UMPool: A Carpooling System

LEE LYE ENG

Bachelor of Computer Science (Software Engineering)

UNIVERSITI MALAYSIA PAHANG

UNIVERSITI MALAYSIA PAHANG

DECLARATION OF THESIS AND COPYRIGHT

Author's Full Name : LEE LYE ENG

Date of Birth

Title : UMPool: A Carpooling System

Academic Session : SEMESTER I ACADEMIC SESSION 2022/2023

I declare that this thesis is classified as:

☐	CONFIDENTIAL	(Contains confidential information under the Official Secret Act 1997)*
☐	RESTRICTED	(Contains restricted information as specified by the organization where research was done)*
☑	OPEN ACCESS	I agree that my thesis to be published as online open access (Full Text)

I acknowledge that Universiti Malaysia Pahang reserves the following rights:

1. The Thesis is the Property of Universiti Malaysia Pahang
2. The Library of Universiti Malaysia Pahang has the right to make copies of the thesis for the purpose of research only.
3. The Library has the right to make copies of the thesis for academic exchange.

Certified by:

(Supervisor's Signature)

(Student's Signature)

Ahmad Fakhri Bin Ab. Nasir

Name of Supervisor

Date:

Date: 20/02/2023

NOTE : * If the thesis is CONFIDENTIAL or RESTRICTED, please attach a thesis declaration letter.

THESIS DECLARATION LETTER

Librarian,
Perpustakaan Universiti Malaysia Pahang,
Universiti Malaysia Pahang,
Lebuhraya Tun Razak,
26300, Gambang, Kuantan.

Dear Sir,

CLASSIFICATION OF THESIS AS RESTRICTED

Please be informed that the following thesis is classified as RESTRICTED for a period of three (3) years from the date of this letter. The reasons for this classification are as listed below.

Author's Name
Thesis Title

Reasons (i)

(ii)

(iii)

Thank you.

Yours faithfully,

_____
(Supervisor's Signature)

Date:

Stamp:

Note: This letter should be written by the supervisor, addressed to the Librarian, *Perpustakaan Universiti Malaysia Pahang* with its copy attached to the thesis.

SUPERVISOR'S DECLARATION

I/We* hereby declare that I/We* have checked this thesis/project* and in my/our* opinion, this thesis/project* is adequate in terms of scope and quality for the award of the degree of *Doctor of Philosophy/ Master of Engineering/ Master of Science in ……..

(Supervisor's Signature)

Full Name : Ahmad Fakhri Bin Ab. Nasir

Position : Lecturer

Date : 20/2/2023

(Co-supervisor's Signature)

Full Name :

Position :

Date :

STUDENT'S DECLARATION

I hereby declare that the work in this thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at Universiti Malaysia Pahang or any other institutions.

LyeEng.

(Student's Signature)

Full Name : LEE LYE ENG

ID Number : CB19092

Date : 20/02/2023

UMPool: A Carpooling System

LEE LYE ENG

Thesis submitted in fulfillment of the requirements
for the award of the degree of
Bachelor of Computer Science (Software Engineering)

Faculty of Computing
UNIVERSITI MALAYSIA PAHANG

FEBRUARY 2023

ACKNOWLEDGEMENTS

Throughout the writing of this thesis and project development, I have received a great deal of support and assistance.

Foremost, I would like to express my sincere gratitude to my supervisor, Dr Ahmad Fakhri Bin Ab. Nasir, whose expertise was invaluable in developing and completing this project. Your insightful feedback pushed me to honing my skill and brought my work to a higher level.

This venture would not have been feasible without the gracious help of our Undergraduate Project coordinator, Dr. Danakorn Nincarean A/L Eh Phon, who guided, explained, and tolerated us throughout the project planning and the thesis writing process.

Lastly, I would be remiss in not mentioning my parents and friends who helped me tremendously by sharing inventive ideas and suggestions at the various stages of project completion. Your encouragement and support have always kept me going in times of challenge.

ABSTRAK

UMPool: Sistem Carpooling ialah sistem berasaskan web yang membawa manfaat kepada pemandu carpool, penumpang, alam sekitar dan masyarakat. UMPool menyediakan tempahan awal untuk perkhidmatan carpool di kampus kami. Sistem ini membolehkan pengguna mengurus profil, carpool, mengesahkan pemandu, menyemak carpool mereka dan membuat pembayaran. Permintaan carpool di kampus semakin meningkat disebabkan faktor utama ini, kawasan kampus yang luas dan pengangkutan awam yang terhad. Oleh itu, objektif sistem cadangan ini adalah untuk mengkaji sistem carpooling sedia ada, membangunkan sistem carpooling berasaskan web untuk UMP dan menilai keberkesanan dan kefungsian sistem cadangan. Kertas kerja ini mencadangkan User Acceptance Testing (UAT) dan Functional Testing untuk aplikasi UMPool. Sebanyak 27 kes ujian telah dijalankan untuk mengesahkan prestasinya, dan UMPool mencapai penarafan bebas ralat 96.30% dengan hanya satu kecacatan kecil yang dikenal pasti. Daripada keputusan ujian UAT, majoriti responden bersetuju bahawa UMPool berkesan dalam membantu mereka dengan carpooling. Kesimpulannya, UMPool telah berjaya memenuhi semua keperluan fungsian dan berjaya mengatasi penyataan masalah kajian kes.

ABSTRACT

UMPool: A Carpooling System is a web-based system that brings benefits to carpool drivers, passengers, environment, and society. UMPool provides advance bookings for carpool services on our campus. This system enables users to manage profiles, carpool, verify drivers, review their carpool, and make payment. The demand of carpooling in the campus is increasing due to these major factors, vast campus area and limited public transport. Therefore, the objective of this proposed system is to study the existing system of carpooling, develop a web-based carpooling system for UMP and evaluate the effectiveness and functionality of the proposed system. This paper proposes functional and User Acceptance Testing (UAT) for the UMPool application. A total of 27 test cases were conducted to verify their performance, and UMPool achieved a 96.30% error-free rating with only one minor defect identified. Form the results of UAT testing, the majority of participants agreed that the UMPool was effective in assisting them with carpooling. In conclusion, UMPool has successfully met all functional requirements and managed to overcome the case study's problem statement.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

%	Percent

LIST OF ABBREVIATIONS

ASEAN	Association of Southeast Asian Nations
CSS	Cascading Style Sheets
FAQs	Frequently-Asked Questions
FK	Faculti Komputeran
FTKEE	Faculti Teknologi Kejuruteraan Elektrik & Elektronik
FTKMA	Faculti Teknologi Kejuruteraan Mekanikal & Automotif
FTKPM	Fakulti Teknologi Kejuruteraan Pembuatan dan Mekatronik
GHG	Greenhouse Gas
HTML	Hypertext Markup Language
iOS	iPhone OS
JAD	Joint Application Development
KK5	Kolej Kediaman Kelima
MFA	Multi-Factor Authentication
N/A	Not Applicable
OTP	One Time Password
PAP	Pusat Aktiviti Pelajar
PBM	Pusat Bahasa Moden
PHP	Hypertext Preprocessor
QR	Quick Response
RAD	Rapid Application Development
RON	Research Octane Number
RP DHUAM	Resident Pelajar DRB Hicom University of Automotive Malaysia
SDLC	Software Development Life Cycle
UAT	User Acceptance Testing
UMP	Universiti Malaysia Pahang
U.S.	United States
3D	Three-Dimensional

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Carpooling refers to ridesharing that was introduced by the United States during World War II through "car clubs"(States et al., 2022). The idea of carpooling involves passengers sharing a ride together to a common destination. According to the U.S. Census Bureau report on specific commute modes, the rate of carpooling has declined over the years, declining from 19.7 percent in 1980 to 9.4 percent in 2013 as illustrated in Figure 1.1(McKenzie, 2015). While carpooling continues to decline due to limited understanding of the contributing factors, there is increasing evidence that carpooling offers numerous societal, employer, and individual impacts(Polzin, 2022; Work, 2018).

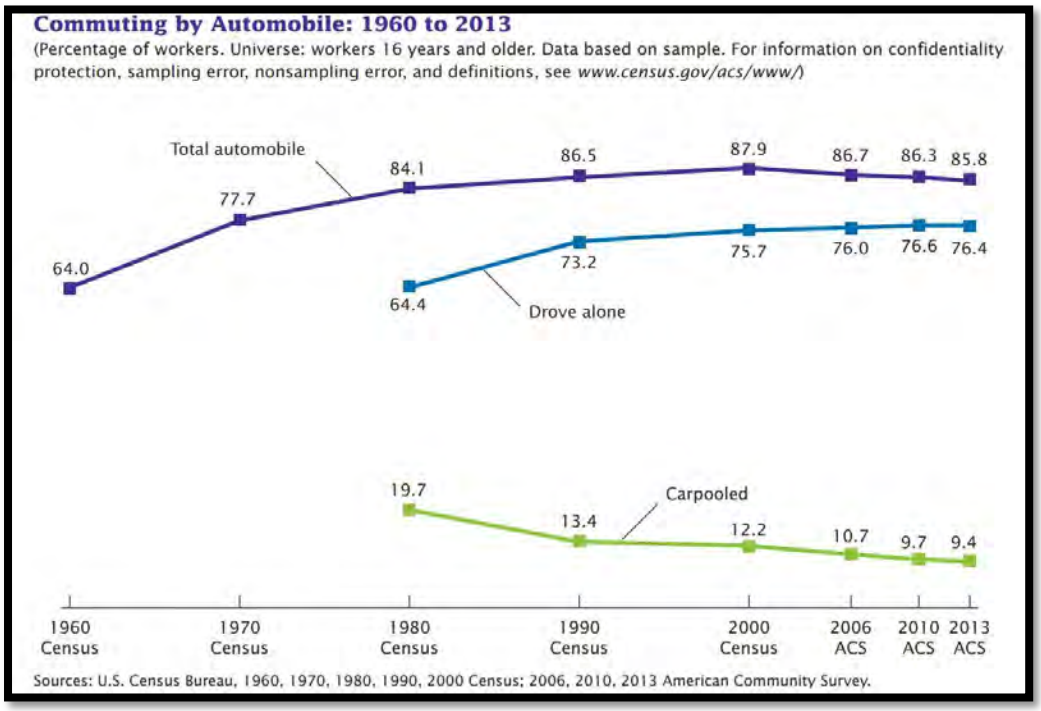


Figure1.1 Commuting by Automobile: 1960 to 2013

The impact of carpooling on society reduces fuel consumption and greenhouse gas (GHG) emissions. The average car and sports utility vehicle consume 550 and 915 gallons of fuel annually, respectively(Work, 2018). Jacobson and King (2009) found that with an additional passenger to every 10 vehicles, fuel consumption could be reduced by 7.54 to 7.74 billion gallons(Work, 2018). It is estimated that using carpools to optimize roadway performance could save 70 million to 190 million tons of carbon dioxide annually.

Cost saving is the main benefit of carpooling for employees and individuals. With ride splitting services, commuters with longer distances to cover and higher commute costs can pay a reduced fare by participating in carpooling services. Commuters who carpool can offset the rise in petrol prices, since the price of RON 97 gradually increased until the third week of March at RM 4/litre as illustrated in Figure 1.2(CompareHero, 2021).

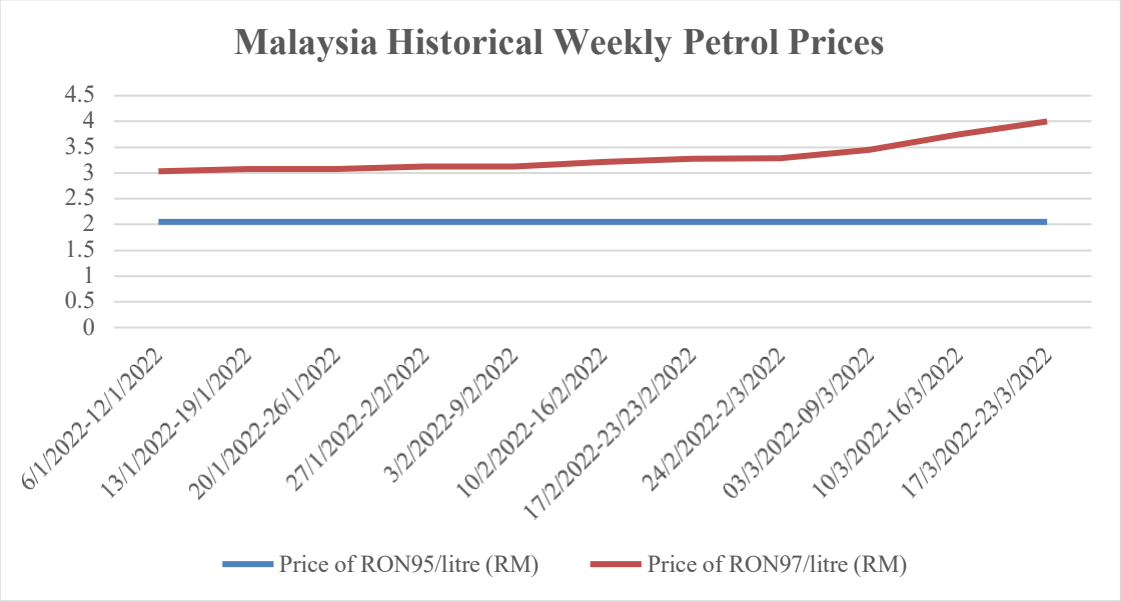


Figure 1.2 Malaysia Historical Weekly Petrol Prices

Numbeo's study of Traffic Index by City 2020 Mid-Year states that Malaysia has the fourth worst traffic congestion in ASEAN and the second highest level of carbon dioxide emissions(Jerrica, 2021). The government has devised a series of strategies to address this issue, such as the National Land Public Transport Master Plan (2012-2030) which promotes public transportation and limits the use of private

cars(Susskind et al., 2020). Therefore, a web-based carsharing application is crucial to encourage more people to use carpooling services.

1.2 PROBLEM STATEMENTS

Traffic is one of the most troublesome things for every university student. It is necessary to provide transportation for students living at UMP Pekan because of the distance between the classroom, dormitories, and the city. This is due to the fact that UMP Pekan is located in a rural area of Pekan(Manoj Kumar et al., 2019). In the absence of a vehicle, some issues can arise that might be frustrating to staff and students. There were four main problems identified as the reason for developing the proposal.

The first problem is the scarcity of parking spaces at each faculty. It will be difficult for students who live off campus since they will need to commute by car to campus. They have to spend a lot of time looking for a parking space, which directly affects their productivity during class.

The second problem is traffic congestion when a majority of students and staff drive to classes. A worsening traffic congestion problem will force students to miss classes or tests. It will inevitably result in a decline in teaching and learning quality and a regression in student performance in the long run.

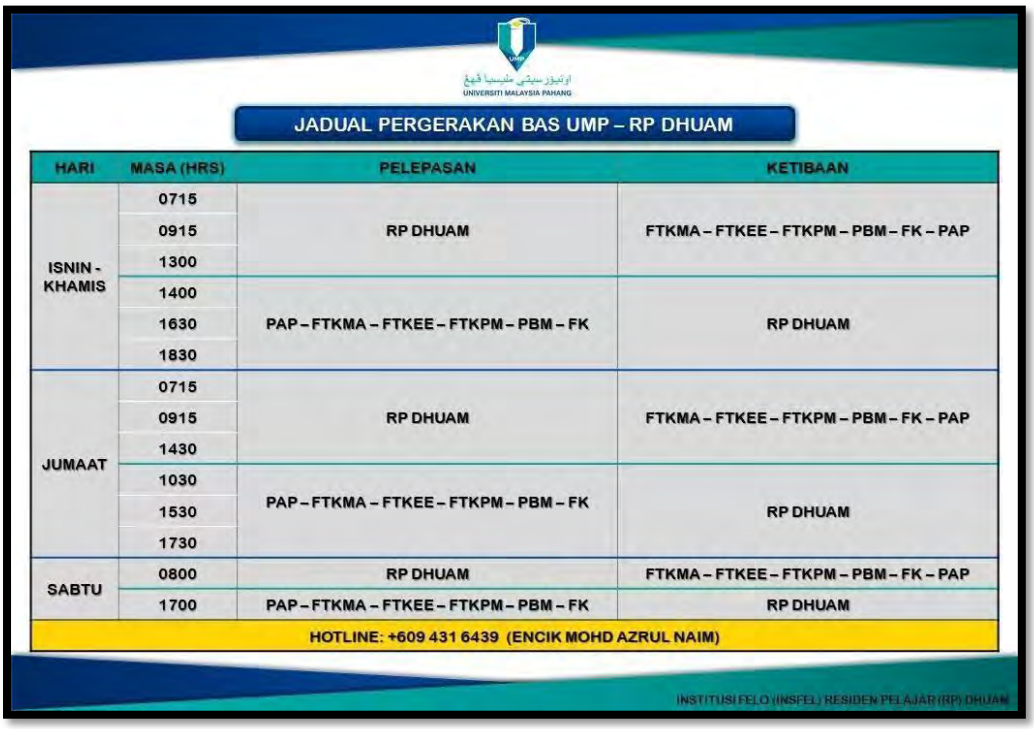
The third problem is UMP students have a long and difficult time getting to and from campus due to the vast campus area. With the campus area in Pekan has 642 acres, staff and students are hard to reach to other facilities without a car. For instance, it takes 30 minutes to walk from hostel KK5 to the faculty of computing. The situation will be more problematic when students are pressed for time because of an unexpected meeting or are late for class.

The fourth problem is the number of public vehicles that are extremely low at the UMP Pekan campus. Currently, shuttle bus services are only available back and forth between faculty buildings (FTKMA, FTKEE, FTKPM, PBM, FK, PAP) and RP DHUAM, but not from one faculty to another as illustrated in Figure 1.3. Besides, shuttle bus services are extremely limited, with buses operating three times per day from campus to the DHUAM on weekdays. The inconvenience of these reasons results

in UMP staff and students without cars having limited opportunities to go out for business, such as delivering parcels.

The fifth problem is the unavailable of reliable carpool reservations platform for staff and students at UMP. Students are typically required to make reservations with driver at least one day in advance for shared rides. Otherwise, it can be difficult to locate available drivers without bookings, so if students without a car need to go out urgently, they need to contact the available driver through the student portal, which often takes some time.

The sixth problem is the registration process and application procedures for vehicles are complicated. UMP students need to submit a variety of materials for UMP Security Division review, and there are too many requirements associated with the application of a student vehicle sticker as shown in Figure 1.4. In addition, renewing a car sticker annually is time-consuming for UMP students since they must submit the same documentation as a new applicant to apply.



UNIVERSITI MALAYSIA PAHANG

JADUAL PERGERAKAN BAS UMP – RP DHUAM

HARI	MASA (HRS)	PELEPASAN	KETIBAAN
ISNIN - KHAMIS	0715	RP DHUAM	FTKMA – FTKEE – FTKPM – PBM – FK – PAP
	0915		
	1300		
	1400	PAP – FTKMA – FTKEE – FTKPM – PBM – FK	RP DHUAM
	1630		
	1830		
JUMAAT	0715	RP DHUAM	FTKMA – FTKEE – FTKPM – PBM – FK – PAP
	0915		
	1430		
	1030	PAP – FTKMA – FTKEE – FTKPM – PBM – FK	RP DHUAM
	1530		
	1730		
SABTU	0800	RP DHUAM	FTKMA – FTKEE – FTKPM – PBM – FK – PAP
	1700	PAP – FTKMA – FTKEE – FTKPM – PBM – FK	RP DHUAM
HOTLINE: +609 431 6439 (ENCIK MOHD AZRUL NAIM)			

INSTITUSI FELO (INSFEL) RESIDEN PELAJAR (RP) DHUAM

Figure 1.3 Timetable of UMP Shuttle

PERMOHONAN PELEKAT KENDERAAN SISWA

Syarat Kelayakan Pemohon
1. Siswa Under Graduate
2. Siswazah (Post Graduate)
3. Dihadkan sebuah bagi seorang pelajar.
4. Kenderaan Sendiri

Kemukakan dokumen asal: (untuk semakan Pejabat Keselamatan)
1. Kad matrik,
2. Lesen memandu,
3. Lesen cukai jalan
4. Pendaftaran kenderaan.
5. Jika tiada pendaftaran (geran) kenderaan, bawa surat jual beli atau insuran.
6. Surat Kebenaran Pemilik Kenderaan (Jika pemilik kenderaan bukan ibu, ayah atau adik -beradik - tiada pertalian yang jelas)

Proses mendapatkan pelekat :
1. *Apply* dalam e-comm (Vehichle Registration)
2. *Print Out* permohonan
3. Buat bayaran RM3 - E-payment (Pejabat Keselamatan Gambang dan Pekan & setiap Jabatan yang ada alat tersebut), Maybank2U(cetak resit bayaran) @ di Jabatan Bendahari (Unit Penerimaan)
4. Serah resit bayaran dan permohonan (cetakan) - Bahagian Keselamatan.
5. Pamer pelekat pada bahagian hadapan kenderaan (mudah dilihat).

Permohonan **TIDAK** Dipertimbangkan :
1. Bertujuan untuk sewa/perniagaan.
2. Pemilik tiada hubungan saudara (tidak jelas)
3. Kenderaan Mempamerkan Imej Persatuan/Kelab/NGO/Kebajikan/Ugama
4. Pelajar tahun 1

Pemohon hanya boleh membuat satu permohonan sahaja (kereta atau motosikal).Pemohonan tidak akan dilayan jika dokumen yang diperlukan tidak lengkap. Bagi pelajar yang pernah memohon untuk sesi yang lepas, dokumen yang diperlukan adalah sama seperti permohonan yang baru. Pelajar tahun 1 tidak dibenarkan membawa kenderaan di kawasan kampus.
Sebarang pertanyaan lanjut boleh hadir ke Pejabat Keselamatan.

Figure 1.4 Application materials and requirements for student vehicle stickers

1.3 OBJECTIVES

There are three objectives are defined in this project which are:

i. To study the existing system of carpooling.

ii. To develop a web-based carpooling system for UMP.

iii. To evaluate the effectiveness and functionality of the proposed system.

1.4 SCOPE

The project scope will be focus on:

i. Users

UMP staff and students are the intended users of the project. There are three roles in the system: admin, driver, and passenger. The registration process

allows a user to register as either a driver or a passenger. Carpooling can be created and accepted by both drivers and passengers meaning that the driver can create the offer and wait for the passenger to accept it, while the passenger can also create the offer and wait for the driver to accept it.

After acceptance of the offer by the driver or passenger, the payment must be completed by the passenger in order for the offer to be confirmed. The administrator will evaluate the driver's identification by accepting, rejecting, and viewing the driving verification.

ii. Software

The software used to develop this project is Visual Studio Code.

iii. Hardware

The hardware used to develop this project is Lenovo Xiaoxin Pro 14. Users are required to use their smartphone to scan the QR code to save contact number of driver, download, and print the carpool invoice.

iv. Environment

The environment of the system used are HTML, CSS, Javascript, PHP and Laravel framework.

1.5 SIGNIFICANCE OF PROJECT

i. To alleviate the scarcity of parking spaces on campus by promoting carsharing.

ii. To prevent traffic congestion on campus.

iii. To assist staff and students who do not own cars in getting to campus and other facilities on such short notice.

iv. To enable users to give review after carpool is completed.

v. To enable users to search the carpool according to their preferences.

1.6 REPORT ORGANIZATION

The thesis is divided into five chapters. Chapter 1 describes the introduction to the project. These include the problem statements of the project, the objectives, scope, significance, and organization of the thesis. A major focus of this chapter is the background of carpooling, the problems arising without the application, the objectives to be accomplished by the project, the scope of the project and the importance of the project.

Chapter 2 discusses the literature review related to the project. It describes the existing system through a comparative analysis. There are three existing systems that will be compared which are CarpoolWorld, WeRide, and ZipShare.

Chapter 3 explains the methodology of the project. It outlines the methodology, use case, storyboard, flowchart, hardware and software specification and gantt chart used.

Chapter 4 describes the implementation of the project results and discussion.

Chapter 5 discusses the conclusion of the project. It describes the conclusion based on the objective, the limitations of the project and its future work.

CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

This chapter consists of five (5) sections, which are introduction, comparison of the advantages and disadvantages of the three existing works, and conclusion. Section 2.2 examines three existing carpooling systems that include web-based applications and mobile applications. Section 2.3 discusses comparative analysis. Section 2.4 discusses security and integrity of the application. This chapter concludes with section 2.5.

2.2 THREE RELATED WORK

Section 2.2 will explain and analyse the current applications in detail, as well as the advantages and disadvantages that must be considered to optimize potential applications. Three existing systems are discussed in the following section are CarpoolWorld, WeRide, ZipShare.

2.2.1 CarpoolWorld (https://www.carpoolworld.com/)

CarpoolWorld provides real-time trip-matching through both web-based and mobile applications. It provides carpooling and vanpooling for public use, while premium carpooling is offered to organizations such as businesses, hospitals, universities, and events. Prior to using the system, users must login to their account or register. Following that, users need to fill in user information and trip information to schedule the trip. The system will calculate and display average savings and costs, including fuel consumption. Available rides will be displayed to users according to their

preferences. Users can contact the driver or passenger by clicking on the contact button on the carpool listing, and the system will redirect them to the contact driver or passenger page where users can either message them or call them if necessary.

Figure 2.1 User Interface of user login and registration of CarpoolWorld

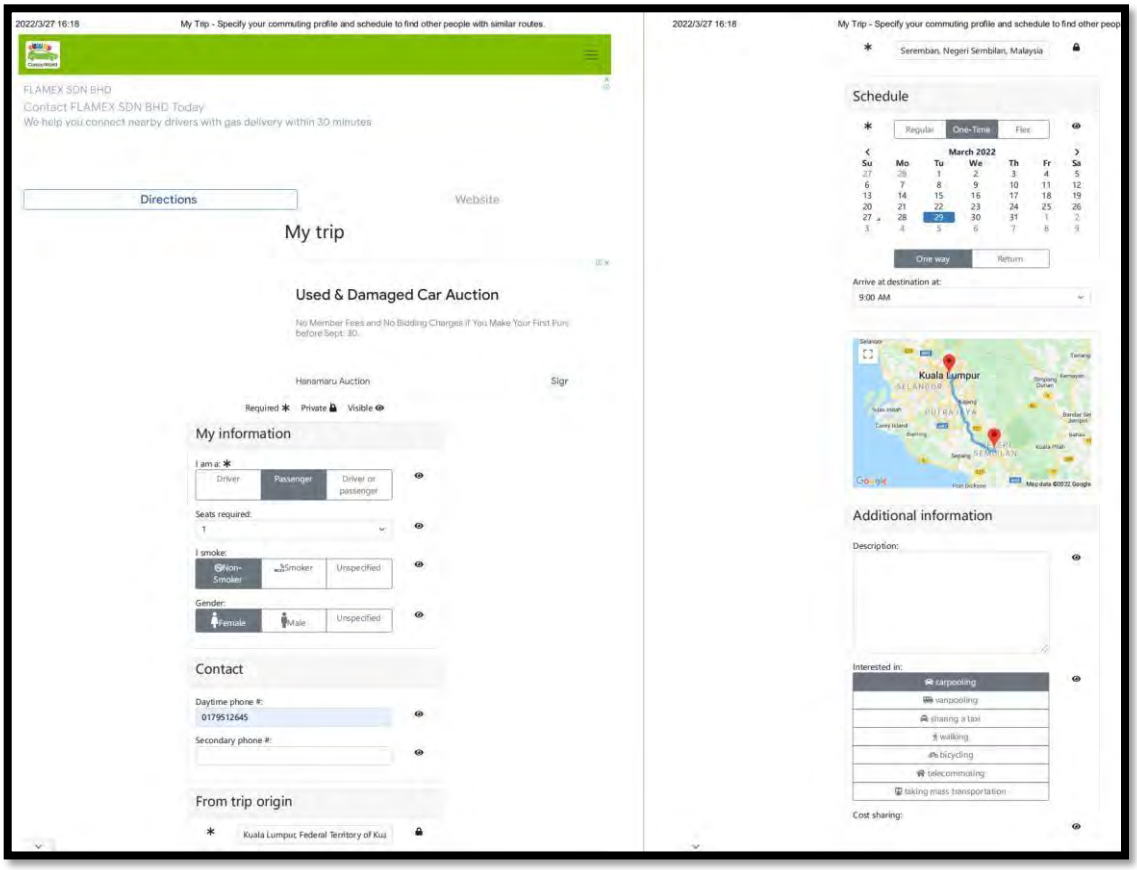


Figure 2.2 User Interface of user and trip information

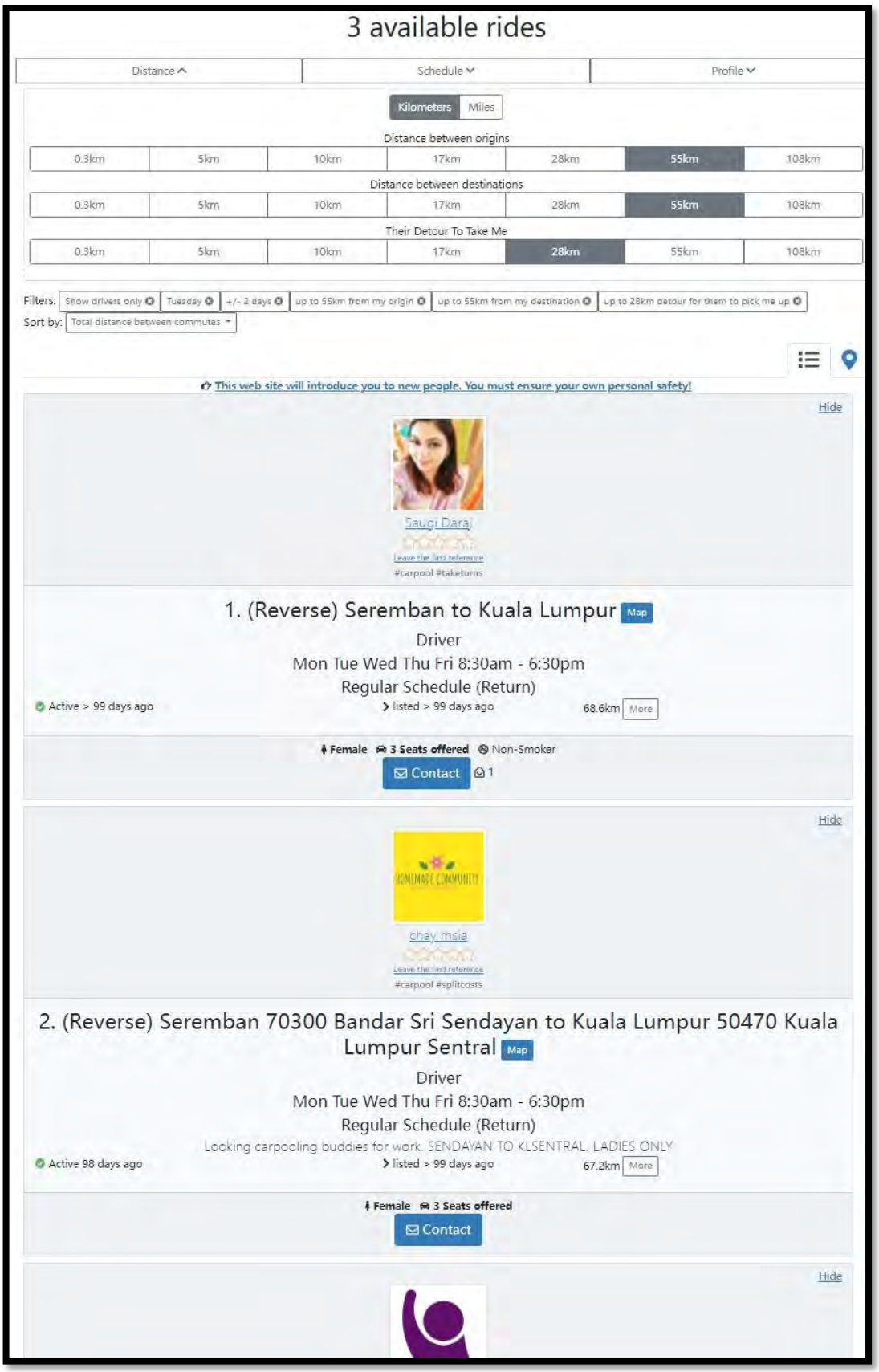


Figure 2.3 User interface of carpool listings available on CarpoolWorld

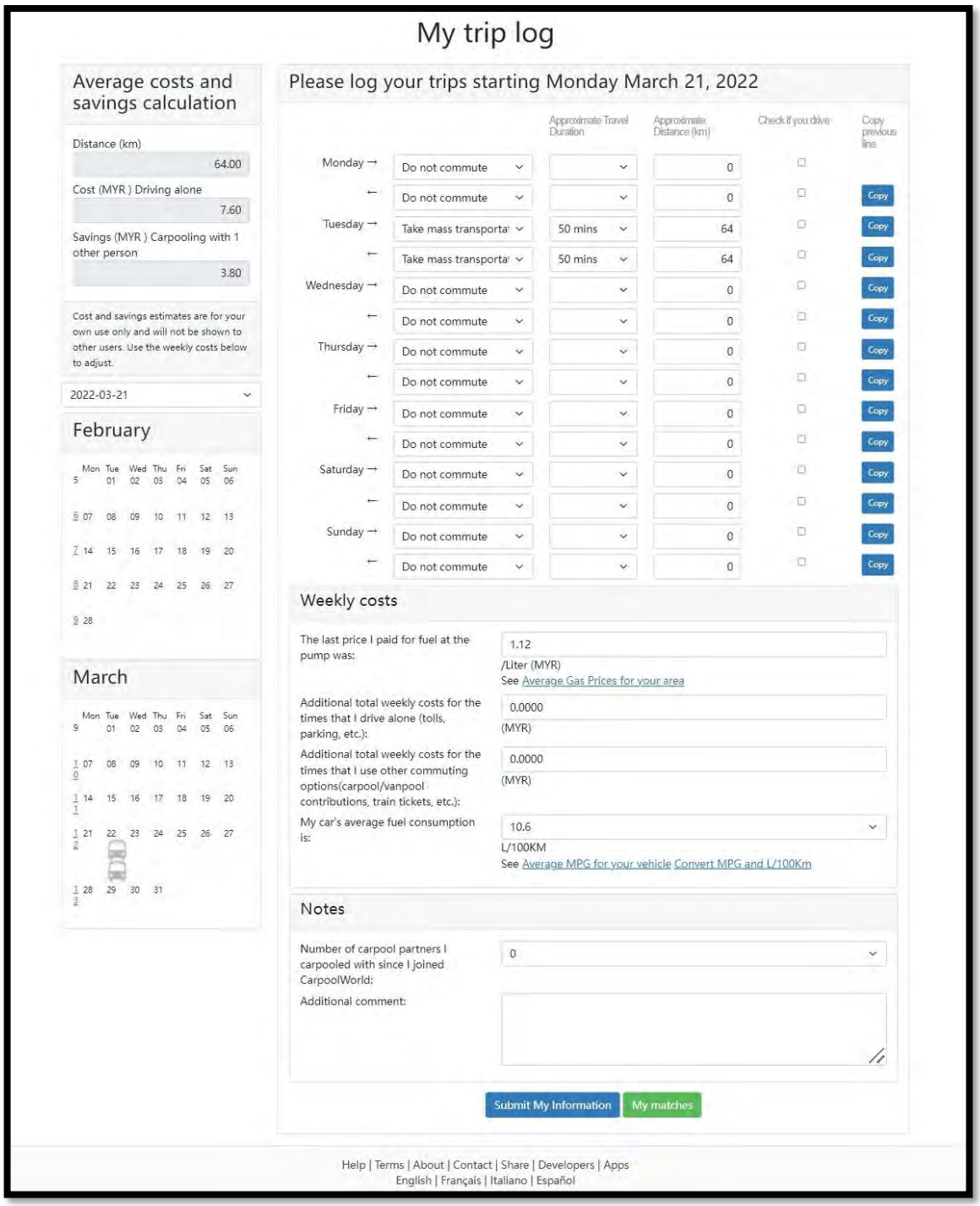


Figure 2.4 User interface of carpooling cost estimation

2.2.2 WeRide (Mobile Application)

The WeRide platform facilitates carpooling by matching passengers and drivers in Malaysia and Singapore. There are 2 types of users in the system which are passengers and drivers. The application offers a variety of services, such as searching and filtering trips, discovering trip requests, monitoring traffic by using traffic camera images, traffic news, and managing rides. WeRide enables users to sign in using Facebook, Google and Apple to speed up the registration process. Users can search for and filter carpool trips by choosing to be the driver, or passenger, as well as origin and destination when accessing the system. Drivers and passengers can contact each other by clicking on the carpool offer on the listing, which will direct users to the contact page where the driver or passenger can be contacted via Facebook, Messenger, or WhatsApp.

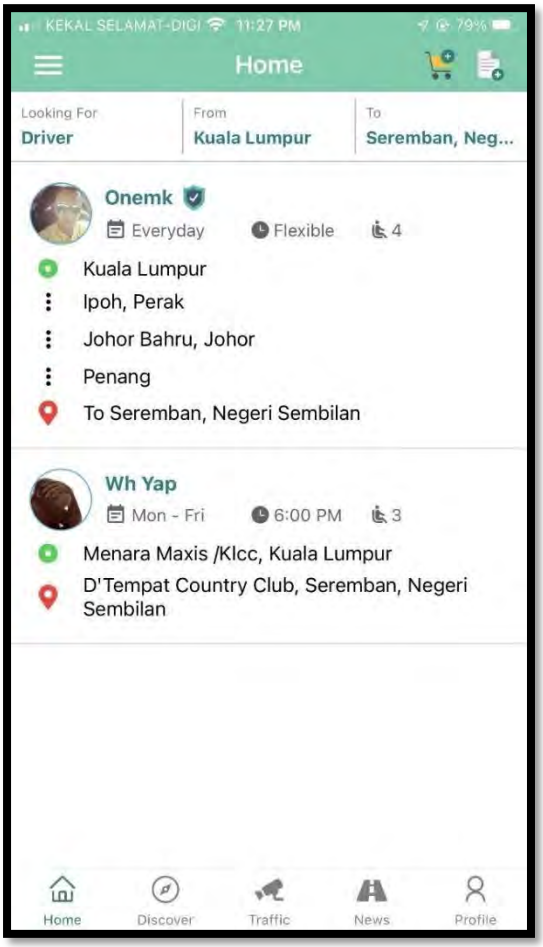


Figure 2.5 User interface of carpool listings available on WeRide

Figure 2.6 User interface of traffic camera view on WeRide

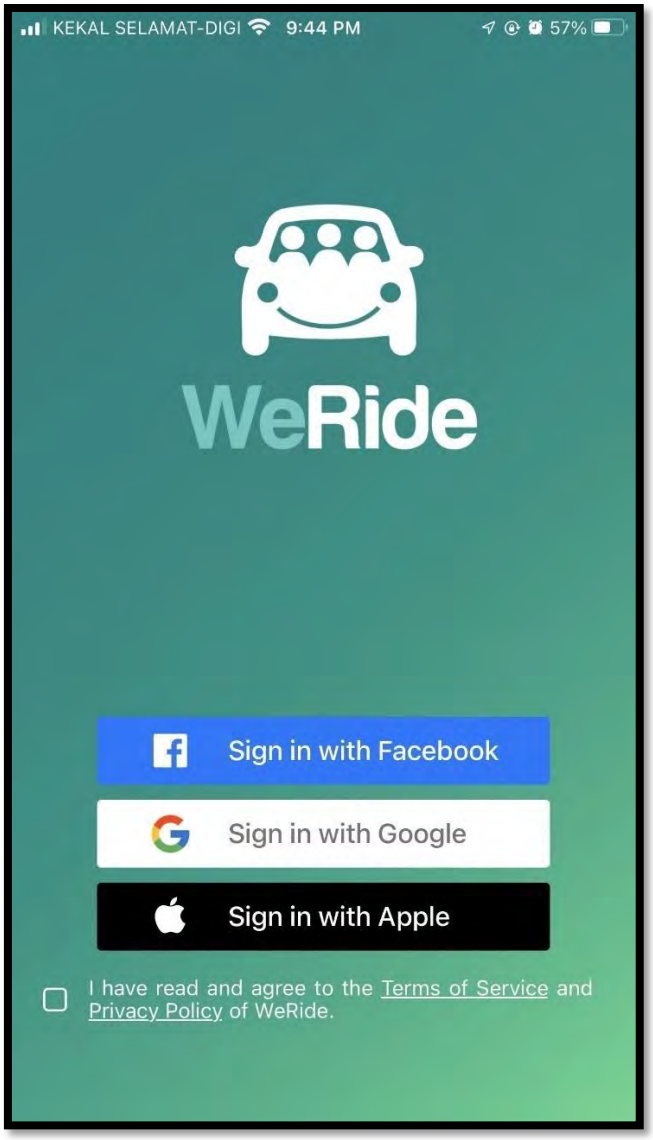


Figure 2.7 User interface of user login on WeRide

2.2.3 ZipShare (Mobile Application)

ZipShare is a mobile ridesharing application that targets schools in United State to simplify carpooling and reduce pollution. There are 2 types of users in the system which are parents and students. In this way, students who are in university can search for car partners in a convenient way. A user can create a carpool group with his/her friends utilizing the invitation link. A built-in calendar allows the user to update the pick-up location, driver, and passengers with time scheduled for carpooling.

Figure 2.8 User interface of route tracking on ZipShare

Figure 2.9 User interface of built-in schedule on ZipShare

2.3 Comparative Analysis

This section examines the unique features of CarpoolWorld, WeRide, and ZipShare applications, which can have a major impact on society. Table 2.1 summarizes the three existing systems in detail in terms of features, platform, and user. Table 2.2 presents the advantages and disadvantages of the three existing systems.

