Mobile Based Real-Time Public Bus Tracking and Contactless Fare Payment System

A dissertation submitted for the Degree of Master of Information Technology

D.C. Liyanaarachchi
University of Colombo School of Computing
2024

Declaration

Name of the student:	Damitha Chamara Liyanaarachchi
Registration number:	2020/MIT/058
Name of the Degree Programme:	Master of Information Technology (MIT)
Project/Thesis title:	Mobile Based Real-Time Public Bus Tracking and Contactless Fare Payment System

1. The project/thesis is my original work and has not been submitted previously for a degree at this or any other University/Institute. To the best of my knowledge, it does not contain any material published or written by another person, except as acknowledged in the text.

2. I understand what plagiarism is, the various types of plagiarism, how to avoid it, what my resources are, who can help me if I am unsure about a research or plagiarism issue, as well as what the consequences are at University of Colombo School of Computing (UCSC) for plagiarism.

3. I understand that ignorance is not an excuse for plagiarism and that I am responsible for clarifying, asking questions and utilizing all available resources in order to educate myself and prevent myself from plagiarizing.

4. I am also aware of the dangers of using online plagiarism checkers and sites that offer essays for sale. I understand that if I use these resources, I am solely responsible for the consequences of my actions.

5. I assure that any work Isubmit with my name on it will reflect my own ideas and effort. I will properly cite all material that is not my own.

6. I understand that there is no acceptable excuse for committing plagiarism and that doing so is a violation of the Student Code of Conduct.

Signature of the Student	**Date (DD/MM/YYYY)**
	28/09/2024

Certified by Supervisor(s)

This is to certify that this project/thesis is based on the work of the above-mentioned student under my/our supervision. The thesis has been prepared according to the format stipulated and is of an acceptable standard.

	Supervisor 1	**Supervisor 2**	**Supervisor 3**
Name	Dr.C. R. Wijesinghe		
Signature			
Date	30/9/2024		

Abstract

This project presents an innovative approach to addressing the absence of a fully functional Mobility as a Service (MaaS) platform in Sri Lanka, with a specific focus on public bus transportation. The goal is to enhance accessibility and popularity of public transportation by creating a comprehensive mobile-based solution, incorporating real-time bus tracking and contactless fare payment.

The system features user-friendly Android mobile applications tailored for passengers, conductors, and managers, alongside a web portal for administrators. These applications enable passengers to find routes, track buses, estimate arrival times, and pay online securely. Managers will have a mobile portal for fleet management and reporting, while conductors can use a mobile app for ticket validation and emergency alerts.

A review of similar systems, including Google Transit, MyBusSL, Uber, and Moovit, was performed. The rationale behind choosing the Modified Waterfall methodology was discussed, comparing it with other methodologies and justifying its selection. System architecture and other design decisions, such as use case diagrams, class diagrams, activity diagrams, sequence diagrams, and entity-relationship diagrams, were elaborated with visual representations.

The frontend development was executed using Android Java and ReactJS, while the backend infrastructure was built with NodeJs and ExpressJS. Databases such as SQL Lite, MongoDB, and custom-built APIs were utilised, along with external services like Google Maps API, PayHere, Render, and GitHub.

Testing strategies and test design approaches, including ziBlackbox and Whitebox testing, utilised both manual and automated techniques employing TestNG and Selenium. Test cases were documented using TestCollab and Jira, and defects were managed accordingly. Finally, user evaluation was conducted, and positive feedback was received.

Overall, the project aims to rejuvenate Sri Lanka's public transportation by developing a comprehensive public transport information system exclusively accessible through widely available mobile smartphones. This initiative addresses convenience barriers that hinder the widespread usage of sustainable transportation among the general public.

Acknowledgements

First and foremost, I would want to express my sincere gratitude to Dr. M.W.A.C.R. Wijesinghe, my supervisor, whose constant direction, support, and encouragement have been the cornerstone of this journey. Her expertise, insight, and constructive feedback have shaped this work in profound ways.

To the University of Colombo School of Computing (UCSC), I offer profound thanks for providing an exceptional platform and opportunity to pursue my postgraduate degree in Information Technology. My heartfelt gratitude extends to all UCSC lecturers, visiting lecturers, and non-academic staff whose dedication has equipped us with the necessary tools to excel in the field of Information Technology.

I am also grateful to the officers of the National Transport Commission for their invaluable insights and support, despite the absence of official collaboration. Their contributions have been instrumental to the success of this project.

Special recognition goes to the volunteers who generously devoted their time and assistance to user testing. Their feedback and insights have played a pivotal role in refining and enhancing the usability of the system.

I extend my heartfelt appreciation to all the researchers and authors whose work has been cited in this project. Their knowledge and insights have illuminated the path of this research journey, enriching its depth and breadth.

Lastly, I extend my sincere gratitude to my family and friends for their unwavering encouragement and understanding throughout this endeavour. Their love, patience, and unwavering belief in me have been an endless source of motivation.

To the readers, I offer my heartfelt thanks for your interest and engagement with this work. Your attention and consideration are deeply cherished.

Table of Contents

List of Figures

List of Tables

List of Acronyms

AFC	-	Automatic Fare Card
APC	-	Automatic Passenger Counters
API	-	Application Programming Interface
APK	-	Android Application Package
APM	-	Application Performance Monitoring
AVL	-	Automatic Vehicle Location
AWS	-	Amazon Web Services
BERT	-	Bidirectional Encoder Representations from Transformers
CSS	-	Cascading Style Sheets
CUTA	-	Canadian Urban Transit Association
EBS	-	Elastic Block Store
EER	-	Enhanced Entity-Relationship
GPS	-	Global Positioning System
GSM	-	Global System for Mobile Communication
GTFS	-	General Transit Feed Specifications
HTML	-	Hypertext Markup Language
IEEE	-	Institute of Electrical and Electronics Engineers
IT	-	Information Technology
ITS	-	Intelligent Transportation Systems
JS	-	Java Script
Ltd	-	Limited
MaaS	-	Mobility as a Service
MIS	-	Management Information System
MVVM	-	Model-View-ViewModel
NLP	-	Natural Language Processing
NTC	-	National Transport Commission
OTP	-	One Time Password
PHP	-	(Personal Home Page) Hypertext Preprocessor
PVT	-	Private

QR	-	Quick Response
RDS	-	Relational Database Service
SDK	-	Software Development Kit
SIM	-	Subscriber Identity/Identification Module
SLTB	-	Sri Lanka Transportation Board
SMS	-	Short Message Service
SRS	-	Software Requirements Specification
SSL	-	Secure Sockets Layer
TLS	-	Transport Layer Security
UI	-	User Interface
UML	-	Unified Modelling Language
USB	-	Universal Serial Bus
UX	-	User Experience

Chapter 1 - Introduction

This chapter presents a concise yet engaging overview of the project, highlighting the motivation behind its development. It also outlines the problem statement that the project aims to address, along with its objectives, scope, expected outcomes and the structure of the thesis.

1.1 Project Overview

This project endeavours to tackle the absence of a fully functional Mobility as a Service (MaaS) platform in Sri Lanka by creating a mobile-based solution that encompasses real-time bus tracking and contactless fare payment. The primary goal is to increase public transit's popularity and accessibility in order to persuade people to choose environmentally friendly modes of transportation for their commuting. The project will develop an intuitive mobile application for passengers, enabling them to effortlessly locate bus routes, check real-time bus locations and estimated arrival times, and conveniently make online payments. Concurrently, bus managers will have access to a convenient mobile-based online portal to track the live locations of their buses and generate comprehensive reports on ticket incomes. Furthermore, bus conductors will be able to validate digital tickets using a simple mobile application, obviating the need for cash handling and facilitating the sending of emergency alerts in unforeseen situations. The transportation authorities will benefit from valuable insights on route and schedule optimizations, ultimately leading to a more efficient service. Throughout the system development process, a user-centered design approach will be employed, ensuring consistent evaluation against user requirements. This project aims to enhance the existing public transportation information systems in Sri Lanka and foster the adoption of public transit for a sustainable future, both economically and ecologically.

1.2 Motivation

Recognizing the fundamental importance of freedom of movement as a basic human right (*Universal Declaration of Human Rights | United Nations*, 1948), it becomes evident that transportation, which facilitates this right and grants access to essential services, intersects with various aspects of society and the environment, presenting numerous challenges. Extensive research demonstrates the unsustainability of prevailing automobile-centric transportation patterns, given their significant impacts on the environment, economy, and society (Litman and Burwell, 2006).

In the pursuit of a more sustainable mode of transportation, it is increasingly clear that public transportation holds the key. Public transportation, as emphasised by some studies (Schiller and Kenworthy, 2010), offers energy-efficient mobility in urban areas that can rival the speed of private car travel. This positions public transportation as a potential competitor to private commuting and a crucial tool in reducing dependence on cars, thereby mitigating the impacts of car-oriented systems. By considering the interconnectedness of energy consumption, pollution, and sustainable mobility, public transportation emerges as a significant provider of sustainable transportation. Evidence shows that the spatial efficiency and social benefits of transit yield positive outcomes across various sustainability criteria (Schiller and Kenworthy, 2010). These benefits encompass reduced emissions, less land consumption, improved accessibility, enhanced productivity, and contributions to economic efficiency.

Studies suggest that a shift towards public transportation can help achieve sustainable development and urbanization goals (Banister, 2008). However, this transition necessitates the reallocation of public space previously dedicated to car travel. The benefits of public transit extend beyond direct transportation services, as the Canadian Urban Transit Association (CUTA) estimates an annual $12 billion economic benefit to Canada attributed to transit (Canadian Urban Transit Association, 2014)

Furthermore, these findings hold significant relevance in Sri Lanka, where the current economic crisis, coupled with fuel shortages and rationing, has underscored the importance of relying on public transportation.

However, despite the multitude of advantages offered by public transportation, convenience remains a substantial challenge. People may lack awareness of public bus schedules or struggle to estimate travel durations, making trip planning difficult. Such perceived unreliability and infrequency can create hesitation and reluctance towards using public transportation (Beirão and Sarsfield Cabral, 2007). Based on studies, it has been found that approximately 38% of personal trips in the western province are made using buses. Similarly, a similar percentage of person trips are made using personal cars, bikes, three-wheelers, and taxis, which are considered relatively unsustainable modes of transportation (Ministry of Transport, 2014)

To address this issue, social and psychological factors have been found to play a significant role in influencing the utilization of travel information. External circumstances, specific to each trip, can also impact these factors. Typically, individuals seek public transport information before travelling, unless time constraints are absent, services are frequent, or the journey is short. Moreover, people tend to rely on consistent sources of travel information, with the Internet being the prevailing medium (Farag and Lyons, 2008). Additionally, in situations where health concerns such as pandemics arise, commuters become more conscious of crowding and seek data on vehicle density. Studies have revealed that providing accessible and straightforward crowding information can positively impact commuters' perceptions and behaviours, increasing their sense of safety and encouraging frequent use of public transit (Zhang-Kennedy *et al.*, 2023).

The motivation behind this project lies in addressing the need for accessible and widespread information on public transportation. By applying software engineering principles and adhering to usability guidelines, the project aims to develop a solution offering comprehensive public transport information to the general public. The ultimate goal is to enhance the accessibility and popularity of public transportation, thereby fostering the adoption of a more sustainable mode of travel. This, in turn, can contribute to improving the environmental, economic, and social impacts associated with transportation.

1.3 Statement of the problem

Multiple studies support the positive impacts of providing real-time arrival information for public transportation. It leads to increased satisfaction, frequency of transit trips, and reduced waiting times (Song, Guan and Ma, 2018), as well as heightened personal feelings of safety (Bian *et al.*, 2021) and a willingness to walk longer distances. These outcomes have significant implications for enhancing public transport usage, reducing traffic congestion, and minimizing the environmental footprint of transportation (Ferris, Watkins and Borning, 2010).

However, the provision of real-time transportation information faces challenges in establishing standardised data protocols, particularly with archived Intelligent Transportation Systems (ITS) transit data such as Automatic Vehicle Location (AVL), Automatic Passenger Counters (APCs), and Automatic Fare Card (AFC) data. While recent advancements like General Transit Feed Specifications (GTFS) have partially addressed this challenge through Open Data specifications, the setup process remains complicated and lengthy (Lawson *et al.*, 2019). Google Transit has been able to implement this technology in collaboration with the National Transport Commission (NTC) in Sri Lanka (*Initial project of Google Transit facility launched - Breaking News | Daily Mirror*, no date), but the provincial busses and majority of private buses are unable to utilise it due to the absence of General Transit Feed Specification (GTFS)-real-time feed.

The NTC introduced the "MyBus-SL" mobile application for real-time journey information on SLTB (Sri Lanka Transportation Board) buses, but it only covers around 1100 interprovincial busses (MyBus-SL mobile app launched for the convenience of public transportation - NewsWire, no date) and it makes only around 4% of total busses operated in Sri Lanka (National Transport Commission, no date). Private bus operators hesitate to invest in Global Positioning System (GPS) devices, which hinders their engagement with the system. Furthermore, neither of these systems offers an online bus ticket purchase option for short-distance travel.

Although the NTC launched the "Touch Travel Card" as part of an e-ticketing concept (*E-Pay system for Bus Travel*, no date) it is only available on a few routes in specific provinces (Touch Travel, no date).

These problems ultimately led to a lack of public interest in these applications and a failure to develop a system that might make using public transit more convenient and effective.

1.4 Aims and Objectives

The primary goal of this project is to create a mobile-based system that enables real-time bus tracking and contactless fare payment. The aim is to offer the general public comprehensive information on public transportation, enhancing its accessibility and popularity. By encouraging the adoption of a more sustainable mode of transportation, this initiative can have a positive impact on the environment, economy, and society, associated with transportation. To reach this goal following objectives are expected to be achieved.

1. To develop a mobile-based (Android) platform for public buses, to share real-time locations with passengers, eliminating the need for additional GPS devices.

2. To create a user-friendly mobile (Android) application with intuitive interfaces that enables people to easily search for available routes, nearby buses, and their live locations, while also providing estimated travel times.

3. To implement a feature that allows buses to send emergency alerts to pre-set contacts in case of an emergency.

4. To facilitate seamless payment options and secure online transactions, reducing reliance on cash handling for bus conductors and passengers, and promoting environmental sustainability by minimizing paper ticket usage.

5. To generate reports for the bus managers and administrators that provide valuable insights and support decision-making, which can be used for demand forecasting, route planning and optimization, and operational management.

6. To ensure the integrity, consistency, availability, and security of user data and system data, prioritizing data protection and privacy.

1.5 Scope of the study

The project aims to develop a comprehensive Mobile-based (Android) application system with a web portal. The system will be divided into four parts, catering to passengers, bus managers, bus conductors, and system administrators. The table below (Table 1.1) outlines the system's main components, along with their intended users and functionalities.

Table 1.1: Main parts of the system and their functionalities

System Part	Functionalities
Mobile application edition – for passengers	1. Register to the system and login through their smart phones 2. Search for bus routes and fares for any destination in Colombo district (limited due to feasibility) 3. Get estimated time duration for the selected destination 4. Get live location and estimated time of arrival of a selected bus on the road 5. Make payment for the bus fare and display a virtual bus ticket with a QR code
Mobile application edition – for bus managers	1. Register to the system and login through their smart phones 2. Create a new account for their fleet of buses and undergo verification by the National Transport Commission by providing necessary business details 3. Create and manage sub-accounts specifically assigned to each bus under their management 4. Authorize access for bus conductors by verifying their email addresses. 5. Set emergency contacts for each sub-account 6. Generate and view reports pertaining to trip and payment data
Mobile	1. Register to the system and login through their smart phones

application edition – for conductors	2. Share the live location of the bus with the web server, after starting a journey until to the end of journey 3. Send alerts to pre saved contacts during in an emergency such as an accident or mechanical failure 4. Allow conductor to verify a virtual ticket presented by a passenger by scanning the QR code
Web portal – for system administrators	1. Login to the system administration dashboard through its web portal 2. Authorize and manage bus manager accounts 3. Create and update routes and fare tables 4. Generate management reports for authorities

For these functional requirements, the application system will need the following modules, each with its respective functionalities outlined in Table 1.2.

Table 1.2: Main modules of the system and their functionalities

1	Administrator Module
1. Provides admins an extensive dashboard for system management. 2. Allows admins the option to add and modify routes and tariffs inside the system. 3. Allows administrators to review and confirm the legitimacy of bus managers' registration requests, then approve or deny them as necessary. 4. Enables administrators to terminate bus manager accounts that are registered if required. 5. Able to generate reports on transactions and fares, bus routes and stops, and fleets.	
2	Authorization Module
1. Controls access privileges for account creation and login for each component of the application. 2. Verifies each user's email address.	

3. Offers a panel where bus owners and managers can register and create an account that the National Transport Commission will review. 4. Allows the creation of sub accounts and grants bus conductors access.	
3	Reporting Module
1. Allows administrator and bus managers to generate reports for trip and transaction data.	
5	Payment Module
1. Allows passengers to purchase a virtual ticket by a preferred payment method. 2. Verify virtual tickets by scanning QR codes presented by a passenger. 3. Crediting verified ticket payments to relevant bus account. 4. Refund unused ticket payments to passenger account/bank account after a specific number of days.	
6	Map Module
1. Displays google map. 2. Allows passengers to select a starting point and a destination which will display an optimized route with estimated time durations.	
8	Live Location Module
1. Sends live location coordinates of a bus account to the web server while a journey is going on. 2. Retrieve live location coordinates of a selected bus and display it on a passenger account.	
9	Emergency Notification Module
1. Allows bus account to send alerts to pre specified contacts. 2. Make live location sharing off and notify the web server.	

In the scope of the project, there are some assumptions made to achieve objectives.

- Users would have compatible Android phones (Minimum SDK version: API level 21 / Android 5.0 Lollipop).
- Mobile connection is satisfactory to provide more accurate and quick data.
- Bus conductors will adhere to the proper procedures for starting and ending journeys.

Due to time constraints and feasibility concerns, certain limitations have been imposed on the scope of the project. These limitations include:

- The mobile-based system will be developed exclusively for the Android operating system. Due to limited testing time and resources, support for other platforms was not achieved.

- The system's availability will be limited to public transport information within the Colombo District. This decision is based on feasibility considerations, focusing on a specific geographic area to ensure a more manageable implementation process.

- The system will be available only in the English language. Support for additional languages is not included within the scope of the project due to time constraints.

1.6 Expected outcomes

The project is expected to deliver a complete mobile-based system for real-time public bus tracking and a contactless fare payment system with the following components.

1. Mobile application for passengers developed in Android Java.

2. Mobile application for bus conductors developed in Android Java.

3. Mobile application for bus managers developed in Android Java.

4. Web application for administrators built on ReactJS.

5. Backend server constructed using NodeJS and MongoDB.

As of now, there is no other functional system in Sri Lanka that provides mobile-based real-time location data of public transport buses via mobile platforms, except for a limited number of buses registered with Google Transit and MyBus-SL systems. Similarly, there is a lack of mobile-based systems facilitating online fare payment for public transport in the country. Hence, the proposed system emerges as a pioneering solution to address these gaps, with the potential for implementation across all public buses. It presents a novel and much-needed solution to enhance public transportation services in Sri Lanka.

1.7 Structure of the thesis

The thesis will be organized into six chapters, each focusing on the different stages of the project, following the Software Development Life Cycle. The chapters will provide comprehensive details, accompanied by appropriate figures, tables, and reports, to facilitate a thorough understanding of the project's work.

This chapter will provide a basic overview of the project, including the underlying motivation behind its development. It will define the problem being addressed and outline the scope of the project. The chapter aims to familiarize the reader with the project's objectives and set the context for the subsequent chapters.

Chapter 2: Background

This chapter will conduct a comprehensive review of existing systems that are similar to the proposed system in the thesis. It will analyse and evaluate these systems, highlighting their strengths, weaknesses, and limitations. Additionally, the chapter will describe the selected development process model, justifying its selection based on the requirement analysis conducted.

Chapter 3: Design

In this chapter, the design methodology, technologies, and tools employed in the project will be thoroughly discussed. It will delve into the details of the design process, including design diagrams and interface designs. Alternative solutions will be considered and discussed, providing a rationale for the approach ultimately adopted.

Chapter 4: Implementation

This chapter will explain the major code and module structures, providing diagrams and justifications for the selection of the implementation environment, encompassing both hardware and software aspects.

Chapter 5: Testing and Evaluation

In this chapter, the testing strategies and tools used for the system will be discussed. It will present comprehensive test plans, test procedures, and test results, ensuring the thorough evaluation of the system's functionality, reliability, and performance. Usability evaluations will also be described comprehensively, focusing on the user experience aspects of the system.

Chapter 6: Conclusion

The final chapter will provide an overview of the project's outcomes. It will include a general appraisal of the work done, highlighting the achievements and contributions of the project. Additionally, the chapter will present prospects for future work, suggesting potential avenues for further improvement and development.

Chapter 2 – Background

In this chapter, a comprehensive discussion of the background research conducted at the outset of the project will be presented. A detailed analysis of the system requirements will be provided, outlining the specific functionalities and features that the system aims to fulfil.

The chapter will also include an in-depth review of existing systems that are similar to the proposed system. These reviews will critically assess the strengths and weaknesses of the existing systems, highlighting any gaps or limitations that the proposed system seeks to address. Furthermore, the technologies utilised in both the existing systems and the proposed system will be examined. A comparative analysis will be conducted, evaluating the advantages and disadvantages of each technology. The rationale behind the selection of the proposed design strategies will be thoroughly justified based on this comparison.

2.1 Requirement gathering

To identify the requirements of the proposed system, a thorough analysis of the stakeholders' needs and expectations was conducted. The main stakeholders involved in the project were identified, including passengers, bus conductors, bus managers, system administrators, and transportation authorities.

To gather the requirements from passengers, observations of their interactions with public transportation were carried out. Additionally, a sample group of passengers was interviewed using questionnaires to gain further insights into their expectations and requirements for a proposed system.

The requirements of bus conductors and bus managers were gathered through interviews with a group of individuals in these roles. By engaging directly with them, their specific requirements and pain points were identified

To gather additional information and insights, various documents were consulted. Annual reports and National Transport Statistics reports published by the National Transport Commission (NTC) were reviewed to gain insights into existing policies, pilot projects, and fund allocations related to electronic services in public transportation. Further clarification and information were obtained by conducting unofficial interviews with NTC officers via telephone.

By utilizing a combination of observations, interviews, questionnaires, and document analysis, the requirements of the stakeholders were thoroughly gathered. This process ensured that the proposed system would be designed to meet their needs and expectations effectively.

2.2 Requirement analysis

2.2.1 PACT Analysis

The gathered information on the requirements of different stakeholders for the proposed system was analysed using a PACT (People, Activities, Context, and Technologies) analysis. This analysis aimed to facilitate the design process by providing a comprehensive understanding of the various aspects of the system's use.

2.2.1.1 People

1. *Physiological differences:* The design of the interface was planned to accommodate a range of visual abilities. Font types, sizes, and colours were carefully selected to ensure readability for users with different levels of eyesight. Accessibility features such as visual enhancements and narrator features were also considered to cater to users with specific needs.

2. *Psychological differences:* Factors like memory span, were taken into consideration during the design process. Feasibility for different language options and additional sign-in options, such as biometrics, were examined. These considerations aimed to accommodate users with varying memory capacities and provide them with convenient and accessible options to enhance their user experience.

3. *Social differences:* The passenger edition of the application is intended for a heterogeneous group of users. To cater to both expert and novice users, flexibility in the

interface design was incorporated. Tutorials on how to use the application were included to assist users in familiarizing themselves with the system and provide support as they navigate through the features.

4. *Mental models:* The user interfaces of existing similar systems, such as Google Transit, MyBus-SL, and Uber Taxi, were thoroughly analysed. By studying these interfaces, valuable insights were gained to create effective and intuitive user interfaces that align with users' mental models. This ensures that the proposed system is familiar and easy to use for users who are already accustomed to similar applications.

2.2.1.2 Activities

To ensure ease of use for users, various aspects of their activities were taken into consideration during the design process:

1. *Temporal Aspects:* The temporal aspects of different activities were considered. For one-time activities, such as signing up and linking payment methods, clear guidance was provided since they are infrequent and more prone to being forgotten. On the other hand, more frequent activities, such as searching for bus locations, were designed to be simpler and easier to perform.

2. *Response time:* The average response time of users was considered. Activities that need to be completed at once, like the steps for buying virtual tickets, were linked together, allowing users to seamlessly continue from where they left off if their progress was interrupted. Messages for delays were also taken into consideration when designing user interfaces, ensuring that users are informed about any delays and can adjust their activities accordingly.

3. *Complexity:* The complexity of activities was taken into account. More complex activities were broken down into manageable steps, guiding users through each stage to ensure a clear and straightforward process.

4. *Safety-critical activities:* Safety-critical activities, such as making payments, were

designed with precautions to prevent serious mistakes. Alerts for important actions and the ability to reverse them within a specified time period were considered to minimize the impact of potential errors and ensure user safety.

5. *Data and media requirements:* The data and media requirements of activities were considered to optimize performance. The use of videos and large images was restricted as much as possible to avoid excessive data usage. Additionally, the data transfer between the mobile application and servers was reconsidered to prioritise speed and efficiency.

2.2.1.3 Context

1. The *organizational and social contexts* were thoroughly analysed to refine the requirements of the system. Considerations were made for the daily tasks of bus managers and system administrators, including their ability to work from different devices. This analysis guided the design of the authorization process and interactions to align with their specific needs and workflows.

2. In unexpected situations such as accidents or breakdowns, the system will have the capability to send alerts to emergency services and display them on passengers' devices, ensuring their safety and providing necessary information.

3. GPS connection failures, whether due to technical or physical reasons, were considered. Special algorithms will be implemented to predict and correct errors in live location data, ensuring accuracy and reliability even in challenging GPS conditions.

4. The *physical environment* in which different groups of users interact with the system was also considered. For example, bus conductors will be standing up in the bus when scanning virtual tickets from passengers. Factors such as the visibility and orientation of the QR code on the virtual ticket were taken into consideration during the design phase to optimize the scanning process and enhance convenience for both conductors and passengers.

2.2.1.4 Technologies

1. The *input methods* for the system, such as keyboards and cameras, were carefully examined to accommodate the technological differences among users' devices. Scalability and the potential for additional input methods, such as voice commands, in the future were also considered and will be assessed for feasibility.

2. Similarly, the *output devices* were analysed, and currently, the system is planned to have the display screen as the primary output method. However, the flexibility to incorporate sound output will be considered, taking into account its feasibility and user requirements.

3. The *communication technology* used in the system was thoroughly evaluated, considering the amount of information that needs to be transferred between different components of the system. The chosen communication methods should be able to handle the data load efficiently and ensure smooth interaction between the central database and peripheral user devices.

4. Validation and authorization methods were also considered, particularly regarding the communication between the central database and user devices. Secure and reliable protocols will be implemented to ensure the integrity and confidentiality of data during the validation and authorization processes.

By considering the people involved, the activities they perform, the context in which the system will be used, and the technologies available, the design process can be aligned with the users' requirements and ensure a high-quality human-computer interaction.

However, it is important to note that all the identified requirements from the stakeholders will undergo a feasibility analysis to assess the practicality and viability of incorporating these requirements into the system design.

2.2.2 Feasibility analysis

After identifying the requirements of all stakeholders, conducting a feasibility study is essential to evaluate the extent to which the proposed solution meets these requirements and to assess the overall viability and practicality of the system. The feasibility study helps determine if the proposed system is technically feasible, economically viable, and operationally achievable.

- Operational Feasibility: The proposed system aligns well with the existing infrastructure, resources, and human capital available within the National Transport Commission. The organization has allocated a significant budget for information operations and has a well-equipped IT department capable of supporting the system. Additionally, the organization has shown political support and plans for implementing such systems (National Transport Commission, 2020). This system is easy to use and does not require special training.

- Technical Feasibility: The system's technical feasibility is supported by the increasing availability of smartphones with GPS tracking, secure payment gateways, and faster connection capabilities among the general public. The number of cellular mobile numbers in Sri Lanka has exceeded 30 million, indicating a substantial increase. Additionally, the count of mobile broadband connections has surpassed 19 million. These figures are particularly significant when considering the country's population size (Department of Census and Statistics Ministry of Finance, 2022). Cloud services can be utilised to ensure faster and more reliable data transfers, maintaining data integrity and consistency (National Transport Commission, 2020).

- Cultural Feasibility: Although previous attempts to implement similar systems have faced challenges, there is a growing acceptance and demand for such systems among passengers, as evidenced by their familiarity and positive experiences with similar systems used for other daily purposes, such as fuel purchasing through QR systems.

