Interim Report

Level 2

Smart Notification System for Bus Management

HACK ELITE

PIGERA A.C.J. (Leader)	204159E
BASNAYAKA C.S.L.	204019C
SILVA W.H.I.	204202G
KAVINDI J.A.D.A.	204100R
ADHIKARI A.N.K.	204002T

Faculty of Information Technology

University of Moratuwa

2022

Interim Report

Level 2

Smart Notification System for Bus Management

HACK ELITE

PIGERA A.C.J. (Leader)	204159E
BASNAYAKA C.S.L.	204019C
SILVA W.H.I.	204202G
KAVINDI J.A.D.A.	204100R
ADHIKARI A.N.K.	204002T

…….

Supervisor

Mr. Upulanka Premasiri

(Department of Information Technology)

…….

Supervisor

Ms. Sashika Kumarasinghe

(Department of Information Technology)

Faculty of Information Technology

University of Moratuwa

2022

Abstract

The smart bus management system is based on monitoring the bus inside and outside the bus stand and notifying the higher officials about the unwanted delay of a bus at a particular place, as well as notifying the passengers about the arrival time of the bus. Considering those requirements, we decided to provide all the facilities in one system, which has web applications and mobile applications.

In our system, a Google map (GMap) is used to locate the bus. When the bus starts its turn, the GPS on the particular bus driver's phone is activated. The database is then updated with the bus's location at regular intervals. If there is a delay based on the breakdown of the bus or any other reason, it will notify the passengers with the new arrival time of the bus through a quick notification. They can get their turns within a certain day after the bus enters the bus stand. A time is fixed for the bus to be at a particular bus halt. It is stored in the database. If a bus occupies that bus halt for longer than the allotted time, a notification is given to the driver and the higher officials.

We organized the project into five core modules: login, passenger view, driver view, depo administrator view, and system administrator view, to facilitate team participation in various elements of software development. To provide better applications, we analyze our passenger requirements and design the UML and EER diagrams before implementation. And all the team members have started to design the frontend parts of their tasks so far.

For developing the system, we choose technologies after a better research program. ReactJS and Bootstrap is used for the front end of the web application, and Flutter is used for the front end of the mobile application. For backend development NodeJS is used. MongoDB is used for the database design, and cloud provider AWS is used for our system. The web application is user-friendly, efficient, and time-saving.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Transport is an important sector of the economy. It can be considered as an important part associated with day-to-day activities of people. If we take public bus transportation plays a key role in the topic of transport. Maintaining the efficiency of bus transportation has a greater impact in making people's lives easier, especially employers who travel by bus daily to their workplaces.

But in the Sri Lankan context there are many problems with bus transport and delays are always present. Here we are introducing a technology-based solution for it. Once a customer installs the app into their smartphones, they can search for the bus that they want to reach their destination, to know whether buses are available at a particular time, how much time it will take the bus to come to the bus halt they are willing to get in and there is a smart notification system to aware them the about location of the bus they searched for. Also, the proposed system provides real-time information about bus location, route, speed, delays and much more. We hope to use the latest technology along with GPS technology for tracking the locations and we also hope to develop a web app. So, the ultimate objective is to make the lives of passengers easier and convenient when it comes to transportation using technological advancements. In this project, we hope to develop,

- Notification for staff
- Notification for passengers
- Oversee location, routes, runs, stops, actual trip time using real-time data
- Docker Kubernetes/Scalability using containerization
- Microservice Architecture

1.2 Problem in brief

Although the public bus transport service is the main public transport service in the country, passengers have to face many problems while using this system. Among them, the main reason is that the buses do not run according to a fixed schedule. Due to this, the passengers face many difficulties. Many passengers use bus transportation to fulfil their daily needs and especially people use this service to go to work. Buses running late and not running according to a proper schedule have been a frequent practice since the past and these days the oil crisis that Sri Lanka is facing has exacerbated this problem. Due to the non-arrival of buses at the respective bus stand on time, passengers have to wait on the road for no reason. They waste their precious time and are unable to complete their daily activities on time. Some buses run at high speed between some areas. Due to this reason, the probability of road accidents is extremely high. As a solution to this problem, we can minimize the effects of these accidents by monitoring the route of the bus and constantly providing relevant information to the staff related to the bus transportation system. This is the main problem we considered for our project.

1.3 Background Motivation

The world has changed in ways that are reflected by modern technologies. It is no secret that modern technologies have provided a completely innovative experience that improves the convenience of our daily lives. In such a world, businesses must successfully use technology to advance their processes and simultaneously improve people's lives. Meanwhile, when we talk about the transportation system of our country, it is in a sad condition. For example, in a bus management system, there are many uncertainties that affect both the vehicles and the public. Traffic jams, unexpected delays, randomness in passenger demand can occur as the day progresses.

Therefore, solutions are needed to improve the travel experience of bus users. One of the current trends is developing solutions to predict bus arrival times. Modern technology has shown the real benefit of this trend in developed countries, where the hardware infrastructure in cities enables citizens to know the status of public buses and therefore

provides a prediction of arrival time. Citizens of these developed countries use these mobility systems to manage their trips in cities. However, this is not the general situation in developing countries.

So, we encouraged to make a smart notification system for bus management. In this project, our focus is on implementing a real-time bus monitoring system inside and outside bus stands by installing GPS devices in buses.

1.4 Aim and Objectives

1.4.1 Aim

The aim of this project is to develop a web app and a mobile app for a smart notification system for bus management using modern technologies to meet the above requirements.

1.4.2 Objectives

We will be meeting the following objectives through our proposed software solution.

- To manage the passenger's time by tracking the location of the relevant bus.
- To facilitate staff taking appropriate action in case of bus delays.
- To develop a mobile app linked to the system for passenger convenience.

1.5 Summery

Through Chapter 1 of this report, we provide a basic introduction to the smart notification system for bus management that we hope to develop. Similar projects currently in the market are discussed in Chapter 2. In Chapter 3, a basic understanding of the technologies used in the smart notification system for bus management is given. Our approach in developing this application is discussed in Chapter 4. Chapter 5 provides a clear understanding of analysis and design. Implementation is discussed under Chapter 6 and Chapter 7, Chapter 8, Chapter 9 consist of discussion, references, and appendices.

Chapter 2

Literature survey

2.1 Introduction

In this chapter, we focus on documenting the different approaches to addressing the same underlying problem, which we have selected to offer a solution for. In the modern world, most of the technological processes are intertwined with the transportation industry. This can also be applied to management in the public transportation industry. Based on our literature survey, here we compare several other existing approaches similar to our desired web application.

2.2 Existing Software

The software listed below can be identified as an important solution that directly focuses on the above problem.

1. Google Maps
2. Trackmycar.lk

2.2.1 Google Maps

Google Maps is a web mapping platform and consumer application offered by Google. It offers satellite imagery, aerial photography, street maps, 360° interactive panoramic views of streets (Street View), real-time traffic conditions, and route planning for traveling by foot, car, bike, air (in beta) and public transportation.

Figure 2.2.1. 1: GoogleMaps

When considering this system there is no way to track a particular bus that passenger wants. Google Maps can only give a suitable route and guide the passenger for their destination. And also, this system can't track the delayments of the transport services. It also the main drawbacks of this system. These are the limitation of this system.

2.2.2 Trackmycar.lk

Track My Car vehicle tracking solutions combine sophisticated GPS tracking technology with flexible, advanced mapping and reporting software. A GPS-enabled Vehicle Tracking Device is installed on each vehicle to collect and transmit tracking data via a cellular or satellite network, whichever works best for users'' operations. The device then delivers the data to Track My Car hosted application, which user can access through the web at any time. You receive real-time vehicle tracking updates, including location, direction, speed,

idle time, start/stop and more, allowing user to manage a tighter schedule and more efficient fleet. This software has mobile app also.

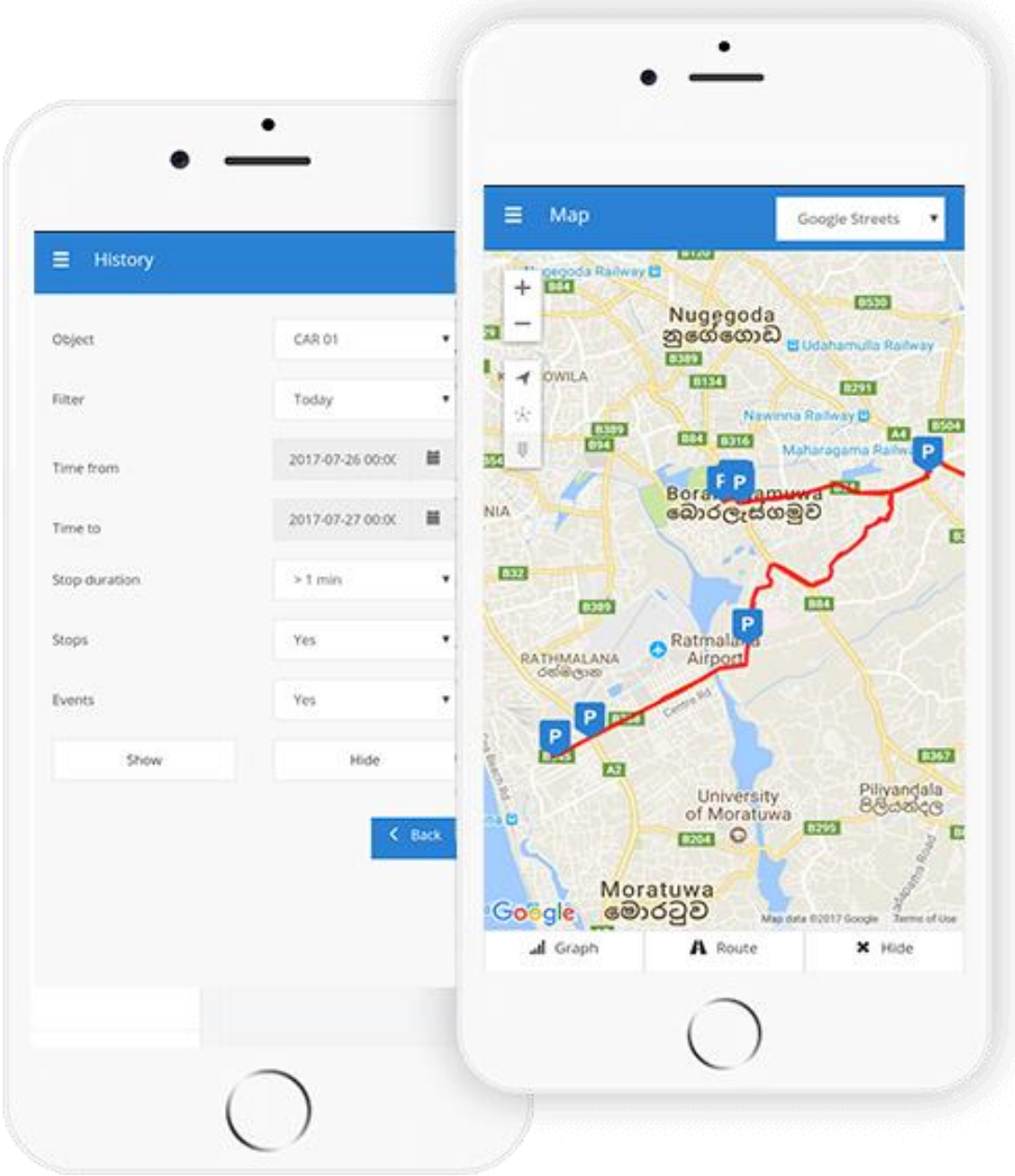


Figure 2.2.2. 1: TrackMyCar Mobile Interface

Figure 2.2.2. 2: TrackMyCar Web Interface

2.3 Summary

As previously mentioned, there are existing software products that are similar to our web application. They may have some strengths (features) over our system, but they do not have all the features that we possess. In our web application, we are hoping to provide many of the same facilities as existing software, but additionally, we have added a few more interesting features. When we are considering our system, we can illustrate certain specific features.

- Passengers can choose a particular bus.
- Sent notifications to passengers about bus breakdown delays.
- Sent notifications to higher officials about unwanted delays.
- Can create chat among the driver and bus administrator about replacement of new bus in a situation like bus breakdown.

Therefore, our system can be considered a unique and user-friendly web application with novelty features for public bus management.

Chapter 3

Technologies

3.1 Introduction

The most suitable and compatible software technologies must be selected after considering the resources at hand and the potential for implementing the desired system.

We will go over the technologies we used to create the suggested software system in this chapter, along with the reasons we opted for those particular versions over others. We can categorize the technologies under the following sections.

1. Frontend Technology – Flutter, ReactJS,Bootstrap

2. Backend Technology – Spring Boot

3. Database Technology- Mongo dB

4. Cloud Provider- Amazon web service

5. Notification Technology- Firebase cloud messaging

3.2 Frontend Technology

The frontend refers to the system's front face. This indicates that this is the point at which end users engage with the system. The system needs to provide a better user experience from the viewpoint of the end user. Therefore, we have selected the following frameworks since we wish to have a speedy development technique.

3.2.1 ReactJS

React is a declarative, effective, and adaptable JavaScript user interface library. 'V' stands for the view in MVC. ReactJS is an open-source front end library that is only in charge of the application's view layer. Facebook looks after it.

The declarative approach used by React, which seeks to be both effective and versatile, makes it simpler to reason about your application. For each state in your application, it creates straightforward views, and React will quickly update and present the ideal component whenever your data changes. Your code becomes more dependable and troubleshoot able when using the declarative view.

Each component in a React application is in charge of rendering a discrete chunk of reusable HTML. The ability to nest components within other components enables the construction of sophisticated applications from simple building blocks. A component may also keep track of its internal state; for instance, a Tablaist component may keep a variable for the open tab in memory.

3.2.2 Bootstrap

Due to its simplicity of usage and availability of templates for icons, typefaces, buttons, and other components, Bootstrap is the best open-source CSS framework currently accessible for frontend software development. Bootstrap makes it easier to create and develop interfaces for our web application.

3.2.3 Flutter

Google created the cross-platform Flutter mobile programming framework. This framework is built using the Dart programming language [1]. It is applied to the creation of an iOS and Android mobile application. Nowadays, practically everyone in the globe owns a smartphone running either Android or iOS [2]. As developers, we typically have to create apps for the aforementioned two platforms. Because of this, we must maintain two separate code bases for the same reasoning. In terms of development time and the dependability of the finished product, this will be a significant burden for developers and end users as time goes on [3]. This is why we require a system to retain a single code base and compile for the appropriate platform. The best cross-platform mobile development framework is Flutter.

3.3 Backend Technology

Backend technologies serve as the foundation for each and every server-side program in our system. Selecting the appropriate backend framework for the applicable needs is crucial. Therefore, in order to handle the pertinent use cases for the various use cases, we choose to adopt a microservice architecture. We will go over the rationale for and specifics of implementing microservice architecture for the backend.

3.4 Database Technology

One of the key components of the system is the database. A database is often created so that users can access and store data. The system relies heavily on the database. Due to the database's ability to record all pertinent system-related information.

3.4.1 Mongo dB

A document-oriented, cross-platform database called MongoDB offers excellent performance, high availability, and simple scalability. MongoDB utilizes the collection and document concepts. Database. A database is a real-world home for collections. The file system receives a unique collection of files for each database.

Although MongoDB is made to make data accessible and rarely needs joins or transactions, it is more than capable of handling sophisticated queries. With just a few lines of declarative code, the MongoDB Query API enables you to run complicated analytics pipelines and deep document queries.

3.5 Cloud provider

Utilizing a variety of services, such as servers, platforms, storage for software development, and software, through the internet is known as "cloud computing." Many

businesses currently use cloud-based technologies. The following factors have a big impact on it.

1.Deployment

Resources are deployed inside and inside the IT infrastructure of an organisation in an on-premises environment. A business is in charge of managing the solution and all of the processes connected to it. Different cloud computing models can access those resources and utilize them as much as they'd like at any particular time.

2.Cost

Businesses that install software on their own premises are liable for the ongoing costs of the server hardware, power use, and space. Businesses that choose to use a cloud computing model only pay for the resources they really use; there are no maintenance or upkeep charges, and the cost changes according to how much is used.

3.Security

Businesses that handle very sensitive data. Security is a major concern for many companies, despite the cloud's potential, so even with some of its limitations and cost, an on-premises system makes more sense. The biggest obstacle to cloud computing is still security worries. Cloud security breaches have been widely reported, and this has worried IT organizations all around the world. Security hazards range from a loss of intellectual property to employee personal information, such as login credentials.

Cloud computing has various benefits when compared to on-premises computing. In order to conclude that the cloud is among these two the better solution**.**

3.5.1 Amazon web service (AWS)

AWS is made to enable suppliers, ISVs, and application providers to swiftly and securely host your apps, whether they are SaaS-based or not. To access AWS's application hosting platform, use the AWS Management Console or well-documented web services APIs.

Over 140 AWS services are offered, including computation, storage, databases, analytics, networking, mobile, and developer tools, through Amazon Web Services. This gives

entrepreneurs, startups, small and medium-sized companies, and clients in the public sector access to the fundamentals they need to quickly adapt to changing business requirements.

3.6 Notification Technology

3.6.1 Firebase cloud messaging

A cross-platform messaging service called Firebase Cloud Messaging (FCM) enables you to send messages dependably and without spending any money. You can alert a client app that new email or other data is available for syncing by using FCM. Notification messages can be sent to encourage user re-engagement and retention. A message can deliver a payload of up to 4 KB to a client program for uses like instant messaging. We are able to send and receive messages and notifications on iOS, Android, and the web without spending any money thanks to Firebase Cloud Messaging (FCM), which offers a dependable and battery-efficient connection between your server and devices.

3.5 Summary

This chapter offers a summary of every technology we used to create our suggested solution, along with an explanation of our decision-making process. After carefully examining the other available technologies, several technologies were selected.

Chapter 4

Proposed solution

4.1 Introduction

Transport is an important sector of the economy. It can be considered as an important part associated with day-to-day activities of people. If we take public bus transportation, it plays a major role in the topic of transportation. Maintaining efficiency in bus transportation has a greater impact in making people's lives easier, especially employees who travel by bus daily to their workplaces. But in the Sri Lankan context, there are a lot of problems related to bus transportation and delays are apparent. We are introducing a technology-based solution to overcome these transportation delays. Once a customer installs the app into their smartphones, they can search for the bus that they want to reach the destination. To understand whether the buses are available at a particular time, how much time it will take for the bus to come to the bus halt, there is a smart notification system to notify them about the whereabouts of the bus they searched for. Nevertheless, the proposed system provides real time information about various aspects of the buses like the location, the route, the speed, delays and much more. We hope to use the latest technologies along with GPS technology for tracking the locations and we thereby hope to develop a web app as well. The ultimate objective is making passenger's life easier and more convenient regarding transportation using technological advancements.

In this project, we hope to develop,

- Notifications for staff
- Notifications for passengers
- Oversee location, routes, runs, stops, actual trip time using real-time data
- Containerization of the modules for scalability of the system with the help of Docker and Kubernetes
- Microservices Architecture

4.2 Software Process Model

We identified Agile scrum model as the best model suitable for our project after analyzing the nature of it. The Agile methodology is used to manage a project by breaking it up into several phases. Continuous improvement at every stage and ongoing collaboration with stakeholders are required in this methodology. Once the project work begins, teams cycle through a process of planning, executing, and evaluating.

After having several meetings with the client company mentors, we got an idea about the requirements. So, we are clear about the basic requirements of the system, and it might slightly change while the development of the project is in progress according to the ultimate goal of the project. Also as similar kind of projects are very less this is a new idea especially considering the Sri Lankan context to minimize the problems in transportation. Those are some reasons for us to select this model.

We divided the whole work in to small parts and created user stories and assigned them to several sprints. We schedule meetings with the company mentors in order to regularly get their feedback and to make sure that we are in the correct path to fulfil all the requirements and reach the desired goal. By presenting the finished parts at the end of each sprint we can assure it and do required changes.

4.3 Users, input, output of the system

user/role		
Passenger	Activities	• Login using username and password • Search starting location and the required time duration • Select required bus • Enable the notification button • Select the time and distance that the passenger wants to notify him
	Input	• Username/email • Password
	Output	• View bus location • Chat interface • Receive the bus arrival notification • Receive the bus breakdown delay notification
Driver	Activities	• Login using username and password • Entering attendance information daily whether present or not • View daily bus schedule and their turns • Report any issues about delays or breakdowns through chat app • Inform passengers about time delays • Request replacement for bus when needed
	Input	• Username/email • Password • Attendence • Delays
	Output	• Chat interface • Messages through chat app • Daily bus schedule

Depo admin	Activities	• Login using username and password. • add routes of the buses. • Check the driver attendance sheet. • Divide routes among the present drivers. • Response through the chat option to handle the replacement.
	Input	• Username/Email • Password • Routes
	Output	• Chat interface • View the driver attendance sheet • Delay notification
System administrator	Activities	• Login using username and password. • Add the new comers to the system like drivers & depo administrators. • Manage their profiles. • Manage the newly added bus routes. • Update the newly added bus schedules. • Delete some details permanently from system. • Removing someone who has been assigned to any other work or retire form the job. • Helps to the passengers if they need some clarification about the logging process. • Generate the reports monthly by using some documents. • Give the positive feedback through the chat option for the passengers' requirements. • Track the bus live location.
	Input	• Username • Password • New comers • All profiles
	Output	• View all the profiles • View bus location • Chat interface • Messages through chat app • Daily notification • Monthly reports

Table 4.3. 1: Users, input and output of the system

4.4 Process

In this section, we discuss the process of our system. Passenger, driver, depo administrator and system administrator can sign up and log in to the system providing the username, email, and password. There are separate interfaces for passenger, driver and depo administrator. Different users can choose their respective interface during login. Through the passenger interface, map is displayed all the time.

Passengers are divided as unregistered passengers and registered passengers. Unregistered passengers can only access limited features; searching buses by entering the passenger's starting point and time duration and viewing all the buses passing the given starting point. If a passenger wants to select the required bus and receive the notification, they have to register. After registered, user has to select the required bus and enable the notification button. Then they can view the current location of the bus and the reaching time to the starting point. After enabling the notification button, they have to select the time and distance that the passenger wants to notify him. Once the bus reaches the given distance, the passenger receives the bus arrival notification. And also, if any delay occurs due to bus breakdown or accident, the passenger will be notified with new arrival time (delay notification). Passengers can view the real time location of the selected bus through the map.