Table 2.1 Comparison Features of Existing Systems

System Features	Existing Systems
Create group	CarpoolWorld • Yes This feature only available for subscriptions plan users as illustrated in Appendix B, Figure 1.
	WeRide • N/A
	Zipshare • Yes It only allow users who have an invitation link to form a group, as shown in Appendix B, Figure 8 to 12.
Communication within platform	CarpoolWorld • Yes • It allows send and receive message to driver within platform as illustrated in Appendix B, Figure 2. • As an alternative, passengers are allowed to use the driver's personal contact information to communicate privately with the driver.
	WeRide • N/A • Instead of connecting users within the platform, it integrates with WhatsApp, allowing the user to communicate with the driver, as illustrated in Figure 4 and 5 of Appendix B.
	Zipshare • Yes • Communicate through group chat by sending message and photo as illustrated in Appendix B, Figure 11.
Display carpool listing	CarpoolWorld • Yes, users can access the latest carpool listing, as shown in Figure 2.3.
	WeRide • Yes, users can access the latest carpool listing, as shown in Figure 2.5.
	Zipshare • Yes • It only available the carpool listing in group as illustrated in Appendix B, Figure 7
Display route	CarpoolWorld • Yes • It provides display route feature by integrating with Google Maps.
	WeRide • N/A

	Zipshare • Yes, it provides navigation and displays the estimated arrival time, as illustrated in Figure 2.8.
FAQs	CarpoolWorld • N/A
	WeRide • Yes, the "help" link in the navigation will direct users to their browser, where it will display FAQs that help them deal with their difficulties.
	Zipshare • N/A
Integrated payment method	CarpoolWorld • N/A
	WeRide • N/A
	Zipshare • N/A
Invite friends	CarpoolWorld • Yes, user allows to invite their friends by disseminating invitation code.
	WeRide • Yes, user allows to invite their friends by disseminating invitation link.
	Zipshare • Yes, user allows to invite their friends by disseminating invitation link as illustrated in Appendix B, Figure 11.
Manage post	CarpoolWorld • Yes, Shown in Appendix B, Figure 3
	WeRide • Yes, users allow to share their post, amend, remove the post, and halt the request, making it suited for non-recurring carpool users as shown in Appendix B, Figure 6.
	Zipshare • N/A
Manage profile	CarpoolWorld • Yes
	WeRide • Yes, it enables users to update their contact information,

	upload carpool photos, and include their Facebook profile link.
	Zipshare • Yes, users can amend their profile image, password, username, email, school, and home address.
Multi-Platform	CarpoolWorld • Yes, it comprises of both a web-based and a mobile application.
	WeRide • N/A, it developed for mobile application only.
	Zipshare • N/A, it developed for mobile application only.
Search and filter	CarpoolWorld • Yes, it allows user to search direction and filter results by showing passengers only, drivers only and both
	WeRide • Yes, it allows user to search direction and filter carpool available by showing (driver / passenger), origin and destination
	Zipshare • Yes, it allows user to search the location and other members nearby
Verify driver and passenger	CarpoolWorld • N/A
	WeRide • Yes • It offers verified and other banners, such as a 10-year driving length, to entice passengers to join their carpool. (Shown in Figure 2.5)
	Zipshare • N/A
Real-time traffic monitoring	CarpoolWorld • N/A
	WeRide • Yes • Monitoring the traffic in Malaysia & Singapore is possible through the traffic cams and news that are available, as shown in Figure 2.6.
	Zipshare • N/A
User Involved	CarpoolWorld

	• Drivers and passengers are involved in this system.
	WeRide • Drivers and passengers are involved in this system.
	Zipshare • Developed for U.S. high schools, parent who are unable to send their children to school can keep track of the routes of their children's carpools and students can discuss the appropriateness of carpool timetables with parents who are able to.

Table 2.2 List of advantages and disadvantages on three (3) existing system

Specification	Advantages	Disadvantages
CarpoolWorld	✓ Provides real time trip-matching services ✓ Allows user to communicate with trip-matching users via the platform to make bookings ✓ Available in multiple platform ✓ Free of charge ✓ Available in Malaysia ✓ Ability to estimate the cost of carpooling in a trip log	✓ Unverified passengers and drivers ✓ Platform not available for trading / payment
WeRide	✓ Provides recurring / non-recurring carpool ✓ Provides real-time traffic images in Malaysia and Singapore ✓ Discover traffic news ✓ Receive notifications of new offers/requests for carpooling ✓ Free of charge ✓ Available in Malaysia and Singapore	✓ View carpool listing by watching advertisement or subscribing to the premium plan ✓ Contact the trip-matching user by social media or phone call to make booking ✓ Platform not available for trading / payment

	✓ Verify passengers and driver ✓ Discover latest carpool post ✓ Rapid registration and login process	
Zipshare	✓ Manage member in carpool group ✓ Provides instant messaging and photo sharing within a group ✓ Schedules carpools with the built-in calendar ✓ Free of charge ✓ Navigation of the route with estimated arrival time	✓ Unverified passengers and drivers ✓ Platform not available for trading / payment ✓ Not available in Malaysia ✓ Unclear user interface, user unable to view the data they entered since the text displayed is the same colour as the background.
All systems	✓ Provide invitation of friends	✓ Unavailable payment gateway supported

Table 2.2 shows the list of advantages and disadvantages of these existing systems. The main advantage of the CarpoolWorld system is its ability to estimate the cost of carpooling in a trip log, as shown in Figure 2.4. Fuel prices per litre and consumption of particular vehicles are factors that affect the cost of carpooling. The app allows users to find other users nearby in real time to share daily commutes. However, it has some limitations, such as the inability to book directly in the system, and the user must contact the trip-matching users to inquire about availability. Currently, the system only offers ride-matching services, but cannot accept payment and trading for bookings. Drivers who have not been verified are also a vulnerability of the system, which indirectly results in scam and unprotected trips.

A major advantage of the WeRide system is the rapid registration and login process. Users can create an account in a convenient way by logging in through Apple, Facebook or Google as illustrated in Figure 2.7. Furthermore, authenticating drivers and passengers provides consumers with an extra layer of security for preventing unauthorized access and finding suspicious activity. Additionally, users able to keep

track of the traffic at popular sites and receive traffic news via the app so that they can avoid gridlock and avoid busy roads. However, users are required to watch an advertisement to unlock the carpool listing if they are not premium subscribers. Nevertheless, it only allows private communication with drivers when making bookings via social media, such as Telegram.

Navigation of the route with estimated arrival time is a major advantage of the Zipshare system, as illustrated in Figure 2.8. Parents can keep track of the route when they are unable to send their children to school. Users can manage members by sharing the group invitation link with friends or removing people from the group. Users can manage their schedule, such as who is the driver and passenger, by using the integrated calendar of the system. However, the design of the user interface is not to be overlooked. User unable to view the data they entered since the text displayed is the same colour as the background. User confusion is a result of the unclear interface, and the user may be unsure about the password they chose, which makes it more difficult to recall the data entered. It also suffers from the similar limitations of WeRide systems which are unverified passengers, and drivers, and not being able to process carpool payments within the platform.

One of the drawbacks of all existing systems is the lack of supported payment gateways. The absence of payment gateways in all these systems results in users conducting their transaction with drivers privately, which makes them vulnerable to fraudulent transactions. As passengers are not required to pay before the trip, they may purposely undercut price during the negotiation of the price. Hence, it is essential that a real time payment gateway is embedded in a carpooling software to improve the user experience.

2.3.1 Comparison Between Existing System And Proposed System

Table 2.3 Comparison between existing system and proposed system

Function	Existing Systems (CarpoolWorld, WeRide, and Zipshare)	Proposed System (UMPool)
Payment Gateway	N/A	Yes. It has two payment method in this proposed system which are Cash and Stripe.
Make Complaint	N/A	It provides the option for the passenger to give the feedback of the trip. User is allowed to communicate/ complaint through the social media platform provided in the footer of the system.
Scan QR code	N/A	Yes. i. Generate E-ticket. User are allowed to scan QR code to save the e-ticket when the payment is made.
Manage review	<u>CarpoolWorld</u> Yes. It allows users to leave references in order to share their experiences with others. <u>WeRide</u>	Yes. It allows users to first read the review before proceeding to join the carpool offer.

	N/A ZipShare N/A	
Chat with other users	CarpoolWorld Yes. It allows users to leave the message within the application. WeRide Yes. It directs the user to connect with other users' WhatsApp accounts. ZipShare Yes. It allows users to chat within the platform. Graphics and other multimedia components can be sent in the group conversation.	Yes. It allows users to interact each other by WhatsApp API.
Search carpool offer	CarpoolWorld Yes. Users can search for origins and destinations, and filter carpool listings by showing specific user types. WeRide Yes. Users can search for origins and destinations, and filter carpool listings by showing specific user types.	Yes. It allows users to search the carpool offer based on origin and destination.

	ZipShare Yes. It allows user to search for orgin and destinations, and filter it based on group.	

2.4 HARDWARE / TECHNOLOGY / TOOLS

Table 2.4 Lists of hardware/ technology/ tools on three (3) existing system

System / Aspect	CarpoolWorld	WeRide	Zipshare
Development tools	✓ PHP	✓ jQuery	✓ Ruby
Server	✓ Apache	✓ Apache	✓ NGINX
Browser	✓ Google Chrome ✓ Mozilla Firefox ✓ Safari ✓ Mobile Application	✓ Android ✓ iOS ✓ Huawei	✓ iOS
Authentication	✓ Google Sign-in ✓ Facebook Login	✓ Google Sign-in ✓ Facebook Login ✓ Apple Login	✓ Email and password
Security	✓ reCAPTCHA	✓ One-Time Password	✓ One-Time Password
UI Framework	✓ Bootstrap	✓ Materialize CSS	✓ Bootstrap
Frameworks	✓ WordPress	N/A	✓ Ruby in Rails

The list of hardware, technology, and tools used in three existing systems are illustrated in Table 2.4. These systems are built using WordPress and Ruby on Rails frameworks in conjunction with PHP, jQuery, and Ruby. Bootstrap and Materialize CSS are used for the UI frameworks of these systems. Open-source servers are applied in these existing systems which are Apache and NGINX which provide security against DDoS attack. CarpoolWorld and WeRide offer multiple web browsers, which have

improved user experiences in terms of compatibility and flexibility, while Zipshare only provides iOS browsers. Two-factor authentications in CarpoolWorld and WeRide applications provide an increased level of security as compared to single-factor authentication in Zipshare. The security technologies reCAPTCHA and One-Time Password (OTP) used in CarpoolWorld and ZipShare respectively provide a measure of protection against spam and abuse on the application.

2.5 SUMMARY

Ultimately, the detailed summary of comparison between existing systems will be the basis for developing the UMPool: A Carpooling System. The advantages and disadvantages of existing systems will be examined in this chapter. The end of this chapter will describe the hardware, technology, and tools used by existing systems.

CHAPTER 3

METHODOLOGY

3.1 INTRODUCTION

This chapter explains the development methodology for the UMPool: A Carpooling System. Throughout this chapter, a detailed description of the methodology, frameworks, tools, and instruments that will be explained in the system development process.

The methodology chosen for this project is Rapid Application Development (RAD). RAD is chosen due to its rapidity of development, which takes only two to six months to develop a new project. Additionally, it is highly flexible as developers can make changes at any time and add new functions.

3.2 PROJECT MANAGEMENT FRAMEWORK

RAD is an approach of software development that aims to deliver an efficient product compared to the traditional software development life cycle (SDLC). It focuses on speed, adaptability, and evolution of software development by reusing components from libraries and integrating them with prototypes, thereby reducing the need for coding (Rapley, 1995). Rapid development makes this model ideal for projects with small scales and frequent changes (Beynon-Davies et al., 1999)(Rapley, 1995). It is recommended that the RAD project development timeline be confined to two to six months (Beynon-Davies et al., 1999). Project will be overtaken by business development if it takes over six months (Beynon-Davies et al., 1999).

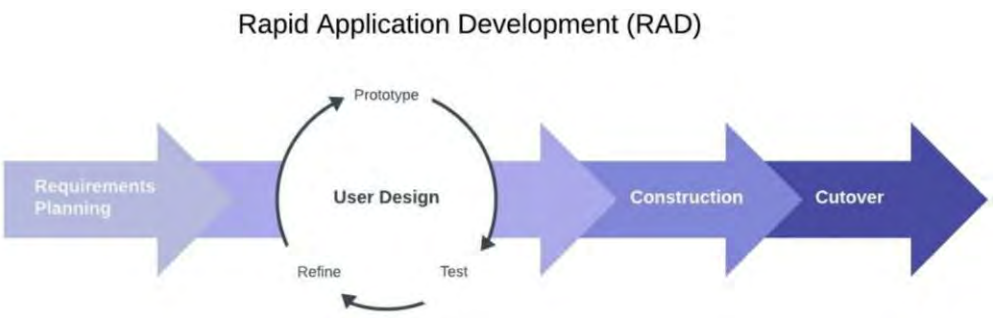


Figure 3.1 Rapid Application Development (RAD) methodology

i. Requirement Planning Phase

Google form survey is conducted for requirement planning that is open to community, UMP staff and students to collect the limitations of the previous system, seek for improvements, and acquire new requirements. It takes 51 respondents from Universiti Malaysia Pahang and community to collect reliable and credible data on the necessity of carpooling systems.

Result obtained from the survey above will be presented in visual form which shown in Appendix A. Figure 3.2 illustrates that 64.7% of respondents are not sure about the UMP shuttle bus, and 27.5% are unsatisfied with its services. Not sure review may be due to inexperience and rarely using UMP shuttle bus. According to the findings in Figure 3.3, 80.4 % of respondents are interested in carpooling to campus if such a system existed. Therefore, the proposed carpooling system must be implemented to make a difference on the campus.

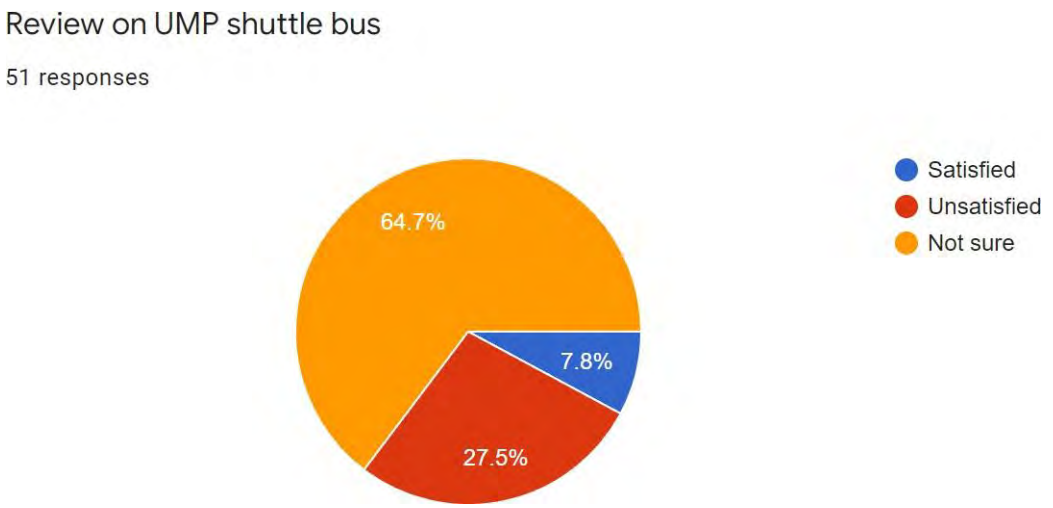


Figure 3.2 Result of google form A - Question 5

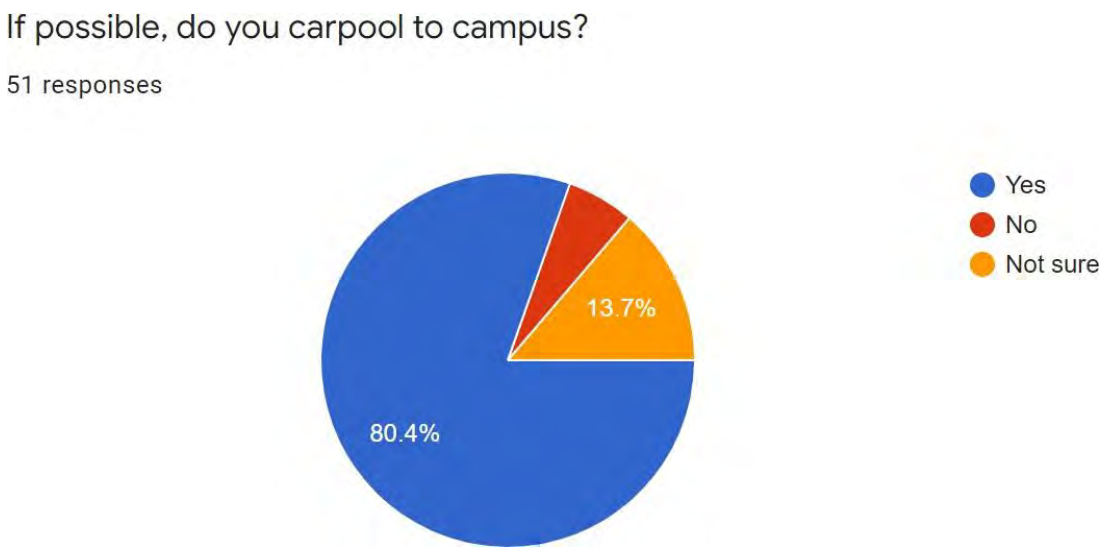


Figure 3.3 Result of google form A - Question 6

All stakeholders, including designers, developers, clients, and other experts are required to meet and discuss to finalize the vision, goals, features, requirements, scope, constraints, timelines, budget, and user expectation for this project once the survey results are obtained. After all these aspects are clearly defined, a formal agreement must be signed by the project owner and the service provider before starting the development process.

ii. User Design Phase

Software Requirement Specification and Software Design Description are constructed as illustrated in Appendix A. Prototypes of the system are shown in Appendix C.

iii. Construction Phase

The proposed project will be built with the Laravel framework and the PHP programming language. The product will be tested by user acceptance testing (UAT) to ensure it meets the expectations of the customer upon completion of the project.

iv. Cutover Phase

The final phase of implementation is the release of the final product. Data conversion, testing, and system transition, as well as user training, are carried out during this phase.

3.3 PROJECT REQUIREMENT

3.3.1 Functional Requirement

i. The system should allow users to login and register an account for accessing system.

ii. The system should enable users to update their profile details.

iii. The system should allow users to save E-ticket and track the carpool details by scanning the QR code provided.

iv. The system should display all aspects of the carpool listing, including carpool availability, pick-up and departure locations, and capacity of passengers.

v. The system should enable users to search carpool listings to locate the best carpool for their needs.

vi. The system should allow users to accept carpool offer that created by other users.

vii. The system should allow carpool owner to delete the carpool offer before someone accept the offer.

viii. The system should allow passengers to make payment. It provides convenience to users by using alternative payment methods if one of the payment methods is unavailable or under maintenance.

ix. The system should allow passengers to check their transaction history for references.

x. The system should enable users to give feedback to the driver by writing a review after the carpool trip as reference for other users and to help the company improve their service.

3.3.2 Non-Functional Requirement

i. Security requirement

- The login session expires for logged-in users after 15 minutes.
- Users are required to create a strong password when registering an account. The password must be at least 8 characters with combination of upper case, lower case, number, and symbol.
- Users will receive an email with a link to reset their password if they select forgot password.

ii. Usability requirement

- Users have the option of resetting their account credentials when creating a profile.
- Users have the option of downloading their E-ticket by scanning the QR code provided after the transaction is completed.

iii. Accessibility requirement

- User can adjust the map size to search for their preferred location with the enlarging and minimizing icons.

iv. Efficiency requirement

- Loading time for each page is less than 1 minute.

v. Integrity requirement

- Backups of user credentials must be performed at least monthly to prevent data loss.

vi. Compatibility requirement

- The application is compatible with Windows 7 and later operating systems.

3.3.3 Constraints

i. This system is designed only to match rides and to provide feedback after each ride has been completed. This system does not monitor and navigate the vehicle during the ride.

ii. Transportation costs can be calculated by using catalogued pricing and then dividing them by the number of users, including the driver, to determine how much each individual user must pay.

iii. Users are only able to register as driver or passenger during registration. No further changes to the character are permitted.

iv. All users are unable to create profile after the registration process. However, users are allowed to update and view their profile information at latest profile page.

v. Reservations are allowed in person only.

vi. Offer cancellation after payment is not allowed.

vii. The admin will hold the payment for passengers and release it to the driver when the carpool status is changed from upcoming to completed.

viii. Cash payments are only available if they accept the carpool created by other users.

ix. Admin has the authority to validate the driver's identity during the driving verification procedure.

x. Carpool offers can be created by both drivers and passengers, meaning that the driver can create the offer and wait for the passenger to accept it, while the passenger can also create the offer and wait for the driver to accept it.

3.3.4 Limitation

i. Ambiguous waiting time. Since the carpool offer cannot be cancelled once payment is made, driver must wait for the passenger and contact them until the passenger arrives. This will cause commuters to be late for work as the travel time will be longer than usual.

ii. System is unable to navigate the car during carpooling. Therefore, carpooling is uncontrollable by the system, and potential issues may arise, such as users losing their way, sexual harassment, or criminal cases.

iii. Route selection is not available to users. As previously stated, the system will offer real time ride matching services, thus the carpool driver has the

right to switch the route to avoid passing through toll which may require more time to reach the destination.

3.4 PROPOSED DESIGN

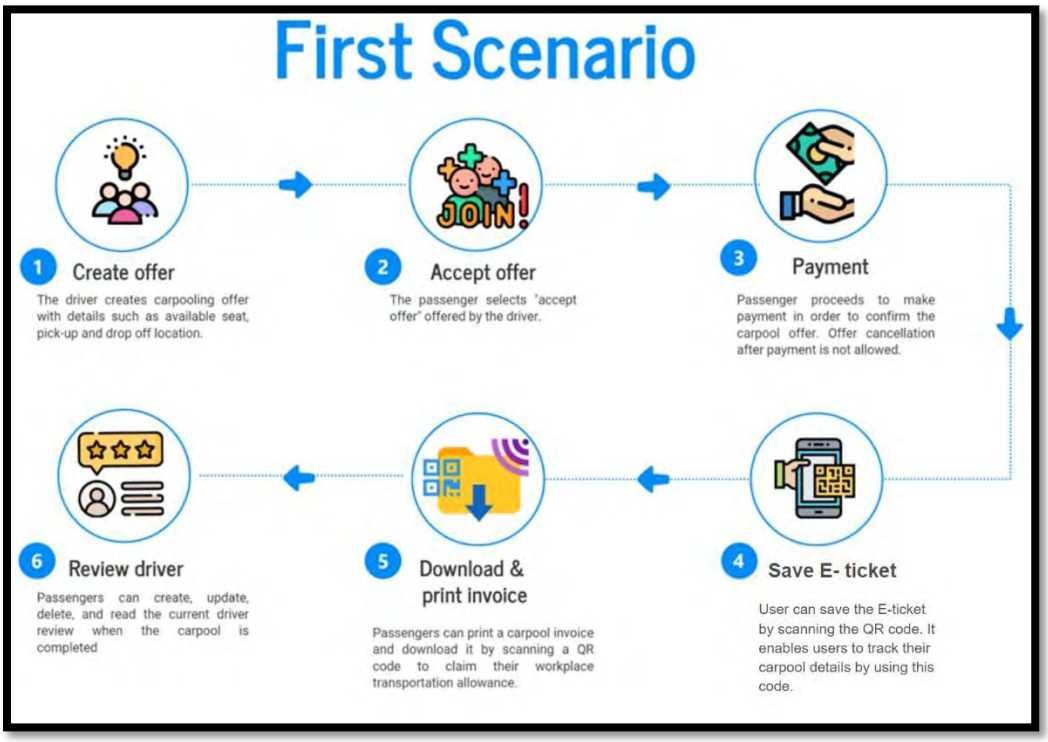


Figure 3.4 Simplified Flowchart for the First Scenario - Initiated by the Driver

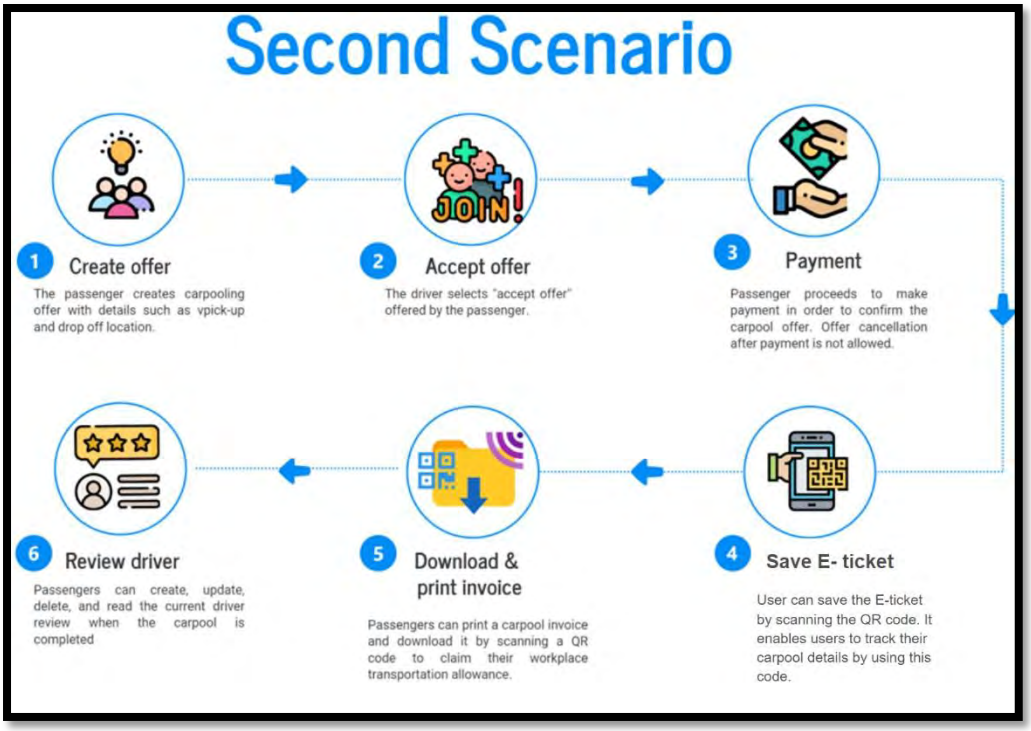


Figure 3.5 Simplified Flowchart for the Second Scenario - Initiated by the Passenger

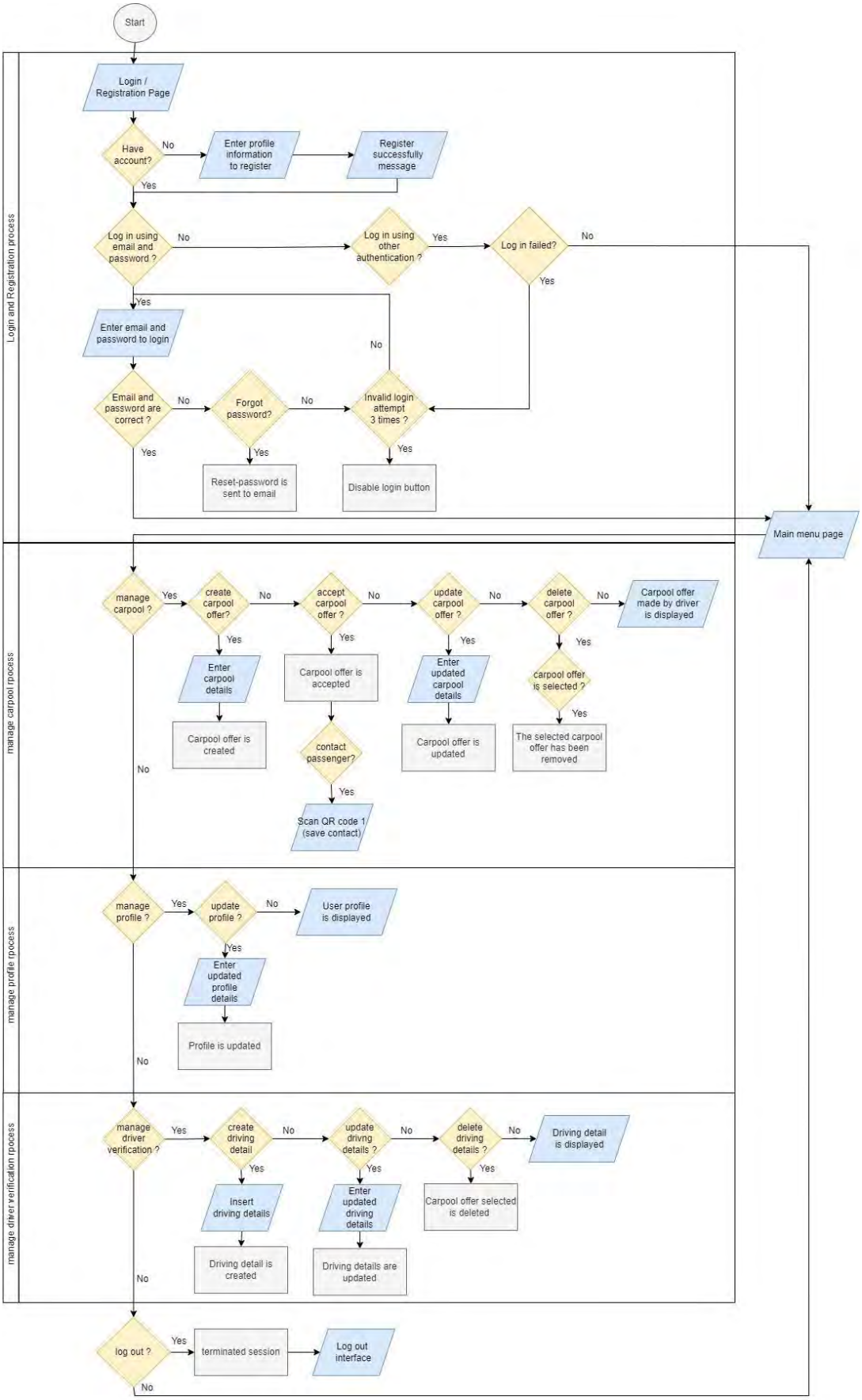


Figure 3.6 General Flowchart of Driver

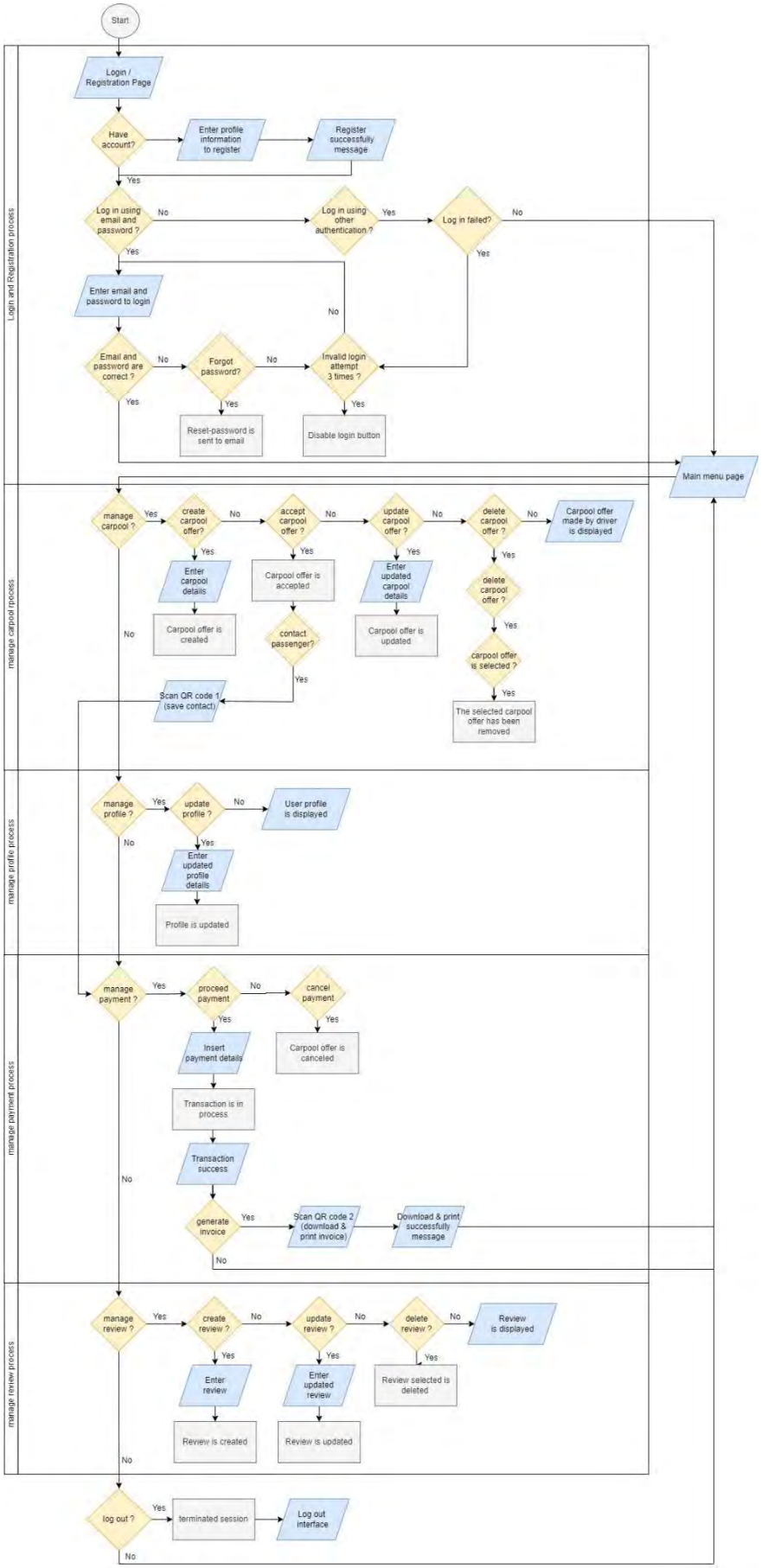


Figure 3.7 General Flowchart of Passenger

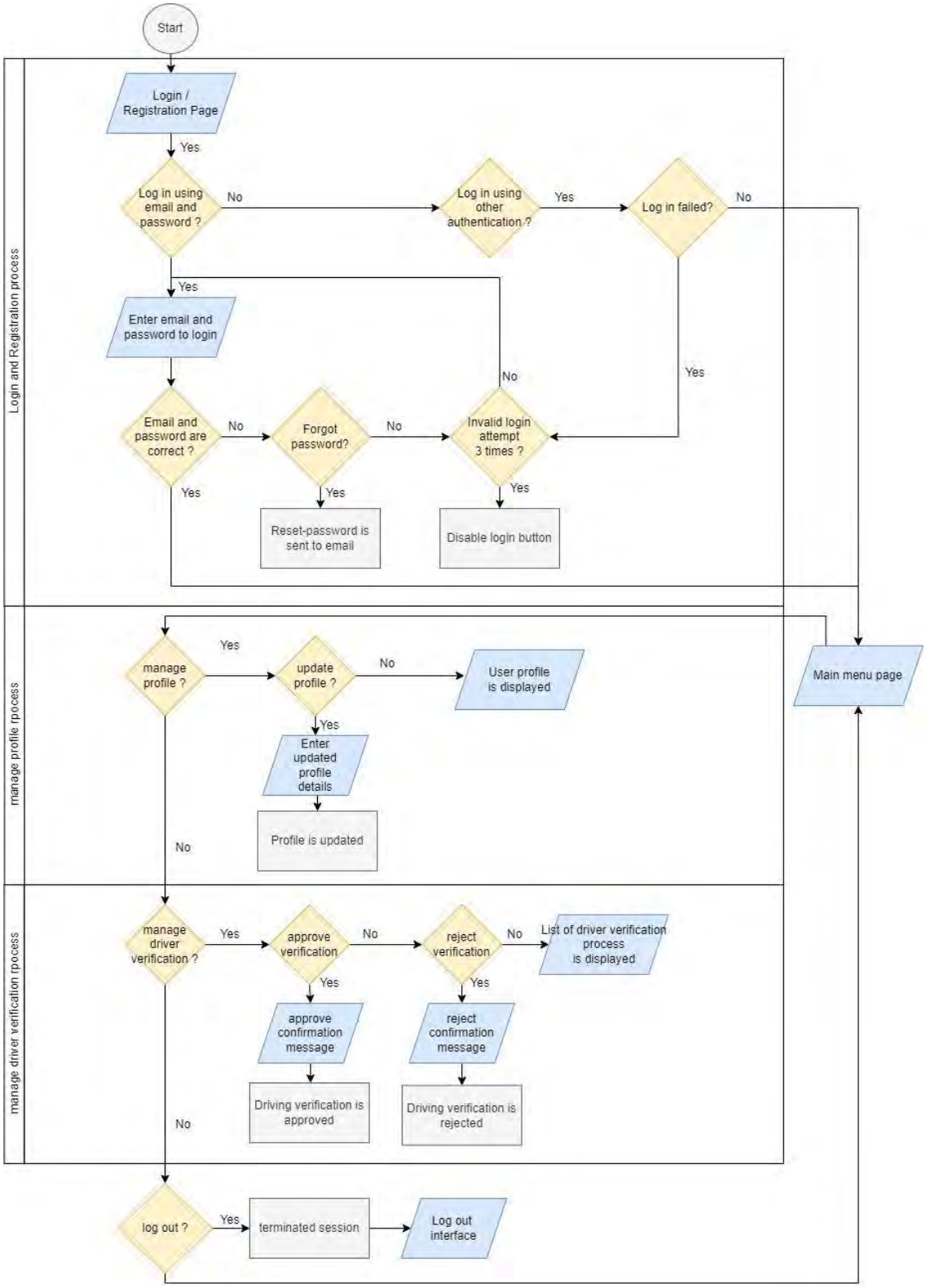


Figure 3.8 General Flowchart of Admin

The UMPool web-based application was designed to address the issues discussed in Chapter 1.2 and to compensate for shortcomings of existing systems discussed in Chapter 2. UMP staff and students who are target users of this application have the option to be either drivers or passengers. There are three (3) roles in this application which are admin, driver, and passenger. Each role of a user will have different access privileges to the system as illustrated in Table 3.1.

In general, the system contains six modules, with the driver having access to four of them, admin has access to three while the passenger has access to five. These six modules are manage user login, manage profile, manage carpool, manage payment, manage review, and manage driving verification. Figures 3.4 and 3.5 represent simplified flowcharts for drivers and passengers, respectively. Both simplified flowcharts depict a situation that has been initiated by distinct users.

The majority of business opportunities are driven by demand, which is why passengers can create carpooling offer for drivers to accept. In accordance with the laws of market equilibrium, the supply will increase if the demand increases. A driver can profit from providing that service if the demand is high for that specific drop off location.

Table 3.1 Access privileges of users

Module	**Description**	**User**
Manage user login	• Users can choose login by email and password. Users without an account must register and then proceed to login.	• Driver and passenger • Admin (Only involve in login, but not registration)
Manage profile	• Users can update, and view their profile	All users
Manage carpool	Driver • Users can create a carpool offer, accept	Driver and passenger

	a carpool offer made by passengers, update, delete the carpool offer and view the carpool offer details. Passenger • Users can create a carpool offer, accept a carpool offer made by the driver, update the carpool offer, delete the carpool offer and view the carpool offer details. • This module also displays the carpool listing, users are allowed to search based on their preference.	
Manage payment	• Users can cancel their carpool offer and proceed to payment. • Two-way payment methods are applied which are Cash and Stripe	Passenger
Manage review	Passenger • Users can create, update, view and delete reviews when the carpool is completed. Driver • Users can view the passengers' review when the carpool is completed.	Passenger and driver
Manage driving verification	Driver • Users can create, update, view and delete their driving details including driving license and driving period.	Admin and driver

	Admin • The admin will approve or reject the driver's driving license. Following that, the admin can view the list of driving verification applications.	

3.4.1 Context Diagram

The context diagram illustrates the interactions between a system and its context, including the admin, driver, and passenger, illustrated in the Figure 3.9.

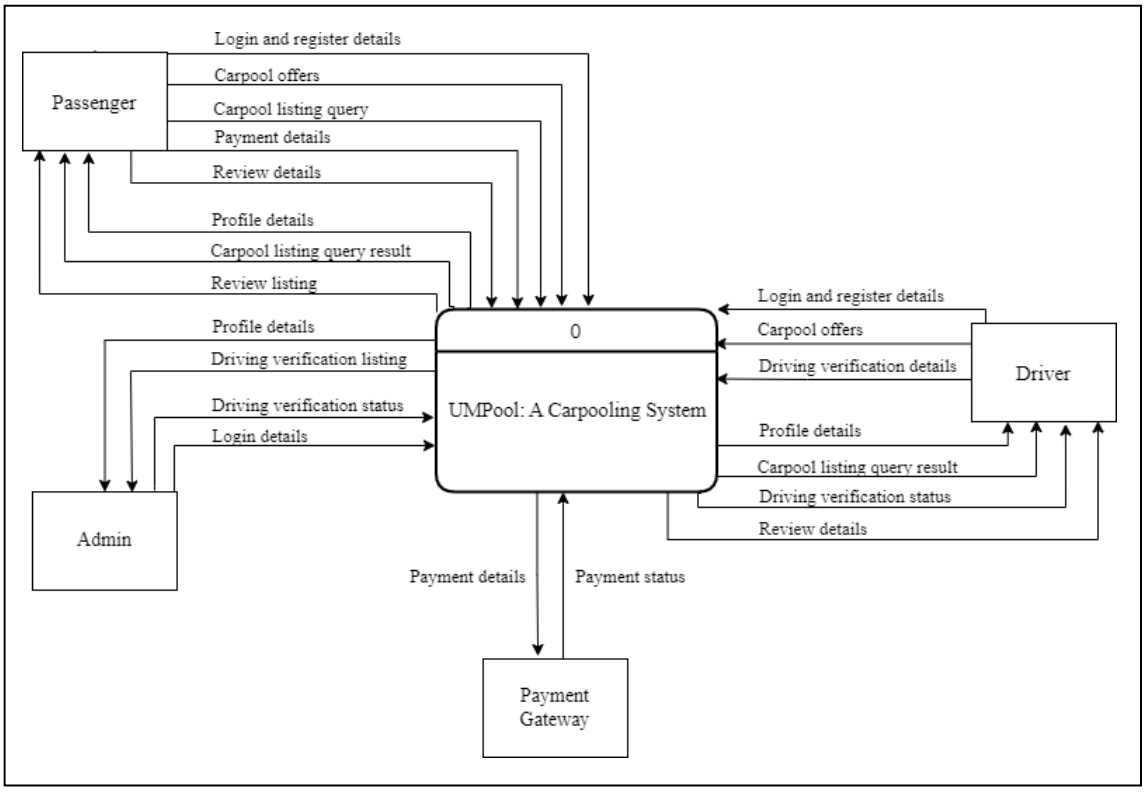


Figure 3.9 Context diagram of UMPool: A Carpooling System

3.4.2 Use Case Diagram

There are three (3) main actors which are admin, driver, and passenger. There are six (6) modules in the system which consist of manage user login, manage profile, manage carpool, manage payment, manage review, and manage driving verification, as illustrated in the Figure 3.10.