  It is identified that bus managers and conductors see the benefits of the proposed system in terms of simplifying their daily tasks. However, there may be some resistance from bus conductors who fear that their jobs may be replaced by the system. Reassurance is necessary to address these concerns and to emphasise that the system is designed to complement their roles rather than replace them. The simplicity of the user interfaces and the ease of training required for bus conductors to operate the application should help in gaining their acceptance.

- Economic Feasibility: The allocated funds for information development within the National Transport Commission support the operational costs associated with system maintenance, updates, server hosting, and support. The adoption of the proposed system is expected to result in cost savings through increased passenger counts, reduced cash handling, and important data-driven reports for future bus route optimization. Additionally, intangible benefits such as customer satisfaction and organizational reputation are considered valuable.

- Legal Feasibility: The proposed system aligns with transportation regulations governing activities such as location sharing and the use of electronic ticketing machines. The Central Bank of Sri Lanka has introduced financial regulations that govern payment methods. Since the proposed system involves electronic payments, it is important to ensure compliance with these regulations. The National Transportation Commission has conducted a formal procurement process to select a suitable institution for developing the relevant software for the system. This demonstrates the commitment to following legal procedures and obtaining the necessary software through an official and transparent process (National Transport Commission, 2022).

  Additionally, this system adheres to copyright laws both locally and internationally.

- Scheduling Feasibility: While the main functional requirements of the system are feasible, some features identified in the PACT analysis may need to be curtailed due to time constraints.

2.2.2.1 Alternative Solutions (Candidate system analysis)

In addition to analysing the feasibility of the above categories, the feasibility of the proposed solution should be assessed by comparing it with alternative solutions. For feasibility analysis, three candidate systems can be identified:

Candidate System 1: A standalone mobile application for passengers that stores predetermined bus schedules.

Candidate System 2: A web-based application for passengers that displays bus locations obtained from third-party GPS devices installed in public buses.

Candidate System 3 (Proposed Solution): A mobile application system enabling real-time bus location sharing from the conductor app and real-time bus location reception on the passenger app.

These candidate systems will be compared based on user requirements, reliability, cost, security, scalability, and the availability of development tools in the following table (Table 2.1).

2.1: Candidate system feasibility analysis

	Candidate system 1	Candidate system 2	Candidate system 3
Route planning	Not available	Available	Available
Real-time tracking	Not available	Available	Available
Digital Ticketing	Not available	Not available	Available
Emergency notification	Not available	Not available	Available
Report generation	Not available	Not available	Available

Reliability	Low	High	High
Cost	Low	High (To purchase third party GPS devices)	Medium
Security	Low risk	Medium risk	High risk
Scalability	Low	Medium	High
Availability of development tools	High (Basic mobile app development tools will suffice).	Medium (Depends on the development or procurement of GPS devices).	High (Open-source tools are available).

Considering the above analysis, it is evident that the proposed solution fulfils all user requirements, whereas the other two candidate systems fall short in this aspect. Additionally, factors such as reliability, scalability, and availability of development tools are higher in the proposed solution compared to the alternatives. Moreover, the cost of the proposed solution is relatively lower than that of candidate system two. Therefore, taking all criteria into account, the proposed solution emerges as the better choice.

Based on the findings of the feasibility study, it is deemed appropriate to proceed with the development and implementation of the proposed system. The study indicates that the system is technically feasible, economically viable, operationally compatible, and legally compliant. Furthermore, it has demonstrated positive potential benefits, ensuring better fulfilment of both functional and non-functional user requirements compared to the alternatives.

2.2.3 System features

The feasibility study results have confirmed the following functional and non-functional requirements, which have been documented by adapting the IEEE SRS (Software Requirements Specification) document template.

2.2.3.1 Passenger edition

User Registration

- The system will have the capability to publish the mobile application (passenger edition) on the Google Play Store for download.
- The system shall enable any person who has downloaded the application to create a personal account by providing their details such as email, name, and password.
- The system will verify the email address provided by the user to ensure it belongs to them.

Login to the passenger account

- The system shall allow users to enter their accounts using valid email and password.
- The system shall allow users with valid credentials to log in.
- The system shall not allow a user with invalid credentials to log in.
- The system shall allow the valid user to reset the password if forgotten.

Logout of the system

- The system shall allow the user to log out of the existing login session.

Search destinations and view routes

- The system shall allow the user to enter a location in the search bar.
- The system shall display the search results that match any keyword or part of the keyword.
- The system shall return “no destination found” if a keyword or part of the keyword does not match.
- The system shall allow users to select any destination in the search results.
- The system shall provide users with the option to select the starting location by searching for a keyword. Alternatively, if the user does not specify a starting location, the system

shall default to the current location of the device.

- The system shall display the route from the starting point to the destination point on a map.

Track live locations of the busses

- The system shall dynamically display the real-time locations of buses that are currently in service and travelling along any portion of the selected route direction on the map.
- The system shall enable users to select a specific bus by interacting with its corresponding marker on the map. Upon selection, users will be presented with detailed information about the selected bus, including its estimated arrival time to the next intersecting point along the route, vehicle number, colour, and a "purchase ticket" button.

Digital ticket purchasing

- The system shall provide users with the capability to purchase of a digital bus ticket.
- The system shall prominently display the ticket price along with the duration for which the digital ticket will be valid. Users will be able to see the amount of money required to purchase the ticket and the duration of its validity period.
- The system shall allow users to purchase the ticket or cancel the process.
- The system shall allow user to select a valid payment method and enter their credentials.
- The system shall integrate with a payment gateway to verify the payment.
- The system shall reserve the ticket if payment is successful.
- The system shall inform the user if the payment is unsuccessful.
- The system shall allow users to check the purchased tickets and their validity.

Digital ticket validation

- The system shall allow users to open any valid ticket and display its QR code.
- The system shall check the validity of the ticket when the conductor application scans a QR code.
- The system shall display a success message on both passenger and conductor applications if the ticket is valid.
- The system shall display an error message on both passenger and conductor applications if the ticket is invalid and give the option for the passenger to purchase a ticket manually.

- The system shall transfer the amount of the ticket to the conductor's account if the validation successful.

2.2.3.2 Conductor edition

User Registration

- The system will have the capability to publish the mobile application (conductor edition) on the Google Play Store for download.
- The system shall enable any person who has downloaded the application to create a personal account by providing their details such as email, name, and password.
- The system will verify the email address provided by the user to ensure it belongs to them.
- The system will grant access to a bus account if the email address is assigned to any bus account by a bus manager.
- The system shall prevent the access if the email address authorization is revoked by the bus manager.

Login to the conductor account

- The system shall allow users to enter their accounts using a valid email address and password.
- The system shall allow users with valid credentials to log in.
- The system shall not allow a user with invalid credentials to log in.
- The system shall allow the valid user to reset the password if forgotten.

Logout of the system

- The system shall allow the user to log out of the existing login session.

Start and end journey

- The system shall allow users to start journey by clicking a button.
- The system shall allow users to end journey by clicking a button.

Digital ticket validation

- The system shall allow users to scan a digital ticket QR code.

- The system shall then check the validity of the scanned QR code.
- The system shall display a success message with digital ticket details if the ticket is valid
- The system shall display an error message if the ticket is invalid.
- The system shall transfer the amount of the ticket to the conductor account if the validation is successful.

Send emergency alerts

- The system shall allow users to send emergency alerts in a situation where the journey is interrupted.
- The system shall send an emergency alert to pre-set contact with the last location.

2.2.3.3 Bus manager edition

User Registration

- The system will have the capability to publish the mobile application (manager edition) on the Google Play Store for download.
- The system shall enable any person who has downloaded the application to create a personal account by providing their details such as email, name, and password.
- The system will verify the email address provided by the user to ensure it belongs to them.
- The system shall allow the user to enter details such as name, email, bus fleet registration number and national identity card number.
- The system shall verify the entered details by sending a request to register to the system administrator.
- The system shall verify the user account as a bus manager account if the verification is successful.
- The system shall terminate the access to the bus fleet data if the verification is unsuccessful and will display the contact details for further inquiry.

Login to the conductor account

- The system shall allow users to enter their accounts using valid email address and password.

- The system shall allow users with valid credentials to log in.
- The system shall not allow a user with invalid credentials to log in.
- The system shall allow the valid user to reset the password if forgotten.

Logout of the system

- The system shall allow the user to log out of the existing login session.

Manage sub-accounts

- The system shall allow user to create sub-accounts with vehicle registration number, route number, route name, bus conductor email address and emergency contact details.
- The system shall allow the user to update or delete sub-accounts.

Track live locations of the busses

- The system shall get updated with the live locations of the busses (sub-accounts) that have started the journey.
- The system shall enable users to select any marker representing a bus on the map and view its details, including the departure time, estimated arrival time at the destination, and the validated ticket amount.

Generate report

- The system shall allow users to generate a report on validated tickets and trip data for a selected duration for any particular sub-account.

2.2.3.4 System administrator web portal

Login to the admin account

- The system shall allow users to enter their accounts using valid usernames and password.
- The system shall allow users with valid credentials to log in.
- The system shall not allow a user with invalid credentials to log in.
- The system shall allow the valid user to reset the password if forgotten.

Logout of the system

- The system shall allow the user to log out of the existing login session.

Create and update routes

- The system shall allow users to create new routes by entering route number and name.
- The system shall allow users to update existing routes.

Manage fare table

- The system shall allow users to update the fare table.

Manage bus manager accounts

- The system shall display registration requests sent by bus managers.
- The system shall allow the user to verify those requests.
- The system shall allow users to terminate bus manager accounts.

Generate report

- The system shall provide users with the capability to generate reports on the number of validated tickets and trip data for a selected duration about a specific route.

2.2.4 Non-functional requirements

2.2.4.1 Performance

Response Time

- The system shall provide real-time bus tracking information with a response time of no more than 5 seconds for retrieving and displaying the current bus location and estimated arrival times.
- The system shall generate digital tickets and process contactless fare payments with a response time of no more than 5 seconds.

Network Latency

- The system shall be designed to minimize network latency for data retrieval and communication with backend services.
- The average network latency for retrieving real-time bus tracking information and processing fare payments shall not exceed 500 milliseconds.

2.2.4.2 Scalability

- The system shall be capable of handling a minimum of 1,000 simultaneous user requests for bus tracking and fare payment without significant performance degradation since its around 1350 trips per minute during a Peak Hour to the Colombo Municipal Council (Ministry of Transport, 2014).
- The system shall scale up to accommodate an increasing number of users, aiming to maintain response times within acceptable limits.

2.2.4.3 Security

- The system shall ensure secure communication and data transmission between the mobile application, backend servers, and payment gateways to protect user information and prevent unauthorized access.
- The system shall comply with industry standards and regulations for data security, encryption, and user privacy. Therefore, the system will not store user passwords as plain text and will always encrypt them before storing.
- The system shall implement password expiry and password lock policies to enhance security.
- The system shall support two-factor authentication as an additional layer of security during login.

2.2.4.4 Availability

- The system shall have a high level of availability, minimizing downtime and ensuring that bus tracking and fare payment services are accessible to users at all times.
- The system shall have a backup and disaster recovery plan in place to handle unexpected outages and quickly restore services.

2.2.4.5 Interoperability

- The system shall be compatible with a variety of smart mobile devices ensuring broad accessibility for users.

2.3 Review of Similar Systems

Before conducting a comprehensive review of a similar system, it is important to briefly review the existing systems in Sri Lanka that have announced similar features to the proposed system (see Table 2.2). While these systems may have some similarities, it is noted that they are currently non-functional or not widely available for a significant portion of public transportation.

Table 2.2: Review of Google Transit and MyBus-SL

System	Features	Limitations
Google Transit (*Google Maps - Apps on Google Play*, no date)	1. Integrated with Google Maps 2. Search for public transit routes and schedules for busses and trains 3. Real-time transit information 4. Estimation of arrival time and destination 5. Available in all 3 languages	1. Limited number of public buses providing real-time location data 2. Majority of private buses not linked with the system 3. Lack of fare details and online ticket purchasing feature 4. Buses require the installation of a separate GPS device to track their locations 5. Lengthy and complicated process to register with GTFS-real-time feed
MyBus-SL (*mybus-sl - Android Apps on Google Play*, no date)	1. Real-time bus tracking 2. View schedules, routes and fares 3. Seat reservation 4. Available in all 3 languages	1. Limited number of buses registered with the system 2. Inability to search for starting and ending locations 3. Limited destinations available for seat reservation 4. Lack of online ticket purchasing feature 5. Buses require the installation of a separate GPS device to track their locations

Although the applications, such as mTrack by Mobitel (PVT) Ltd (*mTrack | Mobitel*, no date), Uber Ride and Uber Drive by Uber Technologies, Inc., and PickMe Ride and PickMe Driver by Digital Mobility Solutions Lanka (PVT) Ltd, are not directly similar to the proposed system as they do not provide information on public buses, it is still valuable to review their features briefly. These applications offer fleet management services and taxi rides in Sri Lanka, respectively.

Uber and PickMe are popular taxi services in Sri Lanka that primarily operate in the Western province and cater to tourist destinations. While these services do not provide information on public buses, it is still relevant to briefly review their features, particularly in terms of mobile phone-based GPS tracking and online payment methods. Here is a comparison of their features and limitations outlined in Table 2.3:

Table 2.3: Review of mTrack, Uber and PickMe applications

System	Features	Limitations
mTrack (*mTrack*, no date)	1. GPS based live vehicle tracking 2. Support GSM SIM with SMS and GPRS communication capability 3. Fuel monitoring facility 4. Ability to store and forward location info in case of Low or no network reception 5. Web interface for fleet admins to check details on their vehicles 6. Track and store movement history 7. Ability to generate reports	1. Does not offer features related to public bus routes, schedules, or ticketing. 2. Tracking is only available for the administrator 3. requires the installation of a separate GPS device to track their locations 4. Only available in English language 5. Web interface is relatively complex and less user friendly for novice users 6. Not having a mobile application

Uber Ride (*uber - Android Apps on Google Play*, no date) & Driver (*Uber - Driver: Drive & Deliver - Apps on Google Play*, no date) PickMe Ride (*pickme - Android Apps on Google Play*, no date) and Driver (*PickMe Driver (Sri Lanka) - Apps on Google Play*, no date)	1. Get estimated fare for a ride 2. Ability to choose from different vehicle categories 3. Find nearest available driver 4. Track selected driver's location and estimated time of arrival in real-time 5. Ability to make calls to driver or riders 6. Make online payments with choice of multiple payment methods 7. Book longer outstation trips 8. Navigation for drivers 9. Track drivers' payments and transfer to local banks 10. Ability to rate riders as well as drivers 11. Available in all 3 languages (PickMe)	1. Does not offer features related to public bus routes, schedules, or ticketing. 2. Limited only to western province for short distance rides 3. Demand is high in peak hours so the limited availability 4. Pricing will surge during peak hours and other times of high demands as festivals 5. Responsibility is upon a private company so the regulation is limited 6. Available in only in English language (Uber)

Upon examining the aforementioned applications available in Sri Lanka, it is evident that they have limited features compared to the proposed system. To further evaluate a system more similar to the proposed system, but not currently accessible in Sri Lanka, a comprehensive review of Moovit will be conducted. Moovit is a mobility-as-a-service provider and journey planner application based in Israel. Since 2020, it has been owned by Intel through its subsidiary Mobileye. The following (Table 2.4) is a summary of the features and limitations of the Moovit application (*Moovit: MaaS Solutions & the #1 Urban Mobility App*, no date).

Table 2.4: Review of Moovit application

System	Features	Limitations
Moovit (*moovit - Android Apps on Google Play*, no date)	1. Real-time bus tracking 2. Trip planning with real-time data 3. Multiple transit modes coverage 4. Contactless fare payment options 5. Service alerts and notifications 6. Crowdsourced data for accuracy	1. Not available in Sri Lanka and many other regions in the world 2. Occasional data discrepancies or delays 3. Availability of features depends on transit agency cooperation and integration 4. Potentially overwhelming user interface

Moovit encountered the task of rapidly expanding its infrastructure to accommodate a growing user community and extend its service coverage. Within a mere four months, Moovit attracted an impressive 100,000 users. In order to meet the demands of scalability, availability, and efficient data management, Moovit turned to Amazon Web Services (AWS) for a comprehensive and dependable solution. AWS possessed the capability to handle high volumes of requests and diverse data formats, making it the ideal choice for Moovit.

To address their requirements, Moovit leveraged various AWS services. Elastic Load Balancing uses geography to route requests, Amazon CloudWatch monitors server health and manages loads, Amazon CloudFront provides fast content delivery, Amazon (Elastic Compute Cloud) EC2 provides application servers, and Amazon CloudWatch stores temporary processing files. Moovit relied on Amazon Relational Database Service (Amazon RDS) and Amazon DynamoDB to store distinct data sets, including schedules, user profiles, and geographic information. Additionally, Amazon Simple Storage Service (Amazon S3) was employed to store tiles and reporting logs. Moovit's server code was predominantly written in Java, JavaScript, HTML, Python, and Oracle's web framework.

Through the utilization of AWS, Moovit experienced remarkable enhancements in performance and availability. With the capacity to process a staggering 85 million requests per day, Moovit seamlessly provided real-time information to users, enabling them to effortlessly check arrivals, plan trips, and access public transportation schedules. AWS's reliability and scalability ensured uninterrupted service availability, catering to the needs of a global user base all day, every day, throughout the year.

In addition to providing the necessary infrastructure, AWS also delivered cost savings and operational efficiency. In comparison to traditional hosting solutions, AWS eliminated the need for significant investments in equipment and large operations teams. Moovit achieved substantial cost reductions by managing all AWS hosting services with just a single person, while simultaneously expanding its services worldwide. (*AWS Case Study: Moovit*, no date)

Moovit's urban mobility app incorporates a remarkable feature, providing worldwide app users with access to transit service alerts sourced from local operators and agencies. These alerts comprise text messages that describe changes in public transit service, which can be either positive or negative. Typically conveyed in lengthy textual formats by the operator, these alerts require analysis to determine their potential impact on a user's trip plan. Accurate classification of these service alerts is crucial, as it influences the manner in which transit recommendations are presented in the app. Incorrect classification could lead users to overlook significant service interruptions that may disrupt their planned journeys.

To address this challenge, Moovit has developed an automated pipeline using Apache Airflow and Amazon SageMaker to train and deploy BERT models. BERT, a neural network framework for natural language processing (NLP), plays a vital role in enhancing the accuracy and coverage of service alerts across multiple metropolitan areas within the Moovit app.

Service alerts hold immense significance in Moovit's urban mobility app, serving as a means to inform users about changes in public transit service. However, accurately classifying these alerts presented a considerable obstacle. Initially, Moovit employed a combination of rule-based classification and manual human classification, but this approach proved to be non-scalable and resulted in gaps in coverage.

To overcome this challenge, Moovit harnessed the power of the BERT model—a bidirectional encoder representation from transformers. By fine-tuning the BERT model using classified data from the rule-based engine, Moovit successfully created an NLP-based classification solution capable of accurately categorizing service alerts.

Following the model's training, Moovit deployed it to an Amazon SageMaker endpoint, enabling real-time inference for service alerts. This separation of model serving and deployment from the backend release schedule facilitated more frequent updates, enhancing responsiveness. The capabilities of SageMaker played a vital role in automating the training and deployment process, minimizing the need for extensive engineering time and infrastructure management.

Furthermore, Moovit leveraged Apache Airflow to establish a comprehensive workflow encompassing dataset generation, model training, verification, and deployment. Known as the "AI lake," this workflow (illustrated in Figure 2.1) proved instrumental in achieving scalability and expansion into new metropolitan areas. With the integration of the AI lake solution and the service alert classification model, Moovit experienced a significant increase in the percentage of classified service alerts. This improvement contributed to an enhanced user experience and operational efficiency.

(*How Moovit turns data into insights to help passengers avoid delays using Apache Airflow and Amazon SageMaker | AWS Machine Learning Blog*, no date)

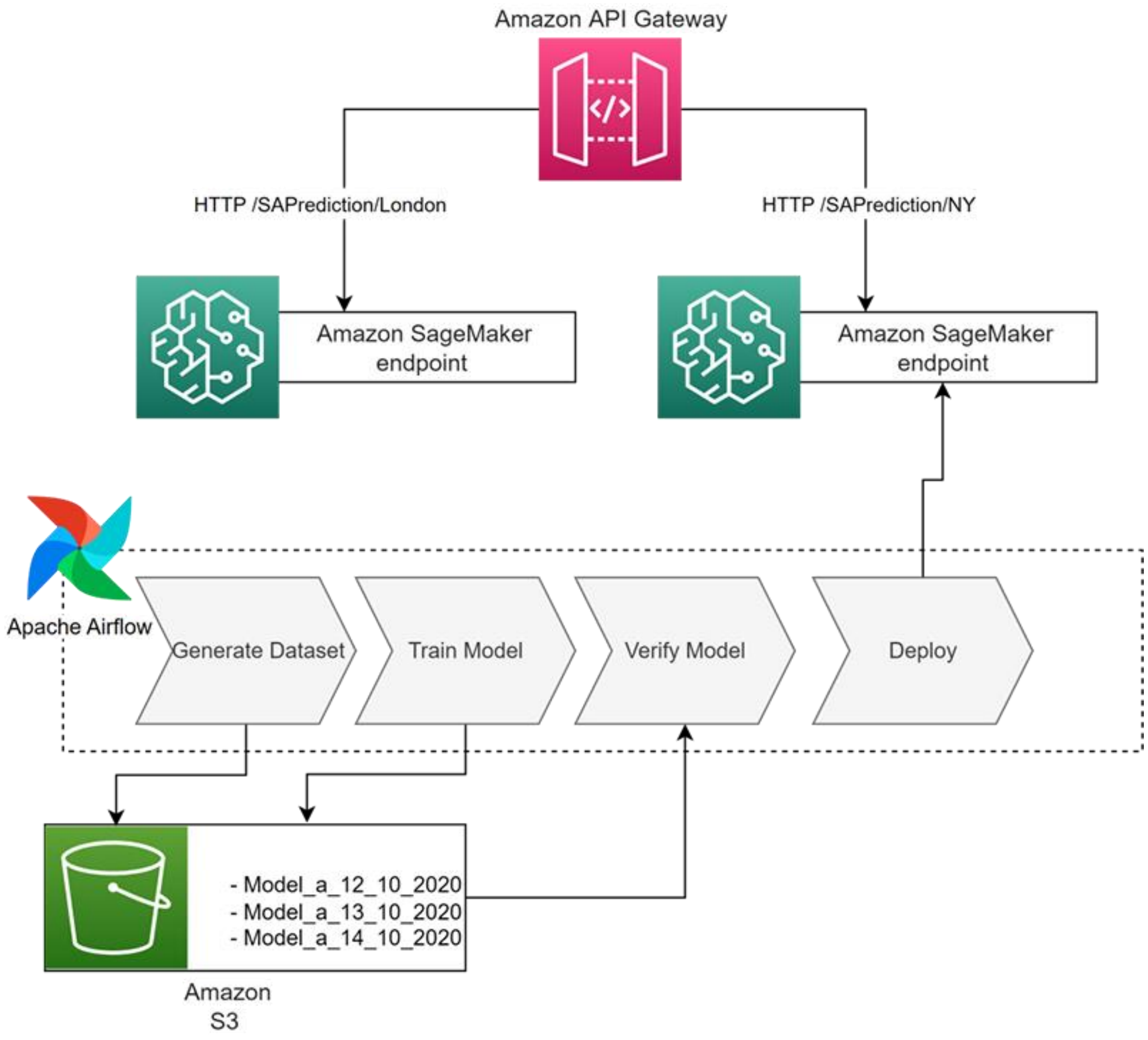


Figure 2.1: AI Lake architecture

(How Moovit turns data into insights to help passengers avoid delays using Apache Airflow and Amazon SageMaker | AWS Machine Learning Blog, *no date)*

While it is not feasible to incorporate data mining processes and machine learning processes of this magnitude into the proposed system due to time constraints and limited resources, it is crucial to examine the design implementation of an advanced system with similar features. This review will enable the proposed system to benefit from route optimization and live location correction algorithms that are proposed in the advanced system.

2.4 Related Design Strategies

2.4.1 Software Development Life Cycle

The modified Waterfall Model has been chosen as the system development methodology for several reasons. Requirements and scope are well defined at the initial phase of the development process, and less likely to change during the course of development.

Following the modified waterfall technique will make it easier to complete the required submission before deadlines, taking into account the milestones set by the faculty and the requirement of preparing elaborate documentation at every phase of the development cycle. This methodology will be more appropriate as more stable technology and tools are anticipated to be used.

The decision to use the modified waterfall approach over the classical method is justified by the fact that the latter is less flexible and does not encourage modifications in response to supervisory feedback.

2.4.2 System Deployment

The proposed system would be implemented as a cloud-based system because users must have access to real-time data from across the island, necessitating availability; when more users connect to the system, it must be scalable to handle the workload; and data backup and recovery must be simple. The configuration of SSL/TLS certificates would be straightforward by picking a reliable cloud service provider.

2.4.3 System Architecture

Given the requirement for processing real-time location data, route optimization, and estimated time calculations on the server side, as well as the need to deliver this information to a large number of clients simultaneously, a client-server system architecture is selected for the proposed system. This architecture allows for efficient data processing and delivery from bus conductor accounts (client) to passenger accounts (client) through the server upon request. To enable the purchase of virtual tickets, a server-side API will be implemented. This API acts as the intermediary between the client's mobile application and the payment gateway, ensuring secure and reliable transactions. The server will host a centralized database that stores and updates data such as routes, bus stops, fares, and other relevant information, ensuring all the passengers receive up-to-date and accurate information. Additionally, the server component allows for efficient management of user accounts, enhancing overall system reliability.

The selection of a client-server architecture aligns with the system's non-functional requirements as well. It offers scalability, enabling the system to handle a large number of concurrent users and deliver real-time updates without compromising performance. The centralization of data management ensures consistency and reliability of data across all clients. Robust security measures, including authentication, authorization, and encryption, are implemented on the server side to protect user information and secure payment transactions. Moreover, this architecture eases maintenance and updates, as server-side updates can be deployed without requiring users to update their mobile applications, resulting in a seamless user experience.

2.5 Related Technologies

2.5.1 Frontend Development

- **Android Java (Mobile Front End)**

The decision to use Java for the mobile application part is influenced by a number of factors. Firstly, it capitalizes on its native performance and seamless integration with hardware services such as GPS and Camera. Notably, Android has a substantial market share with, as of July 2023, 86.48% of Sri Lanka's mobile users utilising Android-based mobile devices (*Mobile Operating System Market Share Sri Lanka | Statcounter Global Stats*, no date). This widespread use ensures that the proposed solution will effectively serve a vast number of users. Additionally, the ability of Android Java to integrate simplified APIs, like Google Maps, guarantees effective real-time tracking and safe contactless fare payment, creating a user-friendly experience.

- **ReactJS (Web Front End)**

ReactJS will be utilised for designing the front-end of the web admin dashboard, using its versatile features to create a robust and dynamic user interface. With its component-based architecture, ReactJS allows for the modular organization of UI elements, enabling the development of reusable components that can be easily managed and updated. This approach promotes code maintainability and accelerates development cycles. Additionally, ReactJS's efficient rendering mechanism ensures faster page load times and smoother user interactions, contributing to a more responsive and seamless user experience. Furthermore, ReactJS's ecosystem of libraries and tools, such as Material UI, Nivo Charts, Formik, Yup, FullCalendar, Data Grid, and React Router, enhances development efficiency and enables the implementation of advanced features with ease.

- **Figma and Proto (Prototyping)**

Figma and Proto are used for UI/UX design, offering powerful prototyping capabilities to visualize and test different design approaches before implementation. Figma enables the creation of interactive prototypes, while Proto provides intuitive tools for visualizing prototypes, facilitating user feedback and design iteration.

2.5.2 Backend Development

- **Node.js (Backend Runtime Environment)**

Node.js serves as the backend runtime environment, handling business logic, data processing, and database interactions. Additionally, Node.js will be utilised for specific server-side operations like ticket order calculations and estimated time calculations. Its efficiency in real-time data processing and scalability make it suitable for managing concurrent connections, ensuring responsive communication. Node.js's extensive documentation and community support reinforce its reliability for server-side development.

- **Express.js (Web Application Framework)**

Express.js, a powerful and flexible web application framework for Node.js, will be utilised to streamline the development of the server-side components. Its minimalist design and robust features make it an ideal choice for building scalable and efficient web applications. Express.js simplifies routing, middleware configuration, and request handling, allowing for rapid development and deployment of RESTful APIs and web services. Additionally, its extensive ecosystem of middleware modules enables easy integration of essential functionalities such as authentication, session management, and error handling.