Routes are added to the system by the depo administrator of each depo. After the driver log into the system, driver has to mark the attendance by enabling relevant button. If driver present for work, he has to enable the "IN" button, otherwise enable "ABSENT" button. After all the drivers have entered their attendance, depo administrator can view the attendance information and divide routes among the present drivers. Then daily turn schedule is updated with the relevant turns for drivers by depo administrator. The schedule includes driver's name, bus number, route and time. Then driver can view the schedule and their specific turns. When driver starts his turn, he has to click the "START ROUTE" button. Tracking the live location of the bus has started since there.

If the bus is traced at the same place (bus stop) for a long time, send warning notification about unnecessary delay to the depo administrator and bus driver. Then depo administrator should check and get an appropriate action about that delay. If breakdown or accident occur, driver has to report the issue by enable the “DELAY” button. Then driver has to select the possible time delay. After that “DELAY” notification will be sent to the passenger. If the repair takes a longer time, driver has to go to chat option and send an emergency message to the depo administrator about need for a new bus replacement. When depo administrator received the message, he response through the chat option to handle the replacement. Also a notification about the breakdown has to be sent to the passenger.

If any driver or depo admin resigns, system administrator has to manage the system by deleting their profiles. When new drivers register to the system, the system administrator has to insert the driver’s name and the license number of the bus assigned to him. And, he has to enter the information about extra buses. System administrator will generate the reports about drivers’ attendance, responsiveness of drivers to the warning messages etc. at the end of each month and send them to the depo administrator. Also, he handles passengers’ feedback and complaints, issues in login process.

4.5 Our Approach

In this chapter, we will discuss the way we designed our system. As we discussed earlier in Chapter 3 we have used several backend technologies to fulfill the use cases. So how to use multiple backend frameworks? The answer is the microservice architecture. We initially think of the system as a conventional monolithic application when designing the backend. The database was created as a single application. The best technology for the needs was then determined. Then we made the decision to continue using Spring Boot Framework. However, we discovered that some requirements do not match the chosen framework. We discovered that some functions can be carried out on their own. Additionally, we came to the realization that designing the backend with a monolithic architecture is not the ideal choice when the client requirements get complex. since this architecture has so many drawbacks. Consequently, we chose to remain with the microservice architecture.

An architectural design known as microservices treats an application as a group of interconnected services. By utilizing this design, we may overcome the limitation mentioned above that arises from a monolithic environment. We construct separate services for each of the services since some of the solutions might be combined and function independently. Then, from our previous monolithic database, we extract the relevant database structure for each service. Because each microservice's database should be viewed as private. Our monolithic architecture, which applies the best back-end framework to the different functionalities, had a problem. However, using a microservice design, we may quickly implement the best framework for the specific service. The system's microservices are illustrated in the following architecture diagram. Finally, we'll discuss each service in turn.

- A bus tracking service that collects data from the bus tracking system and stores it in the database.
- A notification service that sends push notifications to users when a bus is approaching their stop or if there are any issues with the bus.

- A customer service chatbot that handles tasks such as answering common questions, resolving customer issues, and providing information about bus schedules and routes.

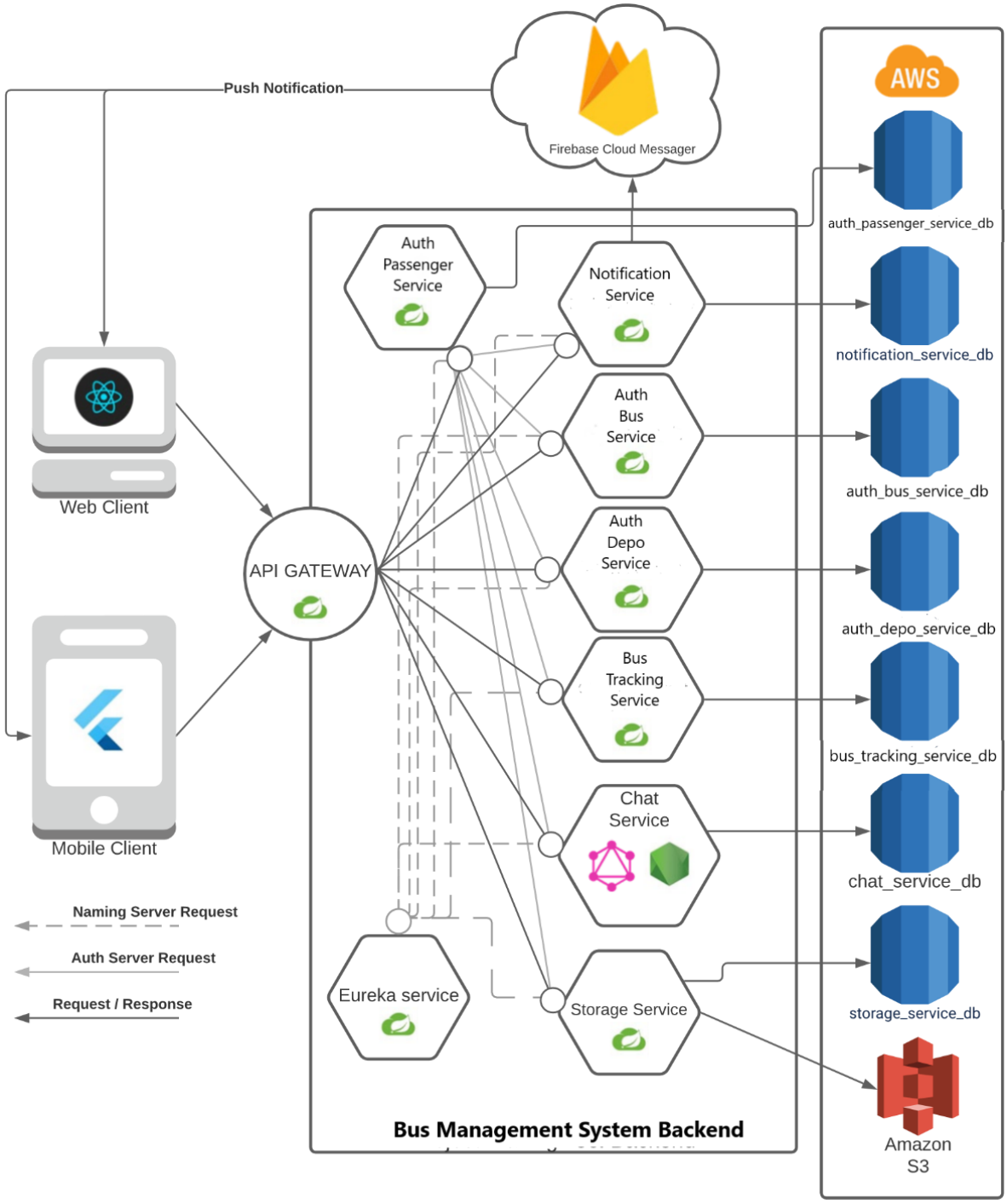


Figure 4.5. 1: Architecture Diagram

4.6 Project Management Plan

4.6.1 Trello Project Management Board

We adhere to a project management strategy in order to achieve a successful product. It is an official, approved document that spells out how our project is carried out, monitored, and managed. We are operating in accordance with our management strategy, and to improve management, we used the Trello project management tool. We use Trello to assign tasks among the group members, establish deadlines, share documents etc. It can be taken as worldwide used project management tool used by several IT professionals.

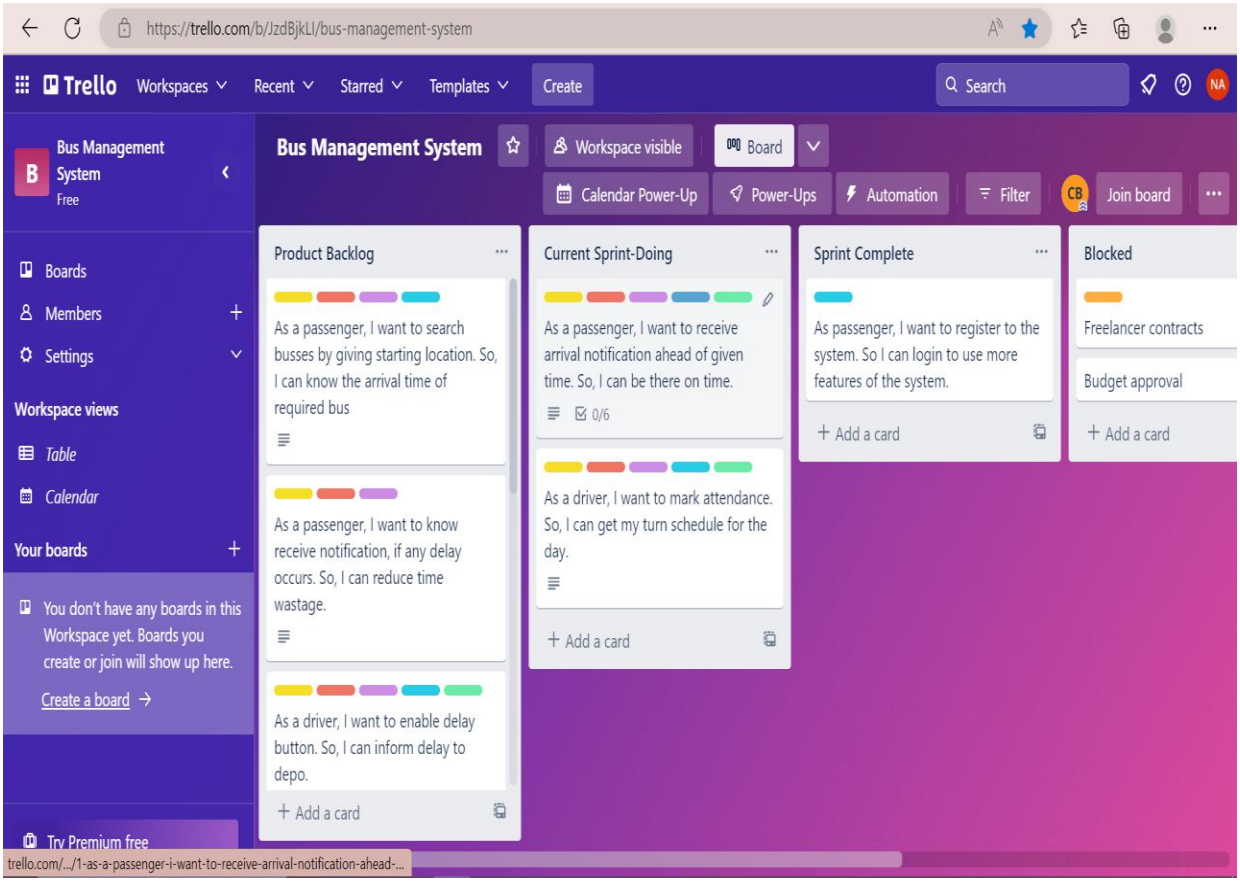


Figure 4.6.1. 1: Trello project management board

4.6.2 Version Controlling - GitHub

Project management involves more than just assigning work to group members. A suitable version control system should be used for the project's product development. These days, many of these version control solutions are open source. For our project, we chose Git as the key technology for version control, and to do so, we created repositories on GitHub.

We established a private organization named "hackelite1" and each of our members has a GitHub account because GitHub is open source. We generated two major repositories for the front end of our web app and the back end of our software system throughout the coding and development stage in order to evaluate and save our scripts in a secure location. When creating the allotted component for ourselves, each of us forked repos to our personal accounts,

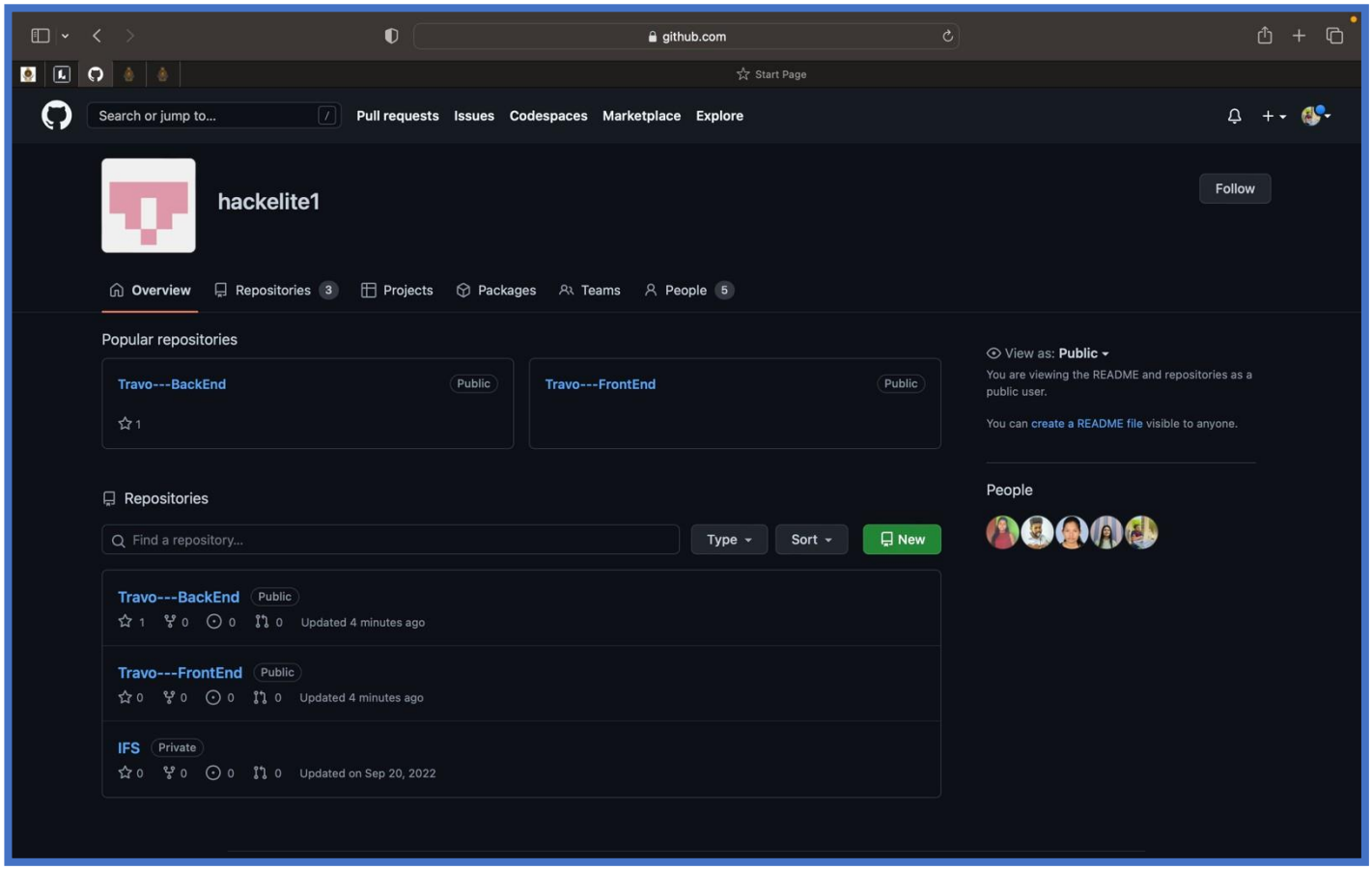


Figure 4.6.2. 1: GITHub

Chapter 5

Analysis and design

5.1 Introduction

The analysis of Smart Appointment System for Bus Management has carried out to understand the requirements, to understand the most suitable software process model, to create the most effective plan and to reduce risks that occur while developing the web application. We used various UML diagrams to clearly visualize the system. An Entity Relationship Diagram has been developed to visualize the database of the system.

5.2 System Analysis

Figure 5.2.1 is a simple system model diagram of the design software solution.

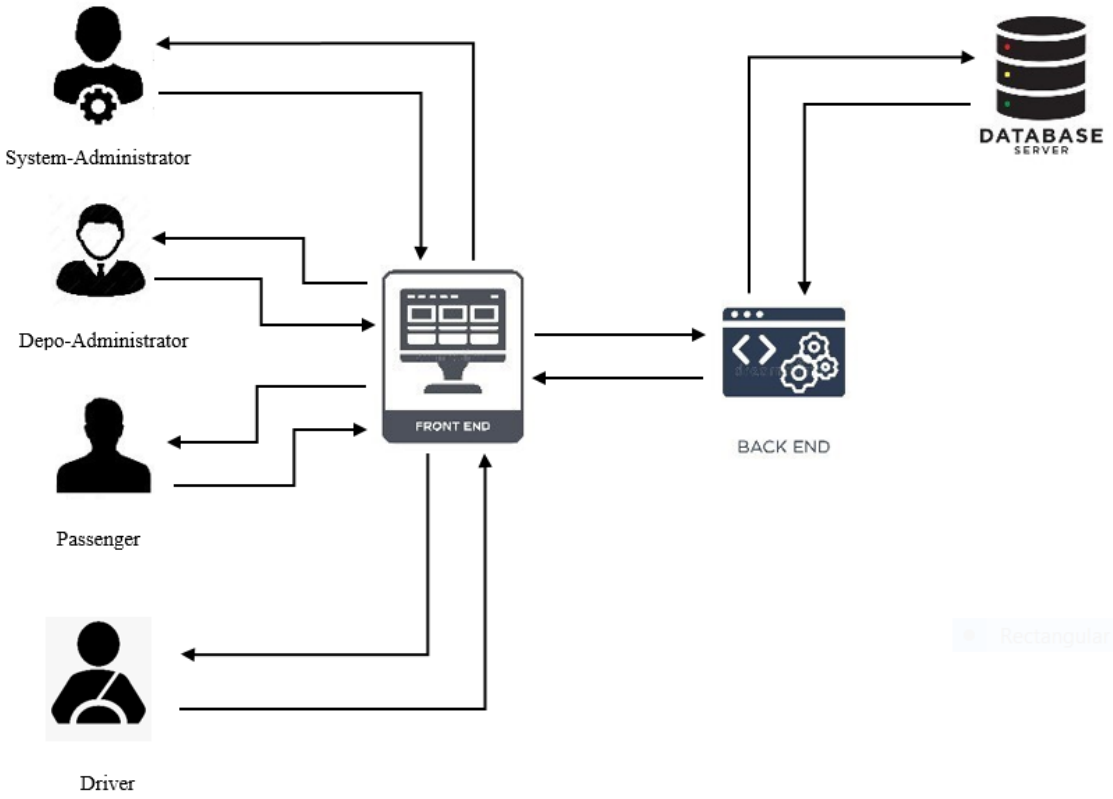


Figure 5.2. 1: Top level architecture

5.3 Modules

After analysis, we have split up our main system into five main modules as mentioned below.

5.3.1 Login module and chat app

In the sign-in, sign-up, and forgotten password processes for web applications, this module incorporates user authentication and verification. A user must use their email address as their username and a password they created themselves to get in to the system; alternatively, they can use their Google account. The user will receive a confirmation email after entering their email address, and the encrypted password will be saved in the database in registration process.

Chap option is there to communicate issues between drivers and depo administrators as well as to get customer complaints.

5.3.2 Driver interface

Bus drivers can mark attendance and after viewing their turn and scheduled time they can start journey by clicking start route button. Any issue occurring due to breakdown or accident also can be handled through the system.

5.3.3 Passenger interface

There are two types of passengers namely registered passengers and unregistered passengers. Only registered passengers can get notifications and view the live locations of the bus while all the passengers can search for buses at a specific given time duration.

5.3.4 Depo-admin interface

Main things that depo admin can do through his interface is that adding routes and dividing daily bus turns among present drivers after viewing attendance information of all the drivers.

5.3.5 System administration

Managing user profiles, deleting user profiles, generating reports about driver attendance and their responsiveness to warning notifications are handled by the system administrator. Also handling complaints and login issues of users, and adding bus information to the system are done by him.

5.4 Design

In this chapter, we will focus on the diagrams that we designed to visualize the functional and non-functional requirements. We have used Lucid Chart to draw the diagrams. There are several UML diagrams we have used,

1. Use case Diagram
2. Activity Diagram
3. Class Diagram
4. Sequence Diagram
5. EER Diagram

For a clear view, all the design diagrams of the project can be accessed via below link.

https://dms.uom.lk/s/9okKANqAkDiRzDY

5.4.1 Use case diagram

At its most basic level, a use case diagram is an illustration of a user's interaction with the system that demonstrates the relationship between the user and the many use cases in which the user is involved.