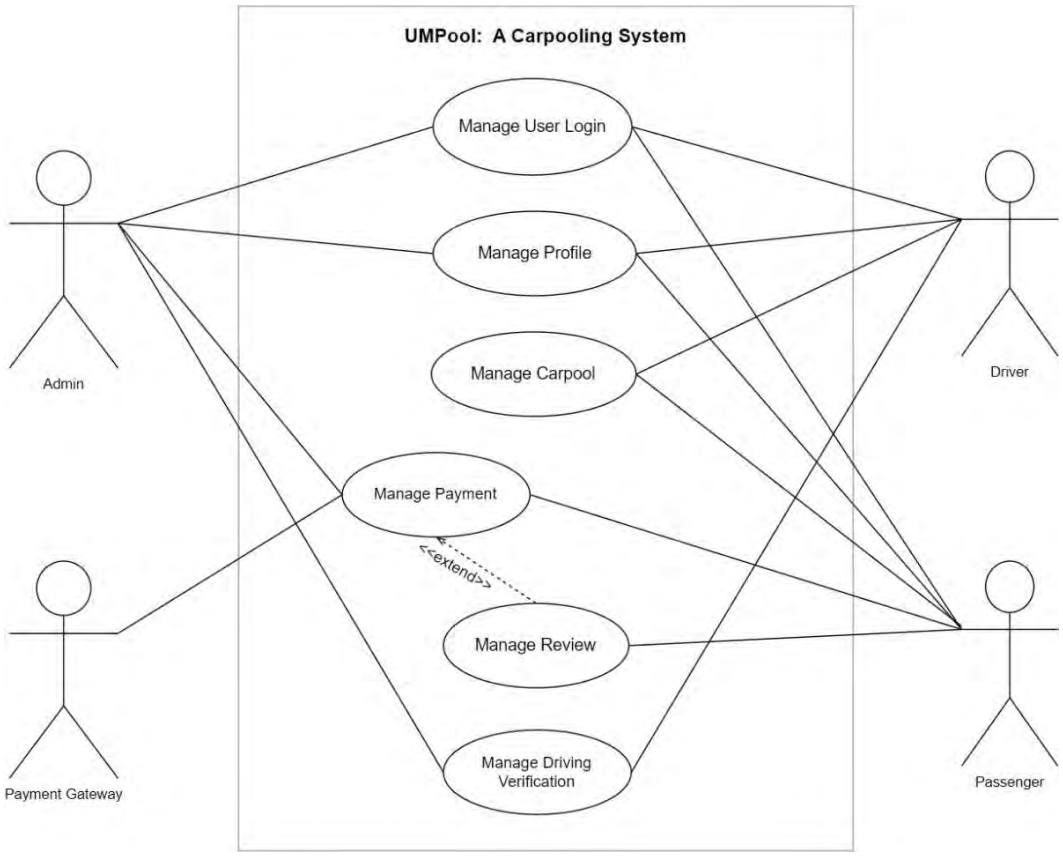


Figure 3.10 Use Case Diagram of UMPool: A Carpooling System

3.4.2.1 Manage User Login

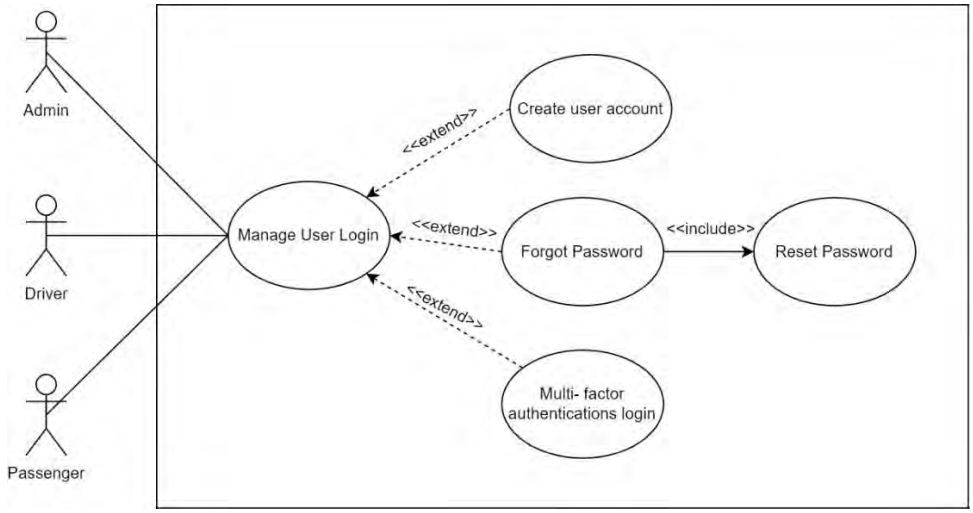


Figure 3.11 User case diagram of the Manage User Login

Table 3.2 Use Case Description of the Manage User Login

Use Case ID	UACS_UC01
Use Case Name	Manage user login
Brief Description	This use case explains the user login process for drivers and passengers. Next, the use case describes how an admin logs into their account. With this use case, users are allowed to use the functionality after login to the system.
Actor	Admin, driver, and passenger
Pre-condition	The server is working normally. Admin have registered for an account.
Basic Flow	1. The use case starts with user login page. [A1: Create user account]

	2. User enters their account credentials. (email and password) to log in their account. [A2: Forgot password] 3. System will begin to account verification process in the database. 4. System redirects to main menu page. 5. The use case end.
Alternative Flow	[A1: Create user account] 1. Users select <<Register>> button to register their account. 2. System redirects to the register page. 3. Users enter register details and select <<Register>> button. 4. The use case continues to step 1 in the basic flow. [A2: Forgot password] 1. Users select <<forgot your password>> hyperlink. 2. System redirects to the reset password page. 3. Users are required to fill in email address and click “Send password reset link” button. 4. The system will generate a reset password verification email and send it to the email address entered.

	5. Users are required to fill in the new password and submit it. 6. The use case continues to the step 1 in the main flow.
Exception Flow	[E1: Invalid password] 1. The system will reject user login. 2. The system will pop out an error message. 3. User is asked to try again their password or select the forgot password option of the login screen. 4. The use case end.
Post Condition	User is logged in to the system and the system displays the user dashboard page.

3.4.2.2 Manage Profile

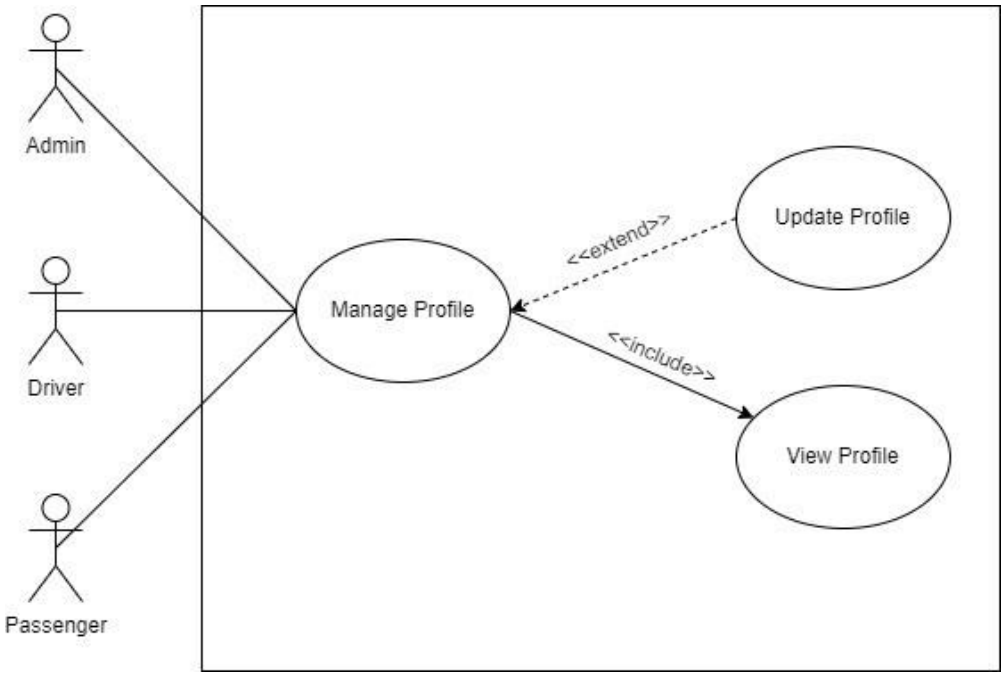


Figure 3.12 User case diagram of the Manage Profile

Table 3.3 Use Case Description of the Manage Profile

Use Case ID	UACS_UC02
Use Case Name	Manage profile
Brief Description	This use case explains the process of manage profile for admin, driver, and passenger. It aims at allowing users to make changes to their profile information as needed. The latest profile is viewable after they make changes to their profiles.
Actor	Admin, driver, and passenger
Pre-condition	User logs in to their account successfully.
Basic Flow	1. This use case begins when the user navigates to the profile page.

	2. System displays the list of user information. 3. Users select the specific user information and select operation. [A1 : Update profile] 4. The system display the latest user profile. 5. The use case end.
Alternative Flow	[A1 : Update profile] 1. Users select update button following the selection of the information. 2. System display the update profile page. 3. Users fill the text entry box with the updated information. 4. Users select the save button to update their profile. 5. The use case continues to the step 4 in the main flow.
Exception Flow	-
Post Condition	Latest profile is displayed.

3.4.2.3 Manage Carpool

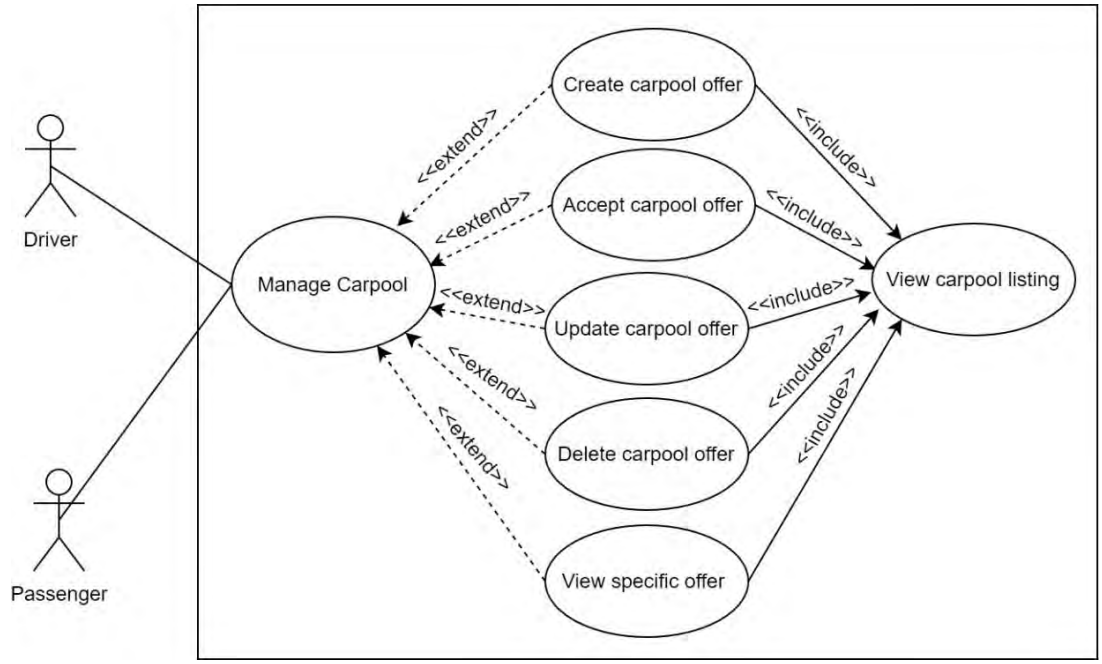


Figure 3.13 Use case diagram for Manage Carpool

Table 3.4 Use case description for Manage Carpool

Use Case ID	UACS_UC03
Use Case Name	Manage Carpool
Brief Description	This use case explains the process of manage carpool for driver, and passenger. It aims at allowing users to make changes on the carpool offer as needed. In this case, both users can create the carpool offer and wait for others to accept it. As an example, passengers can create an offer based on their requirements and wait for drivers to accept that offer.
Actor	Driver and passenger
Pre-condition	User logs in to their account successfully.

Basic Flow	[Driver] 1. This use case begins when the user navigates to the manage carpool page. 2. System displays the carpool listing. [A1: Create carpool offer] [A2: Accept carpool offer] [A3: Update carpool offer] [A4: Delete carpool offer] [A5: View specific offer] 3. The system displays the latest carpool listing. 4. The use case end. [Passenger] 1. This use case begins when the user navigates to the manage carpool page. 2. System displays the carpool listing. [A1: Create carpool offer] [A2: Accept carpool offer] [A3: Update carpool offer] [A4: Delete carpool offer] [A5: View specific offer] 3. The system displays the latest carpool listing. 4. The use case end.
Alternative Flow	[A1 : Create carpool offer] 1. Users select create button on the manage carpool page. 2. System display the create carpool page. 3. Users fill the text entry box with the carpool offer information. 4. Users select the add button to create their carpool

	offer. 5. The use case continues to the step 3 in the main flow. [A2 : Accept carpool offer] 1. Users select accept button following the selection of the carpool listings. 2. The system displays a message to confirm the user wants to accept this offer. 3. Users click on the confirm button to accept an offer from another user role. 4. System display accept offer successfully message. 5. The use case continues to the step 3 in the main flow. [A3 : Update carpool offer] 1. Users select update button following the selection of the carpool listings. 2. System display the update carpool offer page. 3. Users fill the text entry box with the updated information. 4. Users select the save button to update their carpool offer. 5. System display update carpool offer successfully message. 6. The use case continues to the step 3 in the main

	flow. [A4 : Delete carpool offer] 1. Users select delete button following the selection of the information. 2. The system displays a alert message to confirm the user wants to delete this offer. 3. Users select the confirm button to delete the carpool offer. 4. System displays the deleted successful message. 5. The use case continues to the step 3 in the main flow. [A5 : View specific offer] 1. Users select view button following the selection of the information. 2. The system displays the carpool offer details for the selected carpool. 3. Users select the back button to redirect back to carpool listing. 4. The use case continues to the step 3 in the main flow.
Exception Flow	-
Post Condition	Latest carpool offers status are displayed.

3.4.2.4 Manage Payment

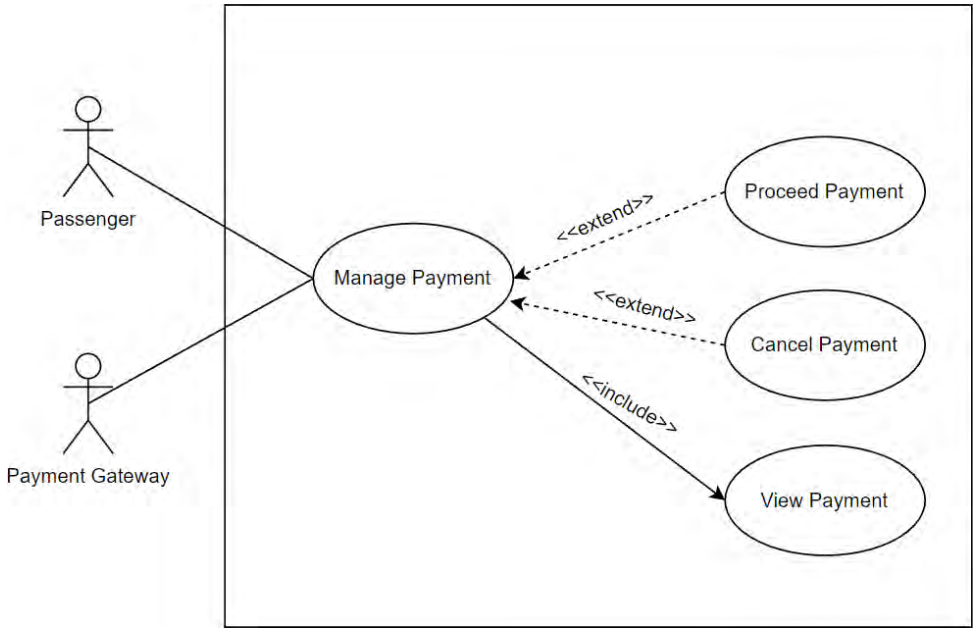


Figure 3.14 Use case diagram for Manage Payment

Table 3.5 Use case description for Manage Payment

Use Case ID	UACS_UC04
Use Case Name	Manage Payment
Brief Description	This use case explains the process of manage payment for passenger. It aims at allowing users to make changes on payment. User can view the latest payment after the driver has accepted the carpool offer or after users have accepted the deal at the carpool listing. Users cannot cancel / refund their payment after they have completed payment.
Actor	Passenger
Pre-condition	1. User logs in to their account successfully.

	2. Passengers accept the carpool offer.
Basic Flow	1. This use case begins when the user navigates to the manage payment page. 2. System displays the view payment page. [A1: Accept payment] [A2: Cancel payment] 3. System displays the latest payment status. 4. The use case end.
Alternative Flow	[A1: Accept payment] 1. Users select <<accept>> button at the view payment page. 2. System displays accept payment page. 3. Users fill the payment details and click <<pay>> button. 4. System validate the payment. [E1: Payment failed] 5. System display payment successfully message. 6. The use case continues to the step 3 in the main flow. [A2: Cancel payment] 1. Users select <<cancel>> button at the view payment page. 2. System display confirmation cancel payment message. 3. Users click <<confirm>> button to cancel

	payment. 4. System display the cancel payment successfully message. 5. The use case continues to the step 3 in the main flow.
Exception Flow	[Payment failed] 1. System displays error message. 2. The user is asked to try again to make payment. 3. The use case end.
Post Condition	The latest payment status is updated and displayed.

3.4.2.5 Manage Review

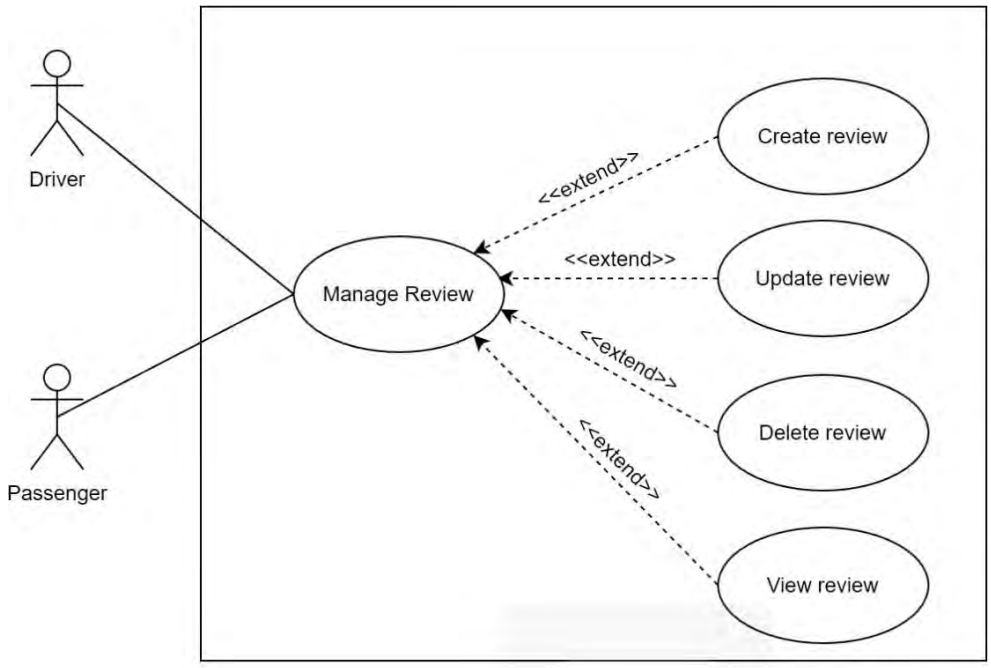


Figure 3.15 Use case diagram for Manage Review

Table 3.6 Use case description for Manage Review

Use Case ID	UACS_UC05
Use Case Name	Manage Review
Brief Description	This use case explains the process of manage review for passenger. It aims at allowing users to make changes to their review as needed. Once the user has made changes, the latest list of reviews will be updated and displayed.
Actor	Passenger
Pre-condition	1. User logs in to their account successfully. 2. The passenger has made payment and the carpool offer is completed.
Basic Flow	1. This use case begins when the user navigates to the manage review page. 2. System displays the review listing. [A1: create review] [A2: update review] [A3: delete review] [A4: view review] 3. System displays the latest review listing. 4. The use case end.
Alternative Flow	[A1: Create review] 1. Users select create button on the manage review page. 2. System display the create review page. 3. Users fill the text entry box with the review of carpool offer.

	4. Users select the add button to create their review. 5. The use case continues to the step 3 in the main flow. [A2: Update review] 1. Users select update button following the selection of the review list. 2. System display the update review page. 3. Users fill the text entry box with the updated information. 4. Users select the save button to update their review. 5. System display update review successfully message. 6. The use case continues to the step 3 in the main flow. [A3: Delete review] 1. Users select delete button following the selection of the information. 2. The system displays a alert message to confirm the user wants to delete this review. 3. Users select the confirm button to delete the review. 4. System displays the deleted successful message. 5. The use case continues to the step 3 in the main flow.

	[A4: View review] 1. Users select view button following the selection of the information. 2. The system displays the review details for the selected review. 3. Users select the back button to redirect back to review listing. 4. The use case continues to the step 3 in the main flow.
Exception Flow	-
Post Condition	Latest review list is displayed.

3.4.2.6 Manage Driving Verification

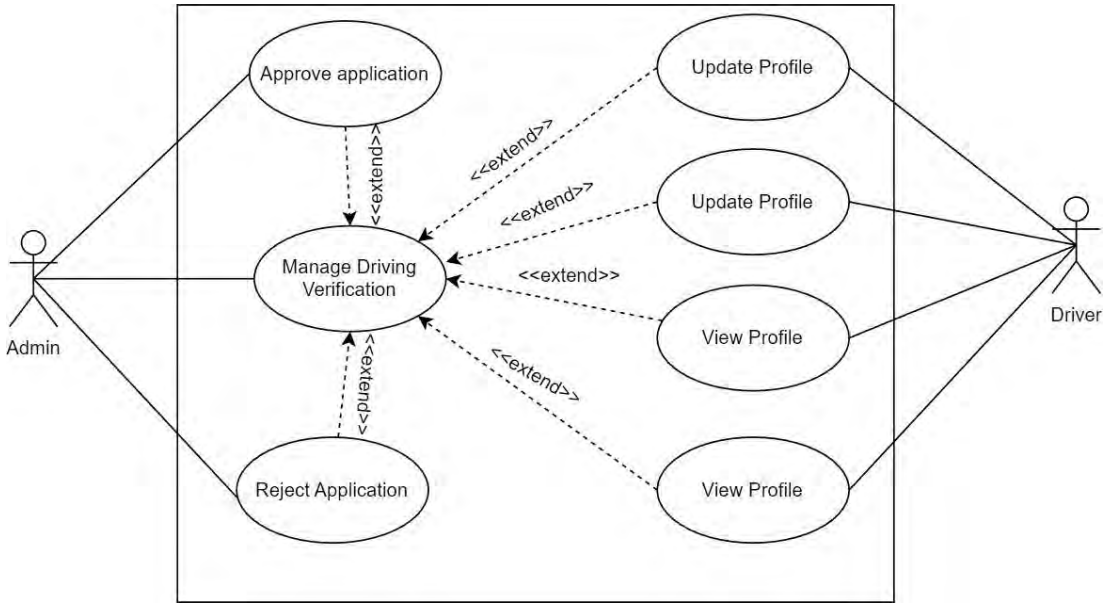


Figure 3.16 Use Case Diagram for Manage Driving Verification

Table 3.7 Use Case Description for Manage Driving Verification

Use Case ID	UACS_UC06
Use Case Name	Manage Driving Verification
Brief Description	This use case explains the process of manage driving verification for admin and driver. It aims at allowing driver to make changes to their driving verification. Admin will verify the driver application and approve it. The admin can reject the application if any misauthentication issue is detected.
Actor	Admin and driver
Pre-condition	User logs in to their account successfully.
Basic Flow	1. This use case begins when the user navigates to the manage driving verification page. 2. System displays the manage driving verification

	page. [A1: Create driving details] [A2: Update driving details] [A3: Delete driving details] [A4: Approve application] [A5: Reject application] 3. System displays the latest driving verification status. 4. The use case end.
Alternative Flow	[A1: Create driving details] 1. Driver selects <<create>> button on the manage driving verification page. 2. System display the create driver details page. 3. Driver fills the text entry box with the driving details. 4. Driver selects the <<add>> button to create their driving details. 5. The use case continues to the step 3 in the main flow. [A2: Update driving details] 1. Driver selects <<update>> button following the selection of the driving details list. 2. System displays the update driving details page. 3. Driver fills the text entry box with the updated information. 4. Driver selects the <<save>> button to update their driving details. 5. System displays update driving details

	successfully message. [A3: Delete driving details] 1. Driver selects <<delete>> button following the selection of the information. 2. The system displays an alert message to confirm the user wants to delete this driving details. 3. Driver selects the <<confirm>> button to delete the driving details. 4. System displays the deleted successful message. 5. The use case continues to the step 3 in the main flow. [A4: Approve application] 1. Admin select and view the driver driving details. 2. Admin selects <<approve>> button to approve the application. 3. The use case continues to the step 3 in the main flow. [A5: Reject application] 1. Admin select and view the driver driving details. 2. Admin selects <<reject>> button to reject the application. 3. The use case continues to the step 3 in the main flow.
Exception Flow	-

Post Condition	Latest driving status is displayed.

3.4.3 Activity Diagram

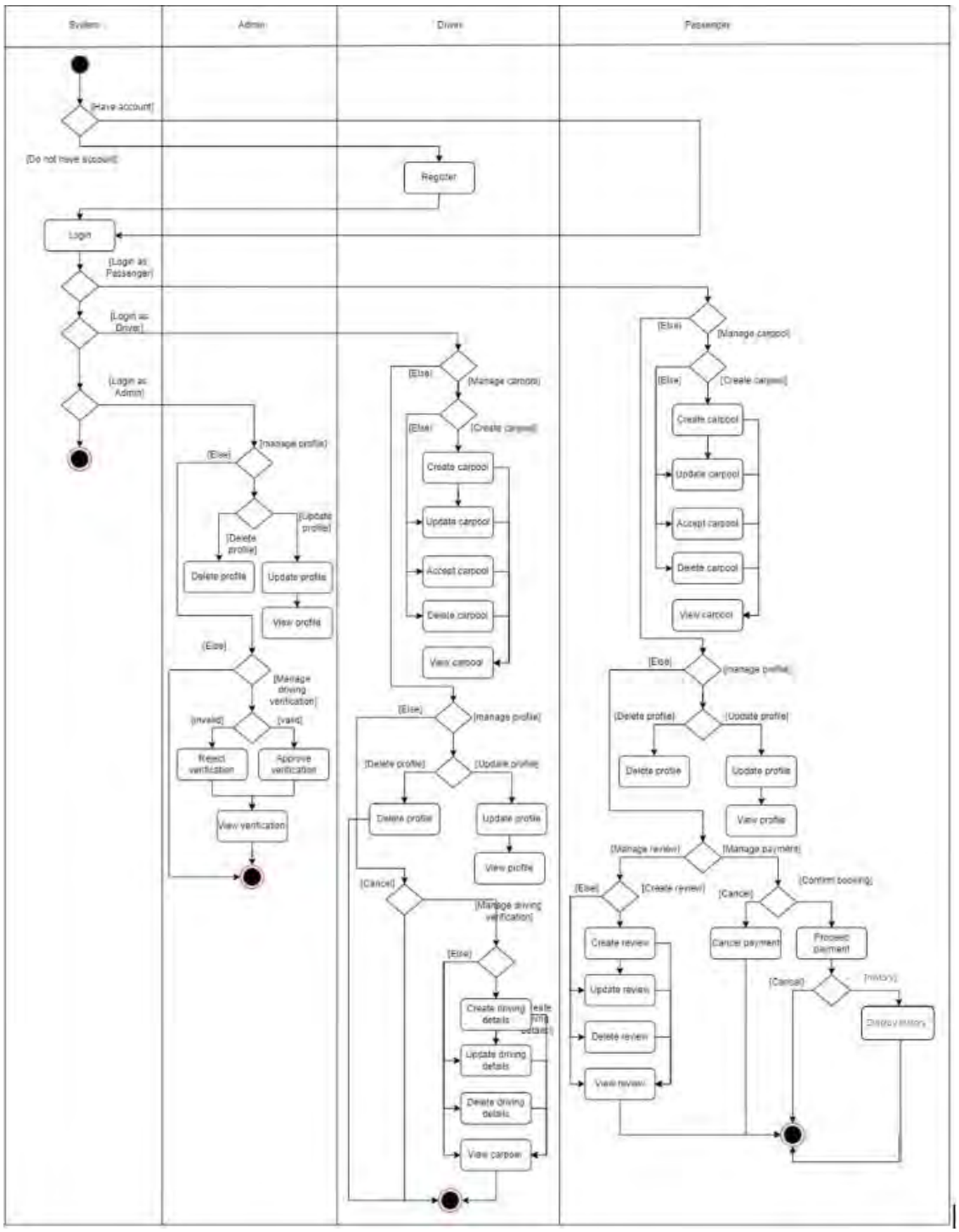


Figure 3.17 Activity Diagram of UMPool: A Carpooling System

3.4.4 Development Framework

Laravel framework is applied in this proposed UMPool: A Carpooling System. Laravel uses model-view-controller to facilitate the development of web applications. The workflow of each element within the Laravel framework is illustrated in Figure 5.14. Controller is responsible for handling user requests and retrieving data from models. Data model allows users to retrieve information from the database about objects. Views render the user interface in the browser.

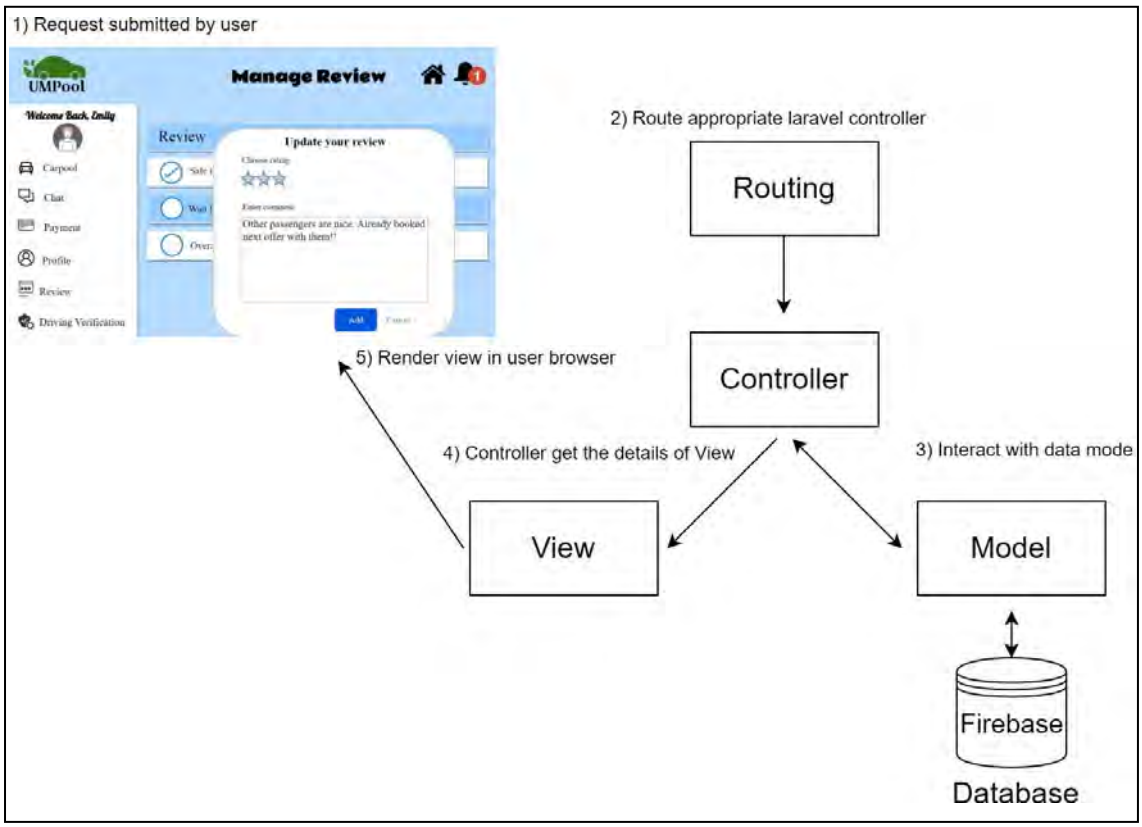


Figure 3.18 Architecture of Laravel framework

3.5 DATA DESIGN

3.5.1 ERD

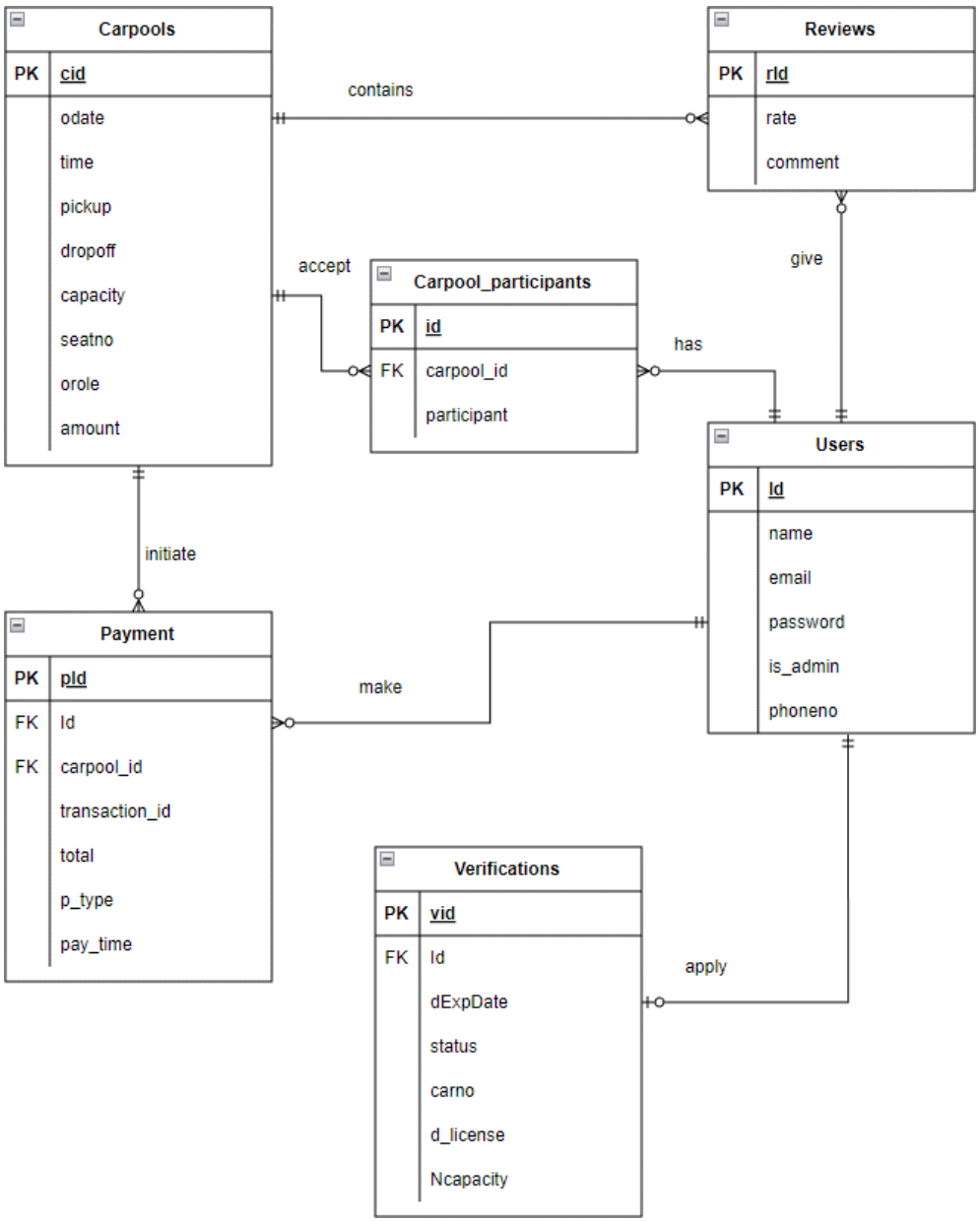


Figure 3.19 ERD Diagram

3.5.2 Data Model

Users

Table 3.8 Data Dictionary of User

Data Name	Data Type	Description	Module	Constraint
Id	Bigint(20)	User identifier	Manage User Login Manage Profile Manage Driving Verification	PK
name	varchar(255)	User Name	Manage User Login	-
email	varchar(255)	Email of user	Manage User Login Manage Profile	-
password	varchar(255)	User password	Manage User Login	-
is_admin	Integer	Role of user	Manage User Login	-
phoneno	varchar(255)	Contact information	Manage User Login Manage Profile	-

Carpools

Table 3.9 Data Dictionary of Carpools

Data Name	Data Type	Description	Module	Constraint
cid	Bigint(20)	Offer identifier	Manage Carpool	PK
odate	varchar(255)	Offer date	Manage Carpool	-
time	varchar(255)	Time to pick up (24-hour clock)	Manage Carpool	-
pickup	varchar(255)	Pick up location	Manage Carpool	-
dropoff	varchar(255)	Drop off location	Manage Carpool	-
capacity	int	Total number of seat	Manage Carpool	-
seatno	varchar(255)	Number of seat reserved	Manage Carpool	-
orole	varchar(255)	Role of offer	Manage Carpool	-

amount	double	Carpool amount of each passenger	Manage Carpool	-

Carpool Participant

Table 3.10 Data Dictionary of Carpool Participant

Data Name	Data Type	Description	Module	Constraint
id	bigint(20)	Carpool Participant identifier	Manage Carpool	PK
carpool_id	bigint(20)	Carpool identifier	Manage Carpool	-
participant	bigint(20)	User identifier	Manage Carpool	-

Payment

Table 3.11 Data Dictionary of Payment

Data Name	Data Type	Description	Module	Constraint
I	bigInt(20)	Payment identifier	Manage Payment	PK
User_id	bigInt(20)	User identifier	Manage	FK

			Payment Manage User Login	
carpool_id	bigInt(20)	Carpool identifier	Manage Payment Manage Carpool	FK
transaction_id	varchar(255)	Unique identifiers for each transaction	Manage Payment	-
pay_time	Date time	Payment date and time	Manage Payment	-
p_type	varchar(10)	Payment method	Manage Payment	-
total	double	Total split per passenger in carpool	Manage Payment Manage Carpool	-

Reviews

Table 3.12 Data Dictionary of Review

Data Name	Data Type	Description	Module	Constraint
rId	Bigint(20)	Review Identifier	Manage Review	PK

rate	Varchar(255)	Rating	Manage Review	-
comment	varchar(255)	Comment of Carpool	Manage Review	-

Verifications

Table 3.13 Data Dictionary of Verifications

Data Name	Data Type	Description	Module	Constraint
vid	bigint(20)	Verification identifier	Manage Driving Verification	PK
user_id	bigint(20)	User identifier	Manage User Login Manage Profile Manage Driving Verification	FK
dExpDate	varchar(255)	Driving license expire date	Manage Driving Verification	-
status	varchar(255)	Status of verification	Manage Driving Verification	-
carno	varchar(255)	Number plate of car	Manage Driving Verification	-

Ncapacity	varchar(255)	Car capacity	Manage Driving Verification	-
d_license	varchar(255)	Car license	Manage Driving Verification	-

3.6 PROOF OF INITIAL CONCEPT

The prototype is sketched by Draw.io which attached in Appendix C, Chapter 3.1 Interface Design.

3.7 TESTING PLAN

The testing plan is conducted by tester to test the functionality of the system which illustrated in Table 3.15 (Appendix A) while the user acceptance testing will be distributed to stakeholders and end users such as admin, drivers, and passengers for user satisfaction and feedback as shown in Figures 5 to 14 (Appendix A). User acceptance testing (UAT) is a type of functional testing that is conducted to ensure that the product works for end users. UAT testing must be performed at the end of the SDLC, as all features are finished.

3.8 POTENTIAL USE OF PROPOSED SOLUTION

The proposed system allows users to manage their carpools online, eliminating the need to wait for buses and other transportation since UMP shuttle buses are limited. Due to the fact that no one was taking the bus during the Covid-19 outbreak in DHUAM and the demand for bus services is unstable as illustrated in Figure 3.21. Thus, bus services were suspended until further notice as illustrated in Figure 3.22. This kind of incidents can be prevented by providing this online carpool system which allowing student and staff to accept and cancel their carpool easily.

Moreover, the proposed system allows users to plan their travels in advance, thereby avoiding the difficulties of accepting carpool offers at peak hours. The proposed system enables users to plan their travels in advance, thus reducing the chance of an insufficient supply of carpool offers during peak hour. A confirmation accept carpool offer message will be sent after the payment is made and the available seats have been confirmed. A carpool may not be cancelled after payment has been made, and fees will not be refunded, unless unexpected conditions such as the absence of a driver and passengers make the carpool inoperable.

It is essential to check reviews and have a certified driver before accepting a carpool. The proposed system allows drivers to manage their driving verification, which is verified by admin. By carpooling with UMP students and staff, the proposed system will be more trusted, since potential risks such as scams and unlicensed drivers can be reduced.

