Android Bus Tracker System

BY

Low Kwang Chien

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF INFORMATION SYSTEMS (HONOURS) INFORMATION SYSTEMS ENGINEERING

Faculty of Information and Communication Technology

(Kampar Campus)

JUNE 2024

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title: **Android Bus Tracker System**

Academic Session: **Y3S3**

I LOW KWANG CHIEN

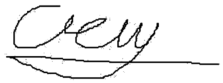
(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

(Author's signature)



(Supervisor's signature)

Address:
No 5, Jalan KJ 5
Taman Krubong Jaya,
75250 Melaka

Cheang Kah Wai
Supervisor's name

Date: 2/8/2024

Date: 13 Sep 2024

FACULTY/INSTITUTE* OF INFORMATION AND COMMUNICATION TECHNOLOGY

UNIVERSITI TUNKU ABDUL RAHMAN

Date: 2/8/2024

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that **LOW KWANG CHIEN** (ID No: **20ACB05483**) has completed this final year project/ dissertation/ thesis* entitled "**Android Bus Tracker System**" under the supervision of **Mr. Cheang Kah Wai** (Supervisor) from the Department of FICT, Faculty/Institute* of Information and Communication Technology.

I understand that University will upload softcopy of my final year project / dissertation/ thesis* in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,

_____

(*LOW KWANG CHIEN*)

DECLARATION OF ORIGINALITY

I declare that this report entitled "**METHODOLOGY, CONCEPT AND DESIGN OF A 2-MICRON CMOS DIGITAL BASED TEACHING CHIP USING FULL-CUSTOM DESIGN STYLE**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : 

Name : LOW KWANG CHIEN

Date : 2/8/2024

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor, Mr. Cheang Kah Wai, for his guidance, advice and constructive comments on the project development process, requirements specification and so on. In addition, he also guided me in the software development life cycle and taught me how to build a good system. He also provided me with a lot of feedback during the system development process, which improved the overall quality of this project.

Besides, I would also like to take this opportunity to thank all my friends for their suggestions and ideas for this project and for helping me to test the system. Their help is greatly appreciated.

ABSTRACT

This paper introduces an innovative Android Bus Tracker System, which through its features can better enable users (students) to provide more accurate bus information. The system incorporates a user-friendly interface with a comprehensive set of features, including a secure login function to ensure personalized access. One of the system's key features is a dynamic reminder system that informs users about the estimated arrival time of the next scheduled bus, providing real-time updates to enhance time management and convenience. The schedule module further allows users to access detailed timeslots of the scheduled buses, facilitating efficient trip planning. To improve overall transportation efficiency, the Android Bus Tracker System integrates Global Positioning System (GPS) technology for accurate and live tracking of bus locations. This feature enables passengers to monitor the real-time location of buses, reducing uncertainties associated with public transportation. The system also introduces a seamless payment method through the implementation of touch and go technology. This not only streamlines the boarding process but also enhances the overall transaction experience for passengers. Furthermore, the Android Bus Tracker System includes a detailed seat occupancy display, providing passengers with information on seat availability and occupancy status. This feature contributes to a more organized and comfortable travel experience, allowing passengers to make informed decisions regarding seat selection. In conclusion, the Android Bus Tracker System amalgamates user-centric features to create an efficient and convenient commuting solution. By addressing key aspects such as real-time tracking, payment convenience, and seat occupancy information, this system aims to elevate the overall public transportation experience for users.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

The Android Bus Tracker System is a response to the existing challenges faced by students relying on the school's bus service. Despite the school's efforts in offering essential information such as bus schedules, route maps, and notifications about cancellations or reschedules on the official webpage, the current system lacks the real time convenience that students require. Having experienced the limitations firsthand, as a student who frequently utilizes the school's bus service, the need for a more user friendly and efficient solution becomes apparent. To address this, the proposed Android Bus Tracker System aims to enhance the commuting experience by providing a dedicated mobile application capable of live tracking the school buses. This system not only ensures accurate real-time information on bus locations but also introduces features like dynamic reminders, precise timeslots, and detailed seat occupancy status, collectively creating a comprehensive solution to streamline and improve the overall efficiency of the school's bus transportation system.

The Android Bus Tracker System, conceived as part of my final year project, represents an innovative approach to addressing the pressing challenges faced by students utilizing the school bus transportation system. Designed and developed using Android Studio, the comprehensive solution encompasses both backend and frontend components, emphasizing a holistic and integrated approach. The motivation behind this project stems from a deep-seated understanding of the need for an upgraded, technologically advanced system to enhance the overall student commuting experience. By harnessing the capabilities of the Osmdroid, the system ensures precise real time tracking of school buses, offering users accurate estimates of arrival times and thereby mitigating uncertainties associated with bus schedules. Beyond mere tracking, the project delves into the integration of Stripe APIs to facilitate seamless and secure payment methods, simplifying financial transactions for students. Furthermore, the system aims to include additional user-centric features such as seat booking, bus scheduling, and reminders, thereby presenting a robust and versatile solution that improve the overall efficiency of the school's bus transportation system.

1.1 Problem Statement and Motivation

1. **Less information on Bus Operation**

On the UTAR campus, students have to rely on the schedule of the school bus service provided by the university to plan their daily journeys. However, this causes confusion and inconvenience to students as the arrival time of the school buses often appears to be early or late. They may miss the bus or have to wait for long periods of time, which affects their studies and lives.

2. **Capacity of a Bus**

During rush hour, students often experience bus crowding, even when buses allow standing. Students may not be able to get to their classrooms in time, and typically a bus is scheduled for 30 minutes. The Android bus tracking system will consolidate the capacity of each bus. Students can track bus information to see if the bus is full.

3. **Payment Issue**

Students need to use TNG to pay for school buses by scanning the QR code on the bus or purchasing a bus ticket on campus. This is not efficient for bus operation as students need to queue up and show proof of payment to the bus driver. If too many students want to ride the bus, the bus will be delayed to the next stop. To overcome this issue, the Touch 'n Go (TNG) API can be integrated into the application. However, a potential issue arises as developers are required to sign up for a TNG Mini Program developer account or a merchant account for the merchant dashboard, which necessitates a valid SSM registration. To address this challenge, exploring alternative payment methods such as Stripe API may be considered to ensure seamless transactions within the app.

The motivation behind the development of Android Bus Tracker is to solve the problem of students not knowing the exact arrival time of the buses, the app will help students to determine the bus routes and track the location in real time, making it easy for students to manage their time travelling to UTAR by bus by optimize the current mode of payment, reducing wait times at bus stops, or optimizing travel routes. Besides, there was some situation that the bus will delay due to some of the bus stop will a lot of students queuing up to take the bus. This problem arises because the current bus system is unable to cope with the uncertain and unpredictable conditions that occur at each bus stop. With this app, students can know the approximate time of bus arrival. This will increase students' self-confidence and enable them to take early action when they realize that the bus is arriving late.

1.2 Objectives

The primary objective of the Android Bus Tracker System is to enhance the overall efficiency and convenience of the bus operation system within the UTAR campus. Firstly, the app aims to provide students with accurate and real-time information regarding the arrival and departure times of school buses. By leveraging GPS technology, students will be able to track the location of buses in real-time, allowing them to plan their journeys more effectively and reduce the likelihood of missing buses or experiencing long wait times at bus stops. This objective addresses the current challenge faced by students who often struggle with the uncertainty of bus schedules, leading to disruptions in their daily routines and academic activities.

Secondly, the app seeks to optimize the capacity management of buses during peak hours. By consolidating data on bus capacities, students will be able to determine if a bus is crowded before it arrives at their location. This feature will help students make informed decisions about whether to wait for the next bus or explore alternative transportation options. By reducing overcrowding on buses, the app aims to improve the overall commuting experience for students, ensuring that they can travel comfortably and safely to their destinations within the campus. Besides, the proposed system provides functionality that allows administrators to update bus schedules and post emergency notifications in the app to notify all students because right now UTAR is posting all schedules and notifications on the website or on notice board in paper form, which is inefficient for students to get the latest news about a new bus schedule update or if any trips will be delayed or cancelled.

Finally, the Android Bus Tracker app aims to streamline the payment process for bus fares. By integrating payment functionalities within the app, such as the Stripe API or alternative payment methods, students will be able to make seamless transactions without the need to queue or show proof of payment to the bus driver. This objective addresses the current inefficiencies in the payment system, which often lead to delays in bus operations and inconvenience for students. Overall, the app aims to revolutionize the bus operation system within the UTAR campus, making commuting easier, more efficient, and more enjoyable for students.

1.3 Project Scope and Direction

The scope of the Android Bus Tracker System encompasses a diverse range of features aimed at creating a robust and user-centric transportation tracking solution. These include:

- Real-time GPS Tracking: Implement a real-time GPS tracking system to provide users with accurate and up-to-date bus location information and time estimates.
- Comprehensive Route Information: Offering detailed route maps, stop details, and schedules to empower users in planning their journeys effectively, enhancing overall user satisfaction.
- Notification System: Integrating a notification system to alert users promptly about bus arrivals, delays, and other relevant updates in real-time, keeping them informed throughout their journey.
- Reliability and Accuracy: A dedicated focus on ensuring the reliability and accuracy of information provided by the system.
- Pay in App: Enabling users to make payments directly within the app, streamlining the payment process for tickets or additional services, promoting a cashless and efficient transaction experience.

1.4 Contributions

The Android Bus Tracker System presents a significant contribution to the enhancement of student commuting experiences within the UTAR campus. Firstly, by providing accurate real-time information on bus locations and schedules, the system alleviates the uncertainties and inconveniences associated with traditional bus tracking methods. Students can now plan their journeys more effectively, reducing the likelihood of missing buses or experiencing long wait times. This aspect not only improves the daily routines of students but also enhances their overall academic and extracurricular engagements by ensuring punctuality and reliability in transportation.

Besides, the implementation of capacity management features addresses the issue of overcrowding during peak hours. By enabling students to check the occupancy status of buses in advance, the system empowers them to make informed decisions about their travel options. This not only improves the comfort and safety of students during transit but also optimizes the utilization of transportation resources within the campus. By reducing overcrowding, the system contributes to a smoother and more efficient commuting experience for all students, fostering a conducive environment for learning and campus life.

Furthermore, the integration of payment functionalities directly within the app streamlines the transaction process, eliminating the need for students to queue or show proof of payment to bus drivers. This not only saves time but also enhances the overall efficiency of bus operations. By promoting cashless transactions and offering alternative payment methods, the system aligns with modern trends in digital payment systems, catering to the preferences and convenience of students. Overall, the Android Bus Tracker System significantly contributes to improving the accessibility, reliability, and efficiency of campus transportation, thereby enhancing the overall quality of student life at UTAR.

1.5 Report Organization

In the following Chapter 2, a number of literature reviews examine the key elements of system functionality, including Global Positioning System (GPS) and distance calculation, and provide a detailed comparison of existing solutions (e.g., Dual Map Bus Tracker, Taiwan Bus Tracker, SITS - Smart Transportation, SG Bus Buddy, and MyTransport SG) with the proposed system, and finally an overview of the proposed system via a system Comparison provides a comprehensive comparison of the proposed system. Chapter 3 delves into the proposed methodology/approach by articulating the methodology, system design diagrams (including system architecture diagrams, use case diagrams and activity diagrams drawn from different perspectives), system design overview, interface design and requirements specification (including user, hardware, functional and non-functional aspects) as well as the implementation issues and timeline. Subsequent Chapter 4, Preliminary Work, provides an overview of the setup process, including software/services and system prototypes. Finally, Chapter 5 summarizes the concept and future improvements or advancements to the Android Bus Tracker System.

CHAPTER 2

Literature Reviews

2.1 System Features

2.1.1 Global Positioning System (GPS)

In recent years, with the rapid increase of devices supporting Global Positioning System (GPS) such as smart phones, CarPlay, etc. GPS technology has become more and more popular. The Global Positioning System (GPS) is a constellation of satellites orbiting the Earth approximately every 12 hours. Each of these satellites is equipped with four atomic clocks, ensuring precise timekeeping. Through a network of these satellites, GPS provides accurate location and time information to users worldwide, enabling navigation, mapping, and a wide range of other applications in fields such as transportation, surveying, and outdoor recreation. These devices make it easier for individuals to track their outdoor mobility and take advantage of location-based applications than ever before (Zheng et al., 2008). Therefore, there are many projects aimed at understanding personal outdoor sports through GPS data analysis. However, many of these projects mainly focus on detecting important locations, predicting movements between these locations, and identifying user-specific activities at each location.

Although location is undoubtedly an important aspect of personal status, it only represents a small part of broader user behavior. Ideally, the scope of analysis should go beyond predicting the activities of specific locations to infer higher-level behaviors, such as the mode of transportation to these destinations. The ability to recognize this advanced behavior not only helps to develop new computing services that can independently meet the needs of users, but also can predict future behavior more accurately.

Another important method to promote GPS-driven computing is to develop GPS trajectory sharing applications. These platforms allow users to record their travel tracks with GPS-enabled devices and share their life experiences with others in online communities. A key problem in this field is the user's GPS trajectory classification based on traffic patterns. By knowing the transportation mode of each GPS track, users can gain valuable insights from other people's travel experiences, including not only where they have been, but also how they got to each location.

Moreover, a key method of enhancing map navigation by GPS in the application of mobile traffic tracking is to provide users with accurate location data. Whether tracking the position of vehicles, monitoring the movement of goods or guiding drivers to drive along optimized routes, GPS can achieve real-time tracking and monitoring with unparalleled accuracy (Verma et al., 2024). This not only improves the efficiency of transportation logistics, but also improves the safety by ensuring that vehicles always travel on designated routes and locations.

In conclusion, the development of applications leveraging Global Positioning System (GPS) technology has witnessed significant strides in recent years, owing to the widespread adoption of GPS-enabled devices like smartphones and CarPlay systems. While existing projects have largely focused on identifying important locations and predicting movements between them, there is a growing recognition of the need to expand the scope of analysis to encompass broader user behaviors. By delving deeper into understanding not just where users go, but also how they get there, such as their mode of transportation, these applications can offer more personalized and efficient services.

2.1.2 Distance Calculation based on GPS

In the fields of navigation and location-based services, GPS (Global Positioning System) plays a key role in determining distance and time calculation. Using the satellite network orbiting the earth, GPS can accurately locate the coordinates of equipment, thus accurately calculating the driving distance and time between two locations. By triangulating the signals received from multiple satellites, the GPS receiver can determine their exact positions very accurately. (Zogg, 2002) Then, this information is used to calculate the distances between waypoints or destinations, and estimate the time required to cross these distances according to factors such as speed and terrain.

The following is some information about Ramer-Douglas-peucker (RDP), Kalman filtering algorithm and Haversine formula, which are used to calculate the distance or optimal path of GPS navigation:

RDP algorithm is a method to simplify curves in data sets. It aims to reduce the number of points in a curve while maintaining its shape within the specified tolerance (Douglas-Peucker Algorithm | Cartography Playground, n.d.). The algorithm recursively divides the curve into smaller segments, and for each segment, it checks the distance between all points and the segment formed by the endpoint. If a point is within the tolerance threshold, it will be discarded, otherwise, the point farthest from the line segment will be marked as a key point. By applying this process iteratively, the algorithm effectively reduces the number of points needed to represent the curve, thus producing a simplified version very similar to the original shape.

Kalman filter is an algorithm for estimating the state of linear dynamic systems affected by noise observation. It operates recursively by combining system state prediction based on previous state and control input with correction obtained from noise sensor measurement (Franklin, 2020). The filter calculates the best estimate of the system state by minimizing the mean square error between prediction and observation measurement. This makes it especially suitable for real-time tracking the position and speed of objects, in which the noise sensor data must be filtered to obtain a more accurate estimate of the real state of the system.

Based on the research, the Haversine formula is a mathematical formula used to calculate the shortest distance between two points on the surface of a sphere, given their latitudes and longitudes. It is commonly used in navigation and geographical

applications to compute the great-circle distance between two locations on Earth. The formula considers the curvature of the Earth's surface and provides a more accurate distance calculation than simple Euclidean distance measurements, which assume a flat Earth. (Veness, 2019) The Haversine formula considers the spherical geometry of the Earth and uses trigonometric functions to calculate the arc length along the surface of the sphere between the two points.

2.2 System Review

2.2.1 DoubleMap Bus Tracker

Figure 2.1 Logo of DoubleMap Bus Tracker

DoubleMap is an innovative real-time GPS bus tracking system that revolutionizes the way users navigate public transportation. Available on both Android and iOS platforms, this cutting-edge application offers a seamless experience for users seeking multiple locations for route lookup. With the ability to select or search for specific destinations, DoubleMap provides comprehensive information on bus details, bus stop specifics, route options, and integrated bus systems from various companies and schools. Designed to enhance the efficiency and convenience of public transportation, DoubleMap ensures users have accurate, up-to-the-minute information about bus locations and routes. The user-friendly interface allows for easy navigation, empowering individuals to plan their journeys with confidence. Whether you're a commuter, student, or traveler, DoubleMap collaborative efforts with numerous companies and schools make it a versatile and reliable tool for diverse transportation needs. The app's compatibility with both Android and iOS systems ensures widespread accessibility, catering to a broad user base. By leveraging state-of-the-art GPS technology, DoubleMap provides a real-time solution that aligns with the fast-paced demands of modern urban living. Through strategic partnerships with various companies and schools. (“DoubleMap Bus Tracker - Apps on Google Play”) DoubleMap establishes itself as a go-to platform for a comprehensive and integrated bus tracking system, enhancing the overall efficiency and reliability of public transportation services. Embrace the future of transit navigation with DoubleMap, where precision meets convenience, transforming the way you experience and interact with bus services.

Strengths

Real Time GPS Bus Route for Multiple System

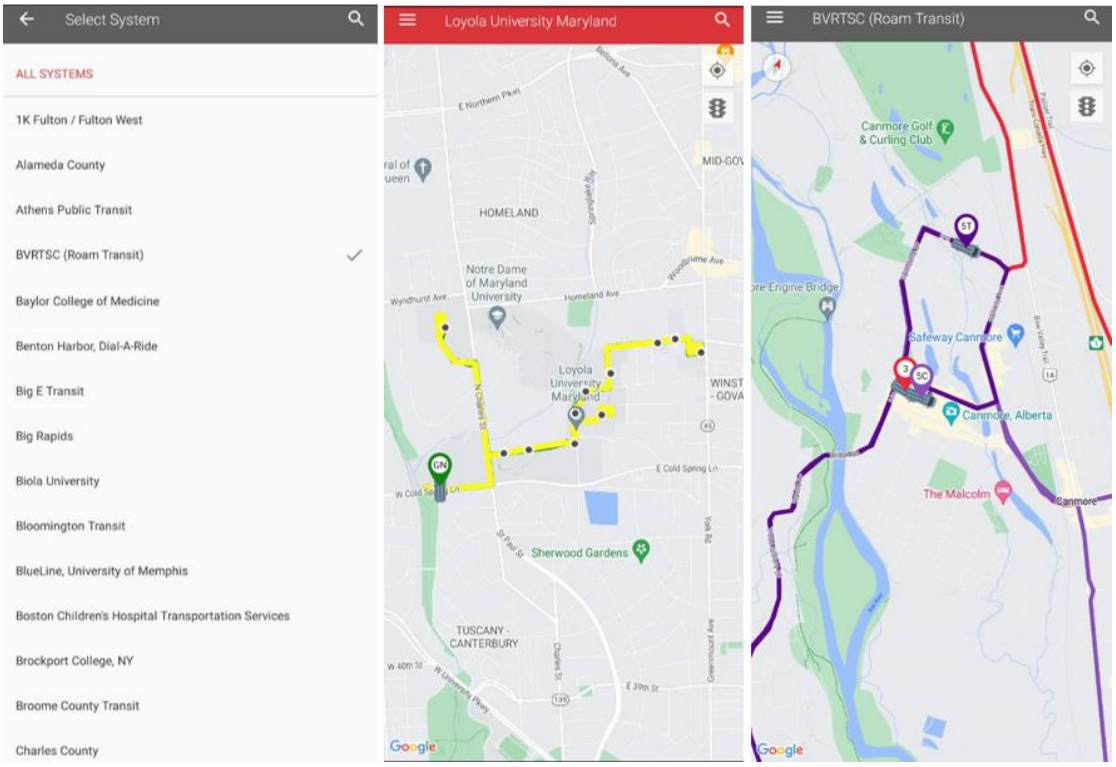
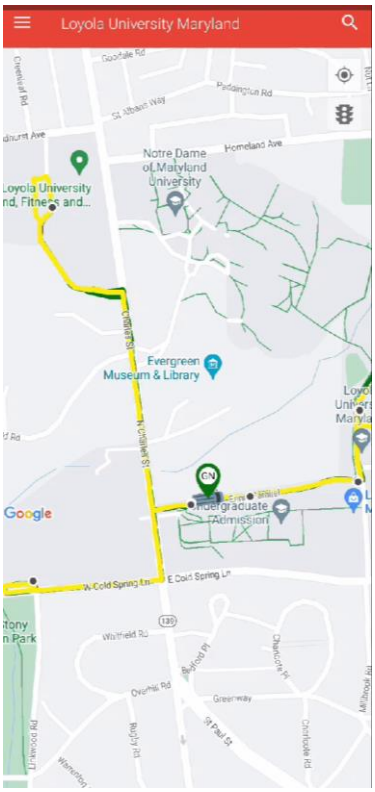


Figure 2.2 Map Route in DoubleMap

In DoubleMap app, users can select the system (location) where they want to know about transit routes. The application uses the Google Maps API to track routes and transit information. It also distinguishes the colors of different bus routes so that users can easily know which bus they want to take. Users can also interact with the map by clicking on the bus icon and bus stop icon to get detailed information about buses and bus stops. Besides, users can use the search function by clicking on the magnifying glass icon to search for the locations they need. This saves users time by eliminating the need to view locations one by one in the Select System option. On other hands, the Double Map has implemented the traffic visibility features from the google map API, traffic visibility features allow users to gain insight into real-time traffic updates, congestion patterns and potential delays.

Bus Movement Animation



Video 1.1 Bus Movement Animation

By looking at some of the regional routes, the movement of bus animation run quite accurately, getting from one stop to the next in just the right place, with no deviations. (“DoubleMap - Live Bus Tracker”)

Weaknesses

Too Few Features

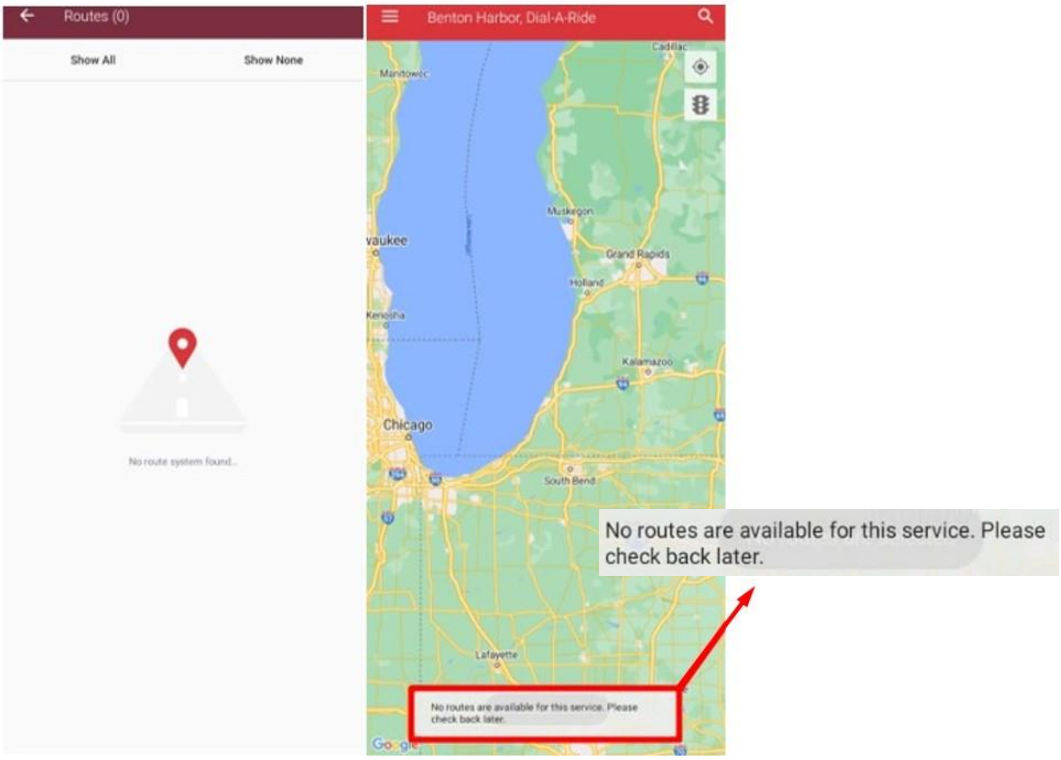


Figure 2.3 Invalid Features

DoubleMap offers a number of systems (locations) to choose from, each with accurate maps, and while the user interface is simple and easy to understand, it lacks the functionality to guide the user to the correct bus route information. Examples include schedules for the next bus trip, estimated times of arrival for every bus in every bus stop, lack of routing at certain locations, and lack of information to notify users of bus delays or schedule changes. The application's inability to provide timely notifications regarding such alterations may lead to inconvenience for commuters who rely on accurate and up-to-date information. Besides, some details related to buses and bus stops are not correctly updated or are not up to date. This lack of real-time accuracy may result in users relying on outdated information, impacting the overall reliability of the service.

Bus Routes Issue

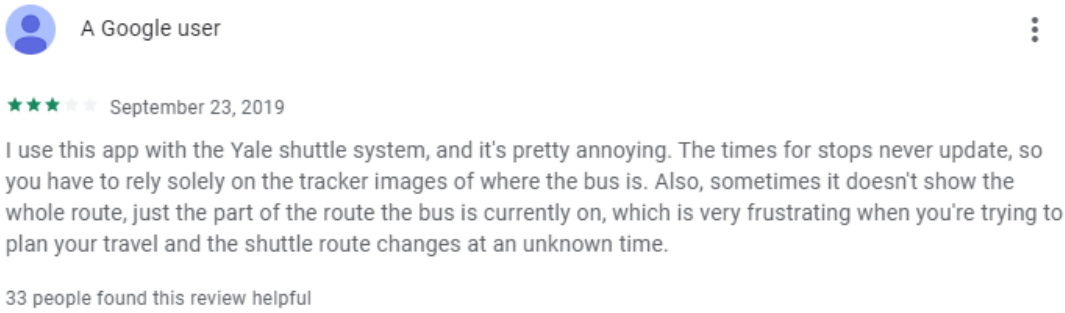


Figure 2.4 User Comment

Some users have complained that the DoubleMap does not show the entire route, only the portion of the route the bus is currently traveling, which is very frustrating when trying to plan a trip and the shuttle route changes at an unknown time.

2.2.2 BusTracker Taiwan

Figure 2.5 Logo of BusTracker Taiwan

The BusTracker Taiwan is a state-of-the-art GPS bus tracking system designed to revolutionize the public transportation experience in Taiwan. This innovative application comes with a range of features and functions designed to improve the efficiency, convenience, and reliability of bus travel across the country. BusTracker Taiwan features real-time tracking, allowing users to monitor the precise location of buses at any time. The app has an intuitive, user-friendly interface that provides a seamless experience for users to access multiple functions. One of the main advantages of the application is its comprehensive route information, which provides users with a detailed overview of bus schedules, stops and routes. To help users plan their trips efficiently, BusTracker Taiwan also offers advanced features such as real-time arrival prediction, ensuring that passengers can accurately predict bus arrival times. The app also fulfills the need for complete route visibility by displaying the entire bus route, enabling users to plan their trips with confidence. In addition, BusTracker Taiwan is designed with the user in mind, supporting both Android and iOS platforms to ensure broad accessibility. Whether you are a daily commuter, an occasional traveler or a tourist visiting Taiwan, BusTracker Taiwan provides a reliable and efficient solution to help you seamlessly navigate the bus network. (Wena's TV)

Strengths

Full Featured

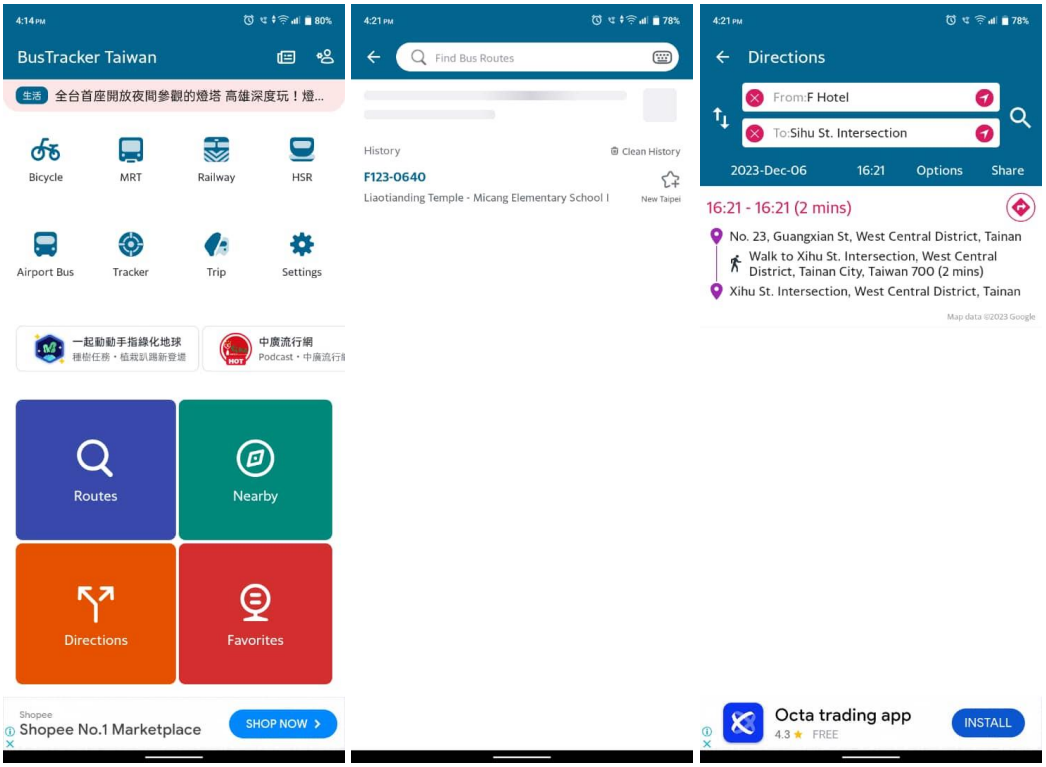


Figure 2.6 Features of Main Page BusTracker Taiwan

BusTracker Taiwan has a powerful and integrated transportation tracking system for buses, bicycles, MRT, railroads, high-speed rail, and airport buses. It also detects nearby transportation by retrieving the user's location, and users can use its app features to plan trips in Taiwan. The functionality inside the BusTracker Taiwan also provides users with great experience, users can search the departure and destination with the routes features and select which type of transportation they want to use, or they can use the directions features to know more details and accurate to reach their destination.