2.5.3 Database Management

- **SQLite and MongoDB**

SQLite manages structured data on the Android platform, ensuring consistency and transactional integrity. MongoDB handles dynamic and unstructured data on the server side, offering scalability and flexibility. This combination optimizes data storage and management across client and server environments.

- **MongoDB Compass and Atlas (Interactive GUI Tool for Database)**

MongoDB Compass and Atlas provide a powerful and intuitive GUI tool for querying, analysing, and optimizing databases. With MongoDB Compass, users can visually explore data, run ad-hoc queries, and perform performance optimization tasks. MongoDB Atlas complements Compass by offering a fully managed database service, ensuring scalability, reliability, and security for data storage needs.

2.5.4 External Services and Platforms

- **Google Maps API (Mapping and location services)**

serves as a critical component in this system, offering a wide range of functionalities such as providing directions, detailed information about places, and sharing real-time location data for public buses.

- **PayHere (Payment gateway integration)**

provides secure payment processing and supports multiple payment methods used in Sri Lanka.

- **Render (Cloud Application Hosting)**

The server backend applications and databases will be hosted on Render, a cloud platform recognized for its scalability, reliability, and high availability. Render's managed services and automated scaling allow for efficient deployment and management of the system without manual intervention. Moreover, Render provides robust security features and regular backups to safeguard user data.

- **GitHub (Code Hosting Platform for Version Control)**

GitHub serves as the primary platform for hosting and managing code repositories, enabling version control. Its robust features, including branching, merging, and pull requests, facilitate efficient code management, code review, and continuous integration.

2.5.5 Testing

- **TestNG and Selenium (Testing Frameworks)**

TestNG and Selenium are indispensable tools for automating testing processes and ensuring the quality and reliability of software. TestNG offers a flexible and potent testing framework, enabling the definition and execution of test cases, the generation of detailed reports, and the execution of parallel testing. Selenium complements TestNG by providing a robust automation tool for web application testing, facilitating the simulation of user interactions and the validation of application behaviour across various browsers and platforms.

- **TestCollab (Test Case Design and Management)**

TestCollab has been indispensable in designing, organizing, and managing test cases for the system. Its user-friendly interface and collaborative features allow for the creation and execution of test cases, tracking of test coverage, and generation of comprehensive reports.

- **Jira (Issue Tracking and Management)**

Jira serves as the central hub for tracking and managing issues throughout the development lifecycle. Its robust features, such as customizable workflows, agile boards, and real-time collaboration, enable streamlined issue tracking and task prioritization.

- **Postman (API Building and Testing)**

Postman has been instrumental in simplifying the process of building and testing APIs in the system. Its intuitive interface and comprehensive features allow for the efficient design, documentation, and testing of APIs.

2.5.6 Integrated Development Environment

- **Android Studio (Android Development):**

Android Studio serves as comprehensive IDE for developing Android applications. Its rich set of tools, including code editor, debugger, and emulator, streamline the development process and enhance productivity.

- **Visual Studio Code (Source-Code Editor)**

Visual Studio Code serves as versatile source-code editor, providing a lightweight yet powerful tool for writing, editing, and debugging code. Its extensive ecosystem of extensions and integrations enhances development workflow, allowing to customize and optimize coding environment according to preferences.

Chapter 3 – Design

3.1 Introduction

The System Design phase is one of the most significant phases of the software development life cycle, if not the most important. It serves as the stage where the overarching architecture and intricate design of a software system or application are meticulously planned and defined. The requirements gathering and analysis phase, which comprises the thorough collection and evaluation of project requirements and functional specifications, is followed by this phase.

The acquired requirements are transformed into a comprehensive blueprint for the system's design, organisation, and integration during the System Design phase. Intricate technical components, including system architecture, data structures, modules, interfaces, and algorithms, are the focus of this phase. The main goal is to make that the system adheres exactly to the specified requirements and functions cohesively as intended.

A key component of this phase is the development of architectural design diagrams, including class diagrams, interaction diagrams, and deployment models. These diagrams strictly follow the Unified Modelling Language (UML) standard notation, allowing for clear stakeholder communication and comprehension.

The System Design phase also involves thorough consideration of design choices, with interface designs taking the stage. These discussions are founded on a careful examination of the justification for each design decision. This rigorous examination makes sure that the chosen design elements perfectly complement the overall architecture and support the system's functional and usability requirements.

All the design strategies that will be discussed in this chapter are referenced from the 'Guide to the Systems Engineering Body of Knowledge (SEBoK), version 2.8 Guide to the Systems Engineering Body of Knowledge version 2.8'.

3.2 Design Strategies

3.2.1 Design Methodology

The agile methodology initially seemed like an appropriate choice when comparing design techniques for the proposed system because of its flexibility and user-centered risk management strategy. However, adopting agile approaches, which are typically intended for bigger team-based projects, offers difficulties given the nature of this individual student project, which is bound by predetermined deadlines for every phase.

In light of this, a practical strategy is desired. Alignment with the faculty's imposed schedule for each design step is essential since the project must stay on track. In contrast to bigger team projects, the distinct advantage of an individual project is the early and precise understanding of requirements. This particular set of requirements is better handled by the traditional waterfall process, which is known for its sequential approach.

The preferable design framework, however, appears to be the modified waterfall methodology when taking into account its flexibility to respond to changing needs during the course of the project while establishing into the pertinent documents seamlessly.

The waterfall model gets its name from the way it progresses in phases, with each phase having a clear beginning and end and contributing to the outputs of the next. This methodical approach is ideal for projects with consistent, clear requirements.

Adapting the modified waterfall methodology, following the gathering and analysis of requirements outlined in the preceding chapter, the present chapter will delve into the design phase of the system. This chapter will cover aspects such as the system architecture and various design considerations.

3.3 System Architecture

Simplified system architecture diagram for the system is shown below (Figure 3.1).

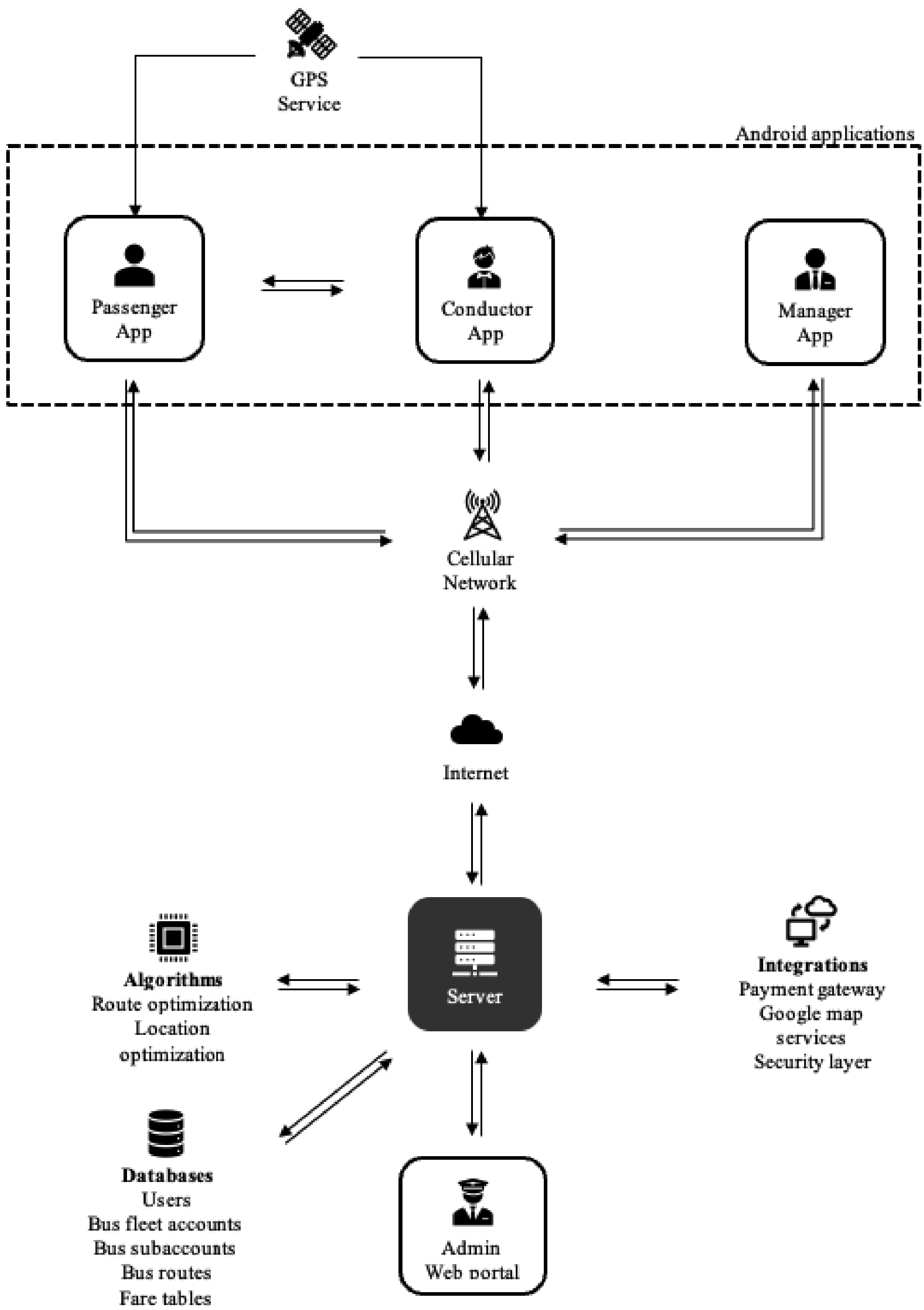


Figure 3.1: Main parts of the system and their functionalities

The above diagram (Figure 3.1) illustrates the planned system architecture and outlines the communication flow between each component. The real-time location of buses will be acquired through the 'Conductor App,' utilizing the GPS capabilities of the mobile devices, and subsequently transmitted to the central server via the cellular network. Similarly, passengers will access their real-time location using their mobile GPS, sending this information to the server for the purpose of receiving directions and information about available buses along their route. Communication between the passenger and conductor applications will occur when virtual tickets held by passengers are verified through the conductor's application.

The Manager application will establish a connection with the server, utilizing either a cellular or Wi-Fi network, to manage their respective bus fleet accounts. Additionally, a web interface is provided for administrators to oversee and manage bus manager accounts, routes, and fare tables. The server maintains communication with various databases, including those related to bus schedules, fares, bus fleets, and bus accounts. Furthermore, the server seamlessly integrates with external services such as Google Maps and payment gateways to enhance functionality and user experience.

3.4 UML Diagrams

This section is dedicated to the systematic representation of the proposed system using Unified Modelling Language (UML) diagrams. UML diagrams are employed as a potent visual tool to capture the architectural, structural, and behavioural facets of the system, facilitating a comprehensive understanding of its design and functionality.

The presentation commenced with a high-level depiction of the system architecture via deployment diagrams, showcasing the allocation of components across diverse nodes and environments. Subsequent to this, the focus shifts towards illustrating the intended use cases along with their associated actors. Furthermore, the interaction and communication among system components are elaborated upon through the utilization of class, activity, and sequence diagrams. These diagrams illuminate the dynamic progression of processes during real-time bus tracking and interactions related to fare payment.

Throughout the expanse of this chapter, a pivotal role is assumed by UML diagrams in articulating the intricate mechanics of the innovative system. Their incorporation not only augments the comprehensibility of the project thesis, but also establishes an indispensable point of reference for the subsequent development and refinement of the system.

3.4.1 User Case diagram

To underscore user interactions and interface design, use case diagrams are employed. This diagram (Figure 3.2) showcases a range of functionalities accessible to passengers, conductors, bus managers, and administrators. These visualizations adeptly encapsulate the breadth of user interactions, forming a fundamental basis for shaping the overall user experience.

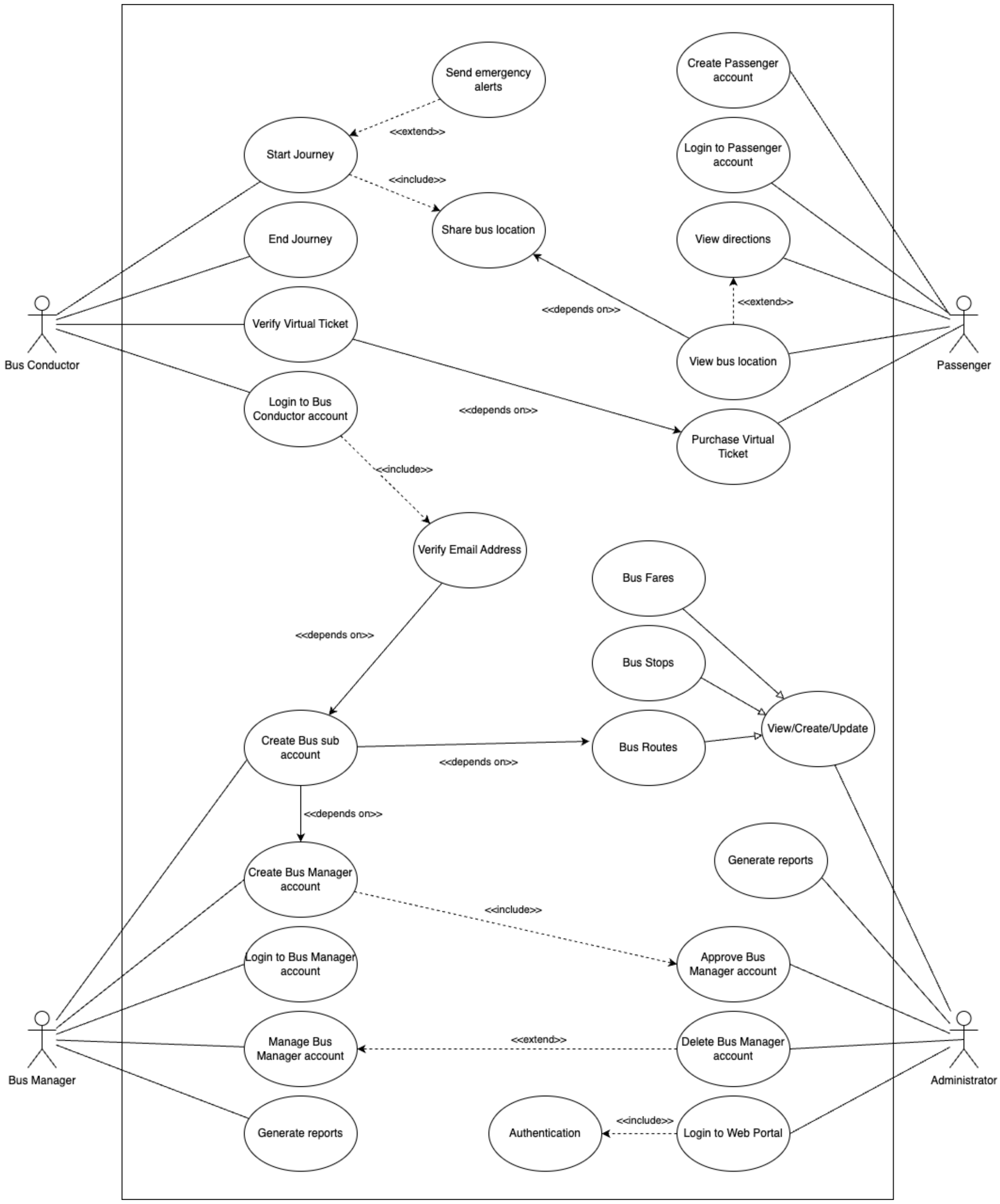


Figure 3.2: High level Use Case diagram

3.4.2 Class diagram

Class diagrams (Figure 3.3) offer an intricate perspective into the system's data structure, revealing the interconnections among classes, their attributes, and methods. These diagrams provide a lucid depiction of the data models that form the foundation of this application's functionality, thereby augmenting both clarity and the ability to ensure ongoing maintainability.

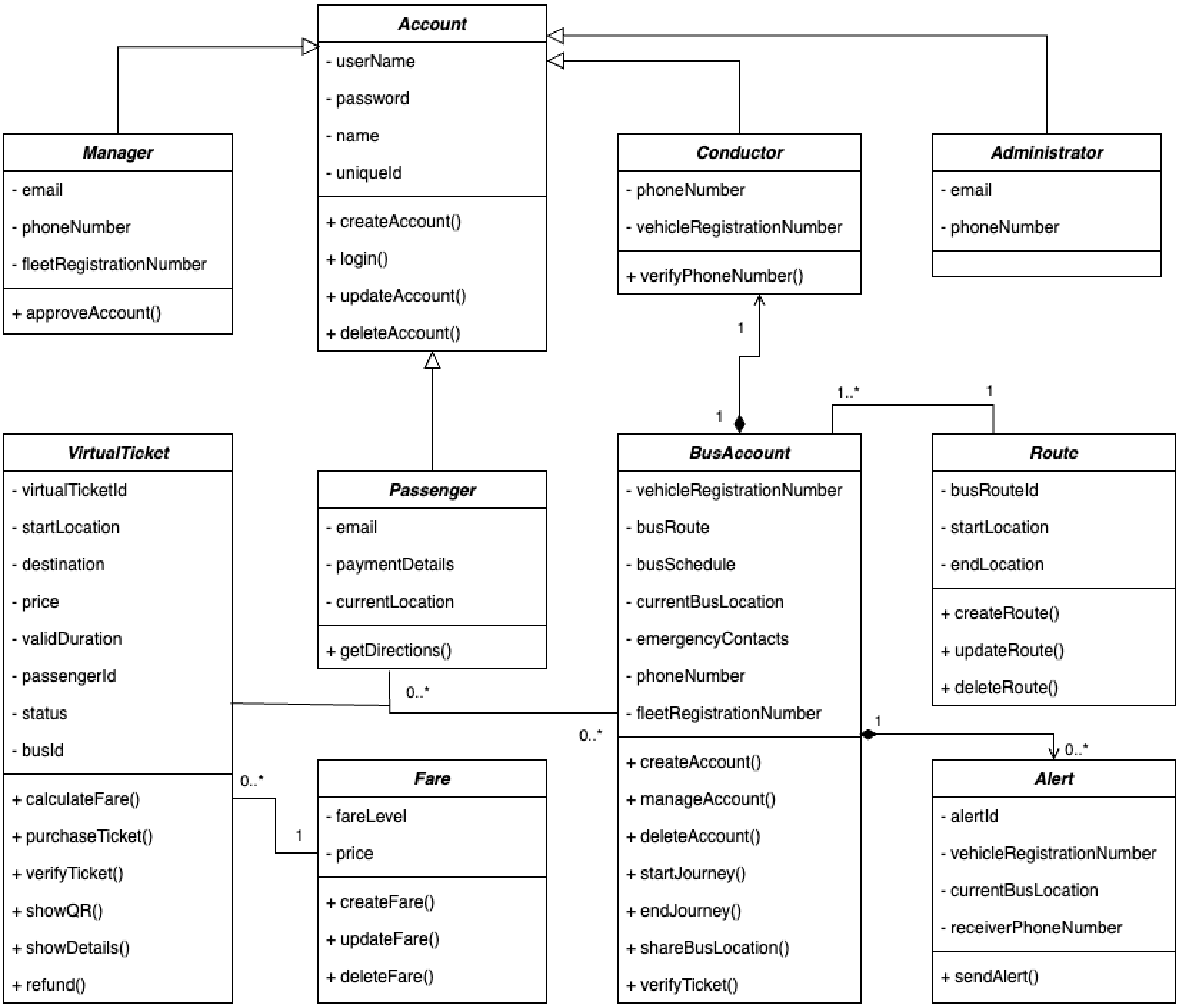


Figure 3.3: Class diagram

3.4.3 Activity diagram

The activity diagram (Figure 3.4) illustrates the sequential flow of actions and decisions involved in the process of purchasing a virtual ticket within the Passenger App, its subsequent validation by the Conductor's App through QR code scanning, and the provision of an option for purchasing a physical ticket in the event of a failure in virtual ticket purchase or validation.

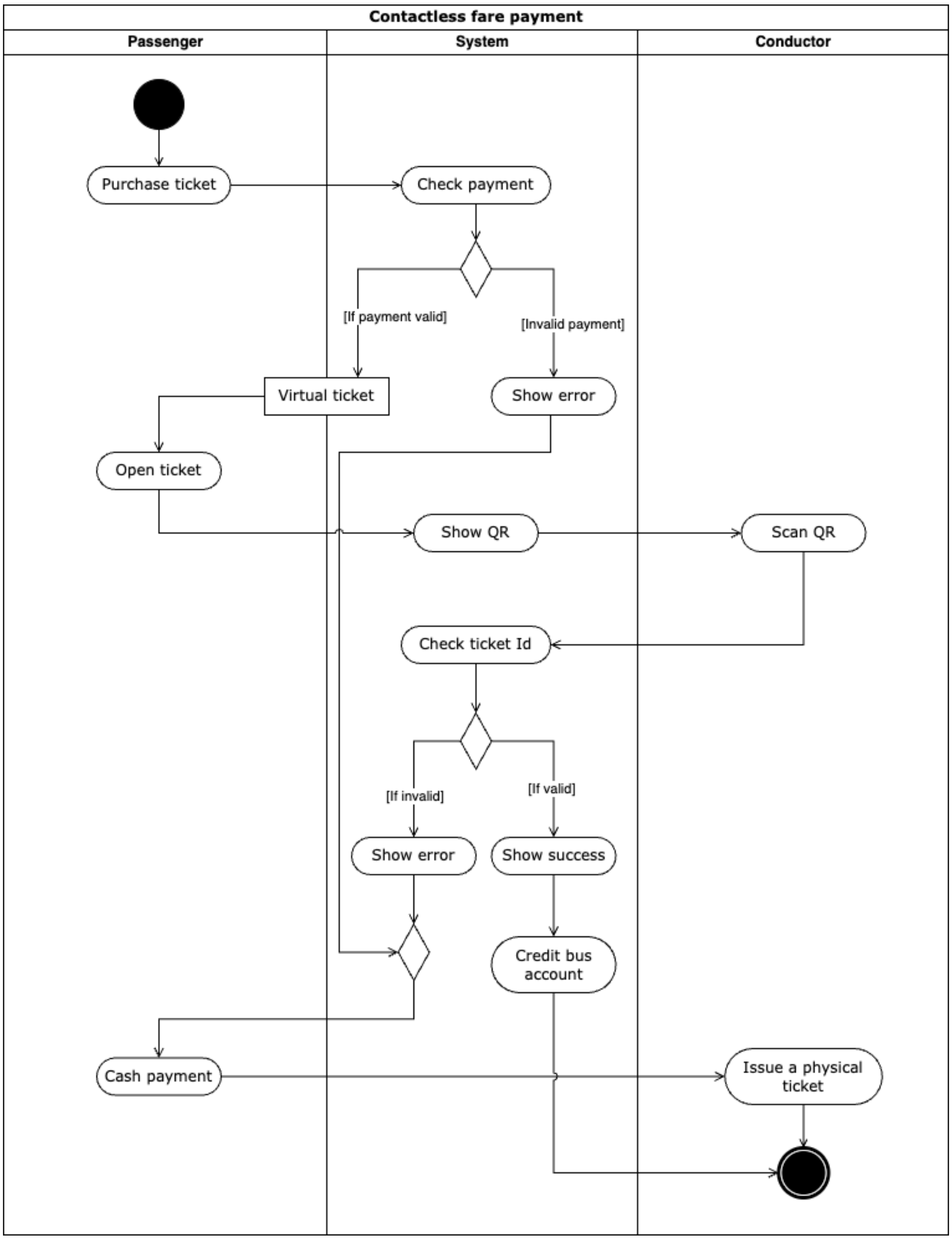


Figure 3.4: Activity diagram of purchasing a virtual ticket

3.4.4 Sequence diagram

This Sequence Diagram (Figure 3.5) reveals the dynamic interactions that shape the functionality of this system. It visually depicts the fluid exchanges occurring among the Passenger App, Server, GPS Tracking Service, Google Routes and Directions API and Map Display Interface, effectively portraying the seamless fusion of technology and transit operations. By tracing the sequential progression of data and actions, this diagram provides a glimpse into the synchronized operation that translates GPS coordinates into real-time bus locations on passengers' screens.

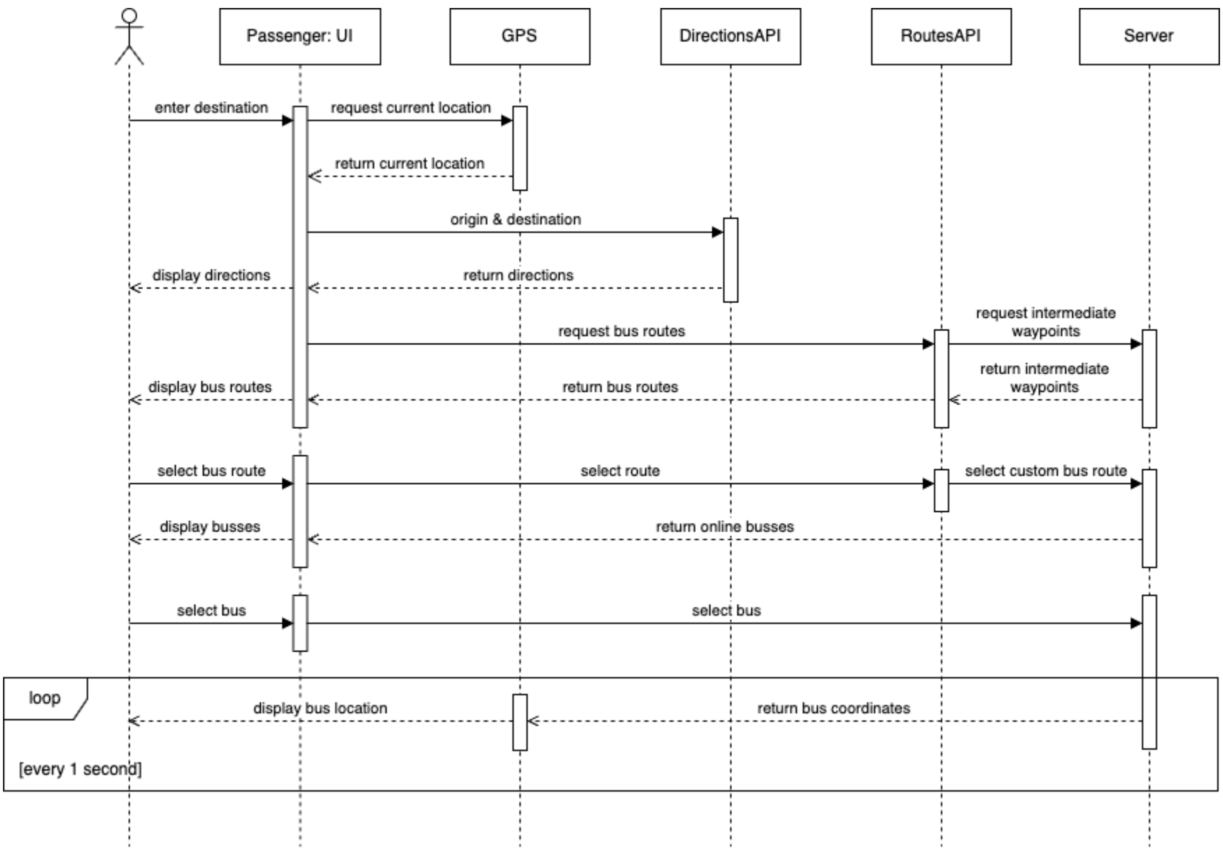


Figure 3.5: Sequence diagram of display real time bus location

3.4.5 EER (Enhanced/Extended Entity-Relationship) diagram

Database design plays a crucial role in the system as it ensures consistency across the database, thereby maintaining synchronized and accurate data. Techniques like normalisation are vital in eliminating redundancy and ensuring consistent reflection of data updates throughout the system. Moreover, the security and performance of databases are heavily reliant on effective design. Utilising relational database techniques, the EER diagram (figure 3.6) illustrates how important entities interrelate and communicate, facilitating seamless database operations.

