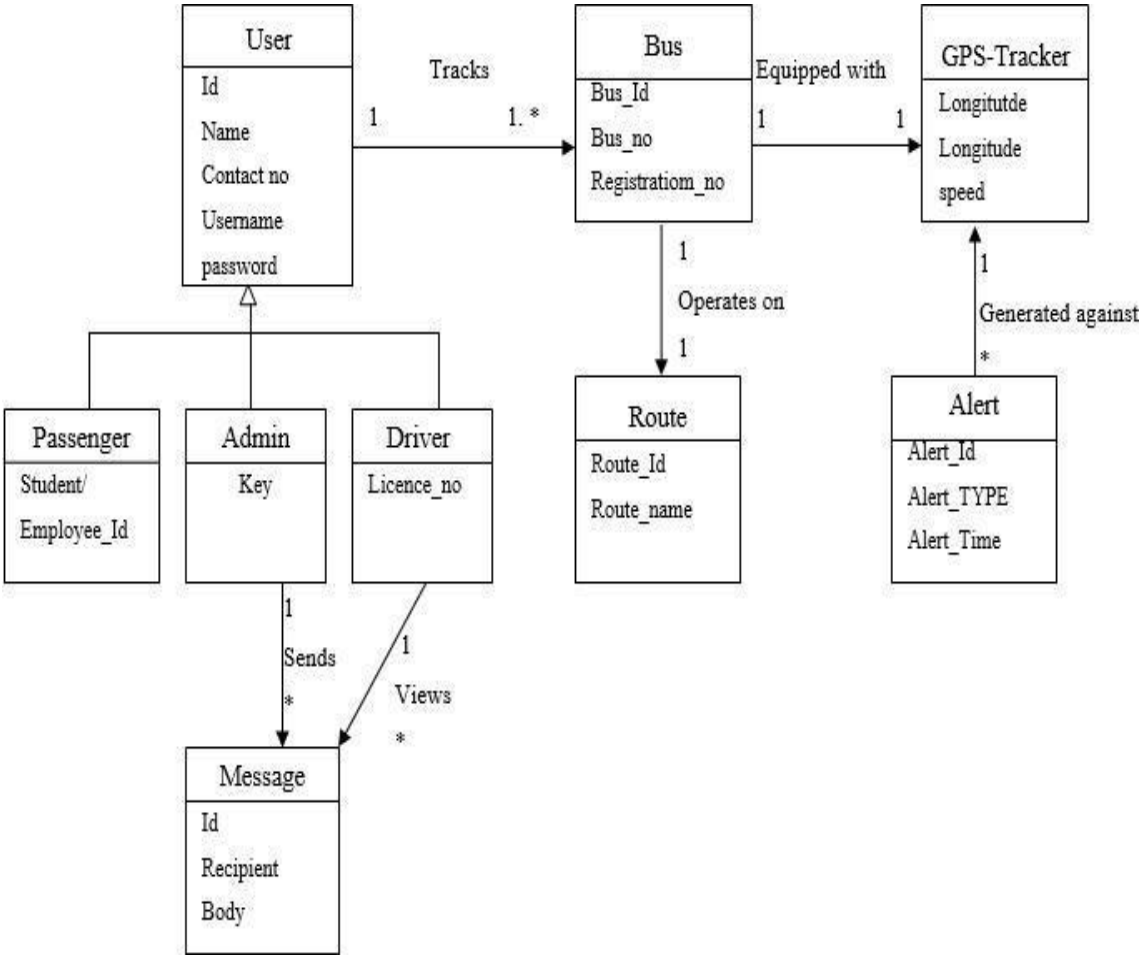


Figure 2. 18 Domain model

2.9 SOFTWARE QUALITY ATTRIBUTES

Software system attributes define overall factors that affect run-time behavior, application design, and user experience. To develop high quality application, software system attributes are the benchmarks that describe system's intended behavior within the environment for which it was built. Here is detail of some software system attributes.

2.9.1 Availability

The system will be available all the time but it requires internet connection and GPS to avail the services of this application.

2.9.2 Security

The security section describes the need to control access to the data. This includes controlling who may view and alter application data.

- Only registered drivers and passengers can use this application.
- A driver/passenger who uses this application should have a login id and password.
- Any modification (insert, delete, update) for the database will be synchronized and done only by the administrator.

2.9.3 Maintainability

- The program will use modular approach so that it would be easier to update or change code when needed.
- Updates in the software will be done in some separate files so that actual product will not be disturbed.

2.9.4 Portability

The system can be easily modified for a new environment. This software can be installed in any university, if modified properly.

2.10 DATABASE REQUIREMENTS

No SQL database will be used for this system to store all information. A No SQL database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations (ERD) used in relational databases. Relational databases rely on tables, columns, rows, or schemas to organize and retrieve data. In contrast, No SQL databases do not rely on these structures and use more flexible data models. No SQL can mean "not SQL" or "not only SQL." As RDBMS have increasingly failed to meet the performance, scalability, and flexibility needs that next-generation, data-intensive applications require, No SQL databases have been adopted by mainstream enterprises. No SQL is particularly useful for storing unstructured data, which is growing for more rapidly than structured data and does not fit the relational schemas of RDBMS. Data is synchronized across all clients in real-time, and remains available when your app goes offline. [3]

There are four categories of No SQL database:

1. Key-value data store
2. Document stores
3. Wide-column stores
4. Graph stores

I have used Mongo DB for my project which is document type as it requires to manipulate real-time data. Data is stored in JSON format.

CHAPTER 3
SOFTWARE DESIGN DESCRIPTION

3.1 INTRODUCTION

3.1.1 Overview

The Software Design Document is a document to provide documentation which will be used to aid in software development by providing the details for how the software should be built. Within the Software Design Document are narrative and graphical documentation of the software design for the project including architecture diagram, sequence diagrams, collaboration models, object behavior models, and other supporting requirement information. It includes the description of how the software will meet the requirements.

3.1.2 Purpose

The purpose of the Software Design Document is to provide a description of the design of a QAU Smart Ride fully enough to allow for software development to proceed with an understanding of what is to be built and how it is expected to build. The Software Design Document provides information necessary to provide description of the details for the software and system to be built.

3.1.3 Requirements Traceability Matrix

Requirement Traceability Matrix or RTM captures all requirements proposed by the client or development team and their traceability in a single document delivered at the conclusion of the life-cycle.
In other words, it is a document that maps and traces user requirement with test cases. The main purpose of Requirement Traceability Matrix is to see that all test cases are covered so that no functionality should miss while testing.

Figure 3. 1 Requirements Traceability Matrix

	Test case ID	**TC 1**	**TC 2**	**TC 3**	**TC 4**	**TC 5**	**TC 6**	**TC 7**	**TC 8**	**TC 9**	**TC 10**	**TC 11**	**TC 12**	**TC 13**	**# Test cases for respective requirements**
Req ID															
UC1		X	X				X	X	X	X	X	X	X	X	10
UC2			X												1
UC3		X		X											2
UC4					X										1
UC5				X		X									2
UC6		X					X								2
UC7		X						X							2
UC8		X							X						2
UC9		X								X					2
UC10		X									X				2
UC11		X									X	X			3
UC12		X											X		2
UC13		X												X	2

3.2 SYSTEM ARCHITECTURAL DESIGN

Architectural design is concerned with understanding how a system should be organized and designing the overall structure of that system. In the model of the software development process, architectural design is the first stage in the software design process. It is the critical link between design and requirements engineering, as it identifies the main structural components in a system and the relationships between them. The output of the architectural design process is an architectural model that describes how the system is organized as a set of communicating components.

3.2.1 Chosen System Architecture

Chosen system architecture is 3-tier. A three-tier architecture is a client-server architecture in which the functional process logic, data access, computer data storage and user interface are developed and maintained as independent modules on separate platforms. Three-tier architecture is a software design pattern and a well-established software architecture.

Three-Tier Architecture provides the following benefits.

Scalability—Each tier can scale horizontally. We can load-balance the Presentation tier among three servers to satisfy more Web requests without adding servers to the Application and Data tiers.

Performance—Because the Presentation tier can cache requests, network utilization is minimized, and the load is reduced on the Application and Data tiers

Availability—If the Application tier server is down and caching is sufficient, the Presentation tier can process Web requests using the cache.
For the admin panel, I have used MVC (Model View Controller) design pattern.

Model

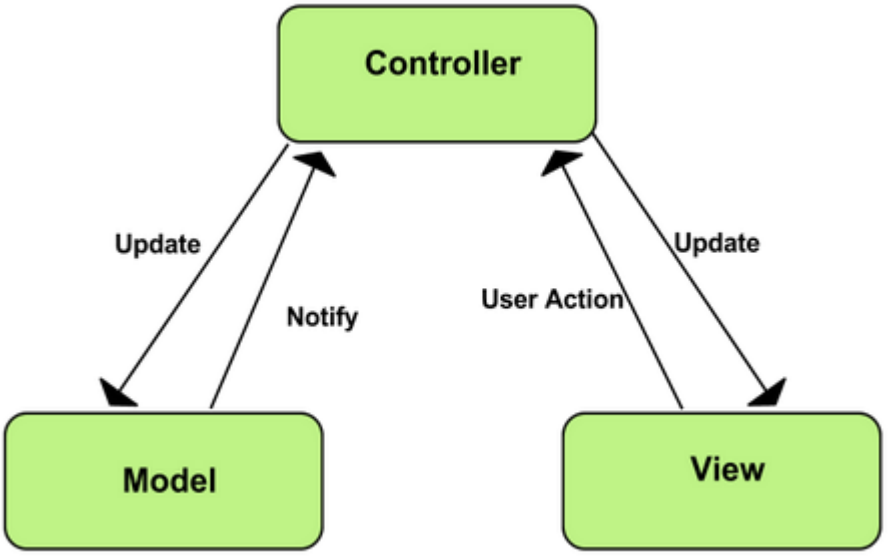
The Model component corresponds to all the data-related logic that the user works with. This can represent either the data that is being transferred between the View and Controller components or any other business logic-related data.

View

The View component is used for all the UI logic of the application.

Controller

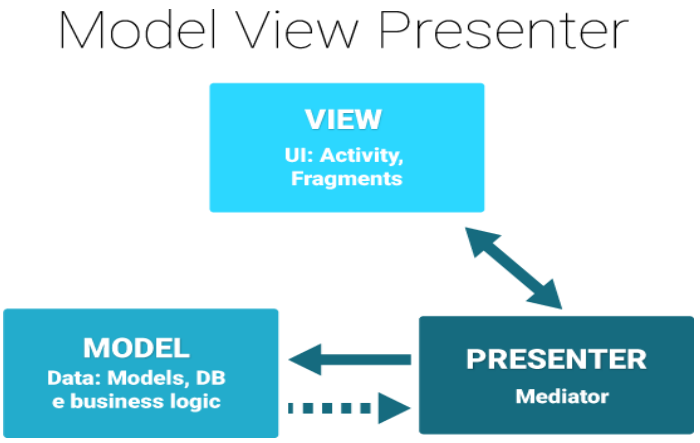
Controllers act as an interface between Model and View components to process all the business logic and incoming requests, manipulate data using the Model component and interact with the Views to render the final output.