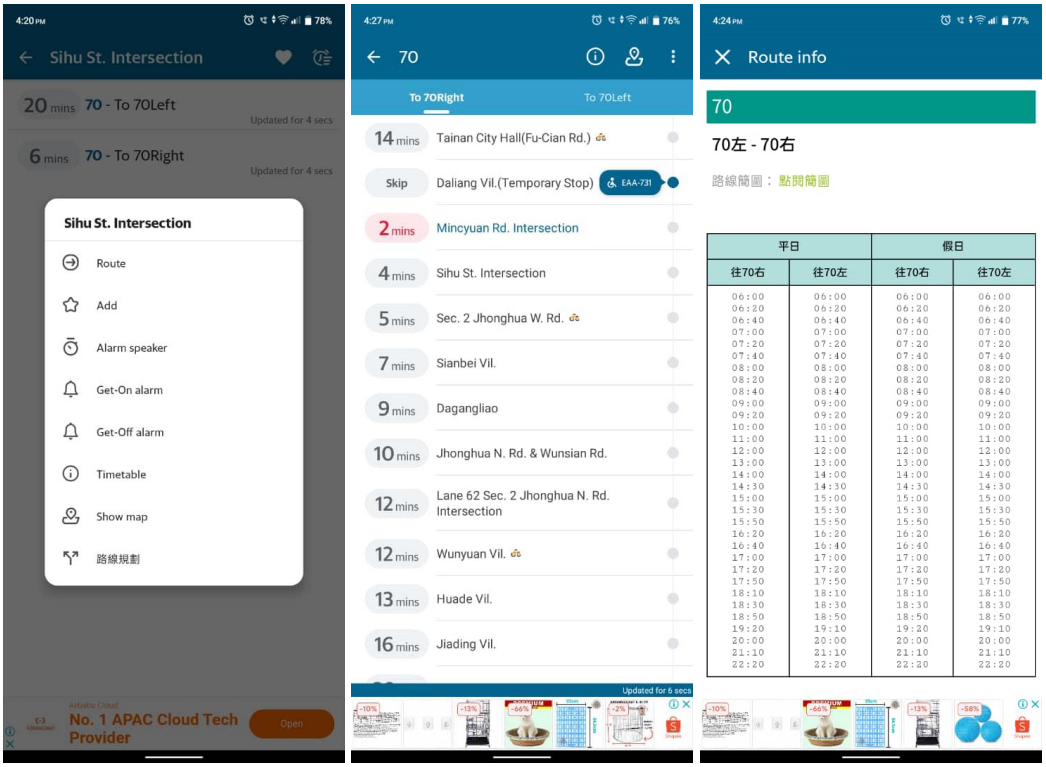


Figure 2.7 Additional Features of BusTracker Taiwan

BusTracker Taiwan has seamlessly integrated a range of user-centric features into its app, enhancing the overall commuting experience. Users can effortlessly access the "Route" feature, offering a detailed overview of bus routes, stops, and time. The convenience of "Add to Favorite" allows users to personalize their most-traveled routes for quick access. For enhanced journey planning, the app introduces "Alarm Speaker," "Get-On Alarm," and "Get-Off Alarm," ensuring passengers never miss a stop. The "Timetable" feature provides a comprehensive schedule overview, while "Show Map" offers real-time visual tracking of buses. Lastly, the "Route Planning" feature empowers users to map out the most efficient journey, solidifying BusTracker Taiwan as a comprehensive and user-friendly GPS bus tracking solution for navigating Taiwan's bustling transportation network.

Weaknesses

Map Route Are Not User Friendly

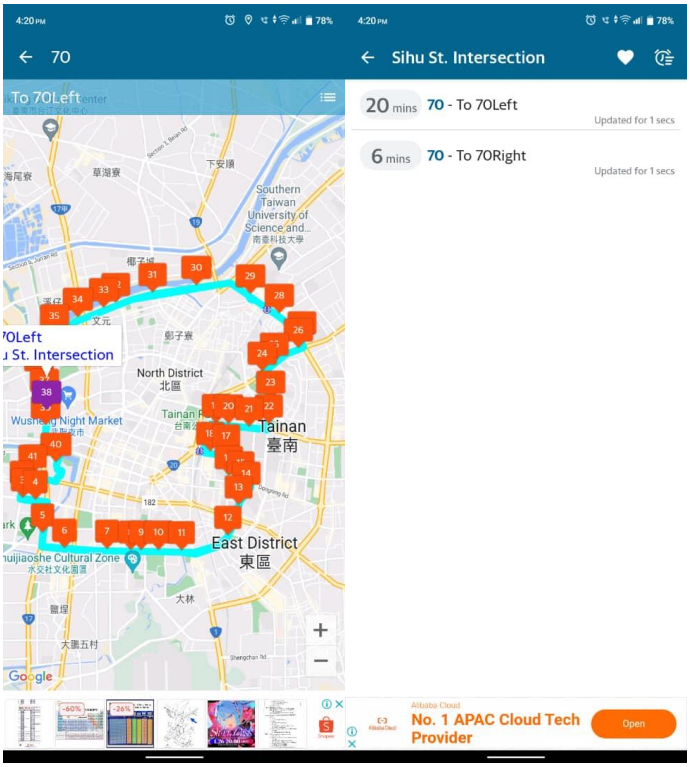


Figure 2.8 Map Route and Route Direction Features

BusTracker Taiwan impresses with meticulously planned map routes, showcasing each stop with precision. While the app provides a comprehensive "Route" feature detailing the time required for transportation to reach all stops, it currently lacks the capability for users to visually track the live movement of transportation on the map. This absence of real-time tracking may limit users' ability to monitor buses in transit and anticipate their arrival at specific stops. Introducing a dynamic map feature displaying the actual movement of transportation would not only enhance the user experience but also provide a more intuitive and engaging way for commuters to follow their bus's progress.

2.2.3 SITS – Intelligent Transport

Figure 2.9 Logo of SITS - Intelligent Transport

SITS, which supports both Android and iOS platforms, is a key component of the "Smart Transport and Mobility" of the Smart Selangor program. It aims to empower users of the Smart Selangor Bus Service by providing all 12 local governments in Selangor with an intelligent platform for efficient trip planning. SITS is committed to enhancing the public transportation experience by allowing users to seamlessly identify the nearest bus terminal, view estimated arrival times and access the Smart Selangor Bus Transportation Timetable. As part of a wider effort to encourage the use of public transportation, especially the Smart Selangor Bus Service, the app joins the Free Bus Service and Smart Selangor Smart Bus Terminal as a complementary ecosystem. Through SITS, not only will trip planning be simplified, but the overall quality of service will be enhanced by monitoring real-time service levels and conducting detailed data analysis to continuously improve and upgrade the Selangor Intelligent Transport System. (“MyGOV - the Government of Malaysia’s Official Portal”)

Strengths

Simple And Clean User Interface and Features

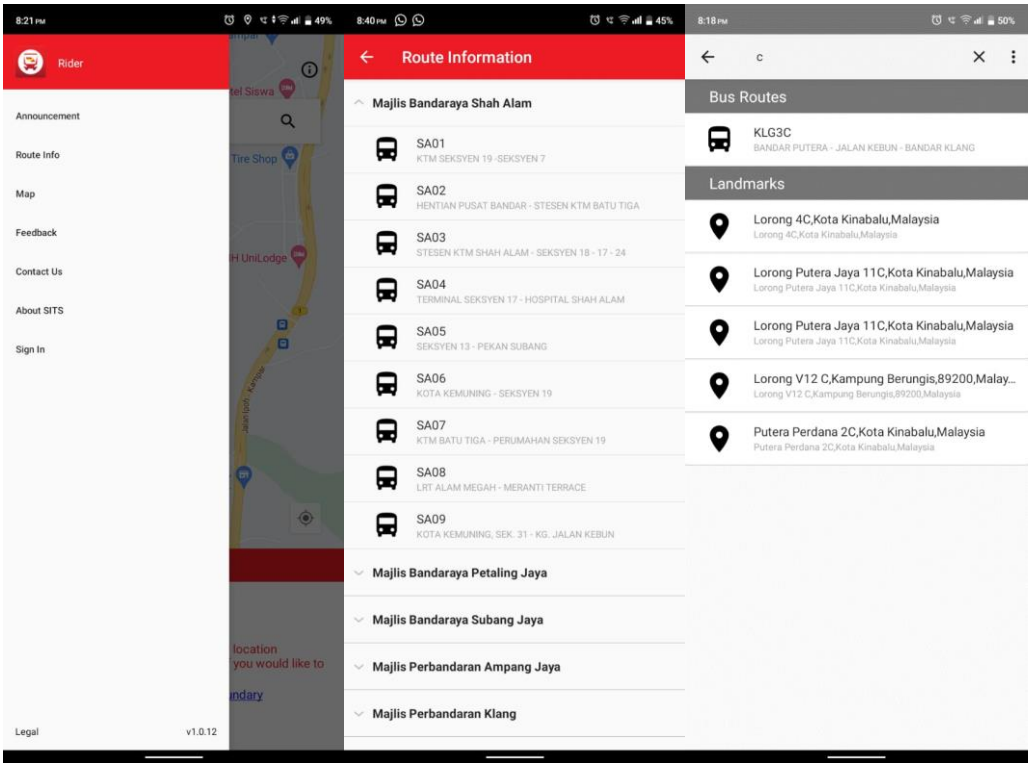


Figure 2.10 User Interface and Features of SITS

The user interface of SITS shares similarities with the DoubleMap Bus Tracker, yet each application diverges in its unique focus and features. While DoubleMap caters to diverse locations, SITS stands out with a specialized emphasis on the Selangor area. SITS has an easy-to-use user interface and straightforward features that allow users to get started with the app after just a few moments. In the figure above, SITS introduces three user-friendly methods for bus tracking. Firstly, users can effortlessly retrieve their location, providing an instant overview of nearby buses. Secondly, a robust search feature allows users to input either the bus name or specific locations, enhancing the precision of tracking. Lastly, the inclusion of detailed route information offers a comprehensive approach, enabling users to find their desired route with ease. This strategic of tracking options demonstrates SITS' commitment to providing diverse intuitive tools for users in the Selangor region.

Weaknesses

Time Estimation and Accuracy Issues

Figure 2.11 Users Comment

The figure above shows the common concerns expressed by users through their comments, indicating that a significant number of users expressed dissatisfaction with the GPS accuracy performance of SITS. It often happens that the location of the bus on the map is not the same as in reality, sometimes the time on the app shows that there are still a few KM before the bus arrives, but the bus has already arrived at the stop. It is also worth noting that many users expressed dissatisfaction, especially with issues related to time accuracy. A recurring issue highlighted in the user comments was the discrepancy between the arrival time shown by the application and the actual delayed arrival time, with buses often arriving up too few minutes later than expected. This discrepancy raises concerns about the reliability of the GPS tracking system within SITS and its ability to provide users with timely and accurate information. (“SITS - Intelligent Transport - Apl Di Google Play”)

2.2.4 SG Bus Buddy

Figure 2.12 Logo of SG Bus Buddy

SG Bus Buddy is a user-centric Android and IOS application designed to revolutionize the public transportation experience in Singapore. With its powerful features, users can effortlessly access real-time bus timings by bus stop code, road, location, or service number with just a single tap, ensuring a seamless and efficient commuting experience. SG Bus Buddy stands out with its ability to schedule daily reminders, providing users with timely notifications for their planned journeys. Supporting multiple language English, Chinese, Malay, and Tamil, this app caters to a diverse user base. SG Bus Buddy also incorporates an EZ Link Card Reader for enhanced payment options and boasts a beautiful Dark/Light Mode for personalized and comfortable viewing. Furthermore, users can stay informed about bus capacity and type (Single/Double Decker) briefly, enhancing their decision-making process. To top it off, the app provides a quick weather overview, allowing users to start their day with all the information they need for a smooth commute. (“SG Bus Buddy”)

Strengths

Time Estimates and Clear Directions for Multiple Routes

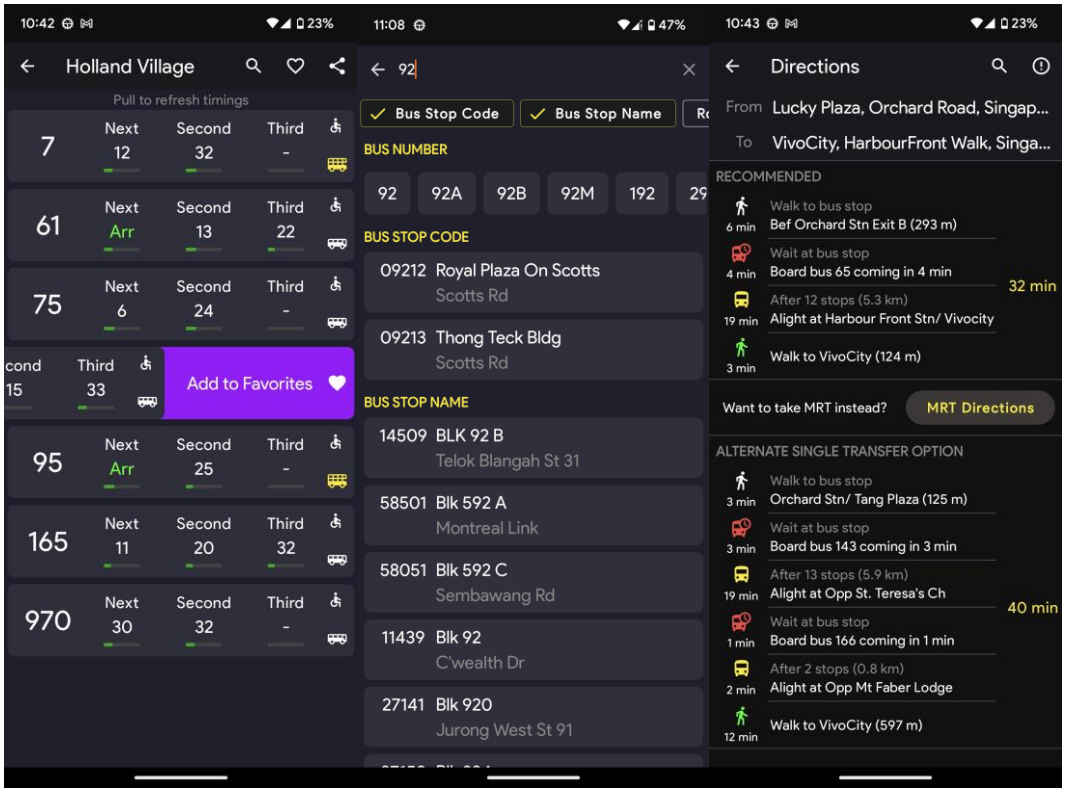


Figure 2.13 Features in SG Bus Buddy

The differentiating factor of the SG Bus Buddy app apart from other existing systems is its ability to predict bus arrival times for multiple trips. This innovative feature provides users with a valuable advantage: even if a user misses the scheduled time for a segment of a trip, SG Bus Buddy doesn't stop at notifying of delays. Instead, it seamlessly realigns the entire trip, dynamically predicting and displaying the arrival times of subsequent buses. In this way, users can schedule efficiently and make plans adaptable. For directions feature, SG Bus Buddy offers a user-friendly and intuitive interface. After inputting the starting point and destination, the app provides clear and detailed step-by-step directions, guiding users through the entire journey. The directions are presented with each step of the journey clearly outlined, including bus transfer points and walking directions between stops. The time estimates for each leg of the journey are dynamically updated, considering real-time traffic conditions and bus timings.

Weaknesses

Route Does Not Show Current Transportation Movements

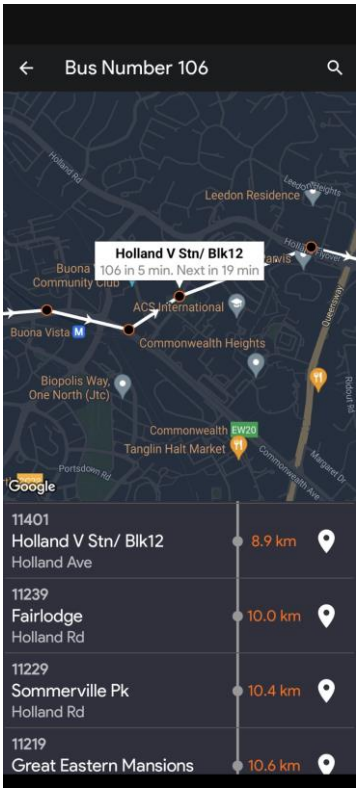


Figure 2.14 Map Route in SG Bus Buddy

One of the weaknesses that SG Bus Buddy is currently facing is its inability to display real-time traffic movements along its routes. Currently, the application is unable to display the current location or travel status of buses in real time. The absence of dynamic tracking on the map may affect the user's ability to understand the ongoing traffic movement instantly and accurately within the system.

2.2.5 MyTransport SG

Figure 2.15 Logo of MyTransport SG

MyTransport SG is a comprehensive user-centric platform that revolutionizes the way individuals navigate and use Singapore's transport system. The platform provides a comprehensive overview of Singapore's various modes of transportation, seamlessly integrating real-time information, intelligent planning tools and a user-friendly interface to empower commuters. Users can access up-to-the-minute information on bus and train schedules, track real-time operations of public transportation, and easily plan trips across multiple modes of transportation.MyTransport.sg goes beyond traditional mapping services to provide invaluable insights into traffic conditions, ensuring that users make informed decisions about their routes. With features such as trip planning, fare information and service alerts, the platform meets the dynamic needs of commuters, creating a smoother, more efficient, and enjoyable transportation experience in the bustling Singapore cityscape. (“MyTransport.SG - Apps on Google Play”)

Strengths

Clear Categorization and Clear Direction of Routes

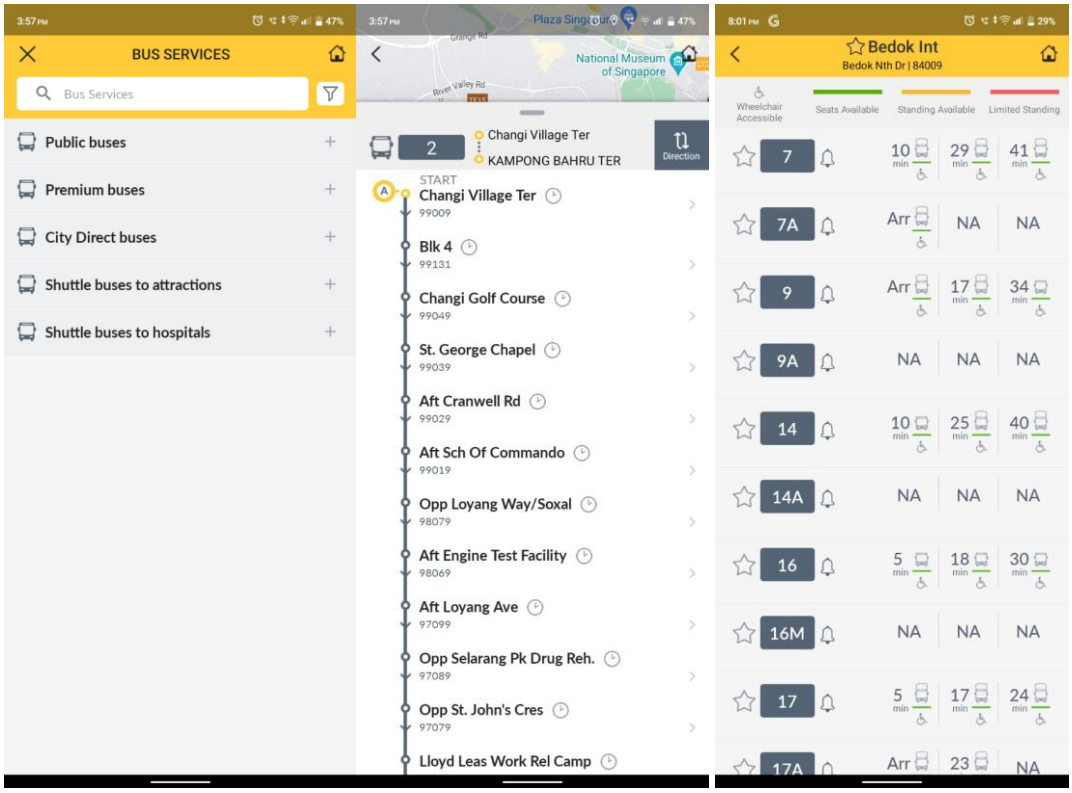


Figure 2.16 Bus Service Selection and Route Direction Features

MyTransport SG offers a well-organized and user-friendly experience by providing clear categorization of bus services. Users can effortlessly customize their trip preferences precisely by choosing from a variety of categories, including public buses, premium buses, direct city buses, and shuttle buses to attractions or hospitals. In addition, MyTransport SG provides users with clear route directions below the map, with the entire bus route categorized in a concise yet detailed manner, and each stop along the way clearly marked. With the direction icon click feature, users can explore reverse routes to get a comprehensive overview of the outbound and return routes. Besides, users can click on each stop to find out the estimated arrival time of the bus and the number of each bus. In this way, users can make better decisions for their trips.

Weaknesses

Route Does Not Show Current Transportation Movements

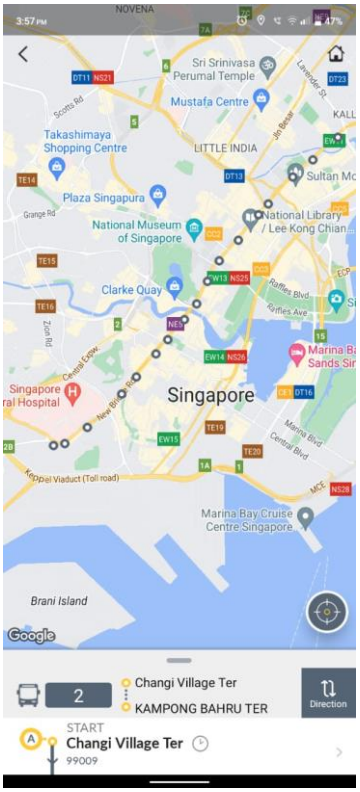


Figure 2.17 Map Route in MyTransport SG

Same as the SG Bus Buddy, MyTransport SG does not show real-time traffic movements along the route. Users can only predict whether the bus will arrive or not by estimating the time.

2.3 Comparison of the System Review and Proposed System

	DoubleMap Bus Tracker	BusTracker Taiwan	SITS	SG Bus Buddy	MyTransport SG	Proposed System
Real-time GPS tracking	YES	YES	YES	YES	YES	YES
Map Route	YES	YES	YES	YES	YES	YES
Time Estimate	NO	YES	YES	YES	YES	YES
Time Schedule	NO	YES	NO	YES	YES	YES
Transportation Movements	YES	NO	YES	NO	NO	YES
Notification	NO	YES	NO	YES	YES	YES
Booking	NO	NO	NO	NO	NO	YES
Payment	NO	NO	NO	NO	NO	YES

Table 2.1 Comparison of the System Review and Proposed Systems

The proposed system aims to bridge the existing gaps and enhance the experience by amalgamating the strengths of various existing bus tracking applications. It builds upon the successful features of popular apps like DoubleMap Bus Tracker, BusTracker Taiwan, SITS, SG Bus Buddy, and MyTransport SG while introducing new functionalities to create a comprehensive and user-centric solution.

CHAPTER 3

System Methodology/Approach

3.1 Methodology

The RAD phased development model will be chosen as the development methodology for the Android Bus Tracker System project. In this phased development approach, there will be 4 phases namely planning, analysis, design, and implementation. After implementation, the implemented part of the system will be rolled out in phases. After the completion of the first phase, the next iteration will continue from the analysis phase to the implementation phase and will be pushed to the client as the next system version until the full end of the project life cycle. Each phase in the system is part of the system.