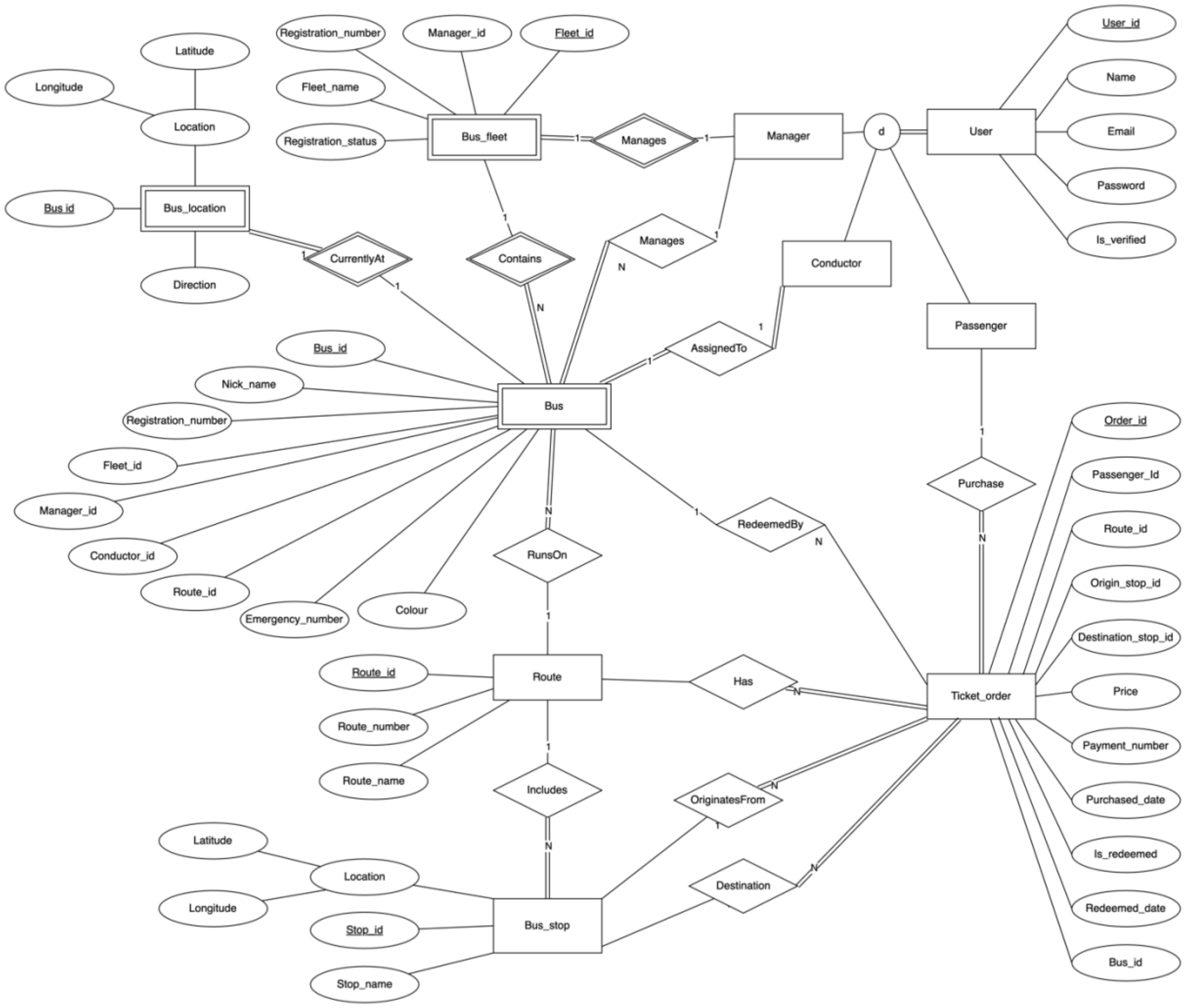


Figure 3.6: EER diagram of the system

3.5 User Interface Design

In the realm of modern technological solutions, the User Interface (UI) serves as a critical juncture where form meets function. In this sub section, the intricate landscape of UI design within the context of the developing system will be discussed. The primary focus lies in the seamless integration of aesthetic appeal and optimal functionality, while aligning with the foundational principles of usability encapsulated within the 5Es framework.

Efficiency: The interface is strategically architected to empower passengers, conductors, managers and administrators in accomplishing tasks swiftly and with precision. The orchestrated interplay between UI elements ensures a coherent flow of operations, translating to expedient and purposeful interactions.

Effectiveness: Beyond mere aesthetics, the UI embodies effectiveness by facilitating tasks seamlessly. Through its design, users are empowered to engage in real-time bus tracking and fare payment with minimum cognitive load, ensuring a frictionless user journey.

Engagement: Aesthetic coherence and interactivity combine to immerse users in an engaging experience. By employing visually captivating displays of bus routes and interactive map features, the UI sustains user interest and active participation throughout their interaction with the system.

Ease of Learning: The UI's design rationale leans towards providing an intuitive and easily graspable environment. Through a logical hierarchy of information, consistent design patterns, and user-friendly controls, the learning curve is mitigated, allowing users of varying expertise levels to seamlessly acclimate to the system.

Error Management: Robust UI design extends beyond error prevention to encompass graceful recovery mechanisms. Users are safeguarded by clear feedback, intuitive prompts, and guided corrective pathways, thereby mitigating the potential fallout of inadvertent errors.

The following four diagrams exemplify the user interfaces and their consistent design across all four sections of the mobile applications.

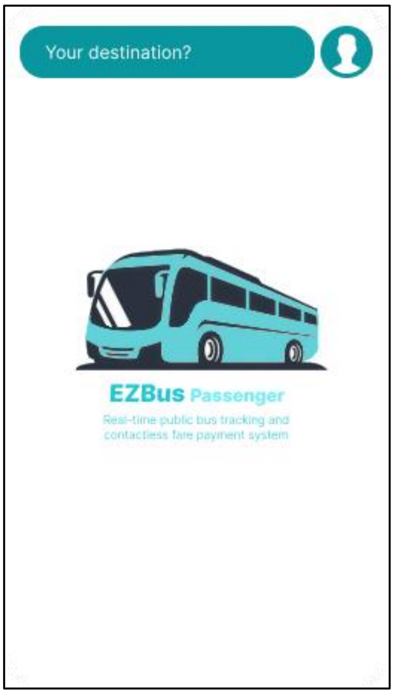


Figure 3.7: Passenger app - Home screen

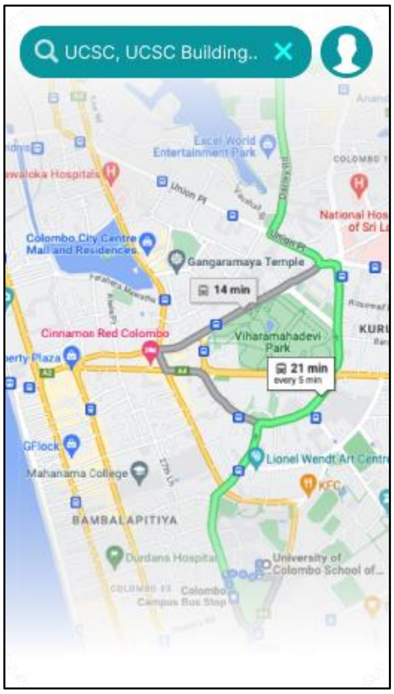


Figure 3.8: Passenger app - Search a destination

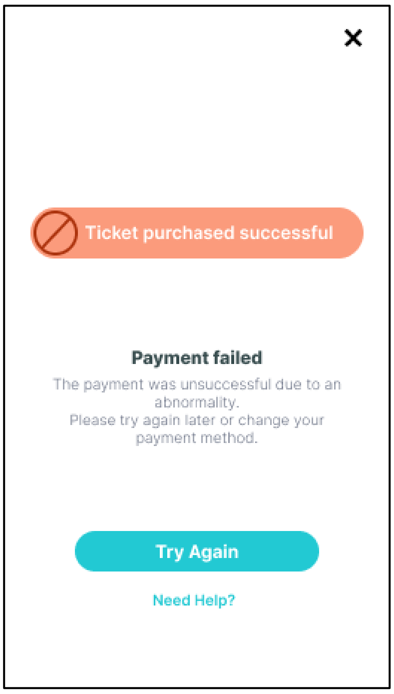


Figure 3.9: Passenger app – Purchasing ticket failed

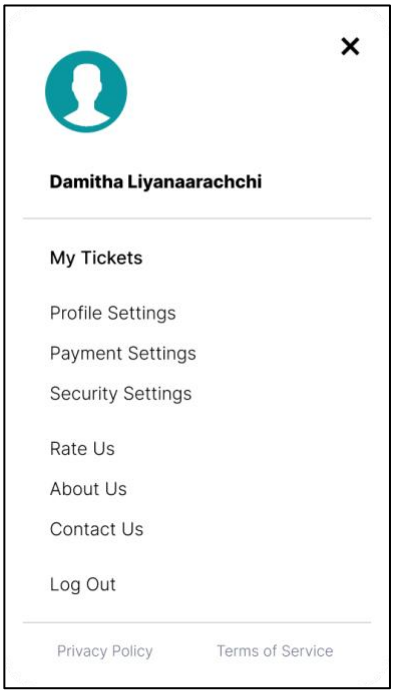


Figure 3.10: Passenger app – Settings screen

It is crucial to follow design guidelines and principles that are intended to improve the user's interaction with the application.

Maintaining **Consistency** throughout the whole application is crucial to ensure users are not disoriented and can easily navigate the application. By establishing consistency in elements such as colours, fonts, and layouts across all components of the system (as depicted in figures 3.7 - 3.10), important actions receive the attention they deserve, enhancing usability and user experience.

Familiar design patterns play a pivotal role in establishing cognitive models in users' minds and bridging the gap between execution and evaluation. To facilitate ease of use, all user interfaces have been meticulously designed with a consistent style similar to popular transportation related applications like Google Maps (refer to figure 3.8). This approach minimises the learning curve for users to learn new systems, as they can leverage existing knowledge and experiences.

Simplicity is key in mobile application design, especially considering the limited screen space available. Interface designs, exemplified in figure 3.7, prioritize clarity and visibility of important features to ensure they are easily accessible to users.

Affordance and **navigation** are essential for seamless user interaction. Clear button designs signal their interactive nature, guiding users on where to tap. Additionally, due to screen space constraints, certain functions are presented on separate screens (as shown in figure 3.10), with intuitive navigation paths to ensure accessibility.

Feedback mechanisms play a crucial role in usability. Providing users with timely updates on the status of actions, whether it be loading, verification, success, or failures (as demonstrated in figure 3.9), is vital. Inadequate feedback can lead to confusion and diminish the perceived efficiency and effectiveness of the application.

Chapter 4 – Implementation

4.1 Introduction

The implementation chapter serves as a practical demonstration of the theoretical framework outlined in the preceding chapters, translating concepts and methodologies into tangible solutions. This section offers an in-depth exploration of the steps involved in building and deploying the system, shedding light on the tools, technologies, and methodologies employed in the process.

Through concise explanations, code snippets, and illustrative diagrams, this chapter offers a comprehensive overview of the technical aspects of the project. It serves as a valuable resource for understanding the practical implications of the research findings and provides guidance for future endeavours in the field.

4.2 Development Methodology

The modified Waterfall Model was selected as the system development methodology for several reasons. At the initial phase of the development process, requirements and scope were clearly defined, reducing the likelihood of changes during development. Following the modified waterfall technique made it easier to complete the required submissions before deadlines, considering the milestones set by the faculty and the requirement of preparing elaborate documentation at every phase of the development cycle. This methodology was deemed more appropriate as stable technology and tools were anticipated to be used.

Therefore, when the implementation began, all the requirements were translated into well-crafted designs. These designs served as the foundation for the implementation process, which commenced promptly. Simultaneously, comprehensive learning sessions were conducted in parallel with the implementation, focusing on each of the selected technology stacks.

4.3 Technology Stack

In the development of the system, a careful selection of technologies and frameworks was made, considering their features, suitability for the project, and potential for seamless integration among them. Detailed justifications for the use of these selected technologies are provided in Chapter 2, Section 2.5, Related Technologies. Therefore, while the introduction of each technology won't be reiterated here, the practical application and implementation within the project will be discussed. This chapter aims to delve into how these technologies were utilised in practice, highlighting their roles, functionalities, and contributions to the overall system architecture and functionality.

Frontend Development

1. Android Java: Front end development of mobile applications
2. ReactJS: Front end development of admin web portal
3. Figma and Proto: UX/UI design and Prototyping

Backend Development

1. Node.js: Backend runtime environment
2. Express.js: Web application framework

Database Management

1. SQLite: Database on client mobile applications
2. MongoDB: Database on server
3. MongoDB Compass: Interactive GUI tool for database design
4. MongoDB Atlas: Cloud storage integration for database

External Services and Platforms

1. Google Maps API: Mapping and location services
2. PayHere: Payment gateway integration
3. Render: Cloud application hosting
4. GitHub: Code hosting platform for version control

Testing

1. TestNG: Testing framework for mobile applications
2. Selenium: Testing framework for web application
3. TestCollab: Test case design and management
4. Jira: Issue tracking and management
5. Postman: API building and testing

Artificial Intelligence

1. GitHub Copilot: Code completion tool
2. ChatGPT: Artificial intelligence chatbot

In the following sections, the utilization of each of these technology stacks in the project will be discussed in detail.

4.4 Development Environment

For mobile application development, **Android Studio Hedgehog** (version 17.0) served as the integrated development environment (IDE), leveraging its simulators, primarily **Pixel 7** (Figure 4.1). The **Java Runtime Environment** (version OpenJDK version 17.0.7) was utilised for executing Java code within the Android Studio environment, ensuring compatibility and runtime support for the application.

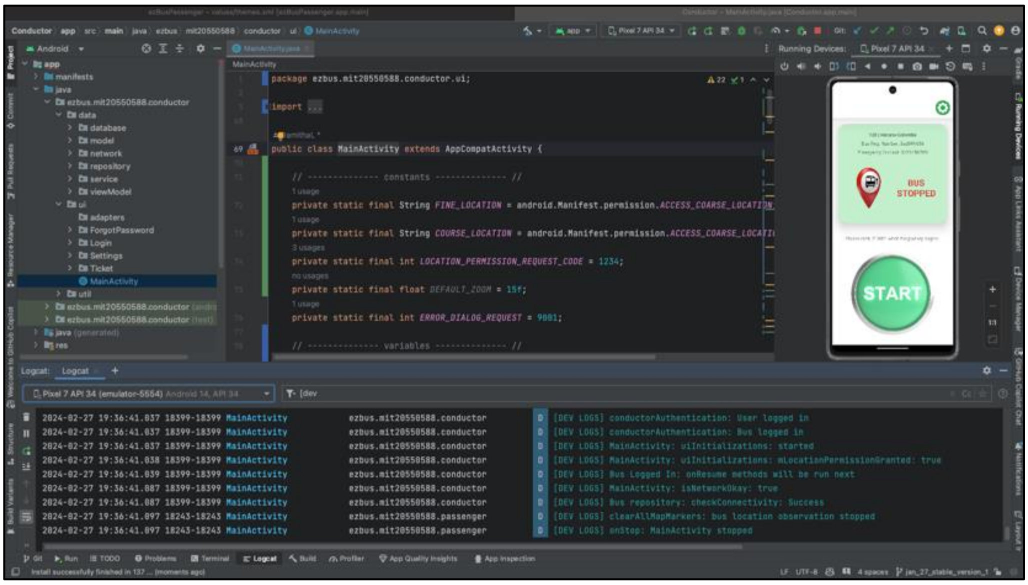


Figure 4.1: Conductor project developing on Android Studio

For backend and web portal development, **Visual Studio Code** (version 1.86.0) was employed as the source code editor, enhanced with several extensions to optimize development workflows (Figure 4.2).

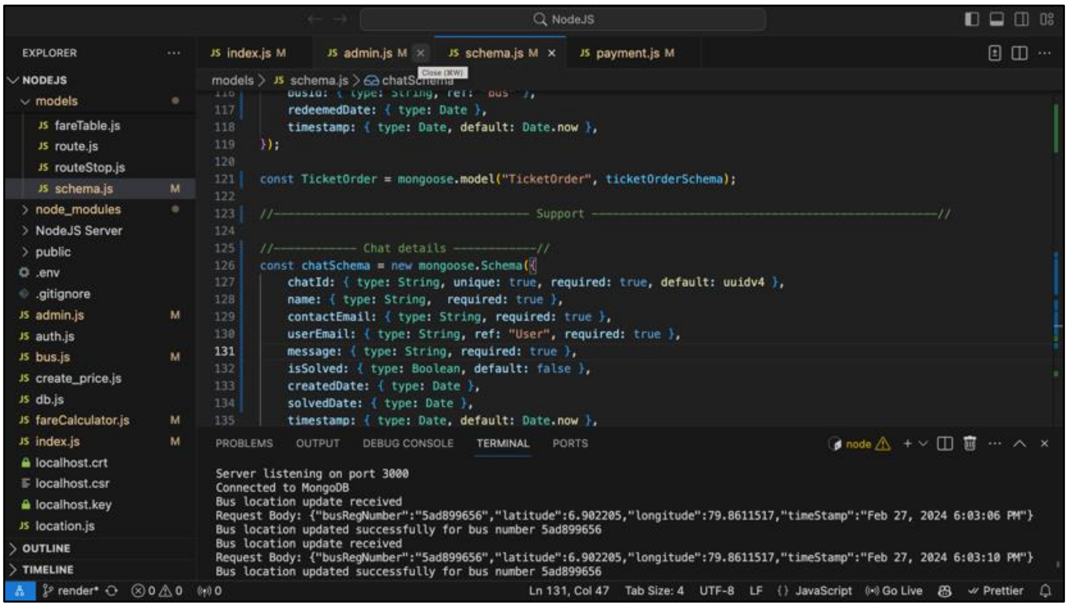


Figure 4.2: Backend server development in VS Code

Chrome (Version 122.0) functioned as the browser, with its inspector tool utilised for debugging and inspecting web elements during development (Figure 4.3).

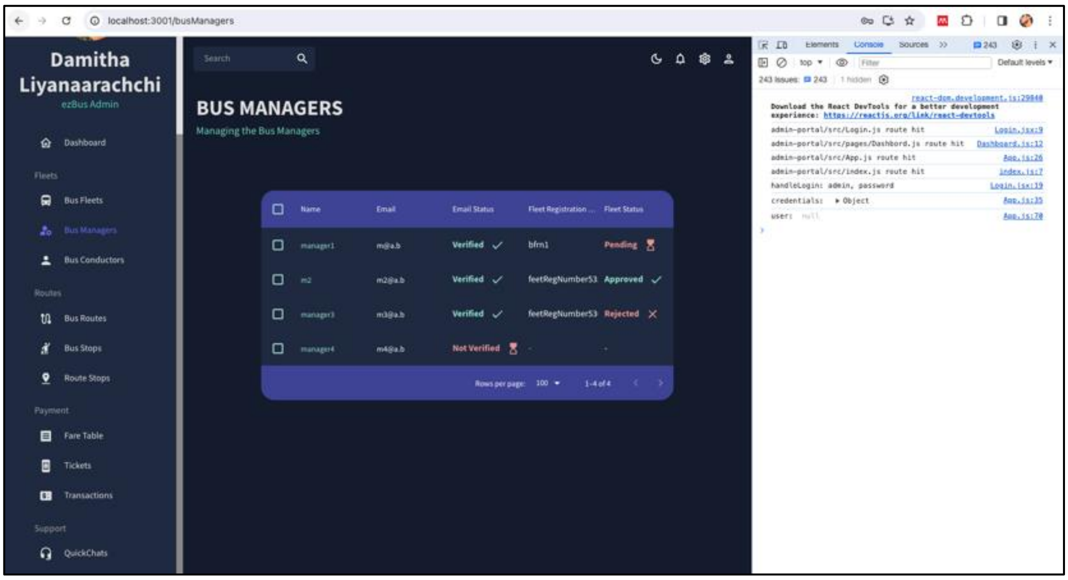


Figure 4.3: Web portal front end inspection in Chrome browser

Development activities were conducted on a personal Mac running **macOS Ventura** (version 13.0). The backend Node.js server was initially hosted on localhost for local testing purposes. Subsequently, the server was deployed on **Render**, a cloud server hosting platform, enabling real-world testing of client-server communication (Figure 4.4). Additionally, a domain was acquired and configured with an SSL certificate to ensure secure communication between clients and the server (Figure 4.5).

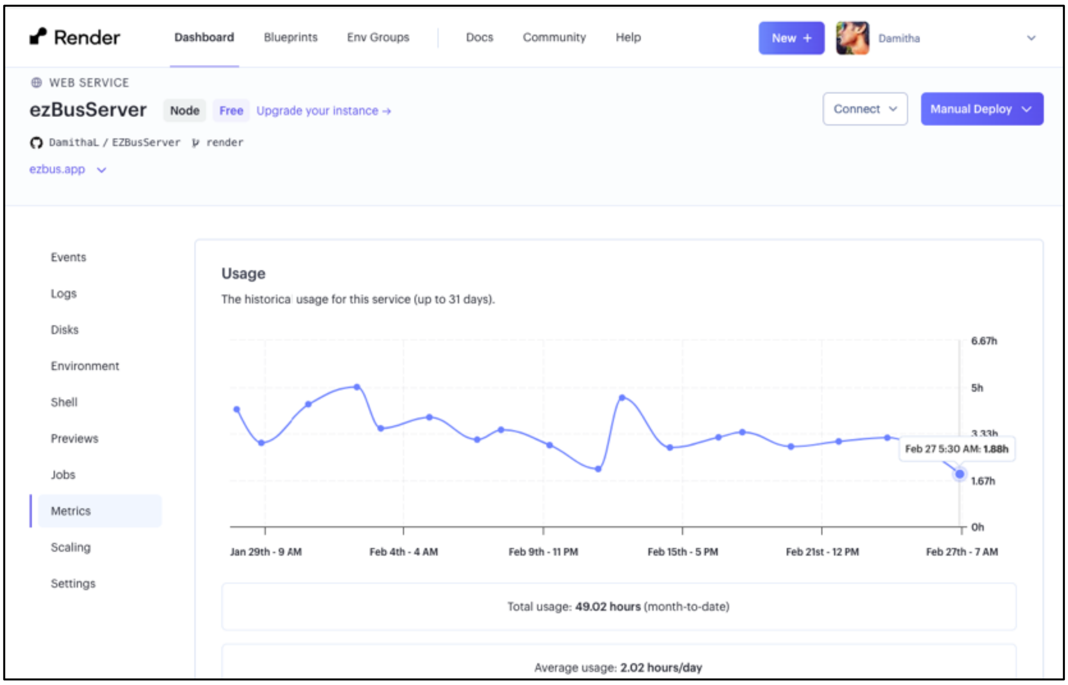


Figure 4.4: Backend server hosted on Render

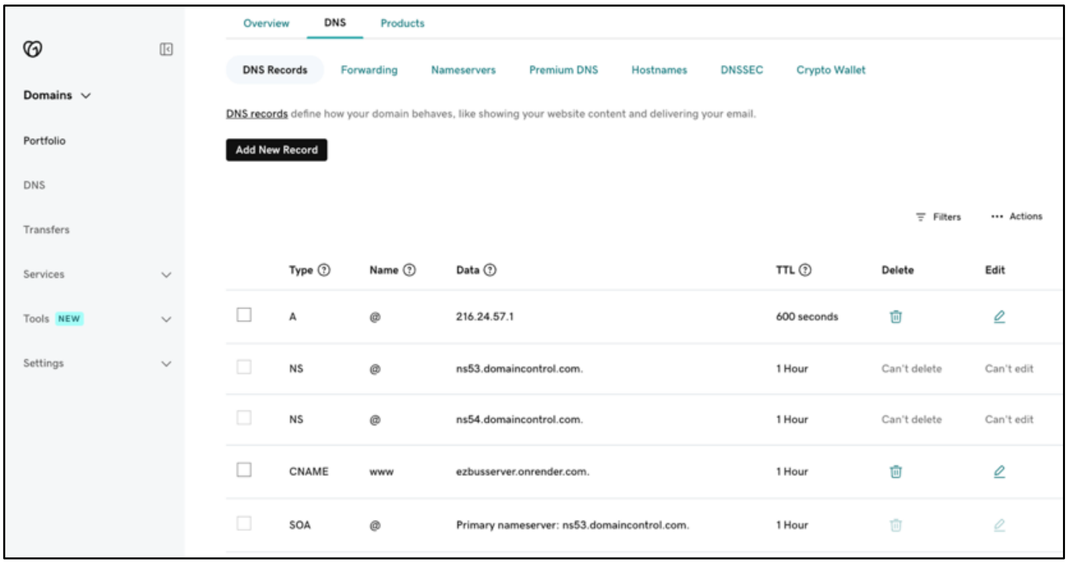


Figure 4.5: Custom domain for the server on GoDaddy

4.5 Implementation of Front End

The system's frontend comprised two components: mobile applications tailored for passengers, conductors, and managers, and a web application for administrators. The mobile applications were meticulously developed with a focus on responsiveness and modularity. The MVVM (Model-View-ViewModel) architecture (Figure 4.6) was employed to separate the user interface logic from the business logic, ensuring code maintainability and testability.

MVVM was chosen because it facilitates easier updates to the user interface without altering the business logic, allows for better separation of concerns, and improves scalability and flexibility, specially for complex applications.

MVVM Architecture

Model: Represents the data and business logic of the application, encapsulating data and defining operations that can be performed on it. In the system, the model was represented by repositories, Data Access Objects (DAO), Room databases, and APIs connecting to the backend server.

View: Responsible for displaying the user interface and facilitating user interaction. In the system, views were represented by XML layout files and UI widgets such as TextViews, EditTexts, and Buttons.

ViewModel: Acts as a mediator between the view and the model, holding the presentation logic and state of UI components. ViewModels are framework-independent, making them easier to unit test. They also manage UI-related data in a lifecycle-aware manner, surviving configuration changes like screen rotations. In the system, ViewModels were implemented as subclasses of the Android ViewModel class.

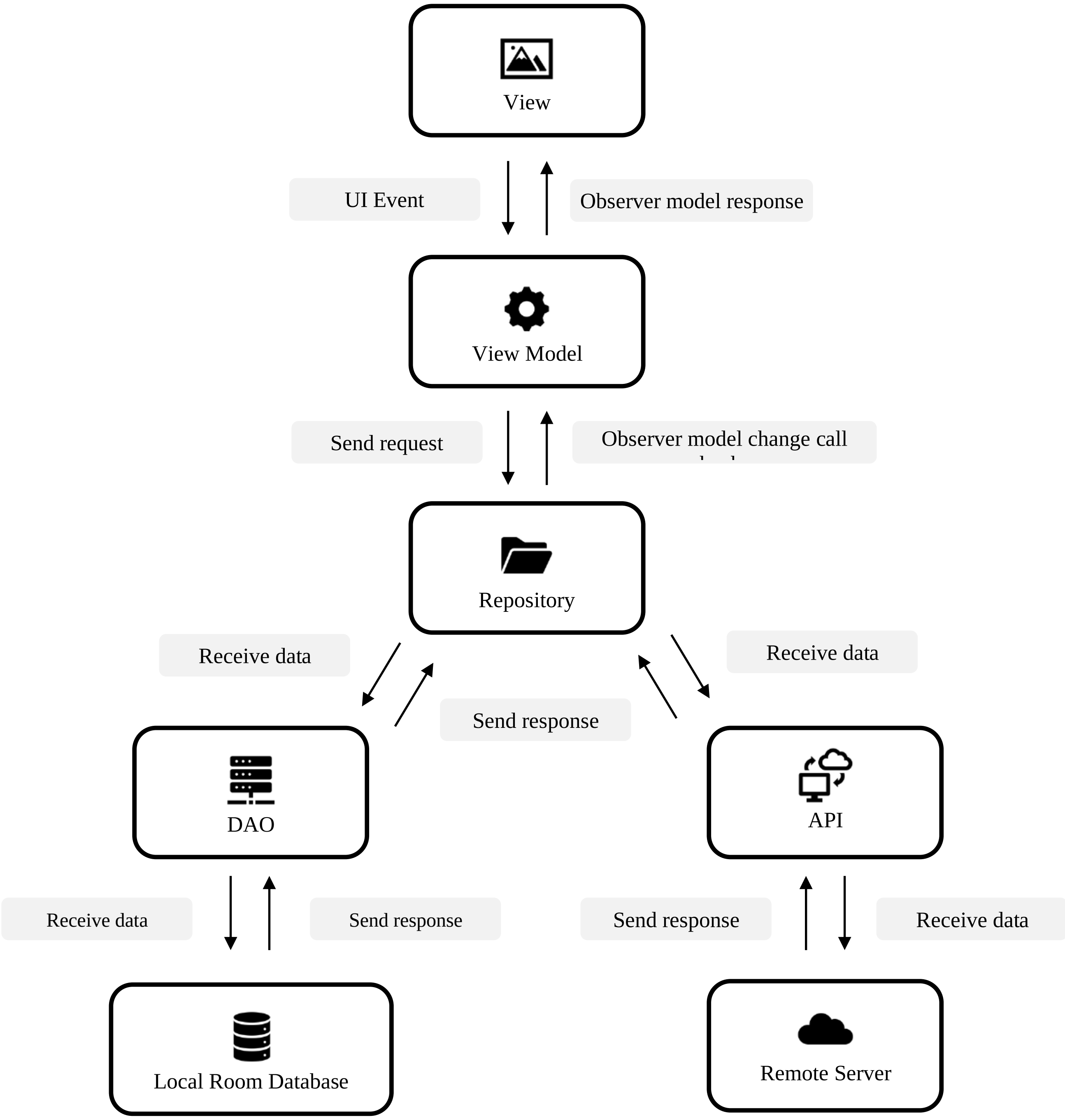


Figure 4.6: Data flow in MVVM architecture

4.5.1 User interfaces

For the implementation of user interfaces, usability principles such as the 5E framework were utilised, as described in Chapter 3, Section 3.5 on user interface design.