For the android app, I have used MVP (Model View Presenter) design pattern.

MVP

Model view presenter is a derivation of the model–view–controller (MVC) architectural pattern. In MVP the presenter assumes the functionality of the "middle-man". In MVP, all presentation logic is pushed to the presenter.



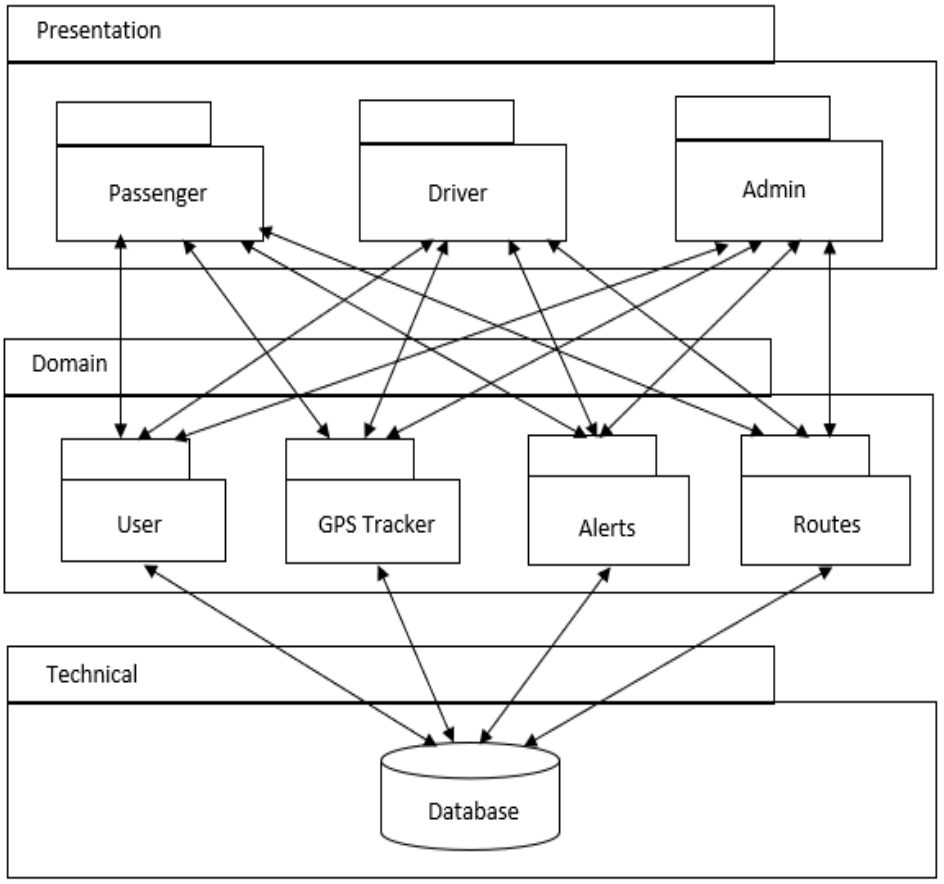


Figure 3. 2 System Architecture diagram

3.2.2 Discussion of Alternative Designs

An alternative design was to use two-tier architecture, based on Client Server. The two-tier architecture is like client server application. The direct communication takes place between client and server. There is no intermediate between client and server. Because of tight coupling a 2-tiered application will run faster. But the main problem of two tier architecture is the server cannot respond multiple request same time, as a result it causes a data integrity issue. Communication is easy but performance will be degraded upon increasing the users, So I rejected this design due to the performance issues and continued the above one.

3.2.3 System Interface Description

System interface describes the flow of resources. It is the logical characteristics of each interface between the software product and the hardware components of the system. Figure shows the software interface of the software. It clearly shows how different entities of system are interacting with each other.

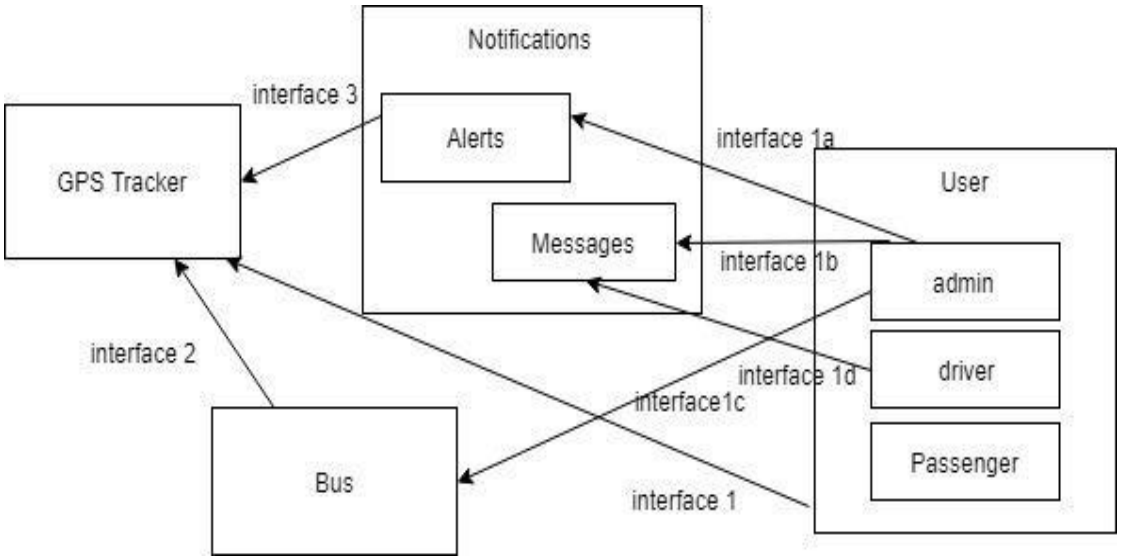


Figure 3. 3 System Interface Description

3.3 DETAILED DESCRIPTION OF COMPONENTS

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical components in a system.

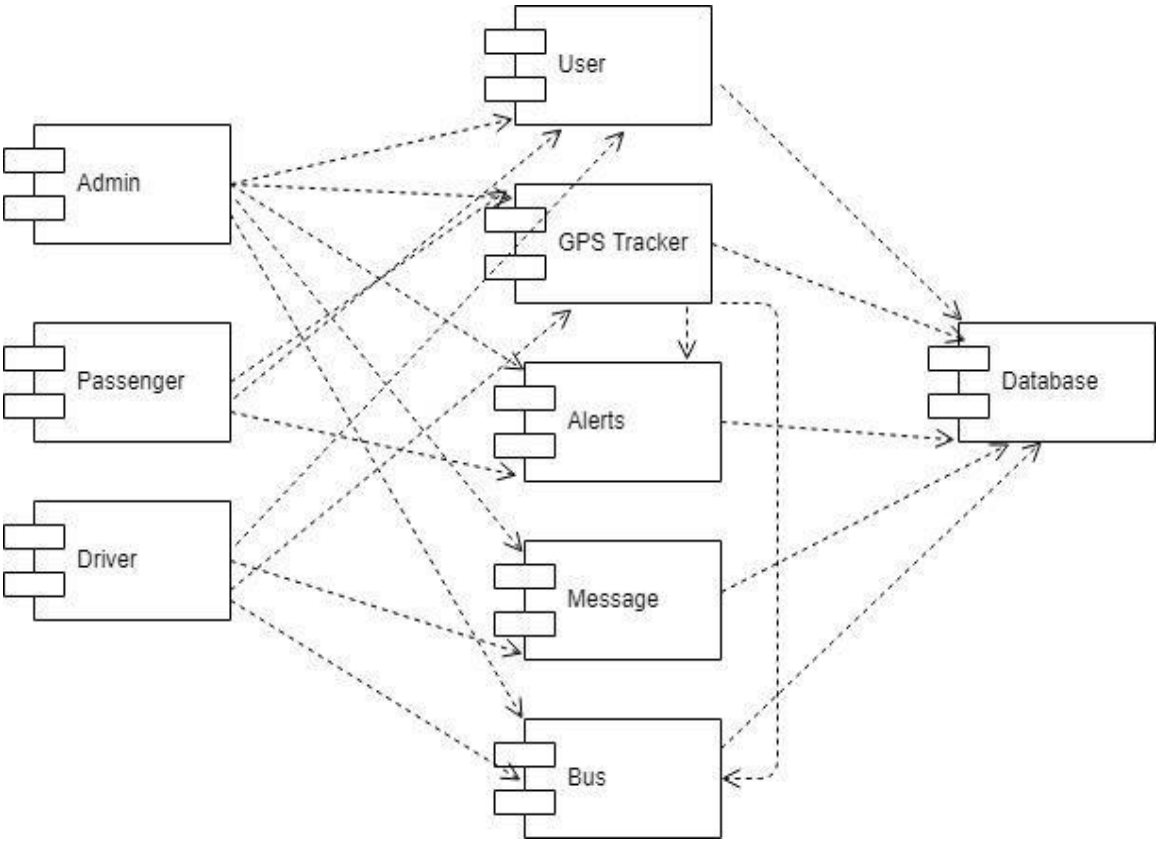


Figure 3. 4 Component diagram

Admin
This component interacts with users, alerts, GPS Tracker, message, bus class.
Admin can receive alerts when the bus exceeds the speed limit, enter or exit the geo-fence, manage drivers accounts, send messages to drivers on real-time.

Passenger
This component interacts with users, alerts and GPS Tracker class. Student/employee can track location of the buses, custom alerts such as geo-fence alert and can get estimated time of arrival of buses.

Driver
This component interacts with users, GPS Tracker, message and bus class.

GPS tracker
This component handles real time location of buses. This component is responsible for sending coordinates (longitude and latitude) of the buses to the server.

Alerts
This component is responsible for sending and managing alerts. This component is linked with integrated android sensor and detects the speed of the bus and the geo-fence area and generates alerts when specified conditions meet.

Message
This component contains messages. It handles the incoming and outgoing messages.

Bus
This component contains bus details for instance bus no, registration Id.

3.4 SEQUENCE DIAGRAM

A sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. It is a construct of a message sequence chart. A sequence diagram shows object interactions arranged in time sequence.