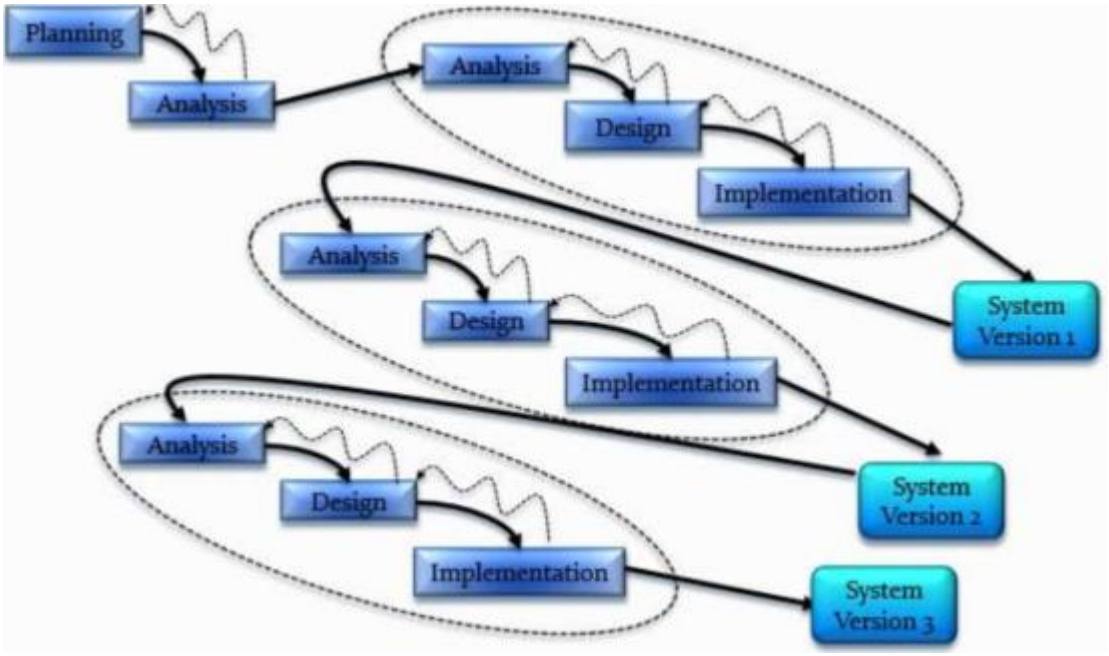


Figure 3.1 RAD Phased Development

3.2 System Design Diagram

3.2.1 System Architecture Diagram

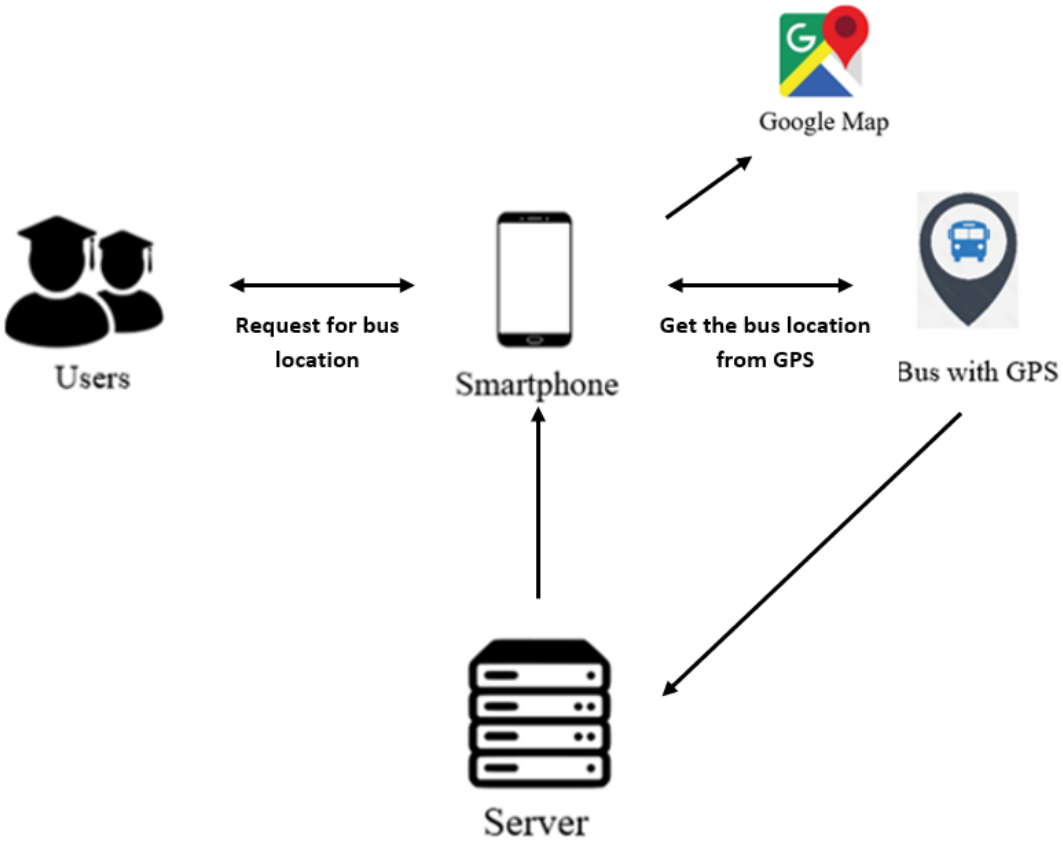


Figure 3.2 System Architecture of Android Bus Tracker System

3.2.2 Use Case Diagram

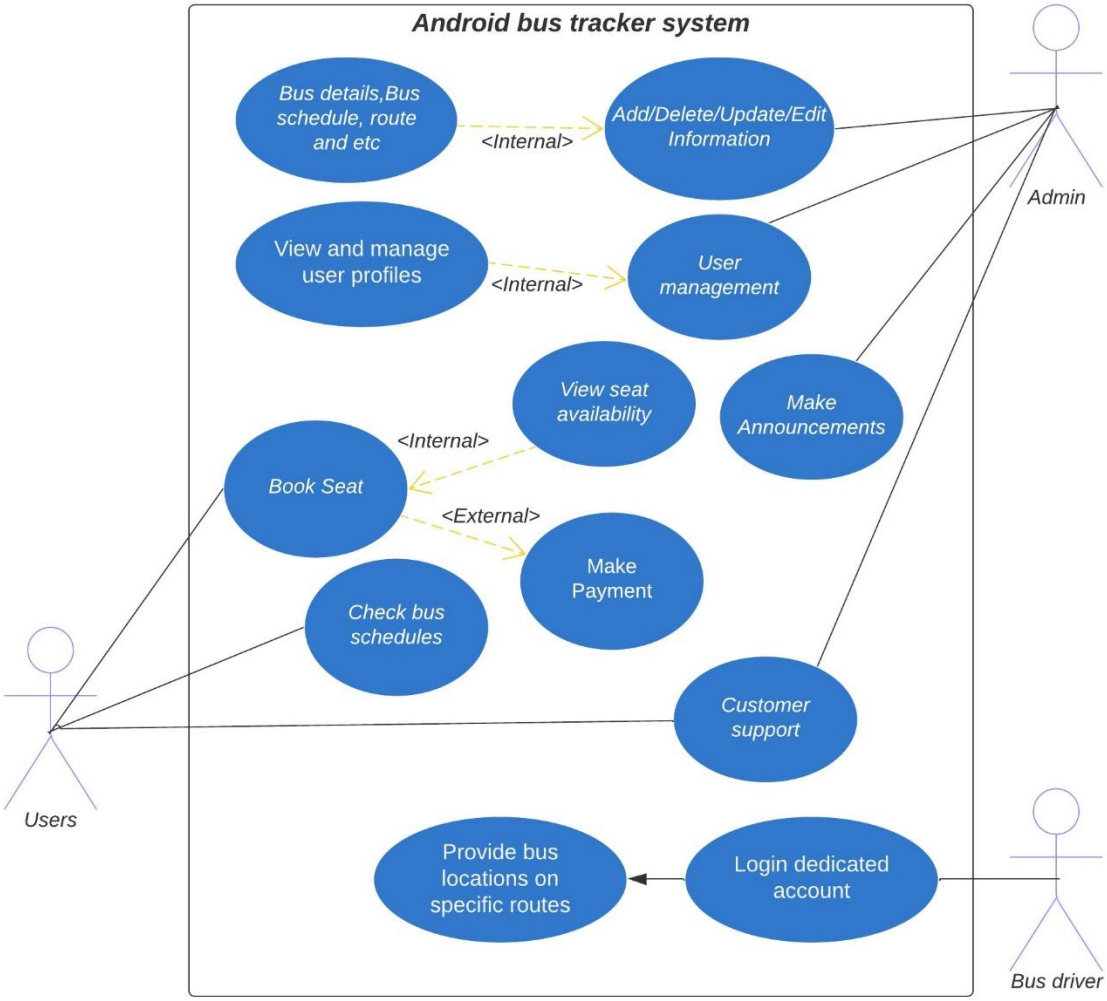


Figure 3.3 Use Case Diagram of Android Bus Tracker System

3.2.3 Activity Diagram from User Perspective

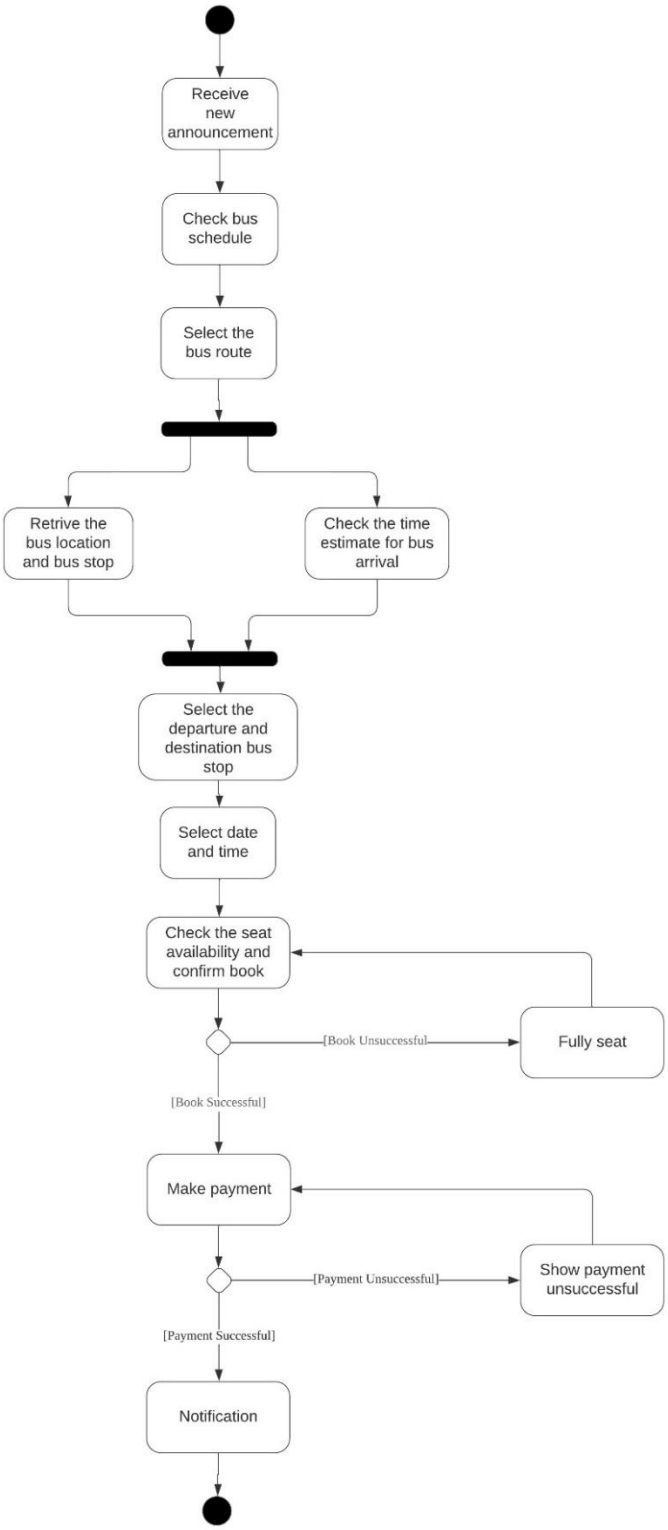


Figure 3.4 Activity Diagram of Android Bus Tracker System (User)

3.2.4 Activity Diagram from Bus Driver Perspective

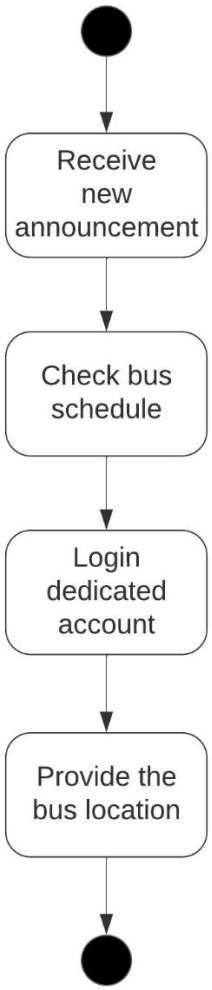


Figure 3.5 Activity Diagram of Android Bus Tracker System (Bus Driver)

3.2.5 Activity Diagram from Admin Perspective

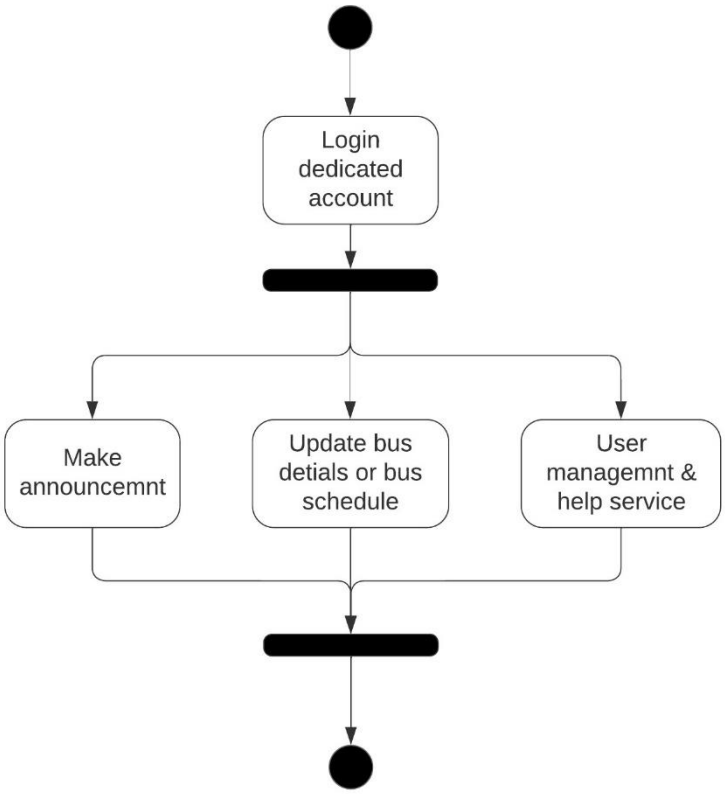


Figure 3.6 Activity Diagram of Android Bus Tracker System (Admin)

3.3 Timeline

ACTIVITY PLAN

#	Activity			Start	End
				WEEK >	
1	Phase 1: Project Setup and Planning	29/01/24	04/02/24	1	7
2	Define Project Scope	29/01/24	30/01/24	1	2
3	Set up development environment	31/01/24	31/01/24	3	3
4	Gather project requirements	31/01/24	03/02/24	3	6
5	Create project plan	03/02/24	04/02/24	6	7
6	Phase 2: Implement Osmdroid Library	05/02/24	14/02/24	8	17
7	Study the Osmdroid library	05/02/24	11/02/24	8	14
8	Implement to the project	08/02/24	14/02/24	11	17
9	Phase 3: Implement GPS retreive feature	15/02/24	10/03/24	18	42
10	Testing GPS tracker device	15/02/24	03/03/24	18	35
11	Updating and fetching data to server	04/03/24	10/03/24	36	42
12	Phase 4: Calculation time estimation feature	11/03/24	09/04/24	43	72
13	Implement calculate distance and time estimation function	11/03/24	09/04/24	43	72
14	Test functions and adjustments	18/03/24	09/04/24	50	72
15	Phase 5: Report and Review the System	10/04/24	24/04/24	73	87
16	Conduct all the details in the report	10/04/24	17/04/24	73	80
17	Conduct system testing	13/04/24	24/04/24	76	87
18	Fix bugs and issues	13/04/24	24/04/24	76	87

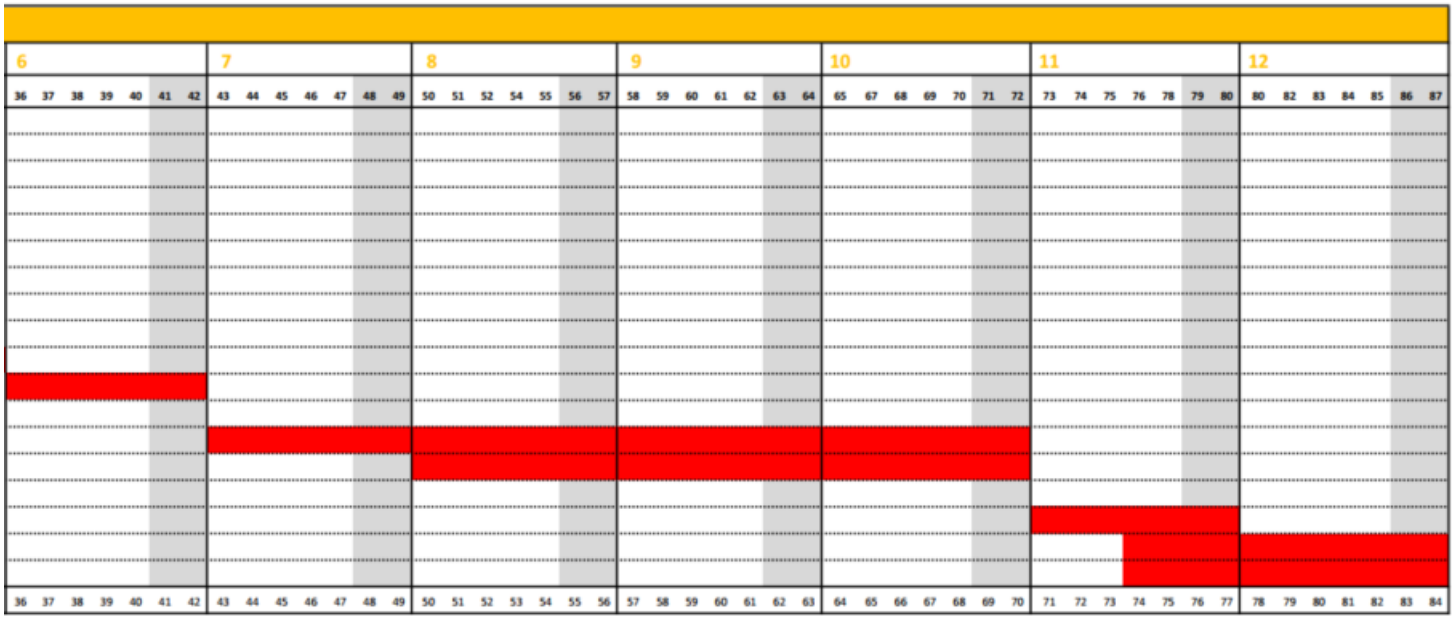
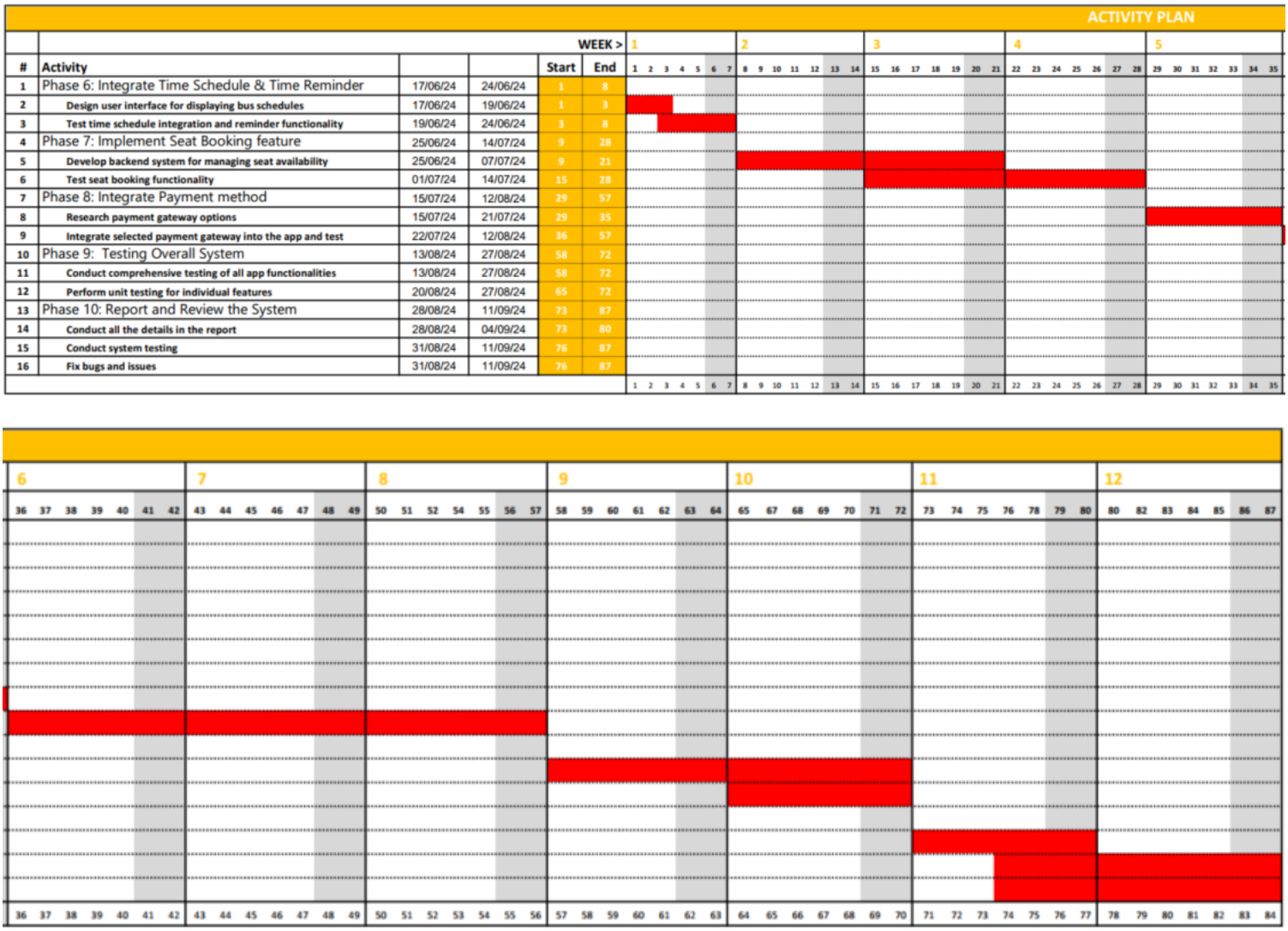


Figure 3.7 FYP 1 Timeline Grant Chart



#	Activity			Start	End
1	Phase 6: Integrate Time Schedule & Time Reminder	17/06/24	24/06/24	1	8
2	Design user interface for displaying bus schedules	17/06/24	19/06/24	1	3
3	Test time schedule integration and reminder functionality	19/06/24	24/06/24	3	8
4	Phase 7: Implement Seat Booking feature	25/06/24	14/07/24	9	28
5	Develop backend system for managing seat availability	25/06/24	07/07/24	9	21
6	Test seat booking functionality	01/07/24	14/07/24	15	28
7	Phase 8: Integrate Payment method	15/07/24	12/08/24	29	57
8	Research payment gateway options	15/07/24	21/07/24	29	35
9	Integrate selected payment gateway into the app and test	22/07/24	12/08/24	36	57
10	Phase 9: Testing Overall System	13/08/24	27/08/24	58	72
11	Conduct comprehensive testing of all app functionalities	13/08/24	27/08/24	58	72
12	Perform unit testing for individual features	20/08/24	27/08/24	65	72
13	Phase 10: Report and Review the System	28/08/24	11/09/24	73	87
14	Conduct all the details in the report	28/08/24	04/09/24	73	80
15	Conduct system testing	31/08/24	11/09/24	76	87
16	Fix bugs and issues	31/08/24	11/09/24	76	87

Figure 3.8 FYP 2 Timeline Grant Chart

3.4 Requirement Specifications

3.4.1 User Requirements

In the proposed system, there are three target users involved which is user, bus driver and admin. The details and requirements will be shown at below.

User

The user segment of the proposed Android Bus Tracker System primarily targets students who rely on efficient transportation to and from their destinations. Through this system, users can conveniently access crucial information such as bus schedules, route details, estimated arrival times, seat booking options, and payment facilities. By utilizing this application, students can effectively plan their travel arrangements, ensuring they reach their destinations on time and with minimal hassle.

Bus Driver

The role of the bus driver in the Android bus tracker system mainly includes logging into a dedicated account to track and report the bus location. After logging in, drivers grant permission to their Android smartphones, allowing the system to automatically capture and update the bus coordinates in real time. Then use these coordinates to display the bus icon on the map interface of the system and provide users with accurate and up-to-date information about the current location of the bus.

Admin

The admin aspect of the Android Bus Tracker System revolves around overseeing and managing various operational aspects of the platform. Administrators are responsible for updating the bus time schedules to reflect any changes or adjustments. They can make announcements through the system to communicate important information or updates to users. Furthermore, one of the role of admin is provide assistance to users, addressing any queries or issues they may encounter while using the application.

3.4.2 Functional Requirements

Functional requirements are functions that need to be implemented in the system. The following are the functional requirements of the Android bus tracker system.

1. The system must be able show the map, bus route, bus stop and bus location to the user.
2. The system must be able to show the time estimate of the bus arrival between each bus stop to the user.
3. The system must be able to provide the payment method to the user.
4. The system must be able to provide booking feature to the user.
5. The system must be able to provide dedicated account to bus driver.
6. The system must be able to retrieve the coordinates from the bus driver's smart phone.
7. The system must be able to let admin post the new announcements and update the bus time schedule.

3.4.3 Non-Functional Requirements

Non-functional requirements are things that have nothing to do with system functions, and they are more concerned with how the system should operate.

1. The system should provide the accurate information to the user.
2. The system should provide user with accurate estimated bus arrival time.
3. The system should reduce the bus delay time because it reduces the time for students to queue up and pay.
4. The system should keep updating the coordinates and provide the bus location accurately.
5. The system should reduce the paperwork done by admin.

CHAPTER 4

System Design

The Android Bus Tracker System is a comprehensive solution designed to enhance the efficiency and convenience of public transportation for users, comprising three integral components: hardware, software, and server infrastructure. The hardware component serves as the foundation for real-time tracking and communication within the system. It primarily involves the use of Android smartphones installed on buses, equipped with GPS capabilities. These smartphones serve as onboard units responsible for continuously broadcasting the current location of the bus to the system's server. By leveraging GPS technology, these devices accurately determine the bus's position in real-time, enabling precise tracking and monitoring of its movements along designated routes. The software component encompasses the user-facing application installed on smartphones and other devices used by commuters to access the bus tracking system. This software is designed with a user-friendly interface, allowing users to view real-time bus locations, check arrival times, plan routes, and receive notifications regarding delays or schedule changes. It interacts seamlessly with the server infrastructure to retrieve and display relevant transit information, ensuring a smooth and intuitive user experience. The server infrastructure serves as the backbone of the Android Bus Tracker System. It comprises a network of servers responsible for processing and storing vast amounts of data, including bus locations, map information, and authentication.

4.1 System Block Diagram

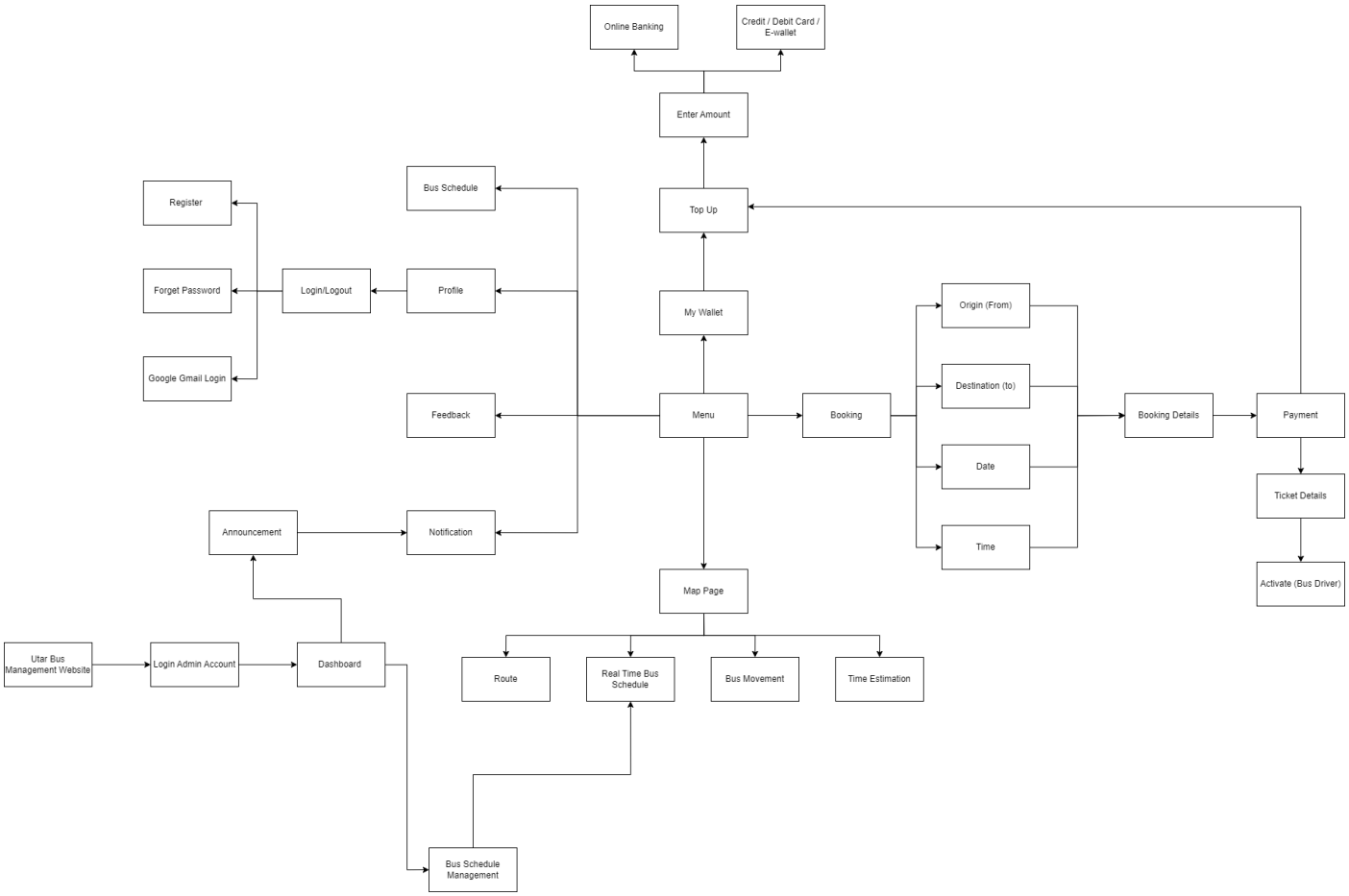


Figure 4.1.1 Block Diagram of Android Bus Tracker System

The system block diagram represents a comprehensive architecture for Android Bus Tracker System designed to facilitate user interaction, booking management, real-time tracking, payment processing and administrative functions. The block diagram is divided into several key modules: the user interface component, the booking and payment system, the real-time tracking, the notification and announcement system, and the administration control panel for managing schedules and routes.

The user interface components include user authentication modules such as registration, login/logout, forgotten password and Google Gmail login. These modules ensure secure access control and enable users to register, log in and manage personal data. The profile management component allows users to update personal information

and view booking history. Once logged in, users can interact with various system functions, including providing feedback and receiving notifications.

The booking and payment system allows users to search for available bus routes and book tickets by selecting trip details such as origin, destination, date and time. The system calculates the fee and provides secure payment options, including online banking, credit/debit card and e-wallet integration. Upon successful payment, the system generates a digital ticket containing the necessary details, which can be accessed by both the user and the bus driver for verification.

The real-time tracking and map page module integrates with GPS services to provide real-time updates on bus location, route and estimated time of arrival (ETA). Users can view this information in real time, enhancing the overall user experience by providing accurate and timely updates. The Notifications and Announcements system allows administrators to send system-wide announcements and push notifications to users via Firebase Cloud Messaging (FCM) to keep them informed of any changes or updates.

The Administrator Control Panel is a web-based interface that provides administrators with a comprehensive tool for managing bus routes, schedules, and announcements. Administrators can perform CRUD (Create, Read, Update, Delete) operations on bus data and monitor the real-time operation of buses to ensure efficient management and timely update of user information. The dashboard also supports the creation and management of announcements, which is essential for effective communication between service providers and users.

4.2 System Components Specifications

Each system component is specified in terms of its functionality, data handling, and integration with other components:

1. User Authentication Components:
    - Register: Allows new users to create an account by providing necessary details such as name, email, and password. Integrates with Firebase Authentication or a similar service for user data storage and validation.
    - Login/Logout: Manages user sessions securely using tokens (e.g., JWT). Supports both manual login via email/password and third-party logins like Google.
    - Forget Password: Implements a secure password recovery process, often via email verification or OTP (One-Time Password).

2. Booking and Payment System:
    - Booking Module: Users select trip details including origin, destination, date, and time. The module queries available routes and buses from the database, calculates fees, and provides seat selection options.
    - Payment Integration: Integrates multiple payment gateways (e.g., Stripe) for secure payments. Supports online banking, credit/debit cards, and e-wallets. Ensures data security through encryption and secure API communications.
    - Ticket Management: Upon successful booking and payment, generates a digital ticket.

3. Real-Time Tracking and Mapping:
    - Map Page: Integrates with real-time GPS tracking services to provide live updates of bus locations and movements. Offers route maps, estimated time of arrival (ETA), and real-time schedule updates.
    - Bus Movement Tracking: Captures bus GPS data and updates the user interface, providing users with real-time bus locations.

4. Admin and Management Components:
    - Admin Dashboard: An interactive web-based dashboard for managing bus schedules, routes, announcements, and user notifications. Allows CRUD (Create, Read, Update, Delete) operations on bus data.
    - Bus Schedule Management: Allows admins to create and modify bus schedules, routes, and timings. Integrates with the backend database (Firestore) to update the data in real-time.

5. Notifications and Announcements:
    - Announcement Module: Allows administrators to create system-wide announcements (e.g., delays, schedule changes) and send them to users.
    - Notification System: Utilizes Firebase Cloud Messaging (FCM) to push notifications to user devices, ensuring timely communication of important information.

4.3 Circuits and Components Design

The components design of the Android Bus Tracker System consists of 7 parts integrate to a page. User can easily know how to handle the application without user manual. The below are the wireframe design layout of all the page have been developed:

Map Page

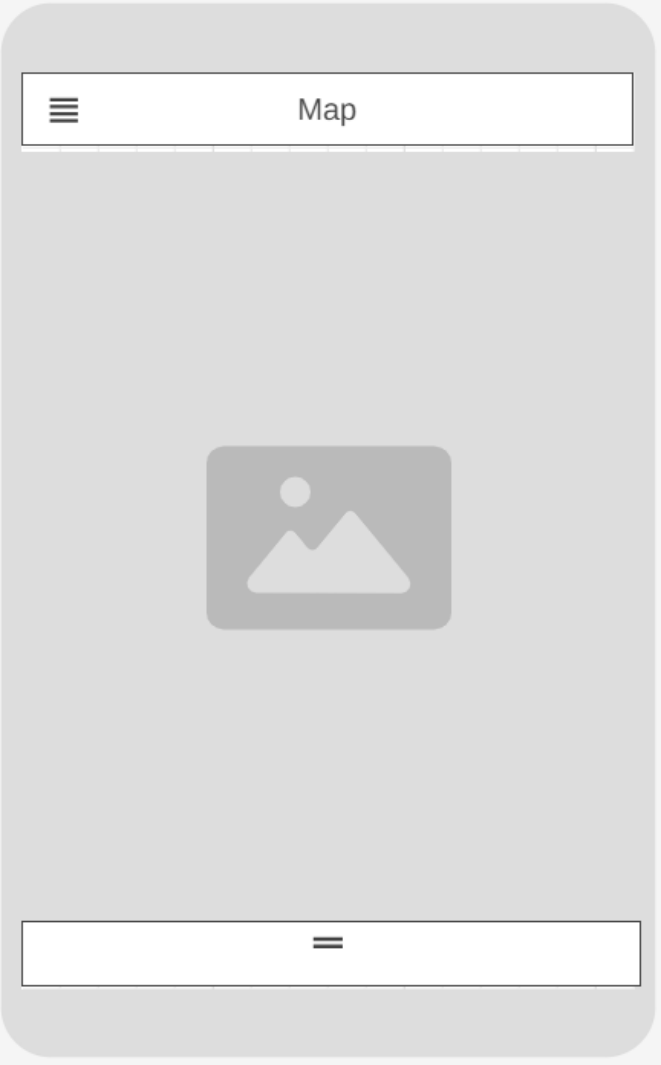


Figure 4.3.1 Wireframe of Map Page

The map page is the core feature of the application's main interface, providing users with a dynamic visual representation of geographic data. Located between the toolbar and the sliding menu, the map occupies a prominent position, ensuring ease of access and visibility. In the upper left corner of the toolbar, there is a menu icon for easy access to other functions and features. When activated, this menu expands to display a variety of option. Meanwhile, a slide-up menu at the bottom of the page

facilitates seamless navigation to the routes page, enabling users to easily plan and explore different bus routes.