The following screenshots (Figure 4.7 – 4.12) showcase the mobile application and web application, highlighting the consistency achieved in both interfaces. These screenshots serve as visual representations of the user interface design, demonstrating adherence to usability principles and consistency across different sections of the applications.

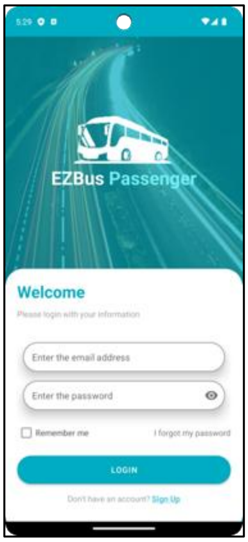


Figure 4.8: Passenger app - Login screen

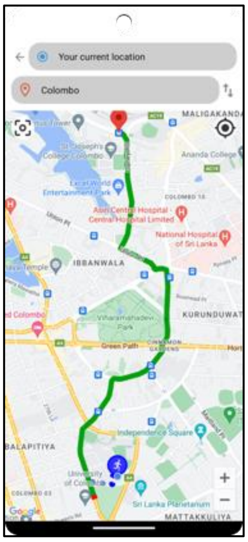


Figure 4.7: Passenger app - Search a destination

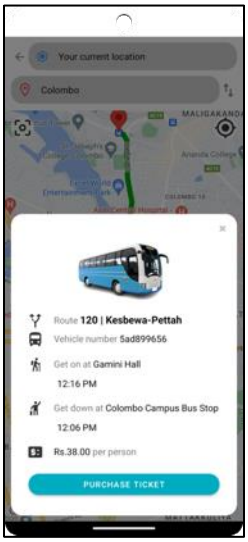


Figure 4.9: Passenger app - View real time bus data

Figure 4.10: Passenger app - View virtual ticket QR

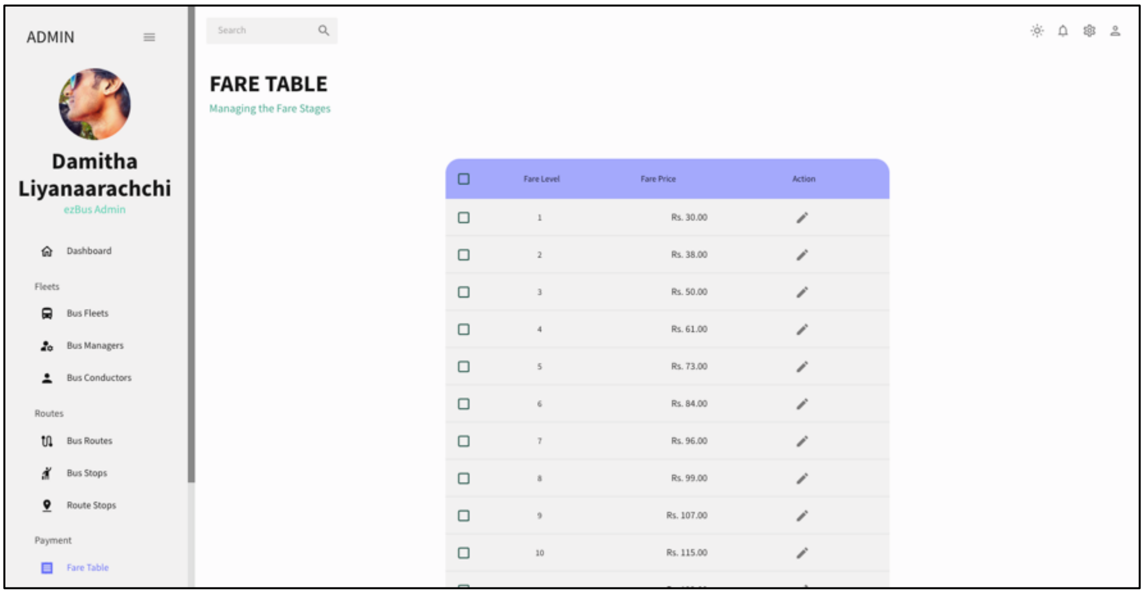


Figure 4.11: Admin Portal - Fate Table Tab (Light Mode)

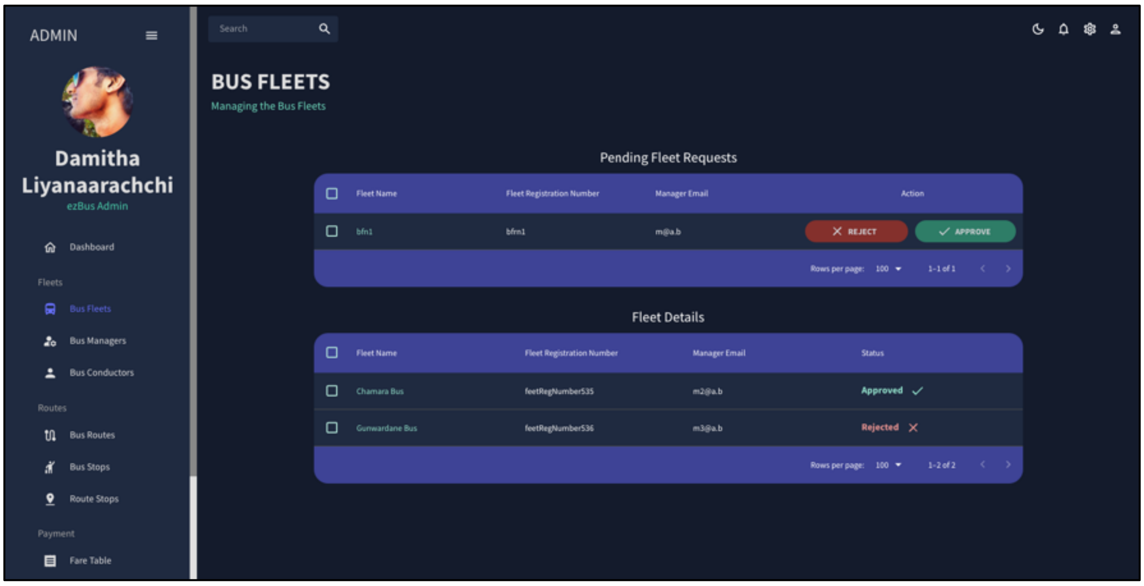


Figure 4.12: Admin Portal - Bus Fleets Tab (Dark Mode)

4.6 Implementation of Back End

Communication between the Node.js backend server and the Android Java and ReactJS front ends was facilitated through the use of APIs. Application Programming Interfaces, or APIs, act as a bridge between various software systems so they can communicate with one another. In this context, APIs were utilised to establish a connection between the backend server, which housed the core business logic and data, and the frontend applications running on Android Java and ReactJS. Through these APIs, data could be retrieved, modified, and exchanged between the server and the client applications, enabling seamless interaction and data flow across the entire system.

4.6.1 Building and testing of APIs

For building and testing APIs, Postman was utilised (Figure 4.13). Subsequently, the mobile applications interacted with the backend API endpoints (Figure 4.16) by sending and receiving data through their public interfaces (Figure 4.14). Similarly, the web application communicated with the backend using libraries such as ‘Axios’ to send and receive data (Figure 4.15).

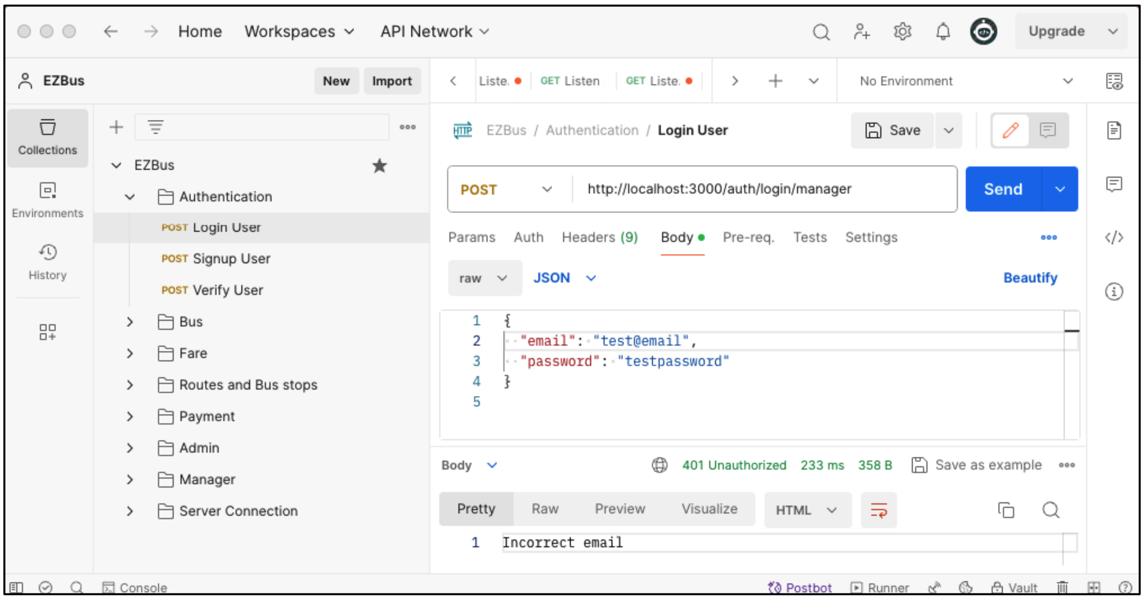


Figure 4.13: Building APIs on Postman API platform

```java
package ezbus.mit20550588.conductor.data.network;

import java.util.Map;
import ezbus.mit20550588.conductor.data.model.UserModel;
import retrofit2.Call;
import retrofit2.http.Body;
import retrofit2.http.POST;

public interface ApiServiceAuthentication {

    @POST("/auth/login/conductor")
    Call<UserModel> loginUser(@Body LoginRequest loginRequest);

    @POST("/auth/signup/conductor")
    Call<Map<String, String>> registerUser (@Body RegistrationRequest
registrationRequest);

    @POST("/auth/verify/conductor")
    Call<UserModel> verifyUser (@Body Map<String, String> email);
}
```

Figure 4.14: API for authentication on Conductor application

```javascript
// Fetch data from the backend and update state

  useEffect(() => {

    const fetchData = async () => {

      try {

        const response = await axios.get(`${serverUrl}/admin/read/chats/all`);

        setQuickChats(response.data);

        setErrorMessage(null);

      } catch (error) {

        console.error("Error fetching chats: ", error.message || error);

        setErrorMessage("Error fetching chats: ", error.message || error);
```

```
        }
    };

    fetchData();
}, []);
```

Figure 4.15: API for fetch support chat data on admin application

```
router.post("/login/manager", async (req, res) => {
  console.log("a manager is trying to login");
  try {
    const { email, password } = req.body;
    console.log(
      "request body: " + "email: " + email + ", password: " + password
    );
    const user = await Manager.findOne({ email });
    if (!user) {
      console.log("Incorrect email");
      return res.status(401).send("Incorrect email");
    }
    const isPasswordValid = await bcrypt.compare(password, user.password);
    if (!isPasswordValid) {
      console.log("Incorrect password");
      return res.status(401).send("Incorrect password");
    }
```

```
    // Check if the email is verified

    if (!user.isVerified) {

      console.log("User is not verified");

      return res.status(406).send("Not verified");

    }

    console.log("Login successful");

    console.log("user: " + user);

    const objToSend = {

      name: user.name,

      email: user.email,

    };

    console.log("objToSend: " + JSON.stringify(objToSend));


    return res.status(200).send(JSON.stringify(objToSend));

  } catch (error) {

    console.log(error);

    return res.status(500).json({ error: error.message });

  }

});
```

Figure 4.16: API endpoint for authorisation of manager login on server

4.6.2 Integration of external APIs

Not only were APIs developed for the system, but third-party APIs from external services and platforms, such as Google Maps and PayHere payment gateways, were also implemented in the system.

Google Maps, Places, and Directions APIs are integrated with the system, and developers are offered initial credits to test these services with applications before incurring charges.

PayHere provides a sandbox mode specifically for software development purposes, allowing developers to monitor transactional data through their portal (Figure 4.17).

They were configured following API documentation provided by those platforms and offer additional services such as tracking usage, generating reports, and providing valuable insights on utilising their services for a software solution (Figure 4.18).

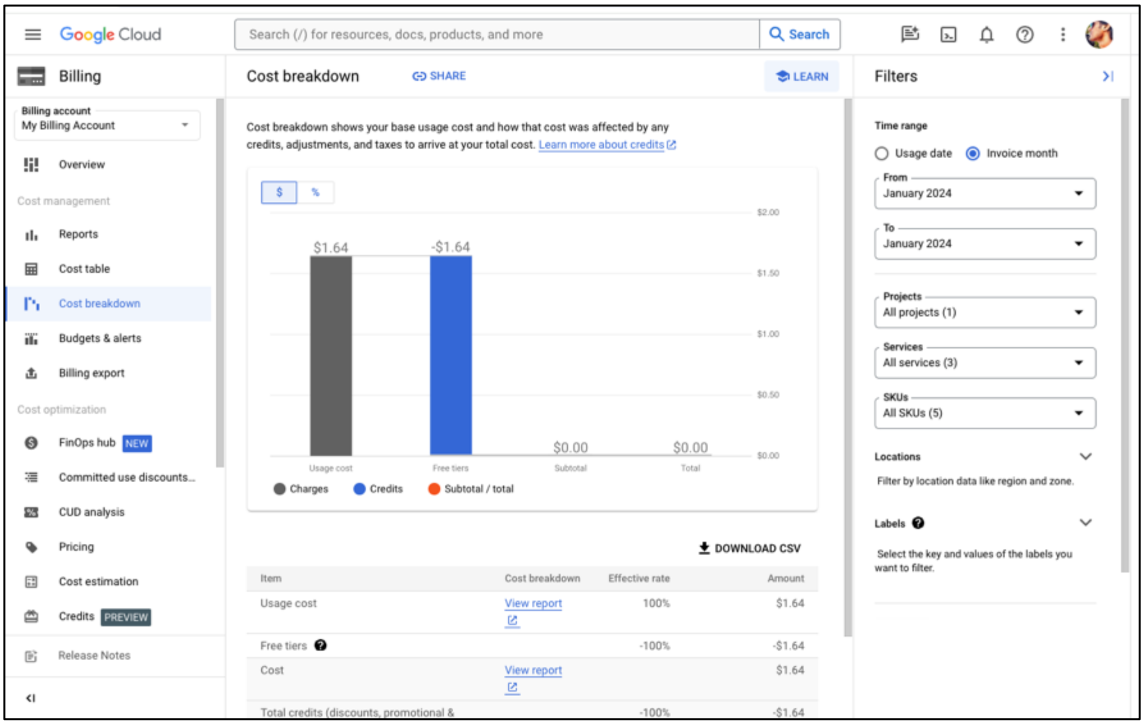


Figure 4.17: Google Cloud Console for managing API services

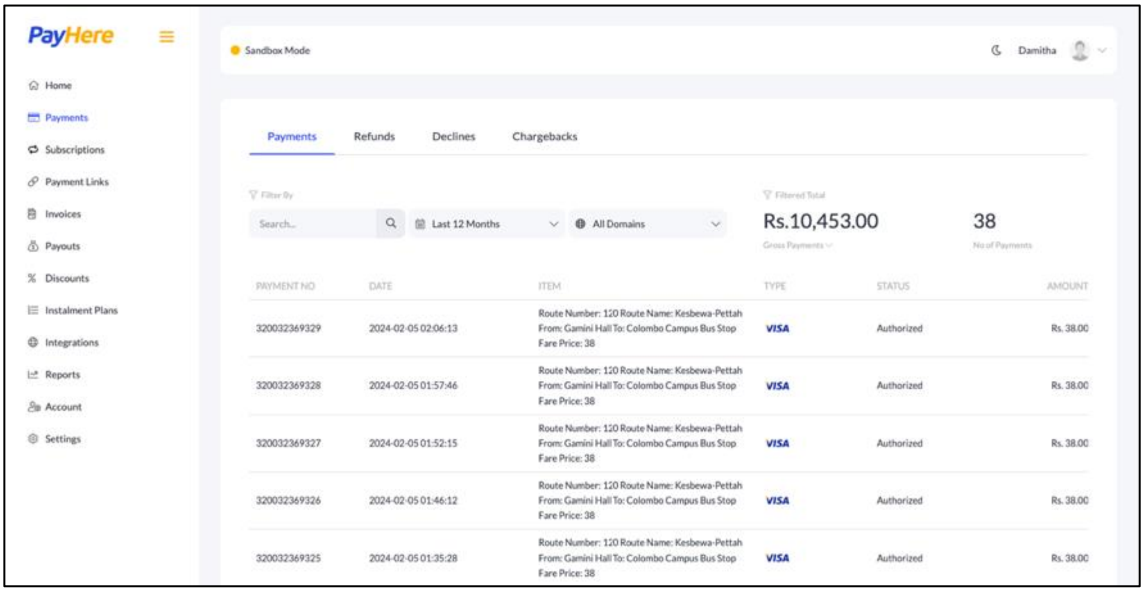


Figure 4.18: PayHere Sandbox merchant portal

4.6.3 Implementation of Database

The database is one of the most critical components of the system, and MongoDB has been selected to fulfil the database requirements. Schemas and relations are modelled based on the project's requirements and design specifications. MongoDB Cloud Atlas offers a user-friendly interface to visualize data collections and provides insights that aid in the development process (Figure 4.19).

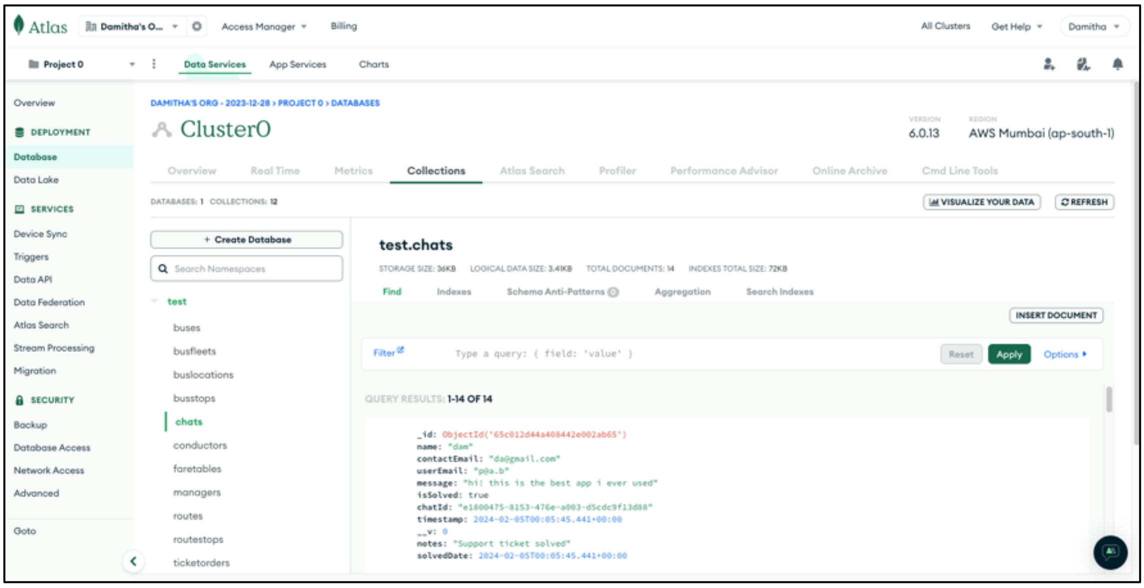


Figure 4.19: MongoDB cloud atlas dashboard

4.7 Version Controlling

GitHub was utilised extensively in the system for several reasons. Firstly, it served as a centralized platform for hosting the code repository, enabling access to the code from anywhere with an internet connection. Secondly, GitHub's version control features enabled efficient tracking of changes made to the codebase, ensuring the integrity and stability of the project. Additionally, GitHub facilitated code review processes through pull requests and branching mechanisms, enhancing code quality and enabling collaboration when needed (Figure 4.20).

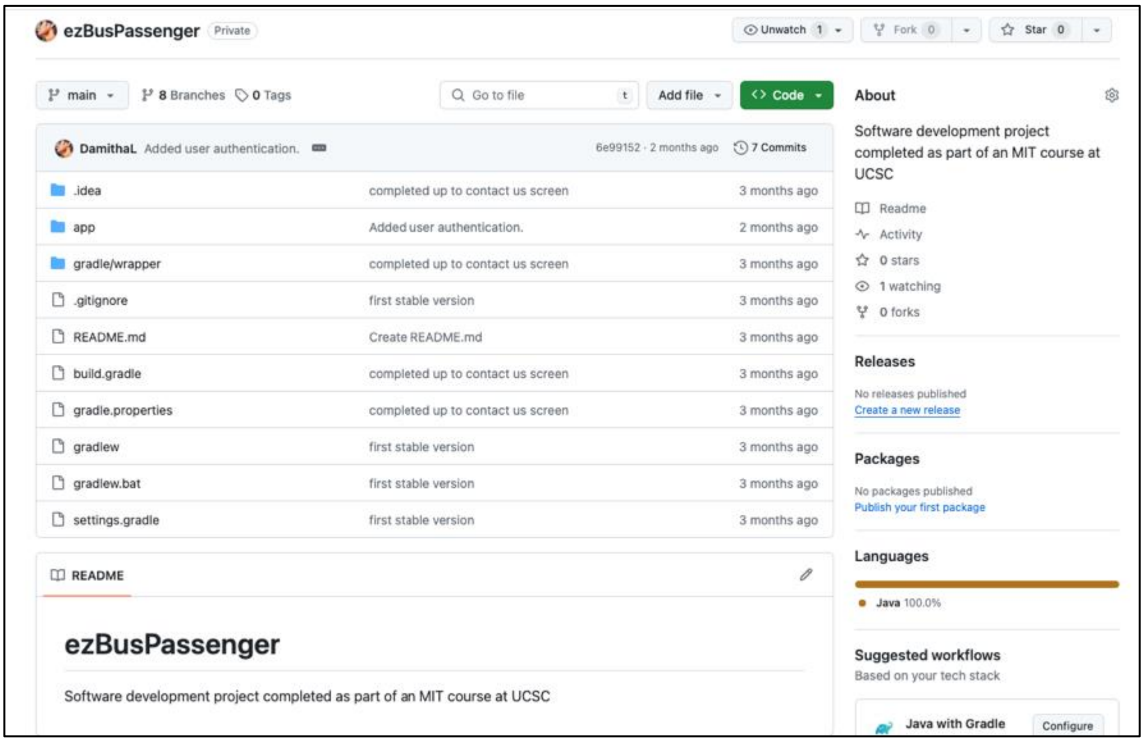


Figure 4.20: GitHub repository for Passenger application

4.8 Artificial Intelligence

Artificial intelligence tools such as GitHub Copilot and ChatGPT were utilised in the development process to streamline tasks and enhance project outcomes. GitHub Copilot offered assistance with code completion, suggesting improvements, and providing insights based on project requirements. This AI-powered tool significantly expedited coding tasks and improved code quality by understanding context and generating relevant code snippets. Additionally, ChatGPT was employed for brainstorming ideas, refining project documentation, and addressing development queries. Leveraging natural language processing capabilities, ChatGPT facilitated communication and provided creative solutions to various development challenges. Overall, these AI tools played a pivotal role in optimizing productivity and fostering innovation throughout the development process.

Chapter 5 – Testing and Evaluation

5.1 Introduction

Software testing is the systematic evaluation and verification process to ensure that a software product or application performs its intended functions. The advantages of testing extend to bug prevention, reduction of development costs, enhancement of security, effectiveness, reliability, and performance, thereby contributing to an overall improvement in the quality of the product and customer satisfaction. This chapter delves into the testing strategies utilised, the executed test cases, user evaluations, and the outcomes derived from the testing process.

The developed system consists of five main components: three Android applications designed for passengers, conductors, and managers respectively; a React JS web portal frontend tailored for administrators; and a Node.js backend server responsible for managing the entire system. Due to the wide range of technologies utilised and the diverse target user groups involved, testing and evaluation require a significant effort. However, effective planning has been implemented to manage the allocated time and resources, ensuring thorough testing to uphold the quality of the system.

5.2 Test Objective

The purpose of testing and the objective we aim to achieve is to validate that the system fulfils the user's requirements as described in Chapter 2, 'System Features' section, and to verify whether the system conforms to its specifications as described in Chapter 3, ‘Design’.

5.3 Scope

Testing and evaluation are planned for the development and implementation phases of the system. Software quality assurance activities for operations and maintenance of the system are not discussed here.

5.3.1 In-Scope

All core functionalities of the system will undergo comprehensive testing to ensure their correctness and adherence to non-functional requirements such as performance, security, and interoperability.

User testing was conducted as Alpha testing, involving a group of volunteers who worked directly with the developer for testing and evaluation.

5.3.2 Out-of-Scope

Some non-core functionalities or optional features that were not essential for the basic operation of the system were not tested due to time constraints.

Additionally, non-functional requirements such as scalability and availability were not tested due to limitations in resources and time.

Testing of third-party integrations such as payment gateways or external dependencies such as Google APIs was not conducted since they were not controlled or maintained by the developer.

User acceptance testing was not carried out since this was an individual project, and expected stakeholders were not directly participating in the project.

5.4 Testing Strategy

Static validation and verification techniques, such as Software Inspection & Review, were performed in the early stages of the development life cycle to identify any anomalies and defects in various representations of the system, including requirement specifications, software architecture, UML design models, and database schemas. However, detailed discussion of these techniques is not provided here, as the focus is on dynamic defect testing, which will be elaborated on in the following sections.

5.4.1 Identifying Risks

First, product risks have been systematically identified, considering their criticality and corresponding mitigation strategies. The main product risks identified along with their mitigation strategies are outlined below:

- **Functionality**

Risk: Ensuring all features of the entire system operate as intended under diverse conditions such as network fluctuations, different device types, and various user interactions.
Mitigation Strategy: Implement comprehensive unit, integration, and system testing protocols to validate functionality across different scenarios.

- **Usability**

Risk: Evaluating the system's ease of use, intuitiveness, and learnability for a wide range of users with varying levels of technical proficiency.
Mitigation Strategy: Thorough user experience (UX) testing involving representative user groups was conducted to gather feedback and refine the interface design. Incorporation of user-centered design principles to enhance usability and accessibility was evaluated.

- **Performance**

Risk: Measuring responsiveness, efficiency, and resource utilization under different loads and network speeds to ensure optimal system performance.
Mitigation Strategy: The applications were tested for performance using application performance monitoring (APM) metrics such as resource usage, error rates, and response times. Android Studio tools such as database and network inspectors, as well as Profiler for CPU and memory usage, were utilised for this testing. Following this, the codebase and infrastructure were optimized to enhance performance and minimize latency.

- **Security**

Risk: Verifying data confidentiality, integrity, and protection against unauthorized access or

manipulations to safeguard sensitive information.
Mitigation Strategy: Employ security testing methodologies such as code reviews to identify and address potential vulnerabilities. Implementation of encryption protocols and access control mechanisms were evaluated during this testing.

- **Compatibility**

Risk: Confirming seamless operation across different Android versions, devices, and screen sizes to ensure broad compatibility and consistent user experience.
Mitigation Strategy: Rigorous compatibility testing was conducted on a diverse range of Android devices and screen resolutions. Additionally, the utilization of responsive design principles and device-specific optimizations to enhance compatibility, such as the MVVM (Model-View-ViewModel) architecture that separates the presentation layer from the business logic, was ensured throughout all the Android front-end components.

- **Reliability**

Risk: Assessing the system's ability to function consistently and recover gracefully from errors or failures to maintain uninterrupted service availability.
Mitigation Strategy: Implement fault-tolerant design patterns to minimize downtime and ensure continuous operation.

5.4.2 Level of Testing

To mitigate the identified risks, the strategy involves a combination of manual and automated testing techniques tailored to the specific requirements of each component of the system across various levels. Here's a breakdown of the related test types and their corresponding levels.

- **Unit Testing**

Each individual module or component of the Android applications, web portal frontend, and backend server underwent unit testing. Unit tests were designed to validate the behaviour of isolated code units, ensuring that they functioned correctly according to specifications.

Unit testing initially began with manual testing to identify and optimize the test cases. Then, automated testing frameworks such as TestNG for Java-based components and Selenium for Web-based components were utilised to automate unit testing wherever possible.