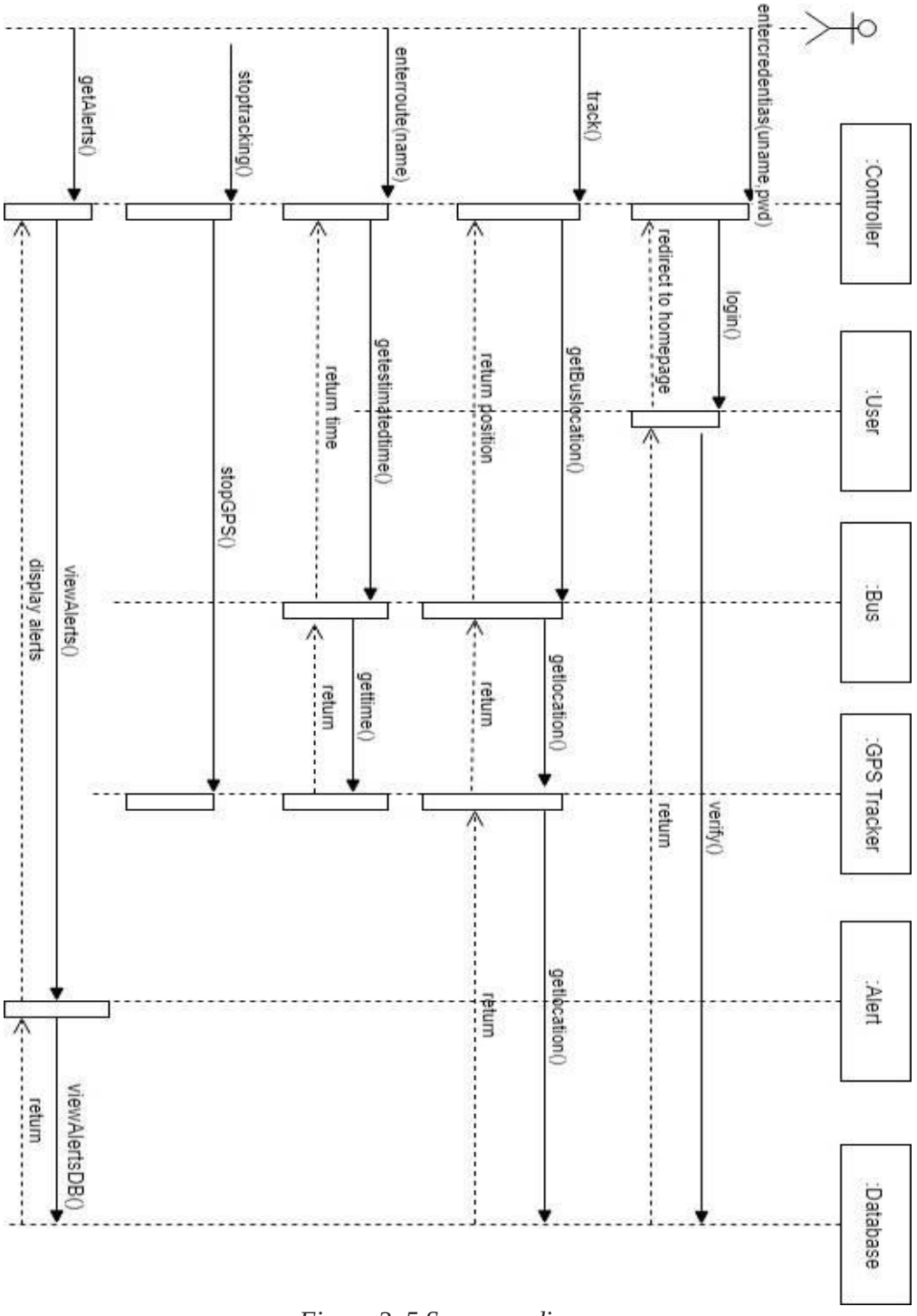


Figure 3. 5 Sequence diagram

3.5 CLASS DIAGRAM

A class diagram is an illustration of the relationships and source code dependencies among classes in the Unified Modeling Language (UML). In this context, a class defines the methods and variables in an object, which is a specific entity in a program or the unit of code representing that entity.[2]

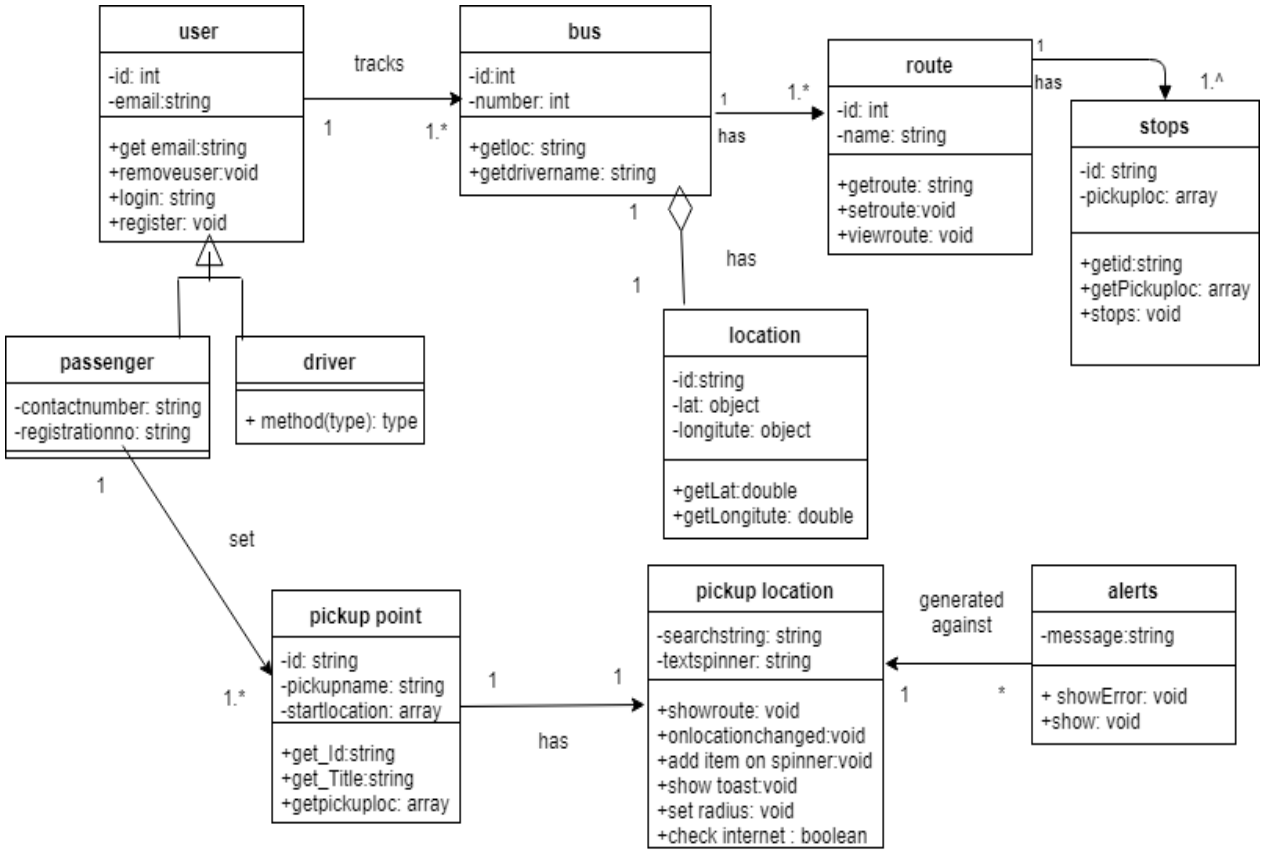


Figure 3. 6 Class diagram

3.6 USER INTERFACE DESIGN

3.6.1 Description of the User Interface

QAU Smart Ride should be designed for ease of use, providing help instructions, and appropriate error messages for user inputs. The QAU Smart Ride makes sure at every point, that the user spends most of the time using the application rather than figuring out how to use it. The background will be light in color with dark colored font to enhance the contrast and visibility. Controls which allow the user to interact with the application will be clear and imply their functionality within the application.

The home screen offers a menu with a list of functions that the application performs. The user can select one of the options on the menu, and is taken to the respective screen. Every screen displays the menu on the top. The user can click on any one of the options and is taken to the screen of their choice.

3.6.2 Screen Images

Following are few screen images of game.