Menu Page

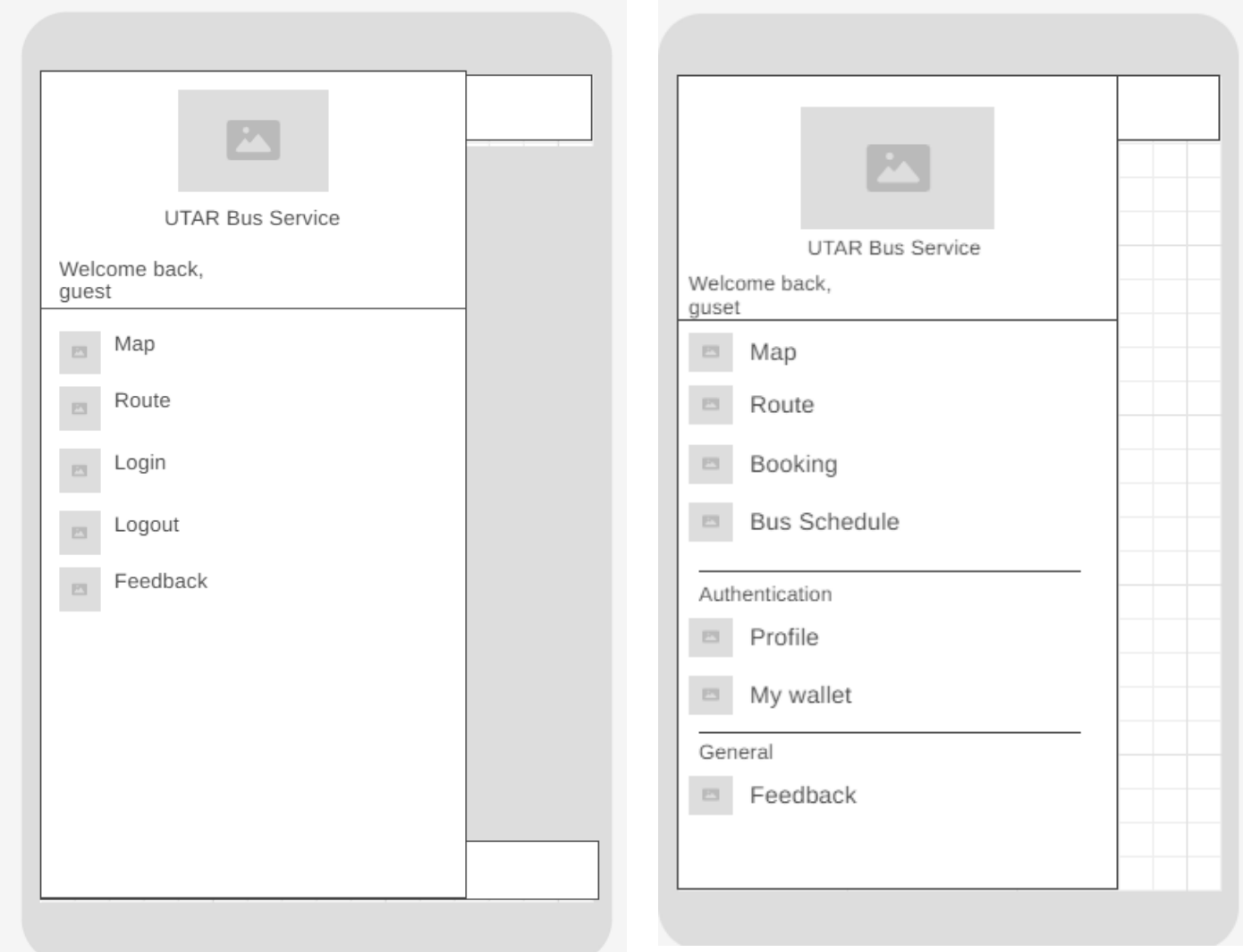


Figure 4.3.2 Wireframe of Menu Page (Before and After)

When a user clicks on the menu icon, it triggers the menu page to slide out from the left side of the screen. This intuitive design feature enhances user experience by providing easy access to a variety of options. Within the menu page, users are presented with a range of choices allowing them to seamlessly navigate through different functionalities. The page has been improved by adding more features that allow users to select features such as bookings, profile (including login and logout) and my wallet.

Route Page

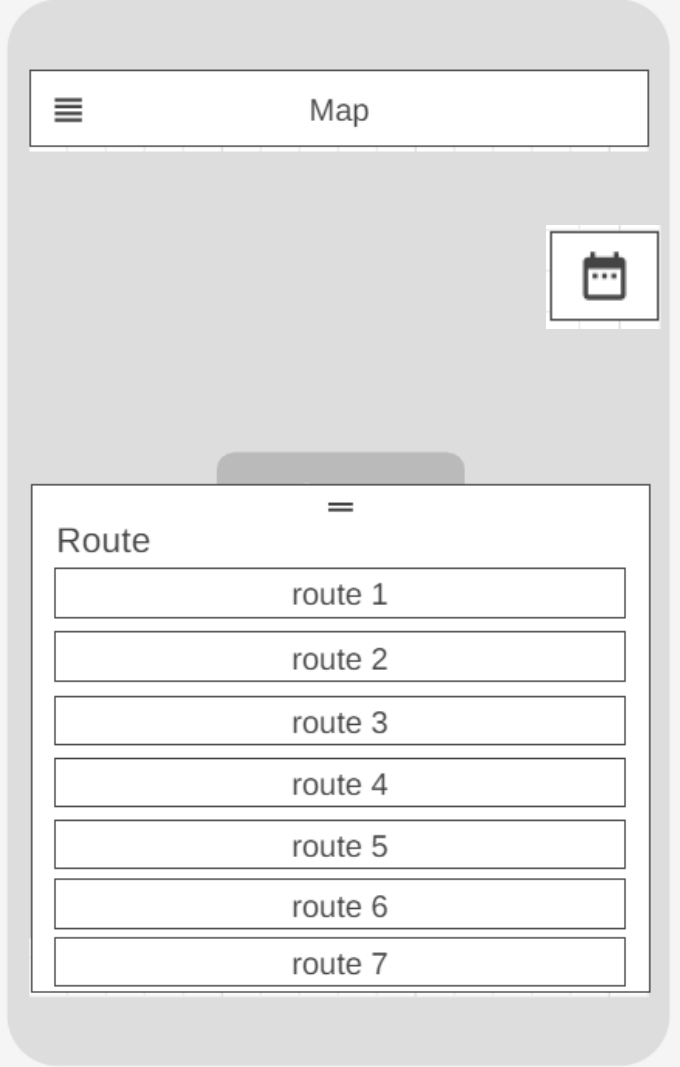


Figure 4.3.3 Wireframe of Route Page

To enhance user experience, the route menu has been designed and presented with a convenient slide-up function for seamless navigation. It has been designed with the user in mind, providing a straightforward bus route selection process that caters to the different needs of passengers. Within this intuitive interface, users can choose from seven different bus routes. This thoughtful design ensures that passengers have plenty of options to customize their journey according to their specific destinations and preferences. We have also added a calendar icon, which can be clicked to view the real-time timetable for each bus route.

Real Time Bus Schedule

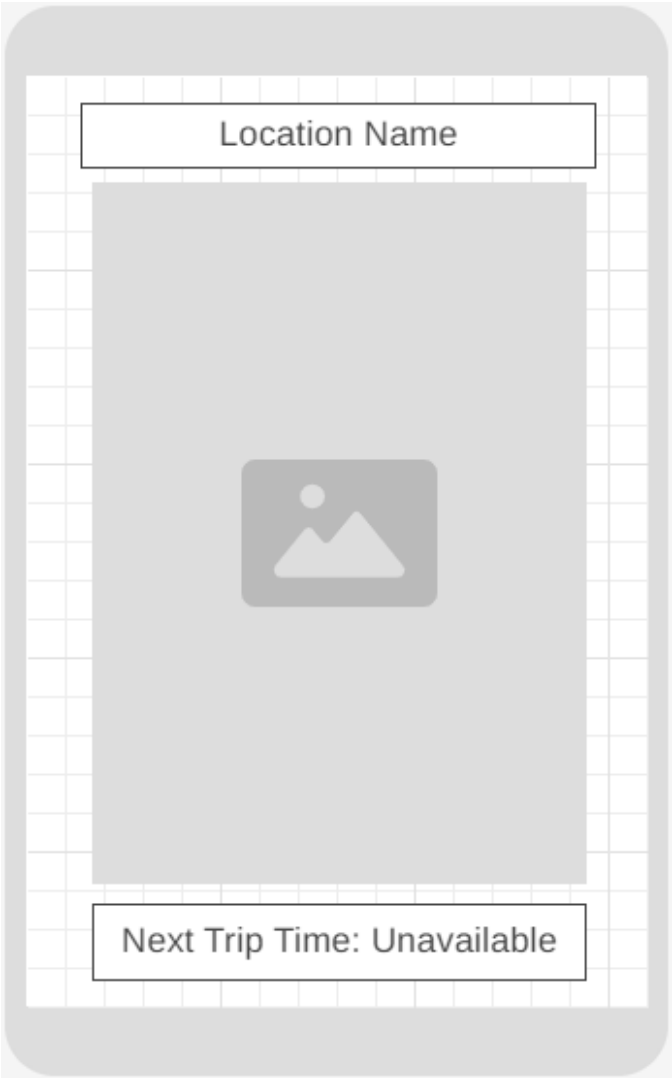


Figure 4.3.4 Wireframe of Real Time Bus Schedule

To make it easier for users to view the bus timetable for each route, real time bus schedule provides users with a concise image and schedule. The name of the bus stop has a placeholder that allows the user to see which stop's timetable they are viewing. This makes it easy for the user to identify the current location. The center shows the time schedule for the user. At the bottom, there is a section displaying the "Next Trip Time." In the current state, it reads "Unavailable", suggesting that in real-time usage, this would dynamically update to show the time of the next scheduled bus arrival at the location.

Profile Page

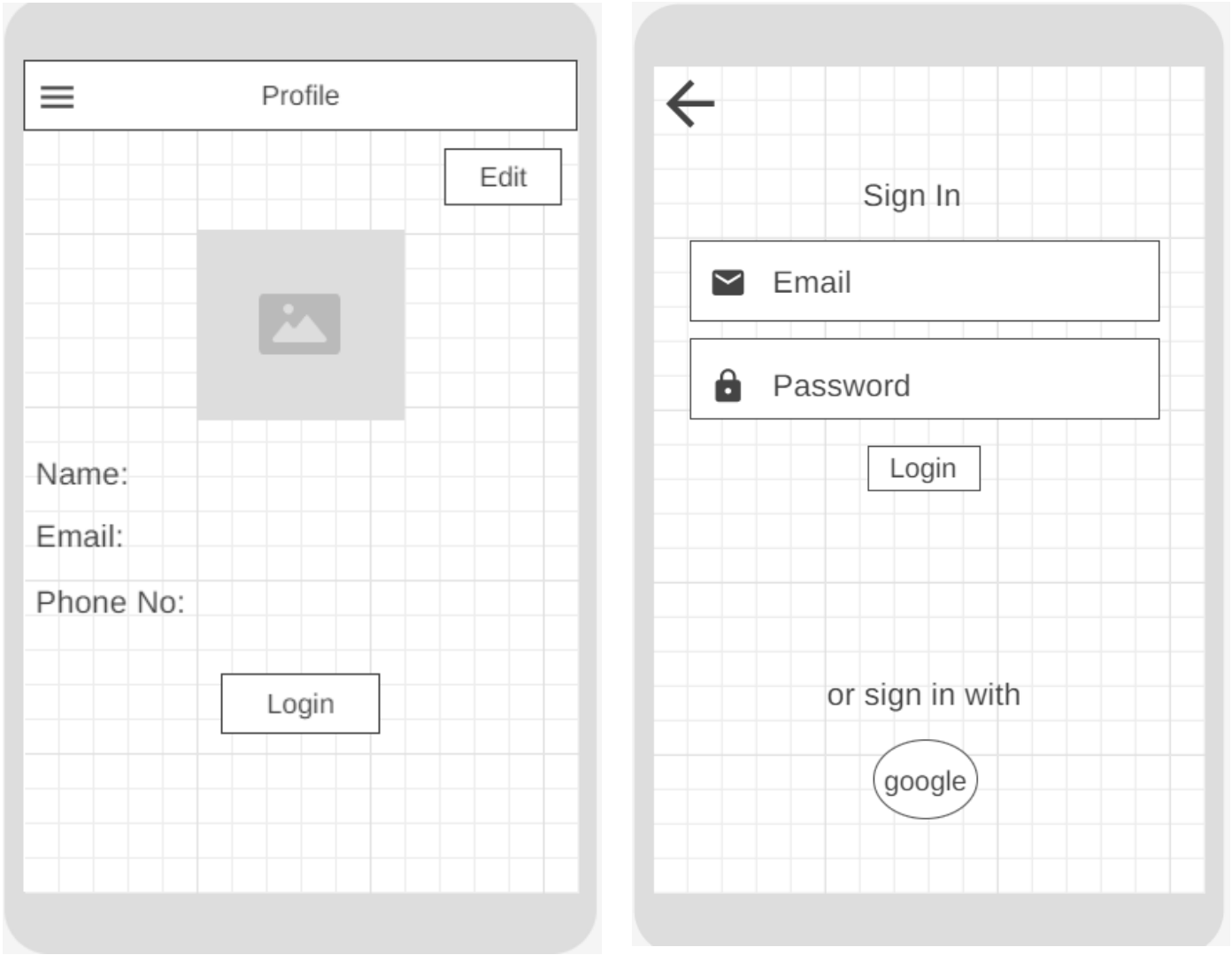


Figure 4.3.5 Wireframe of Profile Page

Profile page is designed to display a user's personal information and account settings in a clear and organized manner. At the top, there will usually be a section displaying the user's profile picture as well as name, email address and phone number for easy identification. The login button will direct the user to the login page, where the user can sign in to their account using their email and password, or they can sign in using Google account.

Edit Profile Page

Figure 4.3.6 Wireframe of Edit Profile Page

On the Edit Profile page, users can easily and flexibly update their profile. Users can select an image from their device and upload or change their profile picture to help them personalise their account. Users can also edit their name to reflect any changes, such as name corrections or updates. Finally, users can also update their phone number to ensure that they can be contacted in case they receive any important notifications or encounter problems related to their account.

Booking Page

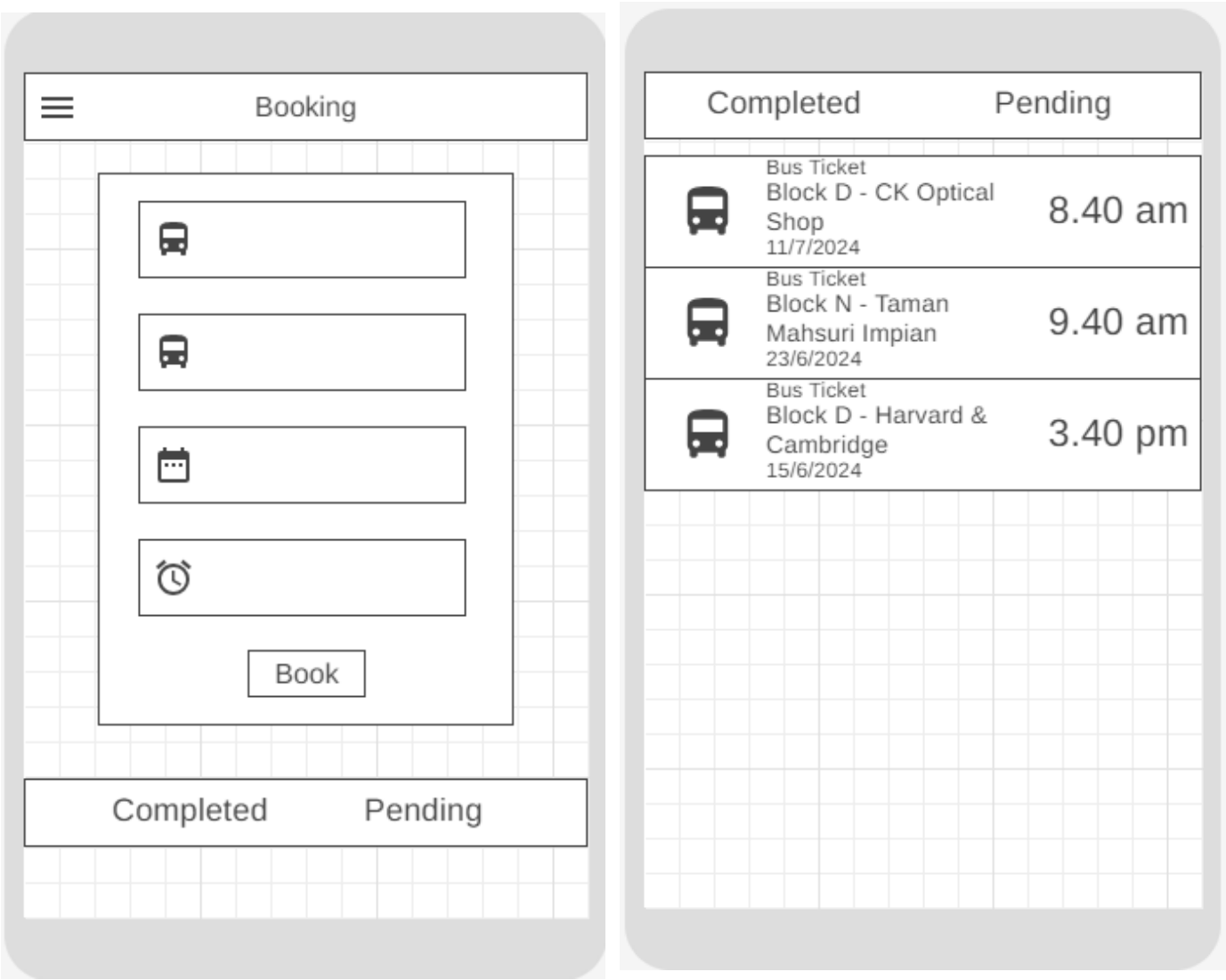


Figure 4.3.7 Wireframe of Booking Page

The booking page designed allows the user to select key details for scheduling a bus trip, including origin (from), destination (to), date and time. Each field has an intuitive icon and corresponding input area, making the selection process simple and straightforward for the user. Once all the details are filled in, the user can press the 'Book' button to complete the trip booking. In addition, there are two tabs on the interface, 'Completed' and 'Pending', which help users keep track of past and upcoming bookings. The Completed Trips section displays basic details such as bus ticket information, destination and departure time, making it easy for users to view trip history.

Booking Details Page

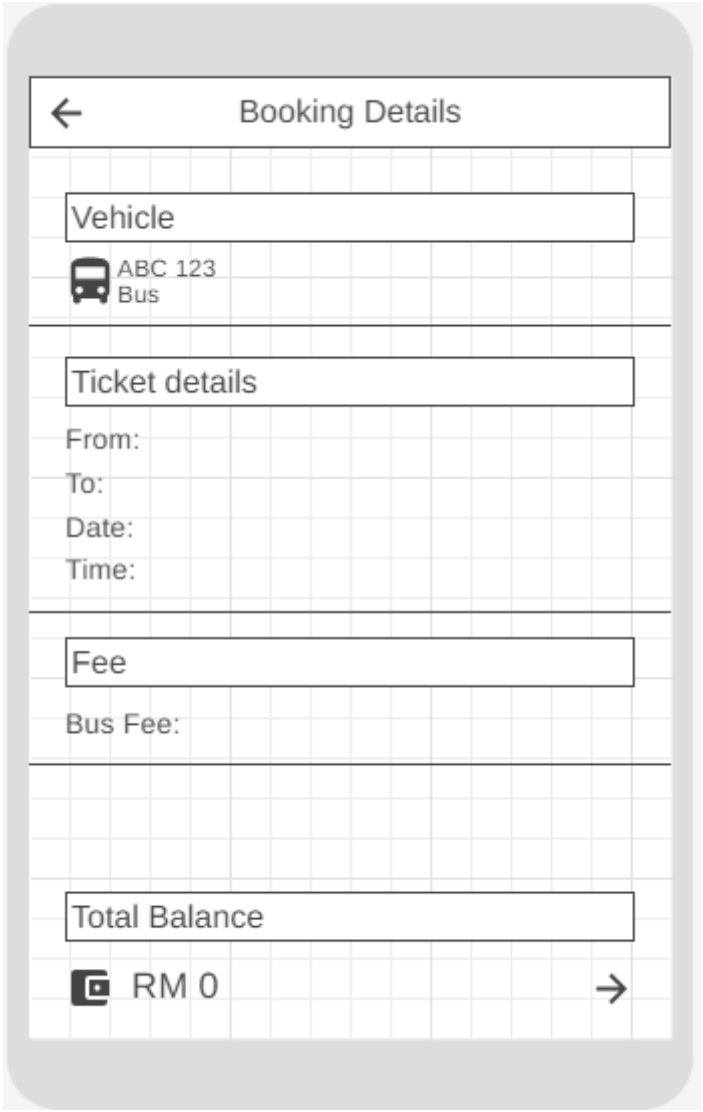


Figure 4.3.8 Wireframe of Booking Details Page

The booking Details page designed allows the user check again the booking details before make a payment. The design concept is to show all the important information to the user including the bus liscene number plate, the origin, destination, date, time, bus fee, and the total balance.

My Wallet Page

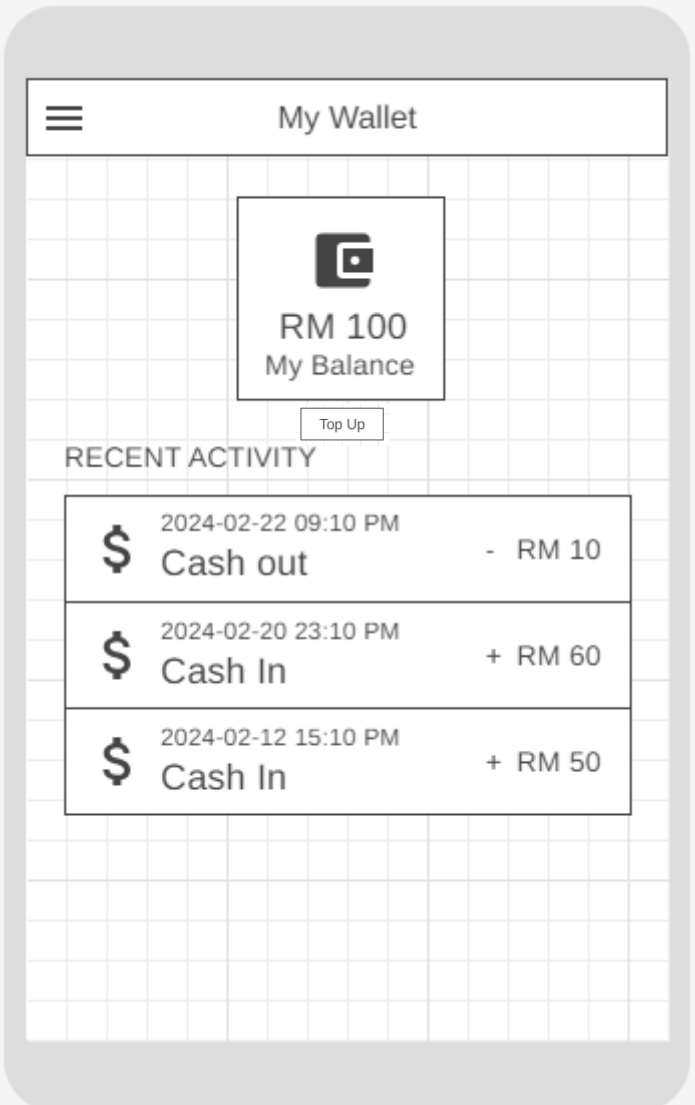


Figure 4.3.9 Wireframe of My Wallet Page

The Wallet page allows users to view and manage their digital wallet balance and recent transactions. The page prominently displays key information, including the current balance at the top, as well as recent transactions such as cashouts and withdrawals. Each transaction entry has an intuitive icon, clear description and corresponding amount, making it easy for users to track their wallet activity. After viewing balances and transactions, users also can click the top up button navigate to top up page. In addition, the interface is simple and straightforward, helping users keep track of their finances with ease. The Recent Activity section displays basic information such as transaction type, amount and date, allowing users to get an overview of past wallet activity briefly.

Top Up Page

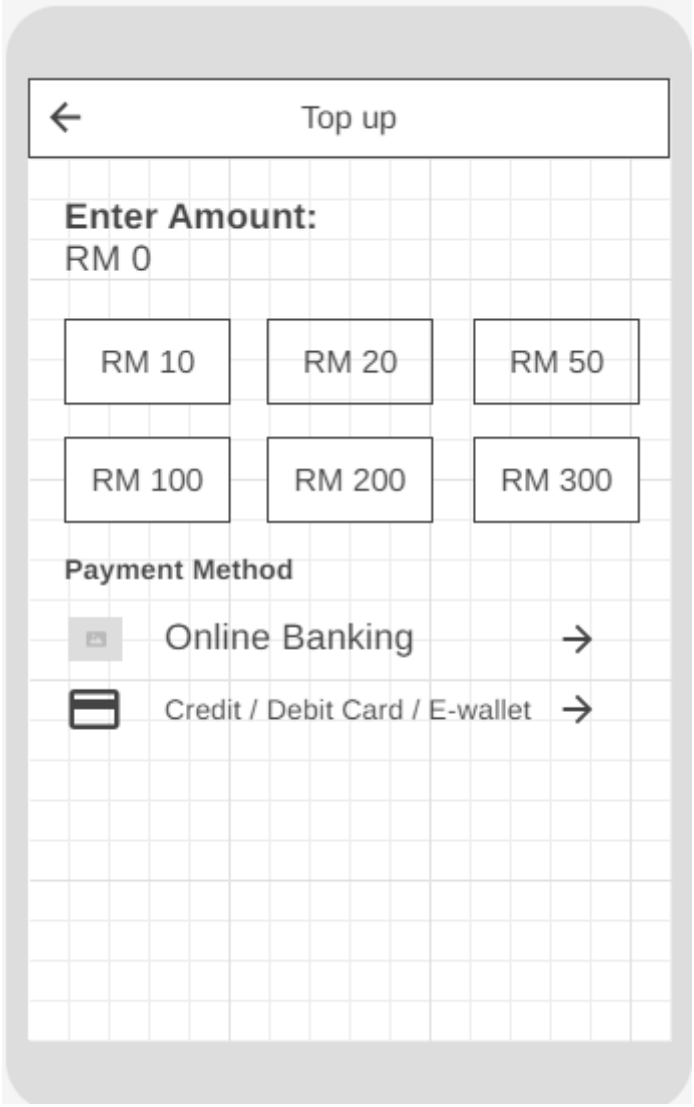


Figure 4.3.10 Wireframe of Top Up Page

The Top up page offers a simple and user-friendly interface for adding funds to a digital wallet. Users can choose from preset amounts such as RM 10, RM 20, RM 50, RM 100, RM 200, and RM 300, making the process quick and straightforward. The page also provides two payment options—Online Banking and Credit/Debit Card or E-wallet—each with clear icons and an arrow for further navigation.

Ticket Details Page

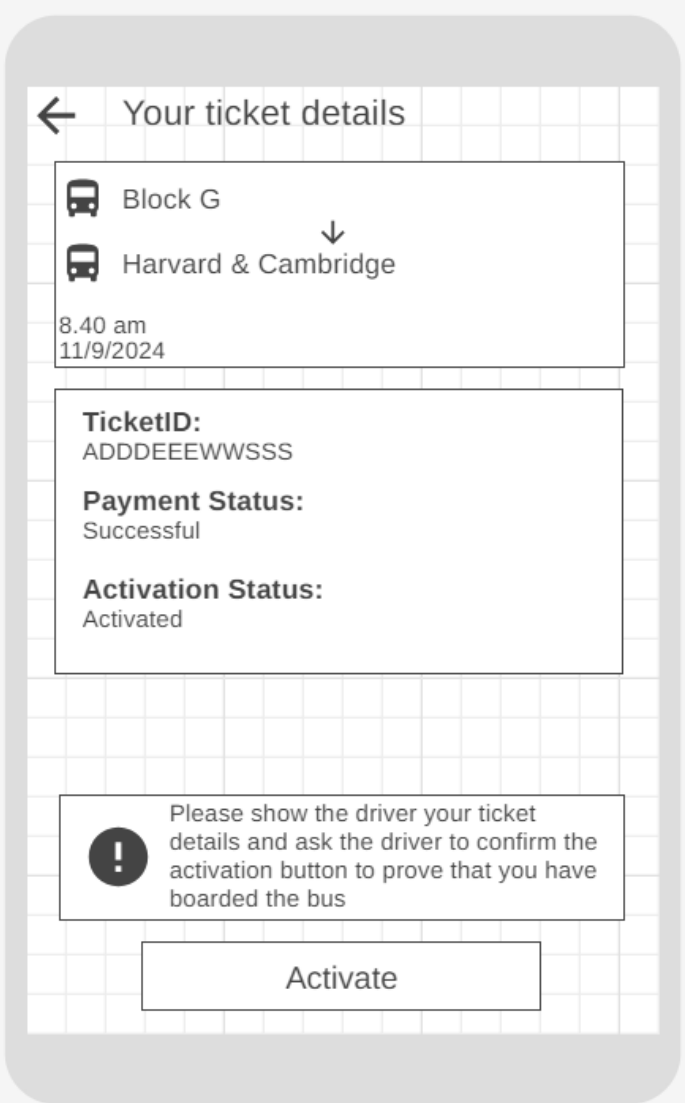


Figure 4.3.11 Wireframe of Ticket Details Page

The ticket details page provides a comprehensive overview of the user's bus journey. Departure and arrival locations are clearly displayed at the top. Departure and arrival times are shown at the bottom. Key details about the ticket are displayed in the next section, including a unique ticket ID, payment status and activation status, ensuring that the user has all the relevant information for the journey. Below this section is a message advising the user to show the ticket details to the driver and ask them to confirm activation as proof of boarding. Finally, an ‘Activate’ button at the bottom provides the user with additional options to complete the ride process.