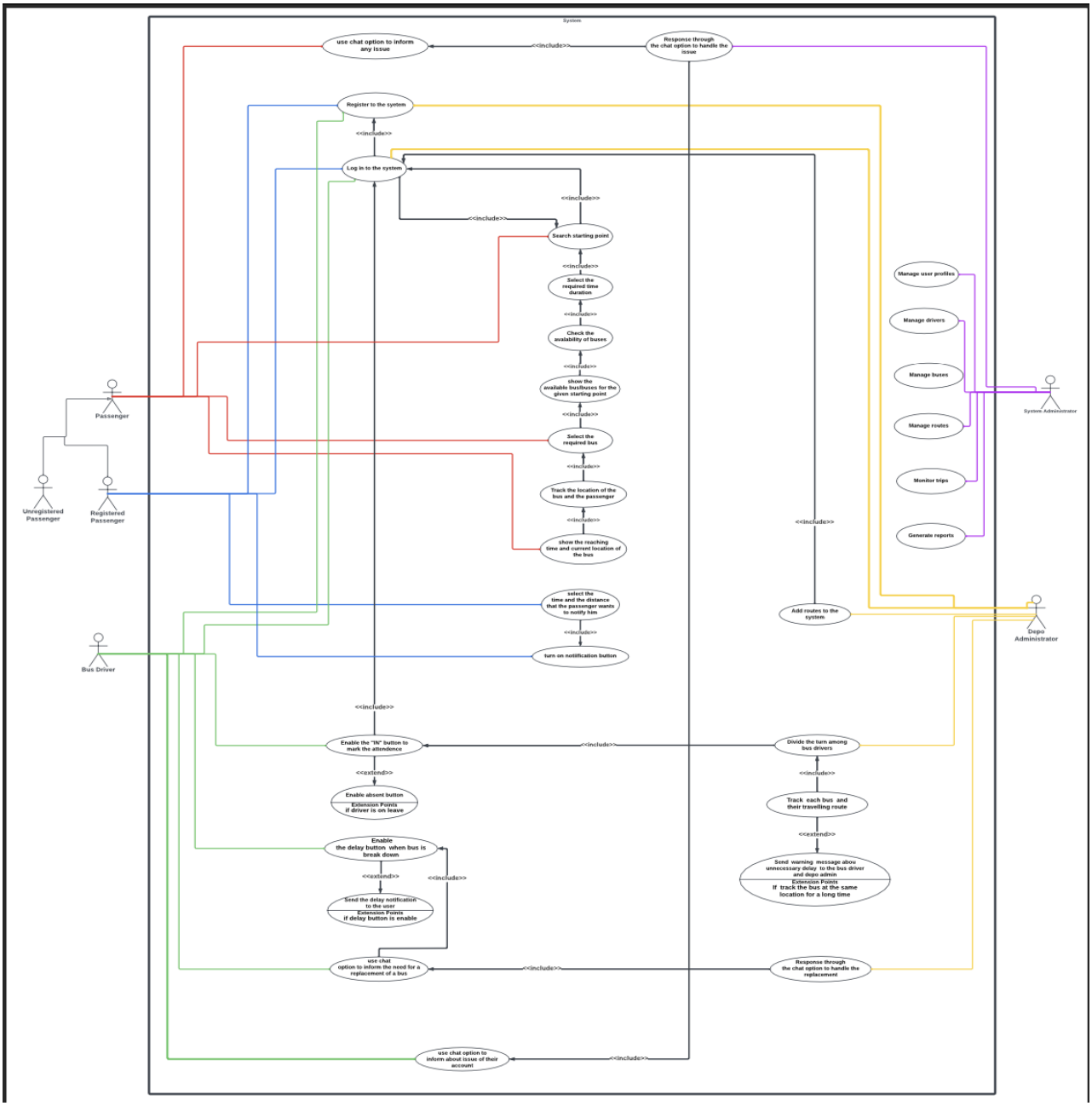


Figure 5.4.1. 1 Use case diagram

5.4.2 Activity Diagrams

Activity Diagram - Login and signup

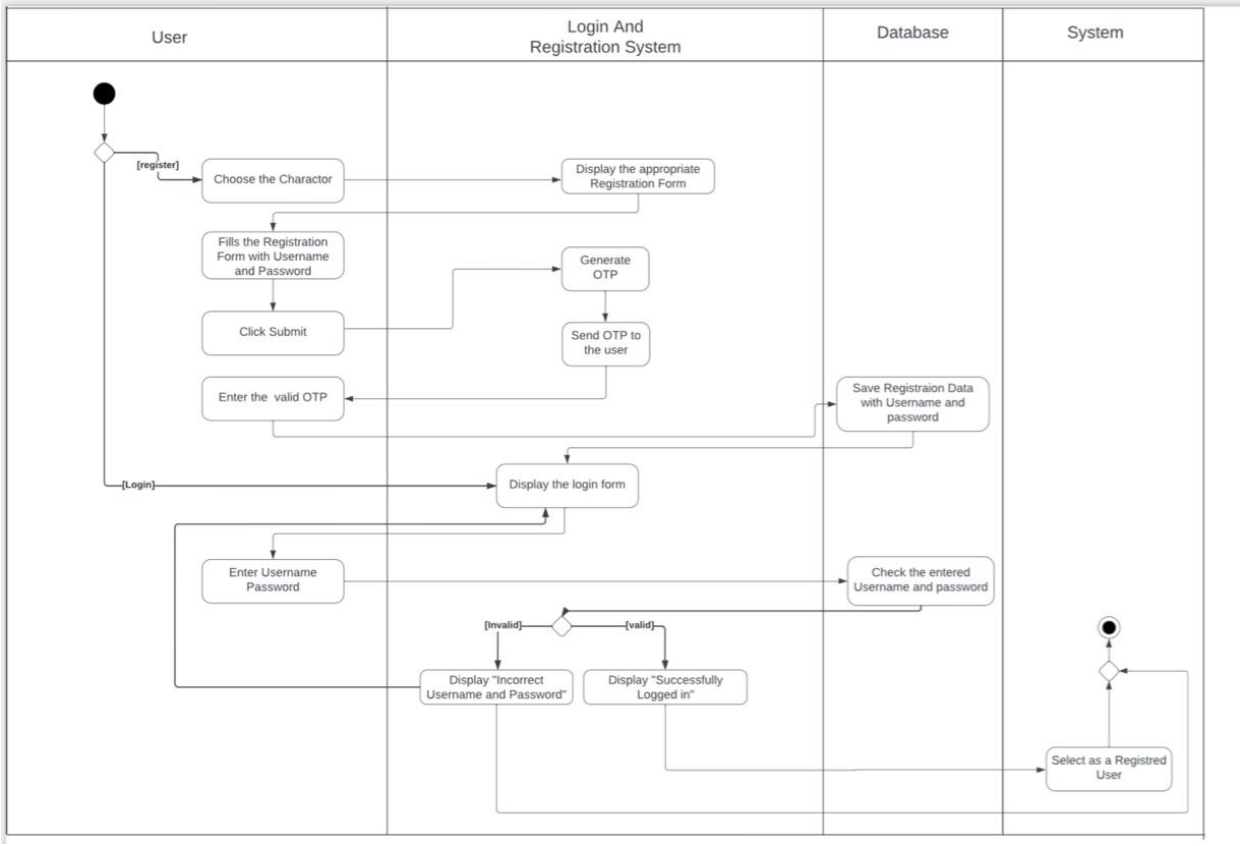


Figure 5.4.2. 1: Activity diagram for login and signup

Activity Diagram - Chat app

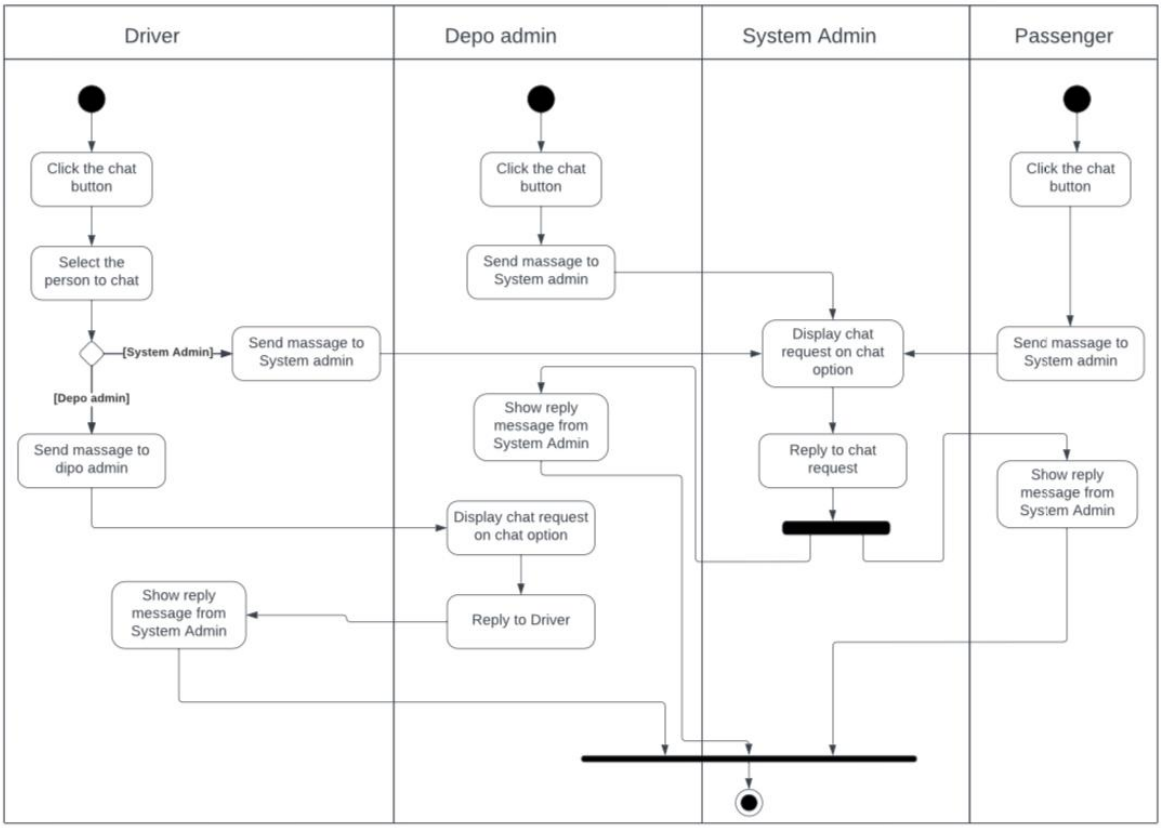


Figure 5.4.2. 2: Activity diagram for chat app

Activity Diagram - Searching buses

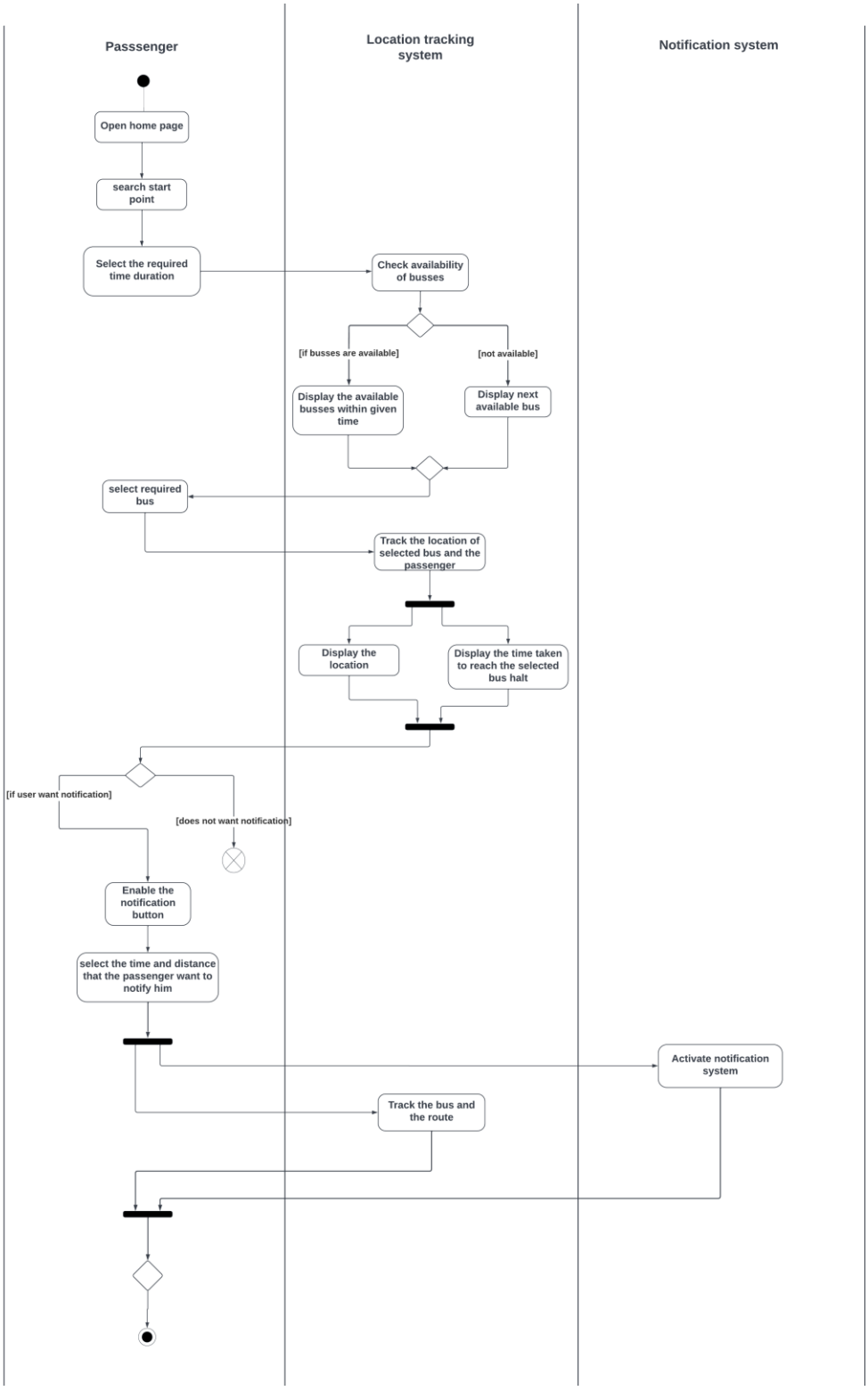


Figure 5.4.2. 3: Activity diagram for searching buses

Activity Diagram - Notification system

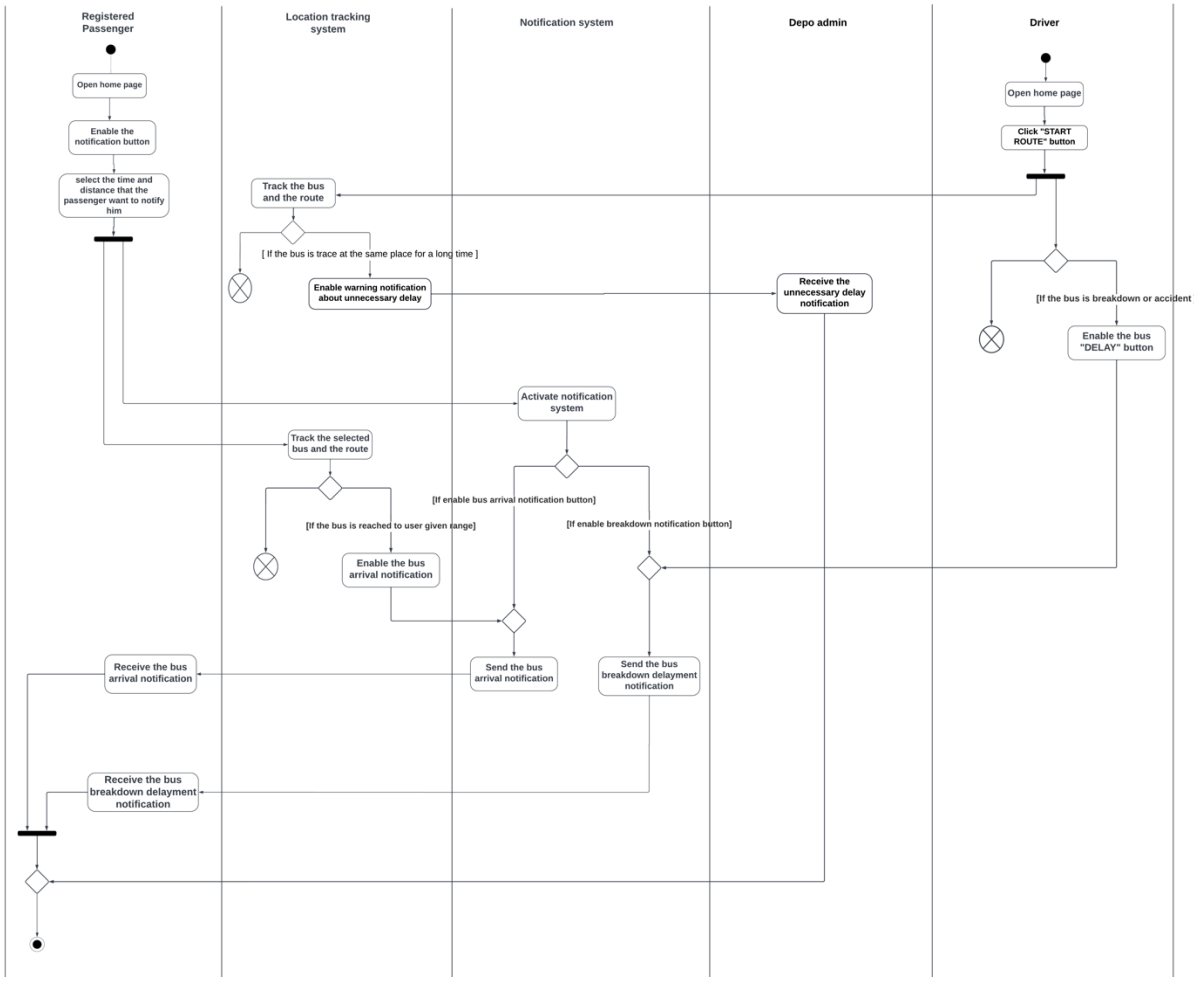


Figure 5.4.2. 4: Activity diagram for notification system

Activity Diagram - Handling breakdown or other issues

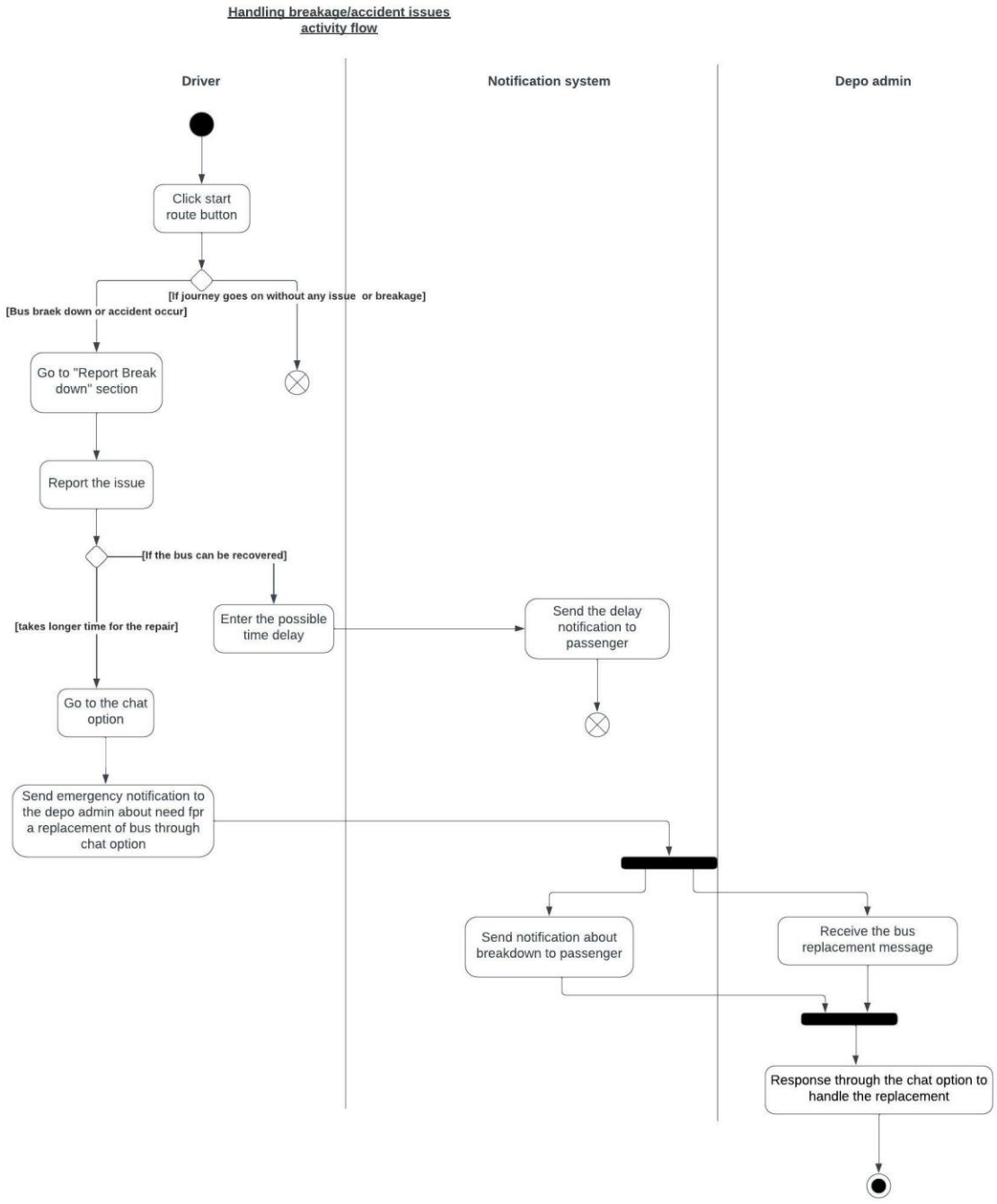


Figure 5.4.2. 5: Activity diagram for handling breakdown or other issues

Activity Diagram - Taking driver attendance

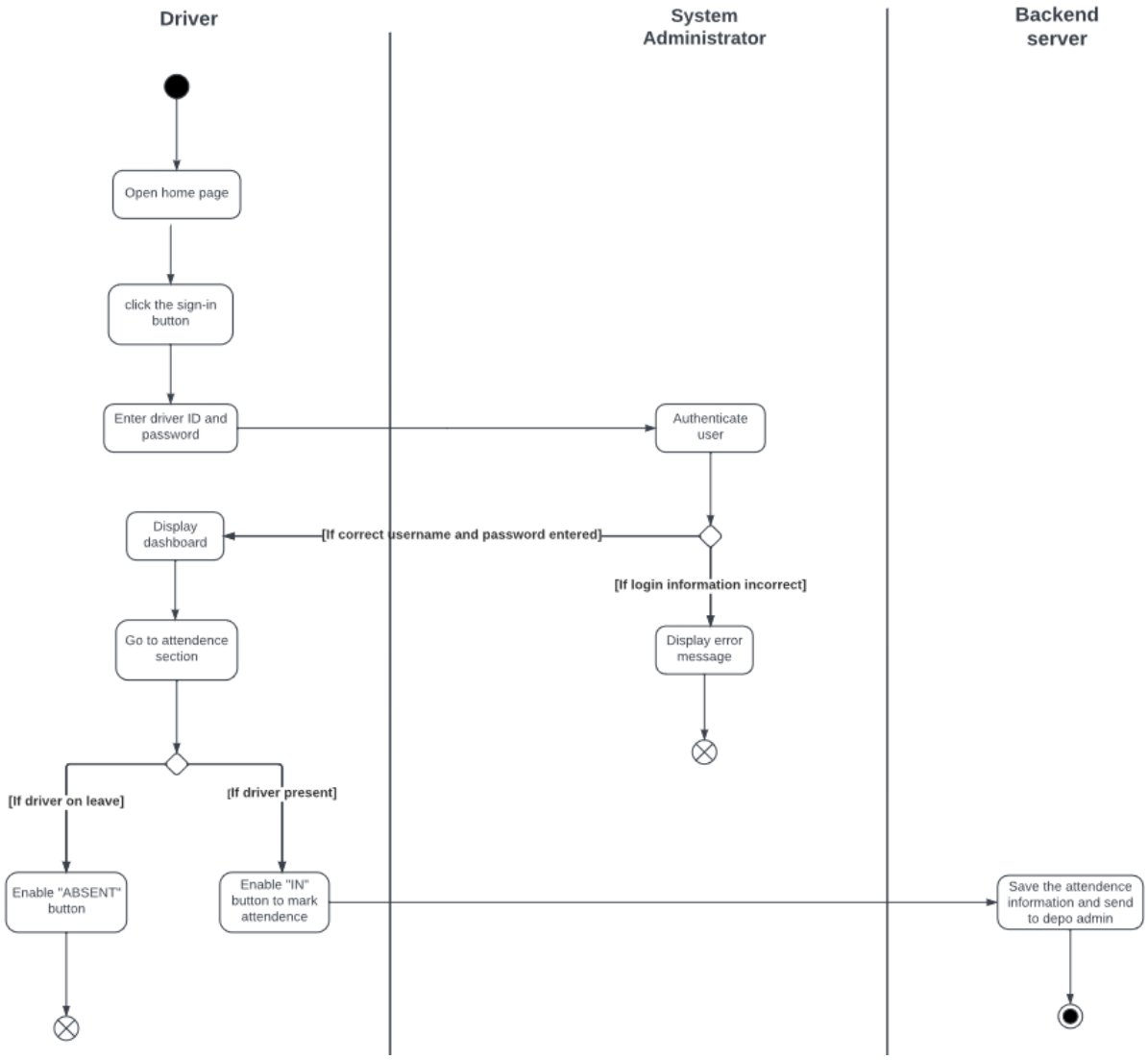


Figure 5.4.2. 6: Activity diagram for taking driver attendance

Activity Diagram - Bus route management

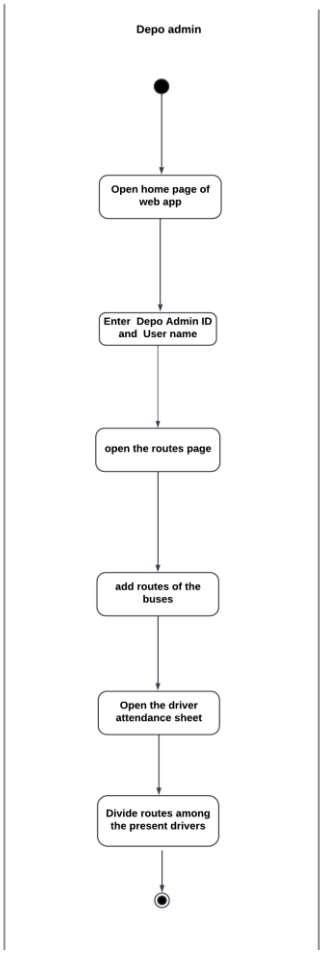


Figure 5.4.2. 7: Activity diagram for bus route management

Activity Diagram – Registration Data (System admin)