JADUAL PERGERAKAN BAS ASRAMA DHUAM BAGI MINGGU PEPERIKSAAN

TARIKH	MASA BERTOLAK	TEMPAT		MASA TIBA	SHUTTLE BUS NO.	KAPASITI PENUMPANG	SUBJEK
		DARI	KE				
7-Oct	7:30	DHUAM	FKOM	8:00	BAS A (CBH 6036)	12 ORANG	Database (9:00 - 11:00) & Software Engineering (14:30 - 17:30)
	12:30	FKOM	DHUAM	13:00			
	13:00	DHUAM	FKOM	13:30			
	18:00	FKOM	DHUAM	18:30			
8-Oct	7:30	DHUAM	FKOM	8:00	BAS A (CBH 6036)	13 ORANG	Programming Techniques (9:00 - 12:00) & Data Communication Network (14:30 - 17:30)
	12:30	FKOM	DHUAM	13:00			
	13:00	DHUAM	FKOM	13:30			
	18:00	FKOM	DHUAM	18:30			
9-Oct	7:30	DHUAM	FKOM	8:00	BAS A (CBH 6036)	17 ORANG	Graphical User Interface (9:00 - 12:00) & (14:30 - 17:45)
	12:00	DHUAM	FKOM	12:30			
	14:30	FKOM	DHUAM	15:00			
	18:00	FKOM	DHUAM	18:30			
10-Oct	12:30	DHUAM	FKOM	13:00	BAS A (CBH 6036)	7 orang	Web Engineering (14:45 - 17:45)
	18:00	FKOM	DHUAM	18:30			
11-Oct	7:30	DHUAM	FKOM	8:00	BAS A (CBH 6036)	10 orang	Fundamental Discrete (9:00 - 12:00)
	12:30	FKOM	DHUAM	13:00			
12-Oct	7:30	DHUAM	FKOM	8:00	BAS B (CCH 1016)	6 orang	OOP & DPC (9:00 - 12:00)
	12:30	FKOM	DHUAM	13:00			
13-Oct	7:00	DHUAM	FKOM	7:30	BAS B (CCH 1016)	5 orang	3D Modelling (9:00 - 12:00)
	12:30	FKOM	DHUAM	13:00			
14-Oct	7:30	DHUAM	FKOM	8:00	BAS B (CCH 1016)	5 ORANG	Crytography (9:00 - 17:30) & Virtual Reality (14:30 - 17:30)
	12:30	FKOM	DHUAM	13:00			
	13:00	DHUAM	FKOM	13:30			
	18:00	FKOM	DHUAM	18:30			
15-Oct	7:30	DHUAM	FKOM	8:00	BAS B (CCH 1016)	5 ORANG	Operating System (9:00 - 12:00) & (14:30 - 17:30)
	12:30	FKOM	DHUAM	13:00			
	13:00	DHUAM	FKOM	13:30			
	18:00	FKOM	DHUAM	18:30			
16-Oct	12:00	DHUAM	FKOM	12:30	BAS B (CCH 1016)	3 orang	all (9:00 - 12:00)
	18:00	FKOM	DHUAM	18:30			

Figure 3.21 Unstable demand for UMP buses in October 2020

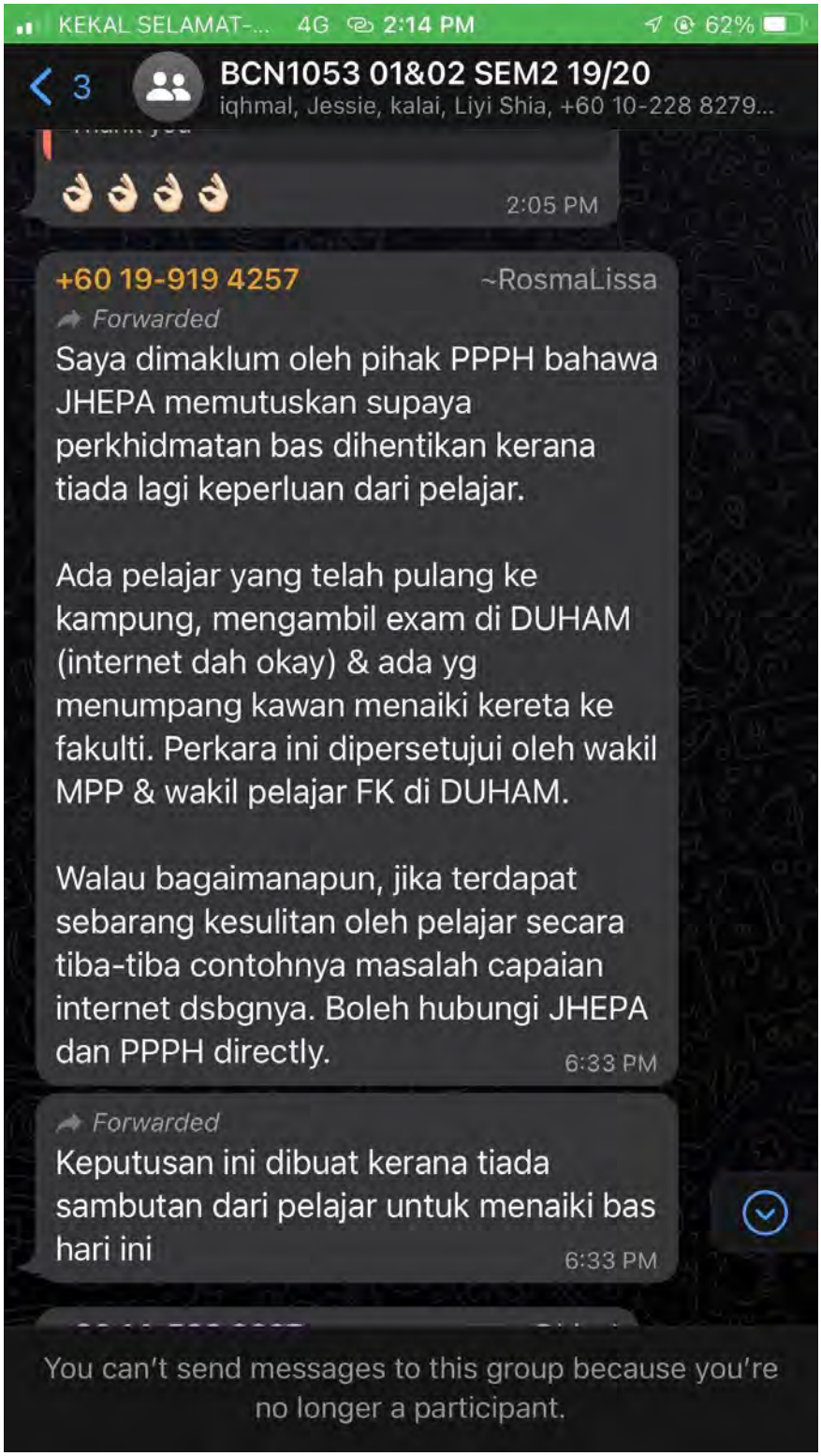


Figure 3.22 Suspension of UMP bus services announced on social media

3.9 SUMMARY

In a nutshell, chapter 3 explains the ways to implements the proposed project by using RAD model. The functional and non-functional requirements are outlined based on the existing system that is being studied. However, a google form survey will be conducted in order to gather user perceptions to make improvement based on the existing system. Hence, the results of the survey will be analysed and incorporated shortly into the function requirement, project flow chart, etc. to fulfil consumer needs as supply is dictated by demand. Functional, non-functional requirements, constraints, limitations, proposed design, data design, proof of the initial concept and testing plan of the proposed project are outlined in the chapter.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 INTRODUCTION

This chapter discusses the UACS system implementation process, including implementation, testing, and result analysis. Each function of the system will be described using a graphical user interface (GUI) in Section 4.3. UACS will be tested thoroughly with functional testing and user acceptance testing.

4.2 IMPLEMENTATION

This chapter outlines the implementation procedures. The development environment and functionality of the system will be discussed in detail in Section 4.2.1 and 4.2.2.

4.2.1 Development Environment

This application is implemented by Laravel framework, which consists of model, view, controller, and routes. In Laravel, routing allows users to route all requests to the respectively controller. The database used in this application is MySQL. MySQL is an open-source relational database management system which provide a great support in web applications.

4.2.2 System Functionality

4.2.3 Welcome

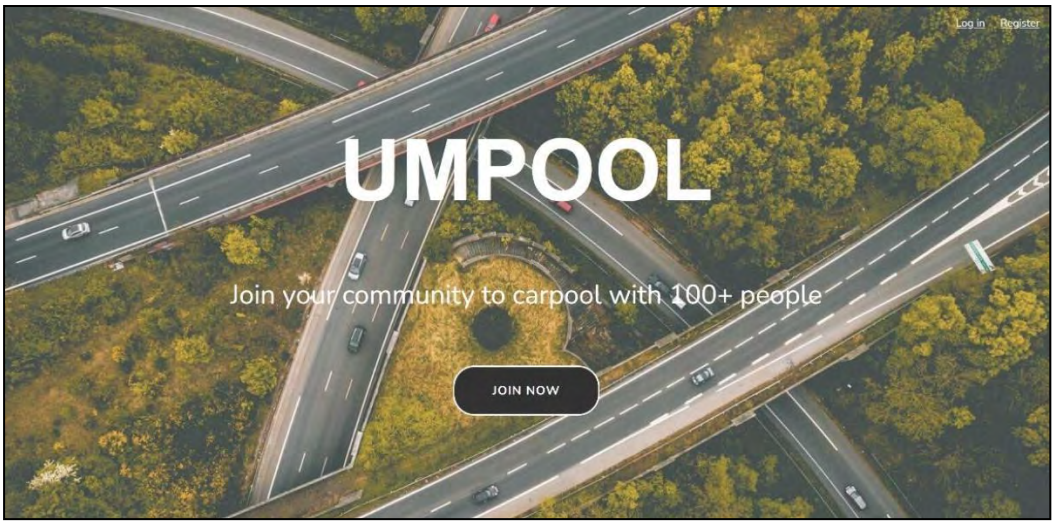


Figure 4.1 Welcome Page

Figure 4.1 shows the welcome page interface design of the UMPool system. Upon entering the system, users are taken to the welcome page, which allows them to register and log in.

4.2.4 Login

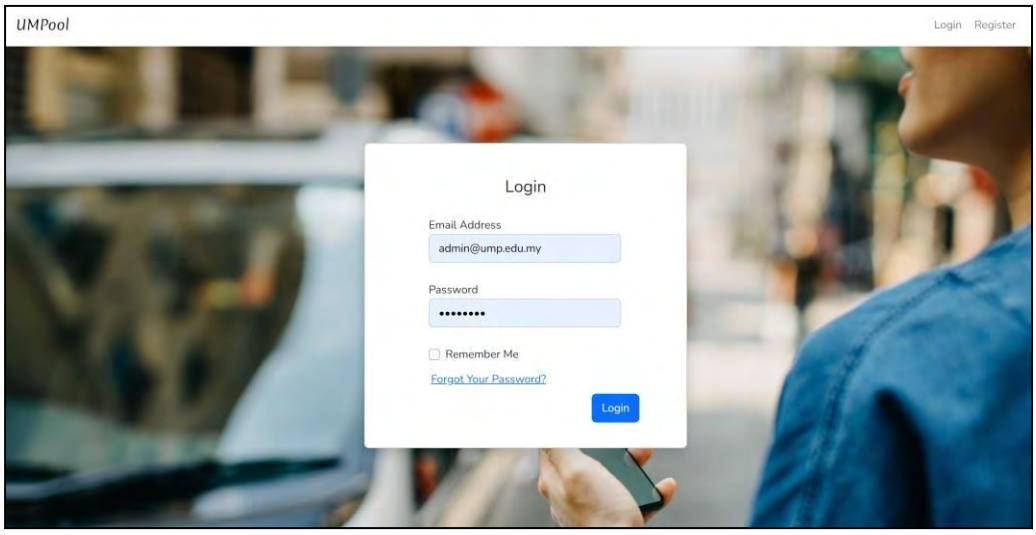


Figure 4.2 Login Page

Users required to login or register their account in order to access the system as illustrated in Figures 4.2 and 4.3. Figure 4.2 shows the login page which allows users to

login account by entering email address and password. Users can reset their passwords by selecting "Forgot Your Password" hyperlink, which redirects them to the reset password page as shown in Figure 4.4.

4.2.5 Register

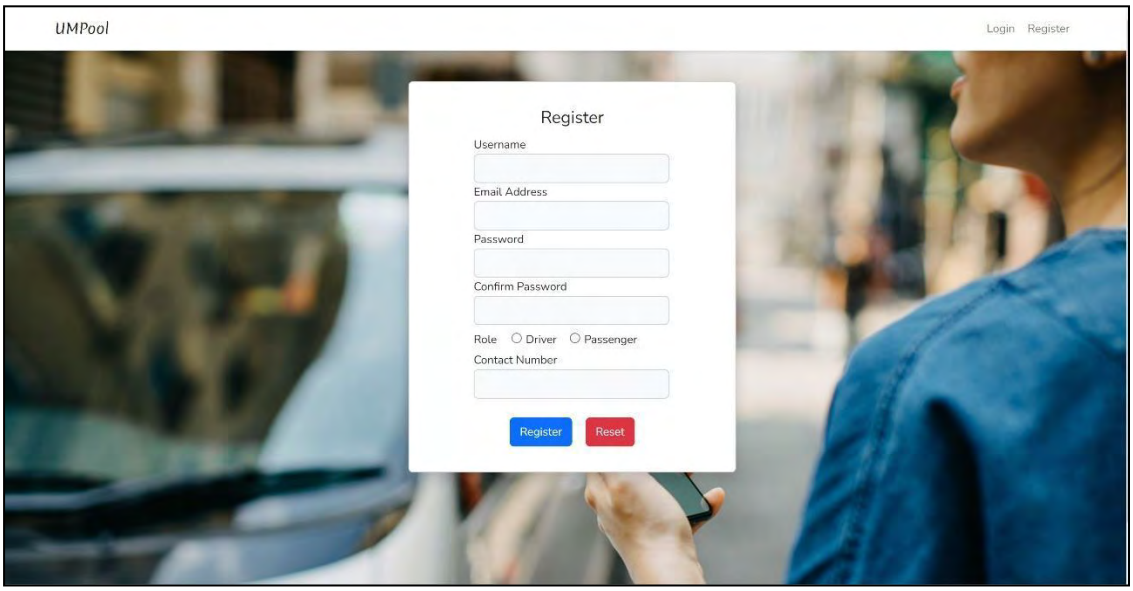


Figure 4.3 Register Page

Figure 4.3 shows the required user information for registering, including username, email address, password, confirmation password, role, and contact number.

4.2.6 Reset Password

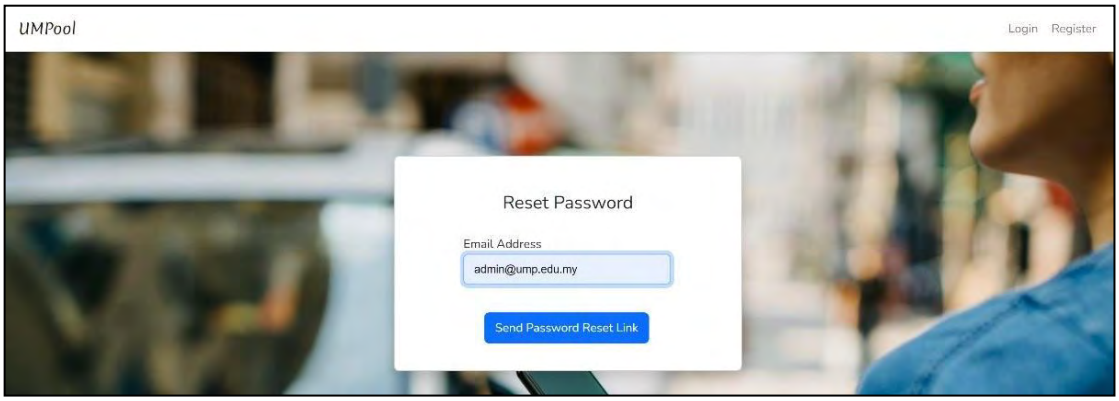


Figure 4.4 Reset Password Page

In the reset password page, user required to enter their email address to retrieve the reset password link and set new password.

4.2.7 Driver Dashboard

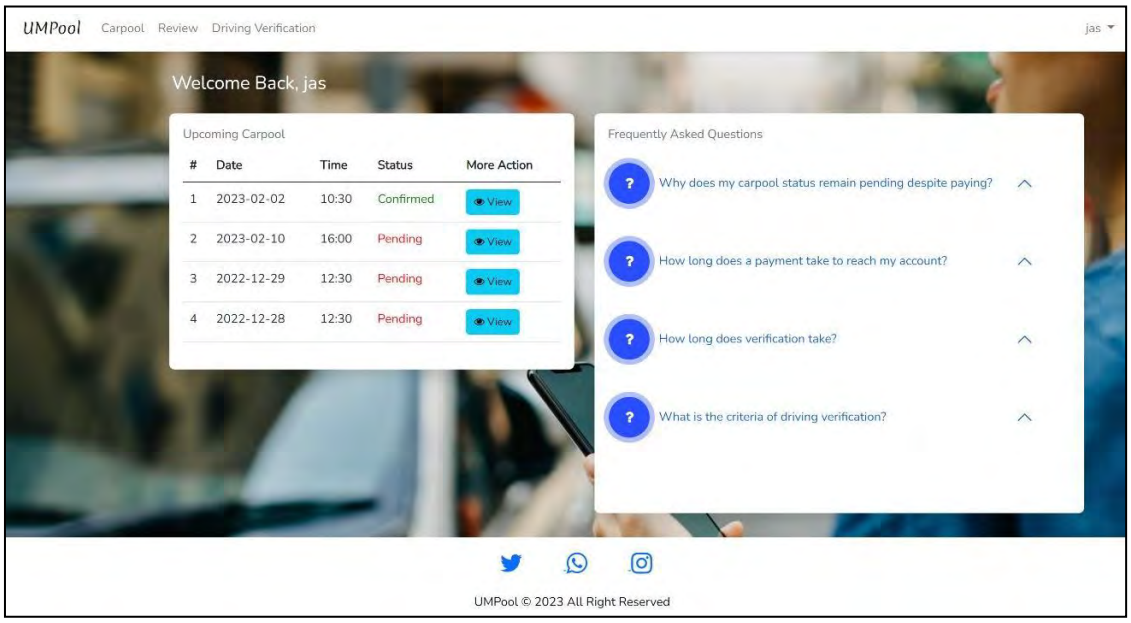


Figure 4.5 Driver Dashboard Page

Users with driver roles will be redirected to the driver dashboard, which has functions for view upcoming carpools, and FAQs. It is also allowing users to redirect to other functionality such as managing carpools, driving verification, and view passengers' review.

4.2.8 Passenger Dashboard

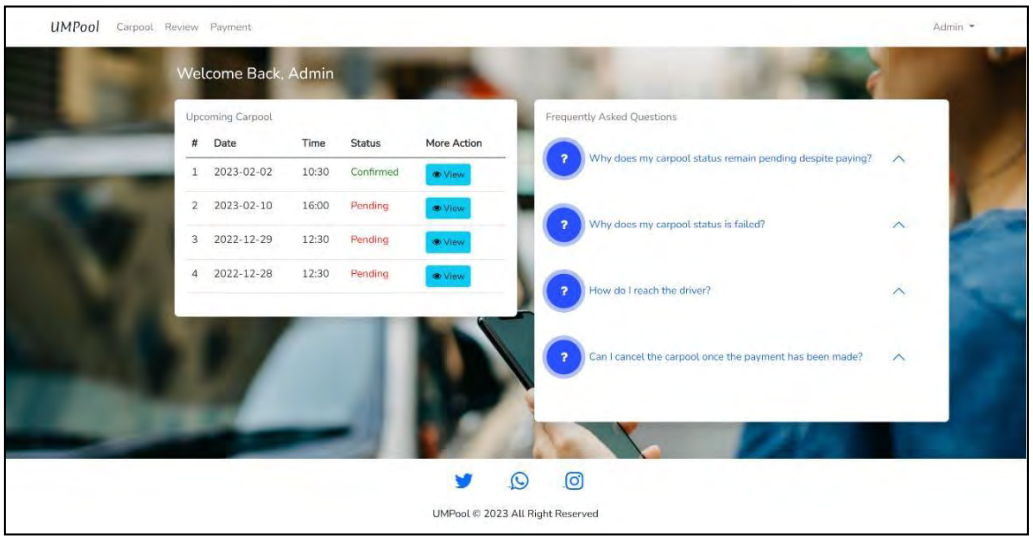


Figure 4.6 Passenger Dashboard Page

Users with passenger roles will be redirected to the passenger dashboard, which has functions for displaying upcoming carpools, and FAQs and enable users to manage carpools, reviews, payment, and profiles.

4.2.9 Admin Dashboard

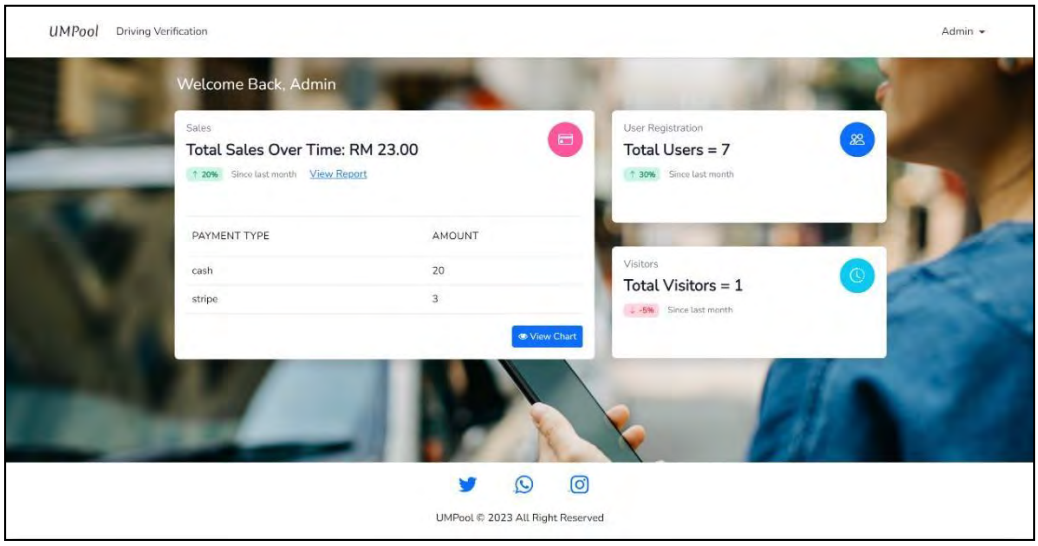


Figure 4.7 Admin Dashboard Page

Users with admin roles will be redirected to the admin dashboard, which has functions for displaying the sales, user registration, visitors and enables users to manage driving verification, and profiles.

Figures 4.5 to 4.7 illustrate how different users will access different dashboards. Users can access their profiles by clicking on their names in the upper right corner, and other functionality in the upper left corner.

4.2.10 Create Carpool

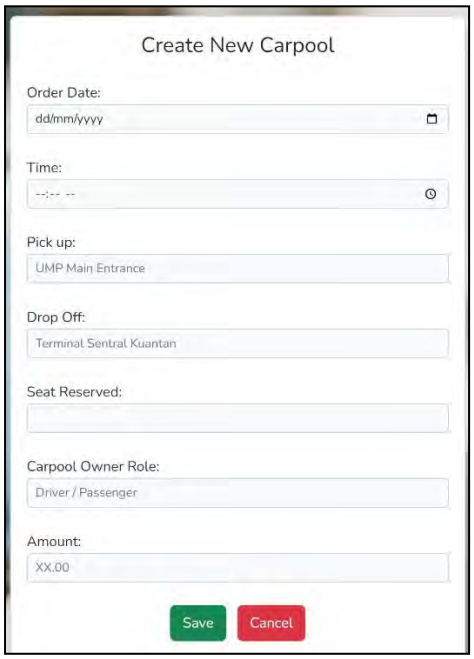


Figure 4.8 Create Carpool Page

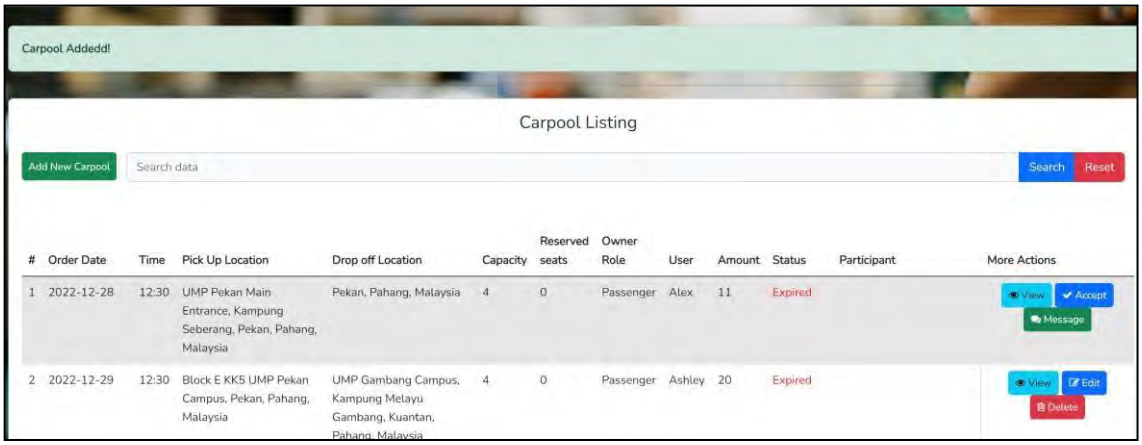


Figure 4.9 Create Carpool Successful Message

Passenger and driver can create the carpool offer by entering carpool details. The seat capacity is assigned to be 4 by default. Figures 4.8 and 4.9 illustrate how the create carpool successful message pops up when the user clicks on the "save" button.

4.2.11 Accept Carpool

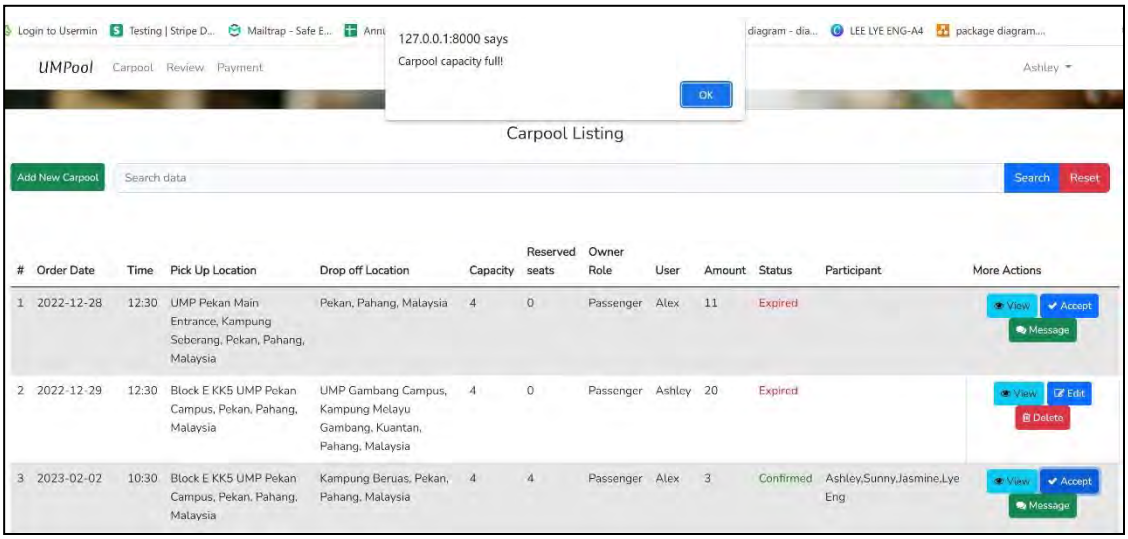


Figure 4.10 Refused to accept carpool

Drivers and passengers are also allowed to accept carpool offers created by others. Upon clicking "Accept" button, the reserved seat will be increase by one. An alert message will pop up “Carpool capacity full!” if the reserved seats exceed the maximum capacity as shown in Figure 4.10.

4.2.12 View Carpool

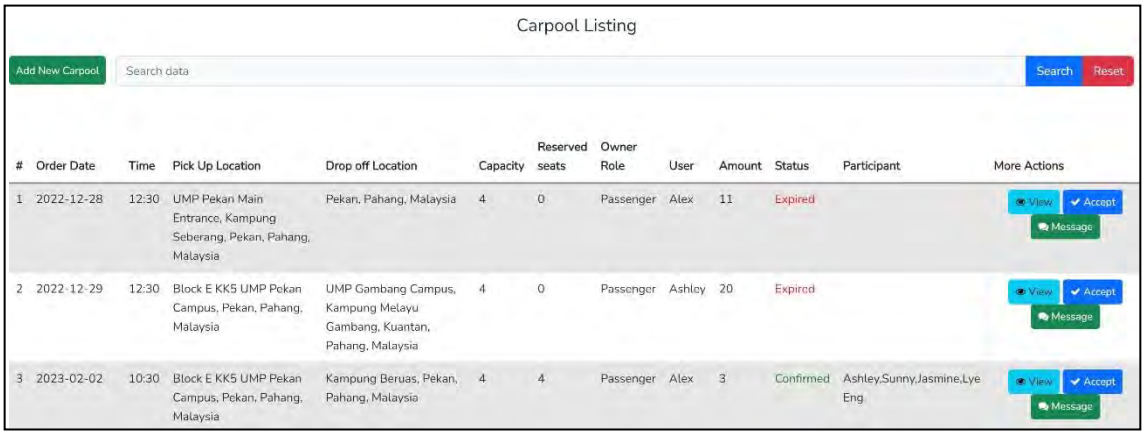


Figure 4.11 View Carpool Page

The system will display the latest carpool listing once the user has created or updated a carpool offer. The carpool listing is shown in Figure 4.11 which

consist of order date, time, pick up location, drop off location, capacity, reserved seat, owner role, user, amount, carpool status, and participant.

4.2.13 Edit Carpool

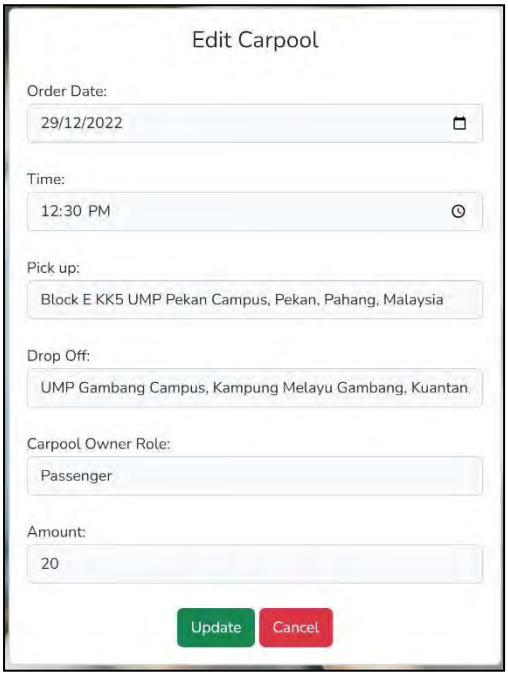


Figure 4.12 Edit Carpool Page

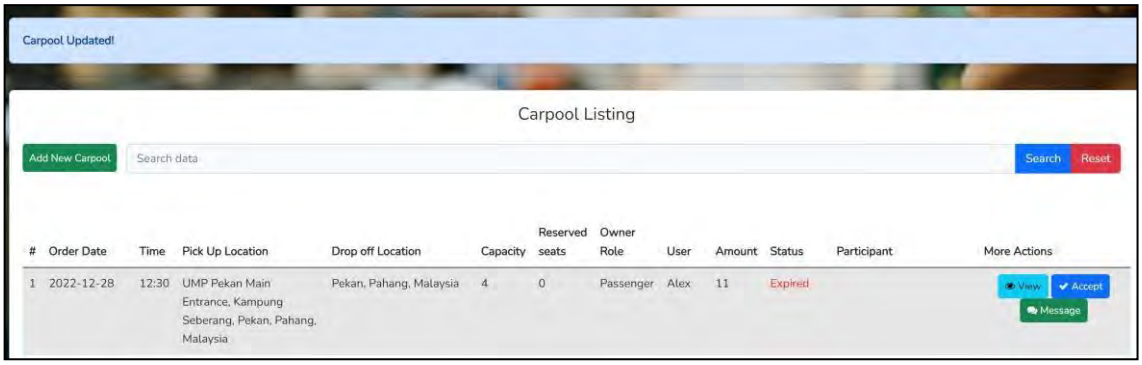


Figure 4.13 Update Carpool Successful Message

In the edit carpool page, both drivers and passengers can make changes to their own offers. The system will redirect to the view carpool page and display update successful message once the changes have been made as shown in Figure 4.13.

4.2.14 Delete Carpool Message

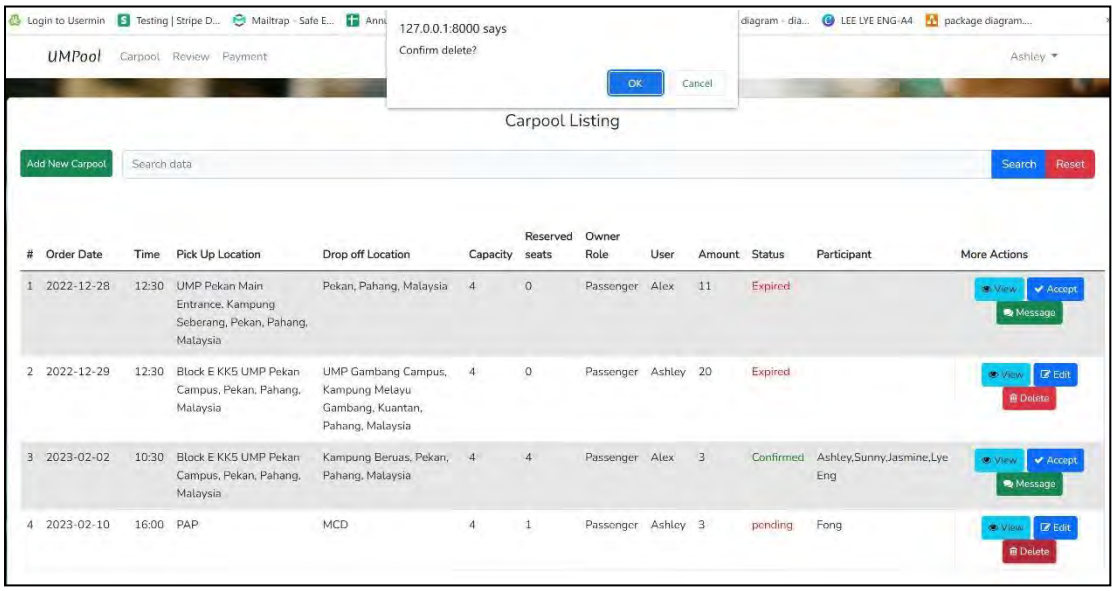


Figure 4.14 Delete Carpool Confirmation Message

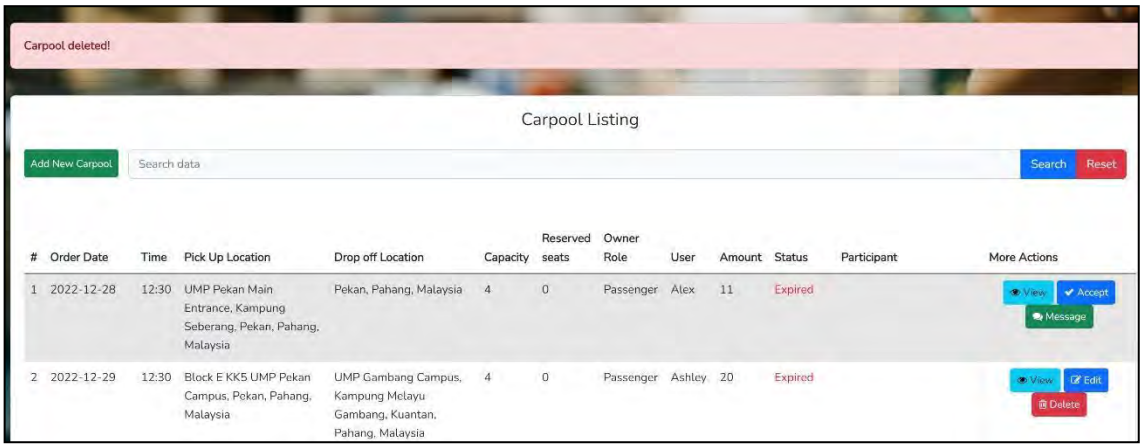


Figure 4.15 Carpool Listing

Delete confirmation message will pop out when users click “Delete” button. When user clicks “OK” on the confirmation message, the system displays “Carpool deleted!” message as shown in Figure 4.15.

4.2.15 Search Carpool

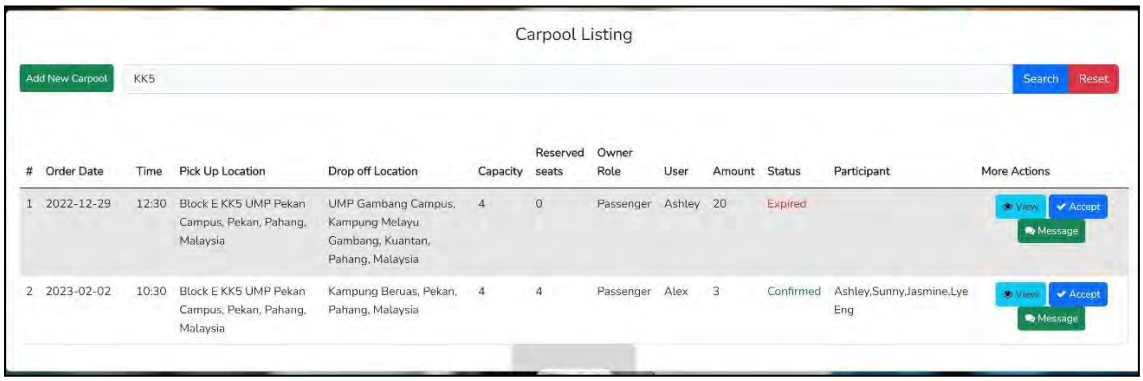


Figure 4.16 Search Carpool Interface

User allows to search their location by entering keyword (pick up or drop off location) in the search box. The system will display all search results with the keyword as shown in Figure 4.16.

4.2.16 Message Carpool Owner

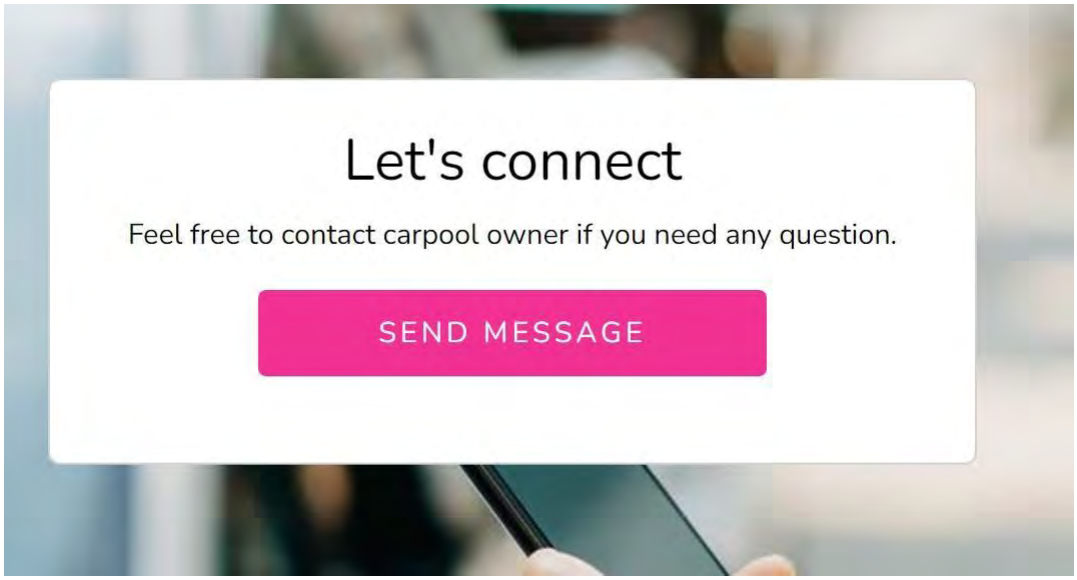


Figure 4.17 Message Carpool Owner Interface

User allows to connect with the carpool owner by WhatsApp if they need further assistance as shown in Figure 4.17.

4.2.17 View Profile

Figure 4.18 View Profile

In Figure 4.18, the profile details will display which including the username, email address, and contact number. Users can update their profile details by selecting the "Update" button, which directs to the profile information update page in Figure 4.19.

4.2.18 Update Profile

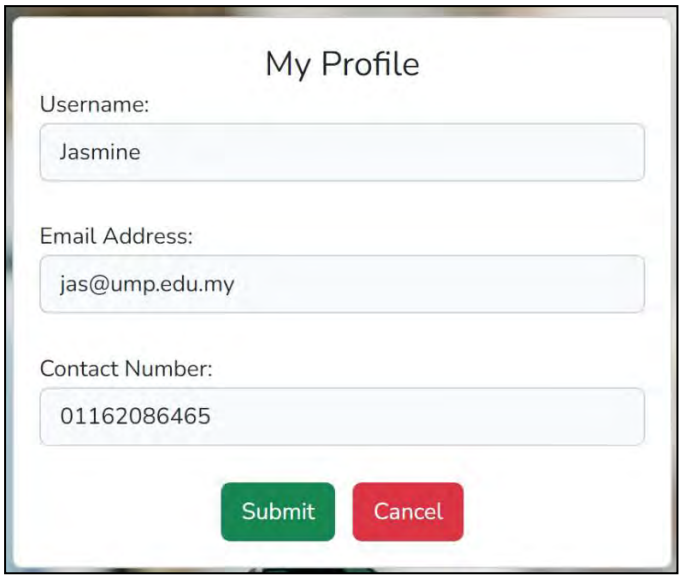


Figure 4.19 Update Profile

4.2.19 Latest Profile

Figure 4.20 Latest Profile

Users can submit updated profile details by selecting the "Submit" button and the system will store, update and redirect to the latest profile. An update successful message will pop up as shown in Figure 4.20.

4.2.20 Create Review

Figure 4.21 Create Review

4.2.21 Update Review

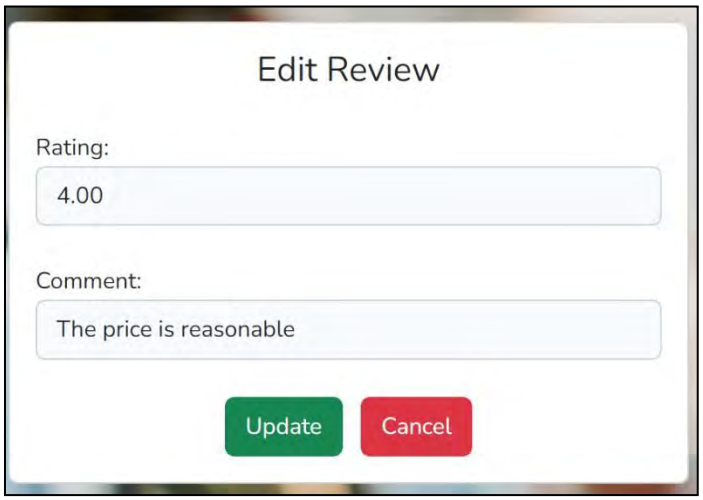


Figure 4.22 Update Review

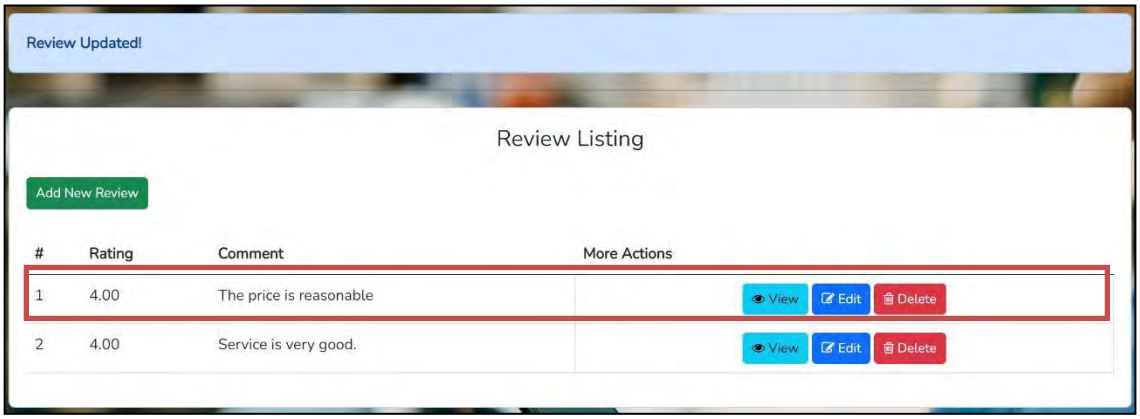


Figure 4.23 Updated Review

Users can create the review when the carpool is completed by selecting the "Save" button in the Figure 4.21. The carpool review will be revised once users enter the updated review and select "Update" button in Figure 4.22. The updated review will be displayed in the view review page in Figure 4.23.

4.2.22 Delete Review

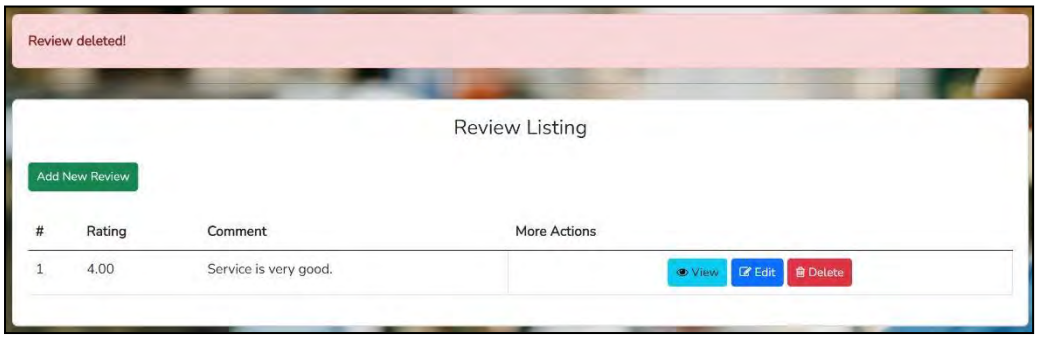


Figure 4.24 Delete Review Message

Figure 4.24 shows the delete review successfully message once the "Delete" button is clicked on the selected row.