INTERFACES FOR DRIVERS APPLICATION

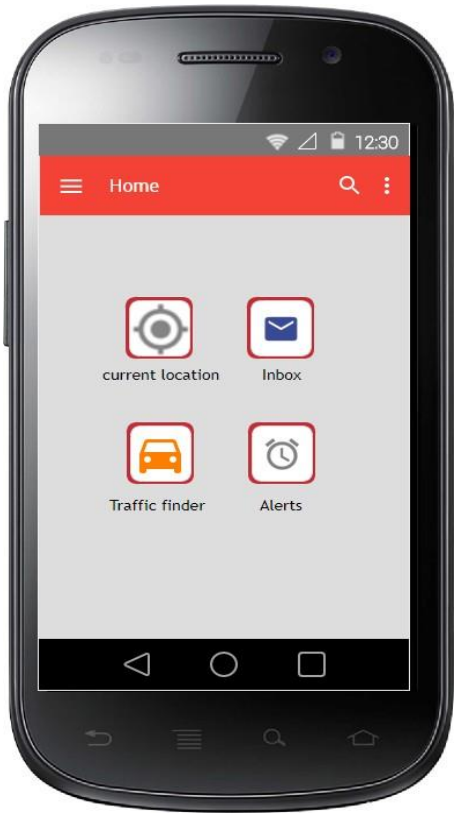


Figure 3. 7 Interface for Home screen

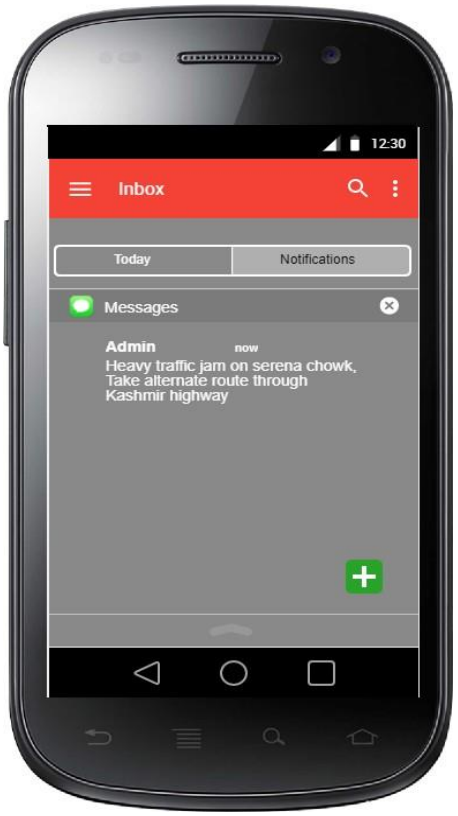


Figure 3. 8 Interface for Inbox

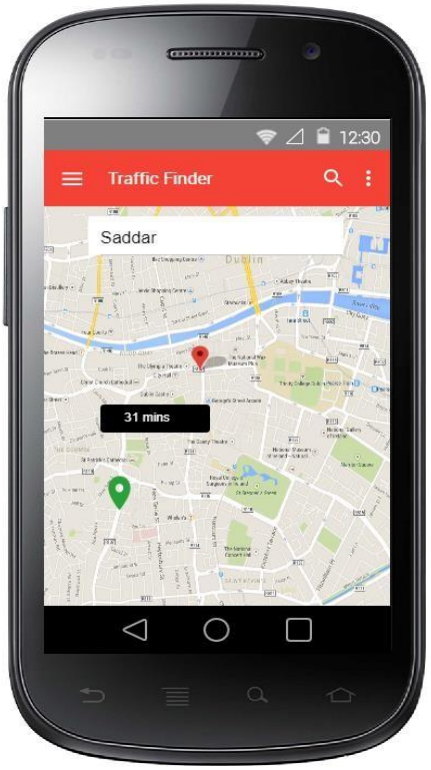


Figure 3. 9 Interface for traffic finder

INTERFACES FOR PASSENGER (STUDENT/EMPLOYEE) APPLICATION

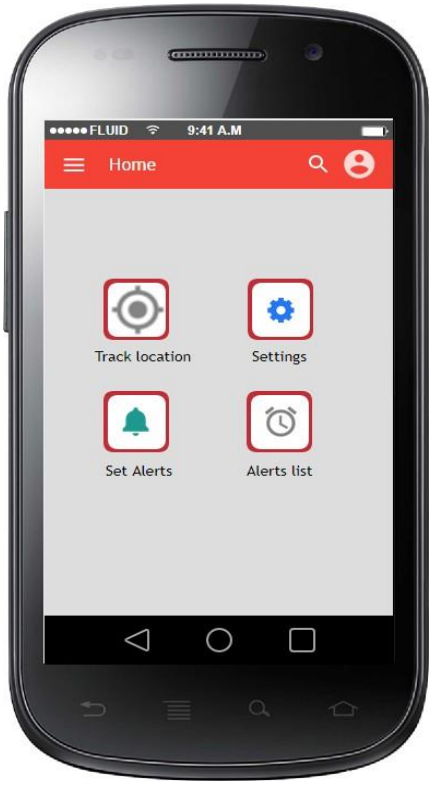


Figure 3. 10 Interface for home page

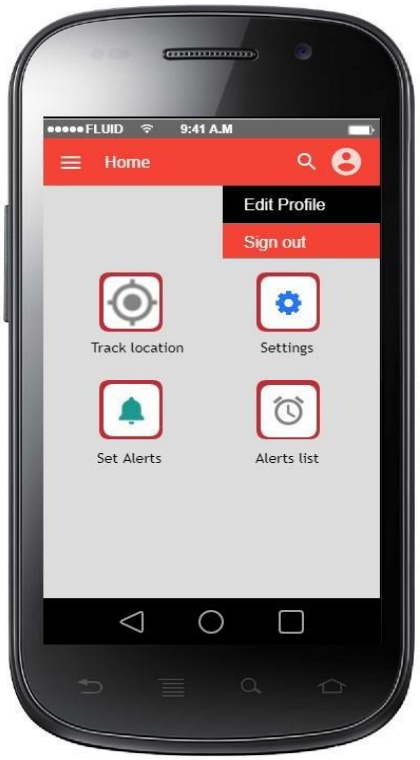


Figure 3. 11 Interface for selecting edit profile option

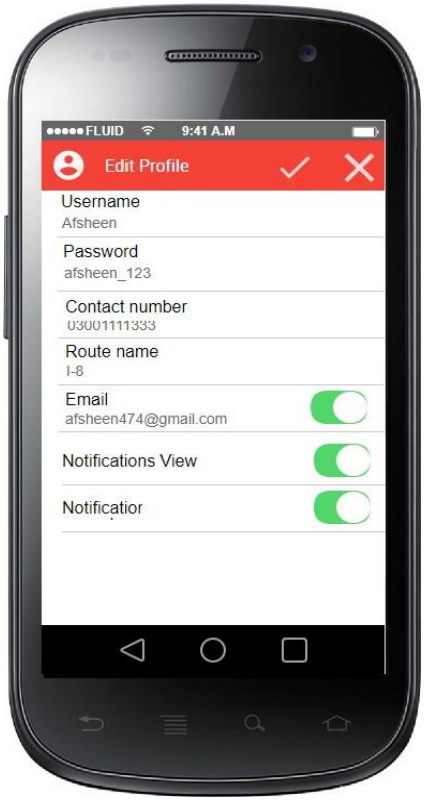


Figure 3. 12 Interface for Edit profile

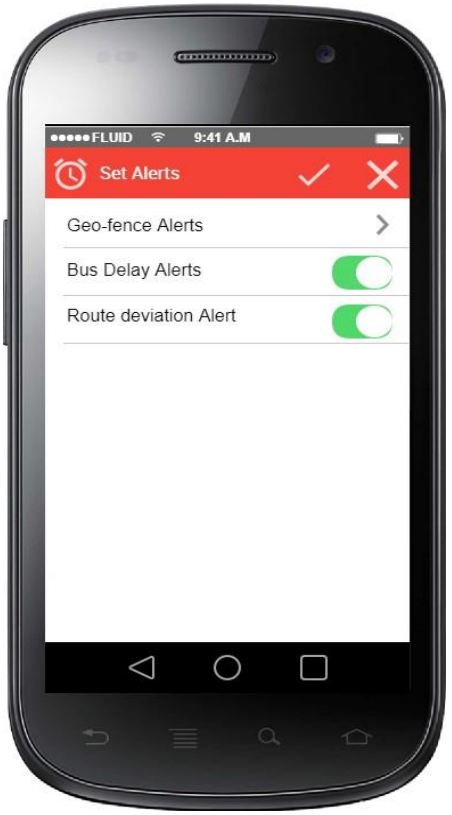


Figure 3. 13 Interface for set Alerts

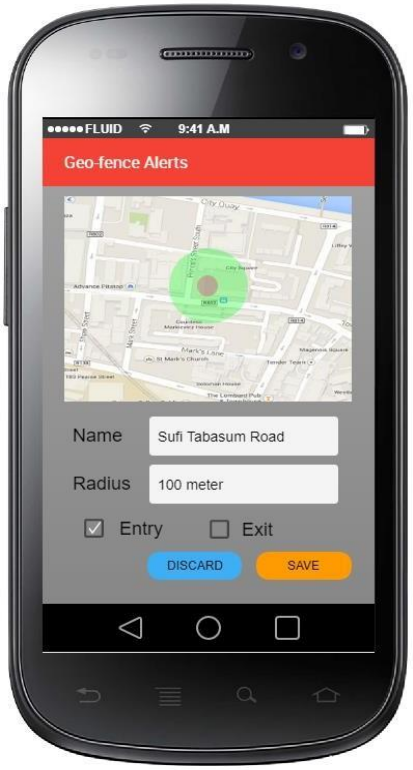


Figure 3. 14 Interface for Add geo -fence

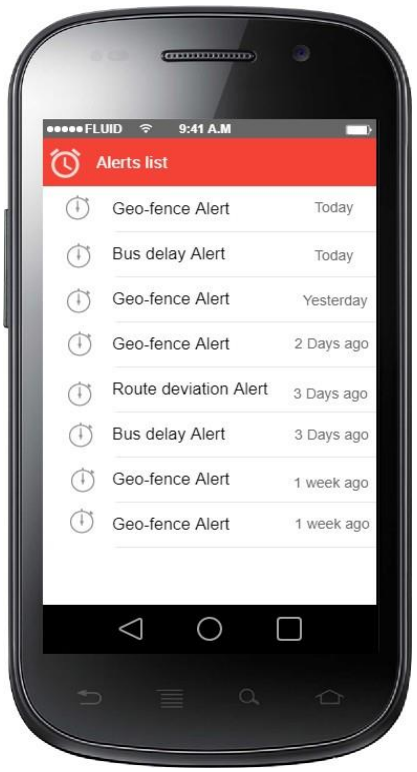


Figure 3. 15 Interface for Alerts list

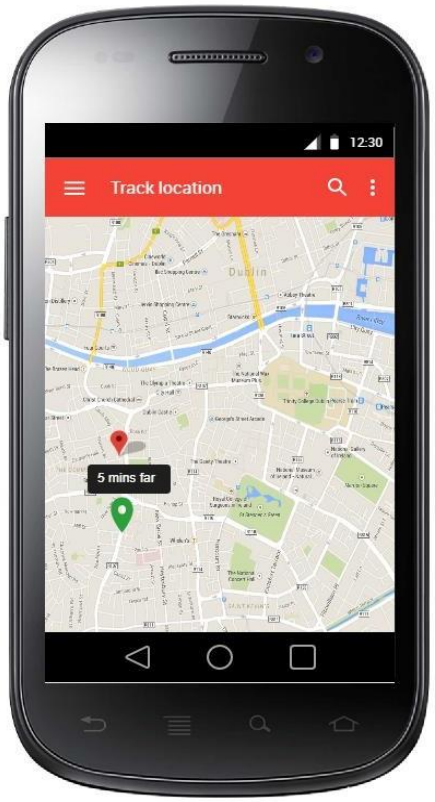


Figure 3. 16 Interface for track location

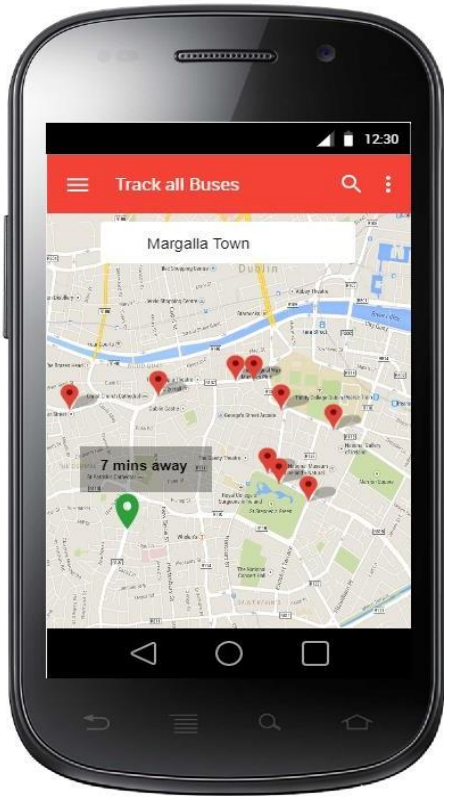


Figure 3. 17 Interface for tracking all buses

CHAPTER 4
SOFTWARE TEST DOCUMENTATION

4.1 INTRODUCTION

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. In simple words, testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements. According to ANSI/IEEE 1059 standard, Testing can be defined as - A process of analyzing a software item to detect the differences between existing and required conditions (that is defects/errors/bugs) and to evaluate the features of the software item.

4.1.1 System Overview

QAU Smart Ride will be designed to track and monitor Quaid-e-Azam University buses. It will allow admin to monitor buses and their speed. Admin will be notified if the buses move out of the route or over speed. Admin will be responsible for maintaining data of drivers, routes and stops. Also, admin can send messages to drivers on real time basis. The application also allows the drivers to check the current location of the bus and to send panic alert. Also, application let students, faculty members and staff of the university to track the location of the buses in real time along with the estimated time of arrival at their stops.

4.1.2 Test Approach

A test approach is the test strategy implementation of a project, defines how testing would be carried out.

Testing technique used for the QAU Smart Ride is beta testing (black box testing) for the time being. The main purpose of this testing is to evaluate the system's compliance with the business requirements and verify if it has met the required criteria for delivery to end users. Beta testing reduces product failure risks and provides increased quality of the product through customer validation.

Another approach that I have used is load testing to determine the system's behavior under both normal and anticipated peak load conditions. This type of testing has been done to check the performance of the product.