Notification Page

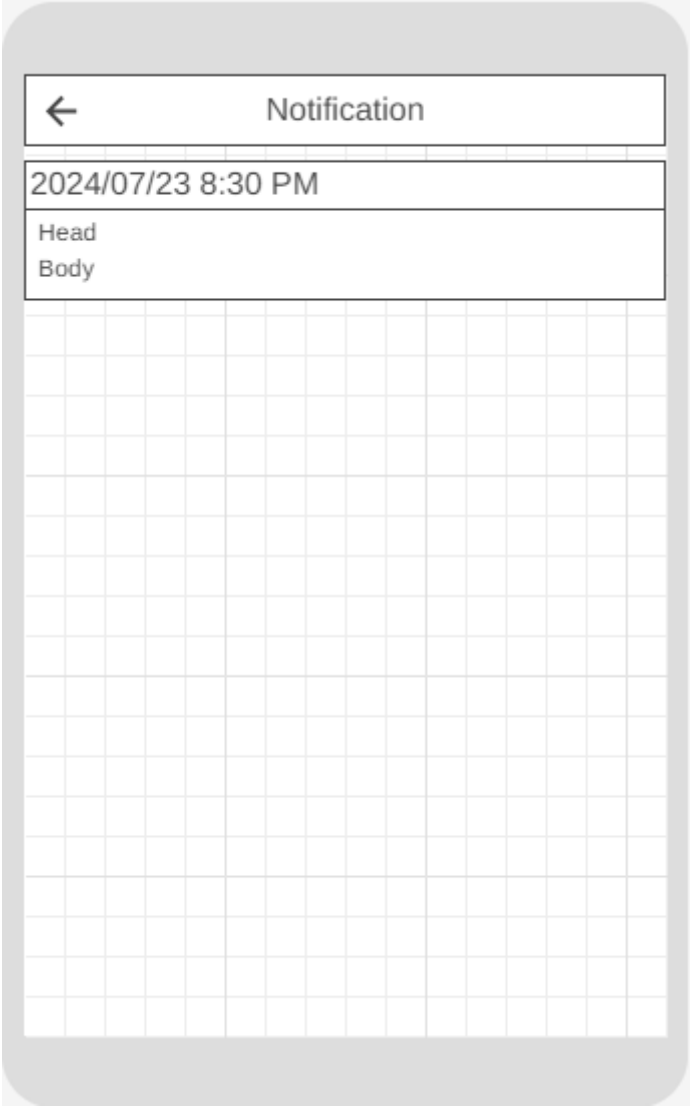


Figure 4.3.12 Wireframe of Notification Page

The notification page is designed with the idea that users can view notifications sent by the administrator. Each notification will be displayed in the recycle view.

Feedback Page

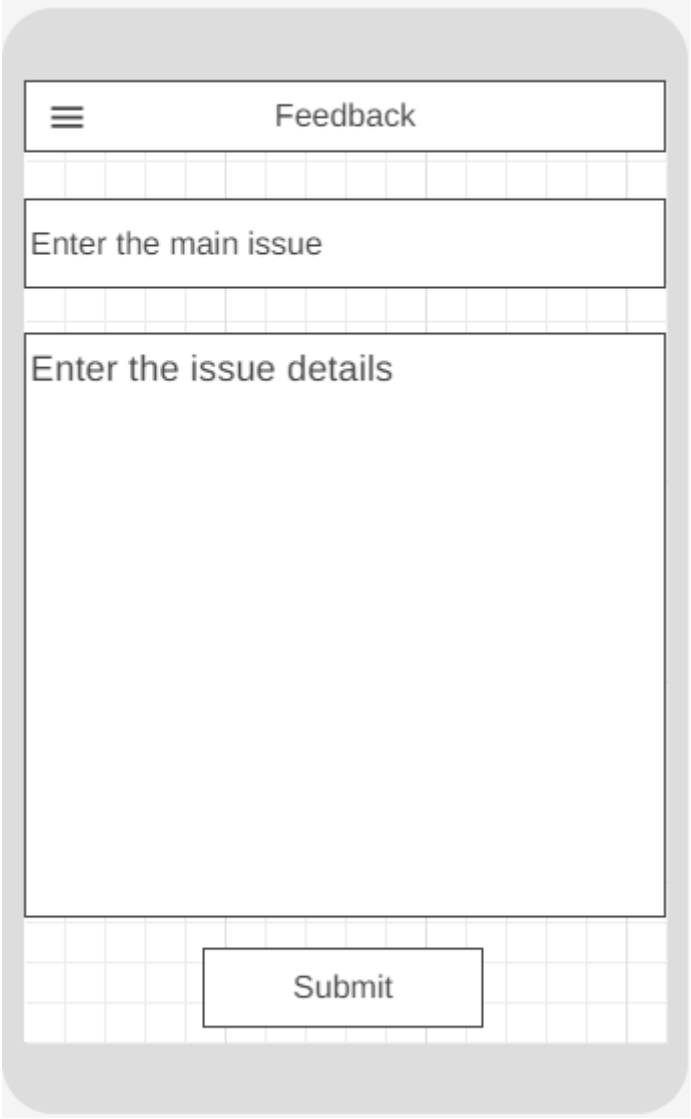


Figure 4.3.13 Wireframe of Feedback Page

The feedback page is designed to facilitate users to give feedback when they encounter any problems in the application.

Admin Website Login

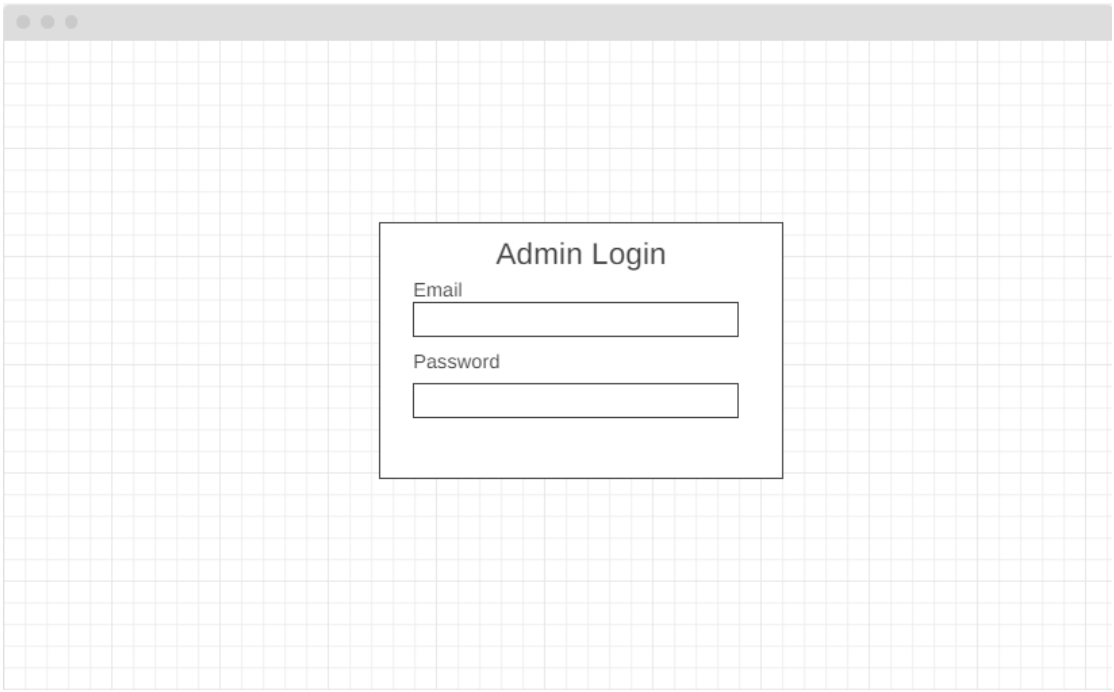


Figure 4.3.14 Wireframe of Admin Website Login

This login page is designed to only for administrators to log in.

Admin Website Dashboard

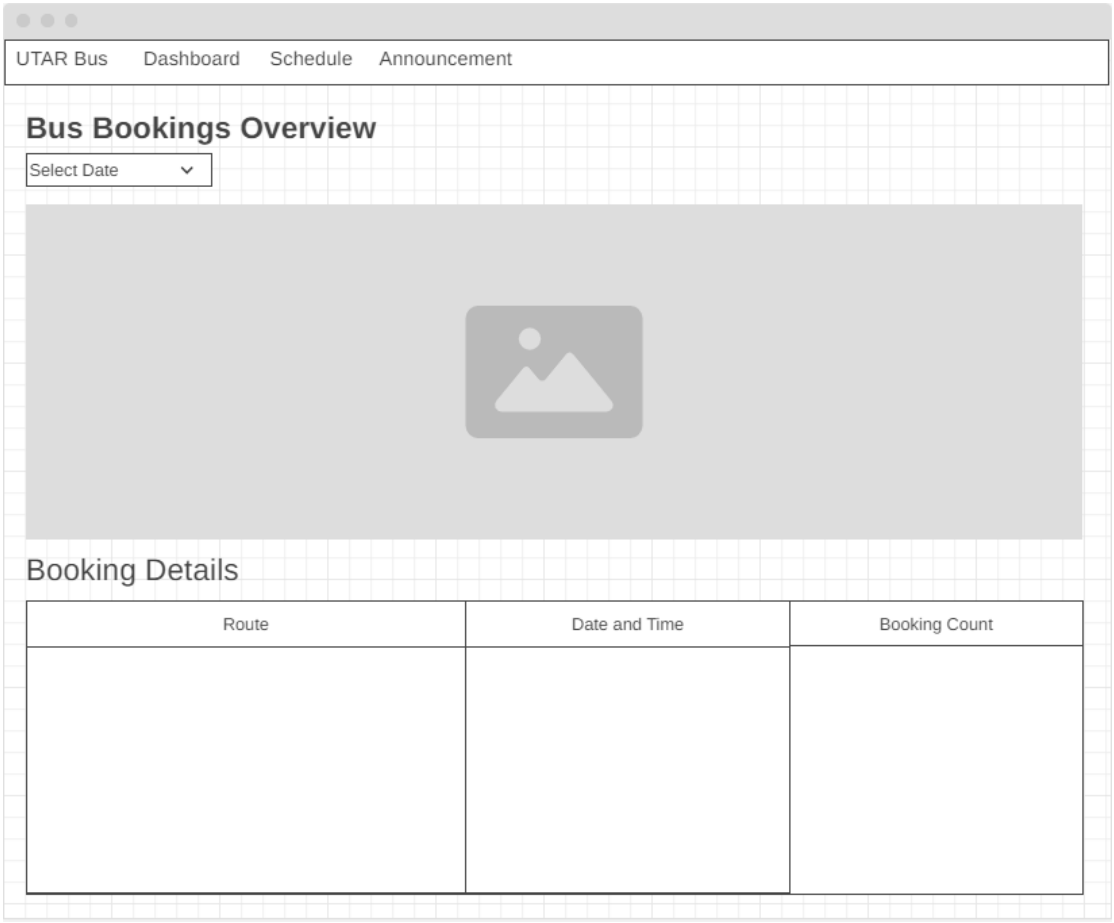


Figure 4.3.15 Wireframe of Admin Website Dashboard

The dashboard page is designed for administrators to view bus booking records. In this page, the heat map will be displayed in the middle of the page, and all the records will be plotted in the red dot around the bus stop icon. The administrator can know which bus station has many users taking the bus. The reservation details below will show the route (from the start point to the end point), date and time, and the booking count (how many times the user has booked).

Admin Website Schedule

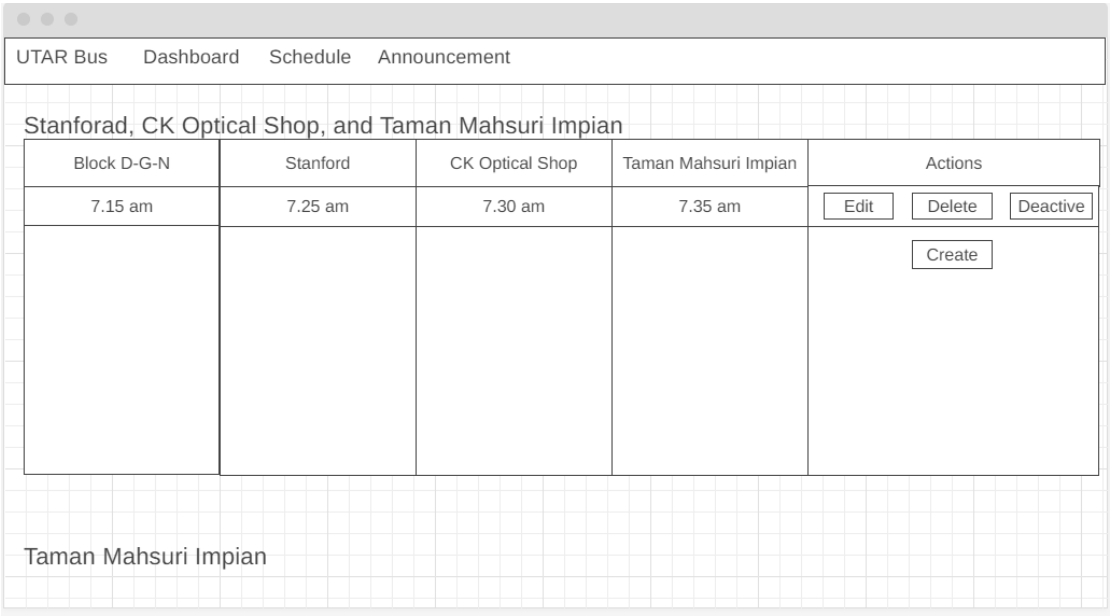


Figure 4.3.16 Wireframe of Admin Website Schedule

The Schedule page is designed for administrators to perform CRUD operations to manage each bus trip in each route, including changing the time of bus trips or deactivate a trip from the route,

Admin Website Announcement

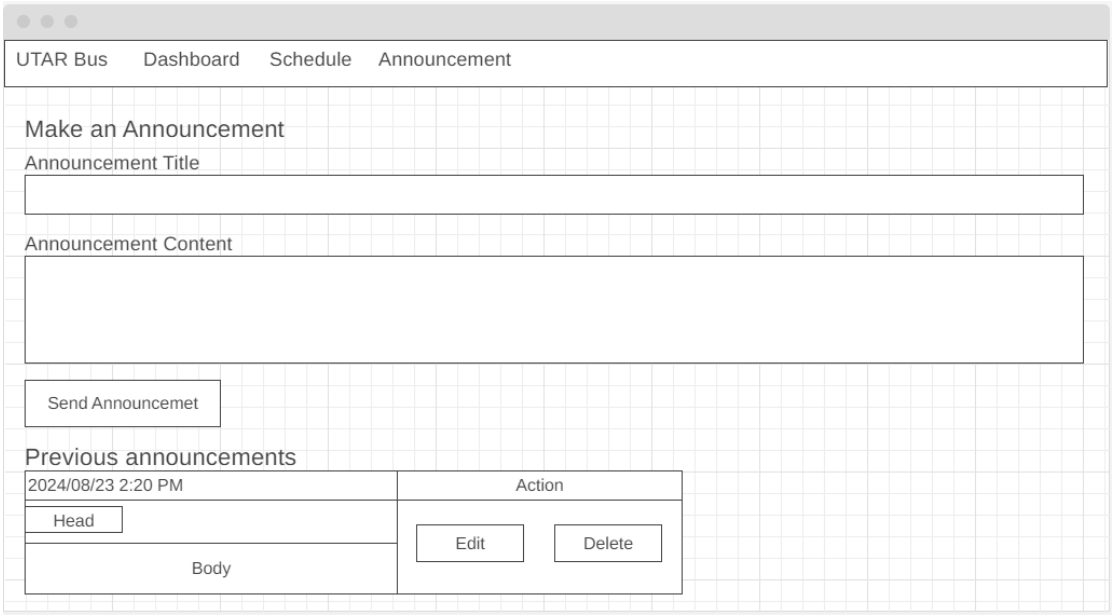


Figure 4.3.17 Wireframe of Admin Website Announcement

The announcement page is designed for administrators to make announcements to inform users of important matters or changes in bus running time. In addition to notifying users, administrators can see past announcements and can change or delete them.

Admin Website Feedback

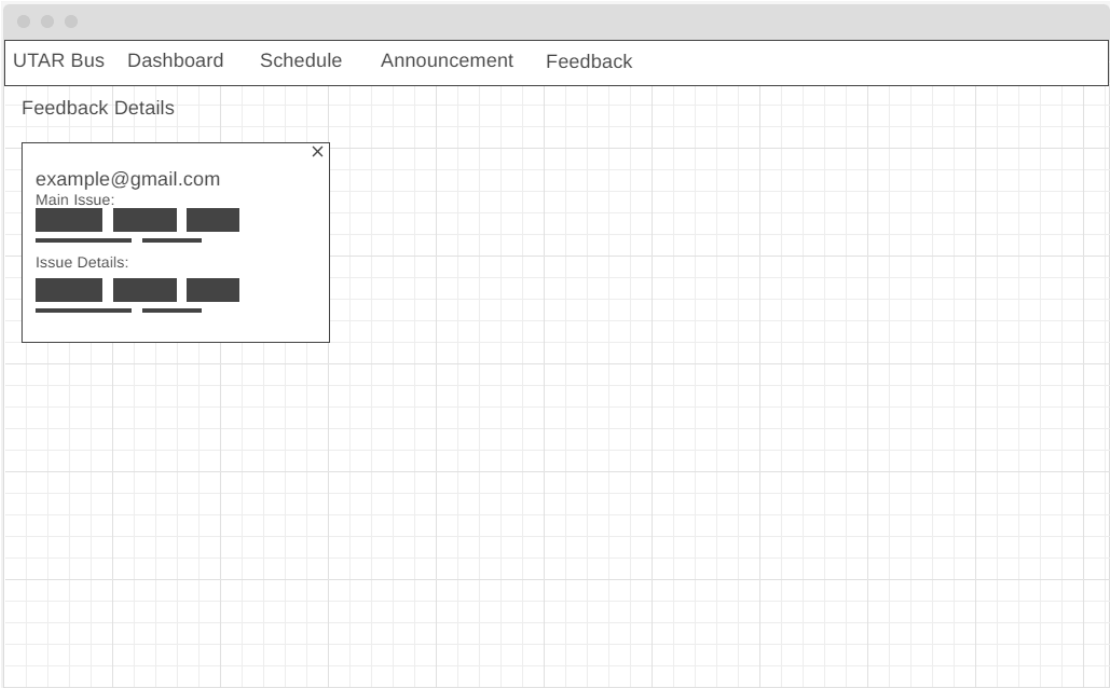


Figure 4.3.18 Wireframe of Admin Website Feedback

The feedback page is designed to allow administrators to view all the feedback submitted by users. In addition, administrators can delete confused feedback or feedback that has existed for too long.

4.4 Firebase Database Design (NoSQL)

4.4.1 Realtime Database Structure

Coordinates

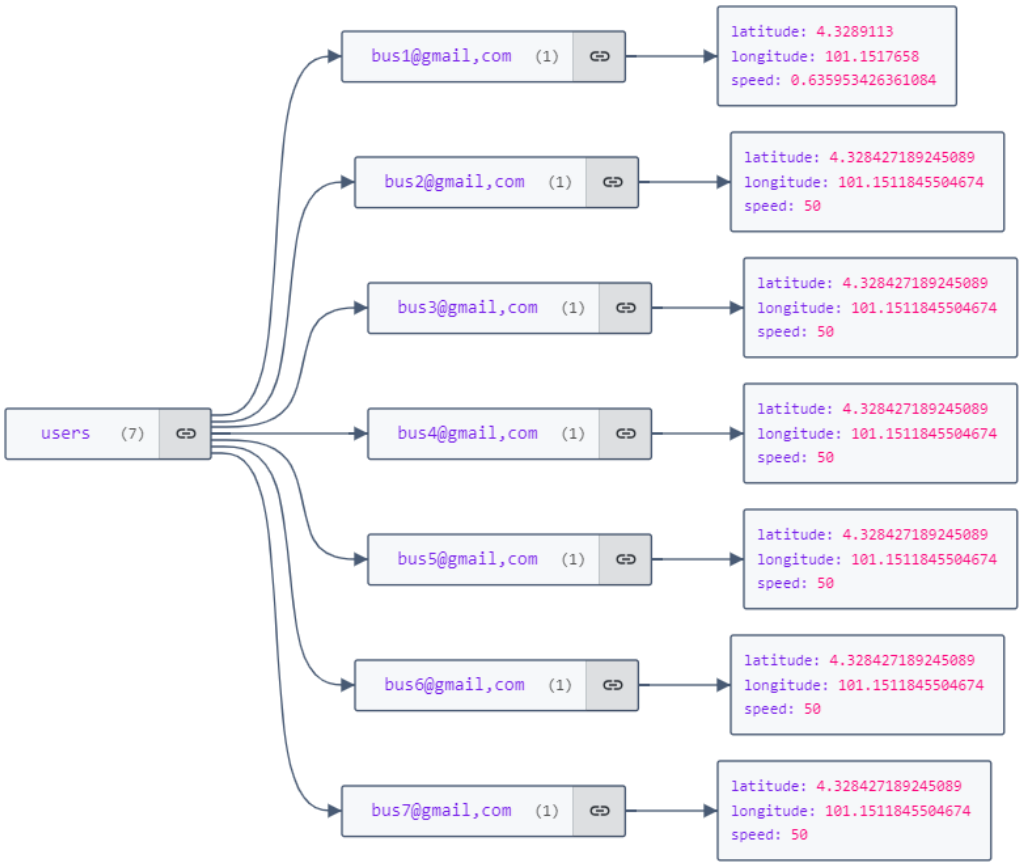


Figure 4.4.1 Coordinates JSON Format

4.4.2 Firestore Database Structure

Locations

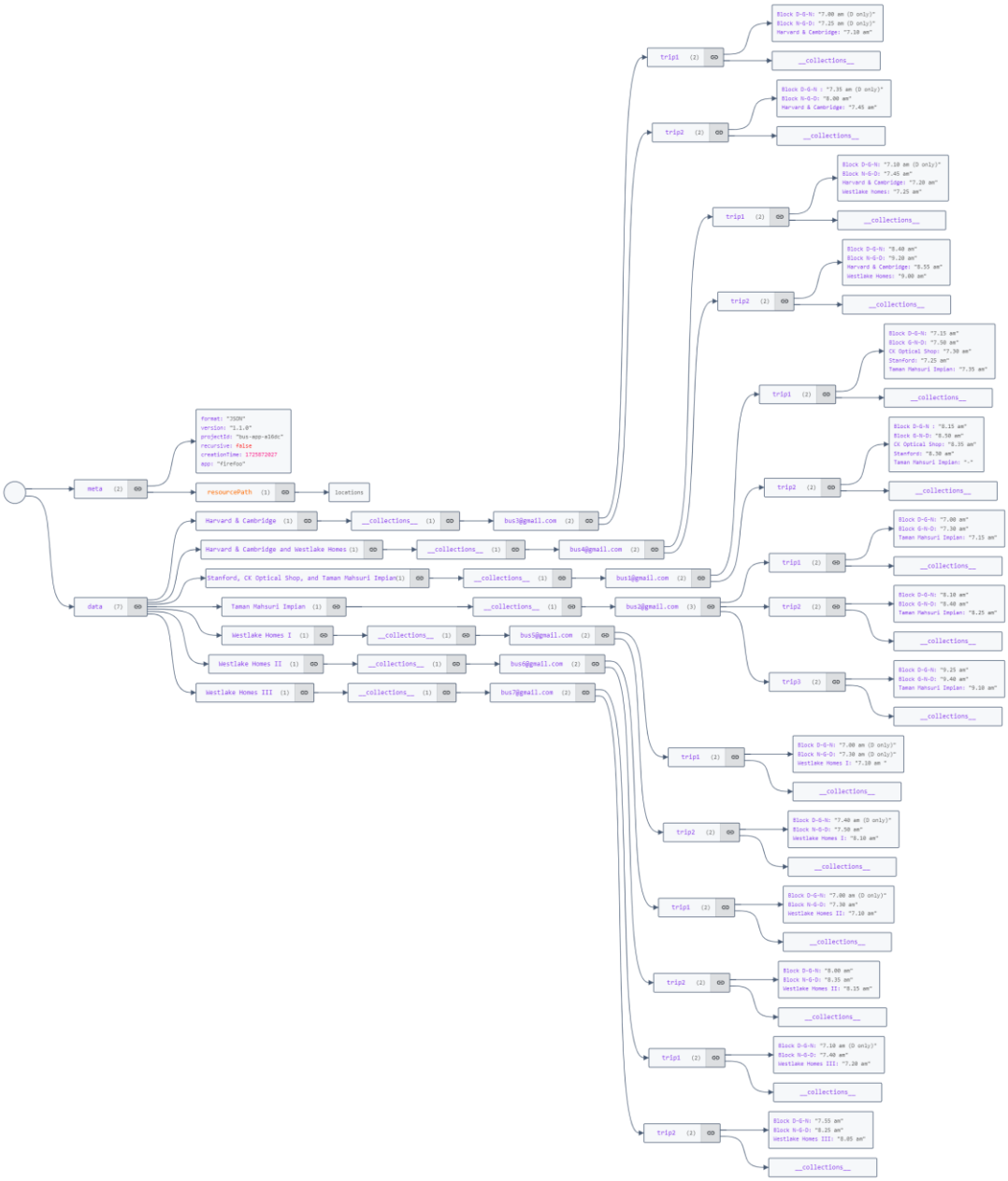


Figure 4.4.2 Locations JSON Format

Users

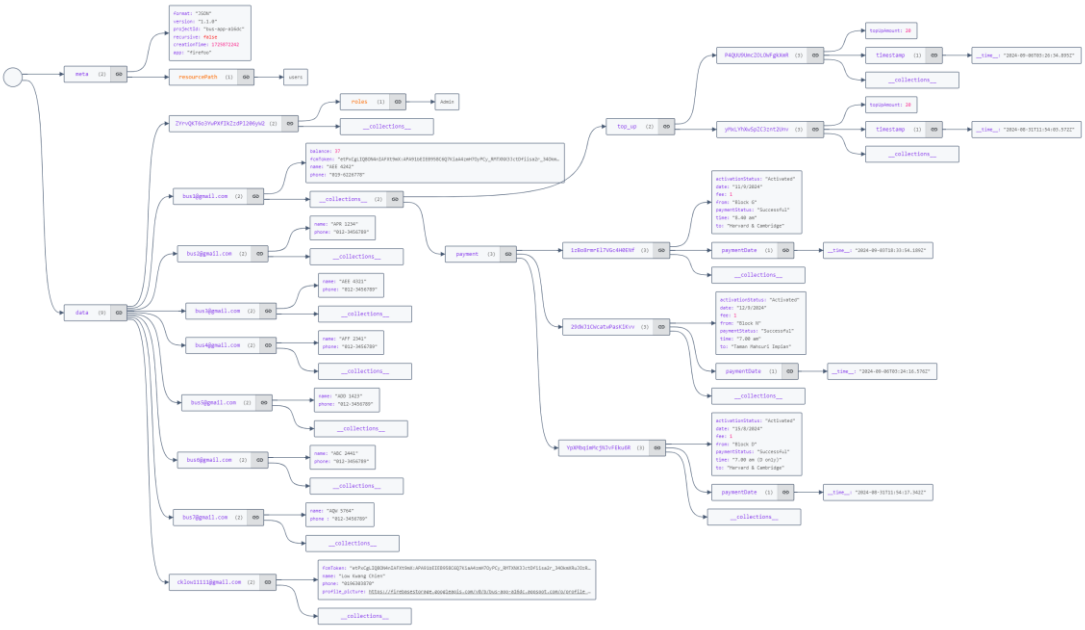


Figure 4.4.3 Users JSON Format

Notifications

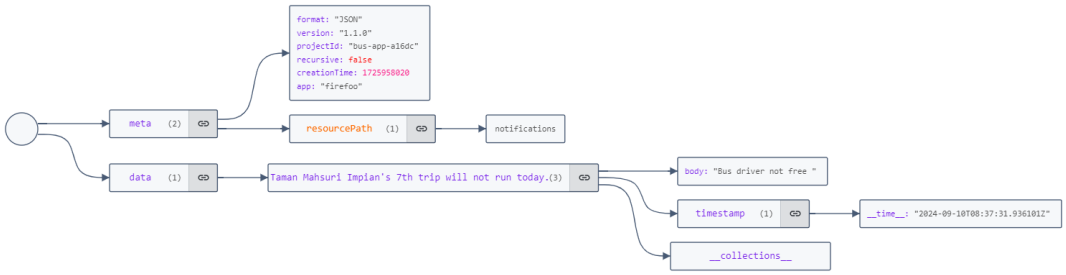


Figure 4.4.4 Notifications JSON Format

Feedback

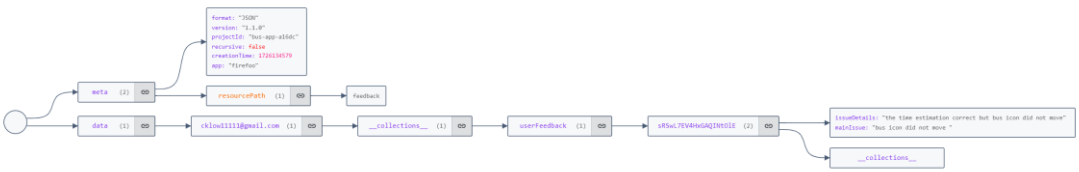


Figure 4.4.5 Feedback JSON Format

CHAPTER 5

System Implementation

5.1 Hardware Requirements & Setup

The Android Bus Tracker System involves both computer hardware for development purposes and Android smartphones for testing and end-user interaction. Here's an overview of the hardware involved:

Description	Specifications
Model	Asus A409 series
Processor	Intel Core i7-1065G7
Operating System	Windows 11
Graphic	NVIDIA GeForce GT MX300 2GB GDDR5
Memory	20 GB DDR4 RAM
Storage	500GB SSD & 500GB SATA HDD

Table 5.1 Specifications of laptop

Description	Specifications
Model	Realme C25Y
Processor	Octa-core (2x1.8 GHz Cortex-A75 & 6x1.8 GHz Cortex-A55)
Operating System	Android 11
Graphic	Mali-G52 MP2
Memory	4GB RAM
Storage	128GB

Table 5.2 Specifications of Android Smartphone

5.2 Software Setup

Android Studio 2024.1.1.13

Android Studio is the official Integrated Development Environment (IDE) for Android app development. Version 2024.1.1.13 offers various features to help design, code, and test mobile applications. Key features include:

- Code editor with real-time error detection and code completion.
- Emulator for testing apps on different devices and configurations.
- Build tools for compiling, debugging, and packaging Android apps for deployment.

Mobile Atlas Creator 2.3.3

Mobile Atlas Creator (MOBAC) is a tool used to create offline maps for mobile applications. Version 2.3.3 enables to:

- Create custom map layers from online map sources.
- Export maps in formats suitable for Android apps or GPS devices.
- Use the maps in apps that require offline geographical data.

Google Cloud Platform – Firebase

Firebase is a comprehensive platform from Google for building mobile and web apps. For the project, several Firebase components are integrated:

- Authentication: Manages user identity and authentication (e.g., email/password, social login).
- Realtime Database: Stores and syncs data in real-time, allowing instant data updates between the mobile app and database.
- Firestore Database: A NoSQL cloud database designed for scalable, flexible queries and storing more complex structured data than Realtime Database.
- Storage: Manages and stores user-generated content like images, videos, and other files.
- Messaging: Allows sending notifications and messages to users, improving app engagement and interaction.
- Functions: Serverless code that can be triggered by Firebase events (e.g., database changes, HTTP requests) to perform backend tasks without managing servers.