Specific tools and software were utilised for certain components of the system. For example, the APIs developed for the system were tested using the Postman API testing platform. MongoDB Compass and MongoDB Atlas were also used to test database integrations. Additionally, manual unit testing was conducted during each component's coding phase.

- **Integration Testing**

Integration testing focused on verifying the interactions and interfaces between different modules and components of the system. Test scenarios were designed to assess the integration points between the Android applications, web portal frontend, and backend server.

- **Regression Testing**

Regression testing was performed after each functionality addition or system enhancement to ensure that existing functionalities remained intact and unaffected by changes.

- **System Testing**

System testing evaluated the system as a whole, including all Android applications, the web portal frontend, and the backend server. Test scenarios were designed to simulate real-world usage scenarios, covering various user interactions and system functionalities. System testing validated end-to-end functionality, performance, security, and usability aspects of the system.

- **User Testing**

User testing was thoroughly discussed in a later section, 5.6 - User evaluation, considering its importance and complexity.

5.5 Test Approach

In this section, the overall approach to testing the system is detailed, encompassing test design strategy, Execution Strategy, and how the identified defects were managed.

5.5.1 Test Design Approach

For the test design, a combination of techniques and methods suitable for the system's complexity and requirements was utilised. This included:

White Box Testing: The internal structure and logic of the system were examined to uncover potential defects and ensure code coverage, especially during unit testing, integration testing and regression testing.

Black Box Testing: While black box testing is typically conducted by testers without knowledge of the system's internal workings, in this project, the developer performed black box testing to ensure that the system met specified requirements without peering into its internal structures or workings. It was utilised in all levels of testing, and the techniques used for designing test cases in black box testing included:

Equivalence Partitioning: Dividing input data into partitions and selecting representative test cases from each partition to ensure thorough testing coverage.

Boundary Value Analysis: Testing the boundaries of input ranges to identify any potential issues at the edges of valid ranges.

State Transition Testing: Verifying the system's behaviour as it transitions between different states or modes.

Error Guessing: Deriving some tests from experience and observation of similar applications.

To identify the test cases in system testing, Use Case Testing was utilised. This technique involves validating the system's behaviour against predefined use cases to ensure that it meets user requirements.

5.5.2 Test Cases

To facilitate clear identification of the system's data flows and to enable easy tracing, maintenance, understanding, and searching of test cases, a mind map method was utilised (Figure 5.1).

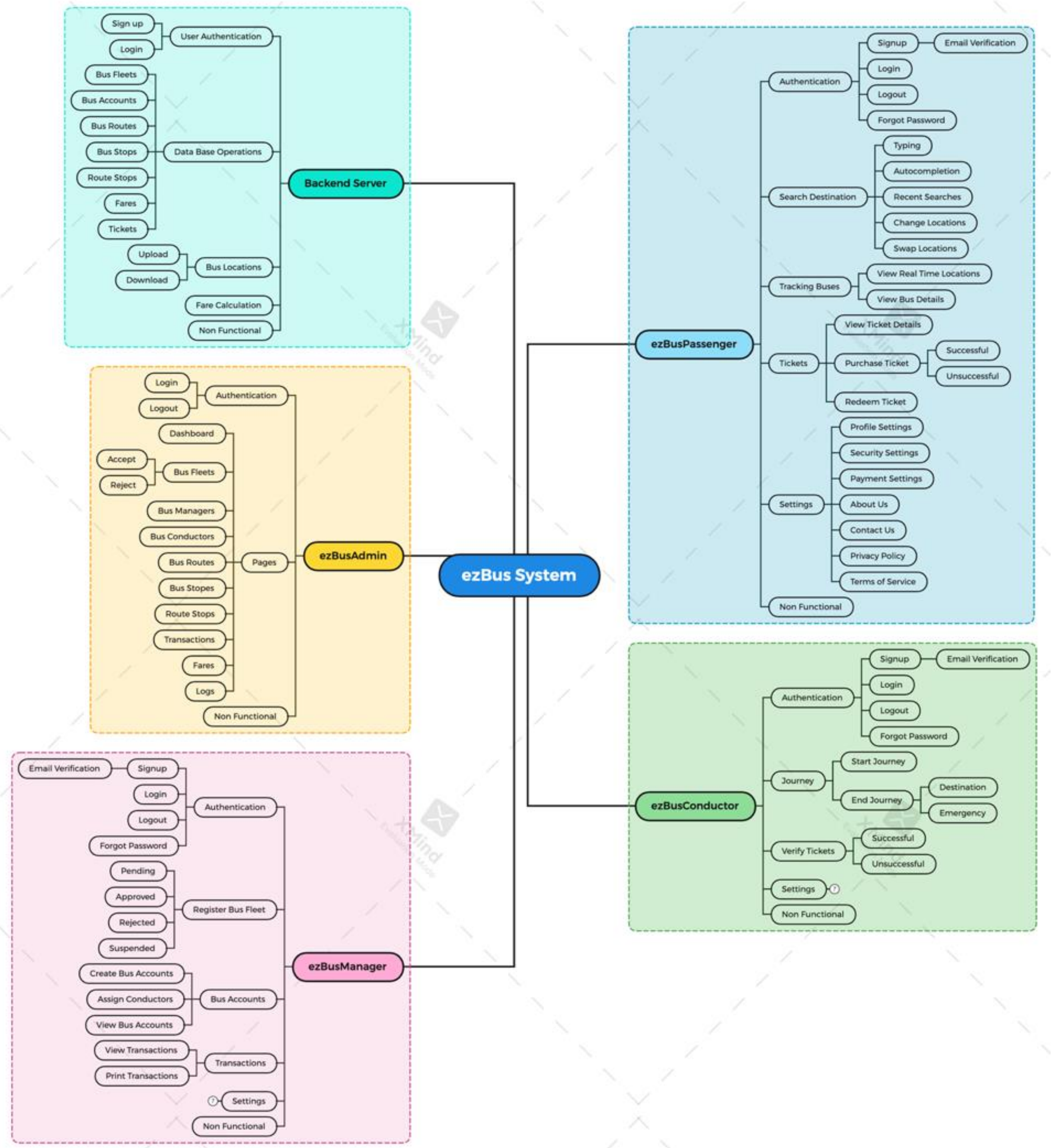


Figure 5.1: Mind map of the test case design

Subsequently, a dedicated set of test cases was developed for each system component. Below (Table 5.1) are examples of test cases specifically designed for the Passenger application, demonstrating how all test cases were designed.

Table 5.1: Sample Test Cases

TC ID	Test Objective	Pre-Conditions	Steps	Expected Result	Actual Result
EBP 01	User can login only with valid credentials.	1. Needs an internet connection. 2. Server should be running. 3. User is in the login screen.	1. User enters a valid username 2. User enters an incorrect password 3. User clicks on the login button	1. A success message should be displayed. 2. The user should be successfully logged into the application.	As expected.
EBP 02	Validate user login with incorrect input.	1. Needs an internet connection. 2. Server should be running. 3. User is in the login screen.	Either email or password is invalid.	An error message should be given as "Invalid Credentials."	As expected.
EBP 03	Verify the user email with the correct OTP.	1. The user should be logged into the system. 2. User in the profile update page.	1. The user should click verify email button. 2. Then enter OTP send to that email address.	If OTP is correct then the email is verified.	As expected.

A test case management tool was employed to facilitate the creation, organization, prioritization, execution, and tracking of test cases. TestCollab.io stands out as a comprehensive and user-friendly platform specifically crafted to streamline test case management processes. In addition to its intuitive interface, TestCollab.io boasts robust reporting features that offer insightful analytics on test coverage, execution status, and project progress. These capabilities empower testers to make informed decisions and efficiently prioritise testing efforts. Furthermore, TestCollab.io seamlessly integrates with Jira Atlassian for defect management, enhancing collaboration and ensuring seamless bug tracking across projects.

Below (Figure 5.2) is a depiction of how test cases are visualized within a well-organized structure.

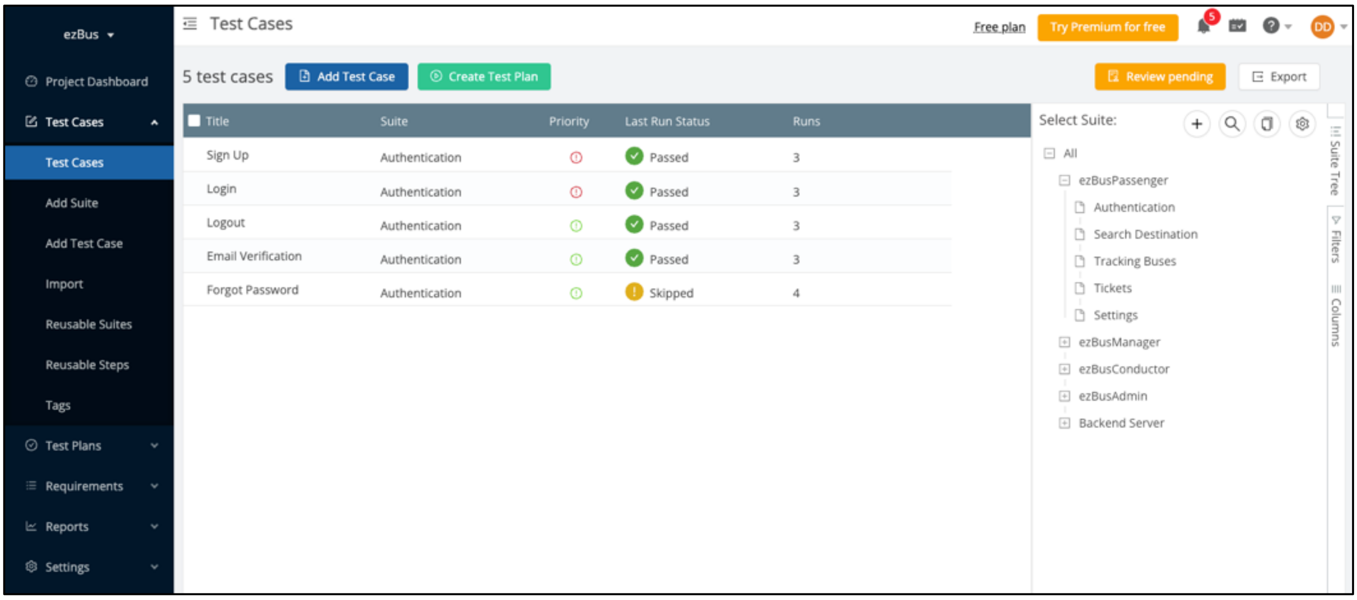


Figure 5.2: TestCollab.io test case dashboard

Test cases adhere to a standard format, providing clear descriptions including preconditions, steps, expected results, and priority levels. Below figure 5.3 shows how a test case is created on TestCollab.

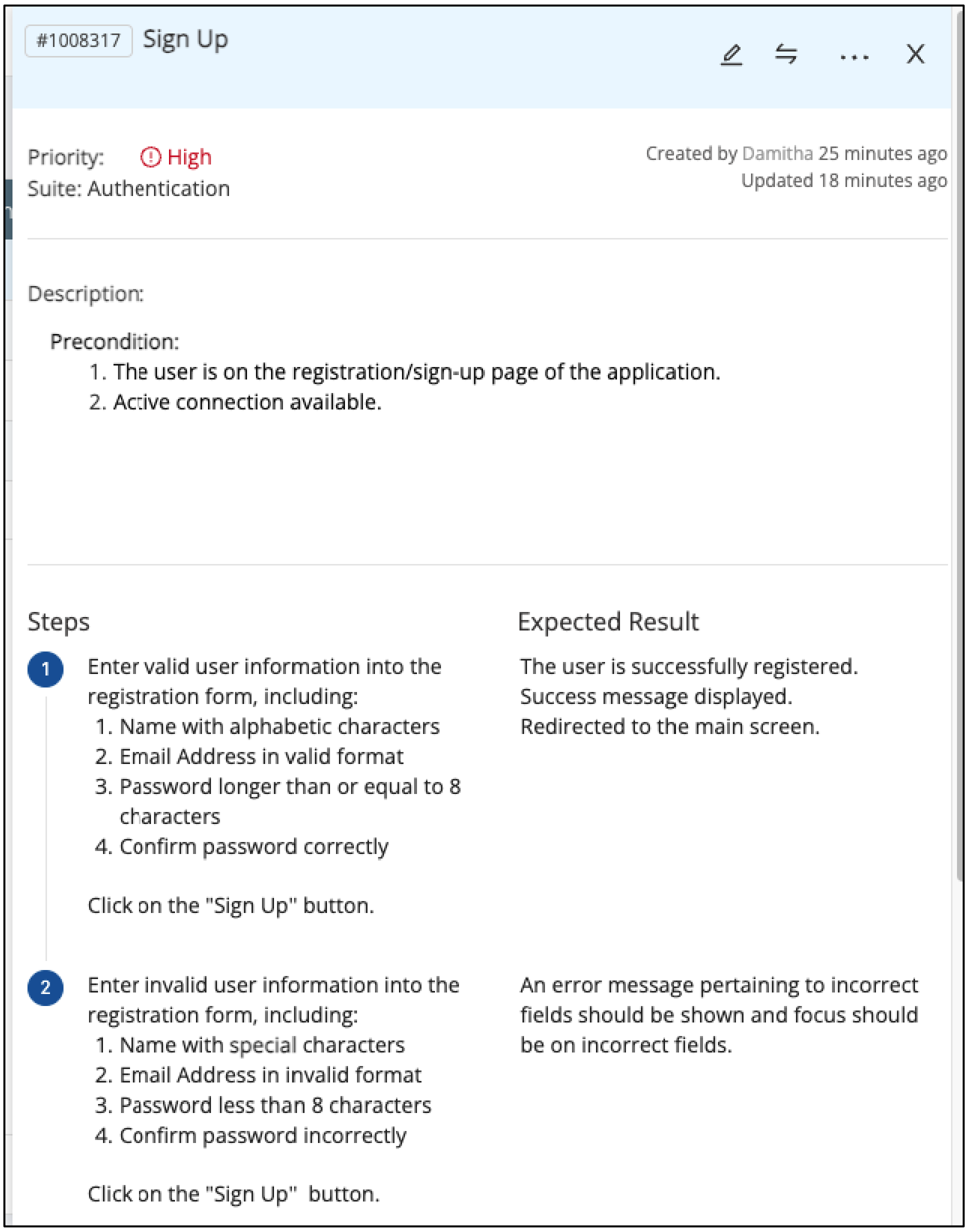


Figure 5.3: Sample test case

5.5.3 Execution Strategy

Executing design test cases involved two approaches:

Manual Testing

Test cases were executed manually to validate system behaviour and uncover defects. This approach was particularly emphasised during the early stages of development when the software was rapidly evolving. Manual testing was preferred during this phase because it was more efficient than investing time in creating and maintaining automated test scripts that could quickly become obsolete. Additionally, for user interface testing and complex scenarios that were difficult to automate effectively, manual testing was preferred.

Automated Testing

Automated testing tools and frameworks were employed to expedite testing efforts and increase test coverage whenever a properly designed test case required frequent execution. Specifically, TestNG was used for Java-based components, and Selenium was utilised for Web-based components.

This approach helped streamline the testing process, especially for repetitive or standardised test cases.

5.5.4 Defect Management

Each test case was executed when its entry criteria were met, and the results were thoroughly assessed. In cases where the actual result deviated from the expected result, a defect or bug was identified and reported for resolution. This was done using a standardised format, including information such as severity, priority, and steps to reproduce.

Jira issue tracking software was employed for enhanced management and tracking of defects. Bugs were assigned their own workflow within Jira (Figure 5.4), and their status was appropriately marked and tracked throughout the resolution process.

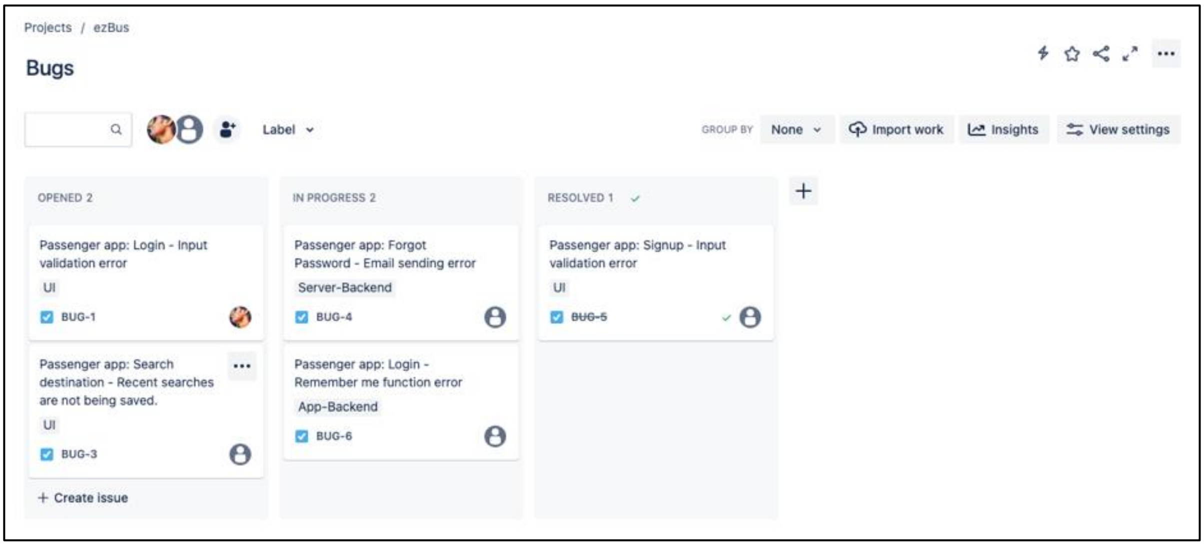


Figure 5.4: Jira workflow for bug issue tracking

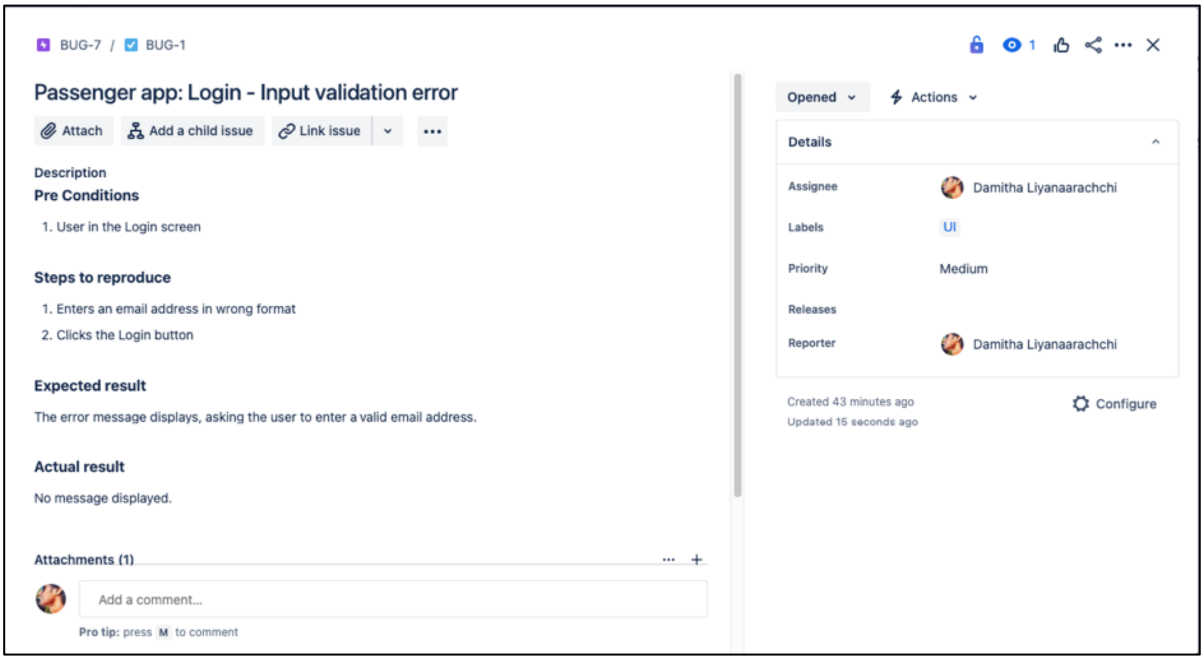


Figure 5.5: Bug issue details

The defects were prioritised based on their severity and impact on system functionality and user experience. Subsequently, the defects were investigated and addressed in the order of their priority. The underlying causes were resolved to prevent recurrence in future development efforts.

5.6 User Evaluation

User Evaluation is essential and crucial, even after comprehensive system testing has been carried out, as it uncovers how users perceive a system, which can differ from developers' expectations. Additionally, a user's working environment significantly impacts the reliability, performance, usability, and robustness of a system, elements that cannot be replicated in a testing environment. Ultimately, User Evaluation provides insights into how real users interacted with the system and offers feedback on its usability, functionality, and overall experience.

5.6.1 Approach

Although conducting User Acceptance Testing at key development milestones with real users was an ideal testing level for the system, since this was an individual project and some expected stakeholder groups were not directly participating, user testing was conducted as Alpha testing. This process involved a group of volunteers who collaborated directly with the developer for testing and evaluation at the end of the development phase. Feedback collected from the users was analysed and implemented to enhance the system, ensuring validation and re-evaluation of the system's functional and non-functional requirements.

5.6.2 Selection of Participants

User testing was planned to involve real users, including passengers, conductors, managers, and administrators, to validate the system's usability and meet their requirements. However, limitations were encountered in selecting volunteers from each group representing actual target user groups. For instance, involving administrators from the National Transport Commission (NTC) for testing purposes was impossible due to the lack of official collaborations with the NTC.

However, the help of a few volunteers, including 6 passengers, 2 managers, and 2 conductors, was enlisted for user testing. An Information Technology industry professional who volunteered to take on the role of an administrator tested the Administrator portal.

The main criteria for participant selection were the ability and willingness to spend a few hours using the designated system component and providing feedback. Participants were required to have an Android mobile phone (or a personal computer for administrator) and agree to install the designated application component on their phone (or computer) for the testing period, as well as have an active data connection plan.

No specific computer literacy or advanced knowledge of technology was expected for the mobile application users. However, understanding instructions in simple English language was considered.

5.6.3 Test Scenarios

Testing scenarios were designed to mimic typical user workflows and tasks, allowing users to provide feedback on the user interface, features, and overall user experience.

The designated mobile application component was installed on the personal mobile phones of several volunteers by the developer using USB debugging. For some volunteers, the specific application was sent as an APK file with instructions for installation. Additionally, test tasks and scenarios were provided for testing purposes.

Following are examples of test scenarios provided to a passenger user to test the Passenger application.

Registration:

Task: Create an account on the system.

Scenario: You are a new user who wants to use the system to track real time public bus locations and purchase virtual bus tickets for your upcoming trip. Please register for an account so you can access the features.

Searching Directions:

Task: Find the directions for a bus journey from City A to City B.

Scenario: You are looking to find the bus routes and real-time locations of available public buses from City A to City B. Please use the application to locate this information.

Purchasing a Ticket:

Task: Purchase a virtual bus ticket for a specific bus journey.

Scenario: You need to travel from City A to City B now. Please use the application to purchase a ticket for this journey.

Redeeming a Ticket:

Task: Display the QR code of a virtual bus ticket.

Scenario: The conductor of the bus requests to see your ticket, and you need to display the QR code of the ticket on your device to the conductor for verification. Please use the application to redeem a ticket for this journey.

Reset Password:

Task: Reset your account password.

Scenario: You have logged out from the application and you cannot remember the correct password. Now you have to reset the password in order to login to your account. Please reset your password and login to your account.

Contacting Customer Support:

Task: Reach out to customer support with a question.

Scenario: You have a question about updating the email address associated with your account. Please contact customer support using the application and ask for assistance with this issue.

5.6.4 Feedback

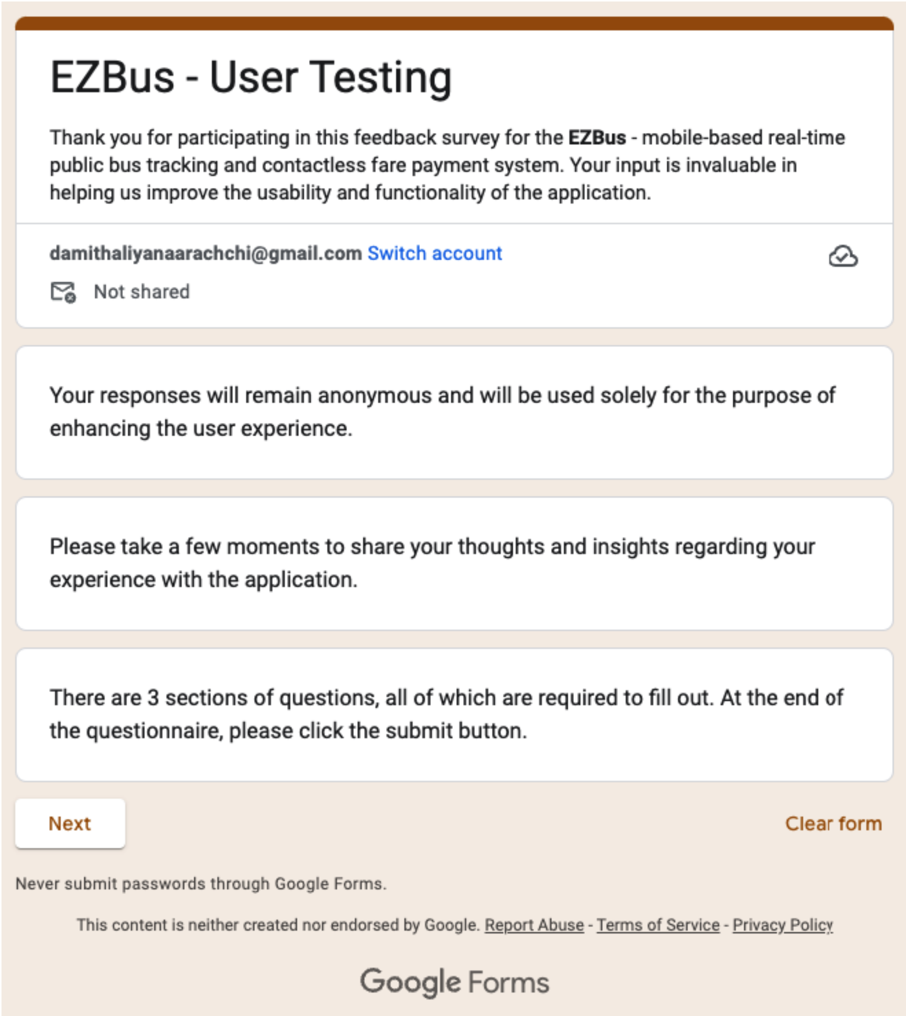
User testing was allocated a three-day period for users to conduct tests. Subsequently, feedback was collected through online questionnaires, face-to-face interactions, and phone interviews.

Likert scale questions and some open-ended inquiries were utilised to capture and clarify the user experience. In-person interviews were conducted with users who were geographically proximate, as this method is more effective for exploratory purposes, especially given the small number of participants. Additionally, in-person interviews allowed participants to ask questions, seek clarification, and provide detailed explanations of their experiences.

However, an online questionnaire was distributed to all users to ensure uniform data collection for better evaluation and visualization of the user experience.

The usability of the system was assessed using the 5E framework of usability, which encompasses five critical aspects essential for evaluating system effectiveness: Efficiency, Effectiveness, Engagement, Error Tolerance, and Ease of Learning (*Using the 5Es to Understand Users - Whitney Interactive Design*, no date). Each of these principles was addressed in the survey questions discussed below, aiming to comprehensively evaluate the user experience and identify areas for improvement based on the framework's guidelines. Most of them were rating questions, which were then assigned marks and evaluated collectively, and a few open-ended questions were included for further identification of improvement areas.

Below is the Google Form questionnaire that users were provided with and submitted after the testing period (Figure 5.6 – 5.14).



EZBus - User Testing

Thank you for participating in this feedback survey for the **EZBus** - mobile-based real-time public bus tracking and contactless fare payment system. Your input is invaluable in helping us improve the usability and functionality of the application.

damithaliyanaarachchi@gmail.com Switch account

Not shared

Your responses will remain anonymous and will be used solely for the purpose of enhancing the user experience.

Please take a few moments to share your thoughts and insights regarding your experience with the application.

There are 3 sections of questions, all of which are required to fill out. At the end of the questionnaire, please click the submit button.