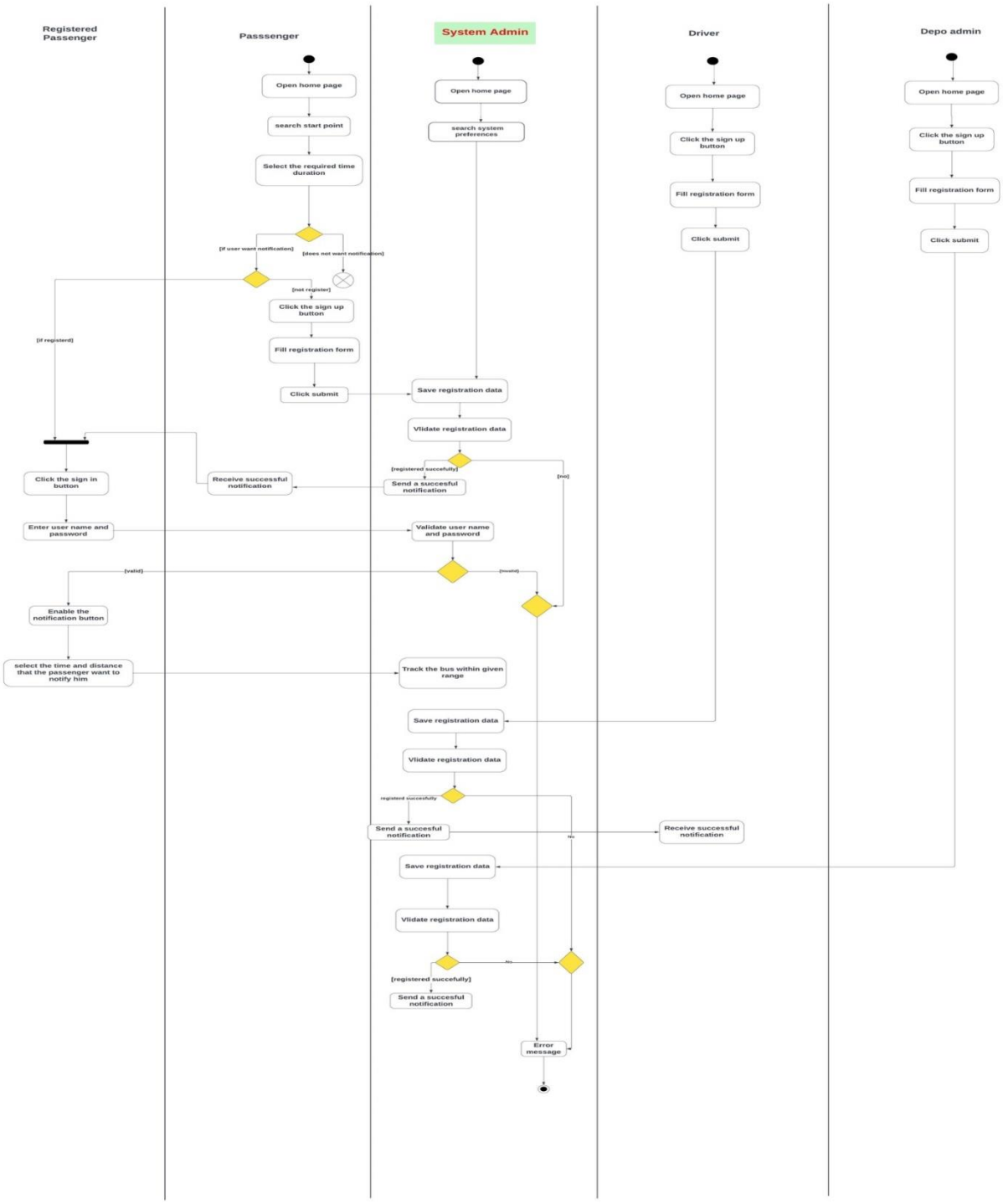


Figure 5.4.2. 8: Activity diagram for Registration data

Activity Diagram – Live location (System admin)

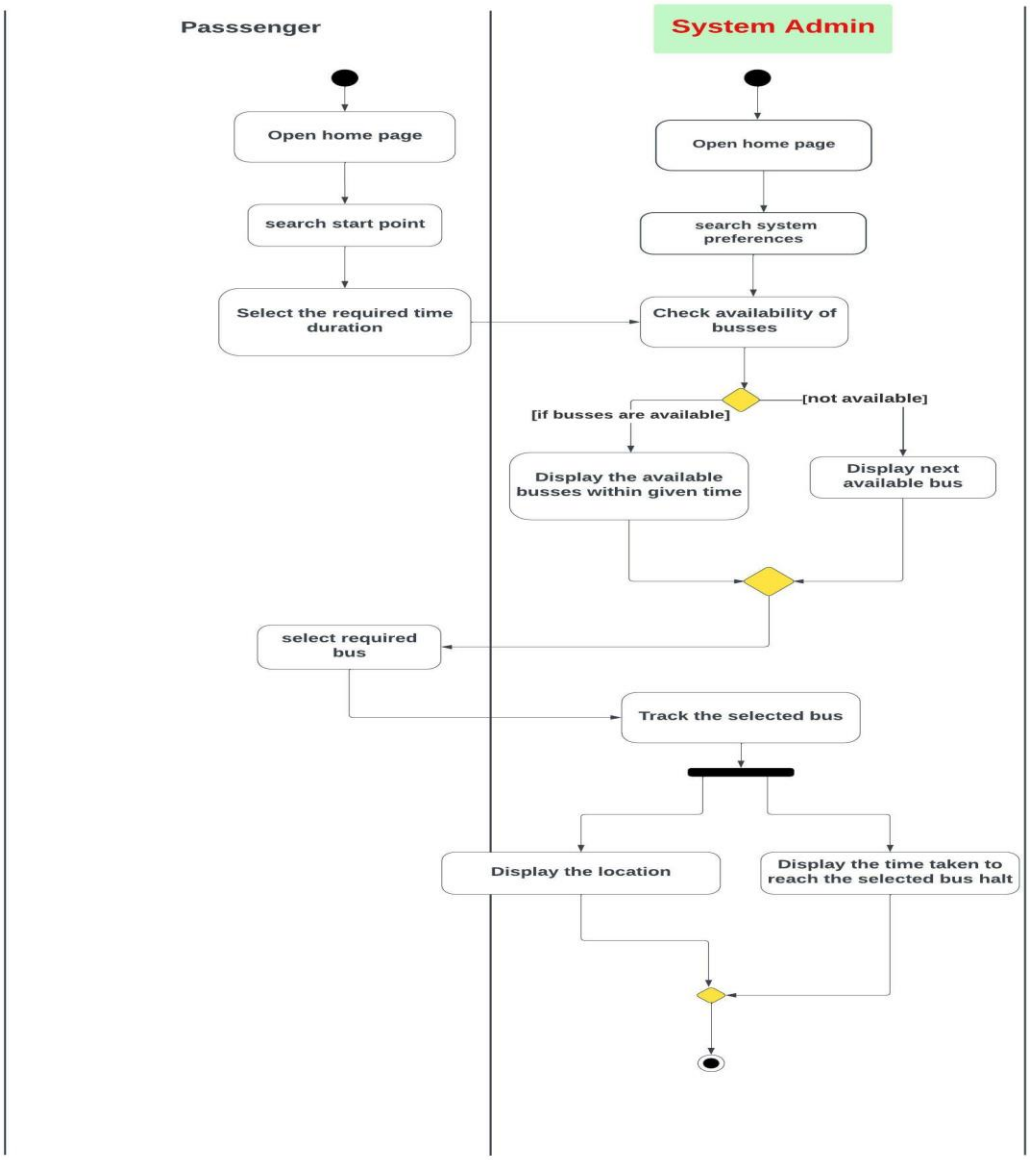


Figure 5.4.2. 9: Activity diagram for Live location tracking

5.4.3 Class diagram

A class diagram is a type of static structure diagram that illustrates a system's classes, their properties, methods, and the relationships between objects to show how the system is organized.

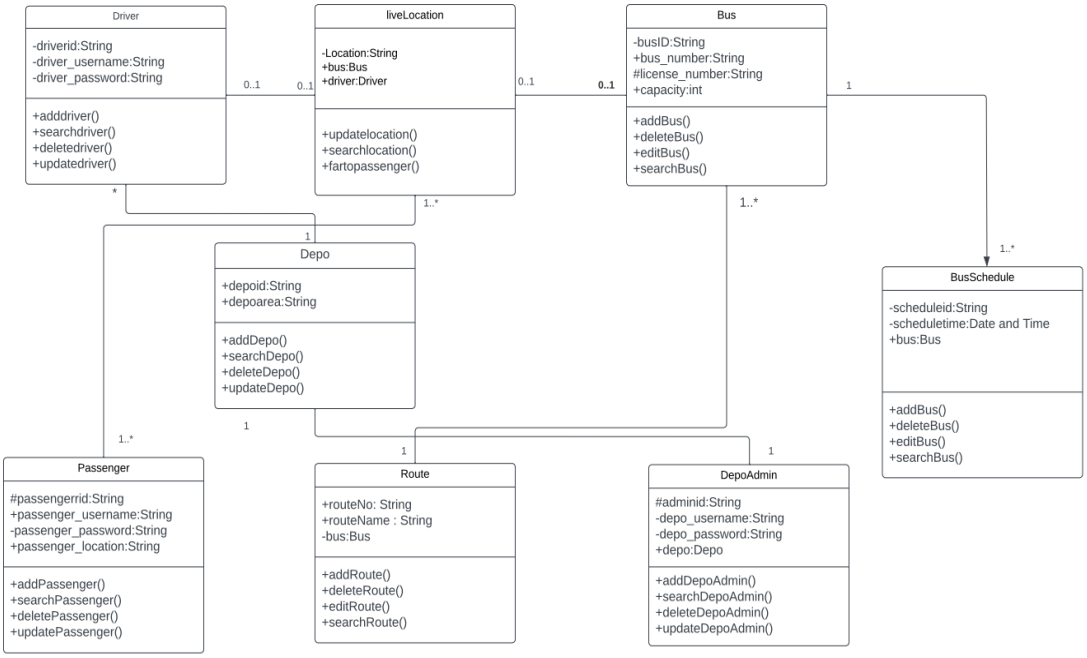


Figure 5.4.3. 1: Class Diagram

5.4.4 Sequence Diagrams

Object connections are grouped in time sequence on a sequence diagram. It illustrates the scenario's objects and the flow of messages that must be passed between them in order for the scenario to be useful. Each activity diagram is represented by a different sequence diagram that we created.