4.2 TEST PLANS

4.2.1 Features to be tested

Features to be tested are all according to user perspective. For example

1. Login
2. Logout
3. Register account
4. Update account
5. Delete account
6. Register a bus
7. Send Message
8. View message

9. Delete message
10. Track buses location
11. Allocate routes
12. Send panic alert
13. Set geo-fencing alert
14. Add stop
15. View routes

4.2.2 Features not to be tested

Features not to be tested are from the developer's point of view. For example

- power used by processor
- memory consumed by the QAU Smart Ride
- Software risk factor
- Maintainability of QAU Smart Ride

4.2.3 Testing tools and environment

Following tools and environment are used for testing:

- Web browser
- PC/laptop
- Android phone/tablet
- Windows operating system

4.3 TEST CASES

A test case describes an input, action, or an event and an expected response, to determine if a feature of a software application is working correctly.

TC-1: Login

Table 4.1 is a test case for login. This test case tells us about testing of login scenario.

Table 4. 1 Testcase for Login

ID	TC-1
DESCRIPTION	Check that the registered user can successfully login account.
SETUP	1. Create an account for user with username “afsheen123” and password “123abc”.
INSTRUCTIONS	1. Display login panel. 2. Enter username “afsheen123”. 3. Enter password “123abc”. 4. Click the submit button.
EXPECTED RESULT	User has successfully login.
ACTUAL RESULT	User has successfully login.
VERDICT	Pass

TC-2: Logout

Table 4.2 is a test case for logout. This test case tells us about testing of logout scenario.

Table 4. 2 Testcase for logout

ID	TC-2
DESCRIPTION	Check that the logged-in user has successfully logged out.
SETUP	1. Display login panel. 2. Enter username “afsheen123”. 3. Enter password “123abc”. 4. Click the submit button.
INSTRUCTIONS	1. Click the logout button
EXPECTED RESULT	System displays log in page.
ACTUAL RESULT	System displays log in page.
VERDICT	Pass

TC-3: Register Account

Table 4.3 is a test case for register account. This test case tells us about testing of register account scenario.

Table 4. 3 Testcase for Register Account

ID	TC-3
DESCRIPTION	Check that the logged-in user has successfully logged out.
SETUP	1. Display the registration screen.
INSTRUCTIONS	2. Enter name “Ali”. 3. Enter username “ali123”. 4. Enter password “xxxxxx”. 5. Enter email“ali123@gmail.com”. 6. Enter contact number “03001233333”. 7. Click the submit button.
EXPECTED RESULT	“Successful registration” message is displayed.
ACTUAL RESULT	“Successful registration” message is displayed.
VERDICT	Pass

TC-4: Update Account

Table 4.4 is a test case for update account. This test case tells us about testing of updating account scenario.

Table 4. 4 Testcase for Update Account

ID	TC-4
DESCRIPTION	Check that the user can update his/her account information.
SETUP	1. Provide name in the name textbox “Ali”. 2. Provide username in the username textbox “ali123”. 3. Provide password in the password textbox “xxxxxx”. 4. Provide email in the email textbox “ali123@gmail.com”. 5. Provide number in the number textbox “03001233333”.
INSTRUCTIONS	1. Clear the password textbox. 2. Enter the new password “ali666”. 3. Click the submit button.
EXPECTED RESULT	System has updated the information.
ACTUAL RESULT	System has updated the information.
VERDICT	Pass

TC-5: Delete Account

Table 4.5 is a test case for delete account. This test case tells us about testing of deleting account scenario.

Table 4. 5 Testcase for Delete Account

ID	TC-5
DESCRIPTION	Check that the user can delete account.
SETUP	1. Login account with username and password. 2. Display the profile page.
INSTRUCTIONS	1. Click the delete button.
EXPECTED RESULT	Account deleted successfully.
ACTUAL RESULT	Account deleted successfully.
VERDICT	Pass

TC-6: Send message

Table 4.6 is a test case for sending message. This test case tells us about testing of sending message scenario.

Table 4. 6 Testcase for Send Message

ID	TC-6
DESCRIPTION	To check that the admin can send messages to drivers.
SETUP	Login account with admin username and password.
INSTRUCTIONS	1. Display compose message screen. 2. Enter the recipient name. 3. Enter the message. 4. Click send button
EXPECTED RESULT	Message sent to the driver.
ACTUAL RESULT	Message sent to the driver.
VERDICT	Pass

TC-7: View message

Table 4.7 is a test case for viewing message. This test case tells us about testing of viewing message scenario.

Table 4. 7 Testcase for View message

ID	TC-7
DESCRIPTION	To check that the driver can receive message.
SETUP	1. Login account with driver username and password.
INSTRUCTIONS	2. Display message screen from driver panel.
EXPECTED RESULT	Driver received the message.
ACTUAL RESULT	Driver received the message.
VERDICT	Pass

TC-8: Track bus location

Table 4.8 is a test case for tracking bus location. This test case tells us about testing of tracking bus location scenario.

Table 4. 8 Testcase for track bus location

ID	TC-8
DESCRIPTION	To check that the user can track the exact location of the bus.
SETUP	1. Login account with username and password.
INSTRUCTIONS	1. Enter the track bus option 2. Google maps points the location of the bus on the map.
EXPECTED RESULT	Exact position of the bus will be tracked.
ACTUAL RESULT	Exact position of the bus will be tracked.
VERDICT	Pass

TC-9: Delete message

Table 4.9 is a test case for deleting message. This test case tells us about testing of deleting message scenario.

Table 4. 9 Testcase for Delete message

ID	TC-9
DESCRIPTION	To check that the driver can delete message.
SETUP	1. Login account with driver username and password.
INSTRUCTIONS	1. Display message screen from driver panel. 2. Select the message. 3. Click the delete button.
EXPECTED RESULT	Driver received the message.
ACTUAL RESULT	Driver received the message.
VERDICT	Pass

TC-10: Allocating Routes

Table 4.10 is a test case for allocating routes. This test case tells us about testing of allocating routes scenario.

Table 4. 10 Testcase for Allocating routes

ID	TC-10
DESCRIPTION	To check that the route has been allocated to the driver.
SETUP	1. Login account with admin email and password. 2. Select the route from the navigation bar. 3. Click the add new route button.
INSTRUCTIONS	1. Enter the route name “Saddar”. 2. Enter the bus no “38”. 3. Enter the driver name “Ali”. 4. Click the submit button.
EXPECTED RESULT	New route has been allocated to the driver.
ACTUAL RESULT	New route has been allocated to the driver.
VERDICT	Pass

TC-11: Viewing Routes

Table 4.11 is a test case for viewing routes. This test case tells us about testing of viewing routes scenario.

Table 4. 11 Testcase for viewing routes

ID	TC-11
DESCRIPTION	To check that the new route has been added to the system.
SETUP	1. Login account with admin email and password.
INSTRUCTIONS	1. Display route screen from the admin panel.
EXPECTED RESULT	Route appears on the map along with start and end location markers.
ACTUAL RESULT	Route appears on the map along with start and end location markers
VERDICT	Pass

TC-12: Add Stops

Table 4.12 is a test case for adding stops. This test case tells us about testing of adding stops scenario.

Table 4. 12 Testcase for Add stops

ID	TC-12
DESCRIPTION	To check that the stops have been added to the route.
SETUP	1. Login account with admin username and password. 2. Select route from the navigation bar. 3. Click add route button.
INSTRUCTIONS	1. Display the map on the screen. 2. Put the marker on the map. 3. Enter the stop name “Gold house”. 4. Click the OK button.
EXPECTED RESULT	Stop has been added to the route.
ACTUAL RESULT	Stop has been added to the route.
VERDICT	Pass

TC-13: View Stops

Table 4.13 is a test case for viewing stops. This test case tells us about testing of viewing stops scenario.

Table 4. 13 Testcase for viewing stops

ID	TC-13
DESCRIPTION	To check that the new stop has been added to the route.
SETUP	Login account with admin username and password.
INSTRUCTIONS	Display route screen from the admin panel.
EXPECTED RESULT	System displays stop on the map.
ACTUAL RESULT	System displays stop on the map.
VERDICT	Pass

TC-14: Sending Panic Alert

Table 4.14 is a test case for sending panic alerts. This test case tells us about testing of panic alert sent by the driver.

Table 4. 14 Testcase for sending panic alert

ID	TC-14
DESCRIPTION	To check that the driver can send panic alert.
SETUP	1. Login account with driver username and password. 2. Home screen appears.
INSTRUCTIONS	1. Click the panic alert icon 2. Popup appears on the screen “panic alert sent to admin”.
EXPECTED RESULT	Confirmation message appears on the screen.
ACTUAL RESULT	Confirmation message appears on the screen.
VERDICT	Pass

TC-15: Setting Geo-fencing Alert

Table 4.15 is a test case for setting geo-fencing alert. This test case tells us about testing of geo-fencing alert set by the passenger.

Table 4. 15 Testcase for setting geo-fencing alert

ID	TC-15
DESCRIPTION	To check that the passenger set the geo-fencing alert.
SETUP	1. Login account with passenger username and password. 2. Home screen appears. 3. Select the set alert screen from navigation panel.
INSTRUCTIONS	1. Select the stop “Gold house” from the stops. 2. Select the radius “300” meters. 3. Click the submit button.
EXPECTED RESULT	Popup message appears on the screen.
ACTUAL RESULT	Popup message appears on the screen.
VERDICT	Pass

CHAPTER 5
SOFTWARE IMPLEMENTATION DOCUMENT

5.1 INTRODUCTION

This document describes the project implementation for developing the QAU Smart Ride.

5.1.1 Language Selection

MEAN Stack

I have used MEAN stack technology for the development of admin panel. MEAN stack refers to a collection of JavaScript based technologies used to develop web applications. MEAN is an acronym for MongoDB, Express JS, AngularJS and Node.js. From client to server to database, MEAN is full stack JavaScript.

Mongo DB

Used for database. Mongo DB is a document-oriented NoSQL database Each database contains collections which in turn contains documents. Each document can be different with varying number of fields. The size and content of each document can be different from each other.

AngularJS

AngularJS is a JavaScript framework. It is used for designing front-end of Admin Panel.

JAVA

Java is used for the implementation of driver and passenger apps. Java is an object-oriented language that enables fewer dependencies in implementation.

5.1.2 Tools Selection

Tools that are used in implementation are:

IDE

- ***Android Studio***

Android Studio is used for the implementation of android apps. Android studio is IDE for developing android apps.

- ***Studio 3T***

Studio 3T is used for MongoDB. Studio 3T is GUI based.

NodeJS

NodeJS is used for server-side development of Admin panel. Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript. [5]

Notepad++
Used for writing code of admin panel.

5.1.3 Resources

- **Google Maps API**

Used for map integration.

- **Nexmo**

API used for SMS messaging.

- **Firebase Cloud Messaging (FCM)**

Used for push notifications in android apps.

- **OneSignal**

Plugin used for push notifications in admin panel. [6]

- **Mongoose**

Plugin used as intermediate between MongoDB and Node JS.