Next

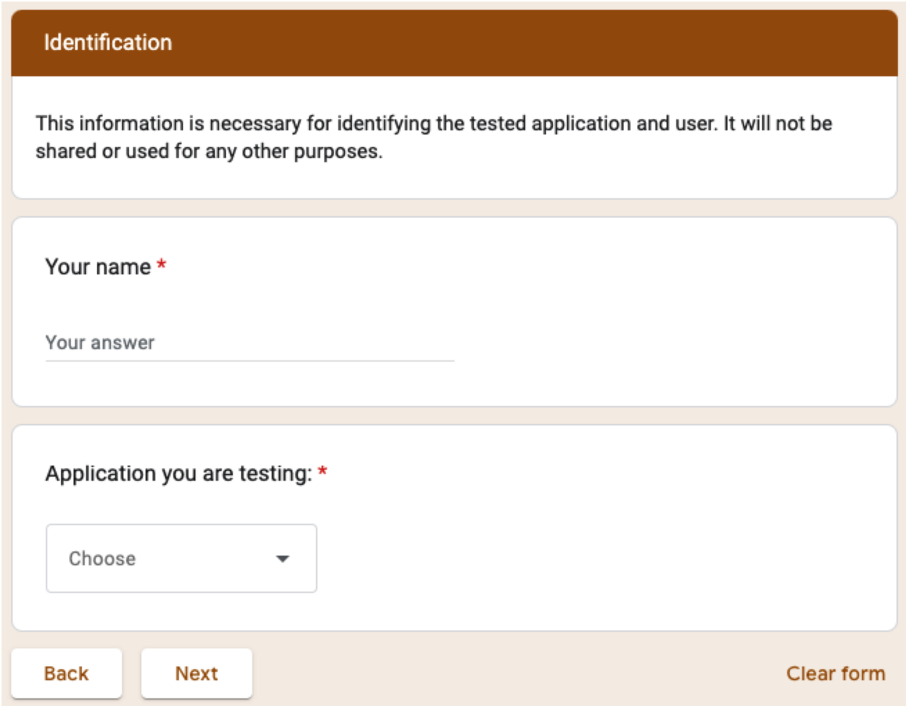
Clear form

Never submit passwords through Google Forms.

This content is neither created nor endorsed by Google. Report Abuse - Terms of Service - Privacy Policy

Google Forms

Figure 5.6: Online Questionnaire - Page 1



Identification

This information is necessary for identifying the tested application and user. It will not be shared or used for any other purposes.

Your name *

Your answer

Application you are testing: *

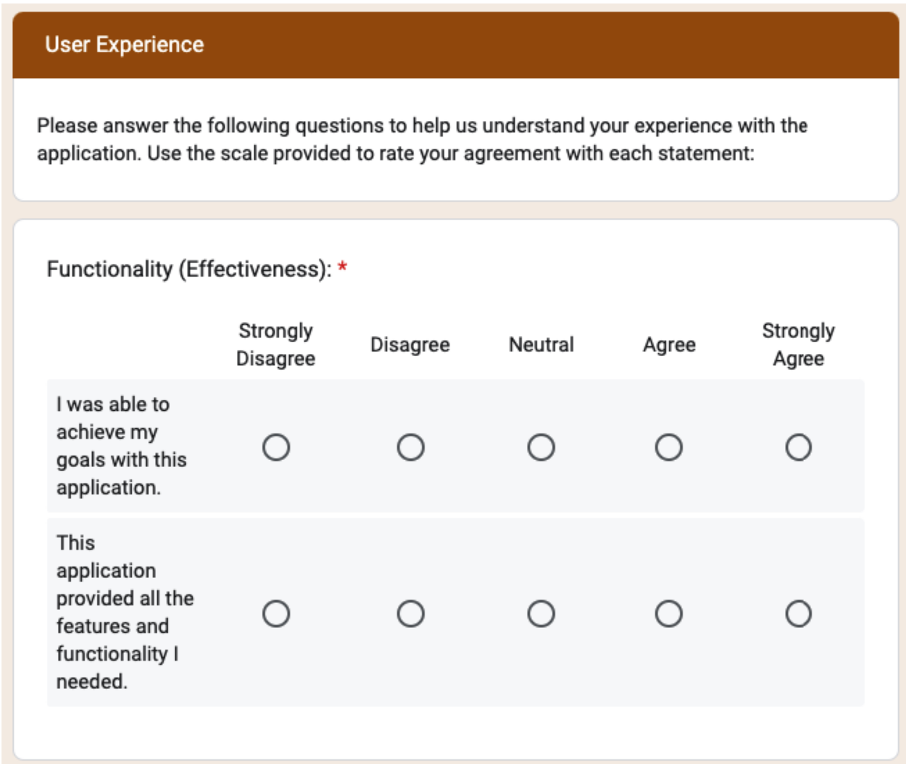
Choose

Back

Next

Clear form

Figure 5.7: Online Questionnaire - Page 2



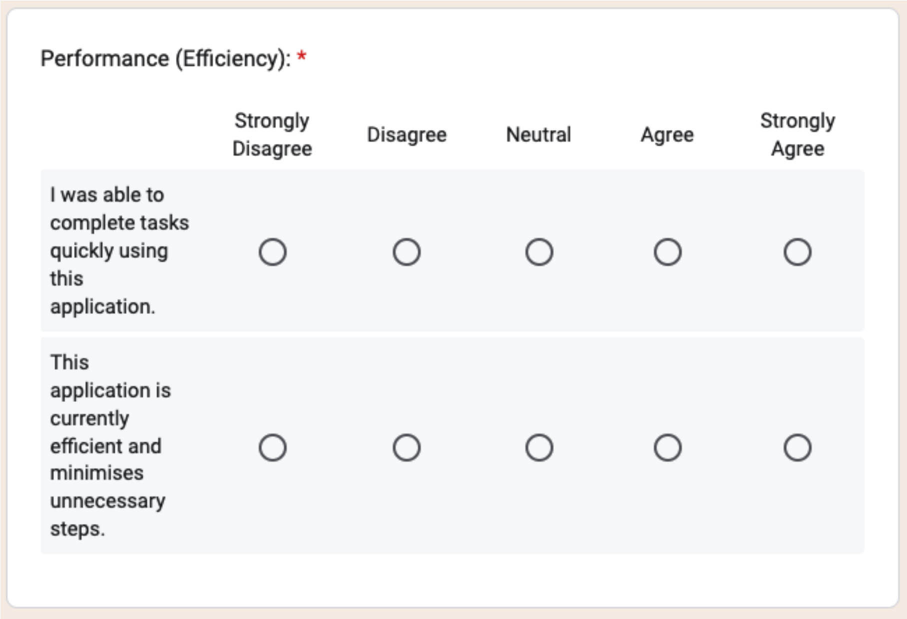
User Experience

Please answer the following questions to help us understand your experience with the application. Use the scale provided to rate your agreement with each statement:

Functionality (Effectiveness): *

	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
I was able to achieve my goals with this application.	○	○	○	○	○
This application provided all the features and functionality I needed.	○	○	○	○	○

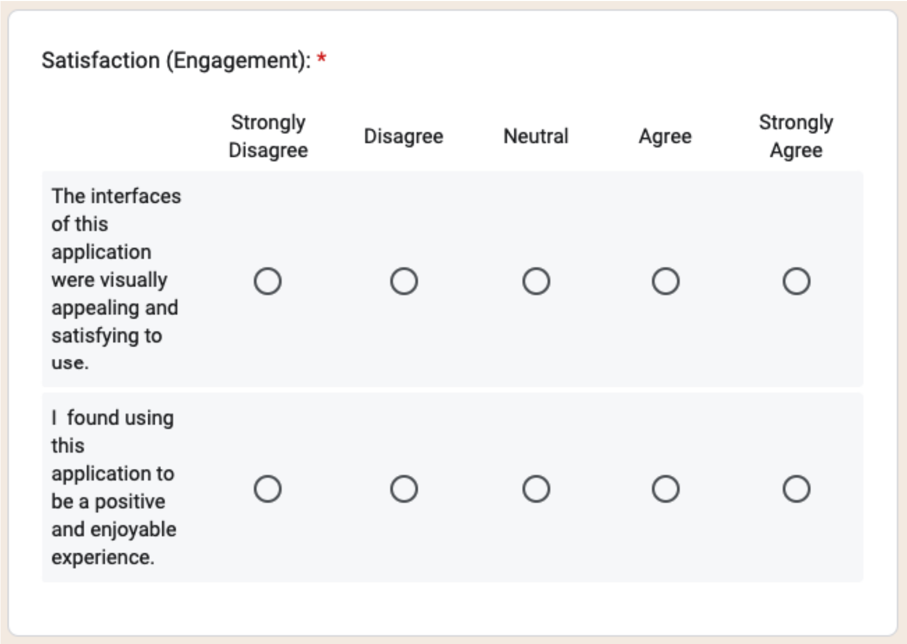
Figure 5.8: Online Questionnaire - Page 3 - Section 1



Performance (Efficiency): *

	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
I was able to complete tasks quickly using this application.	○	○	○	○	○
This application is currently efficient and minimises unnecessary steps.	○	○	○	○	○

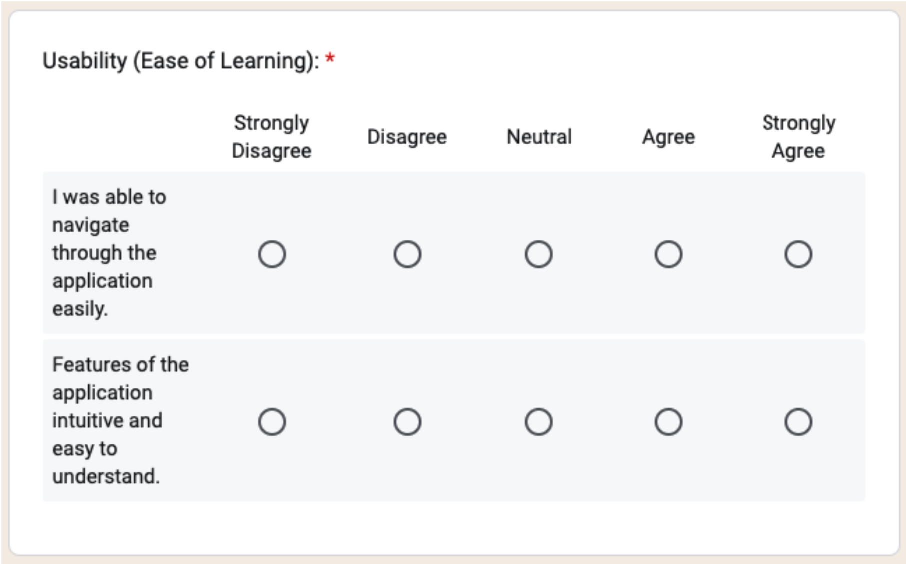
Figure 5.9: Online Questionnaire - Page 3 - Section 2



Satisfaction (Engagement): *

	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
The interfaces of this application were visually appealing and satisfying to use.	○	○	○	○	○
I found using this application to be a positive and enjoyable experience.	○	○	○	○	○

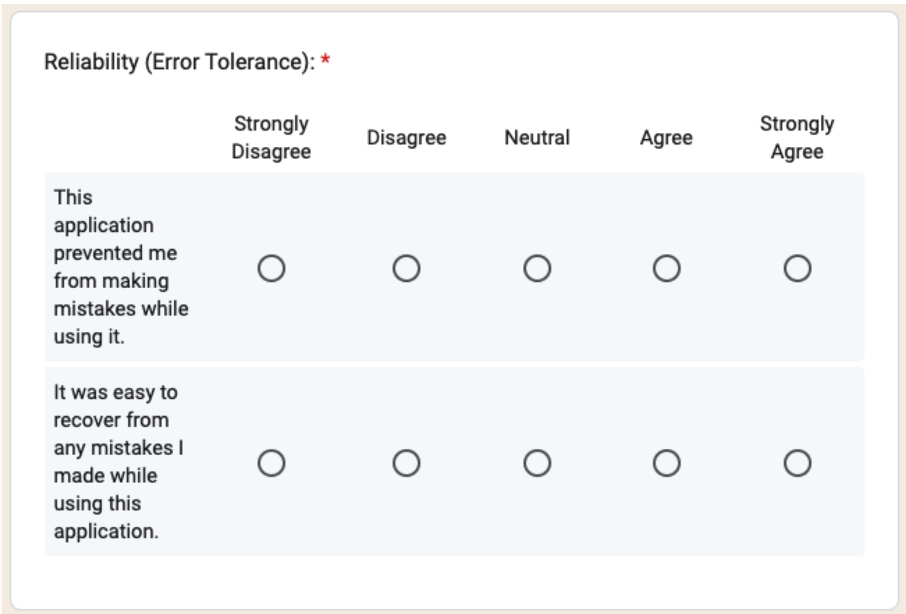
Figure 5.10: Online Questionnaire - Page 3 - Section 3



Usability (Ease of Learning): *

	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
I was able to navigate through the application easily.	○	○	○	○	○
Features of the application intuitive and easy to understand.	○	○	○	○	○

Figure 5.11: Online Questionnaire - Page 3 - Section 4



Reliability (Error Tolerance): *

	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
This application prevented me from making mistakes while using it.	○	○	○	○	○
It was easy to recover from any mistakes I made while using this application.	○	○	○	○	○

Figure 5.12: Online Questionnaire - Page 3 - Section 5

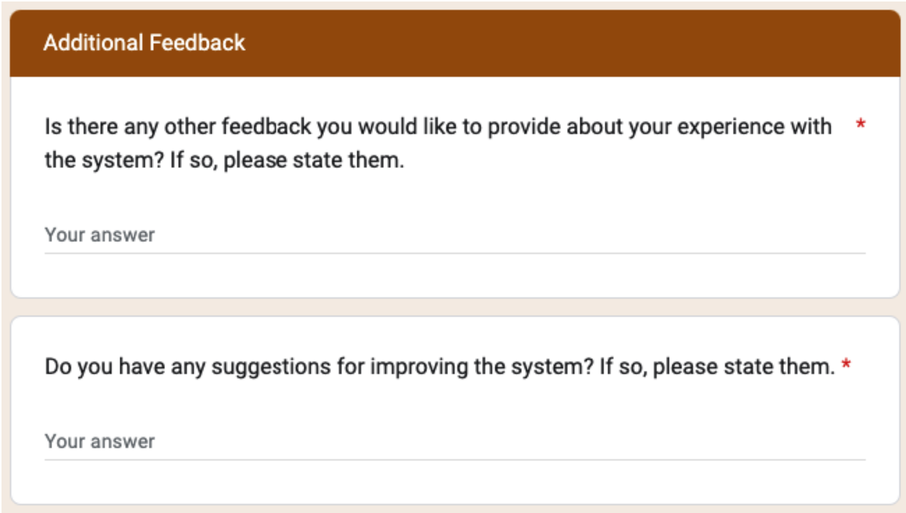


Figure 5.13: Online Questionnaire - Page 4

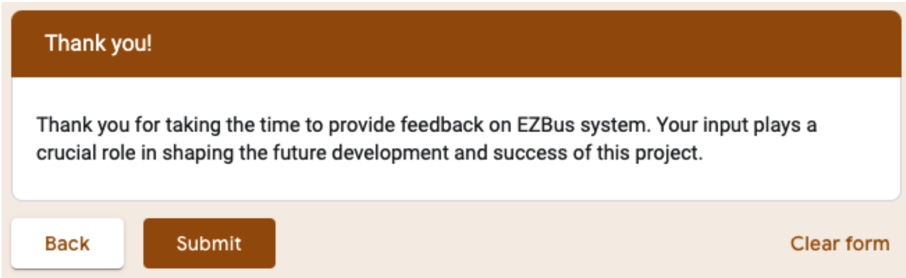


Figure 5.14: Online Questionnaire - Page 5

5.7 Results of the Testing

The results of each testing level, including unit testing, integration testing, regression testing, system testing, and user testing, were closely monitored and assessed to evaluate the software quality of the system. This evaluation aimed to ensure that the system performs its intended functions while adhering to non-functional requirements.

Initially, both manual and automated testing results were evaluated, and defects were prioritised and managed accordingly. Aiming for optimal defect resolution rates, efforts were made to maintain a 100% resolution rate for high and medium severity bugs, while keeping the resolution rate for low severity bugs above 95%. During certain test runs, some low-priority test cases were skipped to accommodate time constraints; however, in subsequent test runs, all test cases were included.

Following this, user testing was conducted as an alpha testing phase, involving 11 volunteers representing the four groups: passengers, conductors, managers and administrator. Feedback was collected through Google form questionnaire, incorporating Likert scale questions and a few open-ended inquiries.

The following are the results obtained for each of the usability criteria (Figure 5.15 – 5.20).

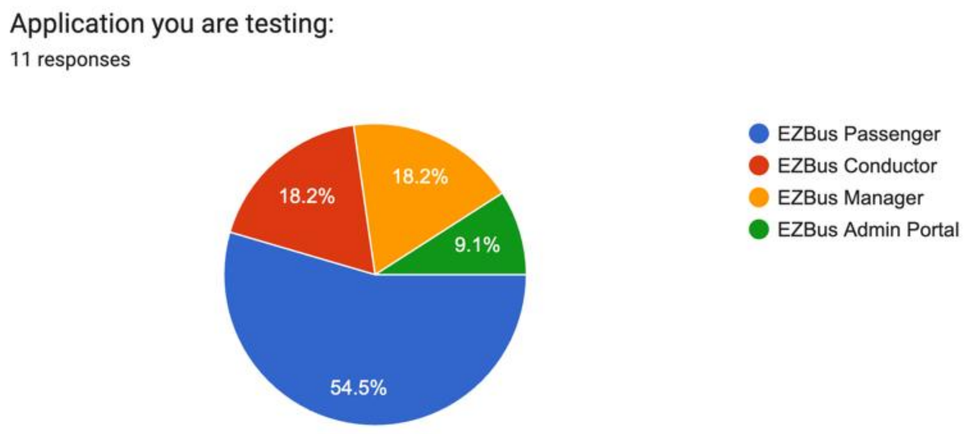


Figure 5.15: User evaluation feedback - Application type

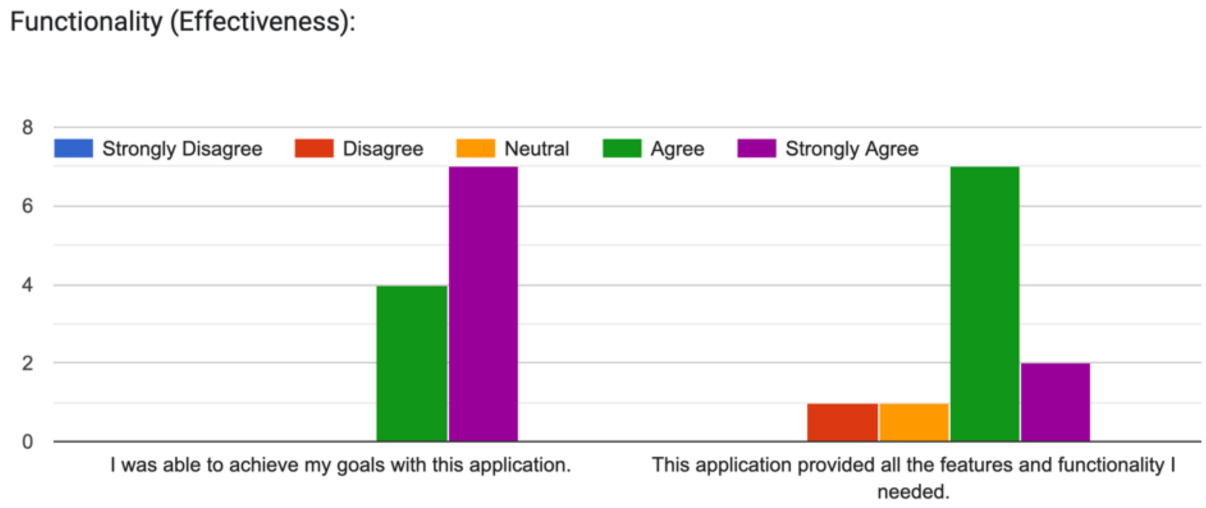


Figure 5.16: User evaluation feedback - Effectiveness

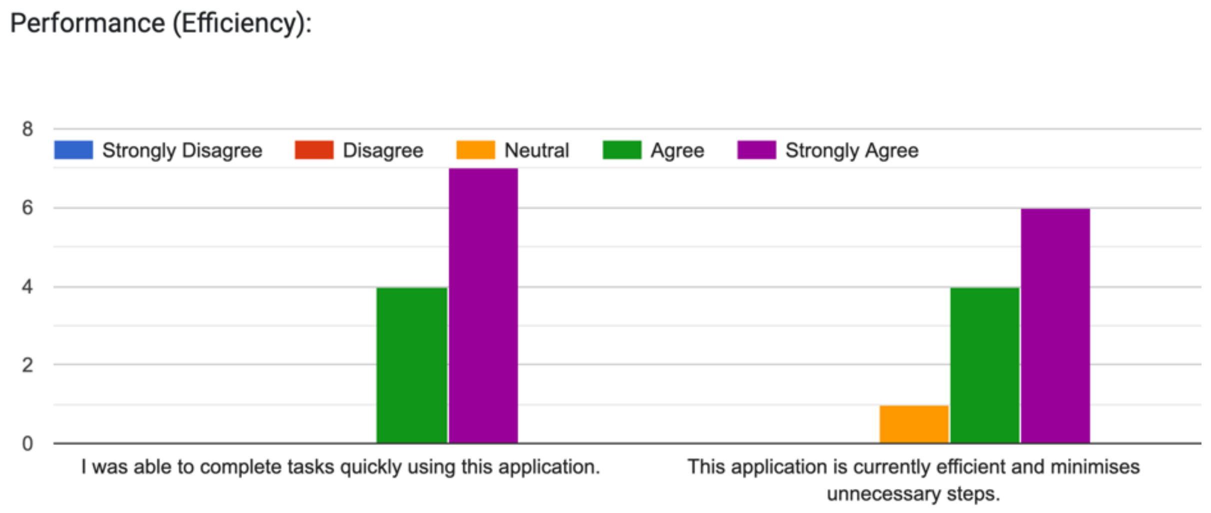


Figure 5.17: User evaluation feedback – Efficiency

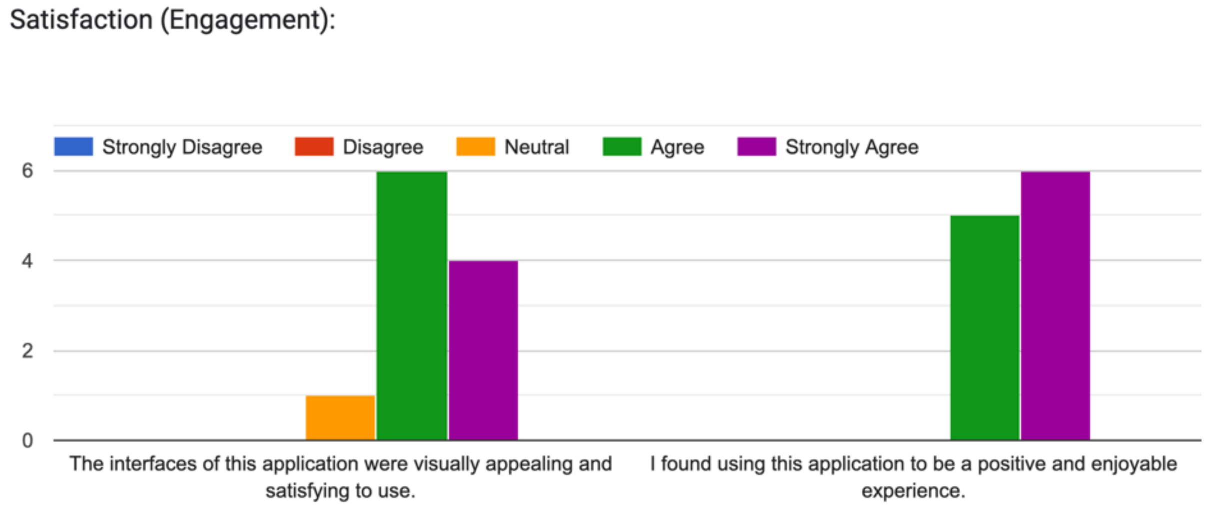


Figure 5.18: User evaluation feedback - Engagement

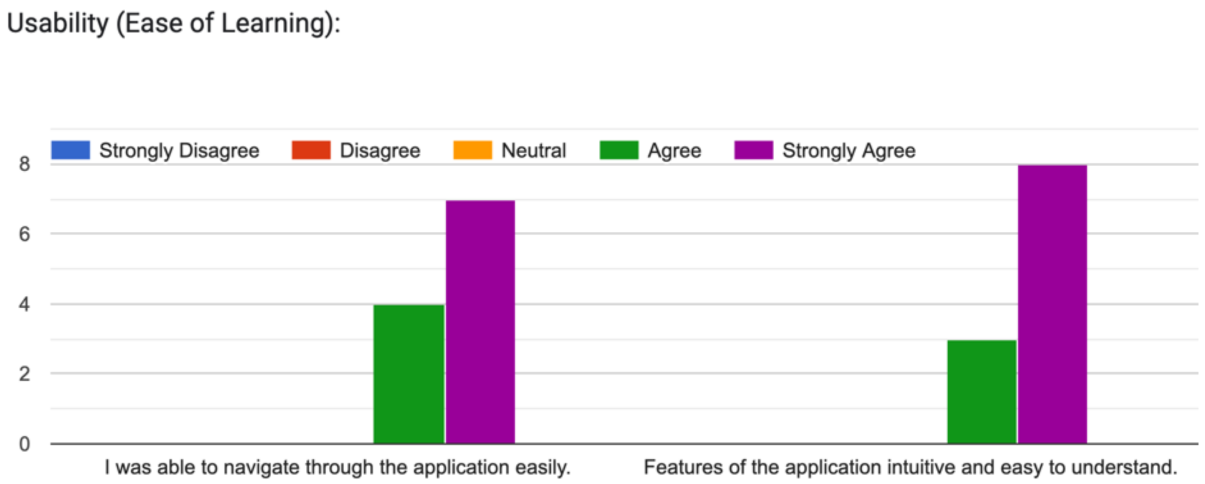


Figure 5.19: User evaluation feedback - Ease of learning

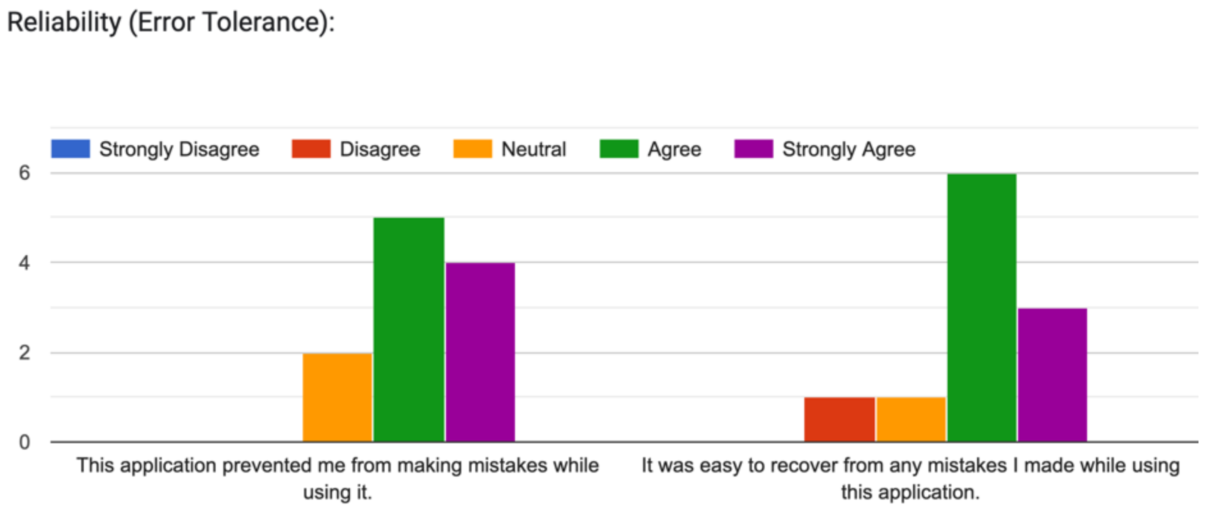


Figure 5.20: User evaluation feedback - Error tolerance

After evaluating the feedback, areas for improvement were identified, and additional clarifications were gathered through in-person and telephone interviews.

Some users pointed out that error messages and confirmation prompts before important actions were missing in the application. Additionally, some users suggested adding more features, such as displaying the number of passengers on a particular bus and notifying the passenger and conductor when the bus arrived at the destination bus stop. All suggestions were carefully analysed, and considering the scope of the project, modifications were prioritised according to time and resource constraints. Critical changes were implemented, resulting in overall system improvement.