4.2.23 Create Driving Verification

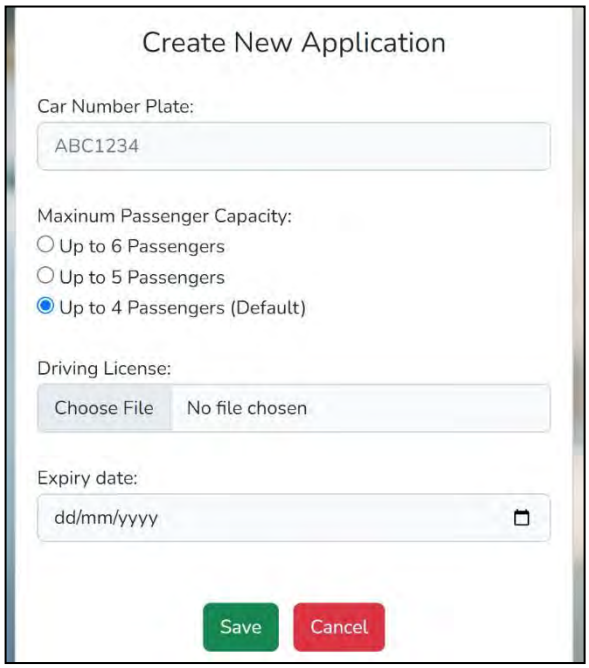


Figure 4.25 Create Driving Verification

Drivers can create the driving verification by entering the car number plate, maximum passenger capacity, driving license and expiry date in Figure 4.25. Applications will be stored at the verification table awaiting admin approval.

4.2.24 Update Driving Verification (Driver)

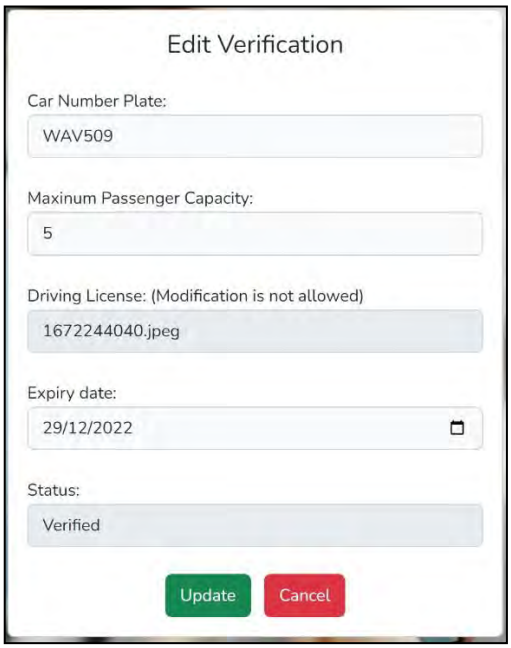


Figure 4.26 Edit Verification by Driver

4.2.25 Update Driving Verification (Admin)

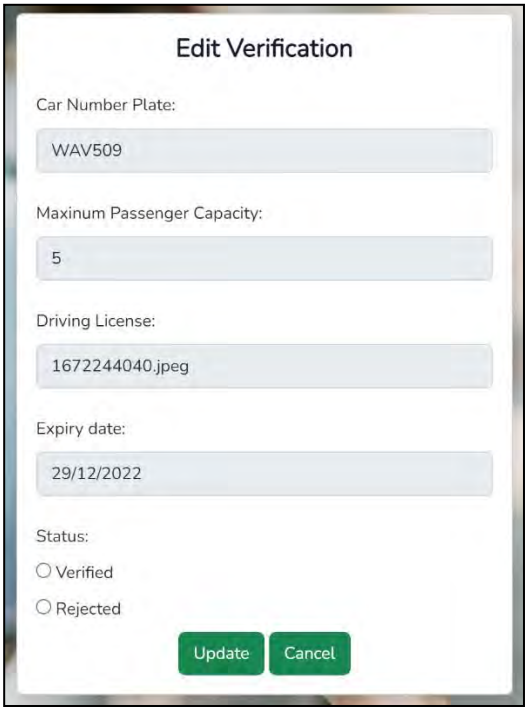


Figure 4.27 Edit Verification by Admin

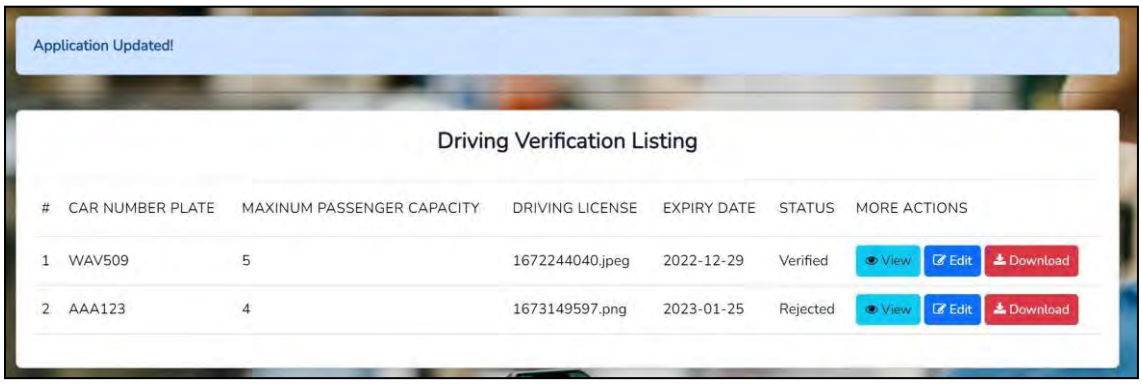


Figure 4.28 Updated Verification Message

4.2.26 View Driving Verification (Driver)

Figure 4.29 View Driving Verification (Driver)

Driver is allowed to update their previous application by entering car number plate, maximum passenger capacity, and expiry date. However, the driving license and application status cannot be modified by the driver as shown in Figure 4.26. The system will display the latest details once the verification has been completed which shown in Figure 4.29.

4.2.27 View Driving Verification (Admin)

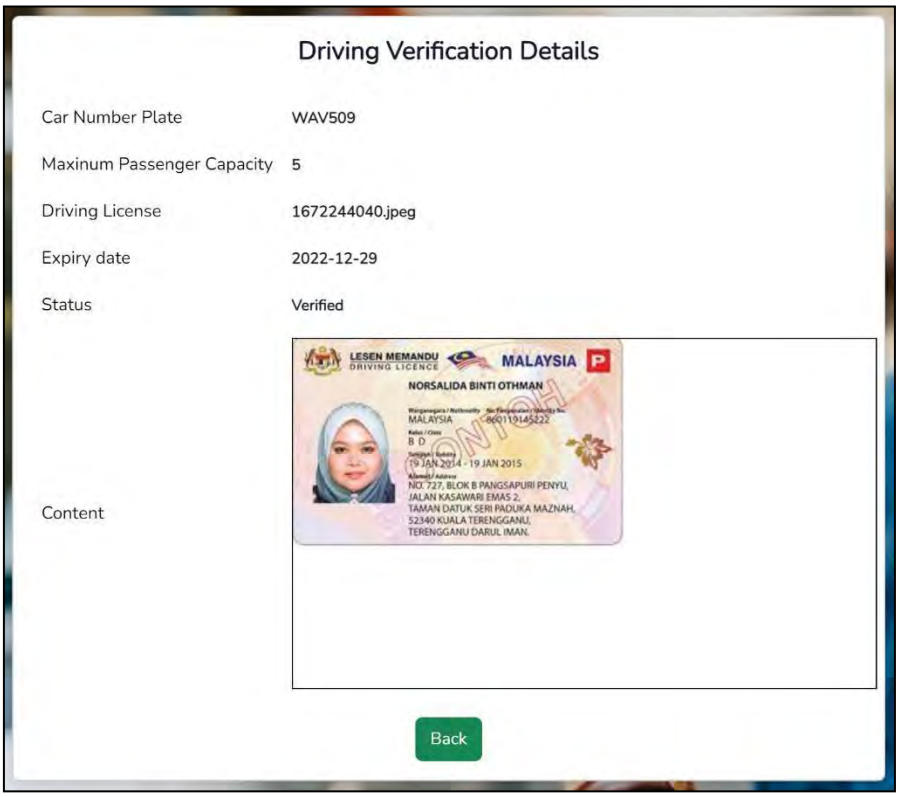


Figure 4.30 View Driving Verification (Admin)

Admin can check the drivers' driving verification details, such as driving licenses, and update their application status in Figure 4.27. Admins can also view and download the driver's license for verification purposes as shown in Figure 4.30.

4.2.28 Cart

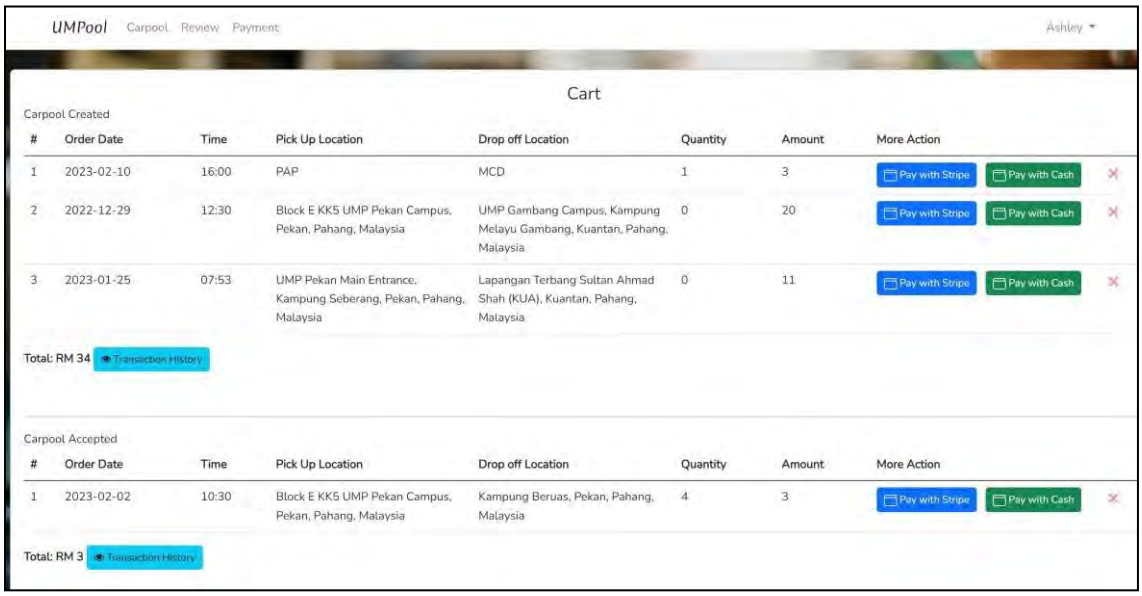


Figure 4.31 Cart Interface

Passenger can proceed payment with cash or stripe payment method. The cart displays the payments including carpool created and carpool accepted as shown in Figure 4.31.

4.2.29 Pay with Stripe

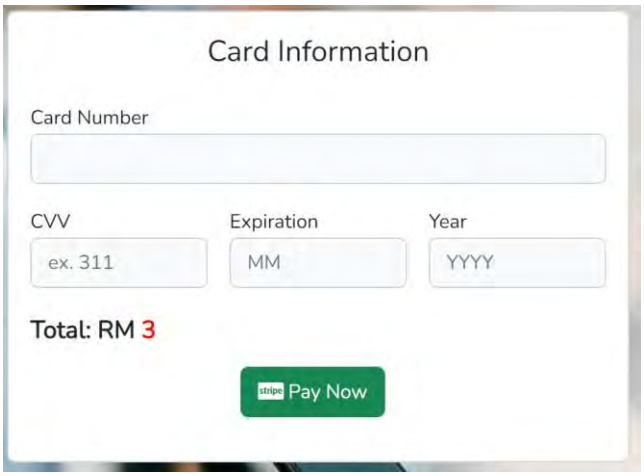


Figure 4.32 Pay With Stripe Interface

4.2.30 Pay with Cash

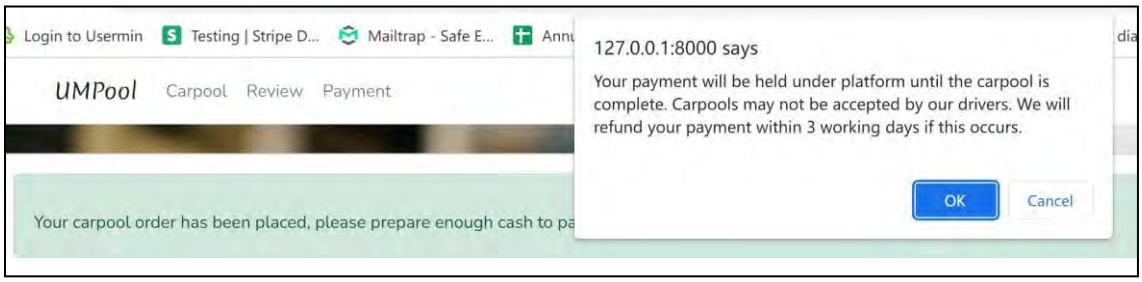


Figure 4.33 Pay with Cash Successful Message

A successful message will pop up when users select pay with cash option as shown in Figure 4.33. An alert message will appear to notify users that the carpool may not be accepted by the driver, they can track the status of upcoming carpools on dashboard.

4.2.31 Transaction History

ID	Transaction ID	Carpool ID	Amount	Payment Method	Status	Time	More Actions
1	ch_3MOZshAZAU1B2Ug32R4CL7Kv	2	3	stripe	1	2023-01-10 04:56:15	Download E-ticket
2		4	20	cash	0	2023-01-10 04:56:25	E-ticket unavailable
3		2	3	cash	0	2023-01-15 11:04:01	E-ticket unavailable

Figure 4.34 Transaction History Interface

User allows to track their transaction history and download the e-ticket of carpool as shown in Figure 4.34.

4.2.32 E-ticket

Figure 4.35 E-ticket Interface

4.3 TESTING AND RESULT DISCUSSION

The proposed system is developed iteratively according to the RAD Methodology. The system is tested constantly by 2 types of testing which are functional testing and user acceptance testing (UAT). The functional testing is conducted by the developers constantly during the development process, while the user acceptance testing is conducted by the users once the project is complete.

In the functional testing, it is conducted on the 6 modules. All the test cases have been planned and executed as table 3.16 Appendix A. 26 test cases were free of errors and 1 incident (minor defect) was recorded. The minor defect was caused by logic error and planned to be corrected by the development team.

A user acceptance test is conducted with 36 participants by Google form. UMPool accomplishes the best with 52.8% of participants agreeing that it provides clear content, while 36.1% strongly agree. 47.2% of participants agree that the content is easy to understand, whereas 33.3% strongly agree. A majority of participants agree that UMPool is a user-friendly interface, with 50% of participants agreeing and 36.1% of participants strongly agreeing.

A total of 63.9% of participants agree and 22.2% strongly agree that the navigation is easy to use. However, 16.7% of participants believed the navigation was not error-free, and corrective maintenance should be conducted in the following step to ensure it works correctly. Majority participants with 88.9% agree that UMPool has a reasonable navigation button.

The majority of participants agree that the graphics, buttons, and icons used in UMPool are proper and appropriate, as shown in Appendix A, Figure 7. Most of the participants are satisfied in the manage user login module, allowing them to log in and register an account.

Furthermore, manage carpool is one of the modules that participants find highly acceptable in creating carpool, updating carpool, accept carpool and view the carpool offer.

For the manage profile, 50% of participants are strongly agree that they are able to view their profile details through application while 36.1% of participants are agreeing with it. According to the survey, 52.8% of participants agree and 41.7% of participants are strongly agree that they are able to update profile details.

There is high acceptance of the Manage Review module by majority participants, as shown in Appendix A, Figure 10 and 11. This application allows the creation, updating, deleting and reviewing reviews.

Other than that, over 80% of participants also agreed that UMPool is able to complete the carpool payment, cancel and view the payment details. 61.1% of participants agreed that the system is able to approve the driver verification while 27.8% of participants strong agreeing. 77.8% of participants also agree that the system is able to reject the driver verification while 8.4% of participants did not agree with the statement.

Based on the results, we can conclude that the system is easy to use since 36.1% of participants didn't encounter any problems, 44.4% rarely encountered problems and 19.4% experienced problems sometimes. They also felt that UMPool was effective in making carpooling easier.

Overall, the majority of participants (38.9%) were strongly satisfied with the effectiveness of the proposed system, while 41.7% were satisfied. It can be concluded from data gathered during the user acceptance test that UMPool application has met the objectives of developing a web application to facilitate carpooling among UMP students and staff.

There are several strengths of the project development, including the availability of multiple payment methods, including cash and stripe, which lessen the dependence on a single payment provider. Passengers can continue to checkout by alternative payment methods if one of the payment gateways is under maintenance. In addition, precision is important when checking driving license data to against synthetic fraud and crimes. Drivers are required to upload their driving licenses and other relevant documents for verification. Administrators will verify those documents online and download them for further verification.

However, some weaknesses were identified in the project development. In the absence of real-time identity verification, fake driving licenses, underage drivers, and frauds become untraceable. Additionally, inefficient complaint services will reduce the user experience. It is inconvenient for users to connect with support and service online because there is no live chat feature. Additionally, the process of getting a complaint resolved is time-consuming since users submit their complaint through the platform and then wait for a response from customer service via email.

There are some challenges faced in the process of project development. The system intents to archive automate ride matching, including last-minute carpooling, dynamic carpooling, and multi-hop carpooling instead of in-advance carpooling. Furthermore, the system should automatically refund passengers if no one accepts their carpool request, rather than relying on administrators to deal with refunds.

4.3.1 Functional Testing

Functional testing is one of the black box tests that checks the functionalities of the system. It is conducted by developer to verify each of the module functions of the system behave as specified in the SRS documentation. Each module is evaluated by using appropriate input and comparing the actual output to expected output as shown in table 3.16 Appendix A.

4.3.2 User Acceptance Testing

The UAT document can be referred in Appendix A.

CHAPTER 5

CONCLUSION

5.1 OBJECTIVE REVISITED

This introduction of the project outlined three objectives that must be accomplished by the end of the development process. These objectives include studying the existing carpooling system, developing a web-based carpooling system for UMP, and evaluating the system's efficiency and functionality. The proposed web application achieves the objectives that stated. The proposed web application can overcome the limitations of the current carpooling service. Following that, the proposed web application was developed in Visual Studio Code utilising PHP and the MySQL relational database. The proposed web application has gone through the requirements planning, user design, construction, and cutover processes. Finally, the proposed web application has tested by functional and user acceptance testing.

5.2 FUTURE WORK

Some recommendations were given by the participants including navigation, location, design, security, and speed. It is important to continue growth and expansion of many suburbs and urban. Future work must continue to enhance the proposed web application. First, the proposed application can increase location availability and navigation, so users can search for and pin their intended destinations easily. The proposed application can continue to grow and expand by providing coverage to many suburbs and cities. The user

interface needs to be intuitive and attractive so that the user can pick it up quickly and easily.

Next, the proposed application can implement with the navigation feature. This application provides efficient and safe trips, which allows us to avoid traffic jams and detect potentially unsafe situations.

The proposed application should provide better performance to load web pages more quickly. Lastly, the proposed application supports e-wallet payments, which are more commonly used by students.

REFERENCES

Beynon-Davies, P., Came, C., Mackay, H., & Tudhope, D. (1999). Rapid application development (Rad): An empirical review. *European Journal of Information Systems*, *8*(3), 211–232. https://doi.org/10.1057/palgrave.ejis.3000325

CompareHero. (2021). Latest Petrol Price for RON95, RON97 & Diesel in Malaysia. In *CompareHero.my*. https://www.comparehero.my/transportation/articles/latest-petrol-price-ron95-ron97-diesel

Jerrica. (2021). Malaysia has the 4th worst traffic jam condition in SEA with second highest CO2 emission levels . *WapCar*, 2–6. https://www.wapcar.my/news/malaysia-has-the-4th-worst-traffic-jam-condition-in-sea-with-second-highest-co2-emission-levels-22569

Manoj Kumar, N., Sudhakar, K., & Samykano, M. (2019). Techno-economic analysis of 1 MWp grid connected solar PV plant in Malaysia. *International Journal of Ambient Energy*, *40*(4), 434–443. https://doi.org/10.1080/01430750.2017.1410226

McKenzie, B. (2015). Who Drives to Work? Commuting by Automobile in the United States: 2013. *American Community Survey Reports*, *ACS-32*. https://www.census.gov/library/publications/2015/acs/acs-32.html

Polzin, B. S. (2022). *The Decline of Carpooling — Can App-Based Carpooling Reverse the Trend ?* 1–4.

Rapley, K. (1995). RAD or TRAD or both? The future of software development. *IEE Colloquium (Digest)*, *237*, 311–312. https://doi.org/10.1049/ic:19951554

States, U., Ii, W. W., Ii, W. W., Office, T. U. S., Defense, C., Sharing, C., & Exchange, C. (2022). *History of carpooling 2020-07-28*. 2020–2022.

Susskind, L., Chun, J., Goldberg, S., Gordon, J. A., Smith, G., & Zaerpoor, Y. (2020). Breaking Out of Carbon Lock-In: Malaysia's Path to Decarbonization. *Frontiers in Built Environment*, *6*(March). https://doi.org/10.3389/fbuil.2020.00021

Work, R. (2018). *The Benefits of C arpooling*. https://doi.org/10.7922/G2DZ06GF

APPENDIX A

GOOGLE FORM - ANALYSIS OF CARPOOLING NEEDS IN UNIVERSITI MALAYSIA PAHANG (UMP)

Figure 1 Analysis of Carpooling Needs in Universiti Malaysia Pahang (Ump) - Part 1

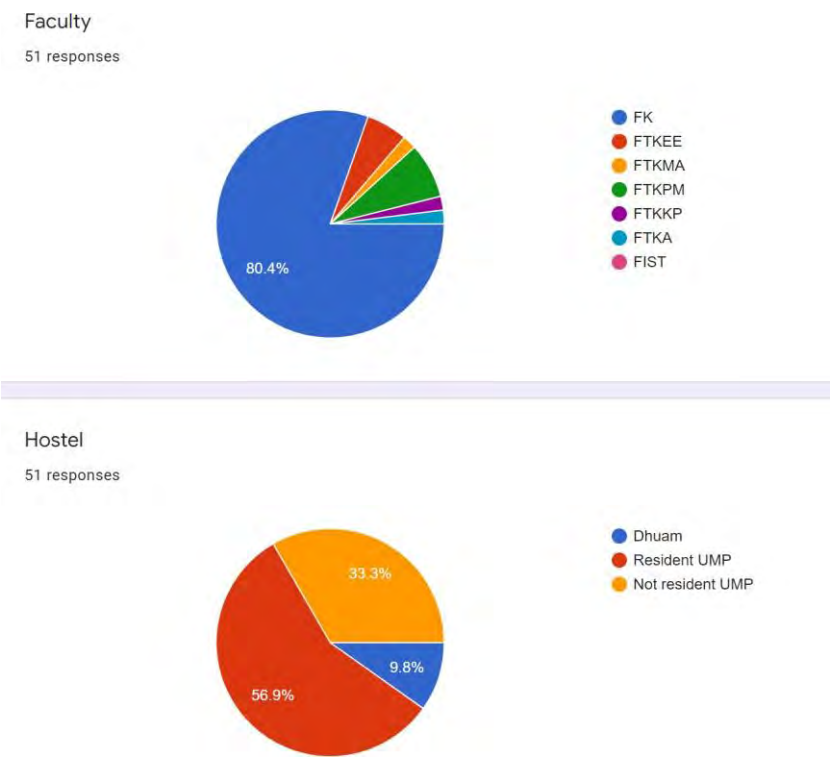


Figure 2 Analysis of Carpooling Needs in Universiti Malaysia Pahang (UMP) - Part 2

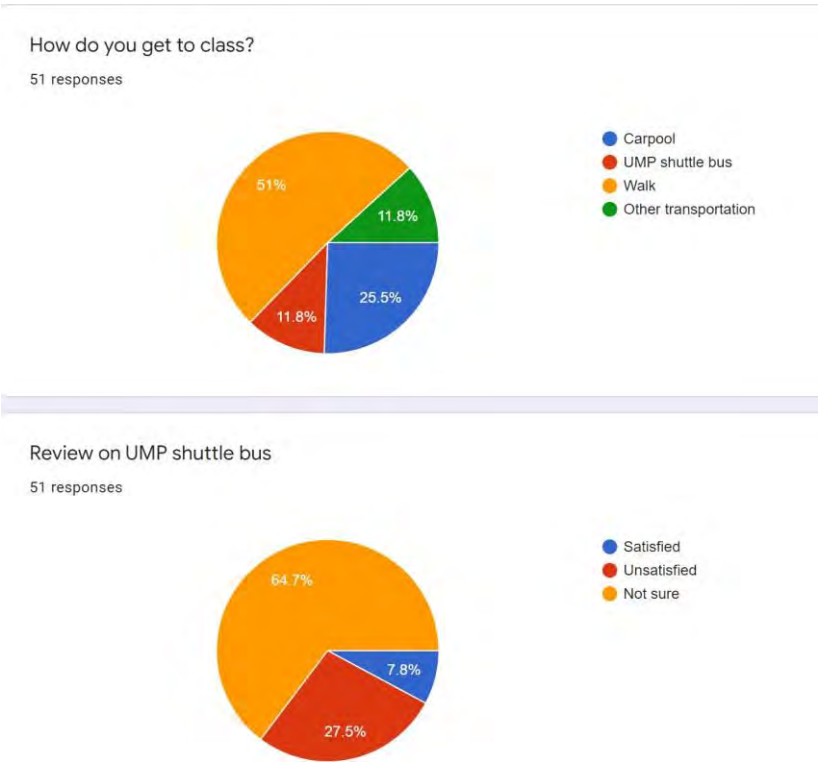


Figure 3 Analysis of Carpooling Needs in Universiti Malaysia Pahang (UMP) - Part 3

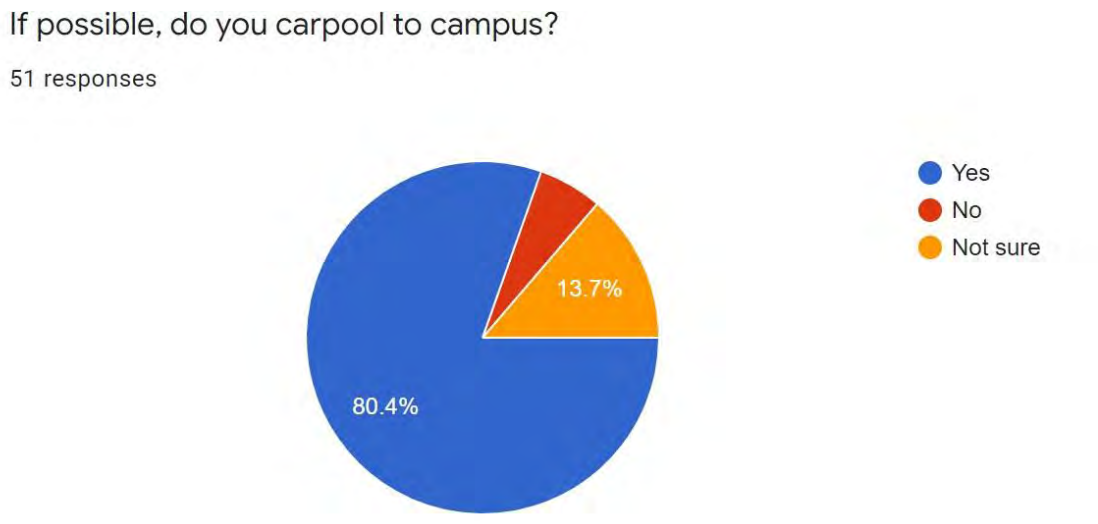


Figure 4 Analysis of Carpooling Needs in Universiti Malaysia Pahang (UMP) - Part 4

TESTING PLAN

Table 3.15 Testing Plan

Event	Test Data	Expected Result	Actual Result	Pass / Fail	Remark
User Login	Admin, Driver, Passenger Email: alex@example.com Password: alex1234	System displays login success message, and it will redirect to main menu page.			
Invalid data for registration	Username: alex@example.com Password: alex1234 Email: empty data Contact Number: 01726453000 Role: passenger	System displays error message, "Please fill out this field".			
Registration	Driver, and Passenger Username: Daniel Password: daniel123 Email: Daniel01@gmail.com	The system displays register success message, and it will redirect to dashboard page.			

	Contact Number: 01726453000 Role: Driver				
Invalid data for login	Incorrect password entered	The system displays error message, "Email address or password is wrong!"			
Reset Password	Admin, Driver, Passenger Email address: Daniel01@gmail.com	The system displays "The reset password link is sent to your email". The email can be retrieved at the inbox of email address entered.			
Update Profile Details	Admin, Driver, Passenger Click "Update" button in manage profile page	The system displays the update profile page. User can select from the following options (Email, and contact number) to update.			

Delete Profile Details	Admin, Driver, Passenger Click "Delete" button in manage profile page	The system displays the delete profile page. User can select from the following options (Email, contact number and WhatsApp QR code) to delete.			
Create Carpool	Driver, and Passenger Click "Add new carpool" button in manage carpool page. Offer date: (28/6/2022) Time: 10:00pm Pick up: UMP Drop off: East Coast Mall Seat reserved: 1 Carpool owner role: Driver Amount: RM15.00 Click "Save" button	The system displays the create carpool offer page. The created offer will be display in the manage carpool page.			
Accept	Driver, and Passenger	The system displays the			

Carpool	Click "Accept" button in manage carpool page	accept confirmation message, "Thank you for accepting this offer! ". The reserved seats will increase by 1.			
Invalid accept carpool	Driver, and Passenger Click "Accept" button on the past booking date	The system displays the error message, "Processing failed, please try again."			
Update Carpool	Driver, and Passenger Click "Edit" button in manage carpool page. Enter the updated carpool information. Offer date: (30/6/2022) Time: 10:00pm Pick up: UMP Drop off: East Coast Mall Seat reserved: 1 Carpool owner role:	The system will redirect to update carpool page. The system will save the updated information in the database and redirect back to the manage carpool page.			

	Driver Amount: RM15.00 Click “Update” button				
Delete Carpool	Driver, and Passenger Click “Delete” button in manage carpool page	The system displays the delete confirmation message, “Confirm delete ?”			
View Carpool	Driver, and Passenger Click “View” button in manage carpool page.	The system displays the carpool information of the selected row of carpool.			
Proceed Payment	Passenger Click “Proceed” button in manage payment page. Payment method: Stripe	System redirects to Stripe payment gateway page.			
Cancel Payment	Passenger Click “x” button in manage payment page.	System displays the cancel confirmation message, “It will be removed from			

		the listing when you decide to cancel your payment.”			
View transaction history	Driver and passenger Click “transaction history” button.	System displays the details of the payment.			
Generate QR code	Driver and passenger Click “Generate E-ticket” button.	System displays the QR code which contains the respective carpool information .			
Create Review	Passenger Click “Create” button in manage review page. Choose new review: 5.00 Enter comment: Good customer services! Click “Add” button	System displays the review success message. The review can be view at manage review page.			
Update Review	Passenger Select the review item to update and click “Update” button. Choose new review:	System displays the review update success message.			

	3.00 Enter comment: Other passengers are nice. Already booked next offer with them!! Click “Add” button				
Delete Review	Passenger Select the review item to delete and click “Delete” button.	System displays delete confirmation message, “Confirm delete ?”			
View Review	Passenger Select the review item to view and click “View” button.	System displays the respective review details.			
Create Driving Verification	Driver Click “Create” button at the manage driving verification page. Upload your driving license: lisence_cb19092.png Expire Date: 09/24 Click “Add” button	System displays create success message.			
Update Driving	Driver	The system displays update			

Verification	Click "Update" button at the manage driving verification page. Upload your driving license: A00100332.png Expire Date: 09/2024 Click "Add" button	success message.			
Delete Driving Verification	Driver Select the driving details item to delete and click "Delete" button.	System displays delete confirmation message, "Confirm delete ?"			
View Driving Verification	Driver System allows drivers to view their details of driving verification.	The system display view verification details at manage driving verification page.			
Approve Verification	Admin System allows admin to approve driving verification of drivers.	System displays the application with the approve status.			
Reject Verification	Admin System allows admin to reject driving	System displays the application with the			

	verification of drivers.	rejected status.			

RESULT OF TESTING PLAN

Table 3.16 Result of testing plan

Event	Test Case	Test Data	Expected Result	Actual Result	Pass / Fail	Remark
User Login	TC01	Email: alex@example.com Password: alex1234	System displays a login success message, and it will redirect to the main menu page.	Same as expected result	Pass	-
Invalid data for registration	TC02	Username: alex@example.com Password: alex1234 Email: empty data Contact Number: 01726453000 Role: passenger	System displays an error message, “Please fill out this field”.	Same as expected result	Pass	-
Registration	TC03	Username: Daniel Password: daniel123 Email: Daniel01@gmail.com Contact Number: 01726453000 Role: Driver	The system displays a register success message, and it will redirect to dashboard.	The system does not display register success message.	Fail	-

Invalid data for login	TC04	Incorrect password entered	The system displays an error message, “Email address or password is wrong!”	Same as expected result	Pass	-
Reset Password	TC05	Email address: alex@example.com	The system displays “We have emailed your password reset link!”. The email can be retrieved at the inbox of the email address entered.	Same as expected result	Pass	-
Update Profile Details	TC06	Click “Update” button in manage profile page	The system displays the update profile page. User can select from the following options (Username, Email, and contact number) to update.	Same as expected result	Pass	-
Create Carpool	TC08	Click “Add new carpool” button in manage carpool page. Offer date: (28/6/2022)	The system displays the successful message “Carpool added! ” in the create carpool	Same as expected result	Pass	-

		Time: 10:00pm Pick up: UMP Drop off: East Coast Mall Seat reserved: 1 Carpool owner role: Driver Amount: RM15.00 Click “Save” button	offer page. The created offer will be display in the manage carpool page.			
Accept Carpool	TC09	Click “Accept” button in manage carpool page	The system displays the accept confirmatio n message, “Thank you for accepting this offer! ”. The reserved seats will increase by 1.	Same as expected result	Pass	-
Invalid accept carpool	TC10	Click “Accept” button on the past booking date	The system displays the error message, “Processing failed, please try again.”	Same as expected result	Pass	-
Update Carpool	TC11	Click “Edit” button in manage carpool page. Enter the updated carpool information. Offer date:	The system will redirect to update the carpool page. The system will save the updated	Same as expected result	Pass	-

		(30/6/2022) Time: 10:00pm Pick up: UMP Drop off: East Coast Mall Seat reserved: 1 Carpool owner role: Driver Amount: RM15.00 Click "Update" button	information in the database and redirect back to the manage carpool page. Update successfully message "Carpool updated" will display.			
Delete Carpool	TC12	Click "Delete" button in manage carpool page	The system displays the delete confirmation message, "Confirm delete ?"	Same as expected result	Pass	-
View Carpool	TC13	Click "View" button in manage carpool page.	The system displays the carpool information of the selected row of carpool.	Same as expected result	Pass	-
Proceed Payment	TC14	Click "Pay with Stripe" button in manage payment page.	System redirects to Stripe payment gateway page.	Same as expected result	Pass	-
Proceed Payment - Cash	TC15	Click "Pay with Cash" button in manage payment	System displays successful message	Same as expected result	Pass	-

		page.	"Your carpool order has been placed, please prepare enough cash to pay the driver when the carpool is complete".			
Cancel Payment	TC15	Click "x" button in manage payment page.	System displays the cancel confirmation message, "It will be removed from the listing when you decide to cancel your payment."	Same as expected result	Pass	-
View transaction history	TC16	Click "transaction history" button.	System displays the details of the payment.	Same as expected result	Pass	-
Generate QR code	TC17	Click "Generate E-ticket" button.	System displays the QR code which contains the respective carpool information.	Same as expected result	Pass	-

Create Review	TC18	Click “Create” button in manage review page. Choose new review: 5.00 Enter comment: Good customer services! Click “Add” button	System displays the review success message. The review can be viewed at manage review page.	Same as expected result	Pass	-
Update Review	TC19	Select the review item to update and click “Update” button. Choose new review: 3.00 Enter comment: Other passengers are nice. Already booked the next offer with them!! Click “Add” button	System displays the review update success message.	Same as expected result	Pass	-
Delete Review	TC20	Select the review item to delete and click “Delete” button.	System displays a delete confirmatio n message, “Confirm delete ?”	Same as expected result	Pass	-
View Review	TC21	Select the review item to view and click “View” button.	System displays the respective review details.	Same as expected result	Pass	-
Create Driving	TC22	Click “Create” button at the manage driving	System displays create	Same as expected	Pass	-

Verification		verification page. Upload your driving license: lisence_cb19092.png Expire Date: 09/24 Click “Add” button	success messages.	result		
Update Driving Verification	TC23	Click the “Update” button at the manage driving verification page. Upload your driving license: A00100332.png Expire Date: 09/2024 Click “Add” button	The system displays an update success message.	Same as expected result	Pass	-
Delete Driving Verification	TC24	Select the driving details item to delete and click “Delete” button.	System displays a delete confirmatio n message, “Confirm delete ?”	Same as expected result	Pass	-
View Driving Verification	TC25	System allows drivers to view their details of driving verification.	The system displays view verification details at manage driving verification page.	Same as expected result	Pass	-
Approve Verification	TC26	System allows the admin to approve driving verification of drivers.	System displays the application with the approve	Same as expected result	Pass	-

			status.			
Reject Verification	TC27	System allows the admin to reject driving verification of drivers.	System displays the application with the rejected status.	Same as expected result	Pass	-

USER ACCEPTANCE TESTING (UAT)