Visual Studio Code

Visual Studio Code is a lightweight, highly customizable text editor that supports multiple programming languages and frameworks. Its features include:

- Extensions: Support for Firebase, Android development, and other technologies through plugins.
- Built-in Git support for version control.
- Debugging and syntax highlighting for various programming languages, which is useful for writing backend services, Firebase functions, and handling app development.

5.3 Setting and Configuration

There are some of the software and services need to be used when develop the Android Bus Tracker System:

i. Android Studio 2024.1.1.13

ii. Mobile Atlas Creator 2.3.3 (2023-10-13)

iii. Google Cloud Platform – Firebase

Map Generator and Storage

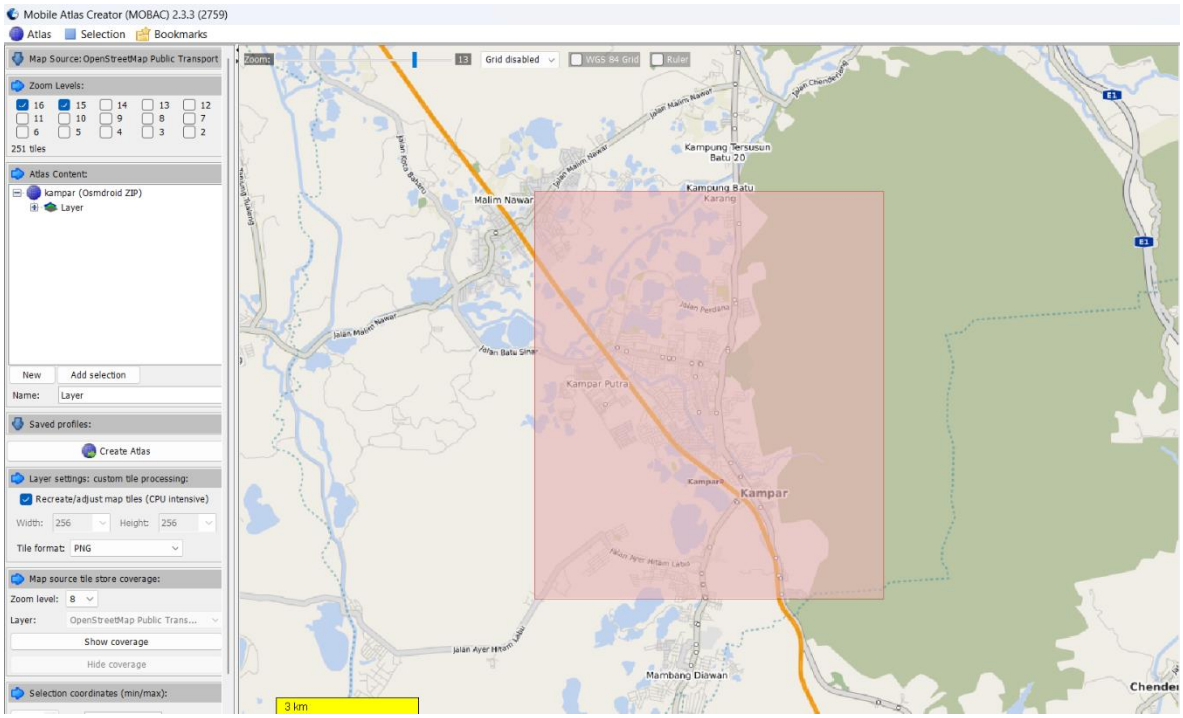


Figure 5.3.1 Capture Kampar Area from Mobile Atlas Creator

Before diving into the development of the Android Bus Tracker System, it's crucial to establish a solid foundation by structuring the custom map source using a mobile atlas creator. This entails selecting the appropriate map data and formats, configuring the atlas creator software, and generating offline maps tailored to the application's needs. By creating offline maps, the system can provide reliable navigation and location services to users even in areas with limited or no internet connectivity.

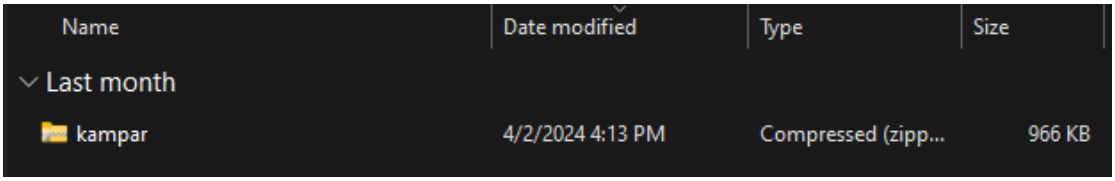


Figure 5.3.2 Kampar.zip file

Many open-source maps are available in Mobile Atlas Creator. To fulfil the requirements of the Osmdroid library, Android Bus Tracker System uses OpenStreetMap Public Transport as the main map source and sets the zoom level to 15 to 16, which means that the user can only zoom in and out of this zoom level. Besides, the maps will be downloaded in ZIP format and named Kampar to save cloud storage space.

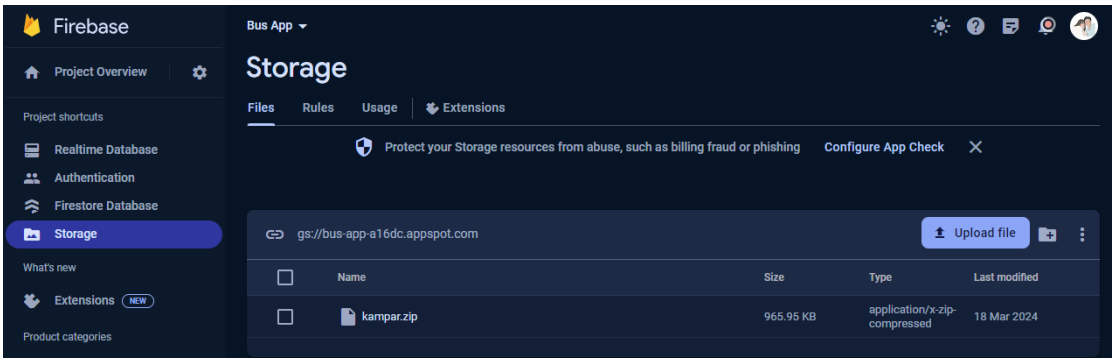


Figure 5.3.3 Store the Kampar.zip file in Firebase Cloud Storage

The purpose of storing the Kampar.zip file in the Firebase cloud storage is to allow users to retrieve maps from the Firebase cloud storage and install them on their smartphones after granting permissions to the Android Bus Tracker System.

Android Studio

Android Studio Manifest Setup

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.POST_NOTIFICATIONS" />
<uses-permission android:name="android.permission.ACCESS_NOTIFICATION_POLICY" />
<uses-permission
    android:name="android.permission.READ_EXTERNAL_STORAGE"
    android:maxSdkVersion="32" />
<uses-permission
    android:name="android.permission.WRITE_EXTERNAL_STORAGE"
    android:maxSdkVersion="32"
    tools:ignore="ScopedStorage" />
<uses-permission
    android:name="android.permission.MANAGE_EXTERNAL_STORAGE"
    tools:ignore="ScopedStorage" />
```

Figure 5.3.4 Permissions needed for the Manifest Setup

The permissions setup in the AndroidManifest.xml file ensures the Android Bus Tracker System by granting access to the system resources. Location permissions (ACCESS_FINE_LOCATION and ACCESS_COARSE_LOCATION) allow the app to track bus locations accurately. Internet and network state permissions enable real-time communication with Firebase and handle network connectivity changes. Notification permissions allow the app to post alerts for schedule updates or emergencies. Storage permissions (READ, WRITE, and MANAGE_EXTERNAL_STORAGE) enable managing map data and user-generated content, while scoped storage policies are bypassed for broader storage access. This setup ensures smooth app functionality, from real-time tracking to offline map management and notification handling.

Build.Gradle.Kts (:app)

```
dependencies {

    // Import the Firebase BoM
    implementation(platform("com.google.firebase:firebase-bom:33.2.0"))

    //Firebase features
    implementation("com.google.firebase:firebase-analytics")
    implementation("com.google.firebase:firebase-database:21.0.0")
    implementation("com.google.firebase:firebase-auth:23.0.0")
    implementation("com.google.firebase:firebase-storage:21.0.0")
    implementation("com.google.firebase:firebase-core:21.1.1")
    implementation("com.google.firebase:firebase-messaging:24.0.1")
    implementation("com.google.firebase:firebase-firestore:25.1.0")
    implementation("com.google.firebase:firebase-functions:21.0.0")

    //Open source map
    implementation("org.osmdroid:osmdroid-android:6.1.18")

    //Profile picture upload features
    implementation ("com.github.bumptech.glide:glide:4.16.0")
    implementation("androidx.activity:activity:1.9.1")
    annotationProcessor ("com.github.bumptech.glide:compiler:4.12.0")

    //Stripe
    implementation ("com.stripe:stripe-android:20.48.6")

    //fuel
    implementation ("com.github.kittinunf.fuel:fuel:2.3.1")
    implementation ("com.github.kittinunf.fuel:fuel-json:2.3.1")

    implementation("androidx.appcompat:appcompat:1.6.1")
    implementation("com.google.android.material:material:1.11.0")
    implementation("androidx.constraintlayout:constraintlayout:2.1.4")
    implementation("com.google.android.material:material:1.11.0")
    implementation("androidx.lifecycle:lifecycle-livedata-ktx:2.7.0")
    implementation("androidx.lifecycle:lifecycle-viewmodel-ktx:2.7.0")
    implementation("com.google.android.gms:play-services-location:21.2.0")
```

```
implementation("androidx.swiperefreshlayout:swiperefreshlayout:1.1.0")
implementation("com.google.android.gms:play-services-maps:18.2.0")
implementation("androidx.navigation:navigation-fragment:2.7.7")
implementation("androidx.navigation:navigation-ui:2.7.7")
implementation("com.google.android.gms:play-services-auth:20.3.0")
testImplementation("junit:junit:4.13.2")
androidTestImplementation("androidx.test.ext:junit:1.1.5")
androidTestImplementation("androidx.test.espresso:espresso-core:3.5.1")
```

Figure 5.3.5 Implementation needed for the Build.Gradle.Kts (:app)

This Android project integrates various libraries and tools to provide a variety of functions. Firebase services, including authentication, real-time database, Firestore and messaging, handle real-time data synchronization, user authentication and notification. Osmdroid library supports map visualization of bus tracking, while Glide handles efficient image loading of user profiles. In addition, Stripe integration helps secure in-app payment, while Fuel library simplifies network communication and JSON processing.

Visual Studio 2022

Admin Website Setup

Program.cs

```
using FirebaseAdmin;
using FirebaseAdmin.Auth;
using Google.Apis.Auth.OAuth2;
using Google.Cloud.Firestore;
using Microsoft.AspNetCore.Authentication.Cookies;
using UTAR_Bus.Firebase;

var builder = WebApplication.CreateBuilder(args);

// Initialize Firebase
FirebaseApp.Create(new AppOptions()
{
    Credential = GoogleCredential.FromFile(builder.Configuration["Firebase:ServiceAccountKeyPath"]),
    ProjectId = builder.Configuration["Firebase:ProjectId"]
});

// Initialize Firestore with the Project ID
FirestoreDb firestoreDb = FirestoreDb.Create(builder.Configuration["Firebase:ProjectId"]);

// Add Firebase services to DI
builder.Services.AddSingleton(FirebaseAuth.DefaultInstance);
builder.Services.AddSingleton(firestoreDb);
builder.Services.AddScoped<FirebaseUserManager>();
builder.Services.AddScoped<FirebaseSignInManager>();
builder.Services.AddScoped<FirebaseRoleManager>();
builder.Services.AddRazorPages();
builder.Services.AddHttpClient();
builder.Services.AddSingleton<FirebaseService>();
builder.Services.AddScoped<FirebaseNotificationService>();


// Configure Cookie Authentication
builder.Services.AddAuthentication(CookieAuthenticationDefaults.AuthenticationScheme)
    .AddCookie(options =>
    {
        options.LoginPath = "/Account/Login";
        options.AccessDeniedPath = "/Account/AccessDenied";
    });

builder.Services.AddRazorPages();
builder.Services.AddDistributedMemoryCache();
builder.Services.AddSession(options =>
{
    options.IdleTimeout = TimeSpan.FromMinutes(30);
    options.Cookie.HttpOnly = true;
    options.Cookie.IsEssential = true;
});

var app = builder.Build();
```

```
// Ensure the admin role and user exist
using (var scope = app.Services.CreateScope())
{
    var userManager = scope.ServiceProvider.GetRequiredService<FirebaseUserManager>();
    var roleManager = scope.ServiceProvider.GetRequiredService<FirebaseRoleManager>();

    // Create the Admin role if it doesn't exist
    string adminRole = "Admin";
    if (!await roleManager.RoleExistsAsync(adminRole))
    {
        await roleManager.CreateRoleAsync(adminRole);
    }

    // Add Admin user
    string adminEmail = "admin@gmail.com";
    string adminPassword = "Abcd1234!";

    var adminUser = await userManager.FindByEmailAsync(adminEmail);
    if (adminUser == null)
    {
        var userRecordArgs = new UserRecordArgs
        {
            Email = adminEmail,
            EmailVerified = true,
            Password = adminPassword,
            DisplayName = "Admin",
            Disabled = false,
        };

        adminUser = await userManager.CreateUserAsync(userRecordArgs);
    }

    // Assign the Admin role to the user
    if (!await userManager.IsInRoleAsync(adminUser, adminRole))
    {
        await userManager.AddToRoleAsync(adminUser, adminRole);
    }
}

app.UseHttpsRedirection();
app.UseStaticFiles();

app.UseRouting();

app.UseSession();

app.UseAuthentication();
app.UseAuthorization();

app.MapRazorPages();