Chapter 6 – Conclusion

6.1 Introduction

In this concluding chapter, the project's outcomes and journey are reflected upon. Achievements and contributions are assessed, recognizing both successes and challenges encountered along the way. Additionally, potential future directions for the project are discussed, considering avenues for further development and improvement. This chapter offers a comprehensive review of the project's progress and explores what lies ahead.

6.2 Project Outcomes

It is satisfactory to affirm that the project has successfully achieved its expected objectives and outcomes. However, to fulfil the main purpose of the project, it is crucial that the system be disseminated among the general public and other target users for extensive usage and testing. This widespread adoption will enable the collection of user feedback, which will serve as valuable input for further improvements and iterations. Through continuous refinement based on user feedback, the project aims to achieve its ultimate goal of popularizing public transportation and encouraging the adoption of a more sustainable mode of transportation.

The initiative of promoting public transportation usage can have a positive impact on the environment, economy, and society associated with transportation. By reducing reliance on individual vehicles, the project can contribute to mitigating carbon emissions, alleviating traffic congestion, and conserving natural resources. Additionally, increased utilization of public transportation can lead to economic benefits by reducing transportation costs for individuals and businesses alike, thus stimulating economic growth. Moreover, by promoting a culture of sustainable transportation, the project can foster social equity and enhance overall community well-being.

6.3 Achievements

The most significant and rewarding achievement of the project lies in the invaluable learning experience it provided. Through hands-on practice of software engineering principles and theories, the project served as a platform to apply and reinforce the knowledge acquired during the postgraduate course. Beginning with the software development life cycle (SDLC), the project journey commenced with learning how to analyse and evaluate requirements through observations, discussions, and further research. Subsequently, requirements were meticulously specified and evaluated using feasibility and viability analysis to determine implementable solutions. The translation of these requirements into detailed technical designs required meticulous consideration and adherence to design principles to minimize risks and future efforts on defect resolution. The implementation phase demonstrated the power of perseverance and commitment, starting from basic code snippets like "print("hello world")" to the development of complex codebases across various technological stacks. Furthermore, the collaborative relationships fostered with supervisors, lecturers, and colleagues throughout the project proved to be immensely valuable. In summary, the project's achievements aptly reflect the dedication and effort invested in its realization.

6.4 Challenges and Difficulties

The initial challenge encountered was conducting research on a solution that had not been successfully implemented despite previous attempts by the government. Identifying the shortcomings of previous efforts and devising solutions presented a significant hurdle. Furthermore, obtaining adequate support for an individual project of this nature was lacking from both society and government institutions. The magnitude of software development necessitates collaboration and team efforts, which were often felt to be essential. Some development errors arose from scarce documentation or community support, requiring extensive research and trial and error to resolve. While open-source solutions were available for most requirements, proprietary solutions were occasionally better suited to project needs, necessitating efficient management to meet trial durations or make purchases when necessary. The most daunting challenge was balancing time between learning new concepts from scratch, academic studies, professional commitments, and personal responsibilities while adhering to project milestones.

Despite these obstacles, each challenge and difficulty provided valuable experience and opportunities for skill improvement and practical application.

6.5 Future Work

Within the project scope, certain requirements and features were overlooked due to resource and time constraints. However, to facilitate the widespread adoption of this system among target users, revisiting some of these out-of-scope requirements will be necessary. For instance, expanding geographical coverage to encompass the entire country and incorporating support for Sinhala and Tamil languages are essential considerations. Expanding the system to include other mobile platforms like iOS and introducing more accessible features such as audio commands for elderly or differently-abled individuals. Additionally, if the system gains traction, there is potential for integration with other modes of transportation, such as trains. There is ample room for improvement within the system, which should be pursued concurrently with the evaluation of the user base and their feedback.

References

AWS Case Study: Moovit (no date). Available at: https://aws.amazon.com/solutions/case-studies/moovit/ (Accessed: 21 June 2023).

Banister, D. (2008) 'The sustainable mobility paradigm', *Transport Policy*, 15(2), pp. 73–80. Available at: https://doi.org/10.1016/J.TRANPOL.2007.10.005.

Beirão, G. and Sarsfield Cabral, J.A. (2007) 'Understanding attitudes towards public transport and private car: A qualitative study', *Transport Policy*, 14(6), pp. 478–489. Available at: https://doi.org/10.1016/J.TRANPOL.2007.04.009.

Bian, J. *et al.* (2021) 'The end-user benefits of smartphone transit apps: a systematic literature review', *https://doi.org/10.1080/01441647.2021.1950864*, 42(1), pp. 82–101. Available at: https://doi.org/10.1080/01441647.2021.1950864.

Canadian Urban Transit Association (2014) 'Submission to the Review of the Canada Transportation Act'.

Department of Census and Statistics Ministry of Finance, E.S. and N.P.S.L. (2022) *Statistical Pocket Book – 2022 Sri Lanka*. Available at: www.statistics.gov.lk (Accessed: 22 June 2023).

E-Pay system for Bus Travel (no date). Available at: https://www.newsfirst.lk/2022/08/19/e-pay-system-for-bus-travel/ (Accessed: 10 May 2023).

Farag, S. and Lyons, G. (2008) 'What Affects Use of Pretrip Public Transport Information?', *https://doi.org/10.3141/2069-11*, (2069), pp. 85–92. Available at: https://doi.org/10.3141/2069-11.

Ferris, B., Watkins, K. and Borning, A. (2010) 'OneBusAway: Results from providing real-time arrival information for public transit', *Conference on Human Factors in Computing Systems - Proceedings*, 3, pp. 1807–1816. Available at: https://doi.org/10.1145/1753326.1753597.

Google Maps - Apps on Google Play (no date). Available at: https://play.google.com/store/apps/details?id=com.google.android.apps.maps (Accessed: 24 June 2023).

'Guide to the Systems Engineering Body of Knowledge (SEBoK), version 2.8 Guide to the Systems Engineering Body of Knowledge version 2.8' (no date). Available at:

www.sebokwiki.orgPDFgeneratedusingtheopensourcemwlibtoolkit.Seehttp://code.pediapress.com/formoreinformation. (Accessed: 13 August 2023).

How Moovit turns data into insights to help passengers avoid delays using Apache Airflow and Amazon SageMaker | AWS Machine Learning Blog (no date). Available at: https://aws.amazon.com/blogs/machine-learning/how-moovit-turns-data-into-insights-to-help-passengers-avoid-delays-using-apache-airflow-and-amazon-sagemaker/ (Accessed: 21 June 2023).

Initial project of Google Transit facility launched - Breaking News | Daily Mirror (no date). Available at: https://www.dailymirror.lk/breaking_news/Initial-project-of-Google-Transit-facility-launched/108-171016 (Accessed: 10 May 2023).

Lawson, C.T. *et al.* (2019) ‘Translation software: An alternative to transit data standards’, *Transportation Research Interdisciplinary Perspectives*, 2, p. 100028. Available at: https://doi.org/10.1016/J.TRIP.2019.100028.

Litman, T. and Burwell, D. (2006) ‘Issues in sustainable transportation’, *International Journal of Global Environmental Issues*, 6(4), pp. 331–347. Available at: https://doi.org/10.1504/IJGENVI.2006.010889.

Ministry of Transport, S.L. (2014) ‘Urban Transport System Development Project - Final Report’.

Mobile Operating System Market Share Sri Lanka | Statcounter Global Stats (no date). Available at: https://gs.statcounter.com/os-market-share/mobile/sri-lanka (Accessed: 24 August 2023).

moovit - Android Apps on Google Play (no date). Available at: https://play.google.com/store/search?q=moovit&c=apps (Accessed: 24 June 2023).

Moovit: MaaS Solutions & the #1 Urban Mobility App (no date). Available at: https://moovit.com/ (Accessed: 24 June 2023).

mTrack (no date). Available at: http://mtrack.mbs.mobitel.lk/gps/login.aspx (Accessed: 24 June 2023).

mTrack | Mobitel (no date). Available at: https://www.mobitel.lk/mtrack (Accessed: 24 June 2023).

mybus-sl - Android Apps on Google Play (no date). Available at: https://play.google.com/store/search?q=mybus-sl&c=apps (Accessed: 24 June 2023).

National Transport Commission (2020) *Annual Report 2020*.

National Transport Commission (2022) *National Transport Statistics 2022*.

pickme - Android Apps on Google Play (no date). Available at: https://play.google.com/store/search?q=pickme&c=apps (Accessed: 24 June 2023).

PickMe Driver (Sri Lanka) - Apps on Google Play (no date). Available at: https://play.google.com/store/apps/details?id=com.pickme.driver.byod (Accessed: 24 June 2023).

Schiller, P.L. and Kenworthy, J.R. (2010) *An introduction to sustainable transportation : policy, planning and implementation*. Available at: https://www.routledge.com/An-Introduction-to-Sustainable-Transportation-Policy-Planning-and-Implementation/Schiller-Kenworthy/p/book/9781138185487 (Accessed: 22 June 2023).

Song, C., Guan, W. and Ma, J. (2018) 'Potential travel cost saving in urban public-transport networks using smartphone guidance', *PLOS ONE*, 13(5), p. e0197181. Available at: https://doi.org/10.1371/JOURNAL.PONE.0197181.

Touch Travel (no date). Available at: https://www.dialog.lk/touch-travel (Accessed: 10 May 2023).

uber - Android Apps on Google Play (no date). Available at: https://play.google.com/store/search?q=uber&c=apps (Accessed: 24 June 2023).

Uber - Driver: Drive & Deliver - Apps on Google Play (no date). Available at: https://play.google.com/store/apps/details?id=com.ubercab.driver (Accessed: 24 June 2023).

Universal Declaration of Human Rights | United Nations (1948). Available at: https://www.un.org/en/about-us/universal-declaration-of-human-rights (Accessed: 26 September 2024).

Using the 5Es to Understand Users - Whitney Interactive Design (no date). Available at: https://wqusability.com/articles/getting-started.html (Accessed: 27 February 2024).

Zhang-Kennedy, L. *et al.* (2023) 'Passenger Perceptions, Information Preferences, and Usability of Crowding Visualizations on Public Displays in Transit Stations and Vehicles', *Conference on Human Factors in Computing Systems - Proceedings* [Preprint]. Available at: https://doi.org/10.1145/3544548.3581241.

Appendix A – MIS Reports

1. Introduction

A management information system (MIS) is a system utilized for decision-making and for organizing, controlling, analysing, and visualizing information within an organization. The exploration of MIS encompasses individuals, procedures, and technology within an organizational framework, essentially facilitating control, planning, and decision-making at the management level. Growing the value and revenues of the organization is the ultimate objective of implementing management information systems in a business setting.

In relation to this system, MIS reports are valuable for a wide range of stakeholders, including not only the administrators of the system, who would ultimately be the National Transportation Commission (NTC), and bus owners or managers who operate their bus fleets through the system, but also passengers. In the subsequent sections, it will be discussed how all the stakeholders benefit from the MIS report system of this application, with necessary examples.

2. For Passengers' Decision-Making Needs

Starting from the smallest to the largest organizations of users, passengers, who may use public transportation for personal or business needs, can utilize this application system to keep track of their transportation costs for managing expenses or to provide evidence of transportation costs to their employers. In either scenario, the purchase of digital tickets containing dates, amounts, and details of redemption will render them useful.

The following screenshot (Figure A.8.1) illustrates how a passenger can check their ticket purchase details.

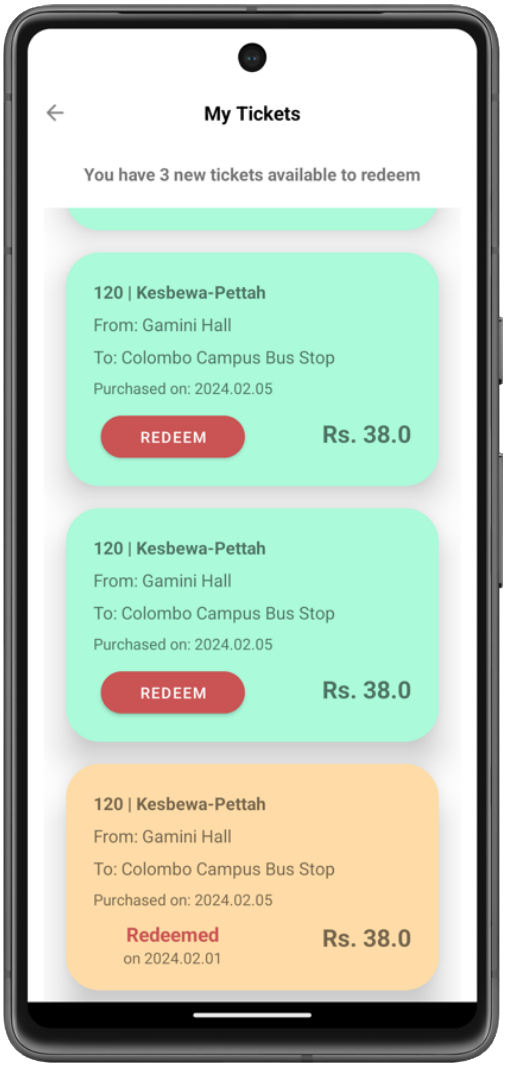


Figure A.1: Purchased ticket details on a Passenger application

3. For Managers' Decision-Making Needs

Bus managers, tasked with overseeing a bus fleet comprising a specific number of bus accounts, have access to information detailing the revenue generated by each bus. When a passenger presents a digital ticket to a bus conductor, it is scanned using the application, and the server validates and redeems the ticket if it's legitimate. Subsequently, the ticket amount is transferred to the bus fleet account, with the intention of transferring the revenue to the bank account of the bus manager on a monthly or specified basis. The ability to display these revenue details is crucial for bus fleet organizations to make informed decisions regarding bus scheduling and service optimization. Transaction details are meticulously organized and sorted by the date of ticket redemption, the bus's name, and the ticket amount. To facilitate decision-making, trends in revenue for each bus are visually represented in chart form for bus managers. These visualizations enhance the insightfulness and effectiveness of managerial decisions.

The revenue details are not the only information provided; the organization details of each bus account are also presented to the bus manager in a tabular form. This allows them to create and manage bus accounts more efficiently.

The screenshots (Figure A.2 – A.4) below illustrate how bus account details, transaction details and revenue trends are presented in the bus manager application.

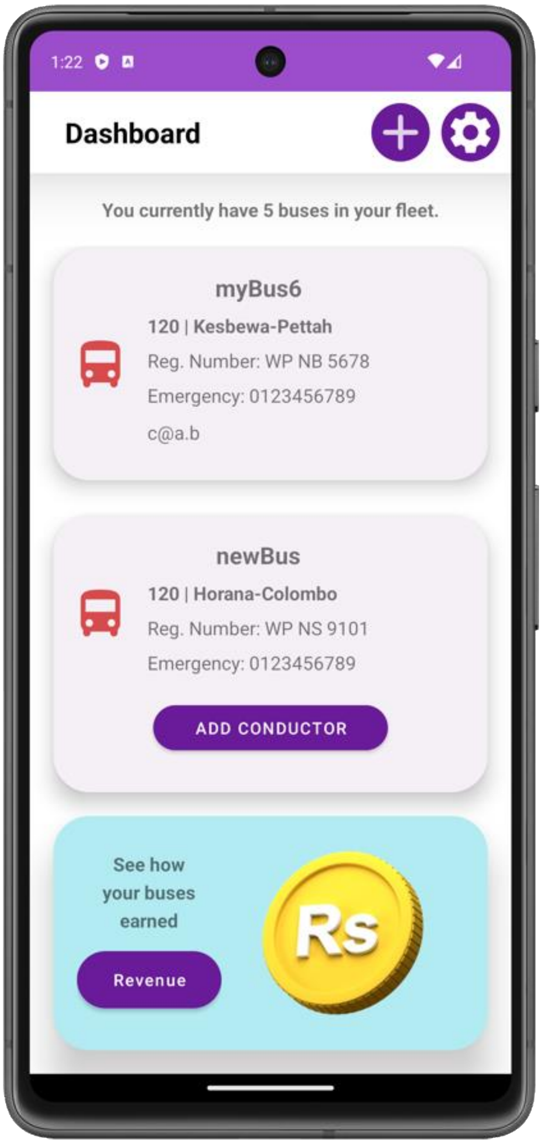


Figure A.2: Bus accounts details on a Manager application

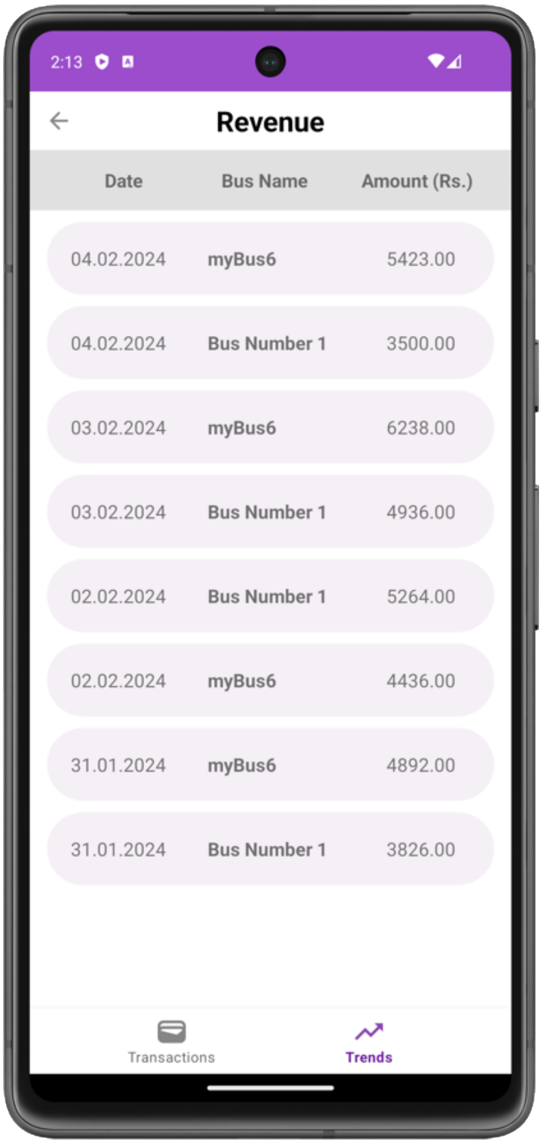


Figure A.3: Revenue trends on a Manager application

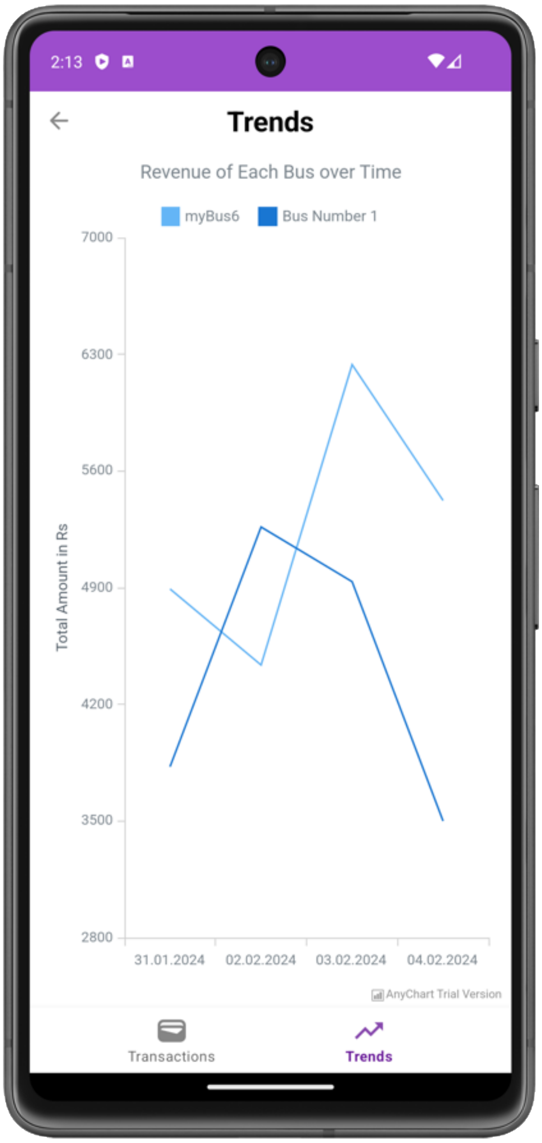


Figure A.4: Revenue trends on a Manager application

4. For Administrators’ Decision-Making Needs

Administrators have the highest level of accessibility to the data and reports within the system. One of their primary objectives is to manage essential data flows and access control by approving, rejecting, or suspending authorizations for bus fleets/bus managers in the system. Simultaneously, much of the data is organized and visualized in a manner that enables the generation of valuable insights regarding the enhancement or maintenance of quality public transportation. Utilizing these insights for the analysis of current trends, predicting future demands, managing and preventing issues, and optimizing and improving service quality are the primary objectives of MIS reporting.

Below are some examples of report generation (Figure A.5 – A.7):

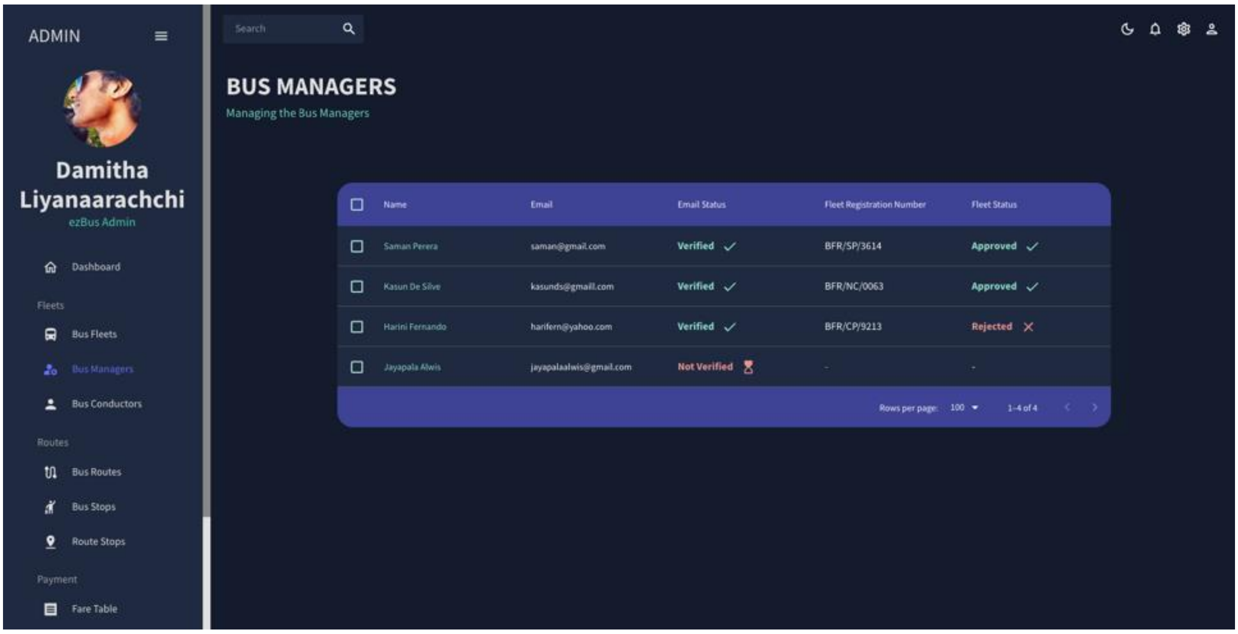


Figure A.5: 'Bus Managers' tab on Administrator portal

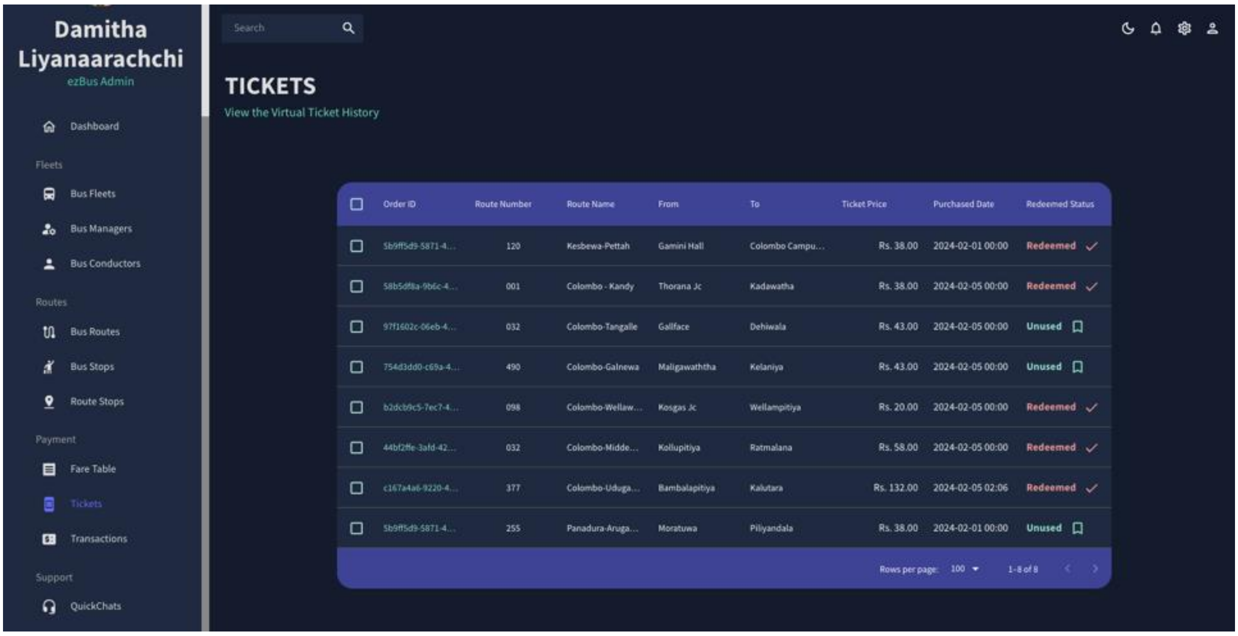


Figure A.6: 'Tickets' tab on Administrator portal

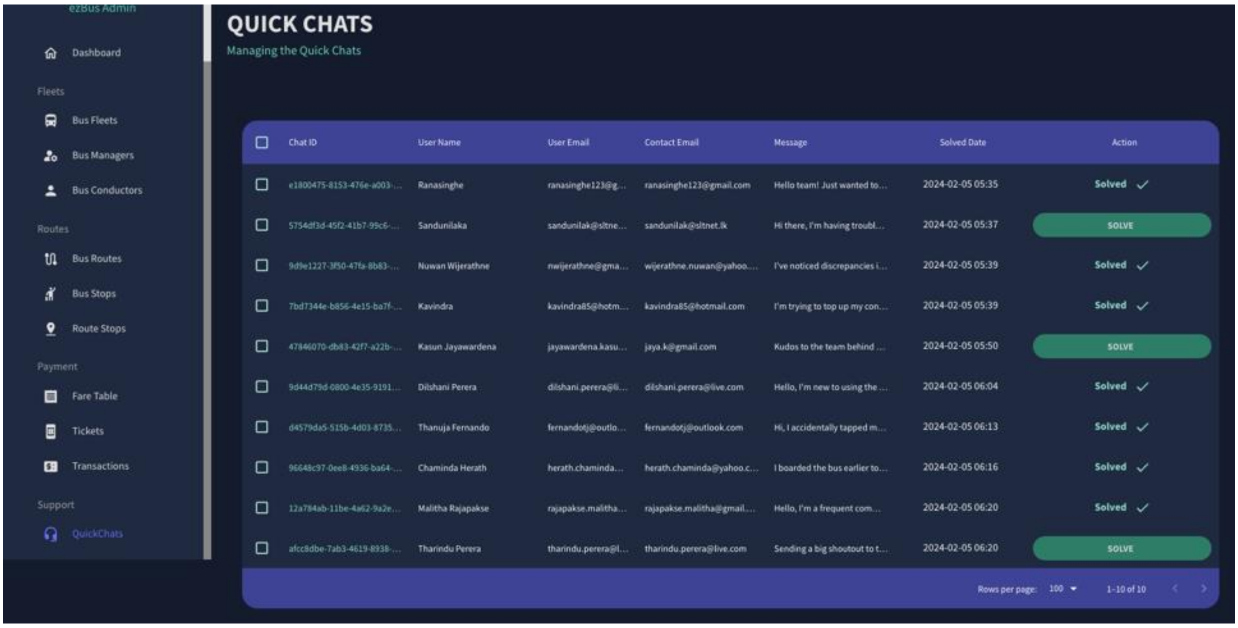


Figure A.7: 'Quick Chats'' tab on Administrator portal

These functions are crucial for decision-making, which ultimately determines the productivity of public transportation. Our overarching goal is to make public transportation convenient for people, thereby increasing its popularity and promoting its sustainability both economically and ecologically.