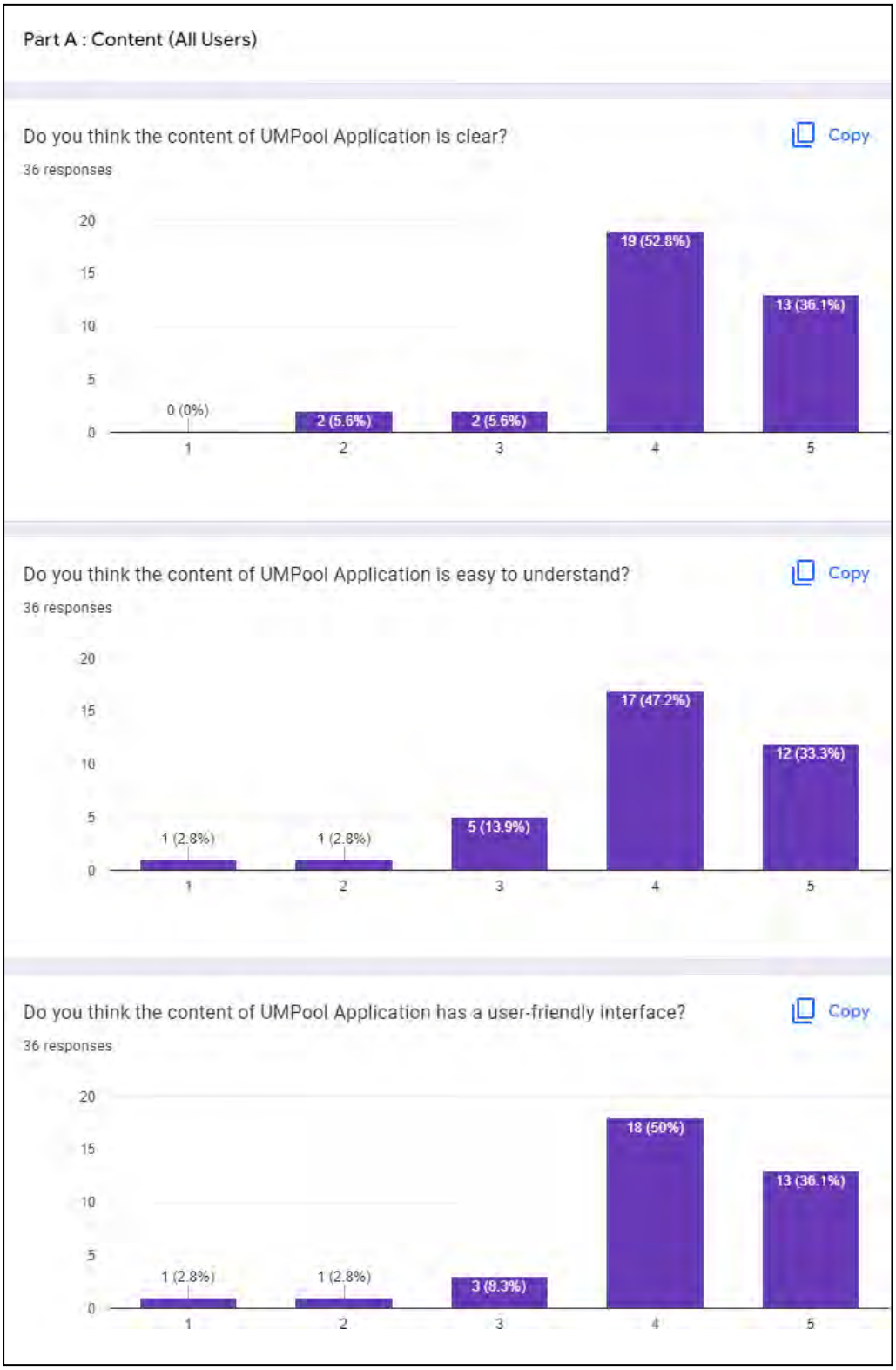


Figure 5 User Acceptance Testing (UAT) - Part A

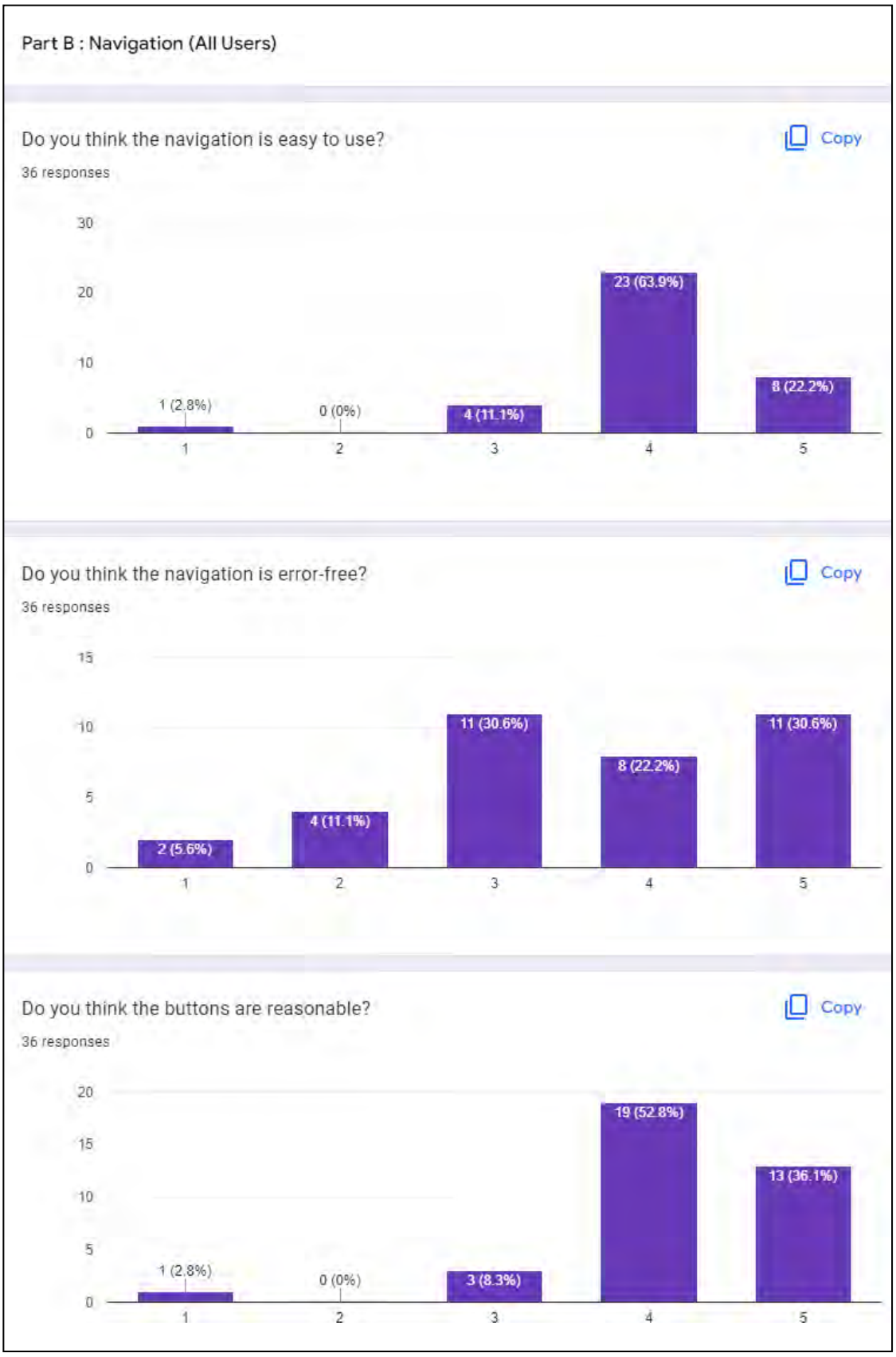


Figure 6 User Acceptance Testing (UAT) - Part B

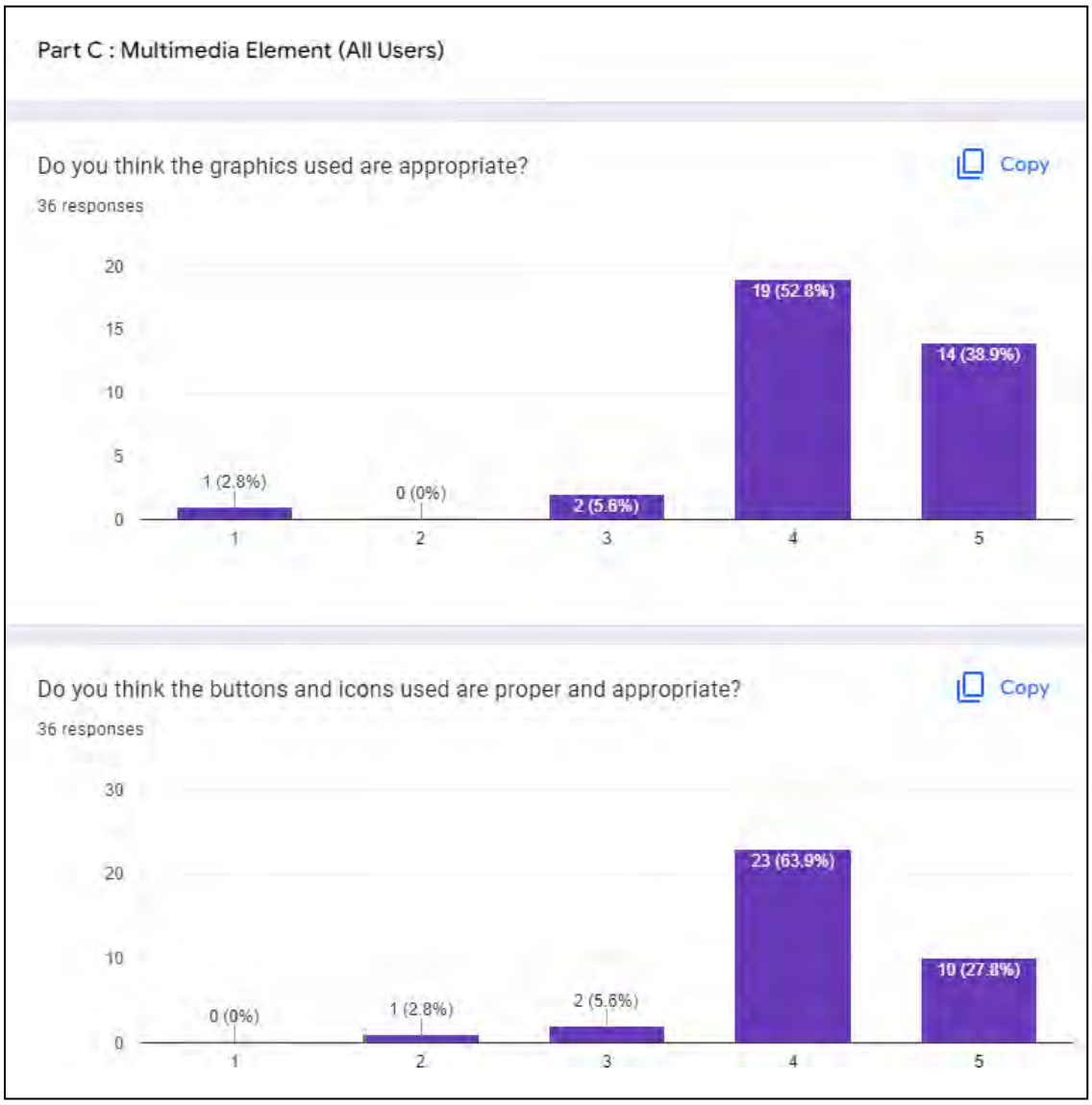


Figure 7 User Acceptance Testing – Part C

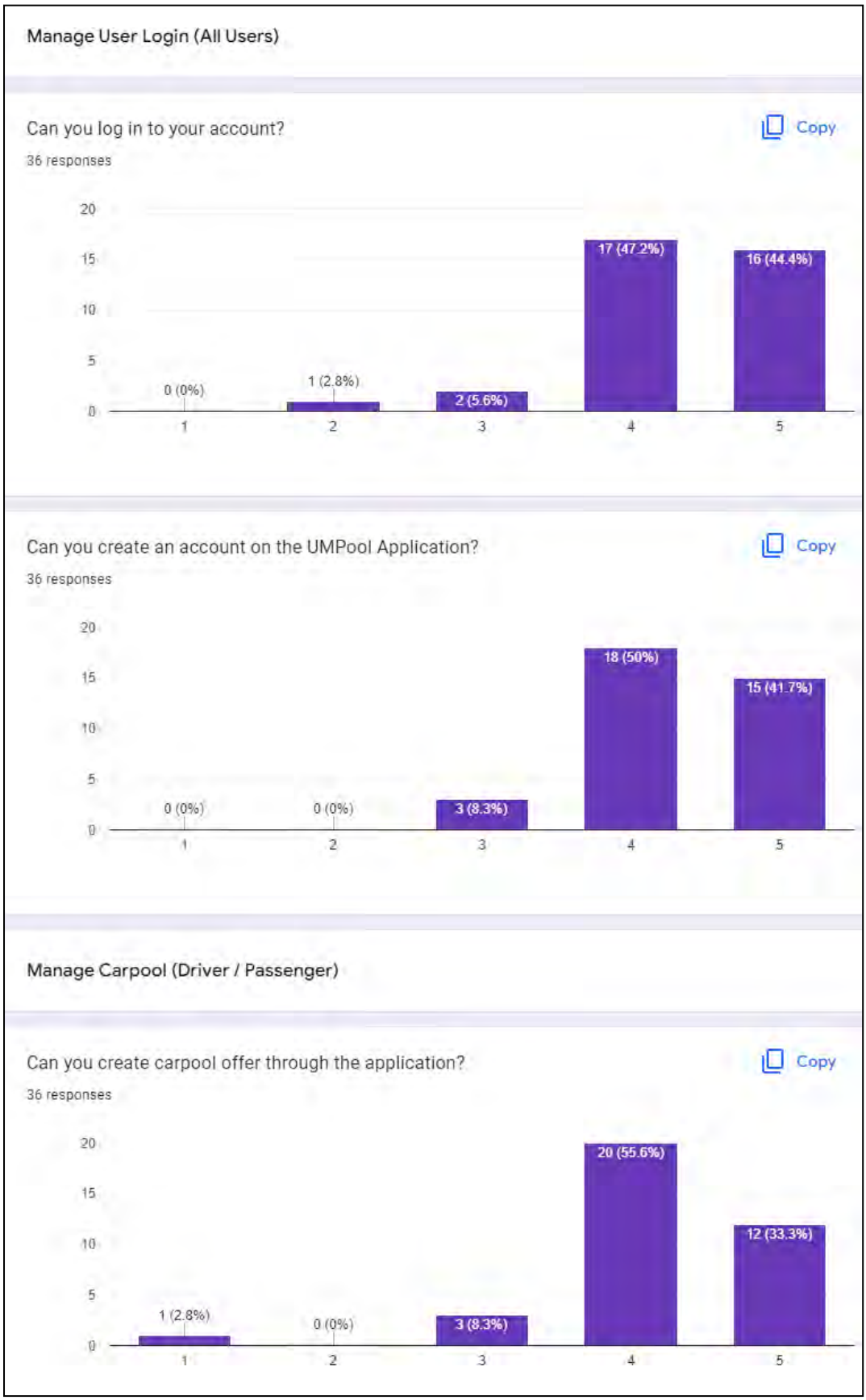


Figure 8 User Acceptance Testing -Part D (i)

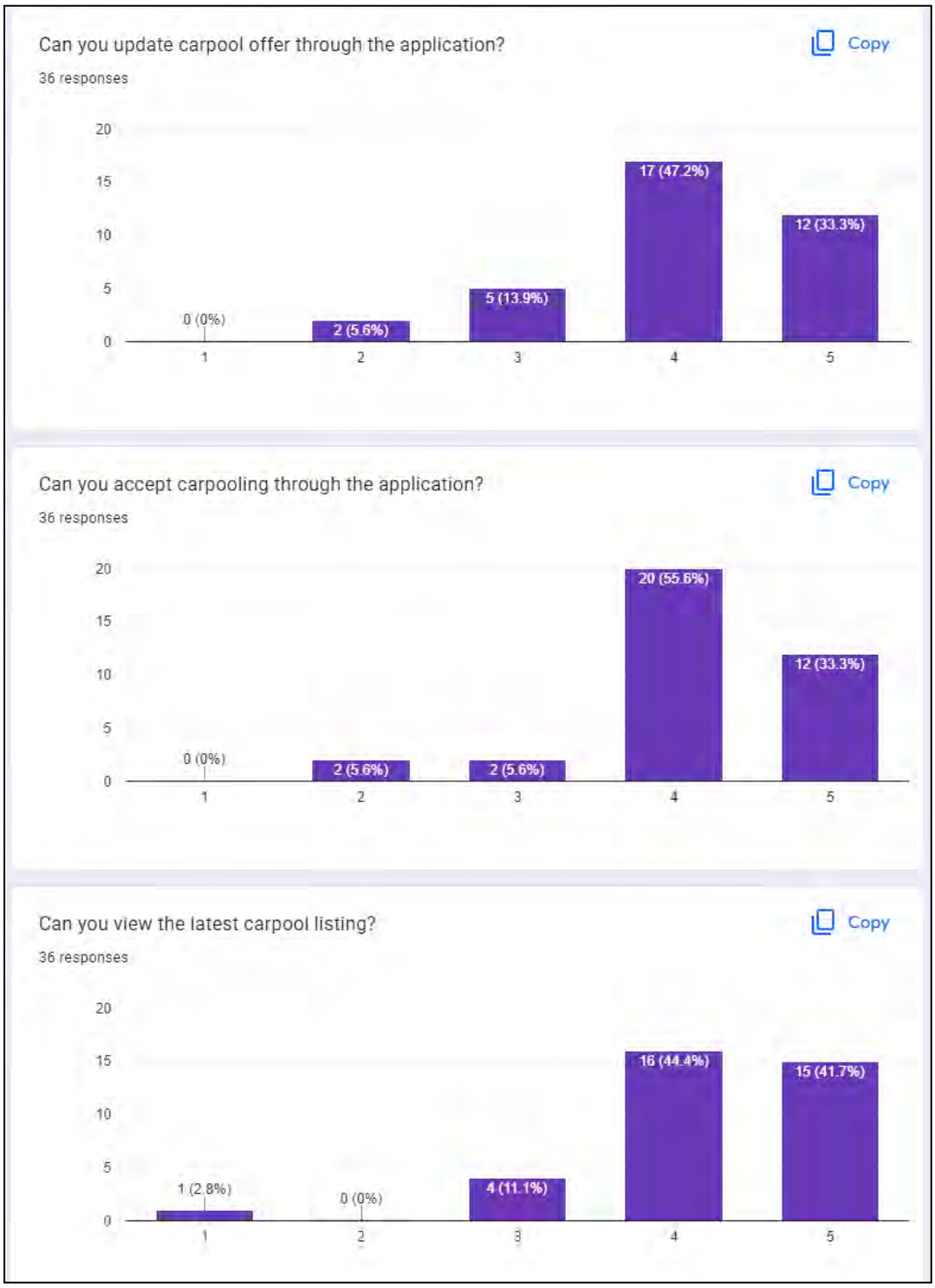


Figure 9 User Acceptance Testing – Part D (ii)

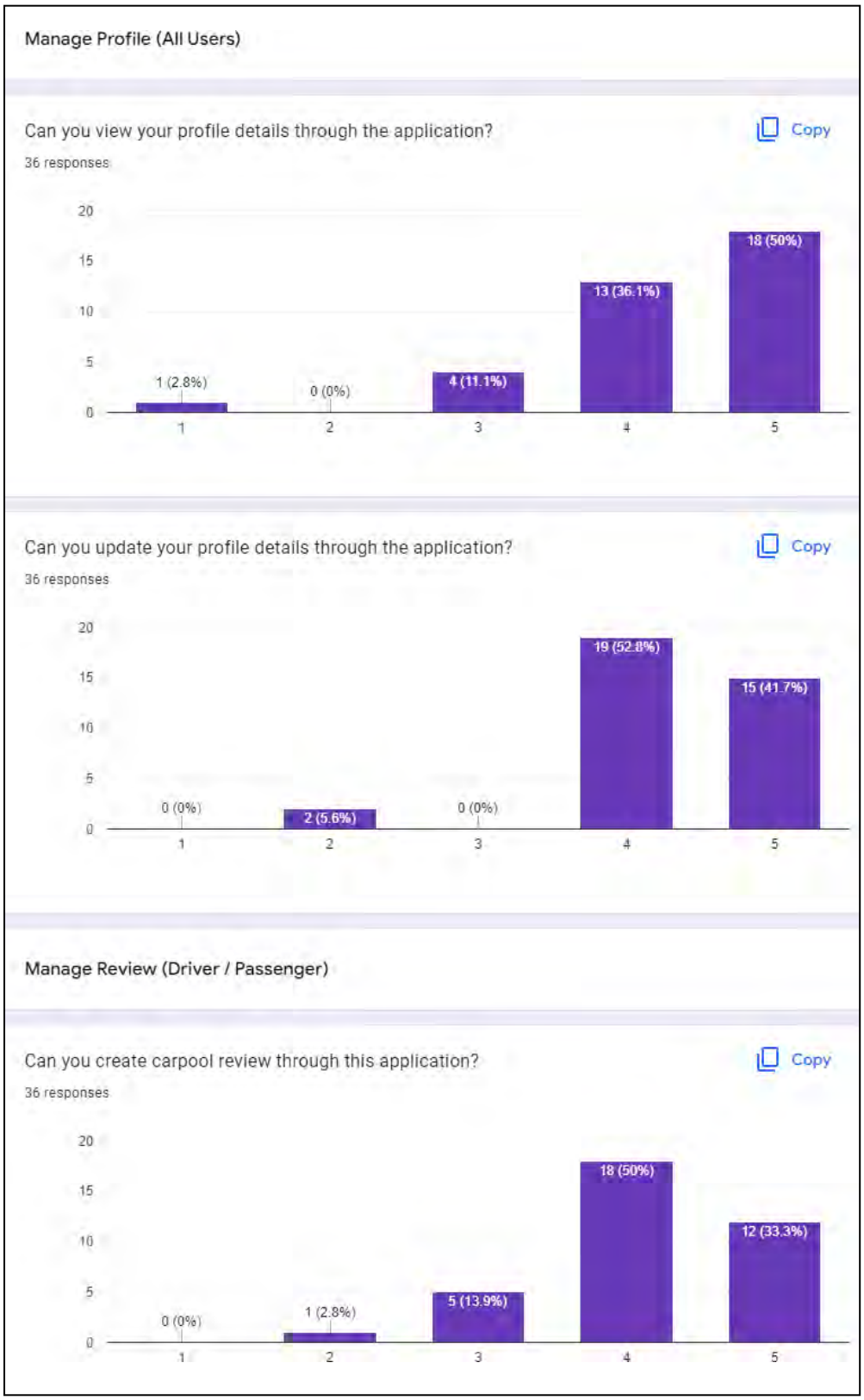


Figure 10 User Acceptance Testing – Part D (iii)

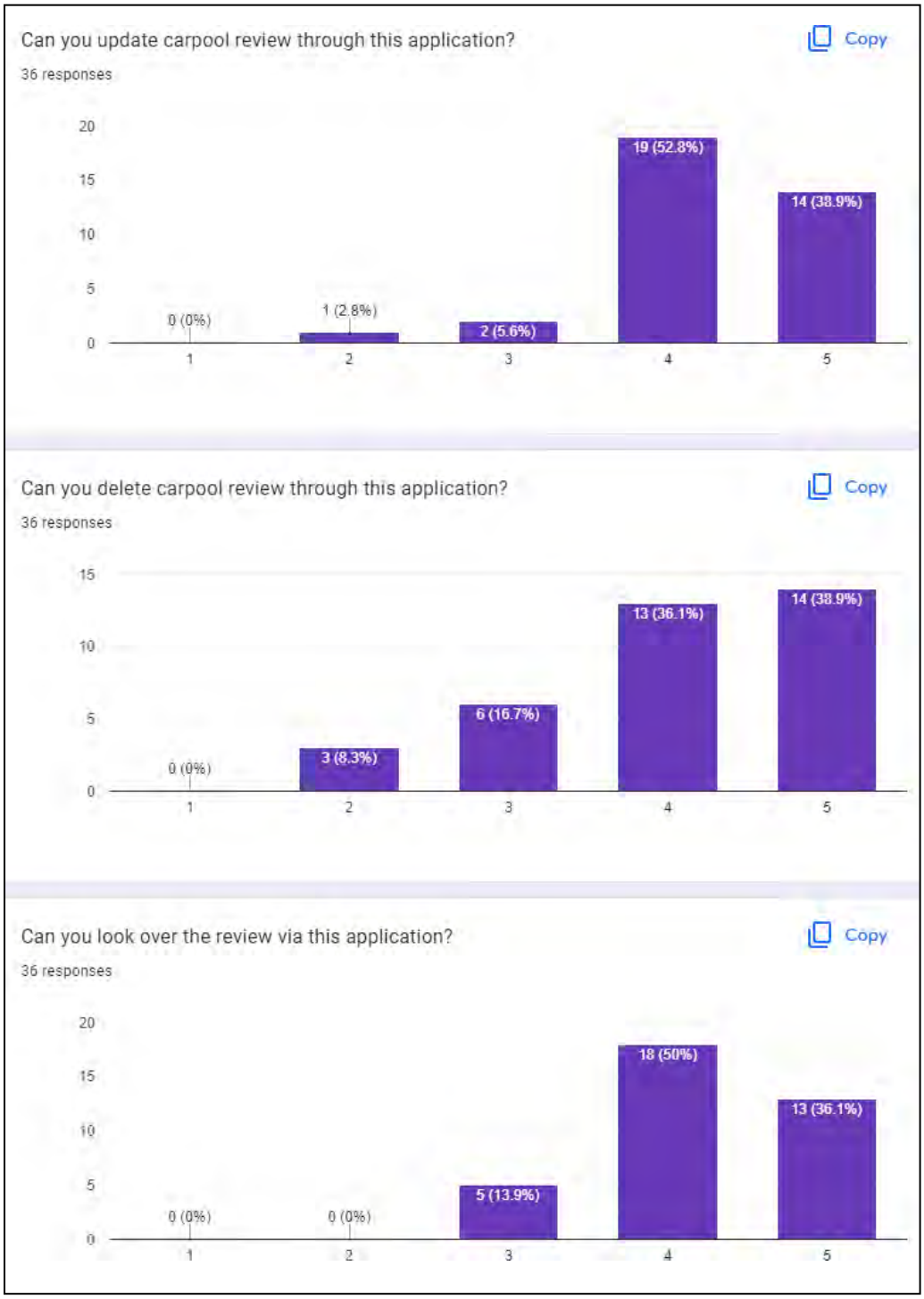


Figure 11 User Acceptance Testing – Part D (iv)

Figure 12 User Acceptance Testing – Part D (v)

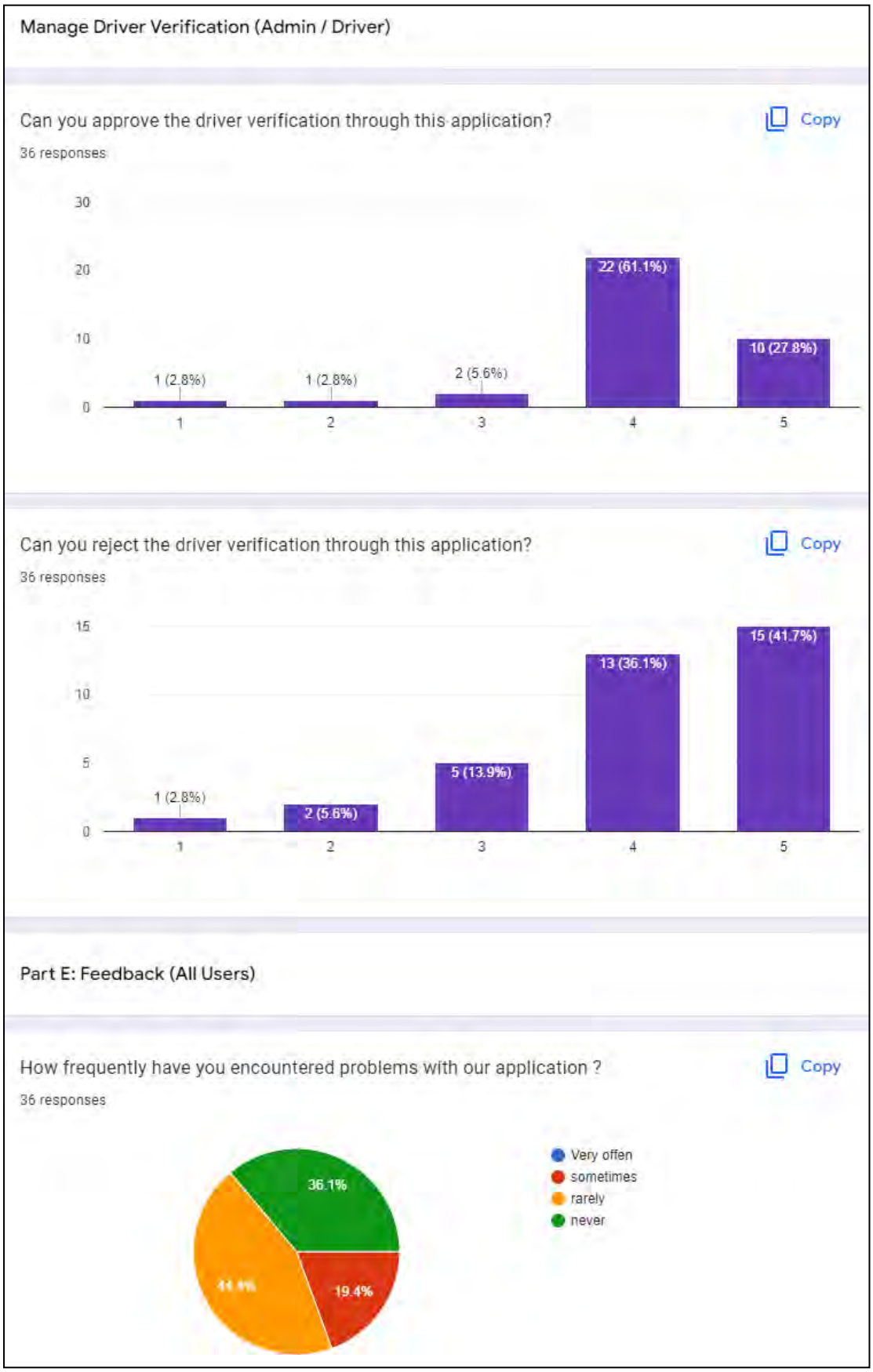


Figure 13 User Acceptance Testing – Part D (vi)

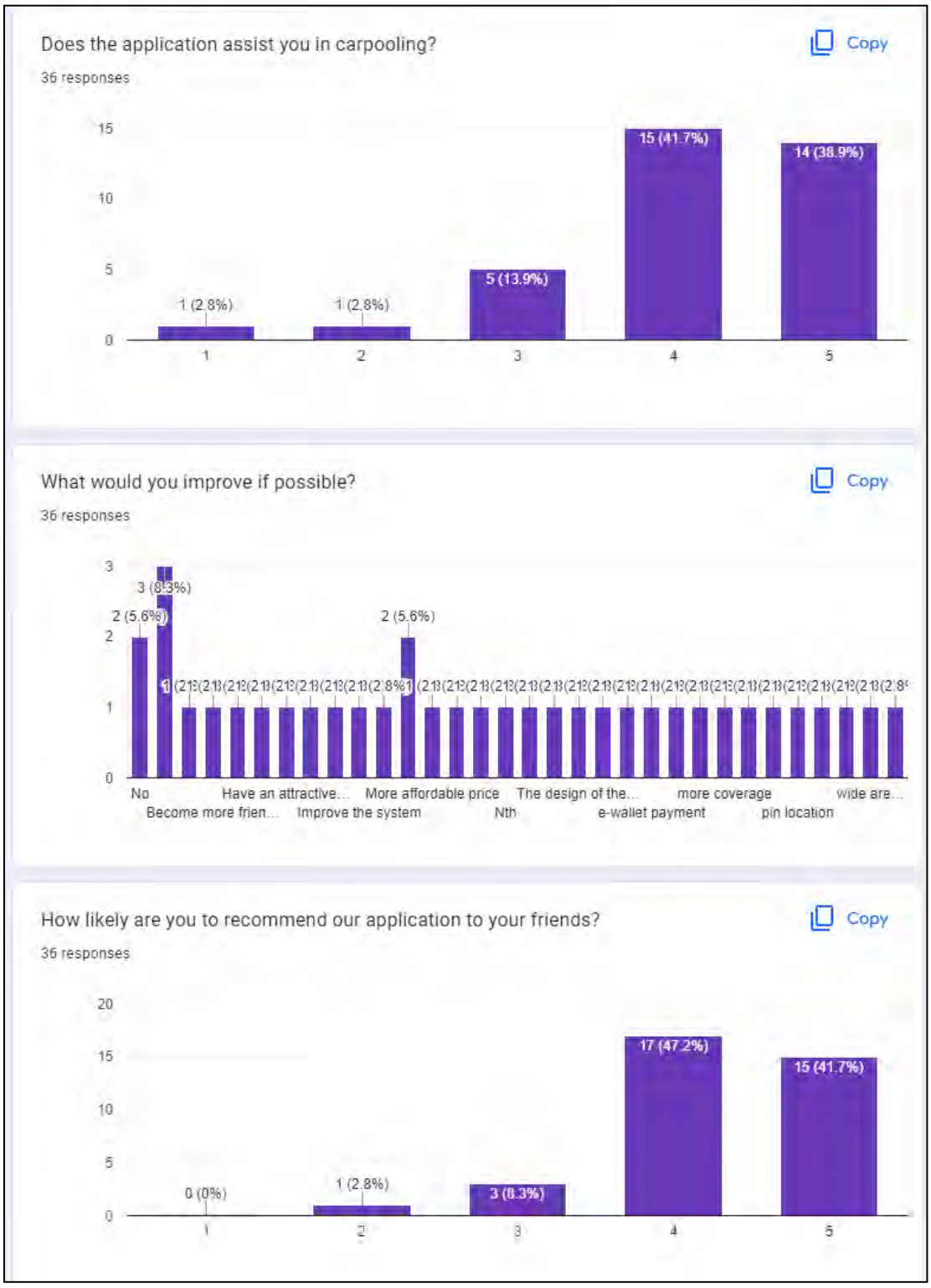


Figure 14 User Acceptance Testing – Part D (vii)

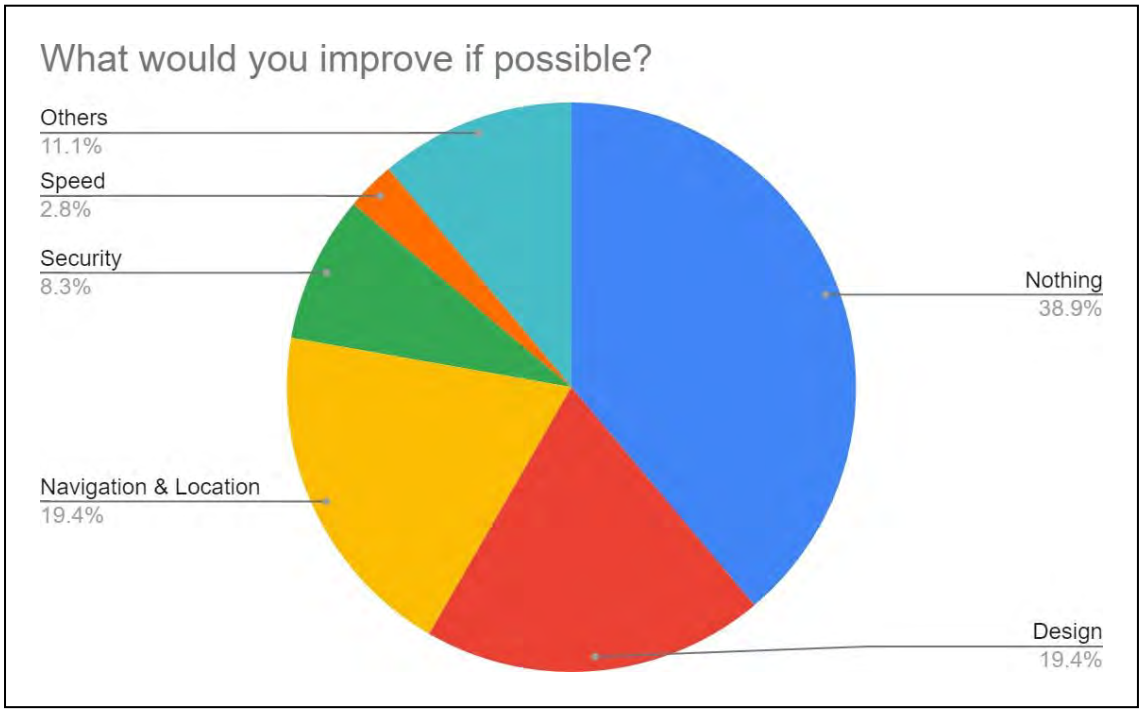


Figure 15 User Acceptance Testing – Part D (viii)

APPENDIX B

User Interface for existing systems

	Basic carpool groups	Standard carpool groups	Premium carpool groups
Industry-leading ride matching technology.	✓	✓	✓
Your own carpool group web portal.	✓	✓	✓
Restrict group membership based on e-mail address domains, if desired.	✓	✓	✓
Restrict matching only within your group, or allow matching with outside users.	✓	✓	✓
Maintain a drop-down list of common origins and destinations.	✓	✓	✓
Free support.	✓	✓	✓
Unlimited geocoding and maps.	✓	✓	✓
Exclusive sponsorship (no third-party advertising).	✓	✓	✓
Enhance your group's portal page and banners with your logo, custom texts, images, and links.	✗	✓	✓
Add your own carpool program documents as needed in addition to the standard CarpoolWorld terms-of-use.	✗	✓	✓
Guaranteed-ride-home program module.	✗	✓	✓
Additional configuration options to tune the system for your group, including social networking preferences.	✗	✓	✓
Add your own points of interest on the maps shown to your users.	✗	✓	✓
Promote your carpool group using our Invitations module, with optional individual or group registration codes.	✗	✓	✓
Automated periodic group activity reports.	✗	✓	✓
Group pageview statistics.	✗	✓	✓
Survey and report your users' trip logs.	✗	✓	✓
Analyze your group's transportation efficiency and carbon footprint over time and compared to regional statistics.	✗	✓	✓
Twilio integration	✗	✓	✓
Twitter integration	✗	✓	✓
Free administrator support.	✗	✓	✓
Import and export membership data.	✗	✗	✓
Send newsletters to your community to inform them about rules, rewards and your latest news.	✗	✗	✓
Notify and organize your community immediately in the event of any planned or unplanned disruption to your normal transportation network or office locations.	✗	✗	✓
Single sign-on integration.	✗	✗	✓
Native iPhone app, private and co-branded.	✗	✗	✓
Recommended for:	Private schools and academies, businesses, other organizations, and institutions with fewer than 200 commuters	Private schools and academies, businesses, other organizations, and institutions looking for standard functionality	Larger companies, hospitals, colleges and universities, and regional transportation authorities
Pricing	$4.99/month per 200 users	$49/month per 1,000 users	Please contact us for pricing information
	VIEW DETAILS	VIEW DETAILS	VIEW DETAILS

Figure 1 Carpool group for subscription users – CarpoolWorld

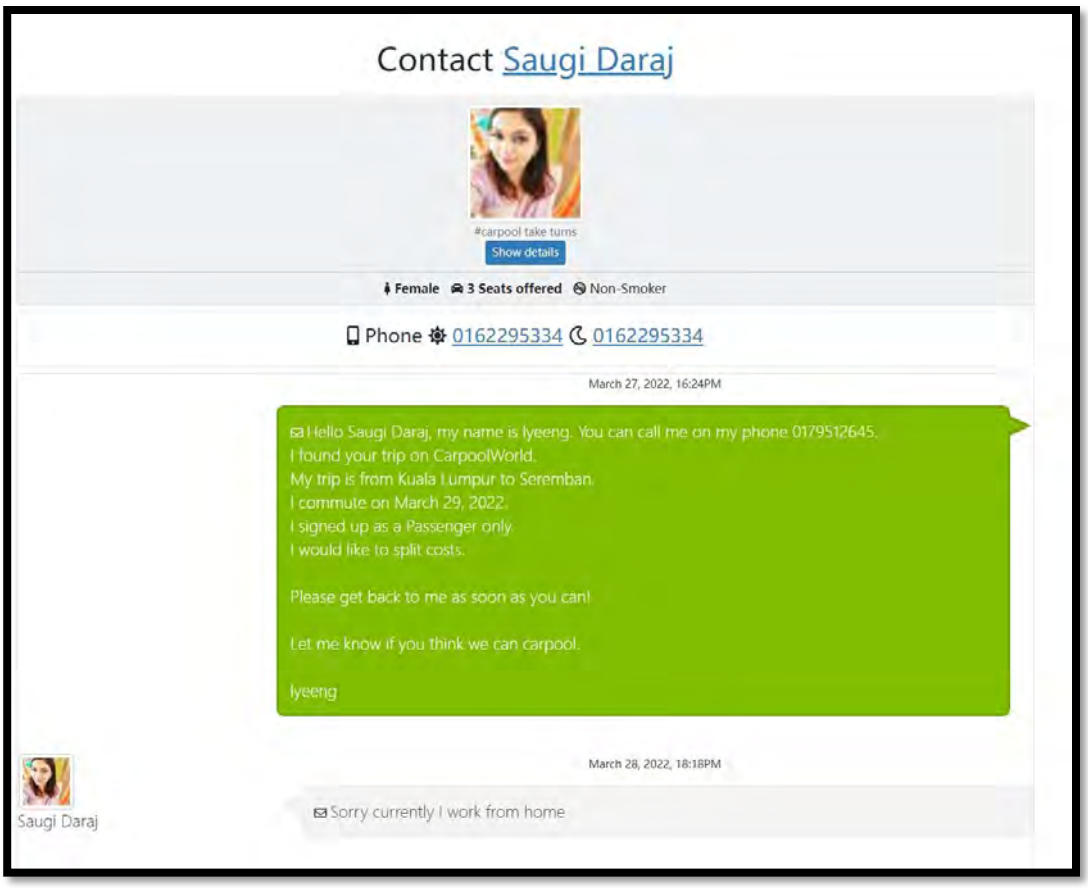


Figure 2 User Interface of Private Message to Driver – CarpoolWorld

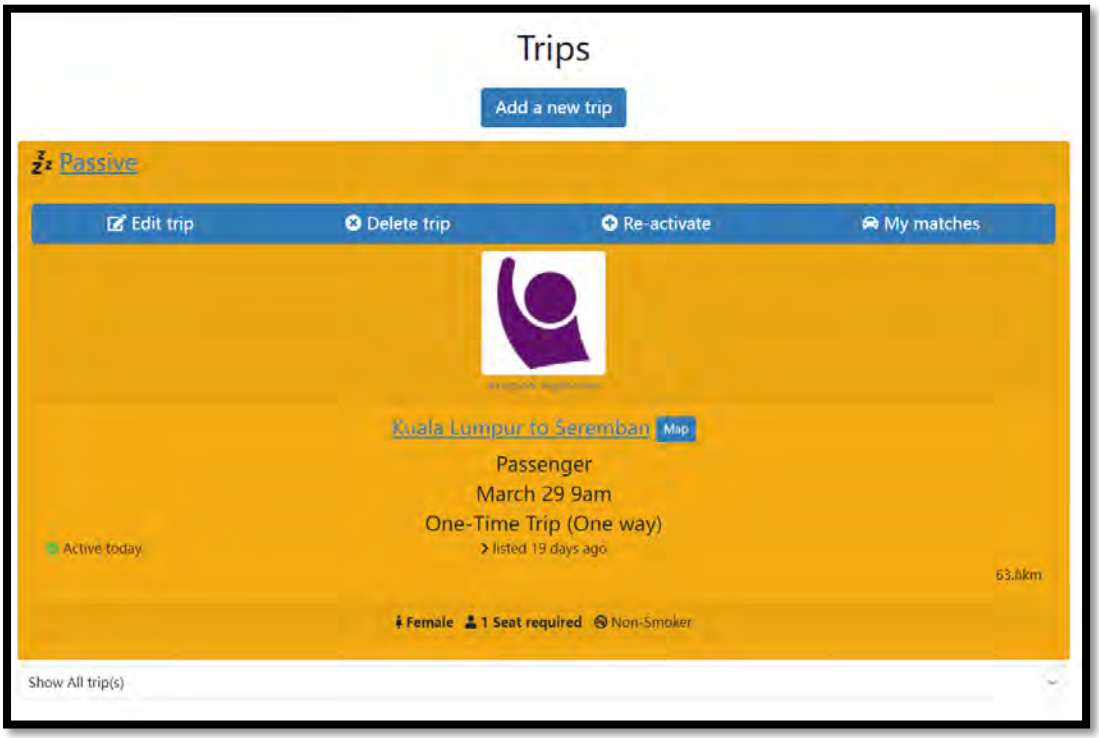


Figure 3 User Interface of Manage Post - CarpoolWorld

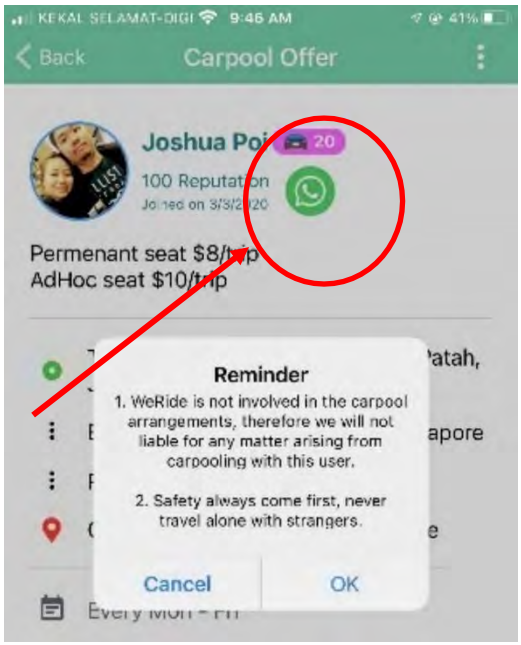


Figure 4 Methods for communicating with the driver -We Ride

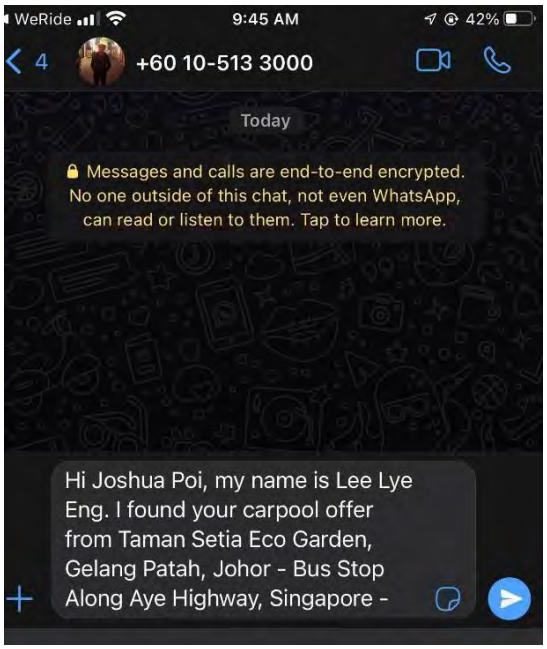


Figure 5 Integrate with WhatsApp communication - WeRide

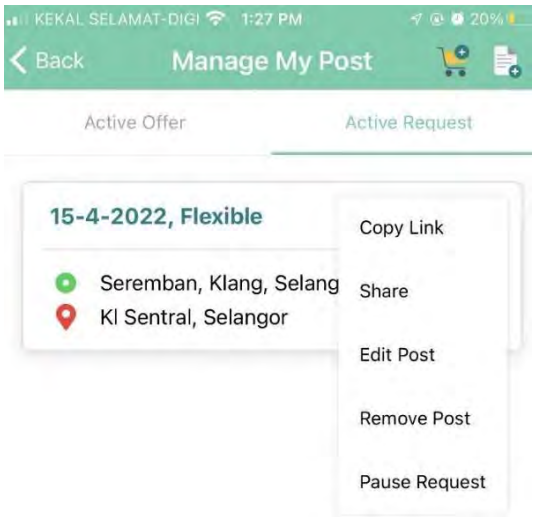


Figure 6 User Interface of Manage Post - WeRide

Figure 7 User interface of carpool listing available in group -Zipshare

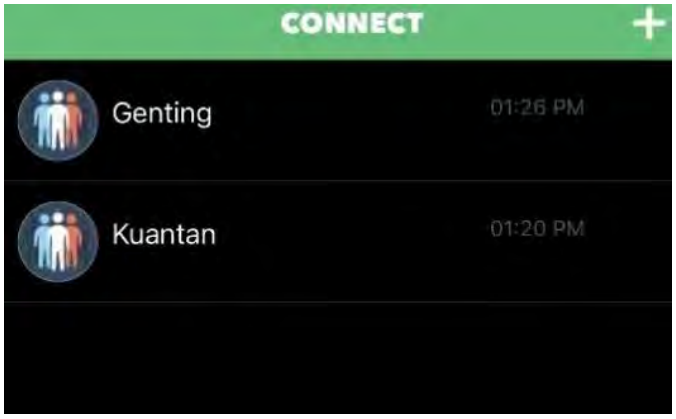


Figure 8 User Interface of Group Listing – Zipshare

Figure 9 Group creation: selecting members - Zipshare

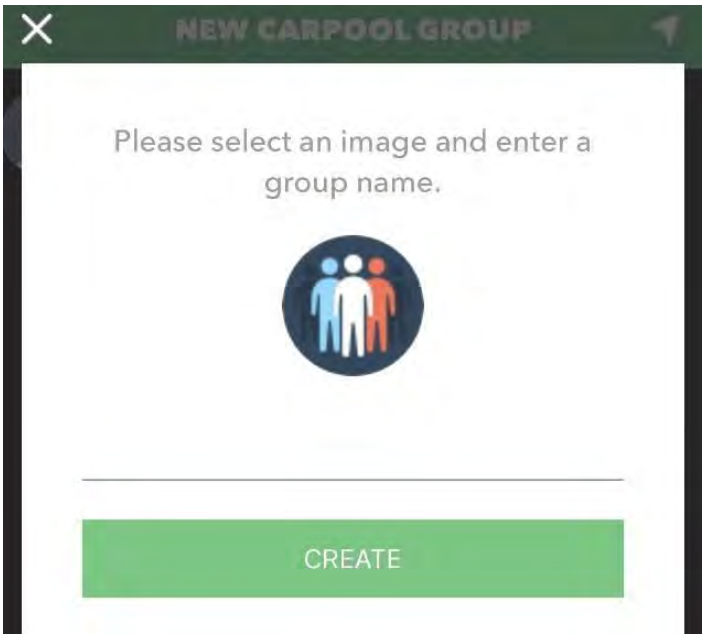


Figure 10 User Interface for Entering Group Name – Zipshare

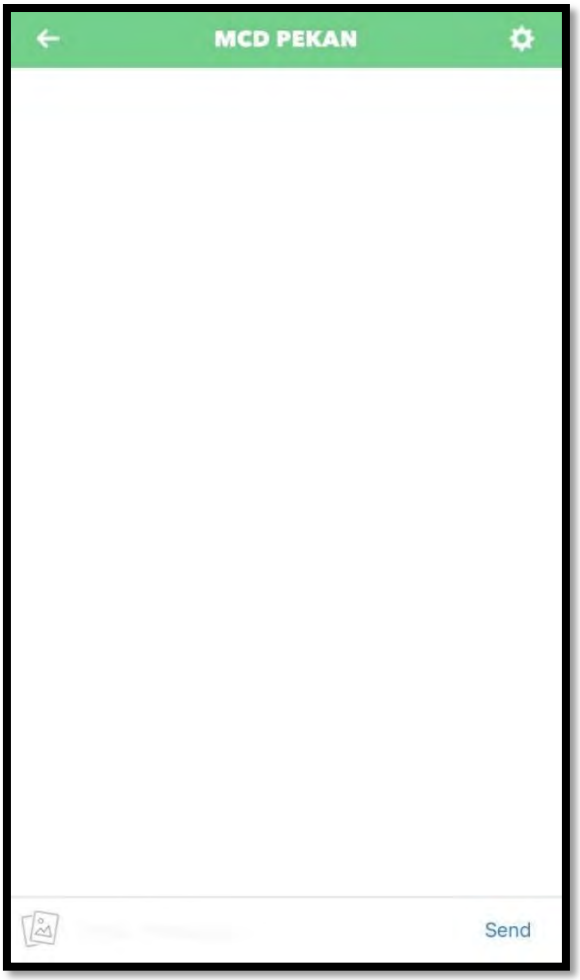


Figure 11 Group Chat for "MCD PEKAN" - Zipshare

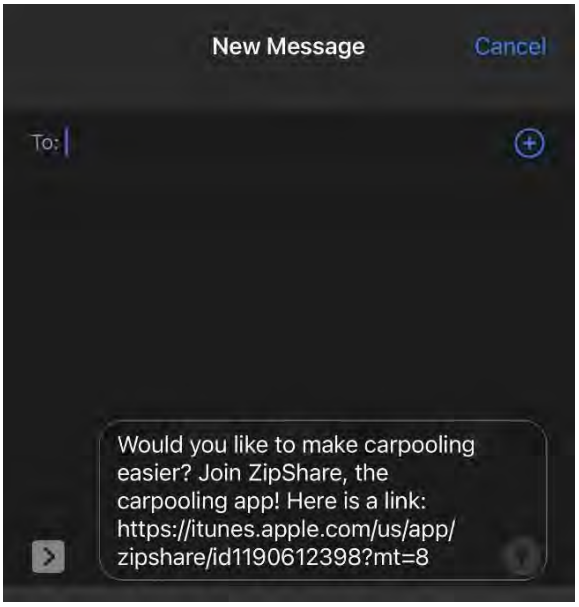


Figure 12 Dissemination of invitation links – Zipshare

APPENDIX C

Software requirement specification

2022

SOFTWARE REQUIREMENT SPECIFICATION (SRS)

[UMPOOL: A CARPOOLING SYSTEM]

DOCUMENT APPROVAL

	Name	Date
Authenticated by: Name: Lee Lye Eng	Lee Lye Eng	20/2/2023
Approved by:	Dr Ahmad Fakhri Bin Ab. Nasir	20/2/2023

Client		

Software : Microsoft Office 2016

Archiving Place : Google Drive

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

CHAPTER 1

1.1 PROJECT DESCRIPTION

UMPool: A Carpooling System is a web-based application is a system that facilitates carpooling services for students and staff at Universiti Malaysia Pahang. There are three (3) users in the system which are admin, driver, and passenger. Users are allowed to create and accept other carpool offer when needed. When the offer is accepted by passengers, passengers will proceed to payment to get carpool offer confirmation. After the payment is completed, users (drivers and passengers) can communicate with one another by scanning the WhatsApp QR code provided at the upcoming carpool listing. At the same time, admin will hold the payment until driver and passengers have completed the carpool.