Sequence Diagram - Login and signup

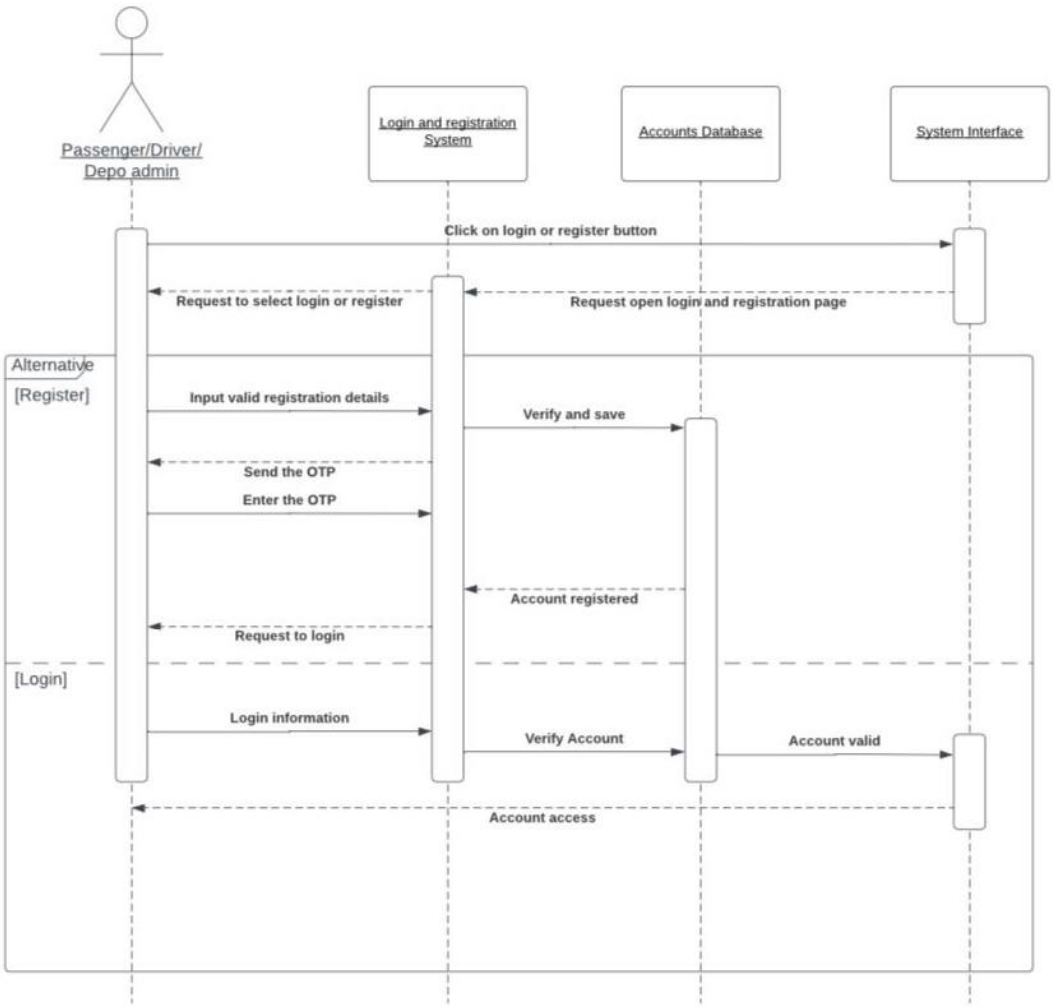


Figure 5.4.4. 1: Sequence Diagram for login and signup

Sequence Diagram - Chat app

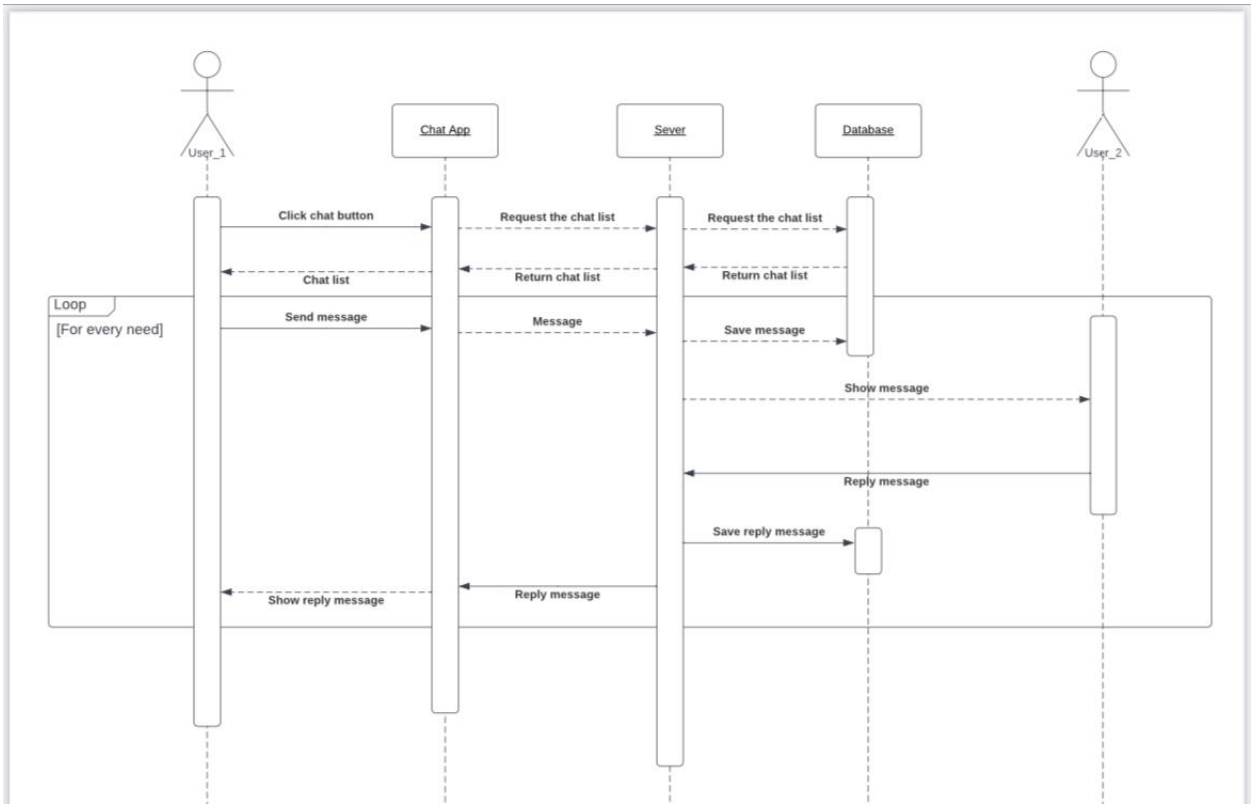


Figure 5.4.4. 2: Sequence Diagram for chat app

Sequence Diagram - Searching buses

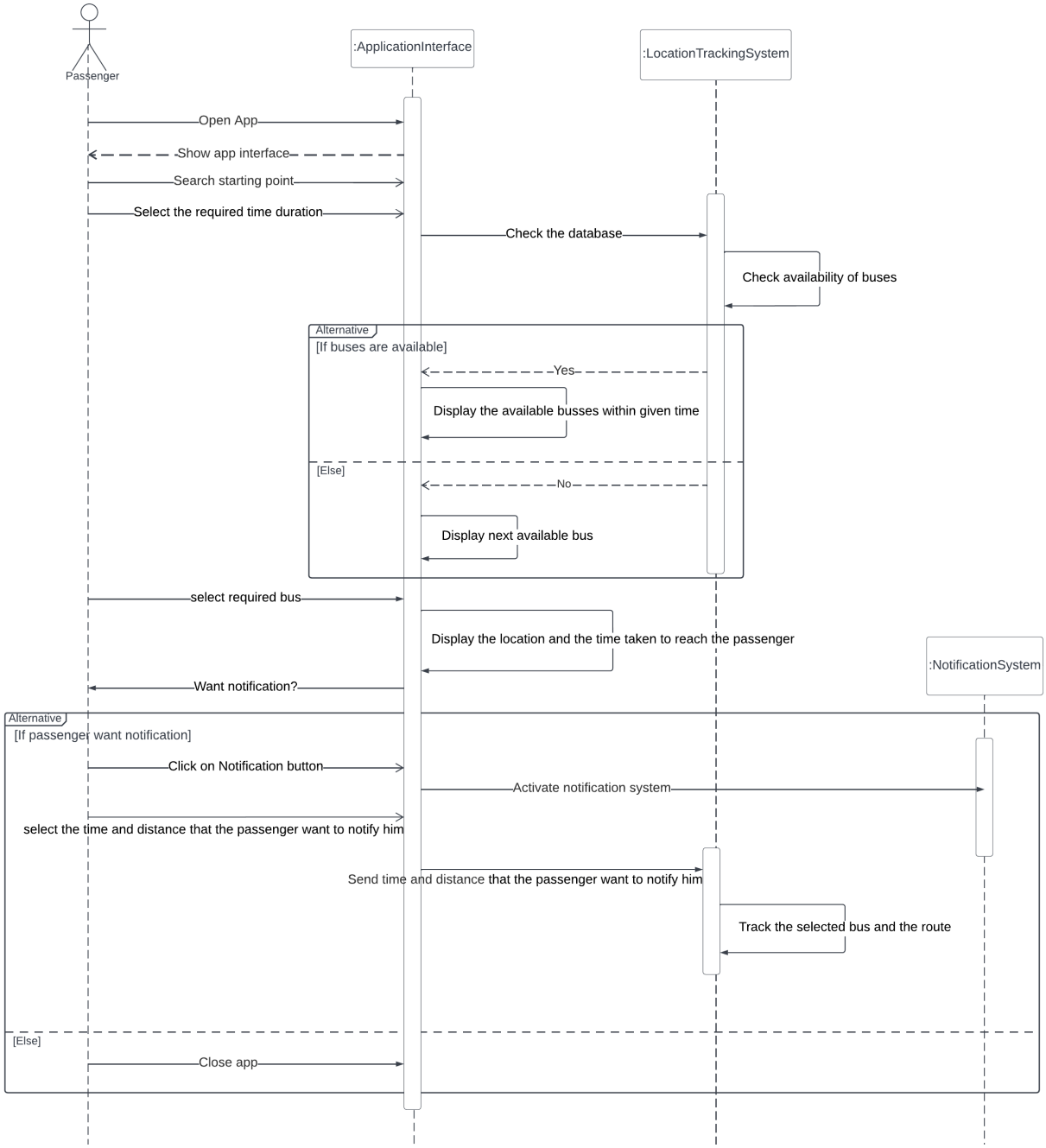


Figure 5.4.4. 3: Sequence Diagram for searching buses

Sequence Diagram - Notification system

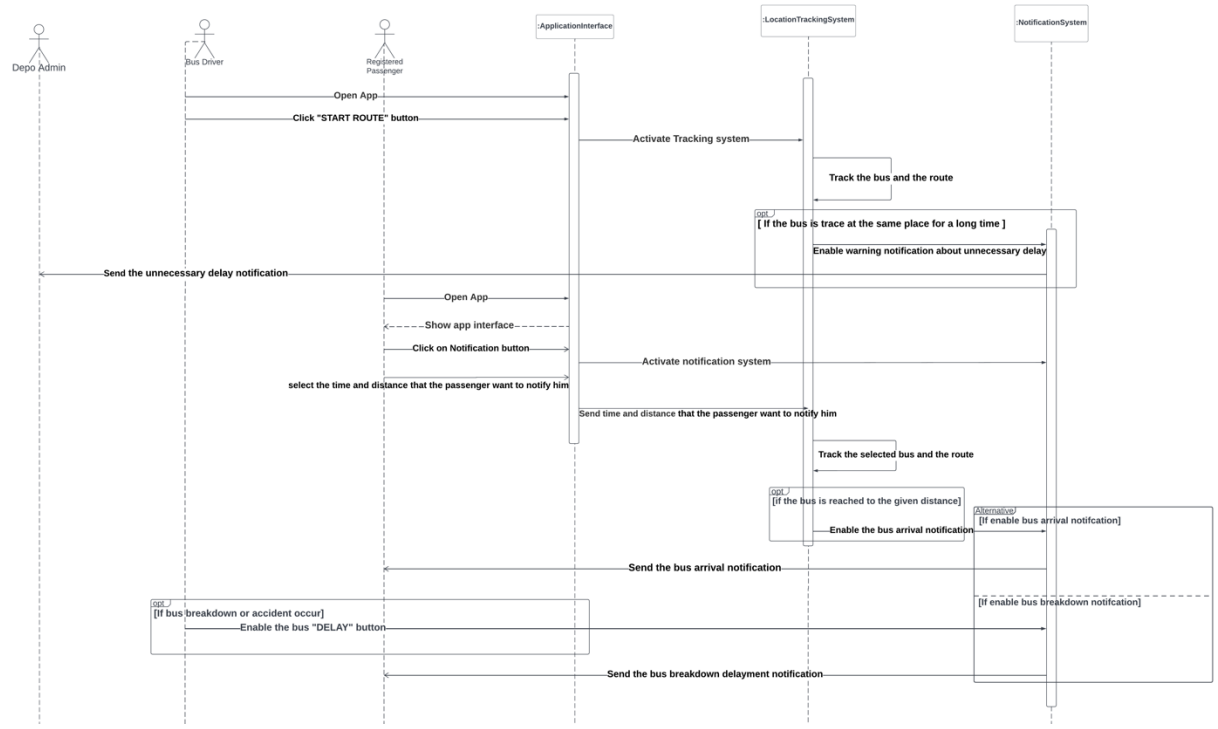


Figure 5.4.4. 4: Sequence Diagram for notification system

Sequence Diagram - Handling breakdown or other issues

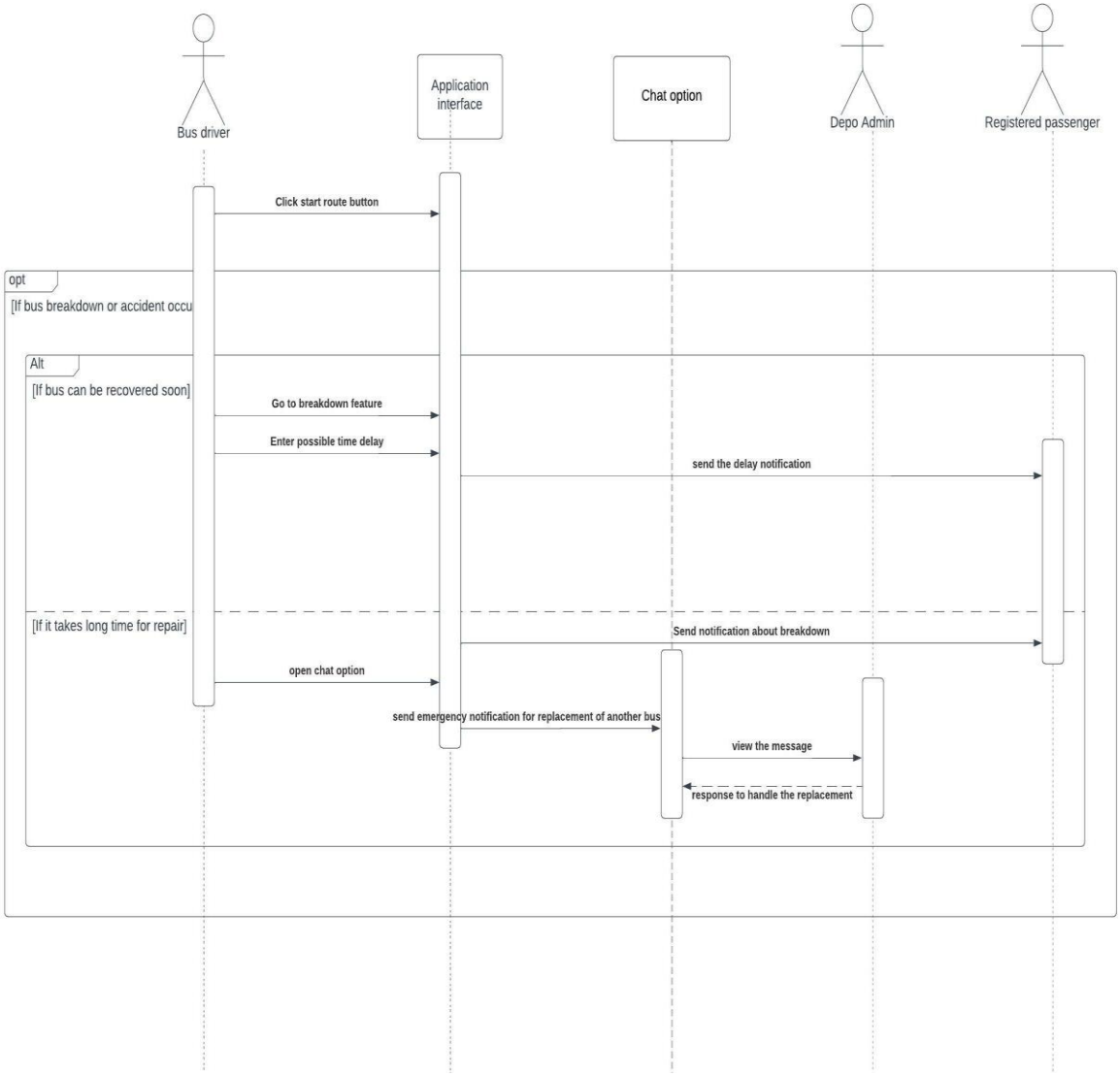


Figure 5.4.4. 5: Sequence Diagram for Handling breakdown or other issues

Sequence Diagram - Viewing daily turns schedule

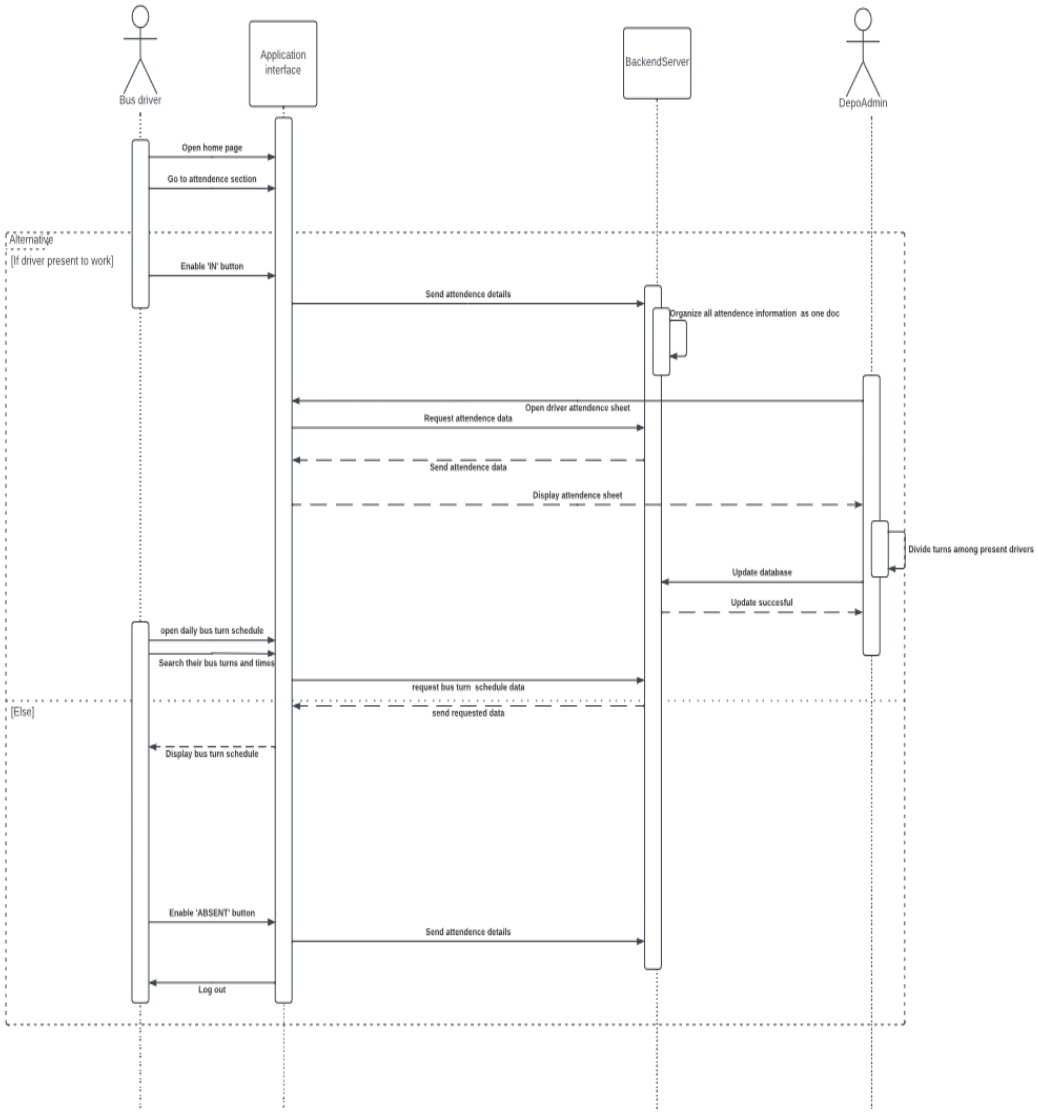


Figure 5.4.4. 6: Sequence Diagram for Viewing daily turns schedule

Sequence Diagram - Bus route management

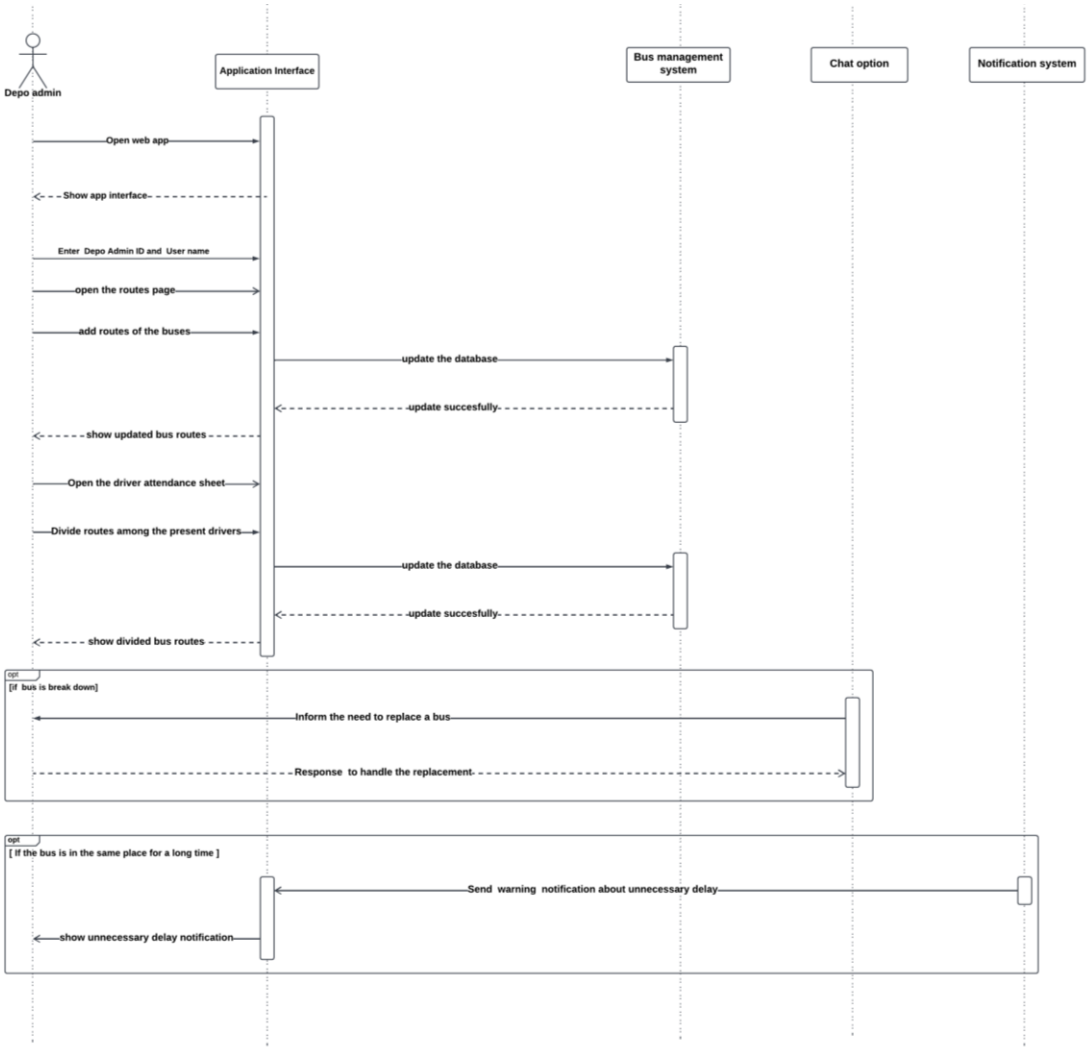


Figure 5.4.4. 7: Sequence Diagram for Bus route management

Sequence Diagram - Live location tracking

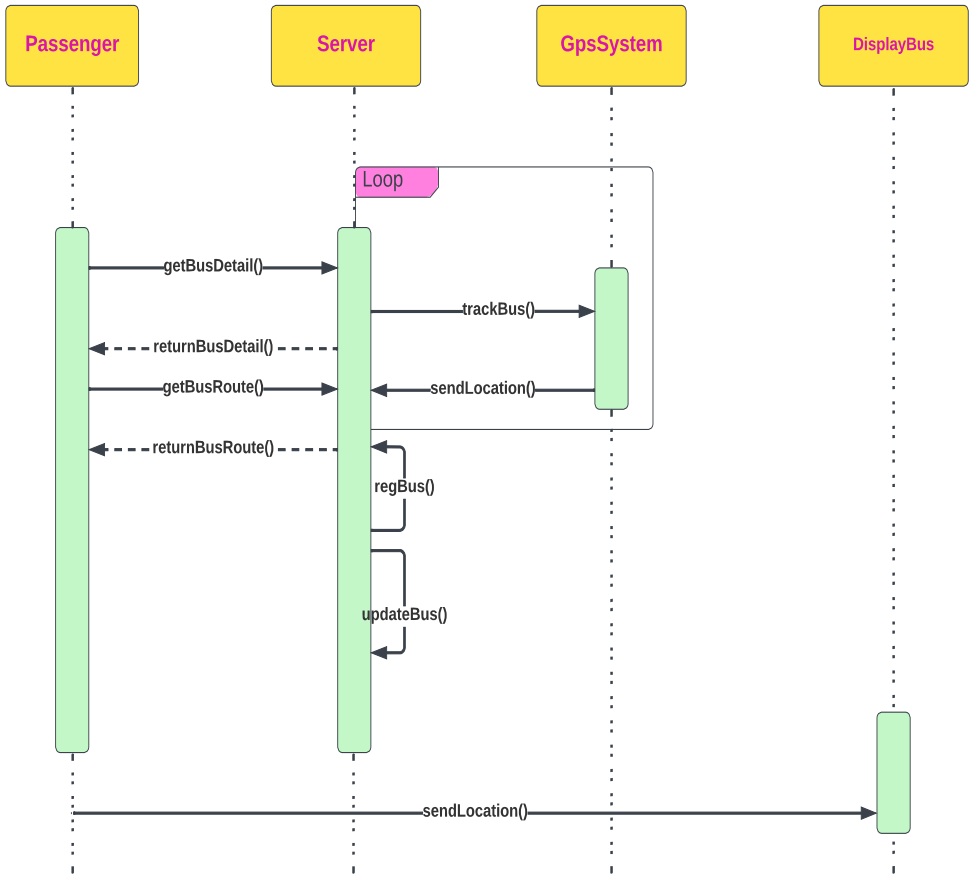


Figure 5.4.4. 8: Sequence Diagram for live location tracking

Sequence Diagram - Managing profiles and updating other information

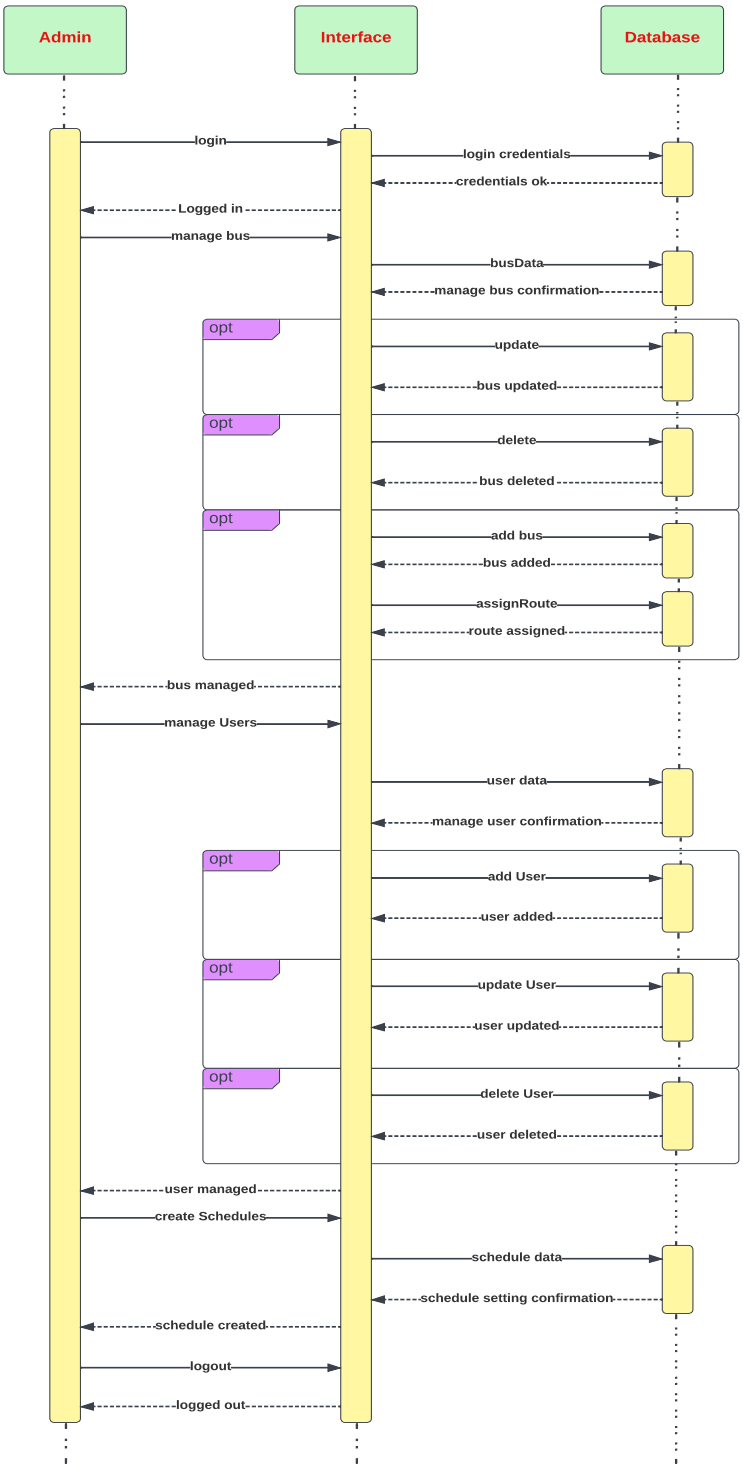


Figure 5.4.4. 9: Sequence Diagram for Managing profiles and updating other information

5.4.5 EER Diagram

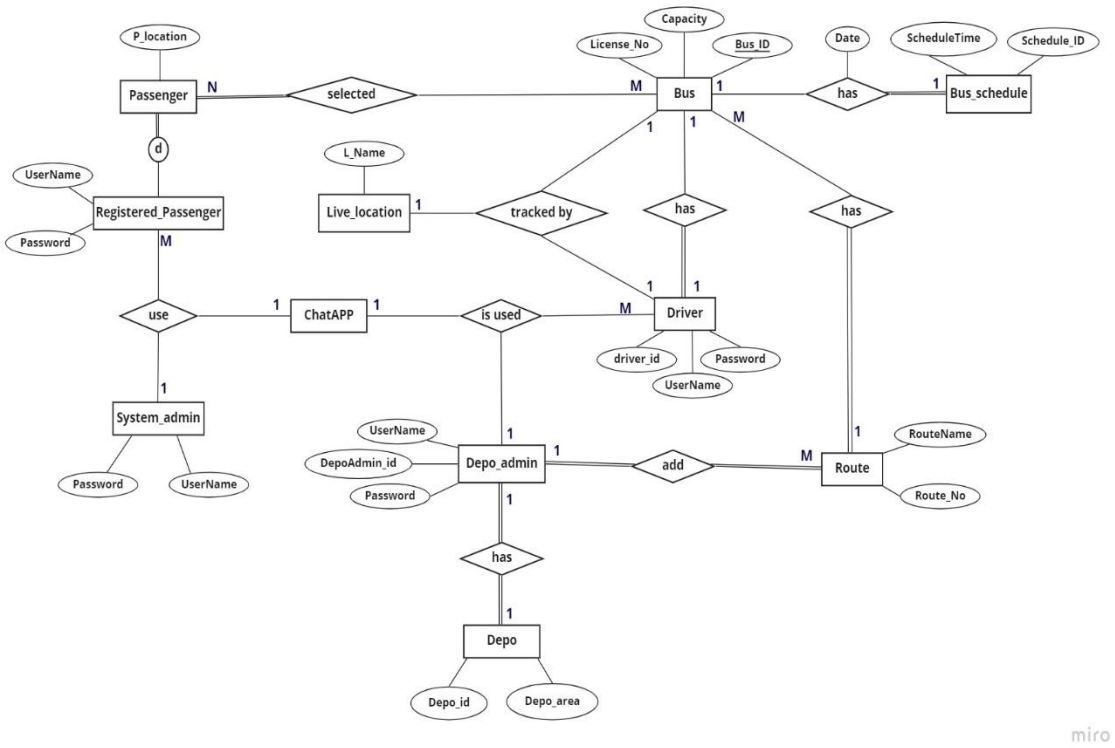


Figure 5.4.5. 1: EER Diagram

5.5 Summary

Through this chapter's analysis and design, we have represented diagrams such as use case, activity, class, sequence, and ER that give a fundamental understanding of our design and analysis mechanism.