5.2 APPLICATION SCREENSHOTS

Here are the screenshots of my application.

INTERFACES FOR DRIVER VERSION OF APP

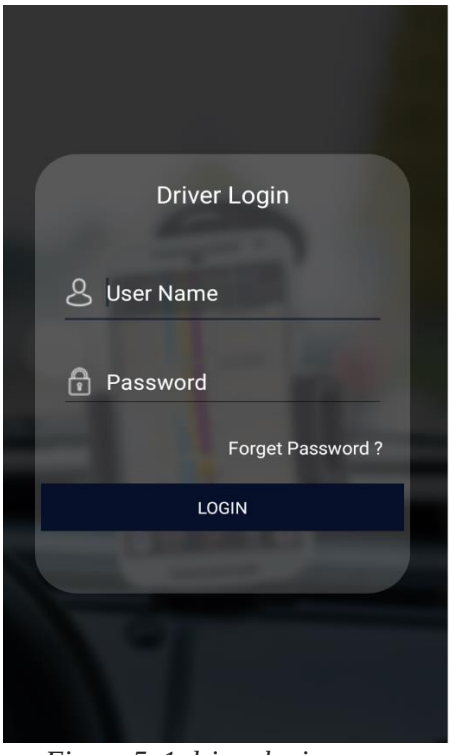


Figure 5. 1 driver login screen

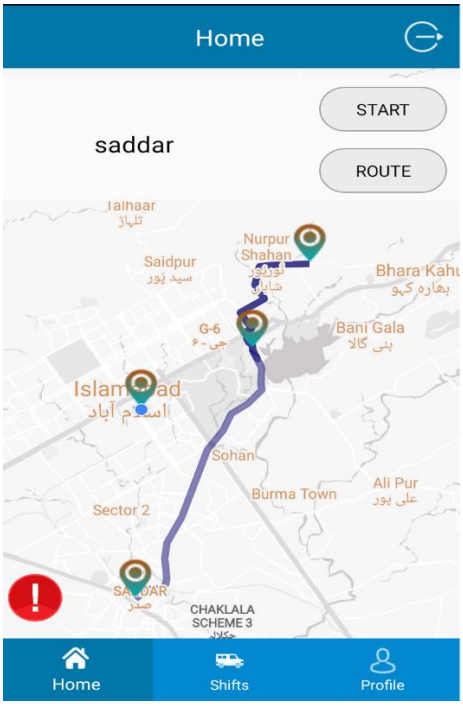


Figure 5. 2 driver home screen

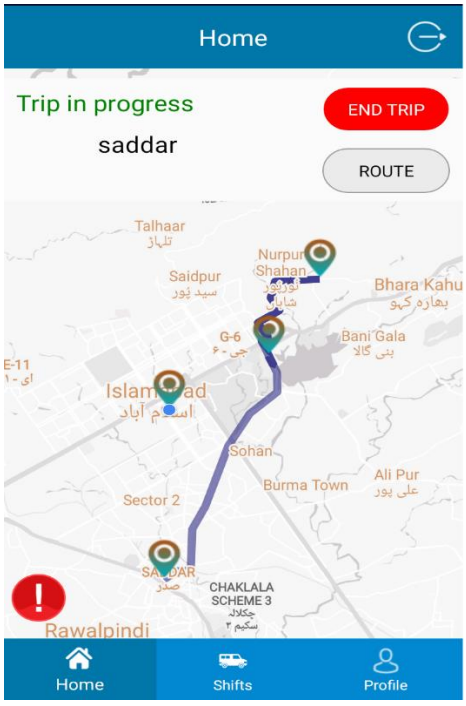


Figure 5. 3 driver trip started screen

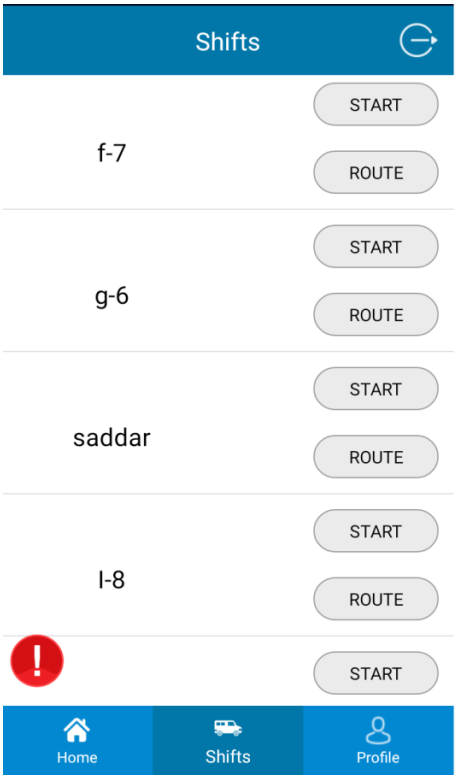


Figure 5. 4 driver shifts screen

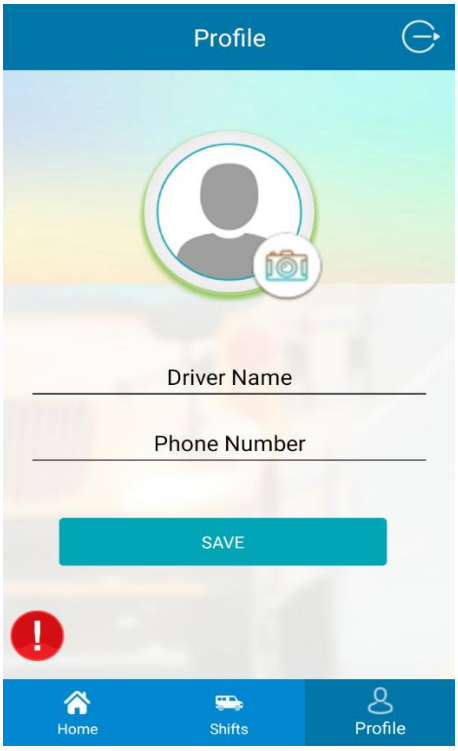


Figure 5. 5 driver update profile screen

INTERFACES FOR PASSENGER VERSION OFAPP

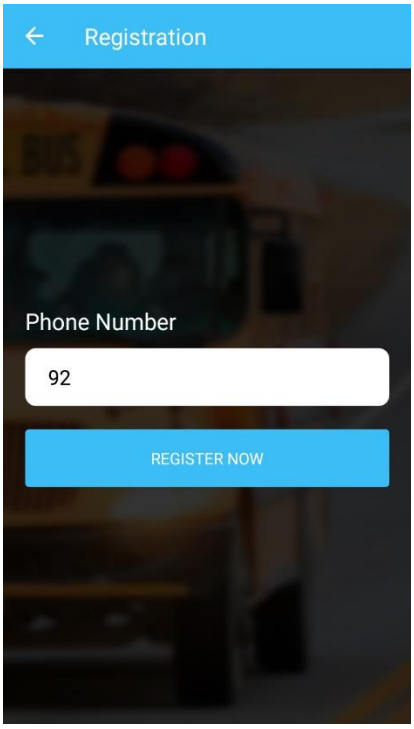


Figure 5. 6 passenger registration screen

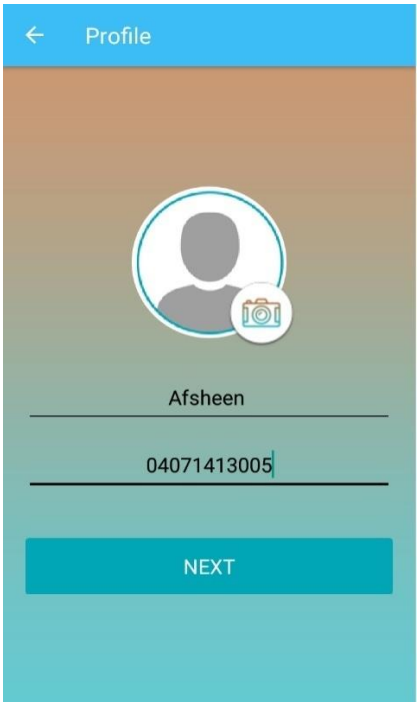


Figure 5. 7 passenger profile screen

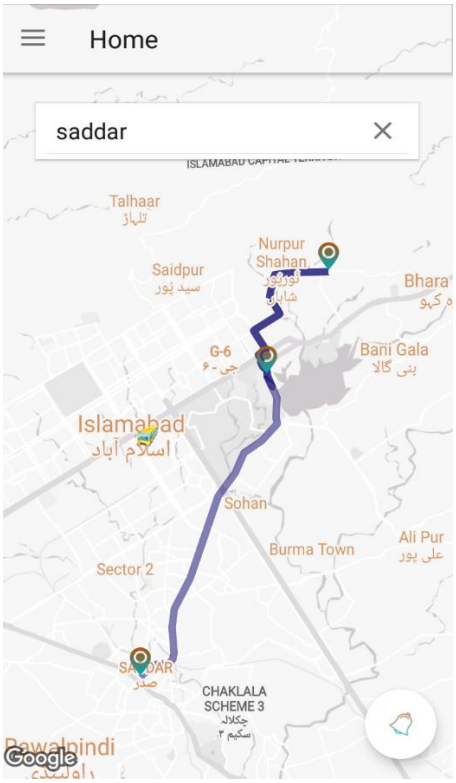


Figure 5. 8 passenger tracking screen

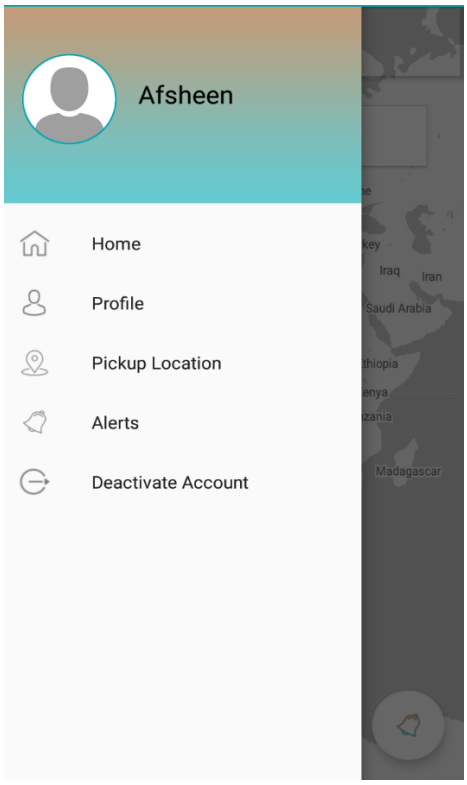


Figure 5. 9 passenger navigation screen

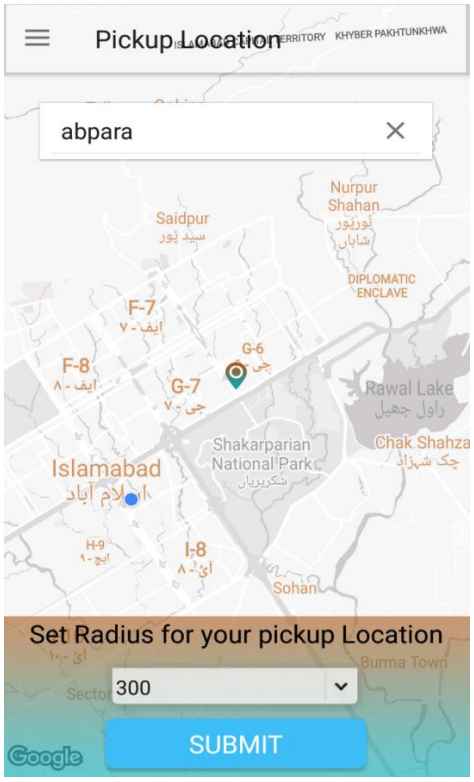


Figure 5. 10 passenger set geo-fence screen

INTERFACES FOR ADMIN VERSION OF APP

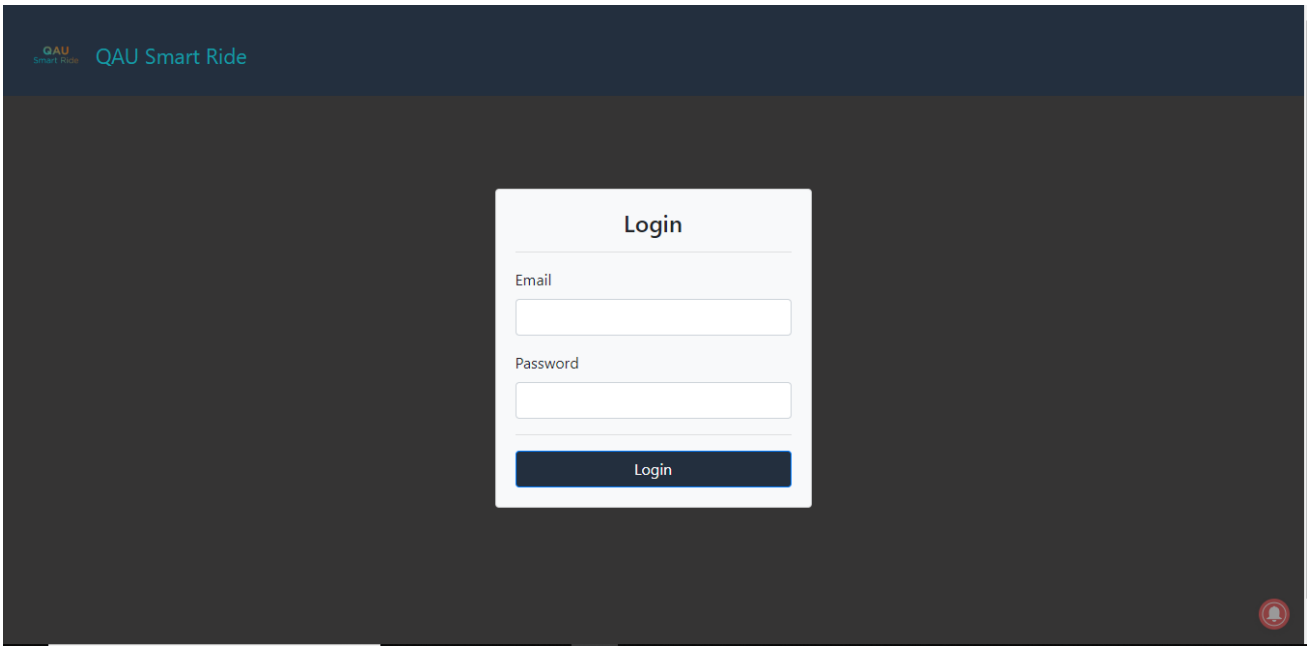


Figure 5. 11 Admin login screen

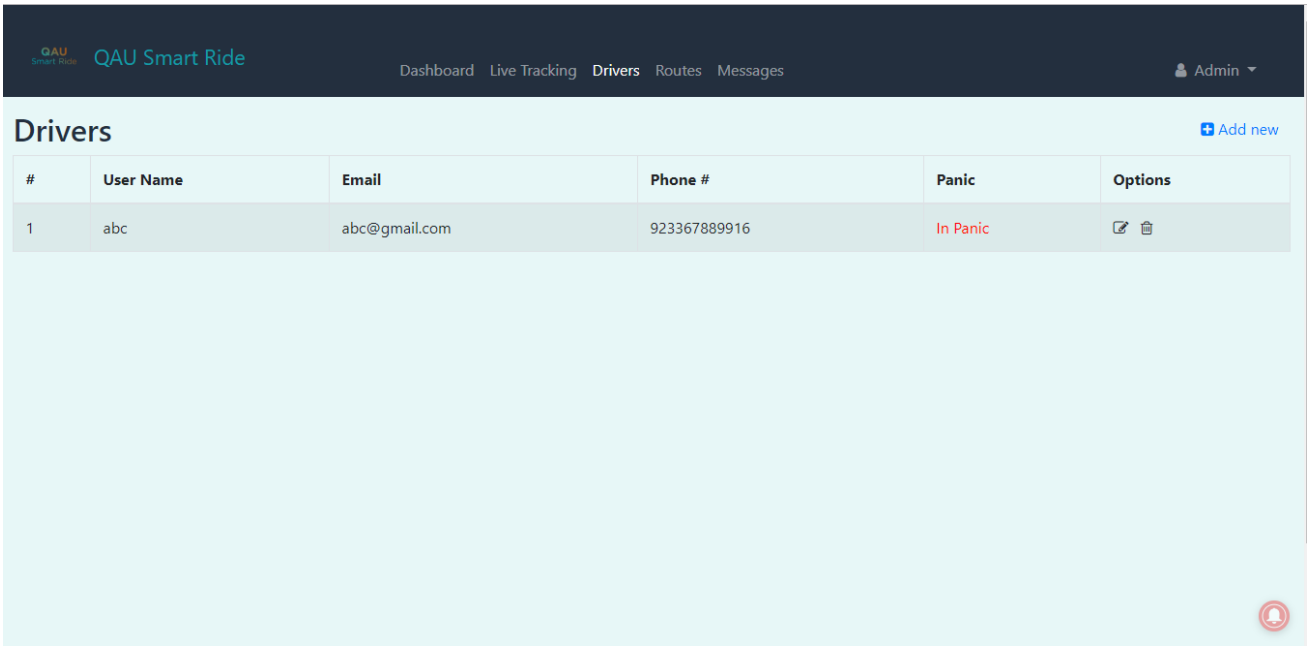


Figure 5. 12 drivers list screen