app.Run();
```

Figure 5.3.6 Firebase Connection and Role Assign for Program.cs

The Program.cs as an ASP.NET Core web application integrated with Firebase services for a bus tracking system. It initializes Firebase with a service account for authentication and Firestore for database management. Firebase services like

FirebaseAuth, FirestoreDb, and custom managers (FirebaseUserManager, FirebaseSignInManager, FirebaseRoleManager) are added to the dependency injection (DI) container, along with support for HTTP clients and notifications.

It also configures cookie-based authentication for user sessions, allowing login and access control using cookies. The session management is configured with a 30-minute timeout.

Upon app startup, the code ensures that the "Admin" role exists and creates an admin user if one does not exist, assigning the "Admin" role to them. Finally, the app sets up essential middleware like HTTPS redirection, static files, routing, session, and authentication/authorization, then maps Razor Pages and runs the application.

Appsettings.json

Figure 5.3.7 API, ClientId, ClientSecret Setup in appsettings.json

This JSON configuration file is for an ASP.NET Core application integrated with Firebase and Google Authentication. It includes logging settings to control the application's log level, with "Debug" as the default and specific levels for Microsoft components. The AllowedHosts setting allows all hostnames. The Firebase section contains the service account key file path (ServiceAccountKeyPath), the project ID (ProjectId), and the Firebase API key (ApiKey) for connecting to Firebase services. Additionally, the Authentication section includes Google's OAuth credentials, with the ClientId and ClientSecret enabling Google login functionality within the app.

5.4 System Operation

Map Page

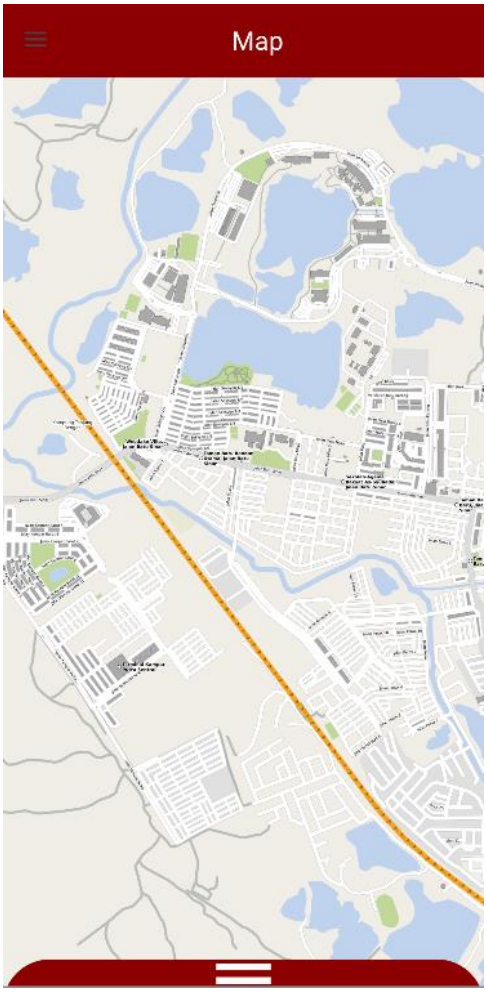


Figure 5.4.1 Map Page of the System Operation

The map page conducts three parts inside a page which is toolbar, map, and slide up menu (route menu). Inside the toolbar, the menu icon is placed at the left conner. After the user grants permissions to Android Bus Tracker, the map will be displayed between the toolbar and the slide-up menu (Route page), in the same wireframe layout as suggested in Chapter 3. The map does not contain any bus stops or bus icons.

Menu Page

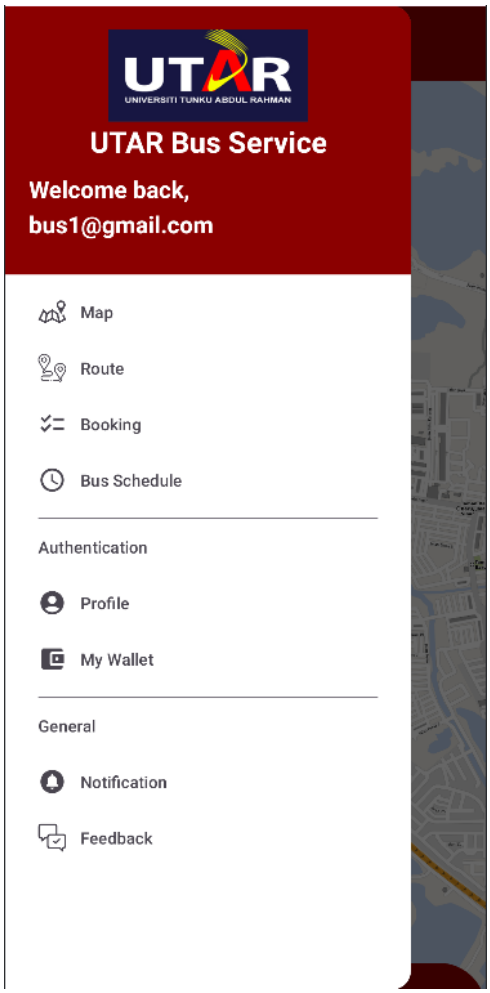


Figure 5.4.2 Menu Page of the System Operation

The menu page is displayed when the user triggers the menu icon on the toolbar. The menu page consists of two parts: the header and the body. In the header section, the UTAR logo and title are displayed, and below that is the information. This information changes when the user logs into their account and the guest word will changes to the email the user had when they logged in. This menu page offers five different options for the user to choose from: Map, Route, Booking, Bus Schedule, Profile, My Wallet, Notification, and Feedback. Each option can be linked directly to the corresponding page, allowing the user to access the desired functionality.

Slide Up Route & Time Fragment Page

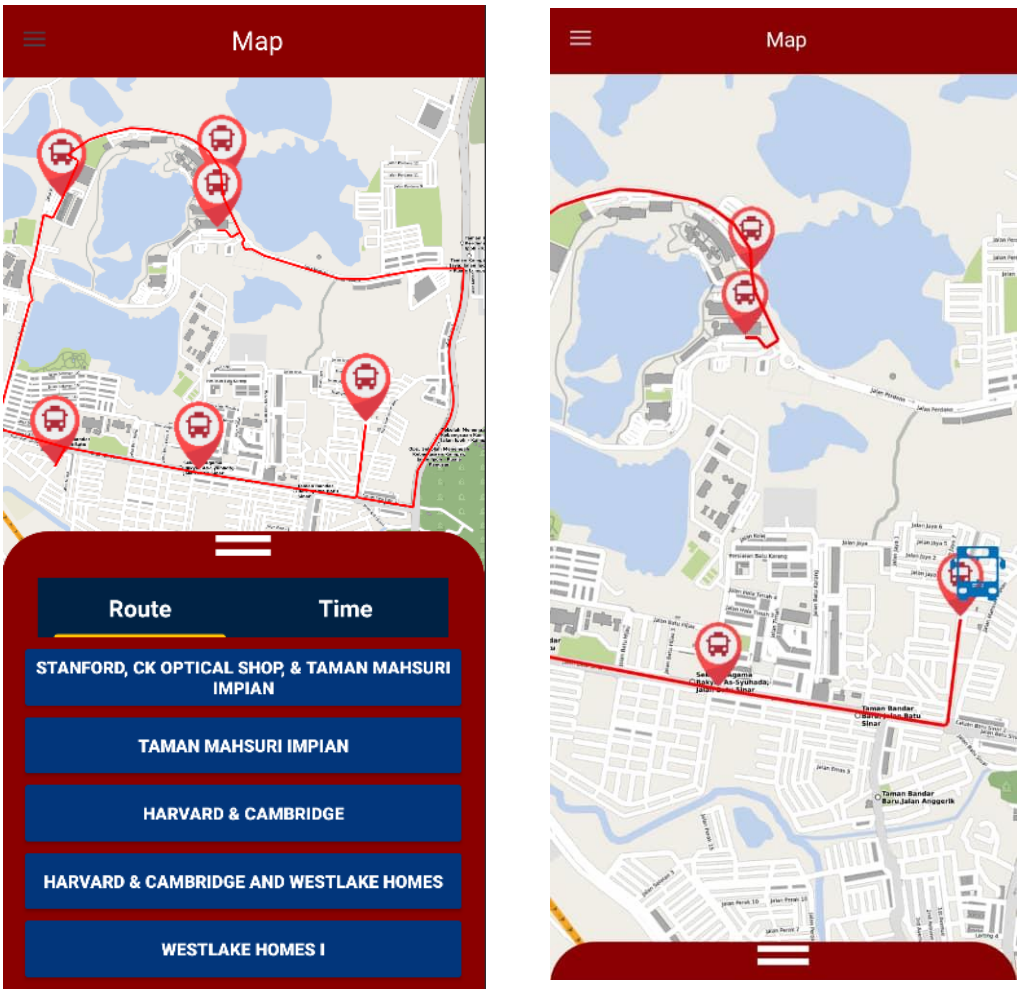


Figure 5.4.3 Route Page of the System Operation

Users can open the route and time fragment page by swiping up or clicking on the route option in the menu. The routes page provides seven bus routes that the user can select according to his/her plan. After selecting one of the bus routes, the bus stops, bus icons, routes selected by the user will appear on the map. Besides, user can track the location of the bus from the map.

Slide Up Time Estimation Page

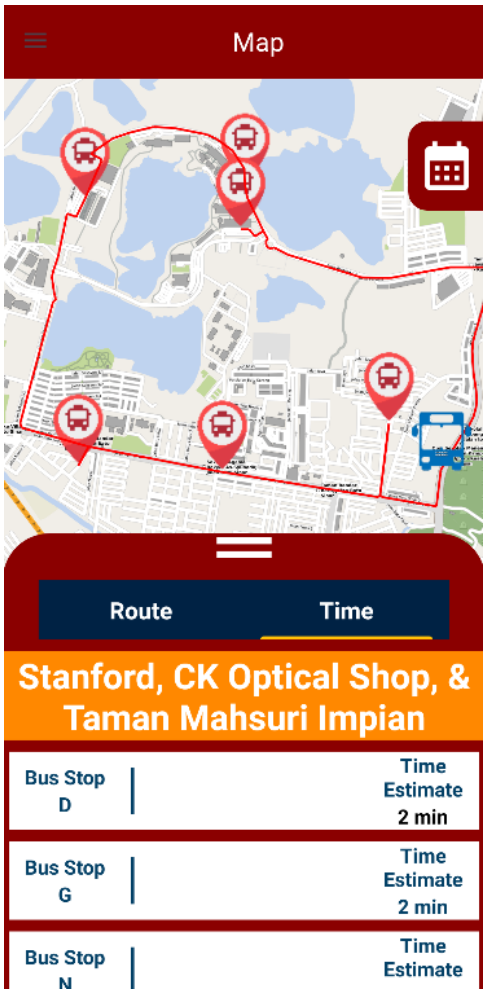


Figure 5.4.4 Time Estimation Page of the System Operation

Once the user clicks on the bus route option, the time fragment page will switch from the route fragment page and fetch specific data from the firebase database. The time estimation function will start calculating the distance between each bus stop coordinate and the bus coordinate. Besides, the calendar icon on the right allows the user to view the real-time bus schedule.

Realtime Bus Schedule Page

Figure 5.4.5 Realtime Bus Schedule Page of the System Operation

This Realtime Bus Schedule will display specific time and locations retrieved from the firebase database according to the route selected by the user. The next trip time will show the most recent time, for example if it is 7am then the next trip time will show 7.15am.

Booking Page

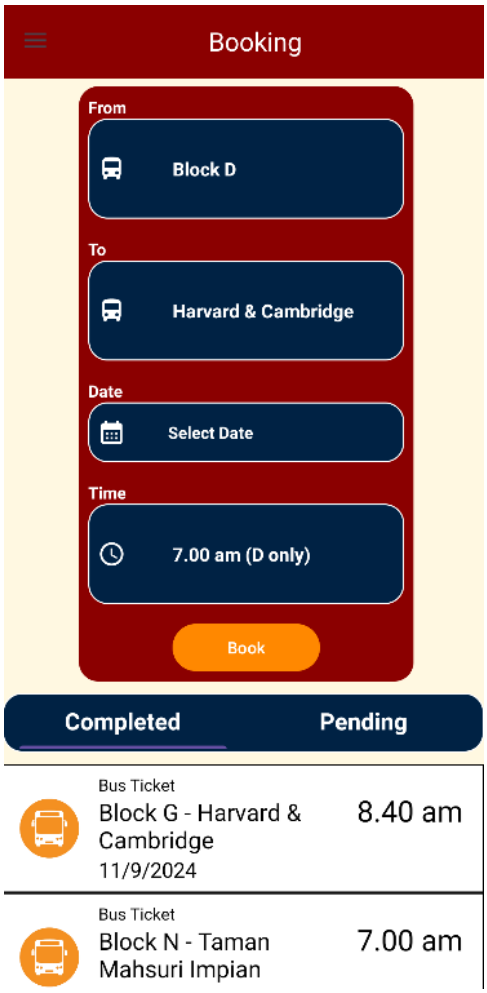


Figure 5.4.6 Booking Page of the System Operation

This Booking page allows users to select their departure point, destination, travel date, and time. The "From" field lets users choose their starting location, while the "To" field is used to select their destination. A date picker is provided for the user to select the desired travel date, and the available bus times are displayed in the "Time" section. Once all fields are filled in, users can click the "Book" button to confirm and reserve their bus ticket. Below the booking form, there are two tabs: "Completed" and "Pending." These tabs organize the user's bookings by their status. The "Completed" tab shows successfully booked tickets with details such as the departure point, destination, date, and time of travel. The "Pending" tab contains bookings that are not yet confirmed.

Booking Details Page

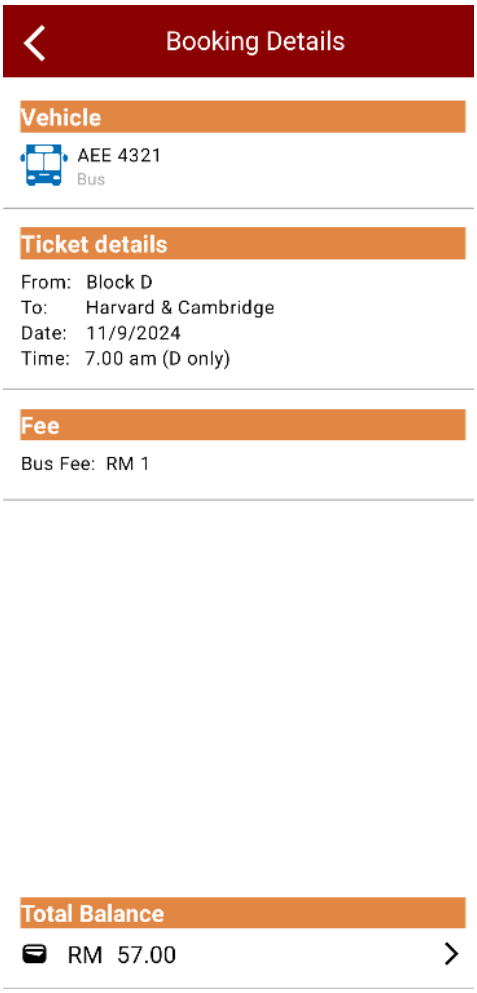


Figure 5.4.7 Booking Details Page of the System Operation

The Booking Details page allows the user to fully view and confirm all relevant information before finalizing the booking. This page clearly displays ticket details such as origin, destination, dates and travel time. This allows the user to verify that the selected travel details are accurate. The vehicle will display a license plate number retrieved from the firebase database based on the time selected by the user. After confirming that all the details are correct, the user can click on the forwarding icon to make payment (Fee: MYR1) or if the balance is insufficient, the user will be redirected to the top-up page.

Ticket Details Page

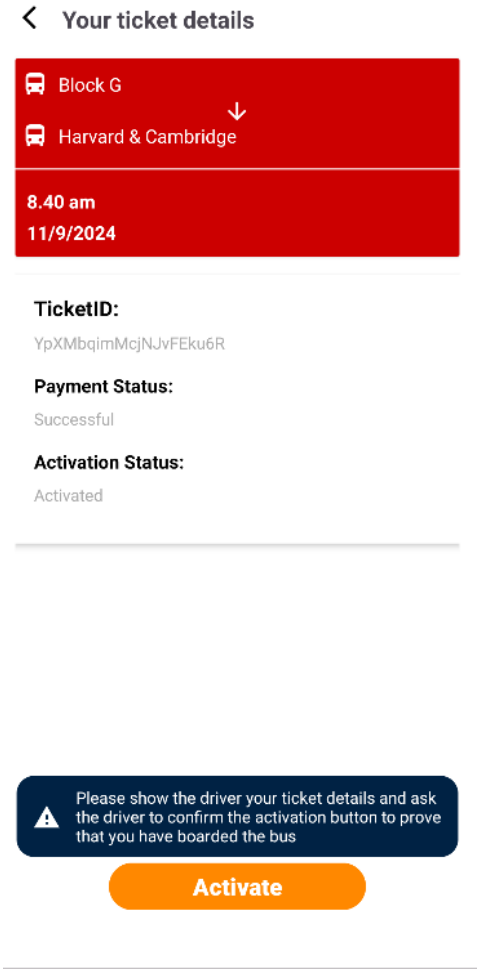


Figure 5.4.8 Ticket Details Page of the System Operation

This ticket details page provides essential information regarding the bus journey. It shows the starting point, and destination with the scheduled departure time and date. Below that, the ticket ID is displayed along with the payment status, which confirms a successful transaction. The activation status indicates that the ticket activated or not. At the bottom, there is an instruction to show this page to the bus driver, and after confirmation, the user need to press the orange "Activate" button to verify the boarding.

Bus Schedule Page

Figure 5.4.9 Bus Schedule Page of the System Operation

The Bus Schedule page provides the full bus timetables for all locations, which users can download or read within the app.

Profile & Edit Profile Page

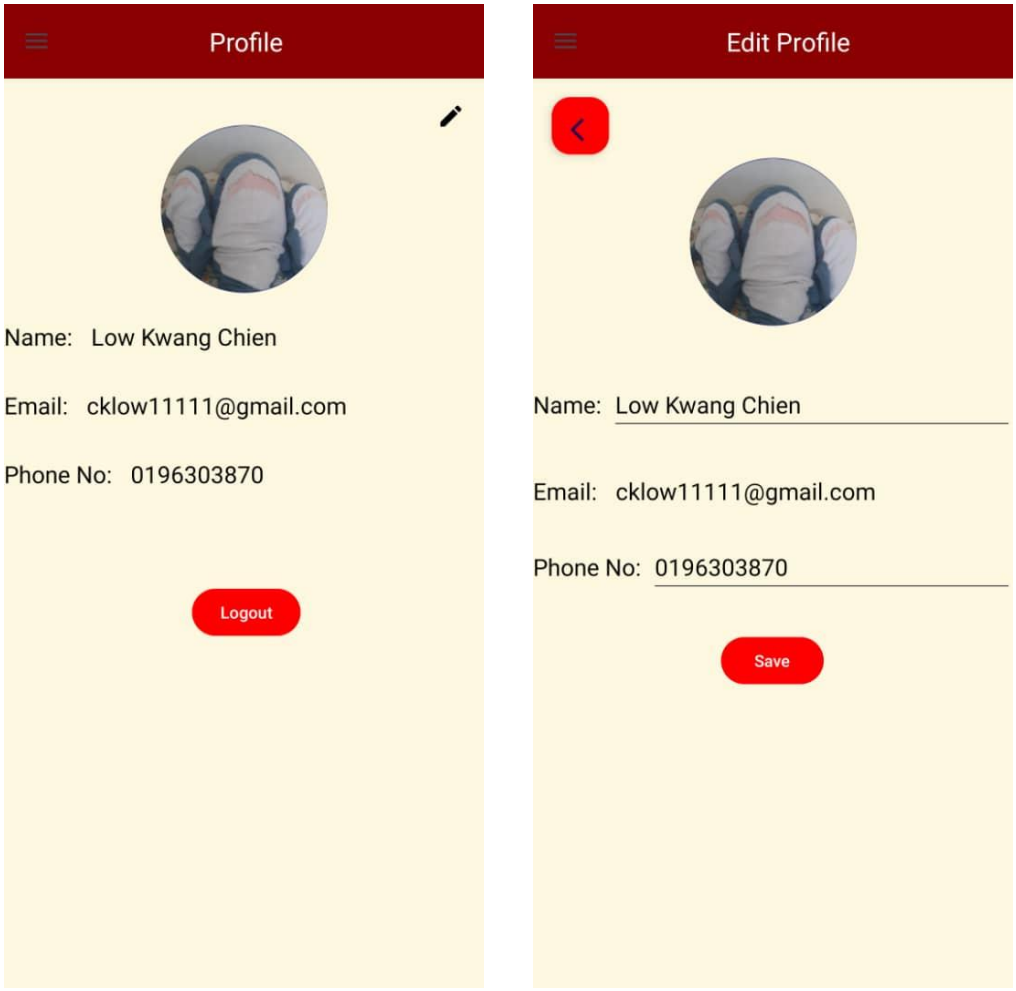


Figure 5.4.10 Profile & Edit Profile Page of the System Operation

The Profile and Edit Profile page provides the features that users can view and modify their name, email, phone number and upload their profile picture.

Login Page

Figure 5.4.11 Login Page of the System Operation

Currently login page is designed for tracking bus location by using the android smart phone, with each driver possessing their own dedicated account corresponding to their assigned bus route.

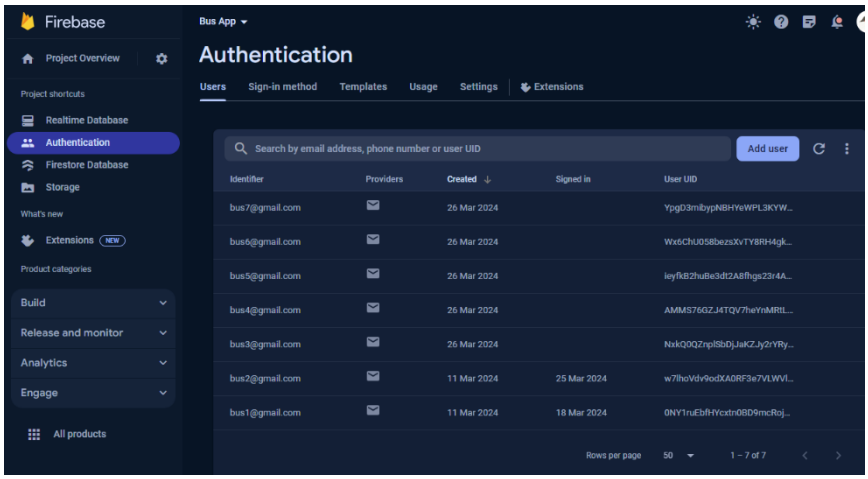


Figure 5.4.12 Dedicated Account for Bus Driver

This setup is managed through the Firebase authentication database, where each account serves as a unique identifier for a particular route.

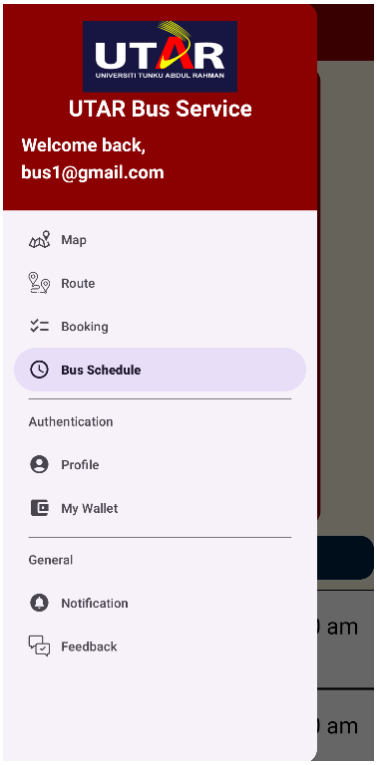


Figure 5.4.13 After Login Dedicated Account

By logging in, the system continuous retrieval of device coordinates for real-time bus location updates, which are then transmitted to Firebase Realtime Database.

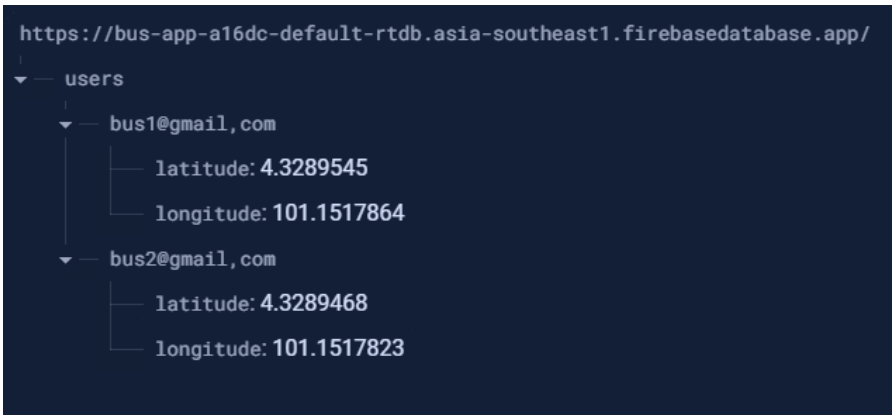


Figure 5.4.14 Coordinates for Dedicated Account (Bus)

This ensures that users' devices receive constant updates on the bus's whereabouts, allowing them to track its movement in real-time. As the bus moves along its route, its icon dynamically changes position on the user interface, providing a visual representation of its journey.

My Wallet Page

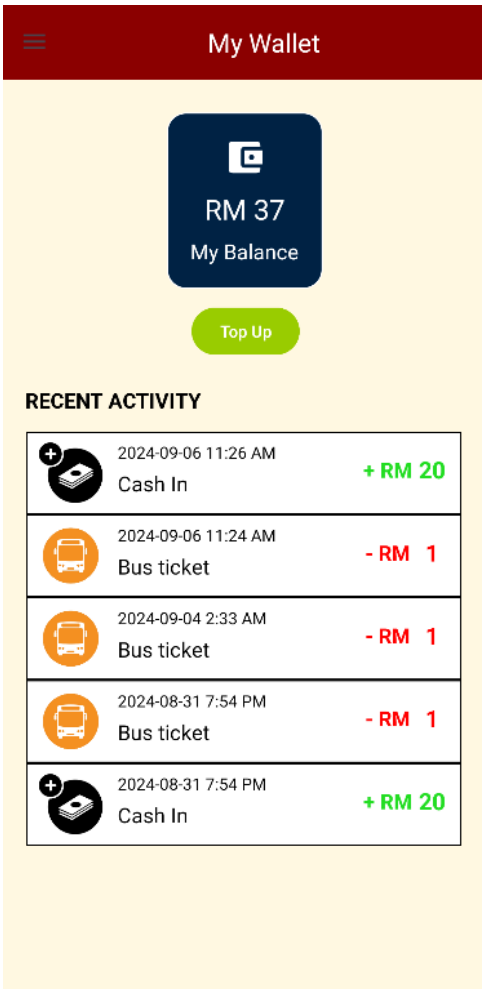


Figure 5.4.15 My Wallet Page of the System Operation

The Wallet page allows the user to manage their funds and track recent transactions with ease. At the top, it clearly displays the current wallet balance, with an option to top up if needed. Recent activities such as cash-ins and bus ticket purchases are listed below, showing the date, time, and amount for each transaction. This helps users verify their spending history and wallet balance. When purchasing a bus ticket (Fee: MYR 1), the amount will be automatically deducted from the balance. If the balance is insufficient, the user will be redirected to the top-up page to add more funds.

Top Up Page

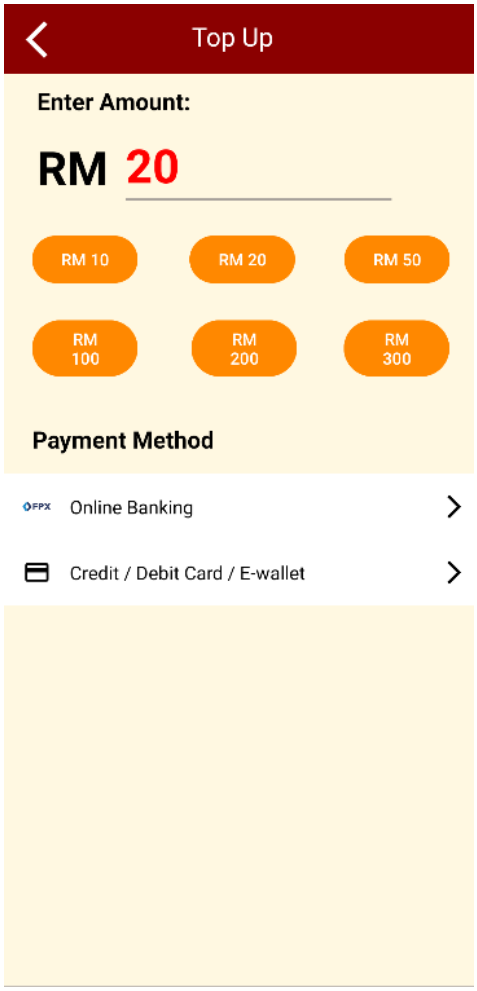


Figure 5.4.16 Top Up Page of the System Operation

The Top Up page allows the user to top up the wallet by selecting the desired amount and payment method. Users can either enter a customized amount or choose from predefined options such as RM10, RM20, RM50 or higher. After selecting the amount, they can choose from payment options such as online banking or credit/debit card and e-wallet services. The page provides a seamless way to ensure that there is sufficient balance for future transactions such as buying bus tickets. Once the payment is completed, the selected amount is deposited in the wallet.

Online Banking Page

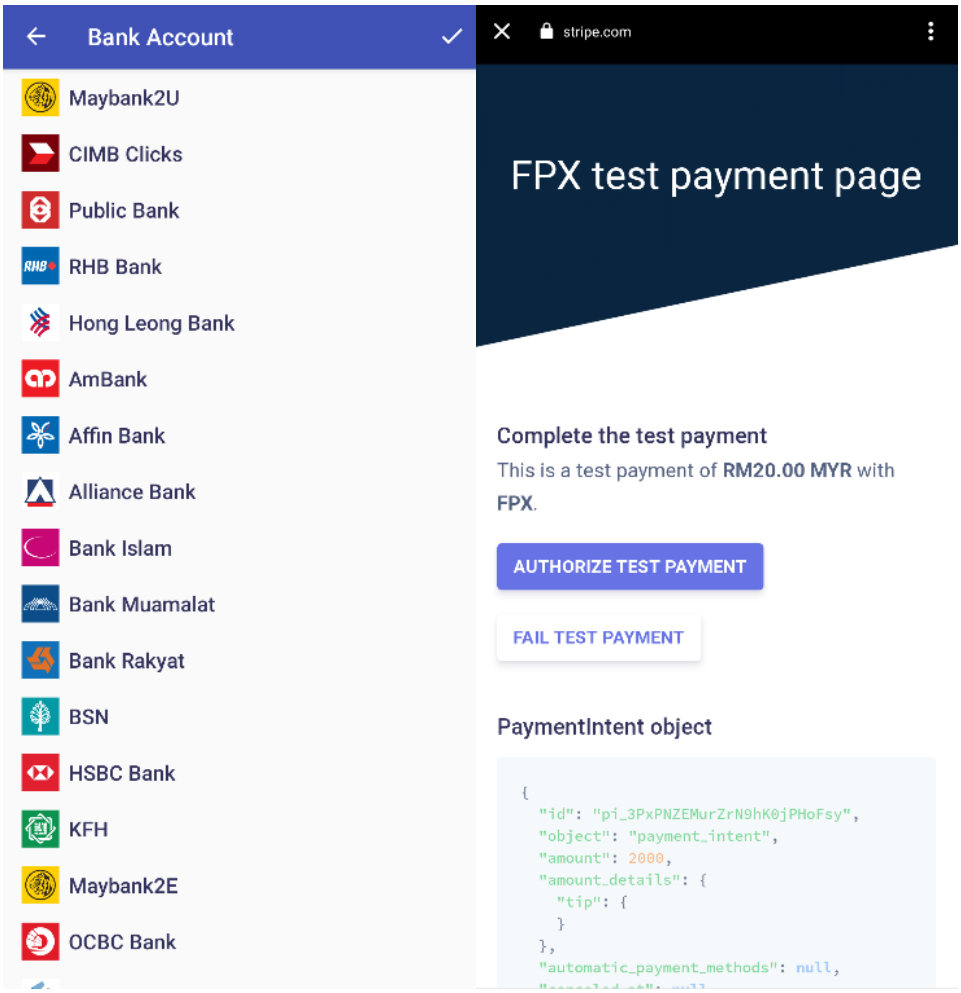


Figure 5.4.17 Online Banking Page of the System Operation

When user selects the online banking method, it redirects to the online banking page, which provides several banks in Malaysia for the user to choose from. As this is a test payment, Stripe provides a method to authorize text payments in the browser.

Credit / Debit Card / E-wallet Page

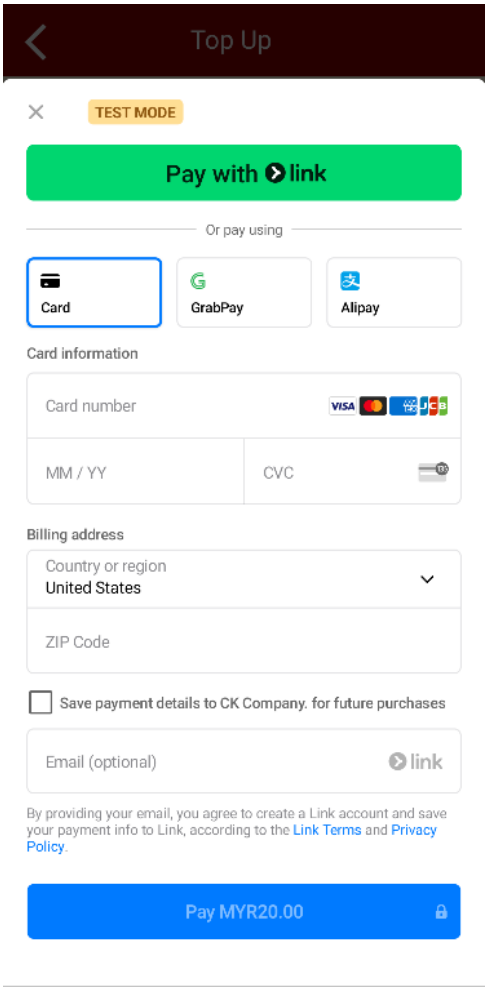


Figure 5.4.18 Credit / Debit Card / E-wallet Page of the System Operation

When user selects the Credit / Debit Card / E-wallet, it will slide up the Credit / Debit Card / E-wallet page, which provides card, GrabPay, AliPay, and Link for the user to choose from. As this is a test payment, Stripe will auto authorize the payment that the user made.

Notification Page

Figure 5.4.19 Notification Page of the System Operation

Users can receive or look back at notifications sent by administrators and check if there are any important content in the notifications page.

Feedback Page

Figure 5.4.20 Feedback Page of the System Operation

Users can submit feedback about application problems or any errors they face. All feedback will be sent and viewed by the administrator of the website page.

Admin Website Login

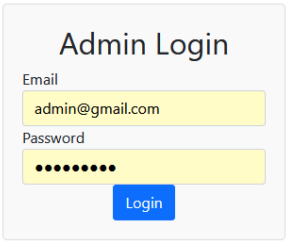


Figure 5.4.21 Admin Website Login of the System Operation

Admin need to login their dedicated account for access all the features in the webpage.

Admin Website Dashboard

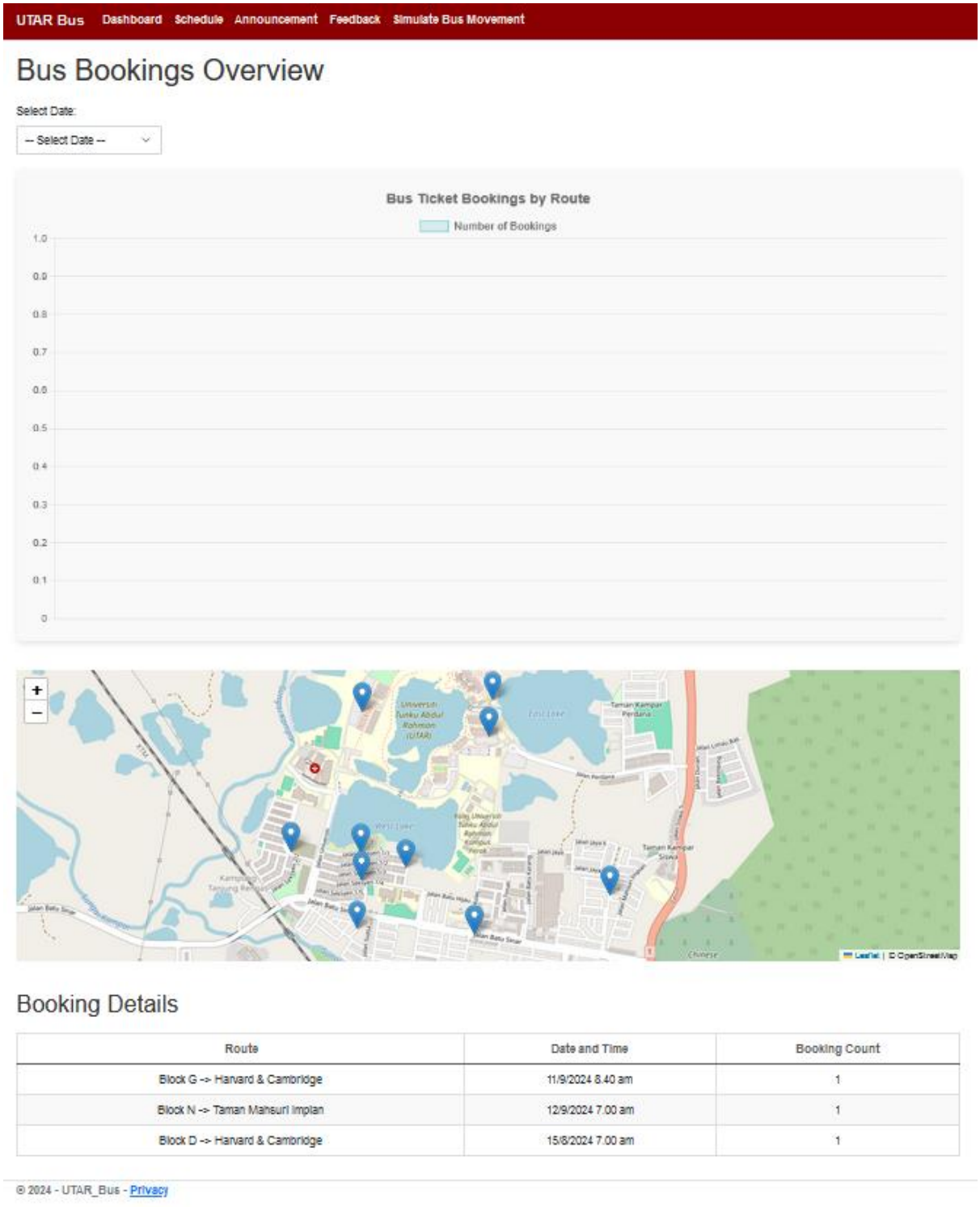


Figure 5.4.22 Admin Website Dashboard of the System Operation

This admin dashboard provides functions for managing and monitoring a bus booking system. It includes a date filter to view bookings by specific dates, a bar chart to display bus

ticket bookings by route, and an interactive map that can be enhanced with a heat map feature to show user concentration at bus stops. The booking details section lists routes, times, and the number of bookings, helping admins efficiently manage daily transportation operations and optimize routes based on user demand.

Admin Website Schedule

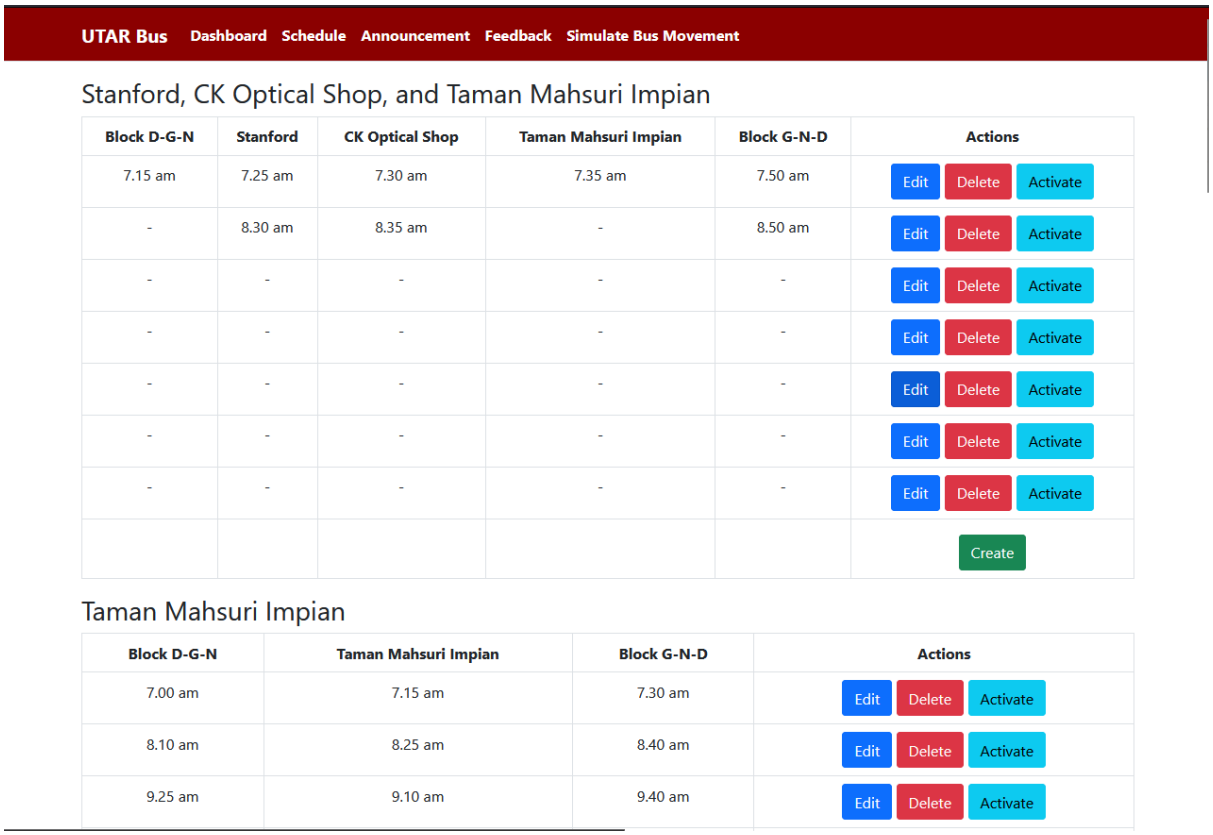


Figure 5.4.23 Admin Website Schedule of the System Operation

Admin can perform CRUD operations to manage each bus trip in each route, including changing the time of bus trips or activate or deactivate a trip from the route,

Admin Website Announcement

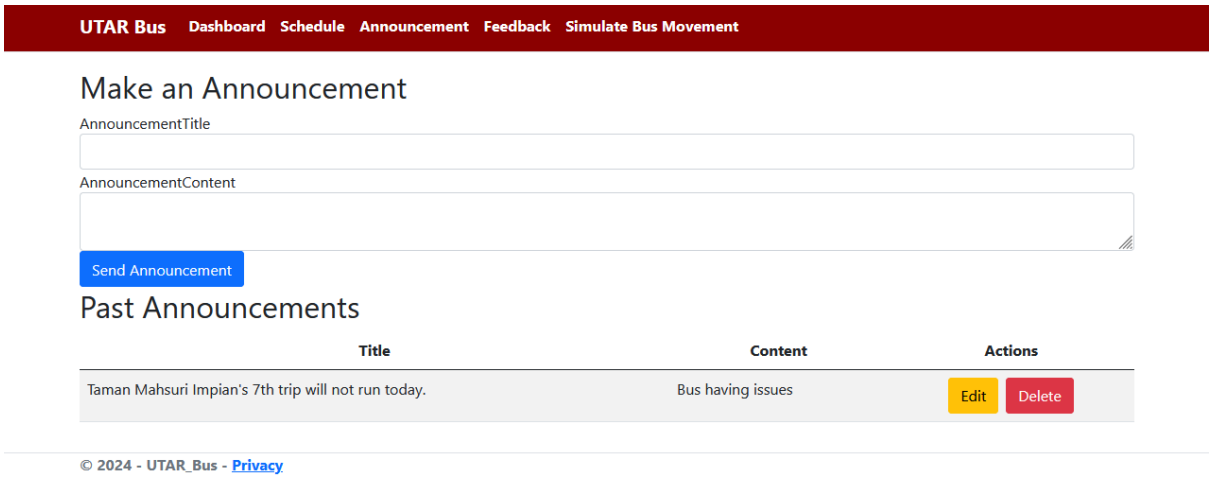


Figure 5.4.24 Admin Website Announcement of the System Operation

Admin can make announcement and view all the past announcements with the operations to edit or delete the past announcements. This can prevent the admin make a wrong announcement.

Admin Website Feedback

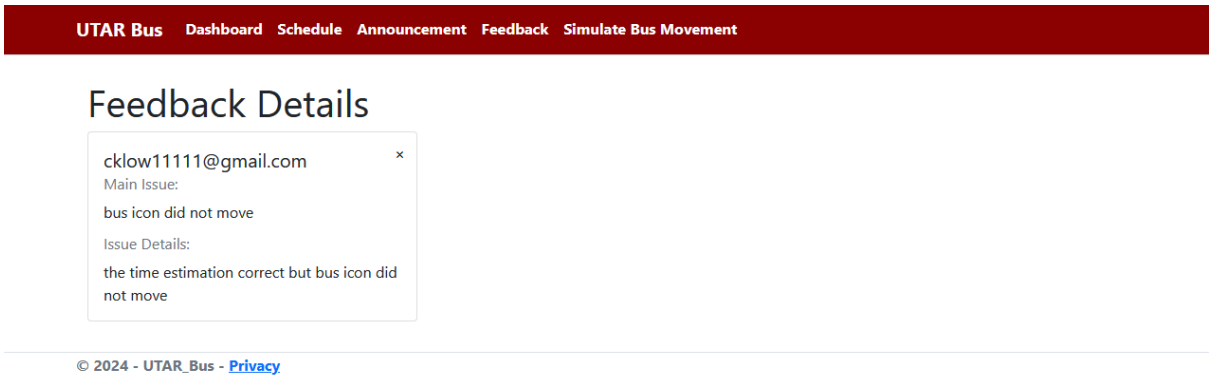


Figure 5.4.25 Admin Website Feedback of the System Operation

Admin can view all the feedback submitted by the users. The feedback can be deleted by the admin.

5.5 Backend Server Deploy in Firebase Function

Stripe Payment

```
const functions = require('firebase-functions');
const stripe = require('stripe')(functions.config().stripe.secret_key);
const logger = functions.logger;

exports.payment = functions.https.onRequest(async (req, res) => {
  try {
    // Ensure amount is an integer
    const amount = parseInt(req.query.amt);
    if (isNaN(amount) || amount <= 0) {
      return res.status(400).send('Invalid amount');
    }

    // Create a customer
    const customer = await stripe.customers.create();

    // Create an ephemeral key
    const ephemeralKey = await stripe.ephemeralKeys.create(
      { customer: customer.id },
      { apiVersion: '2023-10-16' }
    );

    // Create a payment intent
    const paymentIntent = await stripe.paymentIntents.create({
      amount: amount * 100,
      currency: 'myr',
      customer: customer.id,
      automatic_payment_methods: { enabled: true },
    });