Chapter 6

Implementation

6.1 Introduction

This chapter describes the process of implementing the software solution through identifying the system requirements followed by further analysis and the process.

6.2 Implementation process

The bus management project was built using a combination of React.js, Node.js, Express.js, and MongoDB. The user interface was designed using Figma, and push notifications were implemented using Firebase Cloud Messaging.

The development process began with the design of the user interface using Figma. This included creating wireframes, mockups, and prototypes to visualize the layout and functionality of the web application.

Next, the back-end server was set up using Node.js and Express.js. This included connecting to the MongoDB database and implementing the login and signup functionality using bcrypt to securely hash and store user passwords.

The front-end of the web application was then built using React.js, allowing for a dynamic and interactive user experience. This included the search functionality for finding buses, as well as the ability to view the current location of a bus on a map.

Push notifications were implemented using Firebase Cloud Messaging to send notifications to users when a bus was approaching their stop or if there were any issues with the bus. The Geolocation API was also used to determine the location of the bus and the user, and send notifications when the bus was close to the user's location.

To ensure the quality and reliability of the project, thorough testing was conducted. This included both manual testing and automated testing using tools such as Jest and Enzyme.

Finally, the project was deployed to a hosting platform such as Heroku or AWS, making it accessible to users through a web browser.

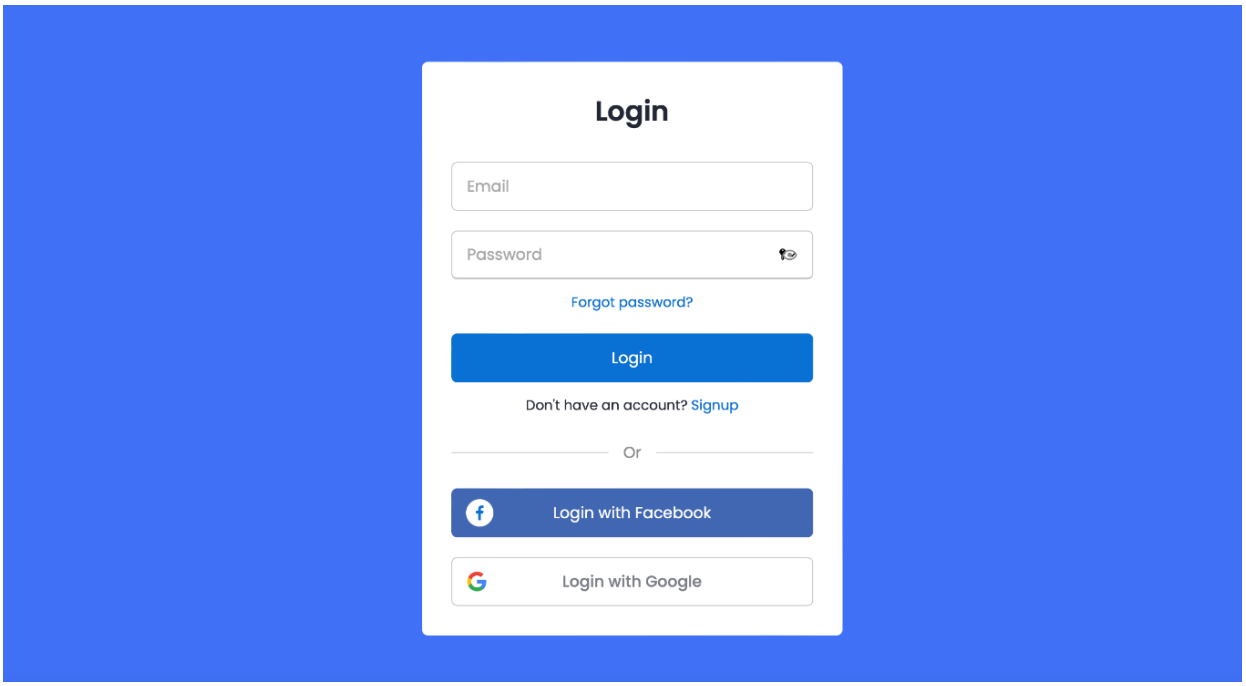


Figure 6.2. 1: Login page for frontend

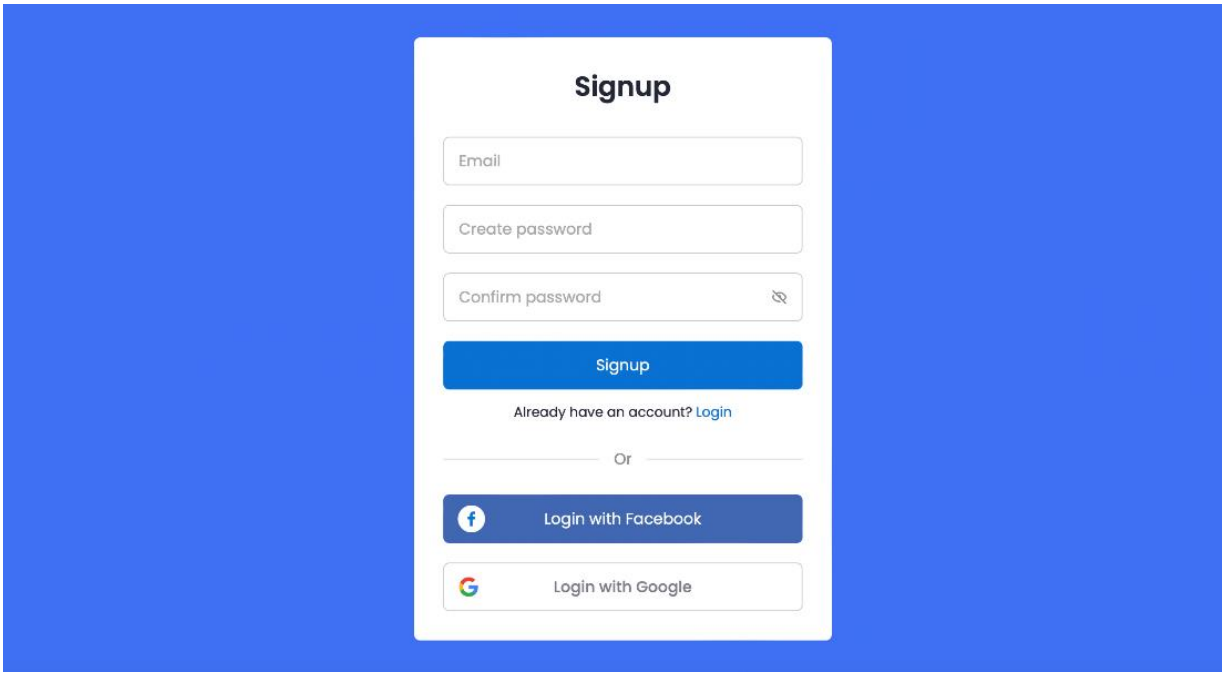


Figure 6.2. 2: Signup page for frontend

6.3 Summary

In this chapter we have discussed the system implementations done so far by us. Some of the frontend development of the overall project has been completed and the backend is under development.

Chapter 7

Discussion

7.1 Introduction

This chapter explains the overall work that we have already done and the approaches we are planning to take in future developments, testing, and implementation. In the latter part, we have included a summary of what was discussed throughout this report.

7.2 Work Done

We first examined the problem and its extent, Prior to constructing the structure of the requirements needed by the IFS project mentors and supervisor. In order to understand the entities of our system and their relationships properly, we designed the EER diagram. The system proved to be a little more difficult than we expected, so it took a long time.

But after the EER diagram was complete, we moved on to other UML diagrams, such as use case diagrams, class diagrams, activity diagrams, and sequence diagrams. In parallel, we researched the technologies we would adapt for the development of our system and designed UI scratch applications using Figma. Once the complete database schema was completed, we began the actual work.

Part of the system interfaces has been developed so far.

7.3 Further Development

In the future, we plan to expand the project by adding additional features such as the ability to purchase tickets. We also plan to optimize the application for mobile devices, allowing users to access it on the go.

7.4 Summary

Throughout this report, we have explained and illustrated the problem we are addressing, the solution we have designed, the approach we have taken to develop the solution, the work we have done up to now and the future work plan to make our system a working, finished product. We made sure to include all the steps that we have taken in developing our system and diagrams, designs and tables and figures to support our facts

Chapter 8

References

1. “System Features - GPS Tracking Sri Lanka | Vehicle Tracking System.” *GPS Tracking Sri Lanka | Vehicle Tracking System*, trackmycar.lk/system-features. Accessed 28 Dec. 2022.

2. “Google Maps - Wikipedia.” *Google Maps - Wikipedia*, 8 Feb. 2005, en.wikipedia.org/wiki/ Google Maps.

3. “Bus Management System Project Idea for Final Year.” LovelyCoding.org, www.lovelycoding.org/bus-management-system. Accessed 1 Jan. 2023.

4. “What Is Agile Scrum Methodology? - businessnewsdaily.com.” Business News Daily, www.businessnewsdaily.com/4987-what-is-agile-scrum-methodology.html. Accessed 1 Jan. 2023.

5. “ReactJS Tutorial for Beginners: Learn With Step by Step Example.” Guru99, 17 Dec. 2022, www.guru99.com/reactjs-tutorial.html.

6. “Software Engineering by Ian Sommerville - 8th Edition.pdf.” Software Engineering by Ian Sommerville - 8th Edition.pdf, docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxwaXBlcnB hc2hhfGd4OjRmODdjNzY3NDU4Njc4ZWE. Accessed 1 Jan. 2023.

7. “Free Cloud Computing Services - AWS Free Tier.” Amazon Web Services, Inc., aws.amazon.com/free. Accessed 1 Jan. 2023.

Appendix A

Individual contributions

Name of student: A.C.J. PIGERA (Leader)

In my role as team leader, I have enormous responsibility for the effective development of this system, for maintaining good contact with the business mentor, and for managing the team members in accordance with their individual skill sets during the system development phase. I therefore set up a meeting with the mentor of IFS Company first. Then my team and I spoke with the IFS mentor about the system user requirements. Following the meeting, we started investigating those customers' needs and looked for existing, comparable items on the market. We couldn’t find much alike systems in existence but there were some applications where some of the features are somewhat similar. So, this will be a big challenge to us as we understood.

Following that, I outlined the features we intend to include in this system in accordance with the needs of our system's users. I then divided up the implementation of those features among our team members based on their skill sets. Since we are following agile scrum method, I decided to use Trello board as a project management tool. When dividing the work among team members I also considered amount of task points in it for one member. We choose to use the GitHub platform to manage team collaboration during development and version control. So, I established "hackelite1" as a GitHub group. I then created and configured the system's file structure, posted it as a repository to the GitHub Organization. I then had to study Git from online tutorials and acquire the fundamental skills for version control, particularly how to manage the branch with my other team members.

My part of the project is to implement the part of system administrator and design the interface for it. I must read blogs online and have a general idea what a system administrator intends to do. Also, I must adjust them accordingly with our proposed system. As the initial step I draw use case diagram, ER diagram and class diagram with my fellow team members and designed activity and sequential diagrams for tasks done by system administrator myself. I also followed some tutorials to learn react JS and flutter as

we decided to use them as front-end technologies after discussing with the company mentors. And I'm following some courses to learn NodeJS and mongo DB to deal with backend of the system and the database. I'm still getting familiarized with these technologies. Using figment, I have designed some interfaces of the system administrator so far. Furthermore, I guided my team members in creating project proposal, interim report, SRS, user stories, and interim presentations and worked with them.

Name of student: C.S.L. BASNAYAKA

As part of the development team for the bus management system, I was responsible for implementing the login and signup functionality of the system. This included designing the user interface for the login and signup pages using Figma, setting up the back-end server using Node.js and Express.js to handle the authentication process, and integrating the system with the MongoDB database to store user information securely.

I also implemented to securely hash and store user passwords. In addition to the login and signup functionality, I developed a chat app using Socket.io that allows users to communicate with the driver and depo admin in the event of a bus breakdown or other issue. My contributions to the project have helped to make the system user-friendly and reliable for passengers and have played a key role in ensuring that the system is able to handle the needs of users and provide a seamless experience for those using the system to plan and take bus trips.

Name of student: W.H.I. SILVA

I am responsible for designing and implementing driver interface for both web and mobile applications. At the designing stage firstly, I also contributed designing the use case diagram, ER diagram and the class diagram with my fellow group members. Then as the initial step to clearly understand my part and things need to be implemented in driver's view I designed Activity diagrams and sequence diagrams using lucid chart. Furthermore, I helped in designing interfaces of Web application and mobile application using Figma scratch together with other members. When designing the UI, I had to make sure that it is simpler to understand and easy to get the users tasks done as they do not come from high educational backgrounds. So, I always had to assure its user-friendliness and simplicity.

Also, I had to do some research about public bus transportation and how things going on in traditional bus management systems. The main tasks had to be prioritized after identifying the bus drivers' general requirements regarding bus transportation.

We decided to develop web interfaces and frontend using ReactJS framework, bootstrap and for mobile interface we use flutter. For the backend we decided to use NodeJS. Therefore, first I followed some tutorials and studied the frameworks thoroughly. And also I studied about mongo DB as we are going to use that for databases. Still I'm getting familiarized with these technologies. Furthermore, we decided to use GitHub for version controlling. Since I am not familiar with it, I studied about it also to get understanding of how to use and work with it. I also went through official documentation of all those technologies.

Also, I contributed when creating documents regarding our project such as the project proposal, interim report, SRS, user stories and interim presentation.

Name of student: J.A.D.A. KAVINDI

I was assigned mainly to develop the passenger interface of the web application and mobile application. In the passenger interface, the unregistered passengers should be able to search a bus by giving the starting location. If the passengers want to select a specific bus and get notification regarding that selected bus, they should be register to the system. And also, registered passengers should be able to enable the notification button and select a time duration that the passenger wants to notify them.

We have decided to develop the web application frontend with React and backend with NodeJS. I was not that much familiar with those technologies at that moment. Therefore, I had to learn those technologies thoroughly. So, I am currently following several courses, YouTube videos and other documents to get familiar with them. Meanwhile, I got familiar with Bootstrap to make interfaces responsive to several screen sizes. Furthermore, we decided to use GitHub for version control. Since I am not familiar with it, I studied and worked with it before we start development.

At the designing stage, I contributed to the team in designing the class diagram, the use case diagram and ER diagram together with other members. I designed the activity diagrams and sequence diagrams by myself for searching a bus by giving starting location, selecting a required bus, enabling the notification button to receive arrival and breakdown notification.

When creating the User Interfaces, we used Figma tool for it. As I was not experienced with it, I had to follow some tutorials and videos to get familiar with it. I contributed to the team in designing the interfaces for the web application using Figma. Also, I contributed when creating documents regarding the project such as the project proposal, interim report, SRS, user stories and interim presentation. During the process of developing our system, as a team, we managed to clarify uncleared points by having team discussions and by arranging meetings with the supervisor and the mentors.

Name of student: A.N.K. ADHIKARI

I am responsible for creating the depo admin interface of the web application and mobile application. In the depot admin interface, the system facilitates adding routes for each depot. The interface is capable of providing facilities to the depot administrator such as viewing the attendance of bus drivers, allocating routes among present drivers, updating the daily turn schedule, getting notifications about unnecessary delay and receiving an emergency message from the bus driver through chat option.

We have decided to develop the web application frontend with React and backend with NodeJS. I didn't know much about those techniques at that time. So, I had to learn those technologies well. For that, I study specific videos, websites and medium articles regarding those technologies. For web development, I study react tutorials on YouTube. Additionally, we chose GitHub as our version control system. Before starting web development, I learned and worked because I had no experience in it. Meanwhile I learned how to create responsive interfaces for different screen sizes using Bootstrap and Media query. During the design phase, I contributed to the team in creating class diagram and use case diagram along with other members. I created activity diagrams and sequence diagrams myself to illustrate the functionality of Depot Administrator. Also, I contributed to the team in creating the interfaces for the web application using Figma scratch.

Additionally, I contributed with the development of project-related documents such the project proposal, interim report, SRS, user stories, and interim presentation. As a team, we were able to clarify unclear points while developing our system by holding more group meetings and discussions and setting up meetings with mentors and supervisors.

Appendix B

Gantt Chart

Task	Duration(week)	Start Date	Timeline																							
			2022								2023															
			November				December				January				February				March				April			
			1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Draw structural and behavioral diagrams	2	01-11-22																								
Learning about technologies	4	16-11-22																								
Interim Report Submission	2	15-12-22																								
Implement business logic in backend	2	06-01-23																								
Create web interfaces	3	21-02-23																								
Create mobile interfaces	3	12-02-23																								
Connect frontend with backend	1	06-03-23																								
Test in the dev environment	2	14-03-23																								
Initialize deployment environment	2	29-03-23																								
Debugging the Beta Release	2	13-04-23																								
Deploy final product	1	28-04-23																								