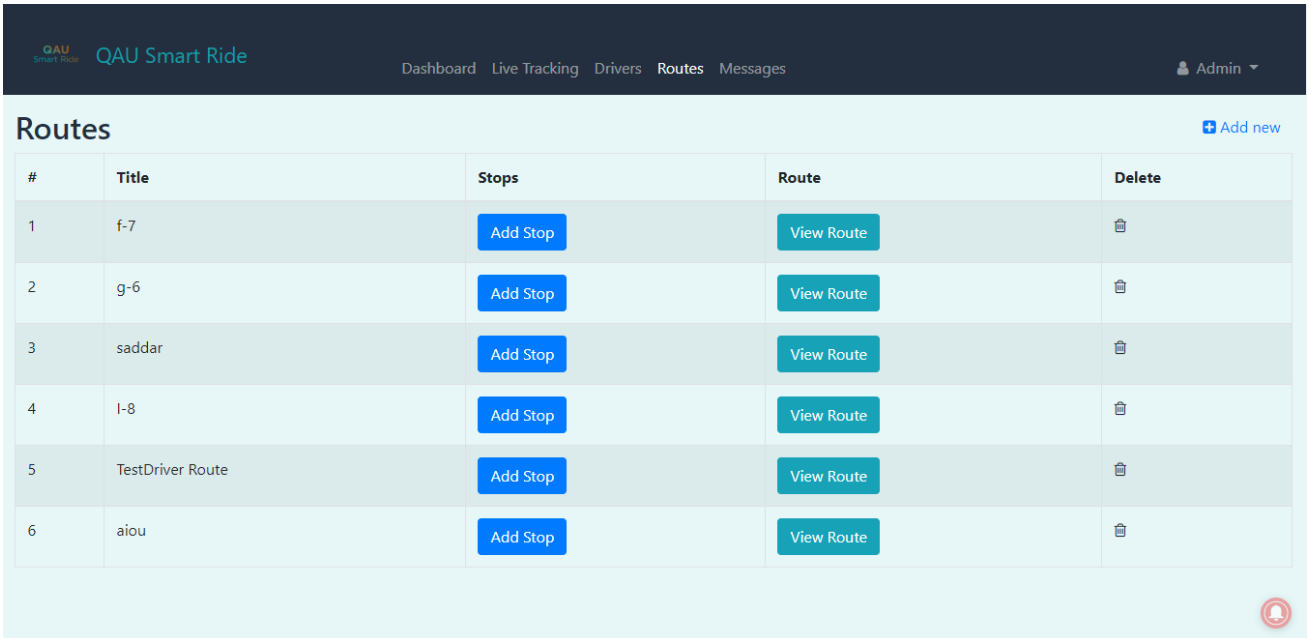


Figure 5. 13 route list screen

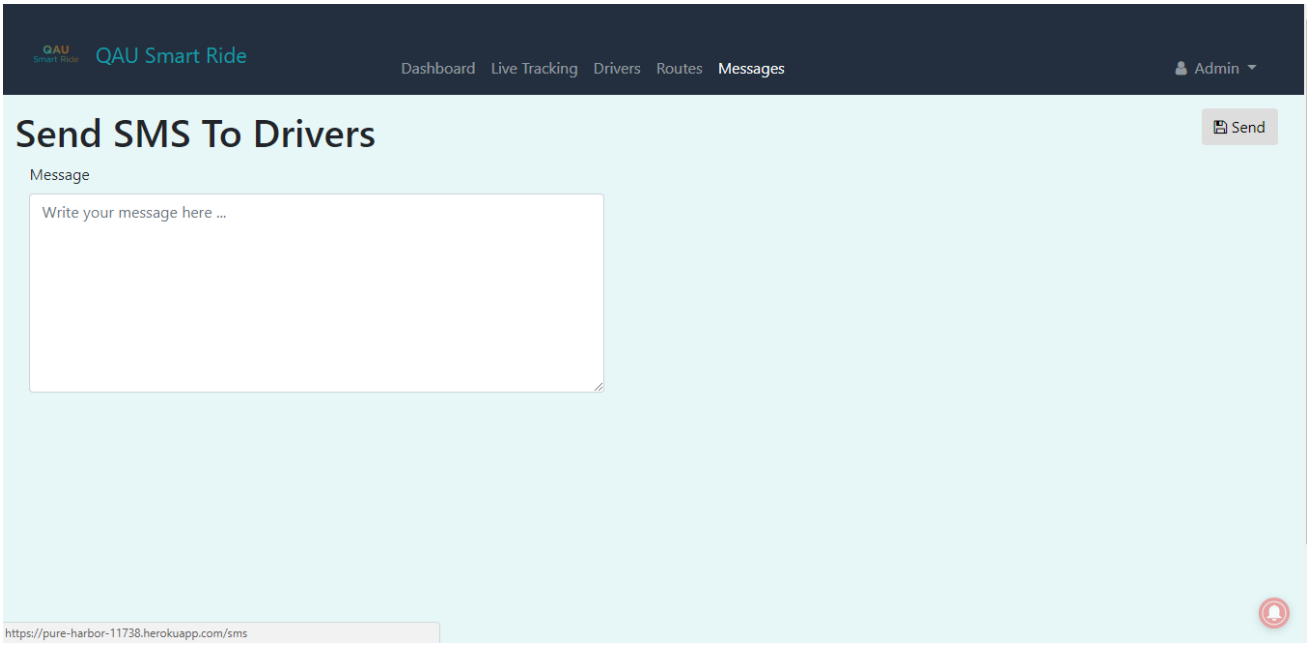


Figure 5. 14 send message screen

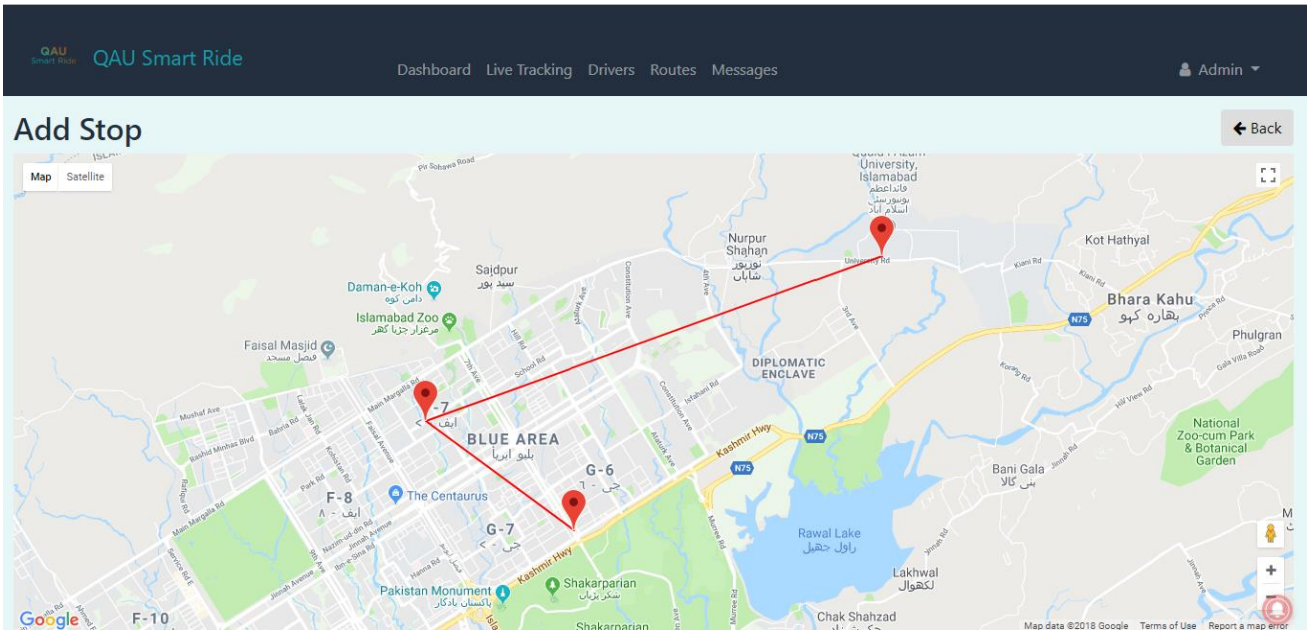


Figure 5. 15 route screen

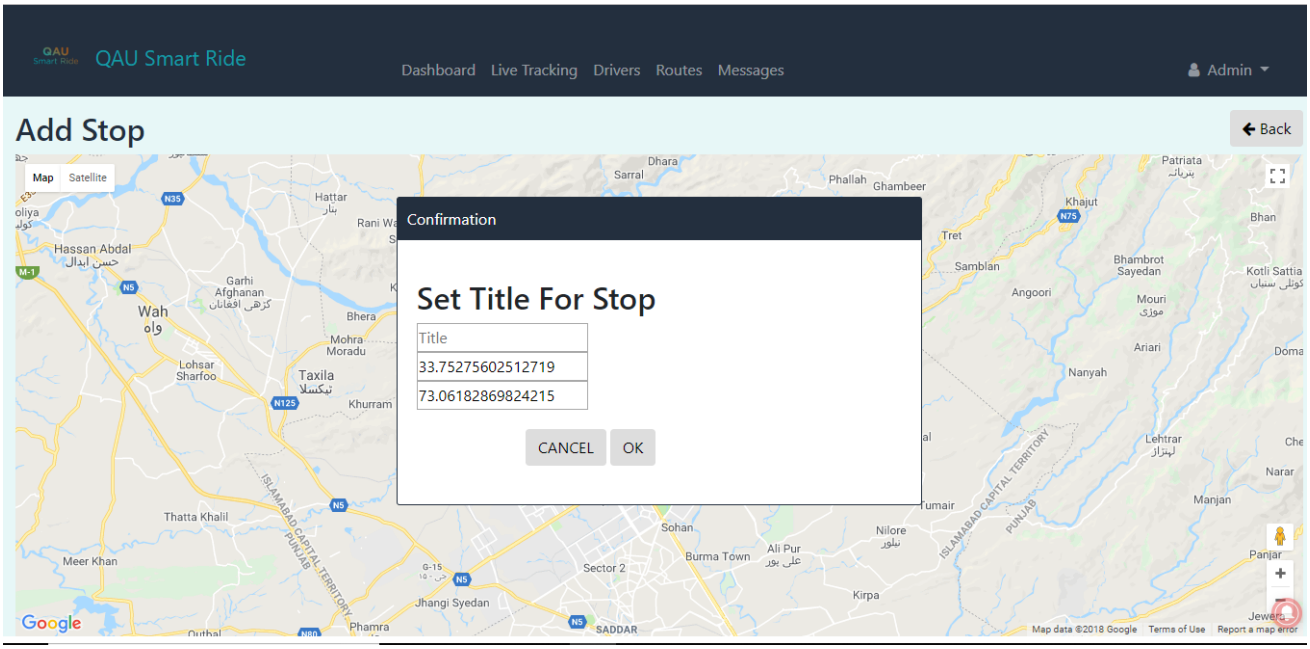


Figure 5. 16 add stop screen

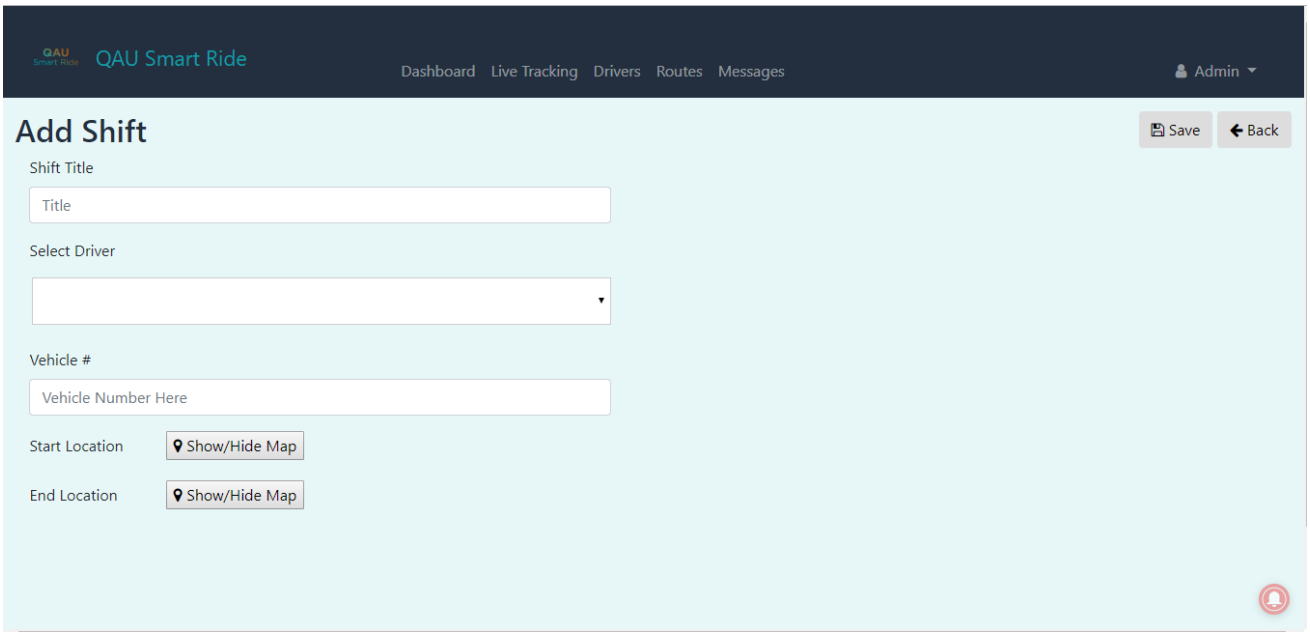


Figure 5. 17 add route screen

CHAPTER 6
CONCLUSION AND FUTURE ENHANCEMENTS

6.1 INTRODUCTION

This document describes the project conclusions and future enhancements -what type of new features can be added with time.

6.1.1 Summary

This project allows organization to track and monitor buses in an organized way. This application provides functionalities to driver, students and employees to perform their roles in a better and efficient manner.

6.1.2 Conclusions

- We are now able to track buses to get precise location.
- We are able to view routes and stops of any route.
- We are able to check idle and running buses.
- We are able to keep track routes and drivers.
- We are able to allow students to rate drivers on the basis of their driving skills.
- We are able to keep record of over-speeding alerts generated against each driver.
- We are able to receive geo-fencing alert when bus enters the specified proximity.
- We are able to evaluate drivers' performance on the basis of students' rating and over-speeding alerts.

6.1.3 Future Enhancements

In future application can be enhanced by:

- Drivers' performance will be calculated monthly and system will classify performance as EXCELLENT, AVERAGE or POOR.
- Complaint module can also be added in which passengers can file complain against any mis-management.

REFERENCES

[1] C. Larman, Applying UML and Patterns 2nd edition.

[2] “Class Diagram,” Google, [Online]. Available: www.tutorialspoint.com/uml/uml_class_diagram.htm. [Accessed 24 12 2017]

[3] “MongoDB,” Google, [Online]. Available docs.mongodb.com/manual/core/document. [Accessed 4 1 2018]

[4] “Google Maps,” Google, [Online]. Available: developers.google.com/maps/documentation. [Accessed 18 2 2018]

[5] “Node JS,” Google, [Online]. Available: nodejs.org/en/docs. [Accessed 3 5 2018]

[6] “OneSignal,” Google, [Online]. Available: documentation.onesignal.com/docs. [Accessed 16 5 2018]

[7] “Geofencing,” Google, [Online]. Available: developers.google.com/location-context/geofencing/. [Accessed 9 6 2018]