There are six (6) modules in the system which including manage user login, manage profile, manage carpool, manage payment, manage review, and manage driving verification. For manage user login, users are allowed to login by email or multi-factor authentications (MFA) which are Facebook Login and Google Sign-in. Users do not have account requires to register and then proceed to login.

For the manage profile module, all users enable to update and view their profile information. After user make changes on their profile, the system will display the latest profile page.

For the manage carpool module, both driver and passenger can create, accept, update, delete carpool offer. The latest carpool offer displays when users made changes to the carpool offer. In addition, this module displays the carpool listing for user to search, and filter based on their preference.

For manage payment, passenger is required to complete the payment to get confirmation of the offer. There are two (2) payment method provide in the system which are Paypal and FPX. Users are not allowed to cancel their offer after payment is completed.

Admin will hold the payment and release to driver when the carpool offer status is changed from upcoming to completed.

For manage review, passengers are able to create, edit, delete, and view their reviews once their trip has ended.

For manage driving verification, driver can create, update, delete, and view their driving details including driving license and driving period. Admin will verify the driving verification by approving and rejecting the driver application.

1.2 SYSTEM IDENTIFICATION

Format: XXX-CB19092-XXXXX-XXXX-VX

System ID: SRS-CB19092-UACS-2022-V1

Document: Software Requirement Specification

Developer ID: CB19092

System Name: UMPool: A Carpooling System

System Abbreviation: UACS

Development Year: 2022

Version: 1

1.3 CONTEXT DIAGRAM

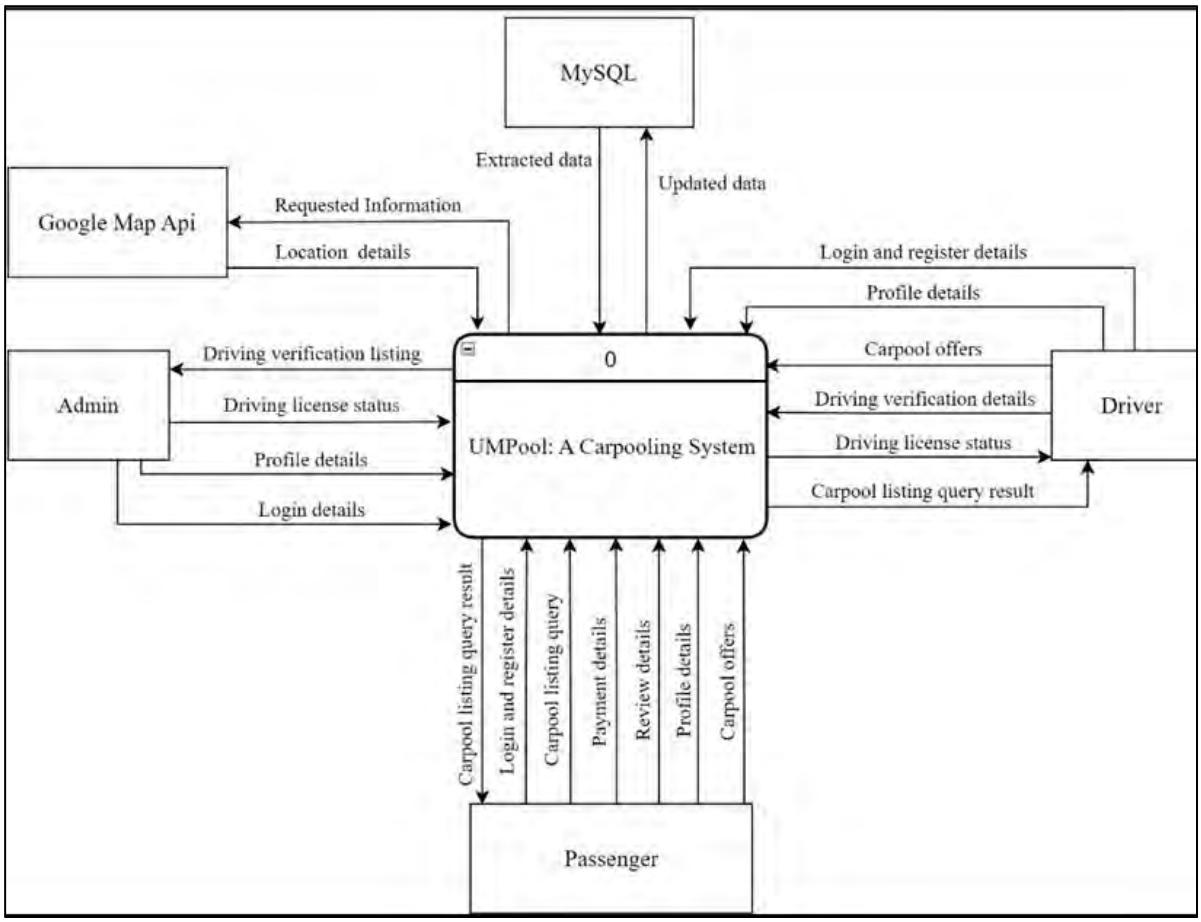


Figure 1 Context diagram for UACS

This system consists of four (4) entities, which are admin, driver, passenger, and payment gateway. Admin allows to input their login details, profile details, and driving license status while the system will display driving verification listing. Passengers allow to input login and register details, carpool listing query, payment details, review details, profile details and carpool offer while the system will display the carpool listing query result. Driver is allowed to input carpool listing query, login and register details, profile details, carpool offers, driving verification details while the system will display driving licence status and

carpool listing query result. Payment gateway can input the payment details which the system will output the payment result.

1.4 DATA FLOW DIAGRAM

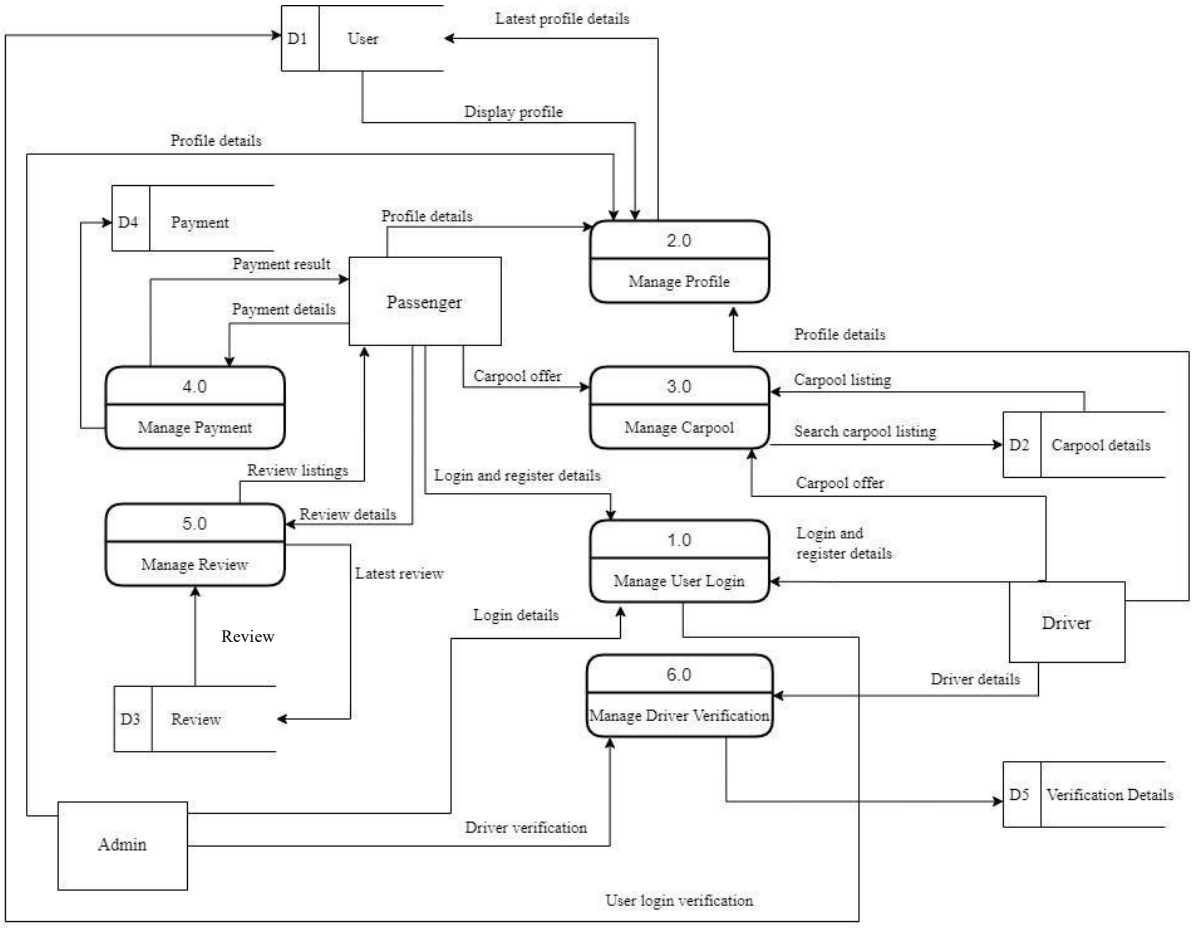


Figure 2 Data Flow Diagram of UACS

Users are required to login in order to access the system. Users without an account must register then proceed to the login. The system will verify the login details entered with the User datastore. If the credentials are correct, the system will redirect users to the manage profile page.

For the manage profile module, users are allowed to update and view their profile details. The latest profile details will be saved in the User data store while the data store will display the profile to user.

For the manage carpool module, both driver and passenger are allowed to create, update, accept and delete carpool offer. The system will search for available carpool listings from the Carpool details data store and display to user.

For the manage payment module, passengers will input their payment details and the system will display the payment result to user.

For the manage review module, passengers will input the review details and system will display the review listing from the review data store.

For the manage driving verification module, driver will input the driver details and the admin will approve and reject the driving verification to the system.

CHAPTER 2

2.1 USE CASE DIAGRAM AND DESCRIPTION

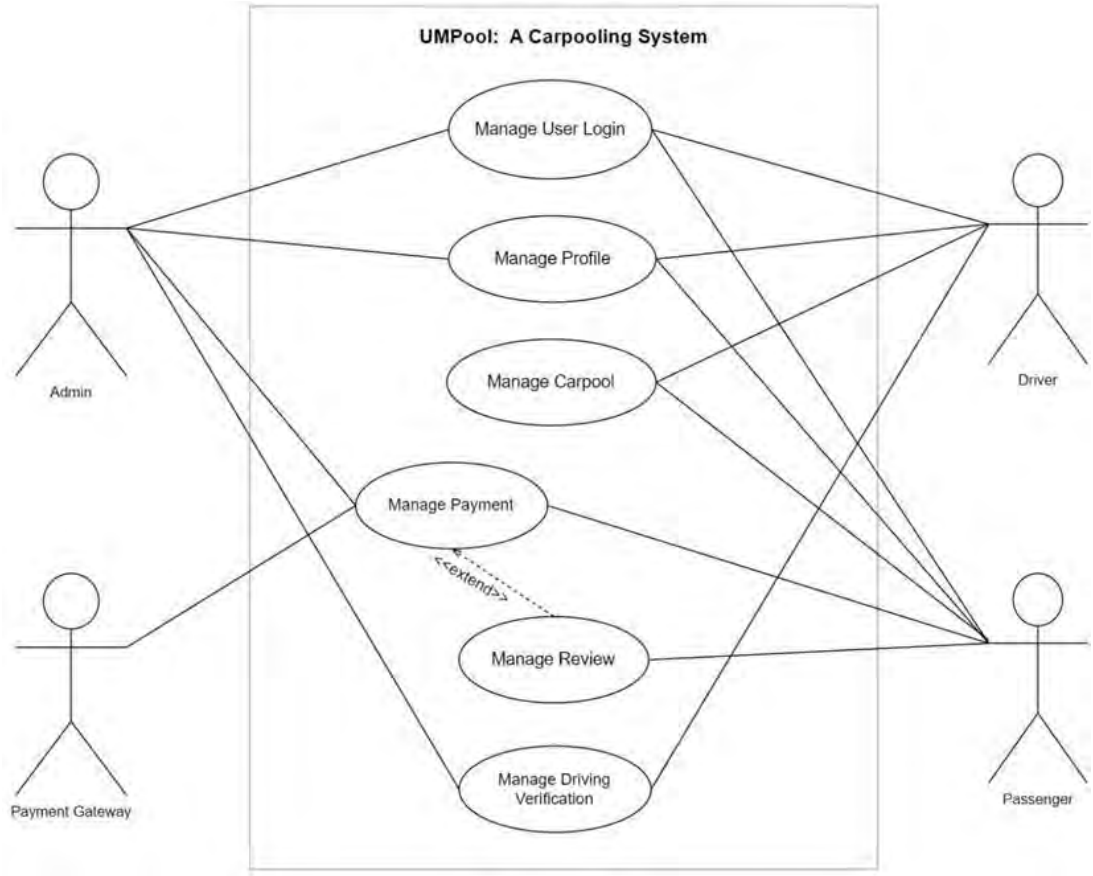


Figure 2.1 Use Case Diagram of UACS

2.1.1 Manage User Login

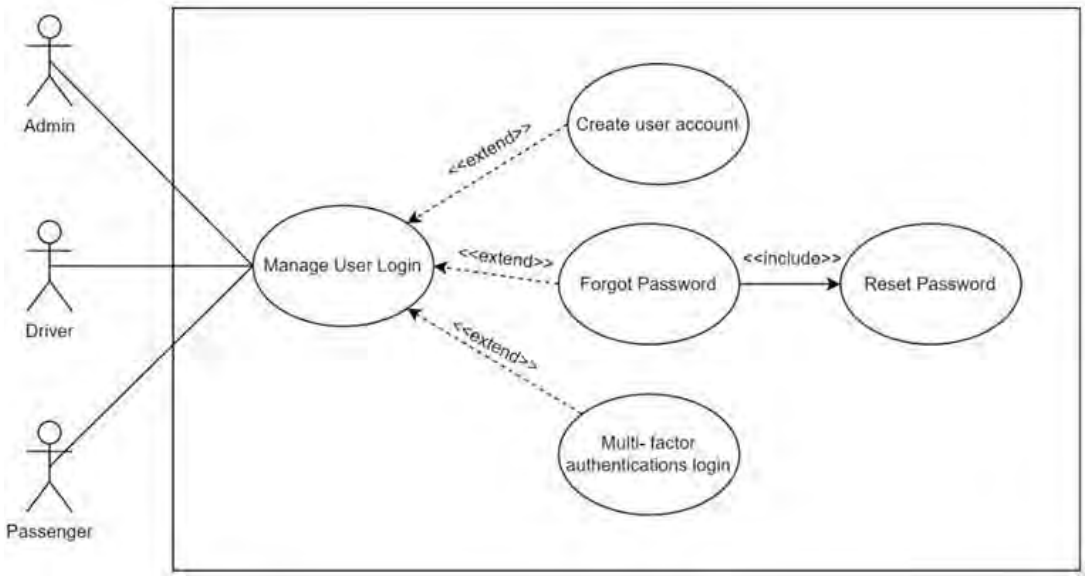


Figure 2.2 User case diagram of the Manage User Login

Table 2.1 Use Case Description of the Manage User Login

Use Case ID	UACS_UC01
Use Case Name	Manage user login
Brief Description	This use case explains the user login process for drivers and passengers. Next, the use case describes how an admin logs into their account. With this use case, users are allowed to use the functionality after login to the system.
Actor	Admin, driver, and passenger

Pre-condition	The server is working normally. Admin have registered for an account.
Basic Flow	1. The use case starts with user login page. [A1: Create user account] [A2: Multi-factor authentications login] 2. User enters their account credentials. (email and password) to log in their account. [A3: Forgot password] 3. System will begin to account verification process in the database. 4. System redirects to main menu page. 5. The use case end.
Alternative Flow	[A1: Create user account] 1. Users select <<Sign Up>> button to register their account. 2. System redirects to the registration page. 3. Users enter register details and select <<sign up>> button. 4. The use case continues to step 1 in the basic flow. [A2: Multi-factor authentications login]

	1. Users select a multi-factor authentication method on the user login page. 2. System starts to verify user account. 3. The use case continues to the step 4 in the main flow. [A3: Forgot password] 1. Users select forgot password hyperlink. 2. System redirects to the reset password page. 3. Users are required to fill in email address and submit. 4. The system will generate a reset password verification email and send it to the email address entered. 5. Users are required to fill in the new password and submit it. 6. The use case continues to the step 1 in the main flow.
Exception Flow	[E1: Invalid password] 1. The system will reject user login. 2. The system will pop out an error message. 3. User is asked to try again their password or select

	the forgot password option of the login screen. 4. The use case end.
Post Condition	User is logged in to the system and the system displays the main menu page.

2.1.2 Manage Profile

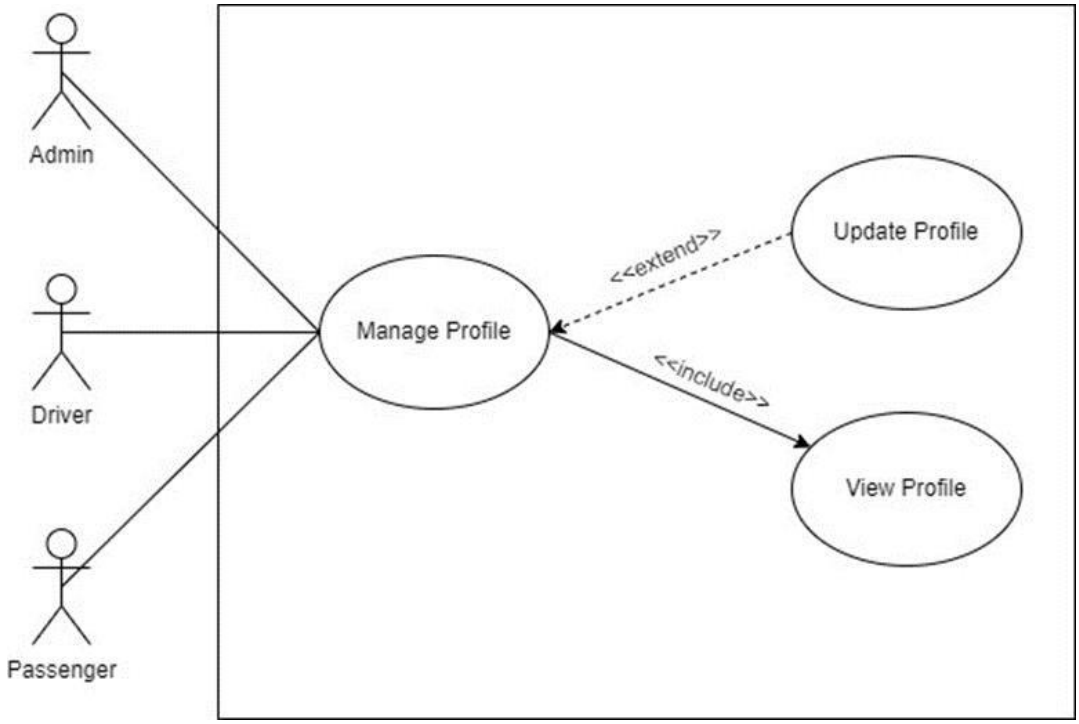


Figure 2.3 User case diagram of the Manage Profile

Table 2.2 Use Case Description of the Manage Profile

Use Case ID	UACS_UC02

Use Case Name	Manage profile
Brief Description	This use case explains the process of manage profile for admin, driver, and passenger. It aims at allowing users to make changes to their profile information as needed. The latest profile is viewable after they make changes to their profiles.
Actor	Admin, driver, and passenger
Pre-condition	User logs in to their account successfully.
Basic Flow	1. This use case begins when the user navigates to the manage profile page. 2. System displays the list of user information. 3. Users select the specific user information and select operation. [A1 : Update profile] 4. The system display the latest user profile. 5. The use case end.
Alternative Flow	[A1 : Update profile] 1. Users select update button following the selection of the information. 2. System display the update profile page. 3. Users fill the text entry box with the updated information.

	4. Users select the save button to update their profile. 5. The use case continues to the step 4 in the main flow.
Exception Flow	-
Post Condition	Latest profile is displayed.

2.1.3 Manage Carpool

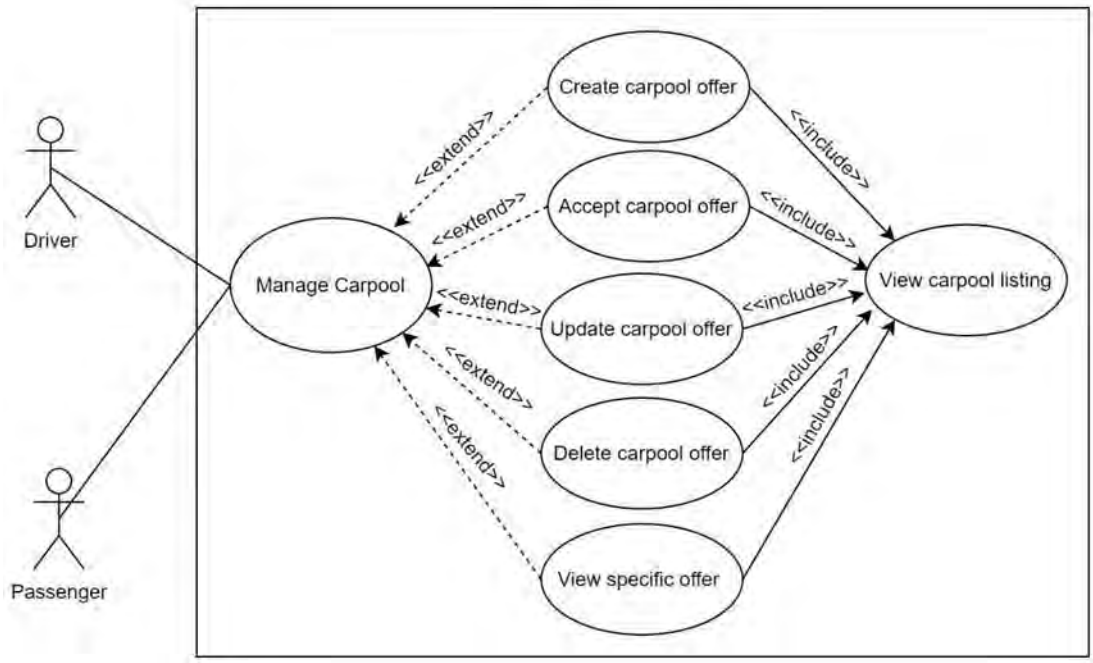


Figure 2.4 Use case diagram for Manage Carpool

Table 2.3 Use case description for Manage Carpool

Use Case ID	UACS_UC03
Use Case Name	Manage Carpool
Brief Description	This use case explains the process of manage carpool for driver, and passenger. It aims at allowing users to make changes on the carpool offer as needed. In this case, both users can create the carpool offer and wait for others to accept it. As an example, passengers can create an offer based on their requirements and wait for drivers to accept that offer.
Actor	Driver and passenger
Pre-condition	User logs in to their account successfully.
Basic Flow	[Driver] 1. This use case begins when the user navigates to the manage carpool page. 2. System displays the carpool listing. [A1: Create carpool offer] [A2: Accept carpool offer] [A3: Update carpool offer] [A4: Delete carpool offer] [A5: View specific offer] 3. The system displays the latest carpool offers status, including offers created by themselves and accepted by passengers.

	4. The use case end. [Passenger] 1. This use case begins when the user navigates to the manage carpool page. 2. System displays the carpool listing. [A1: Create carpool offer] [A2: Accept carpool offer] [A3: Update carpool offer] [A4: Delete carpool offer] 3. The system displays the latest carpool offers status, including offers created by themselves and accepted by driver. 4. The use case end.
Alternative Flow	[A1 : Create carpool offer] 1. Users select create button on the manage carpool page. 2. System display the create carpool page. 3. Users fill the text entry box with the carpool offer information. 4. Users select the add button to create their carpool offer. 5. The use case continues to the step 3 in the main flow. [A2 : Accept carpool offer]

	1. Users select accept button following the selection of the carpool listings. 2. The system displays a message to confirm the user wants to accept this offer. 3. Users click on the confirm button to accept an offer from another user role. 4. System display accept offer successfully message. 5. The use case continues to the step 3 in the main flow. [A3 : Update carpool offer] 1. Users select update button following the selection of the carpool listings. 2. System display the update carpool offer page. 3. Users fill the text entry box with the updated information. 4. Users select the save button to update their carpool offer. 5. System display update carpool offer successfully message. 6. The use case continues to the step 3 in the main flow. [A4 : Delete carpool offer] 1. Users select delete button following the selection of

	the information. 2. The system displays a alert message to confirm the user wants to delete this offer. 3. Users select the confirm button to delete the carpool offer. 4. System displays the deleted successful message. 5. The use case continues to the step 3 in the main flow. [A5 : View specific offer] 1. Users select view button following the selection of the information. 2. The system displays the carpool offer details for the selected carpool. 3. Users select the back button to redirect back to carpool listing. 4. The use case continues to the step 3 in the main flow.
Exception Flow	-
Post Condition	Latest carpool offers status are displayed.

2.1.4 Manage Payment

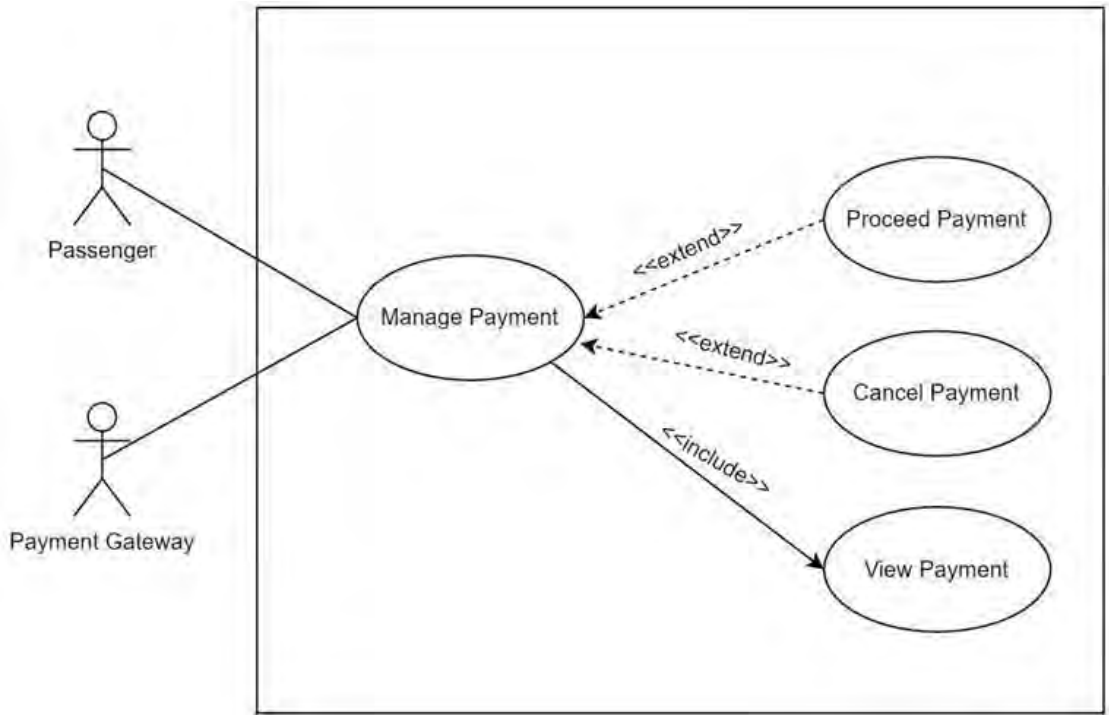


Figure 2.5 Use case diagram for Manage Payment

Table 2.4 Use case description for Manage Payment

Use Case ID	UACS_UC04
Use Case Name	Manage Payment
Brief Description	This use case explains the process of manage payment for passenger. It aims at allowing users to make changes on payment. User can view the latest payment after the driver has accepted the carpool offer or after users have accepted the deal at the carpool listing. Users cannot

	cancel / refund their payment after they have completed payment.
Actor	Passenger, payment gateway
Pre-condition	1. User logs in to their account successfully. 2. Passengers accept the carpool offer.
Basic Flow	1. This use case begins when the user navigates to the manage payment page. 2. System displays the view payment page. [A1: Accept payment] [A2: Cancel payment] 3. System displays the latest payment status. 4. The use case end.
Alternative Flow	[A1: Accept payment] 1. Users select <<accept>> button at the view payment page. 2. System displays accept payment page. 3. Users fill the payment details and click <<pay>> button. 4. System validate the payment. [E1: Payment failed] 5. System display payment successfully message.

	6. The use case continues to the step 3 in the main flow. [A2: Cancel payment] 1. Users select <<cancel>> button at the view payment page. 2. System display confirmation cancel payment message. 3. Users click <<confirm>> button to cancel payment. 4. System display the cancel payment successfully message. 5. The use case continues to the step 3 in the main flow.
Exception Flow	[Payment failed] 1. System displays error message. 2. The user is asked to try again to make payment. 3. The use case end.
Post Condition	The latest payment status is updated and displayed.

2.1.5 Manage Review

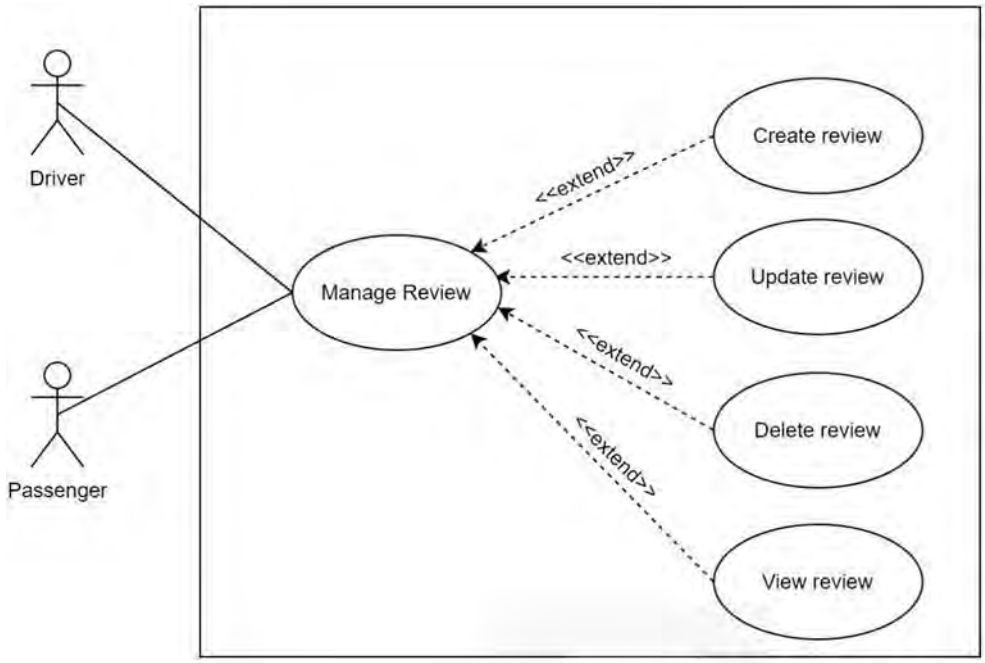


Figure 2.6 Use case diagram for Manage Review

Table 2.5 Use case description for Manage Review

Use Case ID	UACS_UC05
Use Case Name	Manage Review
Brief Description	This use case explains the process of manage review for passenger. It aims at allowing users to make changes to their review as needed. Once the user has made changes, the latest list of reviews will be updated and displayed.

Actor	Passenger
Pre-condition	1. User logs in to their account successfully. 2. The passenger has made payment and the carpool offer is completed.
Basic Flow	1. This use case begins when the user navigates to the manage review page. 2. System displays the view review page. [A1: create review] [A2: update review] [A3: delete review] [A4: view review] 3. System displays the latest review status. 4. The use case end.
Alternative Flow	[A1: Create review] 1. Users select create button on the manage review page. 2. System display the create review page. 3. Users fill the text entry box with the review of carpool offer. 4. Users select the add button to create their review. 5. The use case continues to the step 3 in the main flow. [A2: Update review]

	1. Users select update button following the selection of the review list. 2. System display the update review page. 3. Users fill the text entry box with the updated information. 4. Users select the save button to update their review. 5. System display update review successfully message. 6. The use case continues to the step 3 in the main flow. [A3: Delete review] 1. Users select delete button following the selection of the information. 2. The system displays a alert message to confirm the user wants to delete this review. 3. Users select the confirm button to delete the review. 4. System displays the deleted successful message. 5. The use case continues to the step 3 in the main flow. [A4: View review] 1. Users select view button following the selection of the information. 2. The system displays the review details for the

	selected review. 3. Users select the back button to redirect back to review listing. 4. The use case continues to the step 3 in the main flow.
Exception Flow	-
Post Condition	Latest review list is displayed.

2.1.6 Manage Driving Verification

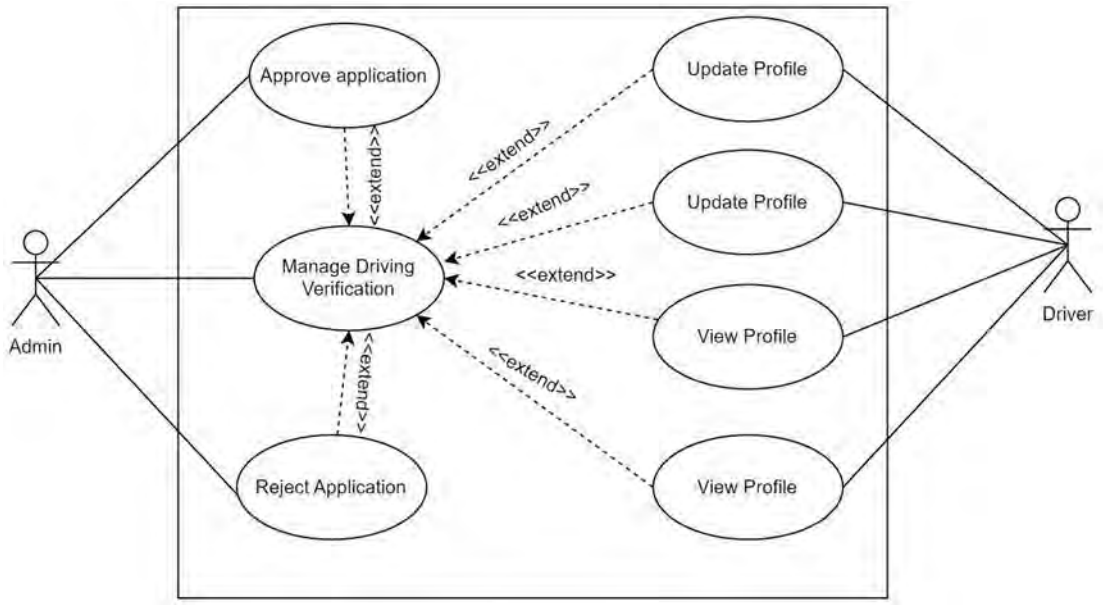


Figure 2.7 Use Case Diagram for Manage Driving Verification

Table 2.6 Use Case Description for Manage Driving Verification

Use Case ID	UACS_UC06
Use Case Name	Manage Driving Verification
Brief Description	This use case explains the process of manage driving verification for admin and driver. It aims at allowing driver to make changes to their driving verification. Admin will verify the driver application and approve it. The admin can reject the application if any misauthentication issue is detected.
Actor	Admin and driver
Pre-condition	User logs in to their account successfully.
Basic Flow	1. This use case begins when the user navigates to the manage driving verification page. 2. System displays the manage driving verification page. [A1: Create driving details] [A2: Update driving details] [A3: Delete driving details] [A4: Approve application] [A5: Reject application] 3. System displays the latest driving verification status. 4. The use case end.
Alternative Flow	[A1: Create driving details] 1. Driver selects <<create>> button on the manage driving verification page.

	2. System display the create driver details page. 3. Driver fills the text entry box with the driving details. 4. Driver selects the <<add>> button to create their driving details. 5. The use case continues to the step 3 in the main flow. [A2: Update driving details] 1. Driver selects <<update>> button following the selection of the driving details list. 2. System displays the update driving details page. 3. Driver fills the text entry box with the updated information. 4. Driver selects the <<save>> button to update their driving details. 5. System displays update driving details successfully message. [A3: Delete driving details] 1. Driver selects <<delete>> button following the selection of the information. 2. The system displays an alert message to confirm the user wants to delete this driving details. 3. Driver selects the <<confirm>> button to delete the

	driving details. 4. System displays the deleted successful message. 5. The use case continues to the step 3 in the main flow. [A4: Approve application] 1. Admin select and view the driver driving details. 2. Admin selects <<approve>> button to approve the application. 3. The use case continues to the step 3 in the main flow. [A5: Reject application] 1. Admin select and view the driver driving details. 2. Admin selects <<reject>> button to reject the application. 3. The use case continues to the step 3 in the main flow.
Exception Flow	-
Post Condition	Latest driving status is displayed.