Table 4.3. 2: Gantt Chart

Appendix C

Mockups

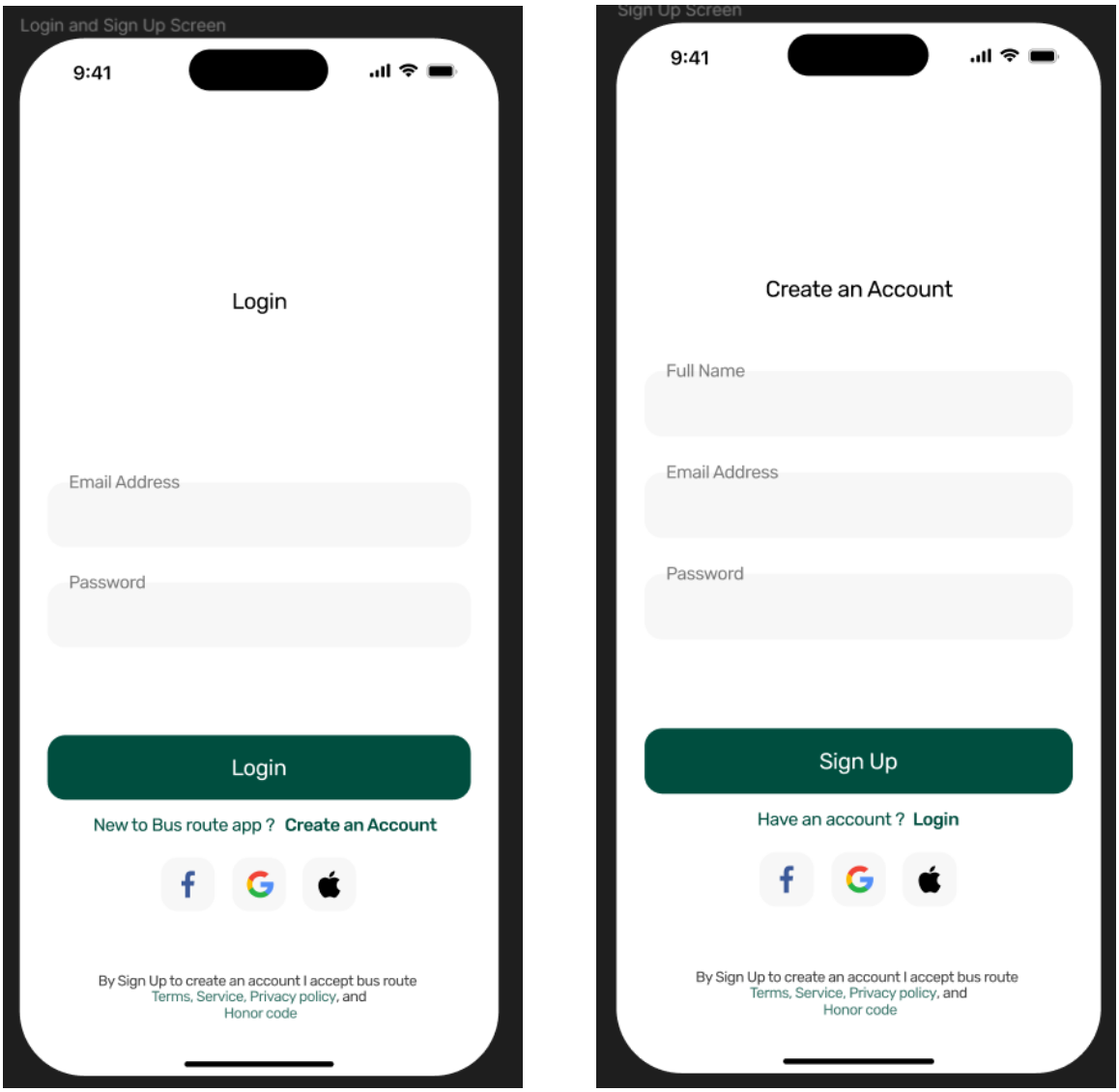


Figure 9. 1: Interface for Signup and login page

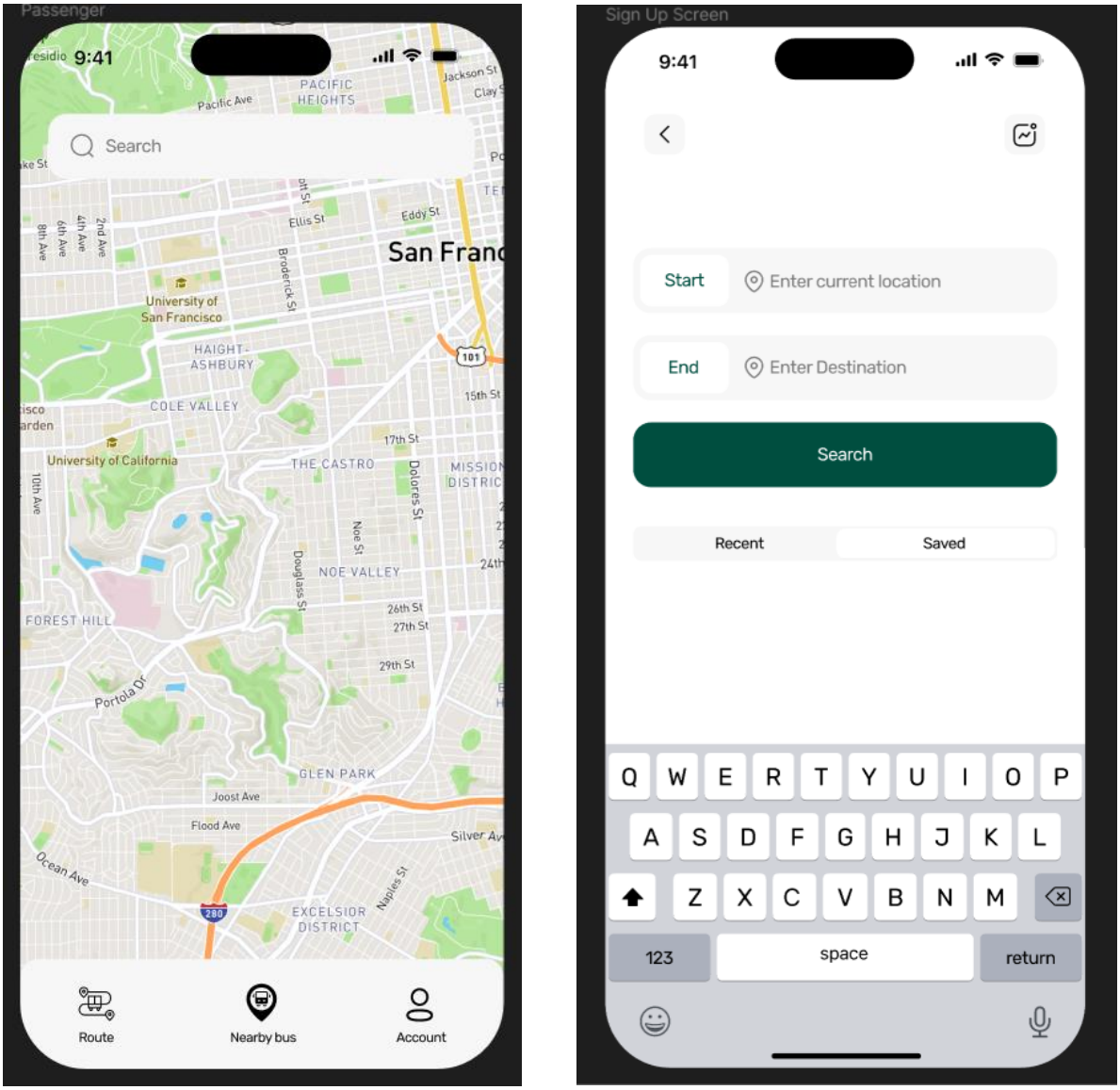


Figure 9. 2: Interfaces of Search buses by passenger

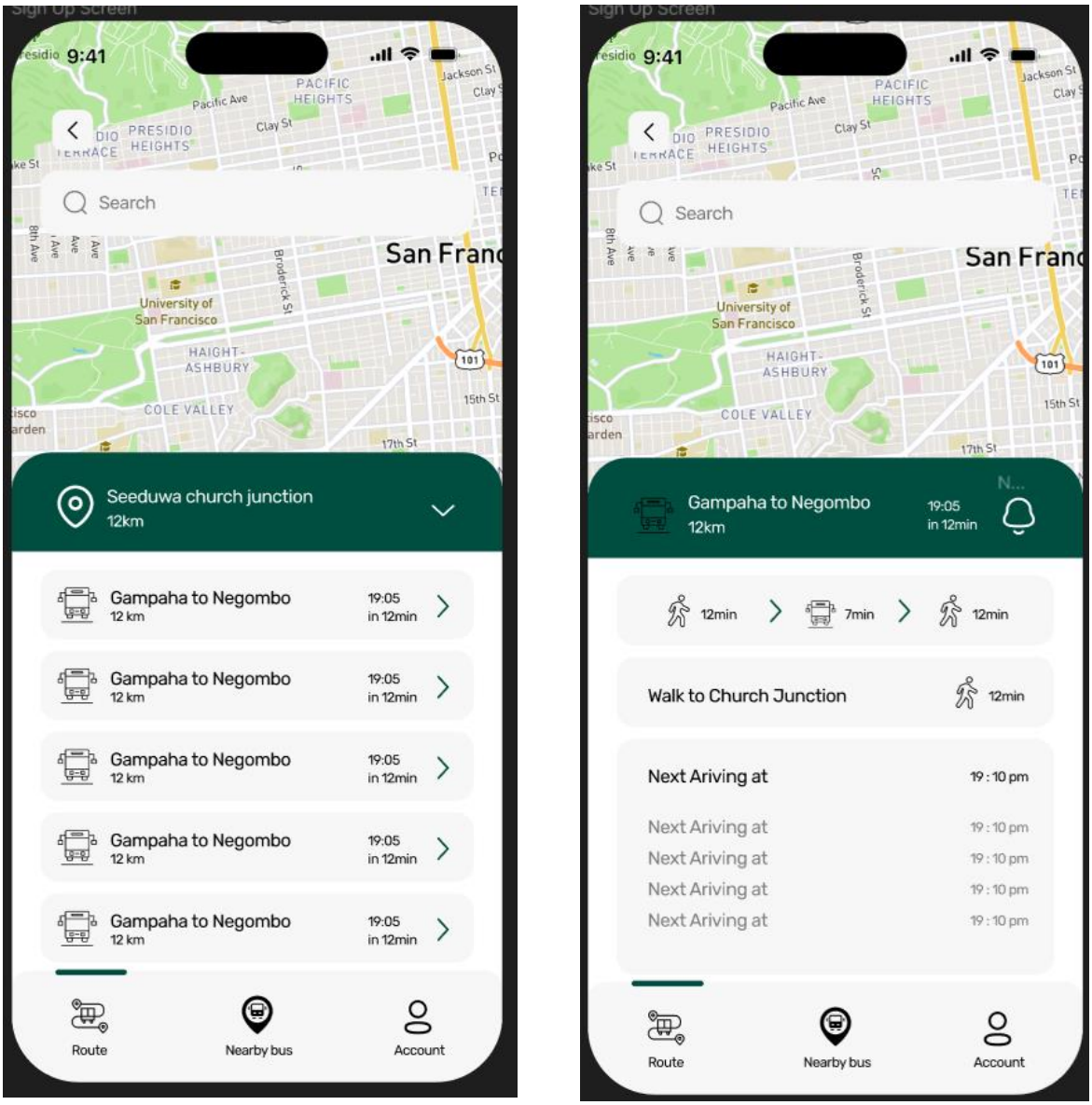


Figure 9. 3: Interfaces of available buses

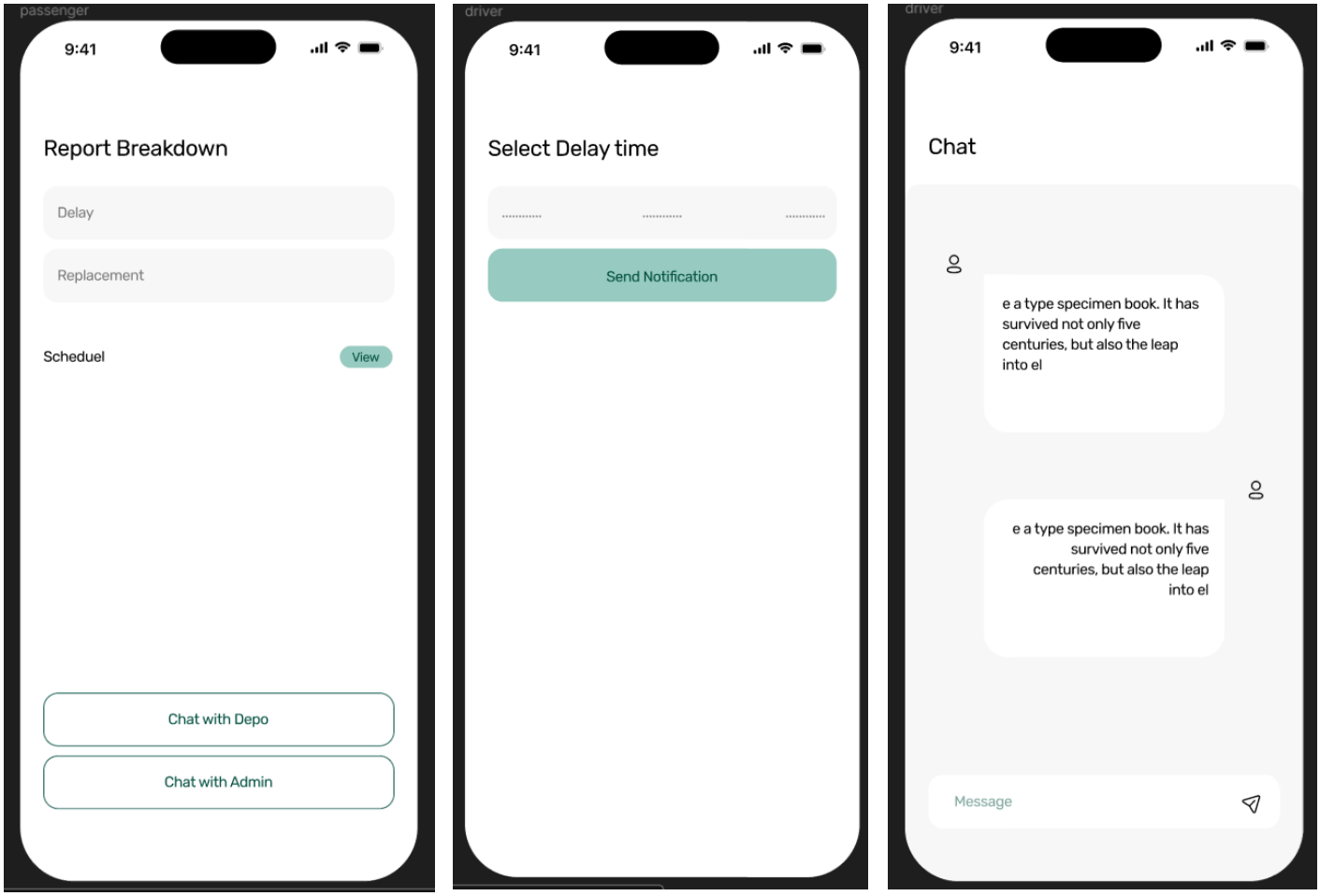


Figure 9. 4: Report breakdown page of Driver Interfaces

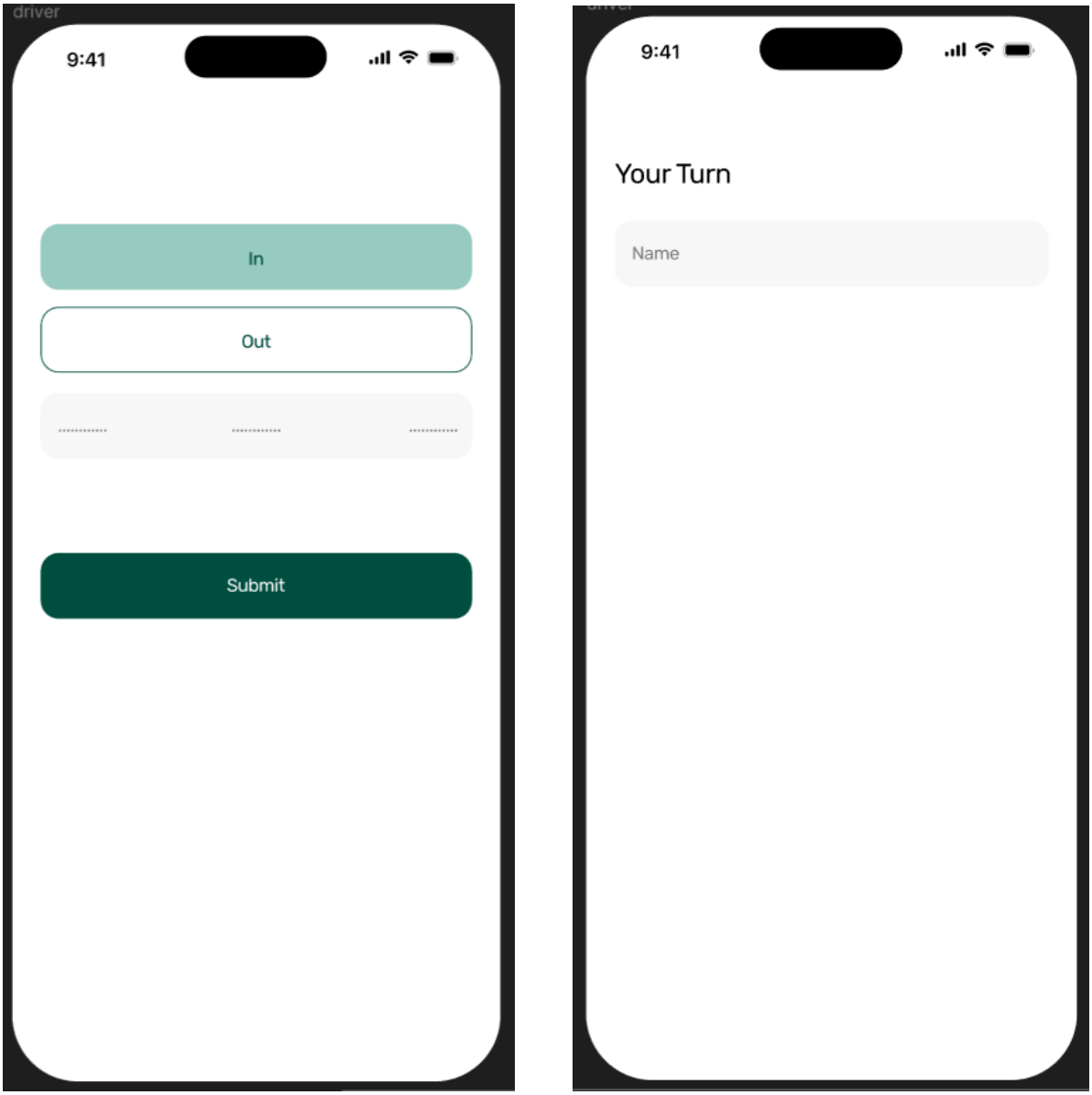


Figure 9. 5: Mark attendance page of Driver Interface

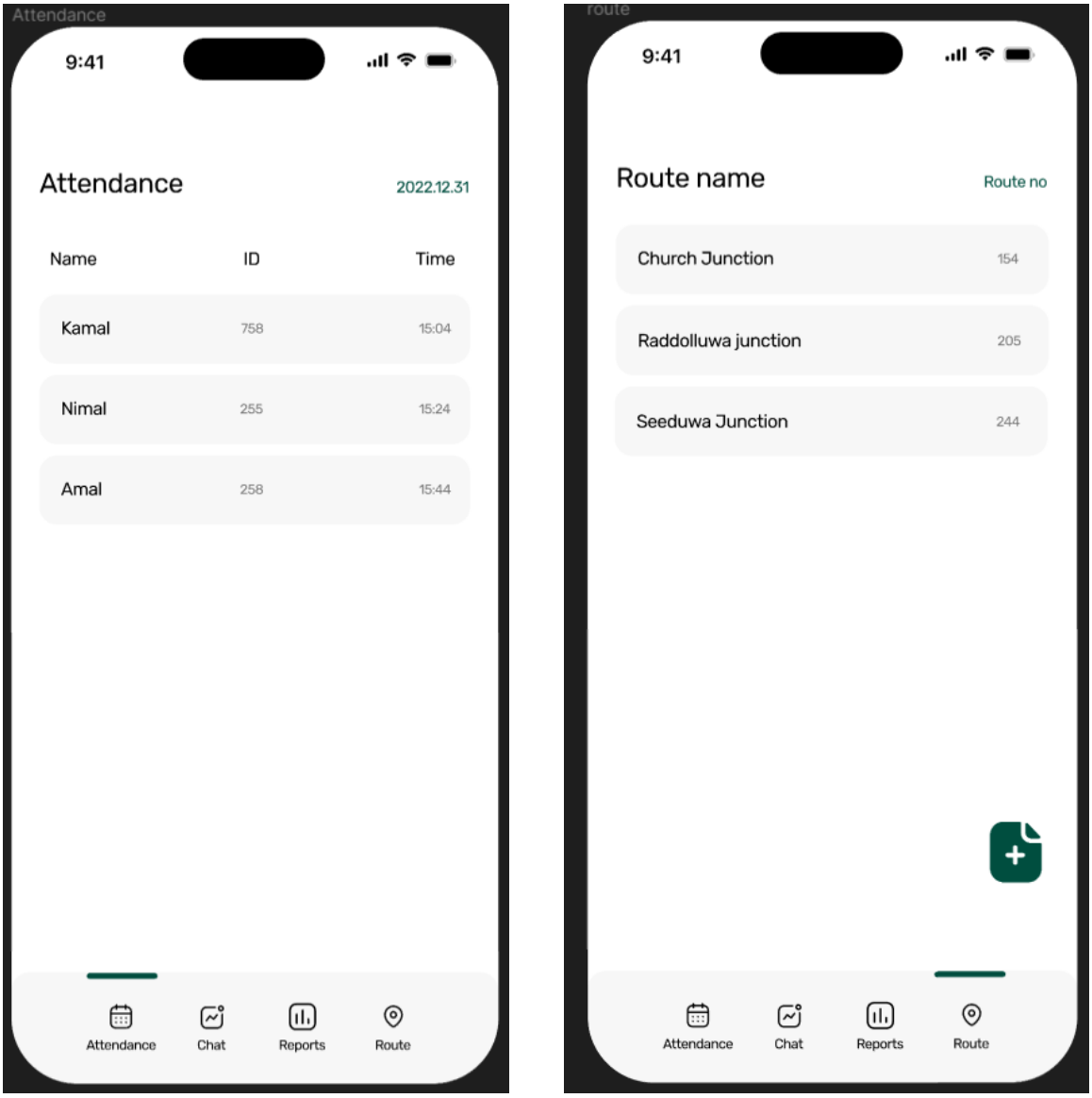


Figure 9. 6: Route page of Depo Administrator Interface

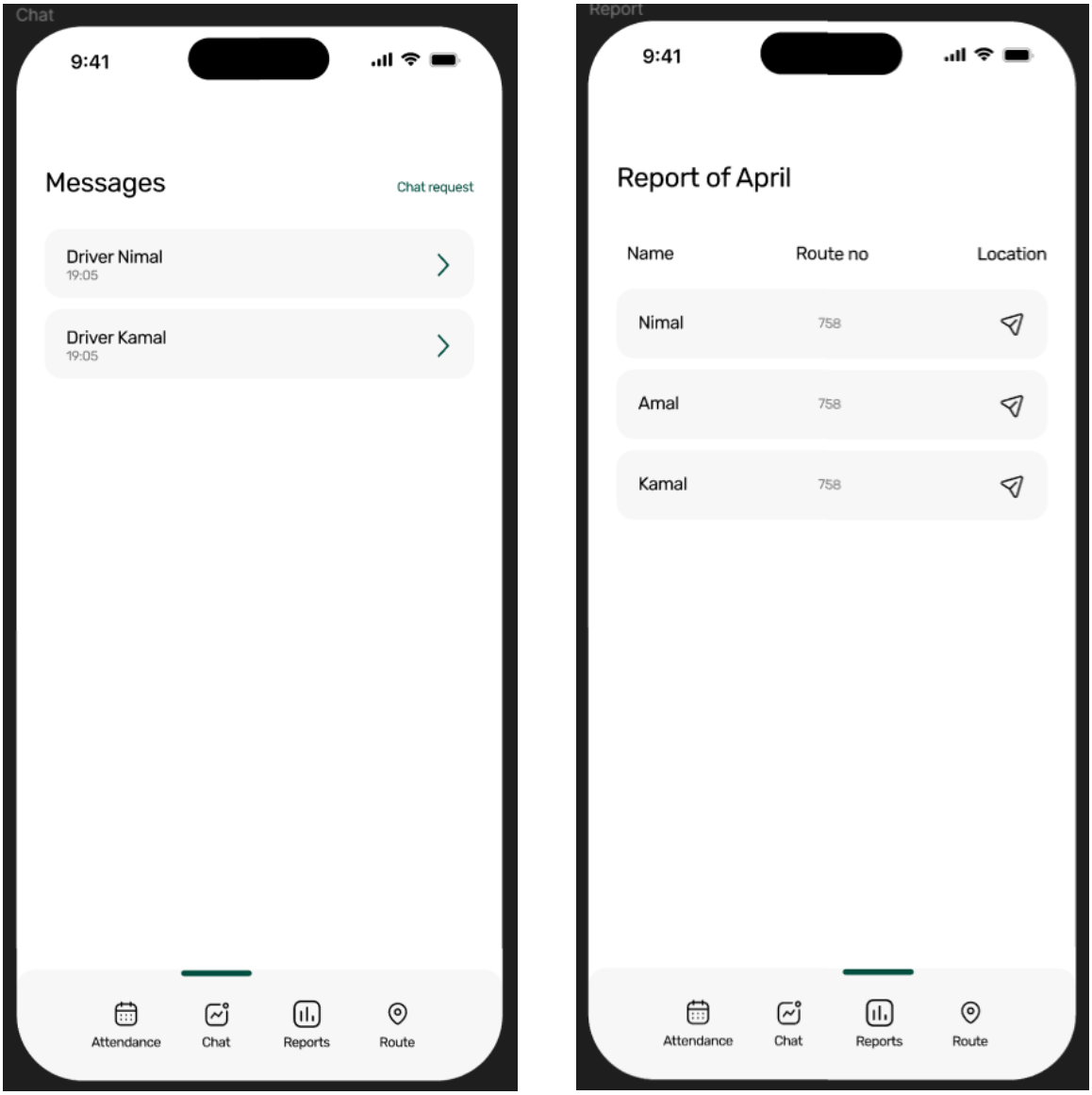


Figure 9. 7: Report Page of Depo Administrator Interface

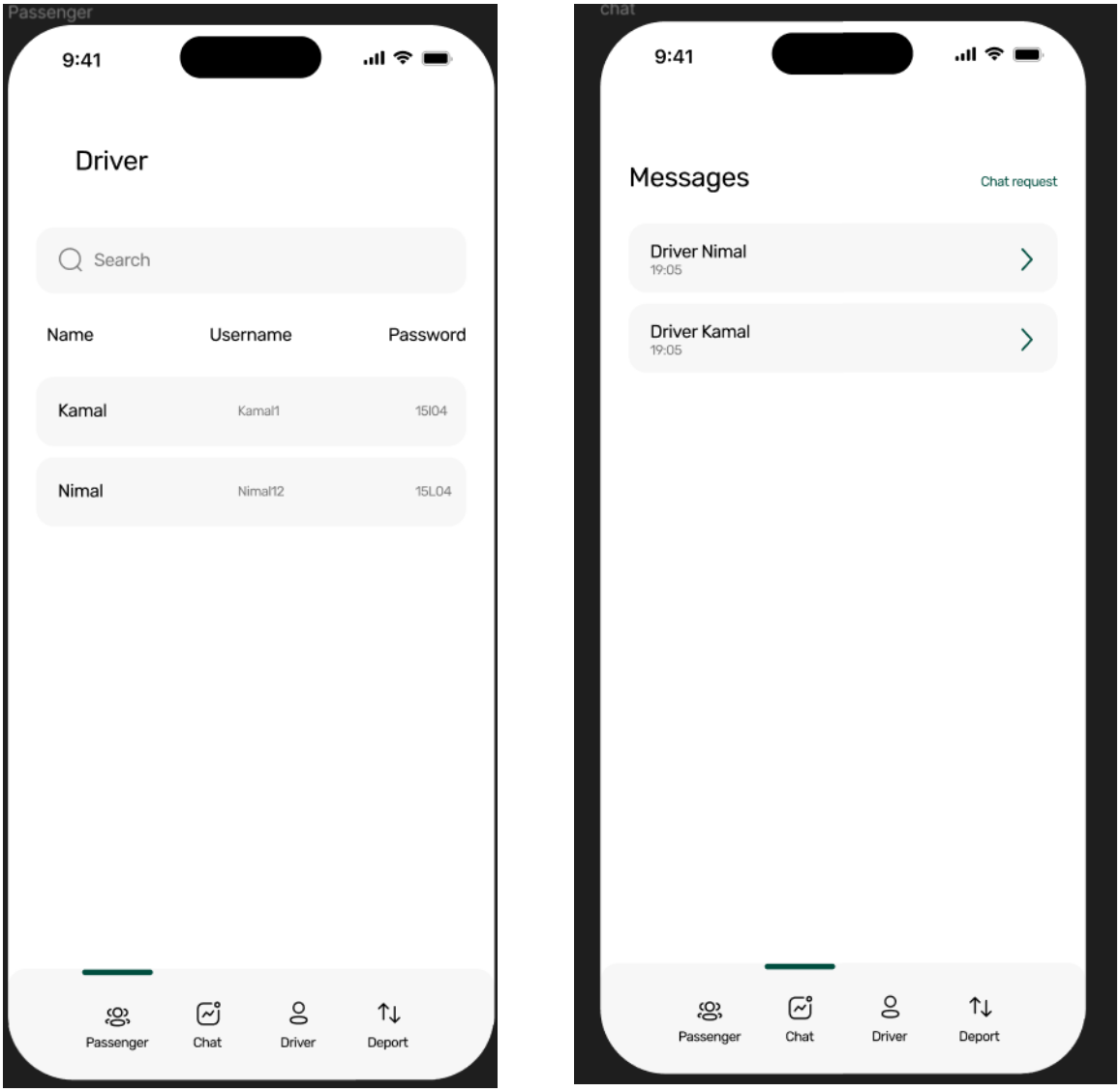


Figure 9. 8: System Administrator Interfaces

Appendix D

Software Requirement Specification

The software requirements specification report has been attached to this document as an appendix. Note that captions, page numbers, figures numbers, and table numbers in the SRS document are not linked with the project interim report.

Software Requirements Specification

for

Smart Notification System for Bus Management

Version 1.0 approved

Prepared by Team HACK ELITE

Faculty of Information Technology

University of Moratuwa

30.12.2022

Table of Contents

Revision History

Date	Version	Description
December 30, 2022	1.0	Initial Version

1. Introduction

1.1 Purpose

The main purpose of this document is to describe and to demonstrate the functionality of the smart appointment system for bus management. It includes a thorough explanation of all the criteria for this system's external interface, both functional and nonfunctional requirements. Additionally, this document will also provide a detailed idea about how this system can be replaced to the existing manual system and make the task easy for all the parties involved in bus transport and for passengers.

1.2 Document Conventions

This SRS document adheres to the IEEE 830-1998 standard for producing SRS documents.

1.3 Intended Audience and Reading Suggestions

This document will be helpful to depo administrators, drivers and passengers who use public bus transportation.

We recommend all the users of this system to go through the overall description of this document.

Testers and developers are recommended to read the nonfunctional and functional requirements and the external interface requirements in this document.

1.4 Product Scope

The aim of this project is to design an application where passengers can search for the needed bus in a specific time duration, to reach their destination, and get notifications (ex: delays, arrival time etc.) and necessary information at their fingertips. Also keeping the passenger aware of the location of the buses in real-time. The system is also making the tasks of the depo easier and efficient. They can add routes, divide routes and handle breakdowns of buses through the system. Drivers also can report issues if any breakdown

occur and mark their attendance through the system in order for the depo administrator to prepare bus turns as a schedule. So that drivers can view them. Also getting passenger complaints can also be done through the system.

1.5 Definitions, acronyms, and abbreviations

Passenger-people who use public bus transportation and use the system.

Depo admin- person in the depo in each area who is assigned to handle the tasks regarding day-to-day bus transportation such as dividing bus turns etc.

System admin-person who handles the general system operations and fixing issues in it.

1.6 References

IEEE 830-1998 standard for writing SRS document

Somerville, Software Engineering, 8th ed. England: Addison-Wesley, 2007

2. Overall description

A system overview will be provided in this section. The system will be explained in its context to highlight the basic functionalities of it. Additionally, it will outline the various user types who will utilize the system and the features that each type of user will have access to. The system's limitations and presumptions will then be presented.

2.1 Product perspective

The system proposed through our project is available as a web application and a mobile application.

We are developing this system in a way that makes it possible for any type of individual to use it without depending on their educational level as transportation is a common need for any person. As it has been designed using a three-tier architecture, this system is anticipated to be considerably more efficient and effective for its users.

The system also allows users of the system to communicate through chat option in specific instances where it is needed for handling certain issues.

2.2 Product functions

For all users -

1. Register and login to the system
2. Search starting location and the required time duration
3. Select required bus

Registered passengers -

1. Enable the notification button
2. View the location of the bus real-time
3. Select the time and distance that the passenger wants to notify him (bus arrival time notification)

4. Submitting complaints of any issue through chat option

Driver -

1. Entering attendance information daily whether present or not
2. View daily bus schedule and their turns
3. Report any issues about delays or breakdowns through chat app
4. Inform passengers about time delays
5. Request replacement for bus when needed

Depo administrator -

1. Add routes of the buses
2. Check the driver attendance sheet.
3. Divide routes among the present drivers.
4. Manage the newly added bus routes.
5. Update the newly added bus schedules.
6. Response through the chat option to handle the replacement.

System administrator -

1. Login using username and password.
2. Add newcomers to the system like drivers & depo administrators.
3. Manage their profiles.
4. Delete some details permanently from system.
5. Removing someone who has been assigned to any other work or retire from the job.
6. Helps to the passengers if they need some clarification about the logging process.
7. Generate monthly reports by using some documents.
8. Give the positive feedback through the chat option for the passengers' requirements.
9. Track the bus live location.

2.3 User characteristics

Users of this application are registered passengers, unregistered passengers, bus drivers, depo administrators and system administrators. Users need not have complex technological knowledge as users can get services from the system or do the thing they want by simple operations such as from simple button clicks.

2.4 Operating environment

The application can run on any machine that has a windows operating system with a version greater than windows 7 and a stable internet connection. Mobile application can be run on any smart phone with proper internet connection.

2.5 Design Development and Implementation Constraints

There is a certain number of data that the users should provide to the system when registering to the application. Once the details are authenticated, all the collected data are stored in a MongoDB database. The backend will map the data in the database and provide it to the frontend in a JSON format whenever the frontend requests any data that is present in the database. Because an indirect method increases the likelihood of being breached by an outsider; the frontend is not permitted to access the database directly.

For the purpose of logging into their accounts and carrying out the necessary operations, registered users must have their right username and password. The system has both private and public information which can be accessed up to different levels by different users.

Users can access the platform using any computer, tablet PC, or smartphone with an internet connection and web browsing capabilities.

2.6 User Documentation

Customers will be provided with a user manual along with this software. We also include online help in the application itself.

2.7 Assumptions and Dependencies

- For the system to function properly, every user must have an internet connection.
- The system must be up to date all the time.

3. External interface requirements

3.1 User Interfaces

1. **Passenger Interface**: Through the passenger interface, map is displayed all the time. The web platform will have a passenger interface that allows passengers to create accounts, login, searching buses by entering the passenger's starting point and time duration, view all the buses passing the given starting point, enable notification button, receive bus arrival notification and bus breakdown notification. And also, passengers can view the real time location of the selected bus through the map.

2. **Driver Interface**: The web platform will have a driver interface for drivers that allows them to create accounts, login, mark attendance, view the daily turn schedule and their specific turns, enable the "DELAY" button due to breakdown or accident, select the possible time delay, and send emergency message through the chat option for a new bus replacement.

3. **Depo Admin Interface**: The web platform will have a depo admin interface for depo administrators that allows them to create accounts, login, add the routes to the system, view the attendance information, divide routes among the present drivers, response through the chat option to handle the replacement, and receive warning notification about unnecessary delay.

3.2 Hardware Interfaces

1. The software system will support Windows and Mac, android operating systems.

2. The software system will require sufficient memory to run the web platform.
3. The software system will require an internet connection to function properly.
4. The software system will require the necessary hardware and software to access the web platform and mobile app.

3.3 Software Interfaces

1. The web application allows users to create an account, manage their profile, select their respective user interface, send feedback to the system administrator.
2. The system should use a database (such as MongoDB) to store user information and bus data.
3. The system should be able to retrieve, update, and delete data as needed.
4. The interface should be visually appealing and use consistent design elements such as fonts, colors, and layout.