    // Send response
    res.set('Content-Type', 'application/json');
    res.json({
      paymentIntentClientSecret: paymentIntent.client_secret,
      ephemeralKey: ephemeralKey.secret,
      customer: customer.id,
      publishableKey: functions.config().stripe.publishable_key
    });
  } catch (error) {
    logger.error('Error creating payment intent:', error);
    res.status(500).send('Internal Server Error');
  }
});
```

Figure 5.5.1 Stripe Payment Code

The code above outlines a backend server function deployed on Firebase Cloud Functions, designed to facilitate payment processing through Stripe. This function is triggered by an HTTP request and validates the payment amount received via a query parameter. Upon receiving a valid amount, it creates a new customer in Stripe and generates an ephemeral key for secure interactions. The function then creates a payment intent for the specified amount in Malaysian Ringgit (MYR), associating it with the newly created customer and enabling automatic payment methods. The response includes the payment intent's client secret, the ephemeral key, the customer ID, and the publishable key from the Firebase configuration, allowing the frontend to complete the payment process securely.

Stripe FPX Payment

```
const functions = require('firebase-functions');
const stripe = require('stripe')(functions.config().stripe.secret_key);
const logger = functions.logger;

exports.Fpxpayment = functions.https.onRequest(async (req, res) => {
  try {
    if (req.method !== 'POST') {
      return res.status(405).send('Method Not Allowed');
    }

    const amount = parseInt(req.query.amt);
    if (!amount || isNaN(amount) || amount <= 0) {
      return res.status(400).send('Invalid amount');
    }

    const customer = await stripe.customers.create();

    const ephemeralKey = await stripe.ephemeralKeys.create(
      { customer: customer.id },
      { apiVersion: '2024-06-20' }
    );

    const paymentIntent = await stripe.paymentIntents.create({
      amount: Math.round(amount * 100),
      currency: 'myr',
      payment_method_types: ['fpx'],
      metadata: {
        integration_check: 'accept_a_payment'
      }
    });

    res.status(200).json({
      paymentIntentClientSecret: paymentIntent.client_secret,
      paymentIntentId: paymentIntent.id,
      ephemeralKey: ephemeralKey.secret,
      customer: customer.id,
      publishableKey: functions.config().stripe.publishable_key
    });
  } catch (error) {
    logger.error('Error creating FPX payment intent:', error);
    res.status(500).send('Internal Server Error');
  }
});
```

Figure 5.5.2 Stripe FPX Payment Code

The code above defines a backend server function deployed on Firebase Cloud Functions to handle FPX payments via Stripe. It is triggered through an HTTP request and processes payment amounts using Stripe's API. Upon receiving a valid POST request with an amount parameter, the function creates a Stripe customer and an ephemeral key for the customer. It then initiates a payment intent for the specified amount in Malaysian Ringgit (MYR) using FPX (an online banking payment method). The response includes the payment intent's client secret, ephemeral key, customer ID, and the publishable key from Firebase configuration, which is used by the frontend to complete the payment.

5.6 Implementation Issues and Challenges

Developing an Android Bus Tracker System presents several implementation challenges, primarily cantered around navigating the complexities of integrating various technologies. One major difficulty lies in managing the Android Studio environment, which serves as the primary platform for Android application development. This involves configuring the development environment, understanding its features, and effectively utilizing its tools for coding, debugging, and testing. Additionally, incorporating the Osmdroid library, a popular open-source mapping library for Android, poses its own set of challenges. Integration with this library requires a deep understanding of its APIs and functionalities to effectively utilize mapping and location-based features within the application. Furthermore, leveraging Firebase services adds another layer of complexity. Setting up Firebase involves configuring authentication, managing real-time databases, and implementing cloud messaging for seamless communication between the application and server.

Besides, the GPS distance and time estimate features also a challenge for provide an accurate data to the users. The implementation of GPS tracker devices with SIM cards within the system has been proposed as a solution for obtaining real-time bus location updates. These GPS trackers operate by transmitting coordinates via signal transmission, enabling the system to ascertain the precise location of buses in transit. However, ensuring the reliability and accuracy of these coordinates poses a notable challenge, particularly in scenarios where factors such as signal interference or device malfunction may impact data integrity. As a result, Android smartphones are used as GPS trackers to provide bus location information on a map. On the other hand, the key challenges lie in accurately estimating the time it takes for buses to reach their respective stops. This involves intricate calculations of the distance between each bus stop and the current location of the bus. This seemingly simple task becomes complex due to factors such as traffic conditions, route variations, and unexpected delays. Additionally, the need for real-time updates further intensifies the challenge, as the accuracy of time estimates heavily relies on constantly changing variables.

CHAPTER 6

System Evaluation and Discussion

6.1 System Testing and Performance Metrics

System testing of the Android bus tracker system focused on evaluating key performance metrics to ensure that the application achieves the desired functionality and performance. Key metrics tested included bus location accuracy, response time, bus arrival time estimates, application reliability, and concurrency management. These metrics are critical to the system's ability to provide real-time tracking and enhance the overall commuter experience for students.

Bus location accuracy was a top priority, as the application needed to accurately display the real-time location of the buses. The GPS tracking system was tested in a variety of conditions around the UTAR campus, and the results showed an average accuracy of within a 5-metre radius. This accuracy allowed students to effectively track the location of buses, helping them to plan their trips with confidence. However, in areas with weak GPS signals, such as areas with buildings or trees, slight deviations can occur.

The responsiveness of the application, especially in terms of providing real-time updates, was another key factor. The app performs best in good network conditions, providing bus location updates with a delay of less than 3 seconds. In areas with poor network coverage, delays of up to 5 - 10 seconds occurred, but overall, the system was still responsive enough to meet users' expectations. The application also performed well during peak hours when multiple users accessed the system at the same time.

To ensure an accurate estimate of bus arrival times, the system utilizes GPS data and historical traffic patterns. During testing, the application consistently provided reliable estimates with a margin of error of approximately 1-2 minutes under normal traffic conditions. This feature greatly reduces the uncertainty students experience when waiting at bus stops, as they can better predict bus arrival times.

The reliability of the application was tested under various conditions, including network outages, extended usage, and periods of high user traffic. Throughout the testing period, the system demonstrated stability with no major crashes or disruptions. Even during peak usage

times, the application maintained full functionality, delivering notifications and real-time updates as expected. This level of stability is essential to provide users with a dependable service, ensuring smooth operation and timely updates throughout the day.

Finally, concurrency was tested by simulating multiple users simultaneously tracking buses and receiving updates. The system handled the high user load efficiently, with no noticeable slowdowns or delays in delivering real-time data. This demonstrates the system's ability to scale and operate reliably even during periods of high usage (e.g., peak commuting hours).

6.2 Testing Setup and Result

Test Case 1: Bus Location Accuracy

No	Test Objective	Test Step	Expected result	Result
1.	Ensure that the Bus GPS Coordinates and Speed are store inside the Firebase Realtime Database.	1) Turn on GPS feature inside the device. 2) Login with the bus driver dedicated account. 3) Observe that the coordinates and speed data stored in the Firebase real-time database are continuously updated.	The bus icon should be display on the screen in the map page.	Pass
2.	Ensure that bus movements are consistent with reality.	1) Driving a car on a simulated bus route. 2) Login with the bus driver dedicated account. 3) Observe the movement of the	The bus icon should move consistently across the screen and follow the route the car is travelling.	Pass

		bus icon on the screen.		

Table 6.2.1 Bus Location Accuracy

Test Case 2: Responsiveness

No	Test Objective	Test Step	Expected result	Result
1.	Ensure that the Android Bus Tracker System responds quickly and accurately under various network conditions.	1) Launch the app and monitor the real-time location of buses under normal network conditions (5G or Wi-Fi). 2) Simulate weak network conditions (3G/4G or limited connectivity) and observe the system's behavior. 3) Test the app's performance during periods of high usage by simulating multiple users tracking buses simultaneously.	The app should provide real-time bus location updates within 3 seconds under normal network conditions. In weak network conditions, the update delay should not exceed 10 seconds. The app should handle concurrent user activity without delays or crashes.	Pass

Table 6.2.2 Responsiveness

Test Case 3: Accurate Estimate of Bus Arrival Times

No	Test Objective	Test Step	Expected result	Result
1.	Ensure that the Android Bus Tracker System provides accurate bus arrival time estimates.	1) Select a bus from the app and monitor the estimated arrival time at various stops. 2) Compare the system's estimated arrival time with the actual arrival time of the bus at each stop. 3) Repeat the test under different traffic conditions (e.g., peak hours, low traffic).	The app should provide an estimated bus arrival time within a 1–2-minute margin of error compared to the actual arrival time under normal traffic conditions. The estimate should adjust dynamically to reflect any changes in traffic or delays.	Pass

Table 6.2.3 Accurate Estimate of Bus Arrival Times

Test Case 4: Reliability

No	Test Objective	Test Step	Expected result	Result
1.	Ensure that the Android Bus Tracker System remains reliable and stable during extended use, and all features function correctly,	1) Launch the app and use it continuously for 30 min to 1 hour, performing actions such as	The app should remain stable without crashing or freezing. All features (payment,	Pass

	including payment, notifications, wallet balance, and profile login/logout.	making payments, receiving notifications, and checking wallet balance. 2) Log in and log out multiple times to verify session handling. 3) Simulate network disruptions (e.g., temporary disconnection) to check how the app recovers.	notifications, wallet balance, and login/logout) should work correctly, and the app should recover smoothly from network interruptions.	

Table 6.2.4 Reliability

Test Case 5: Concurrency

No	Test Objective	Test Step	Expected result	Result
1.	Ensure that the Android Bus Tracker System can handle multiple users accessing the app and its features simultaneously without performance degradation.	1) Simulate multiple users (e.g., 20+ users) accessing the app simultaneously to track buses, make payments, receive notifications, and	The app should remain responsive, with no significant delays in real-time tracking, payments, notifications, or other features. It should handle	Pass

		update their profiles. 2) Monitor system performance for slowdowns, crashes, or delays in real-time updates or feature responses.	high user load without crashing or degrading in performance.	

Table 6.2.5 Concurrency

6.3 Project Challenges

During the development of the Android Bus Tracker System, several challenges were encountered, each of which required careful problem solving and adjustments to ensure the success of the project. One of the main challenges was integrating a real-time GPS bus tracking system. Implementing a reliable system that provides accurate and up-to-date bus locations requires a robust backend and efficient location data processing. GPS signal instability, especially in areas with low coverage, can cause delayed or inaccurate location updates, which can also be a challenge. Addressing this issue requires optimizing the application's processing of GPS data and building a system that eliminates location discrepancies and provides users with reliable information even when signal strength varies.

Another major challenge is managing network reliability and application responsiveness in an environment where users may experience intermittent connections. Since the application relies heavily on real-time data, it is critical to ensure smooth operation despite temporary network outages. This requires the implementation of data caching mechanisms and allows the application to seamlessly recover from network outages. Testing of the application under various network conditions showed that while it performed well in areas with strong signals, poor connectivity could lead to delays in updates, so further optimization of data handling and synchronization was required.

A crucial aspect of the Android Bus Tracker System was the implementation of a notification system using Firebase Cloud Messaging (FCM). This allowed the application to send real-time notifications to users regarding bus schedules, emergency updates, and other important alerts. A challenge encountered during this integration was the management of user device tokens for FCM, which are essential for targeting notifications accurately. Each user device must register and maintain its token, ensuring that notifications are delivered promptly. Implementing this system required careful handling of token generation and renewal processes, as tokens can change or become invalid over time. Additionally, the notification feature had to be designed to handle varying levels of user engagement, ensuring that users received only relevant updates without feeling overwhelmed by excessive alerts. Overall, leveraging Firebase for notifications not only enhanced user experience by keeping students informed but also streamlined communication between the system and its users.

Additionally, integrating the payment system using the Stripe API presented technical and security challenges. Ensuring secure and seamless transactions, especially with high traffic, requires a strong focus on data security protocols, encryption and user authentication. Ensuring that payment transactions are fast and error-free while handling real-time traffic and capacity management data also requires balancing system resources and optimizing the application architecture to accommodate concurrent processes.

Finally, developing user interfaces that are both intuitive and feature-rich presents challenges in balancing design complexity and user-friendliness. Maintaining ease of use while incorporating a variety of features such as real-time tracking, payment options, notifications, and profile management required iterative testing and user feedback to ensure that the application met users' expectations without overwhelming them.

6.4 Objectives Evaluation

The objectives of the Android Bus Tracker System were evaluated to assess how effectively the system achieves its intended goals. Each objective was analyzed in terms of the problems faced by users, system performance metrics and overall functionality. The table below summarizes the evaluation of each objective, including test results and insights gained during implementation.

Objective	Evaluation Criteria	Result	Insights and Improvements
Provide real-time information on bus arrival and departure times.	Accuracy of bus location tracking and timeliness of arrival updates.	Met	The GPS tracking system achieved an average accuracy within 5 meters, providing reliable real-time updates.
Optimize capacity management of buses during peak hours.	User ability to check bus capacity before arrival.	Met	Users reported being able to make informed decisions about bus wait times, reducing overcrowding during peak hours.
Enable administrators to update bus schedules and send emergency notifications.	Functionality of schedule updates and notification delivery through the app.	Met	Administrators were able to update schedules quickly, and notifications reached users in real-time, enhancing communication.

Streamline the payment process for bus fares through integrated payment functionalities.	Efficiency of payment processing and user feedback on transaction experience.	Met	The integration with Stripe API allowed for smooth transactions, with users appreciating the convenience of in-app payments.
Maintain app reliability under various conditions and user loads.	Stability during testing, including network interruptions and high user concurrency.	Met	The app performed reliably, with no major crashes during peak usage or network disruptions, ensuring consistent service.

Table 6.4.1 Objectives Evaluation

CHAPTER 7

Conclusion & Recommendation

7.1 Conclusion

In summary, the Android Bus Tracker System presented in this report offers a promising solution that can significantly benefit UTAR and its students. By implementing this system, UTAR can enhance its transportation infrastructure and provide students with a more effective and reliable means of commuting within and around the campus. The system not only addresses existing transportation challenges but also sets the stage for a more efficient and user-friendly commuting experience.

One of the primary advantages of the Android Bus Tracker System is its ability to alleviate common issues faced by students, such as uncertainty and inconvenience while waiting for buses. The real-time tracking feature empowers students to access accurate information about bus locations and estimated arrival times directly through their smartphones. This capability not only reduces waiting times but also minimizes the frustration and anxiety often associated with unpredictable transportation schedules. Students can plan their journeys with confidence, knowing they have up-to-date information at their fingertips.

Moreover, the Android Bus Tracker System fosters a sense of community and connectivity among students by facilitating smoother communication and coordination around transportation. Students can utilize the system to plan their travel routes effectively, coordinate meet-up points with peers, and stay informed about any changes or delays in bus schedules. This collaborative aspect encourages social interaction and engagement among students, enhancing their overall university experience.

7.2 Recommendations

1) User Interface Enhancements:

While the current interface is functional, gathering user feedback can identify specific areas for improvement. Simplifying navigation and providing a more intuitive layout could enhance user satisfaction. Incorporating features such as a dark mode or customizable themes can also cater to diverse user preferences, making the app more appealing.

2) Admin Website Features:

Developing a more comprehensive admin website can empower administrators with enhanced tools for managing the bus system effectively. Adding features such as an analysis report dashboard can provide insights into bus utilization patterns, student preferences, and peak usage times. This data can inform decision-making regarding route optimization and capacity planning.

3) Detailed User Profiles:

Enhancing user profiles to include more detailed information can facilitate better personalization of services. Features such as saved routes, favorite bus lines, and user preferences for notifications can improve user engagement and satisfaction. Additionally, allowing users to provide feedback on their commuting experiences can help the administration address any concerns promptly.

4) Integration of Additional Features:

Incorporating features like in-app surveys and community boards can encourage students to actively engage with the system. These tools can provide valuable insights into student needs and preferences, allowing for continuous improvement of the system.

REFERENCES

Douglas-Peucker algorithm | Cartography Playground. (n.d.). Cartography-Playground.gitlab.io. https://cartography-playground.gitlab.io/playgrounds/douglas-peucker-algorithm/

"DoubleMap - Live Bus Tracker." *Udshuttle.doublemap.com*, udshuttle.doublemap.com/map/. Accessed 8 Dec. 2023.

"DoubleMap Bus Tracker - Apps on Google Play." *Play.google.com*, play.google.com/store/apps/details?id=com.doublemap.iu&hl=en&gl=US. Accessed 8 Dec. 2023.

Franklin, W. (2020, December 31). *Kalman Filter Explained Simply*. The Kalman Filter. https://thekalmanfilter.com/kalman-filter-explained-simply/

"MyGOV - the Government of Malaysia's Official Portal." *Www.malaysia.gov.my*, www.malaysia.gov.my/portal/content/30745.

"MyTransport.SG - Apps on Google Play." *Play.google.com*, play.google.com/store/apps/details?id=sg.gov.lta.mytransportsg&hl=en&gl=US. Accessed 8 Dec. 2023.

"SG Bus Buddy." *Busbuddy.app*, busbuddy.app/. Accessed 8 Dec. 2023.

"SITS - Intelligent Transport - Apl Di Google Play." *Play.google.com*, play.google.com/store/apps/details?id=com.geoxspot.rider.mbi&hl=ms. Accessed 8 Dec. 2023.

Veness, C. (2019). *Calculate distance and bearing between two Latitude/Longitude points using haversine formula in JavaScript*. Movable-Type.co.uk. https://www.movable-type.co.uk/scripts/latlong.html

Verma, R., Bipin Kumar Singh, & Zahidi, F. (2024). Management of GPS Tracking Systems in Transportation. *Energy, Environment, and Sustainability*, 251–263. https://doi.org/10.1007/978-981-97-0515-3_11

Wena's TV. "BusTracker Taiwan - Google Search." *Www.youtube.com*, 24 Feb. 2021, www.youtube.com/watch?v=GkBtZpaJ4Go. Accessed 8 Dec. 2023.

Zheng, Y., Li, Q., Chen, Y., Xie, X., & Ma, W.-Y. (2008). Understanding mobility based on GPS data. *Proceedings of the 10th International Conference on Ubiquitous Computing - UbiComp '08*. https://doi.org/10.1145/1409635.1409677

Zogg, J.-M. (2002). *GPS Basics Introduction to the system Application overview*. u-blox ag. https://crlab.ece.ucr.edu/usbgps/doc/gps_basic.pdf

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Sem 3, Year 3	**Study week no.: 11**
Student Name & ID: LOW KWANG CHIEN (20ACB05483)	
Supervisor: Mr. Cheang Kah Wai	
Project Title: Android Bus Tracker System	

1. WORK DONE

- Demo the system with some of the features haven't implement.

2. WORK TO BE DONE

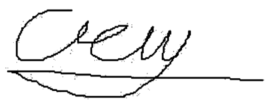
- Need to complete the features within next week.
- Report haven't complete, still doing in halfway.

3. PROBLEMS ENCOUNTERED

- Some of the features still need time to complete it.
- The project having issue when Android studio version updated

4. SELF EVALUATION OF THE PROGRESS

Put more effort.



Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Sem 3, Year 3	**Study week no.: 12**
Student Name & ID: LOW KWANG CHIEN (20ACB05483)	
Supervisor: Mr. Cheang Kah Wai	
Project Title: Android Bus Tracker System	

1. WORK DONE

- Demo full system

2. WORK TO BE DONE

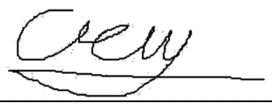
- Need to finish the report
- Some of the features can be improve and add on. Such as Refund action, Schedule improvement, UI improvement.

3. PROBLEMS ENCOUNTERED

- No idea about how to do the System Evaluation.
- Need put more effort in the system

4. SELF EVALUATION OF THE PROGRESS

More experience is still needed to handle and complete the system during project development. More functionality within the system could be enhanced.



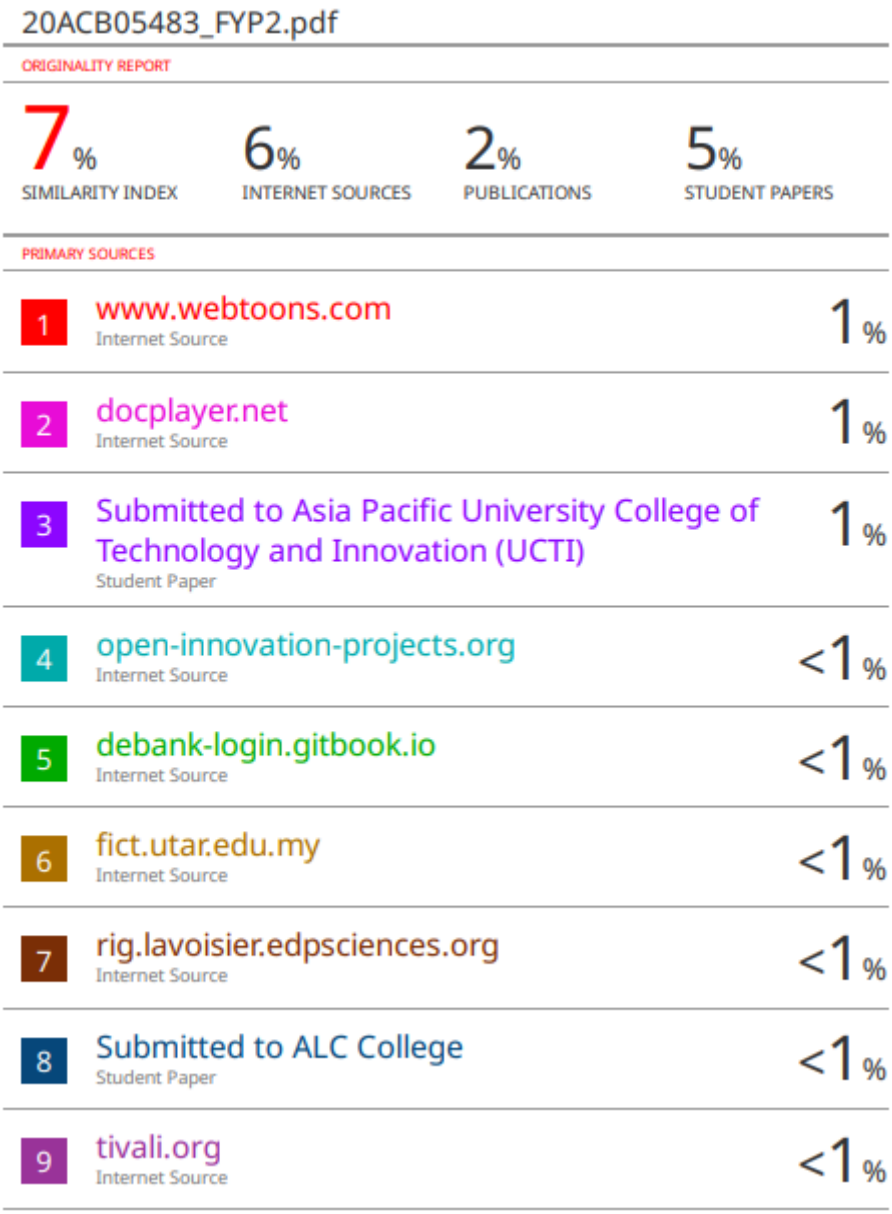
Supervisor's signature

Student's signature

POSTER


UTAR
UNIVERSITI TUNKU ABDUL RAHMAN
FACULTY OF INFORMATION
COMMNICATION AND TECHNOLOGY
Android Bus
Tracker System
1 Track the bus location
2 Check bus schedule
3 Notification Alert
4 Seat booking & made payment
FIND OUR
APP HERE
PROJECT DEVELOPER :
LOW KWANG CHIEN
PROJECT SUPERVISOR :
MR. CHEANG KAH WAI

PLAGIARISM CHECK RESULT



20ACB05483_FYP2.pdf

ORIGINALITY REPORT

7%	6%	2%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

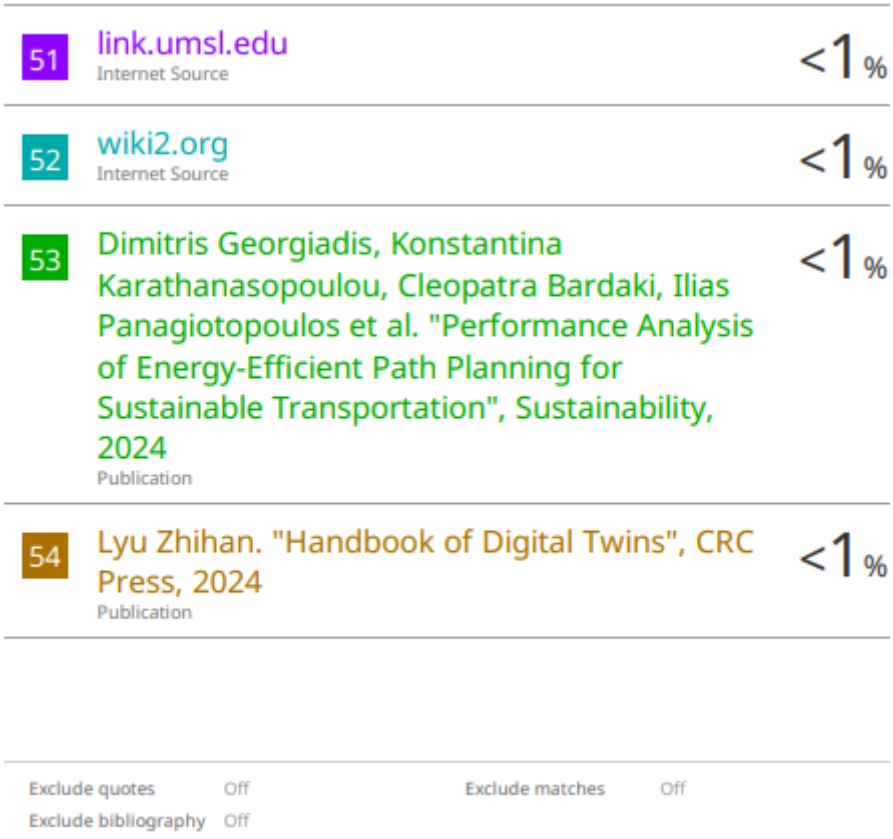
1	www.webtoons.com Internet Source	1%
2	docplayer.net Internet Source	1%
3	Submitted to Asia Pacific University College of Technology and Innovation (UCTI) Student Paper	1%
4	open-innovation-projects.org Internet Source	<1%
5	debank-login.gitbook.io Internet Source	<1%
6	fict.utar.edu.my Internet Source	<1%
7	rig.lavoisier.edpsciences.org Internet Source	<1%
8	Submitted to ALC College Student Paper	<1%
9	tivali.org Internet Source	<1%

10	Submitted to University of Leicester Student Paper	<1 %
11	Submitted to Queen's University of Belfast Student Paper	<1 %
12	Submitted to Indiana University Student Paper	<1 %
13	T. Sayed. "Transferability of travel time models and provision of real-time arrival time information", ITSC 2001 2001 IEEE Intelligent Transportation Systems Proceedings (Cat No 01TH8585) ITSC-01, 2001 Publication	<1 %
14	github.com Internet Source	<1 %
15	www.studyiq.com Internet Source	<1 %
16	Submitted to Florida Virtual School Student Paper	<1 %
17	Jingxin Wang. "Introduction to Computing Applications in Forestry and Natural Resource Management", CRC Press, 2017 Publication	<1 %
18	fkaytechnologies.odoo.com Internet Source	<1 %
19	repository.sustech.edu Internet Source	

		<1 %
20	Submitted to Anglia Ruskin University Student Paper	<1 %
21	Submitted to University of Huddersfield Student Paper	<1 %
22	Submitted to University of Sydney Student Paper	<1 %
23	edoc.unibas.ch Internet Source	<1 %
24	Submitted to Gitam University Student Paper	<1 %
25	Submitted to National School of Business Management NSBM, Sri Lanka Student Paper	<1 %
26	Submitted to Technological University Dublin Student Paper	<1 %
27	celebrating200years.noaa.gov Internet Source	<1 %
28	webcw.heroinewarrior.com Internet Source	<1 %
29	umpir.ump.edu.my Internet Source	<1 %
30	Submitted to INTI International University Student Paper	

		<1%
31	Submitted to Universiti Teknologi Malaysia Student Paper	<1%
32	Submitted to University of Bahrain Student Paper	<1%
33	Submitted to University of Bedfordshire Student Paper	<1%
34	Submitted to University of Central Florida Student Paper	<1%
35	Submitted to University of Greenwich Student Paper	<1%
36	animenite.com Internet Source	<1%
37	Submitted to American Public University System Student Paper	<1%
38	Submitted to Shinas College of Technology Student Paper	<1%
39	asis.awfatech.com Internet Source	<1%
40	hugepdf.com Internet Source	<1%
41	ijariie.com Internet Source	

		<1 %
42	www.devx.com Internet Source	<1 %
43	"Applied Soft Computing and Communication Networks", Springer Science and Business Media LLC, 2024 Publication	<1 %
44	Submitted to Georgia Institute of Technology Main Campus Student Paper	<1 %
45	dk.um.si Internet Source	<1 %
46	www.awo.agency Internet Source	<1 %
47	Submitted to Napier University Student Paper	<1 %
48	Yu Zheng. "Learning Location Correlation from GPS Trajectories", 2010 Eleventh International Conference on Mobile Data Management, 05/2010 Publication	<1 %
49	actabio.pl Internet Source	<1 %
50	izdn7.envpsych2011.eu Internet Source	<1 %


51 link.umsl.edu
Internet Source
<1%
52 wiki2.org
Internet Source
<1%
53 Dimitris Georgiadis, Konstantina Karathanasopoulou, Cleopatra Bardaki, Ilias Panagiotopoulos et al. "Performance Analysis of Energy-Efficient Path Planning for Sustainable Transportation", Sustainability, 2024
Publication
<1%
54 Lyu Zhihan. "Handbook of Digital Twins", CRC Press, 2024
Publication
<1%
Exclude quotes Off
Exclude matches Off
Exclude bibliography Off

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	**LOW KWANG CHIEN**
ID Number(s)	**20ACB05483**
Programme / Course	**IA**
Title of Final Year Project	**Android Bus Tracker System**

Similarity	**Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)**
Overall similarity index: 7 **%** **Similarity by source** Internet Sources: 7 % Publications: 6 % Student Papers: 2 %	
Number of individual sources listed of more than 3% similarity: 5	
Parameters of originality required and limits approved by UTAR are as Follows: **(i) Overall similarity index is 20% and below, and** **(ii) Matching of individual sources listed must be less than 3% each, and** **(iii) Matching texts in continuous block must not exceed 8 words** ***Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.***	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

Signature of Supervisor

Name: Cheang Kah Wai

Date: 13 Sep 2024

Signature of Co-Supervisor

Name: _____

Date: _____

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY
(KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	20ACB05483
Student Name	LOW KWANG CHIEN
Supervisor Name	Mr. Cheang Kah Wai

TICK (√)	DOCUMENT ITEMS Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
√	Title Page
√	Signed Report Status Declaration Form
√	Signed FYP Thesis Submission Form
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
	List of Symbols (if applicable)
	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
√	Appendices (if applicable)
√	Weekly Log
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
√	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report.

Low
_____
(Signature of Student)
Date: 12/9/2024