5. The interface should include clear and concise instructions and explanations to help users understand how to use the system.

3.4 Communication Interfaces

1. The system should use the Geolocation API to determine the location of the bus and the user, and send notifications when the bus is close to the user's location.
2. The system should use Firebase Cloud Messaging to send push notifications to users' smartphones.
3. The system should have a notification page that displays real-time updates on the location and status of buses.
4. The system should have a chat page that allows users to communicate with the driver and depo admin in the event of a bus replacement.
5. The system should have a chat page that allows depo admins to send warning messages to the drivers about unnecessary delay.

4. System features

4.1 System feature 1

Notification system

4.1.1 Description and Priority

This feature enables to send notifications to users. Also notify the passengers about the breakdown delay and arrival notification. Warning notification will be sent to the bus driver and depo admin. Since it is the main functionality of this application, this feature is of high priority.

4.1.2 Response Sequence

- Passenger will enable the arrival notification button.
- System will send the arrival notification to passengers.
- Driver will enable the “DELAY” button.
- System will send the delay notification to passengers.

4.1.3 Functional Requirements

- The passenger will get the arrival notification with the time and distance.
- Notify passenger while any delay occurs.

4.1.4 Non-Functional Requirements

- The driver interface is user friendly with active buttons.

4.2 System feature 2

Communication between Depo admin and Driver through chat option

4.2.1 Description and Priority

If breakdown or accident occur, driver has to report the issue by enable the "DELAY" button. Then driver has to select the possible time delay. If the repair is taking too much time, the driver should go to the chat option and send an urgent message to the depot administrator about the need for a new bus replacement. When depo administrator received the message, he response through the chat option to handle the replacement. Since it is an important safety feature, this feature is of high priority.

4.2.2 Response Sequence

- Driver will login.
- Driver will enable the "DELAY" button.
- Driver will select the possible time delay.
- Driver will go to the chat option.
- Driver will send an urgent message to the depot administrator.
- Depo admin will response through the chat option.

4.2.3 Functional Requirements

- Driver will be able to contact depo admin.
- Depo Admin will be able to send a response message.

4.2.4 Non-Functional Requirements

- The application should handle communication in a fast and stable manner.
- An on-screen notification will pop on both depo admin and driver screens.

4.3 System feature 3

Monitoring and Management of the System Administrator

4.3.1 Description and Priority

System administrator can delete and update user profiles and update the information about extra buses. System administrator will generate the reports about drivers' attendance, responsiveness of drivers to the warning messages etc. at the end of each month and send them to the depo administrator. This functionality will be given top importance because the system wouldn't be complete without the admin panel.

4.3.2 Response Sequence

- Admin would be required to enter his log in credentials on the admin log-in page.
- On the home page, the administrator will see a total of four options: Databases, turn schedule, and Log Out.
- The administrator would get hyperlinks to all the databases (passenger, bus driver, depo admin, buses, feedbacks, turn schedule) after he clicks 'Databases'.
- Each hyperlink would have a 'View' button next to it and clicking it would bring up a tabular display of all the data from that database.
- Options like Edit, Add, and Delete would be available beneath the tabular data.
- By selecting turn schedule, the administrator will see a list of the buses that are scheduled for the day, along with trip information and timings.

4.3.3 Functional Requirements

- The system shall incorporate mechanism to allow admin to authenticate its users.

- The system shall allow quick messages to be exchanged between the user and admin without face-to-face interaction.
- The system shall allow access to maps in admin's device.
- The admin would be authorized to edit data from the databases.
- The administrator would have rights to add records to the databases.
- Reasonable record deletion from databases would be permitted for the admin.
- The administrator will be able to monitor active bus turns.

4.3.4 Non-Functional Requirements

- A functional live chat-box would be given to the user where he would reach out the admin with his queries and problems.
- The administrator's system's cursor would become active, and editing would be enabled after clicking the edit button on the presentation of tabular data from the database.
- For editing purposes, the administrator would go to the search bar and search using the database's unique primary key, which would direct him to all the corresponding fields where changes could be made.
- On clicking the 'Add' button, the admin would be required to fill out all fields in a particular database.
- The admin would be able to delete some or all the fields for a certain entry by selecting the 'Delete' button.

4.4 System feature 4

Access to User (Passenger, Depo admin, Driver) Account

4.4.1 Description and Priority

Individual users will have accounts with personal information. These accounts will also be visible to system admin. Since it is one of the main functionalities of this application, this feature is of high priority.

4.4.2 Response sequence

- User will register and login.
- User will select their respective interface separately.
- Select ‘Change password’.
- System will generate an automatic ‘change password’ email and to user’s email.
- Follow the email’s instruction to change password.

4.4.3 Functional Requirements

- Change personal information as needed.
- Use chat option to solve issues with registration and login process.

4.4.4 Non-Functional Requirements

- System will generate an automated email to possible ‘change password’ requests.
- The updated information by user will be saved and updated in the system database as well.

4.5 System feature 5

Feedback and Complaint Option

4.5.1 Description and Priority

The user will be able to give feedback on all the services offered or submit a complaint about the features of web application. Since it is not the main purpose of the application, it is not given high priority.

4.5.2 Response Sequence

- User will select the Feedback option on the home page.
- The he will be given a choice between “SUGGESTION” or “COMPLAIN”.
- After choosing one of the above, he will have to choose the type and title of his suggestion or complain.
- If he chooses complains then he will have to give the details based on which he is complaining.
- User will then write his comment in a message box provided.
- Then he will submit his suggestion or complain by clicking on the submit button.
- The system forwards his complaint to the system administrator for consideration and possible response.

4.5.3 Functional Requirements

- User can submit feedback and complaints regarding all the services provided.

4.5.4 Non-Functional Requirements

- User will select the Feedback option on the home page.
- Then he will be given two options to select between suggestions and complains. He must select one to proceed to next options.
- Then he will write his comments in a comment box provided that will have the capacity of 500 characters.
- The complaint cannot be processed without the selection of the type, title (an aspect of that type), and comment.
- User will receive the confirmation message upon the successful submission of his/her complain.

5. Other Non-Functional Requirements

Nonfunctional requirements are “how well” the system should perform in one or more areas. The majority of the requirements are quality-related and fall under the categories of performance, availability, reliability, usability, flexibility, configurability, integration, maintenance, portability, and testability. Implicit requirements for modification, upgrades, reusability, and interoperability may also fall under this group.

5.1 Convenient

Creating this new system can bring a lot of convenience to passengers and staff. For example, if passenger need to know about the bus schedules, they no need to go to the bus station to look for them. Simply he can use this bus system to watch the bus schedule today via the internet. It can bring a lot convenience to the passengers who are going to the bus station. It also brings convenience to people who are busy, so they can search online at any time and can avoid wasting time. This system also brings a lot convenient to staff when they are able to search the information for customer such as search the bus available time. It also unable staff to maintain the data more easily. This system also can bring convenient to the manager such as enable manager to view the report easier and this system can use to summaries the top sale or provide the daily, monthly or yearly report and etc.

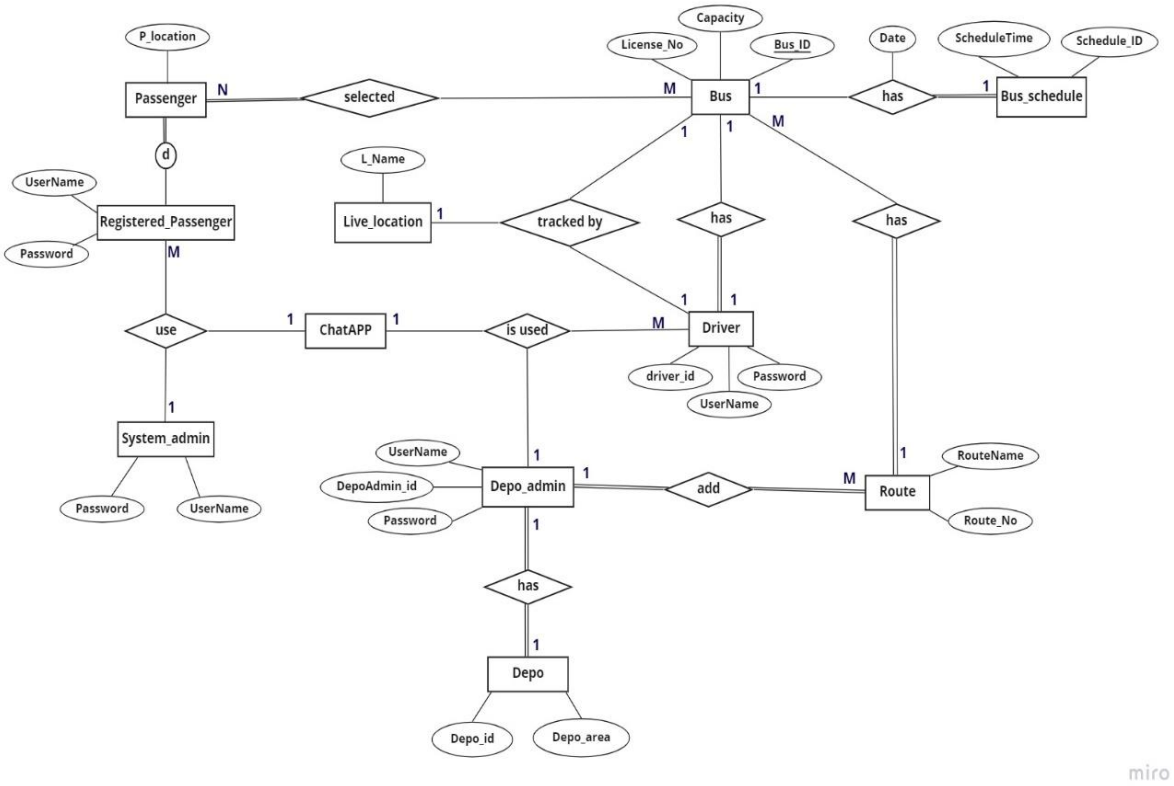
5.2 Improve efficiency

The existing system uses paper to record all the information, processes take a long time to complete. When staff members are looking for a time and a seat, serving the consumer takes a long time. This technology was developed to increase process efficiency and speed. It can enable personnel to access information more quickly and easily, manage data more quickly, and minimize paperwork

5.4 Reduce error data

The system uses a computer to input data, there is a validation process in place to detect when staff enter data incorrectly. Additionally, the system offers a selection for staff to directly select the data, allowing for a reduction in incorrect data entry. It also lessens the amount of paperwork required to record all the information, which benefits the reduction of incorrect data.

Appendix A: EER Diagram



Appendix B: Class Diagram