2.2 SEQUENCE DIAGRAM

2.2.1 Manage User Login

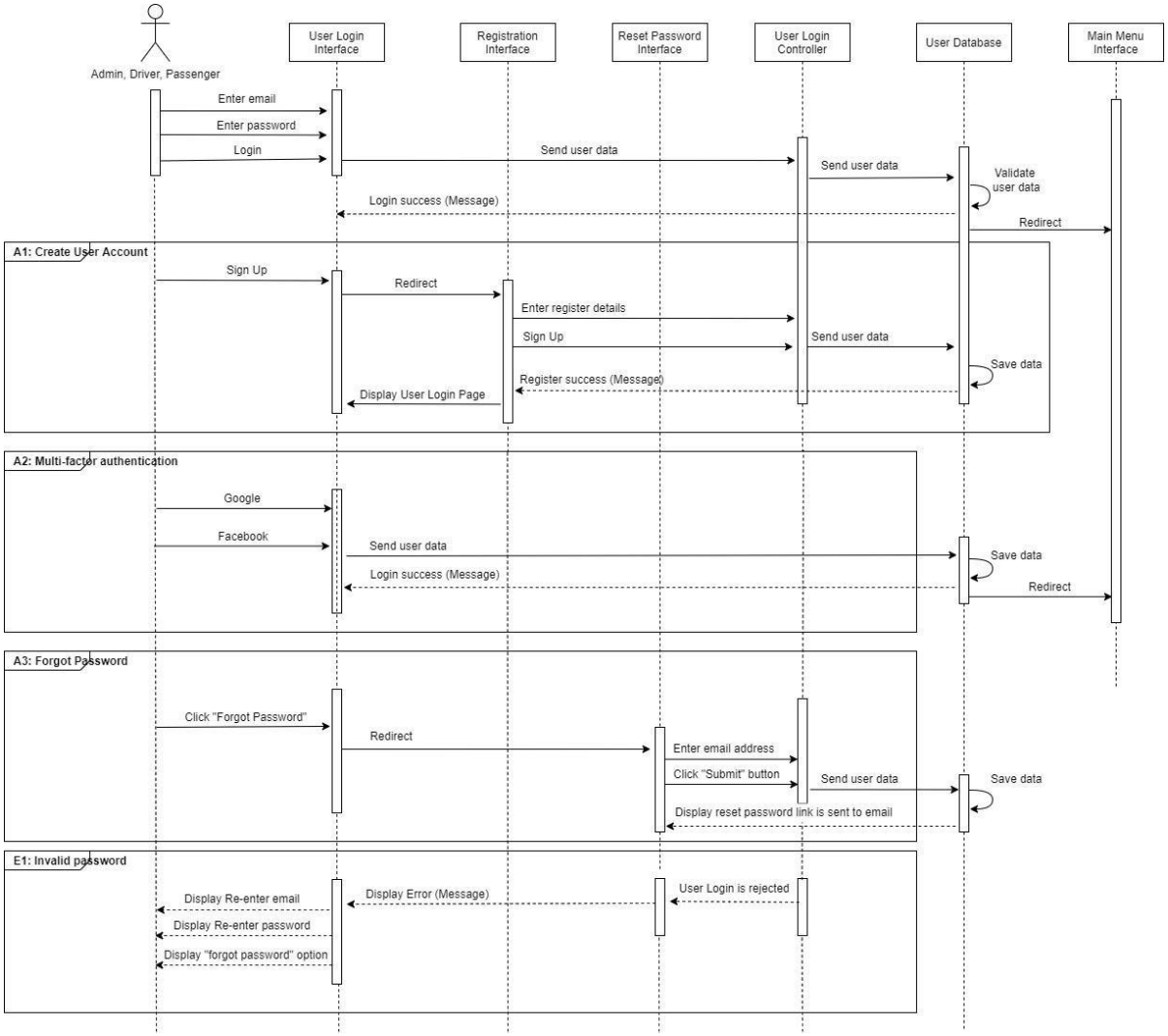


Figure 2.8 Sequence Diagram of Manage User Login

The sequence diagram of manage user login related to UACS_UC01 is illustrated in Figure 2.8. It involves three actors, including admin, driver, and passenger. This module contains three interfaces, one controller and a database that interact with users.

2.2.2 Manage Profile

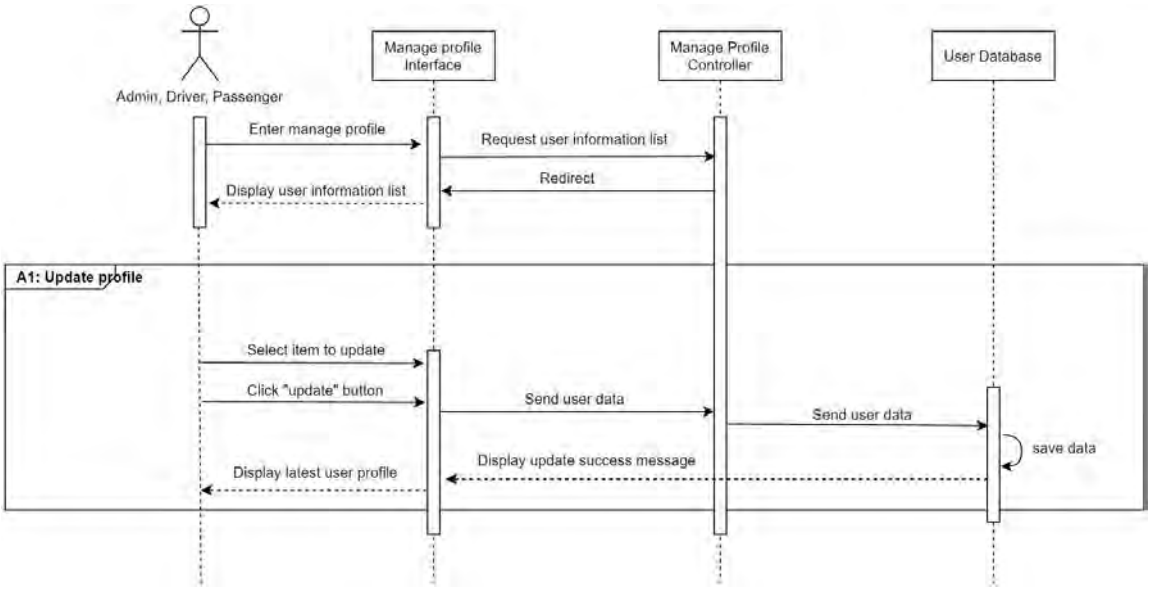


Figure 2.9 Sequence Diagram of Manage Profile

The sequence diagram of manage profile related to UACS_UC02 is illustrated in Figure 2.9. It involves three actors, including admin, driver, and passenger. This module contains one interface, one controller and a database that interact with users.

2.2.3 Manage Carpool

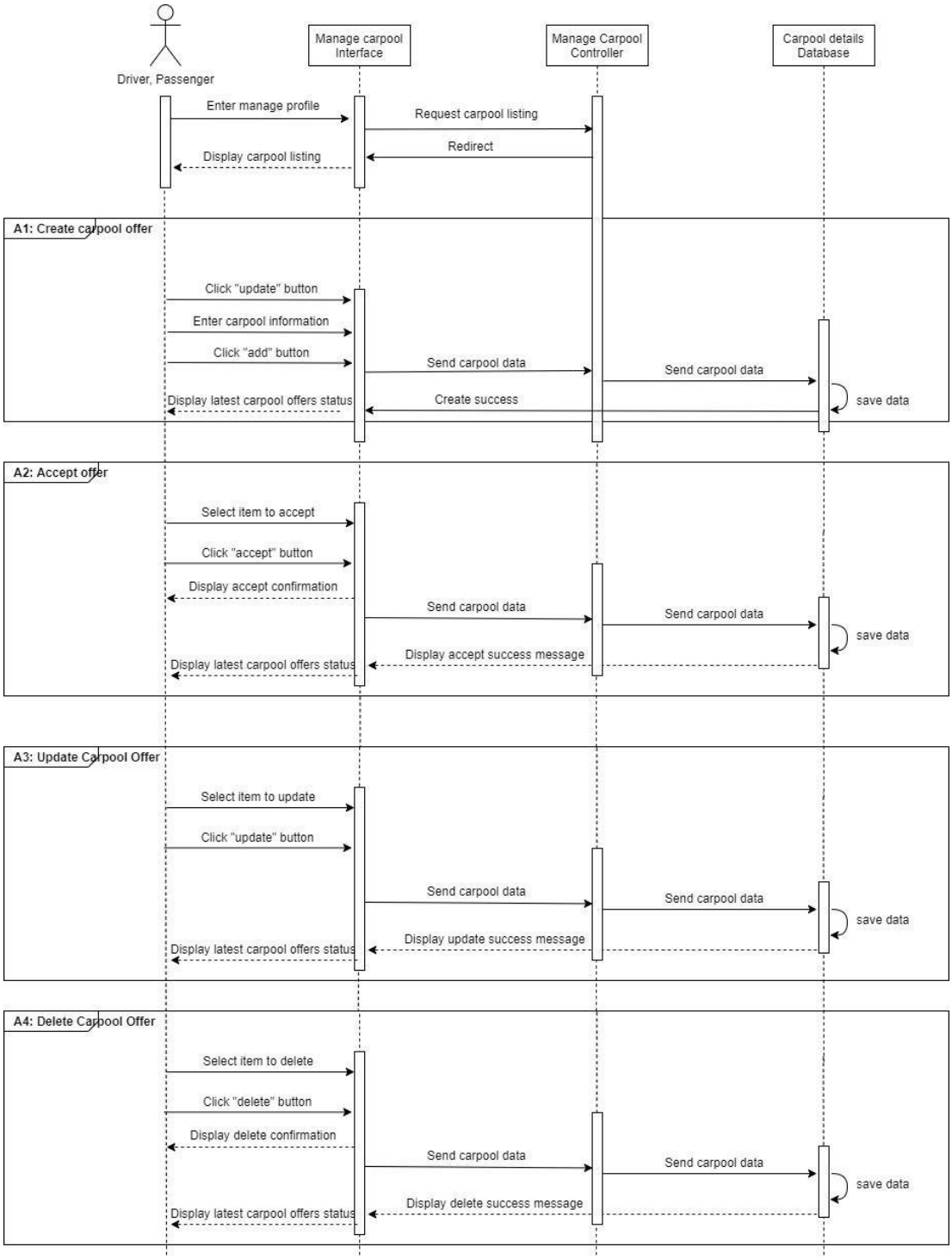


Figure 2.10 Sequence Diagram of Manage Carpool

The sequence diagram of manage carpool related to UACS_UC03 is illustrated in Figure 2.10. It involves two actors, including driver, and passenger. This module contains one interface, one controller and a database that interact with users.

2.2.4 Manage Payment

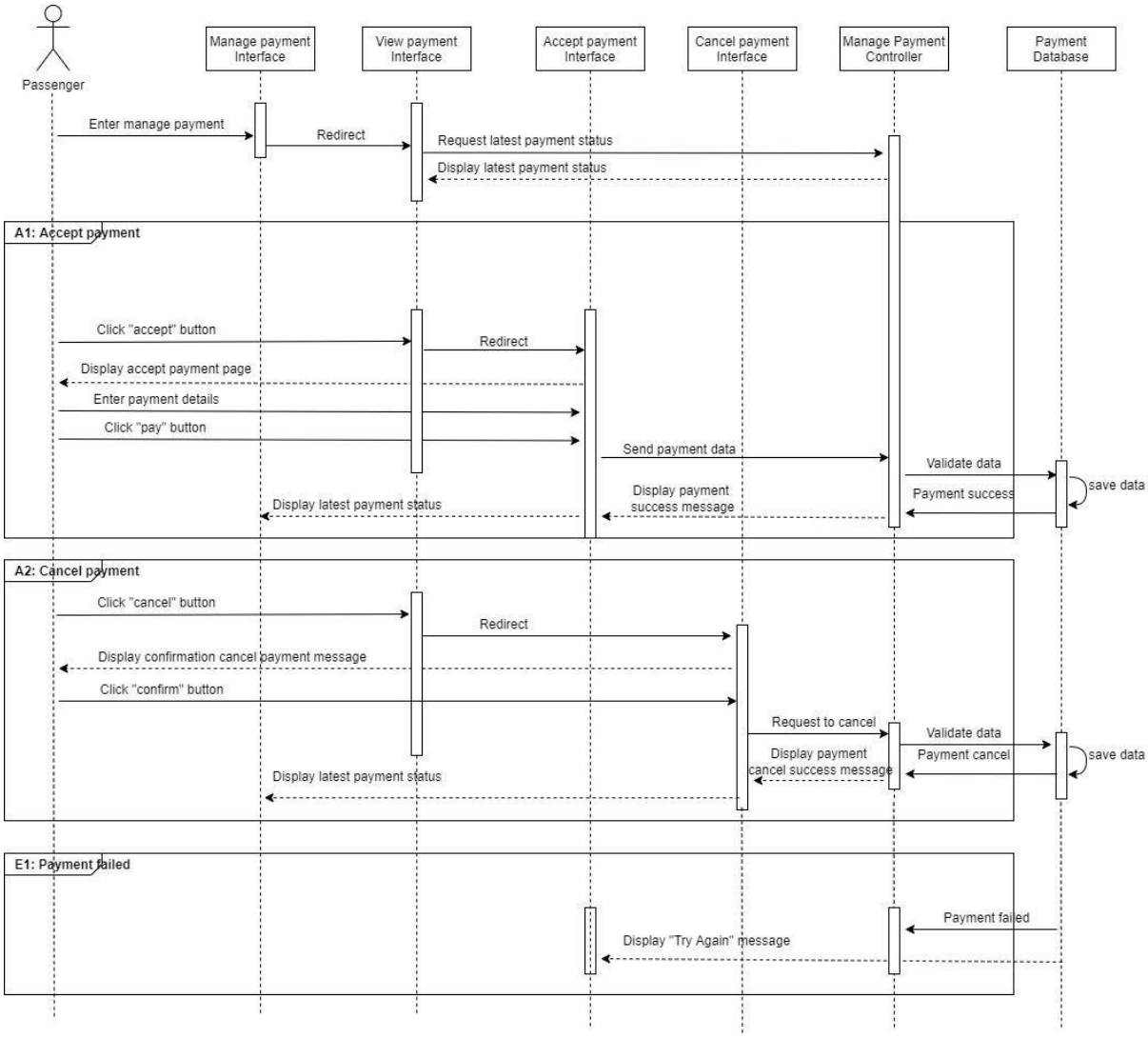


Figure 2.11 Sequence Diagram of Manage Payment

The sequence diagram of manage payment related to UACS_UC04 is illustrated in Figure 2.11. It involves passenger in managing this module. This module contains four interface, one controller and a database that interact with user.

2.2.5 Manage Review

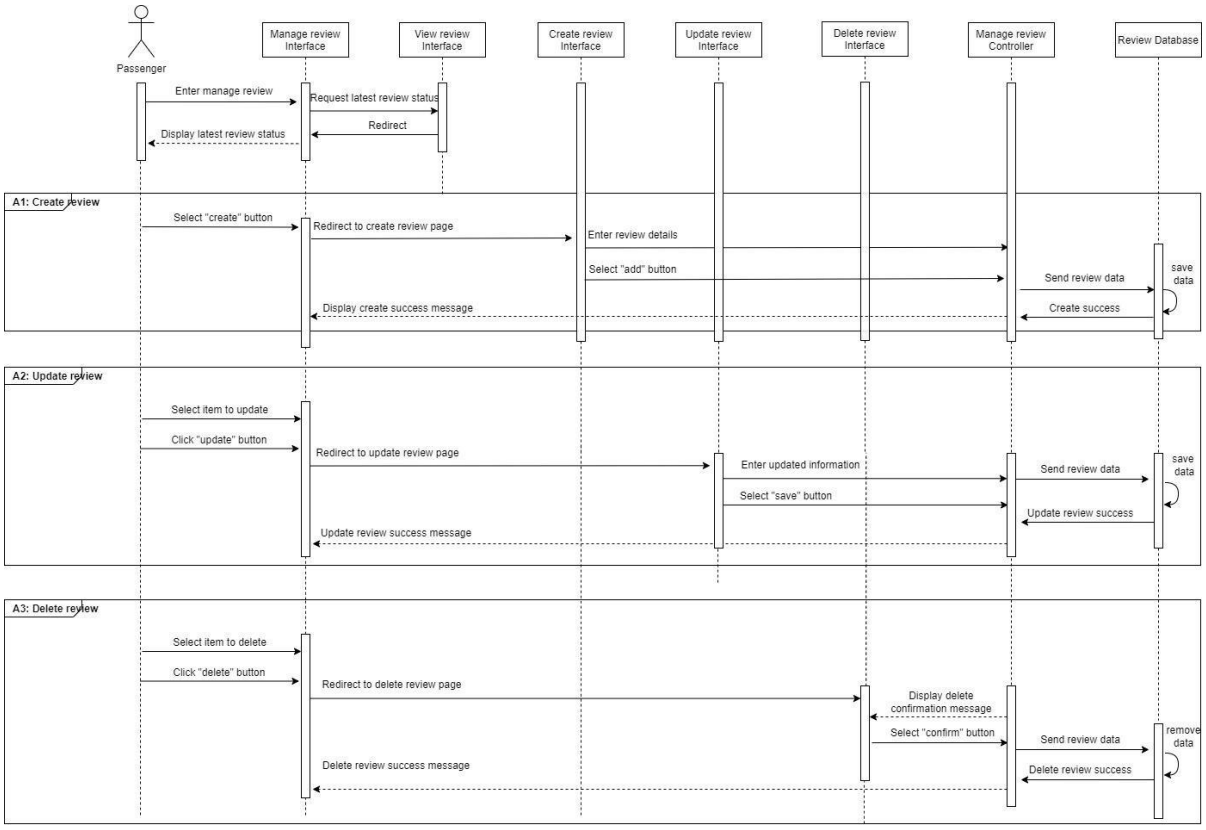


Figure 2.32 Sequence Diagram of Manage Review

The sequence diagram of manage review related to UACS_UC05 is illustrated in Figure 2.12. It involves passenger in managing this module. This module contains five interface, one controller and a database that interact with user.

2.2.6 Manage Driving Verification

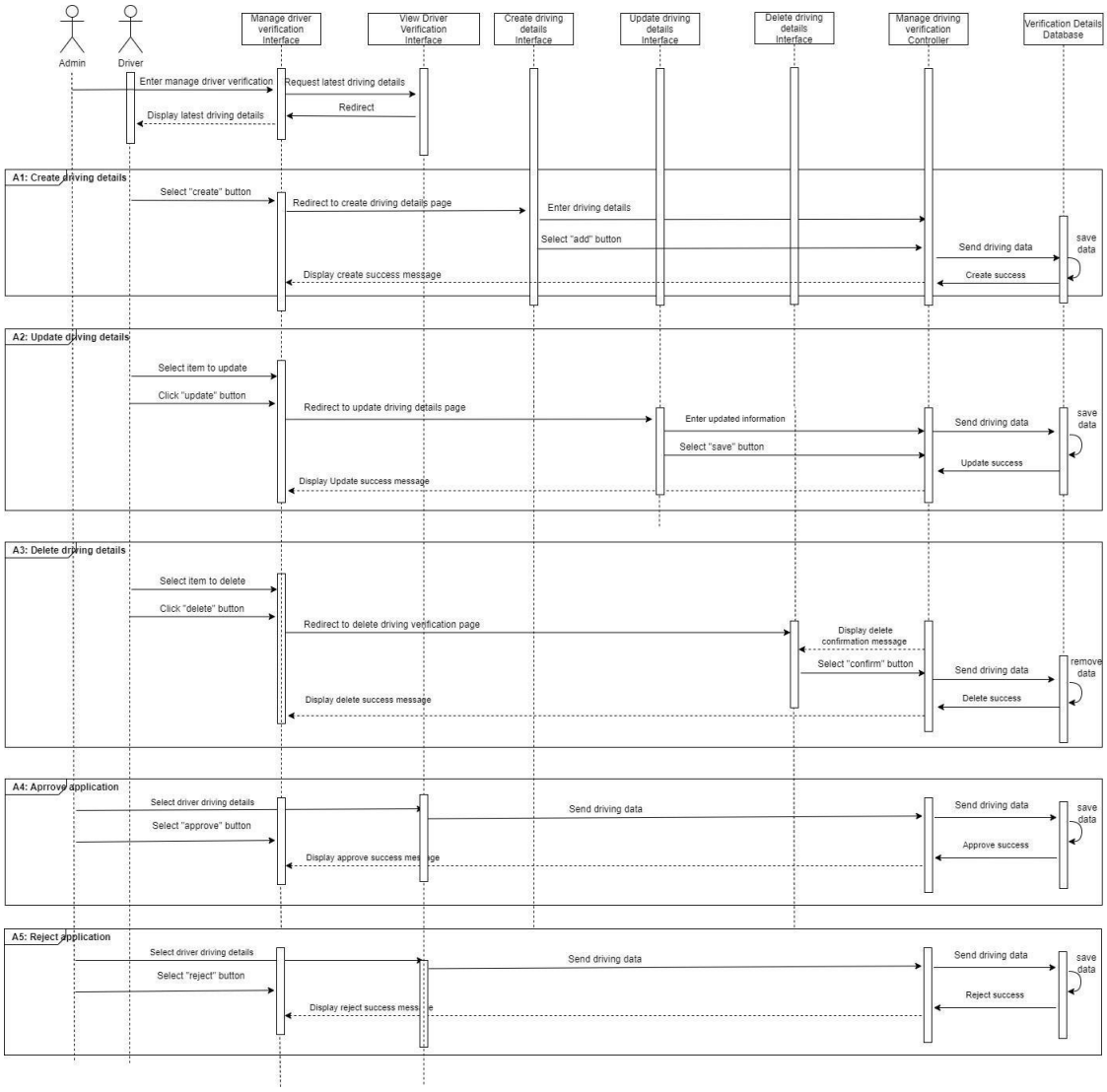


Figure 2.13 Sequence Diagram for Manage Driving Verification

The sequence diagram of manage driving verification related to UACS_UC06 is illustrated in Figure 2.13. It involves admin and driver in managing this module. This module contains five interface, one controller and a database that interact with user.

CHAPTER 3

3.1 INTERFACE DESIGN

Module 1: Manage User Login

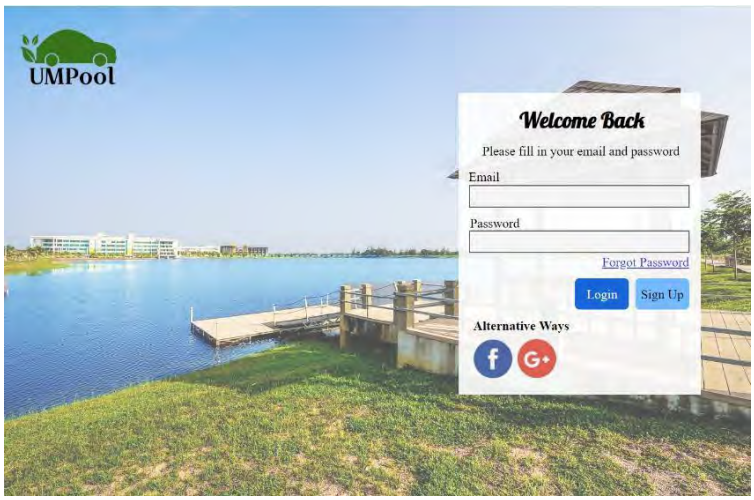


Figure 3.1 User login page for all users

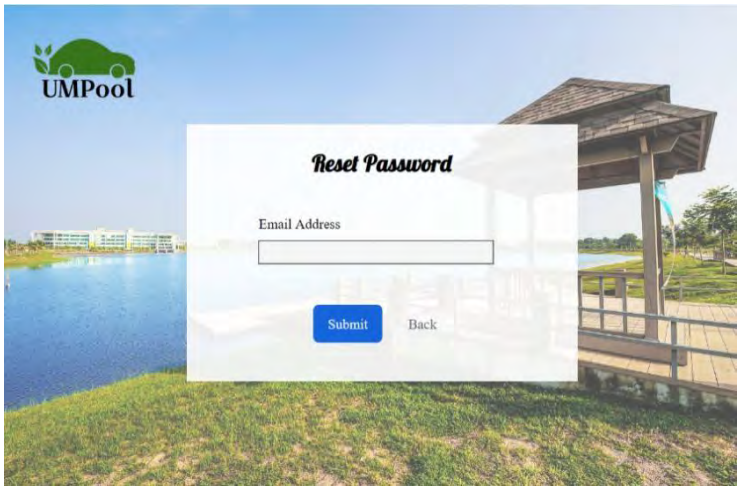


Figure 3.2 Reset password page for all users

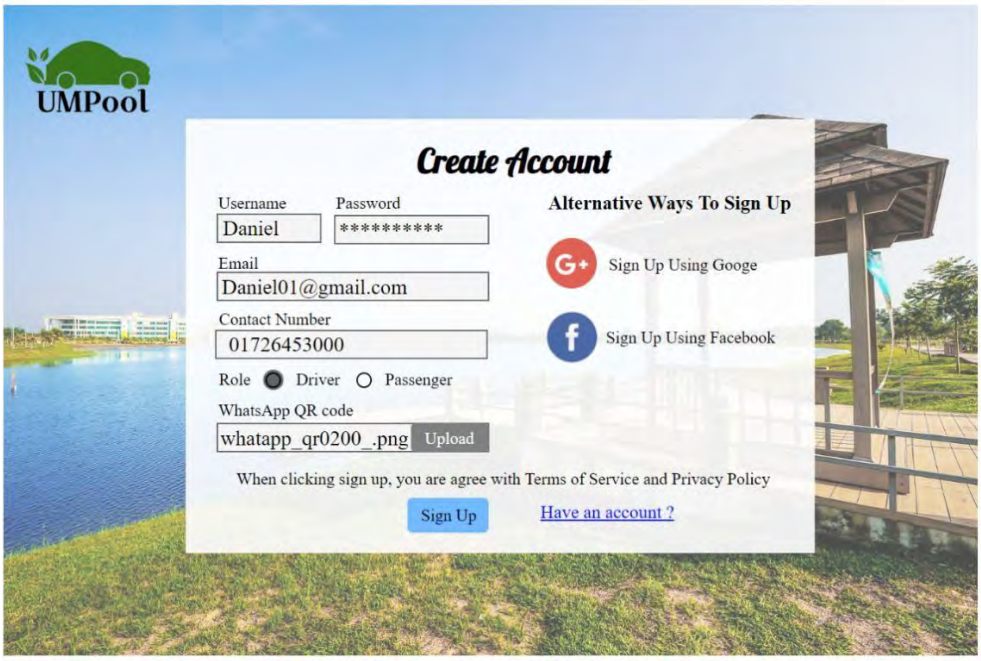


Figure 3.3 Registration page for driver and passenger

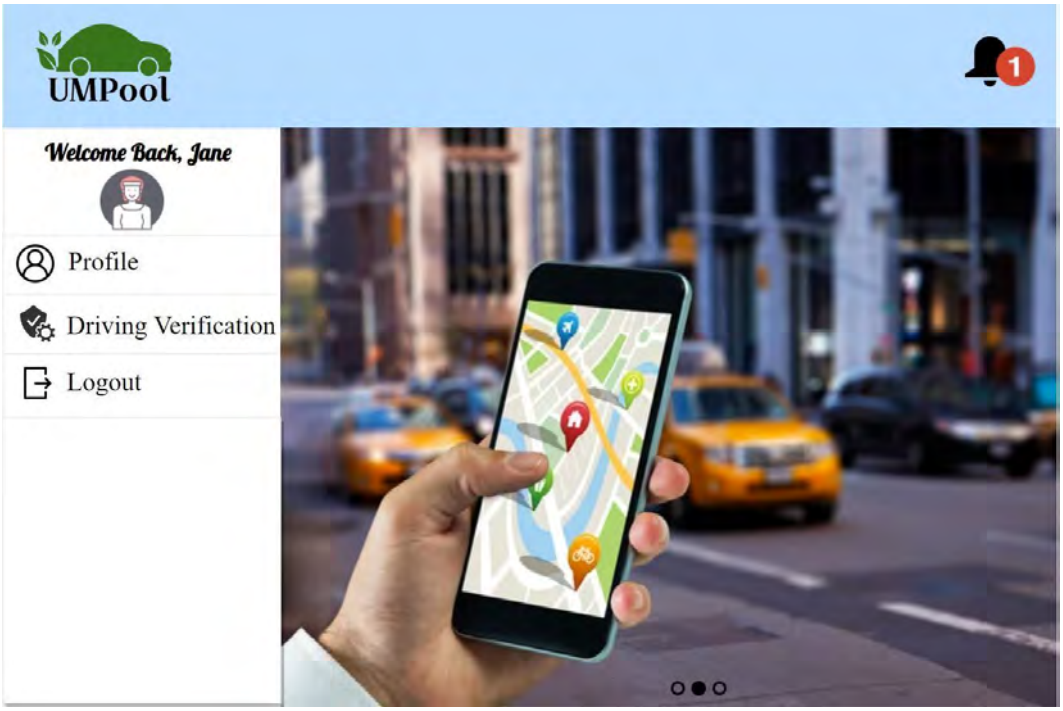


Figure 3.4 Main menu for admin

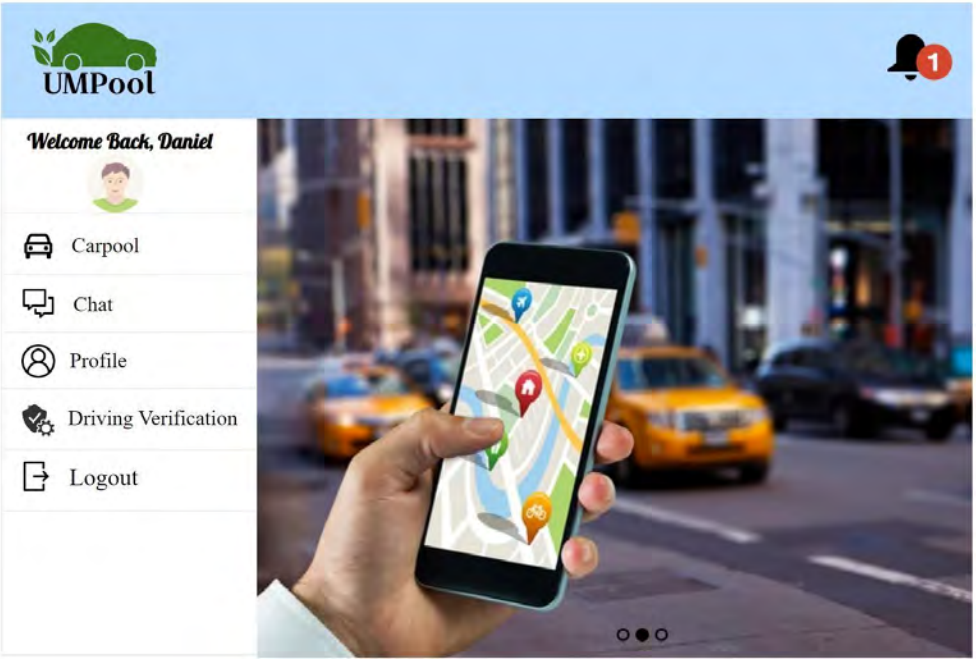


Figure 3.5 Main menu for driver

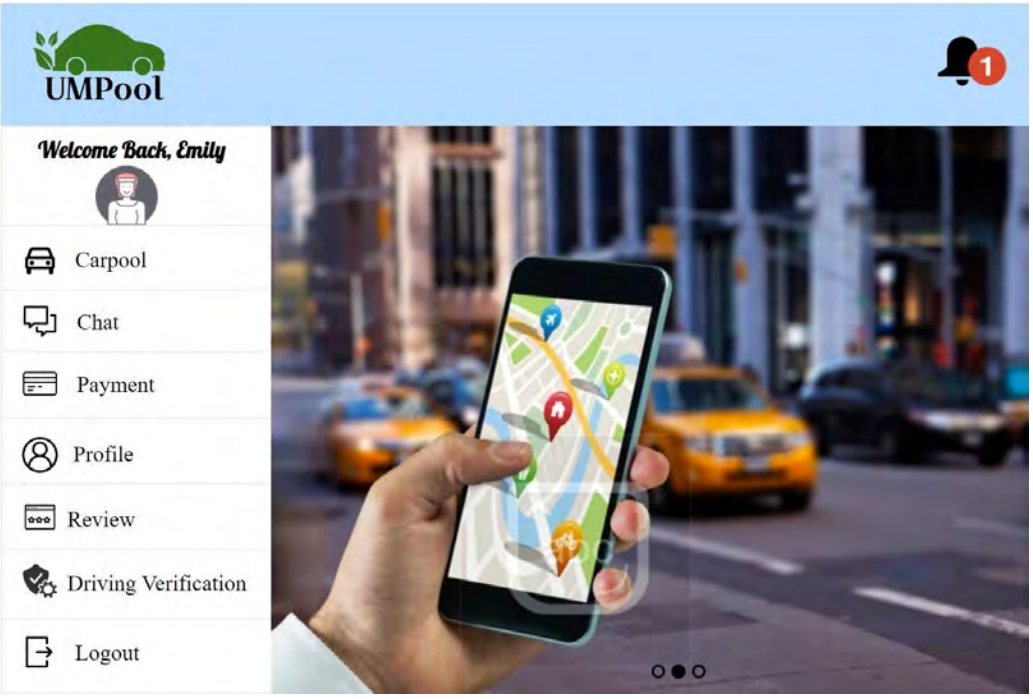


Figure 3.6 Main Menu for passenger

Module 2: Manage Profile

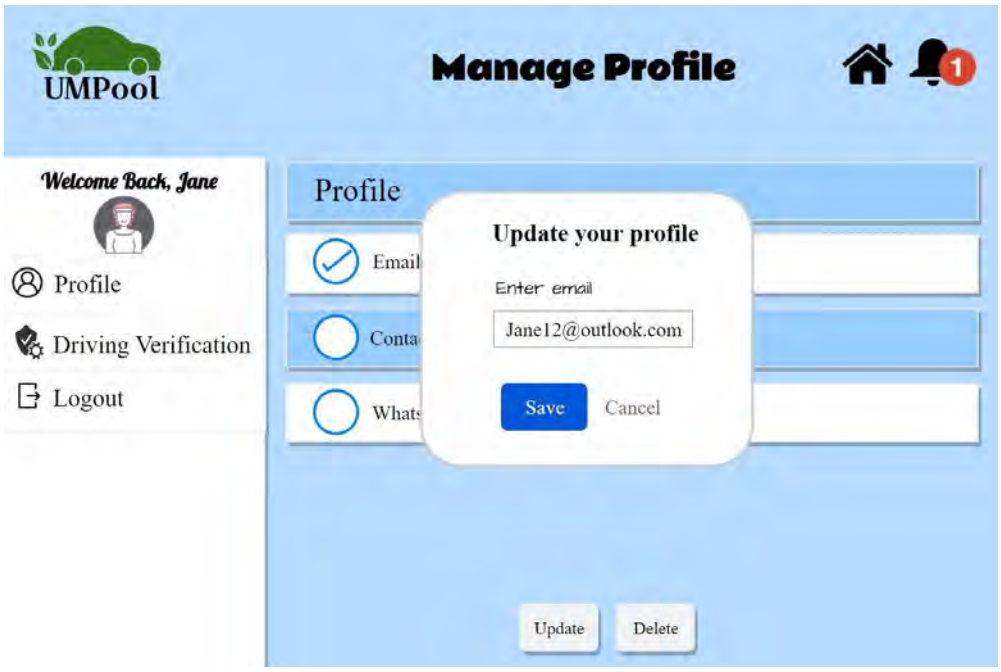


Figure 3.7 Update profile page

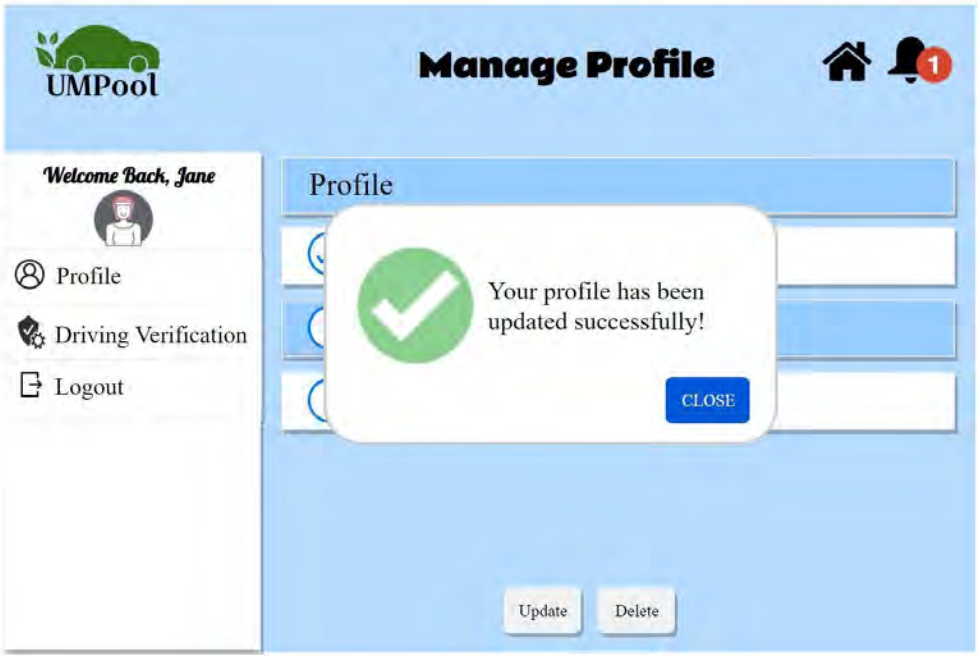


Figure 3.8 Update profile successful message

Figure 3.9 Latest profile page

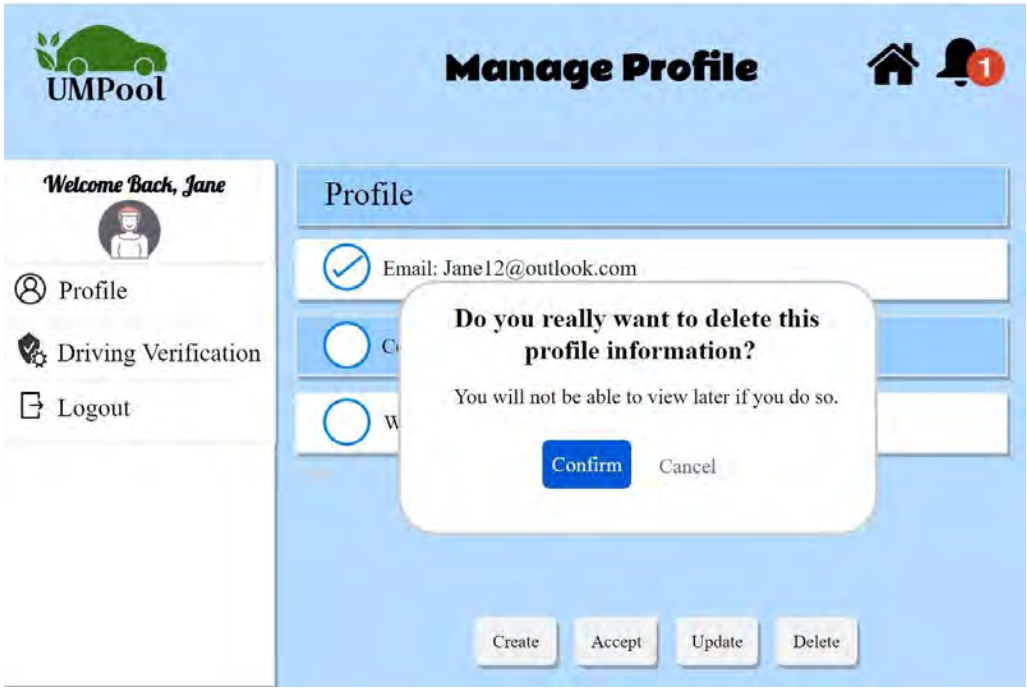


Figure 3.10 Delete confirmation profile message

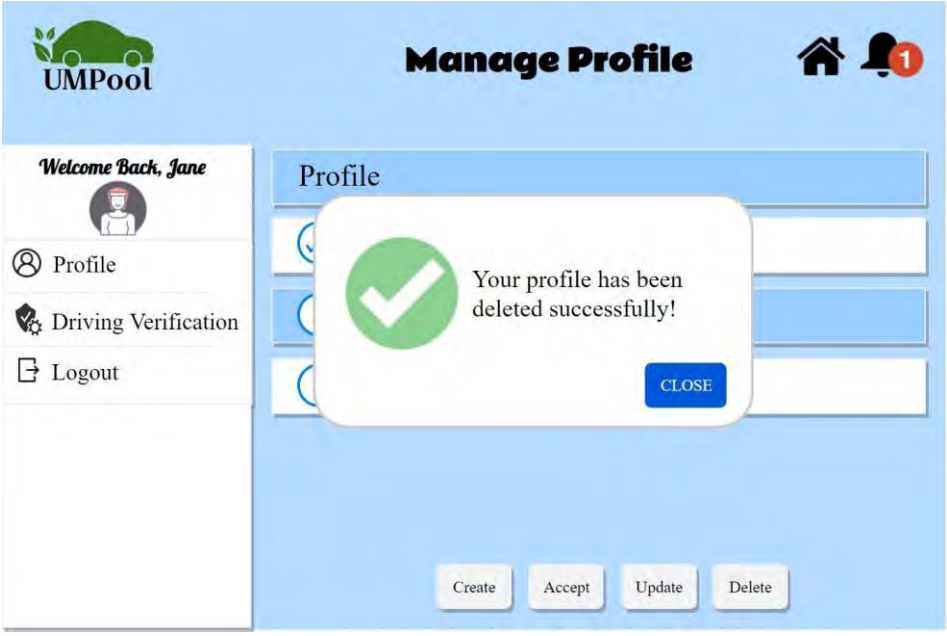


Figure 3.11 Delete profile details message

Figure 3.12 Latest profile page

Module 3: Manage Carpool

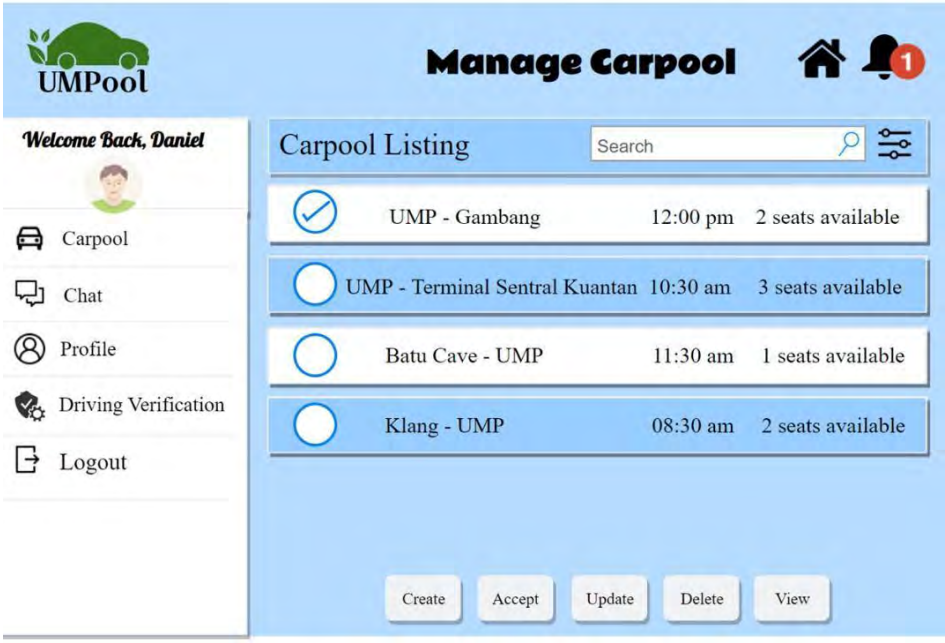


Figure 3.13 Manage Carpool Page

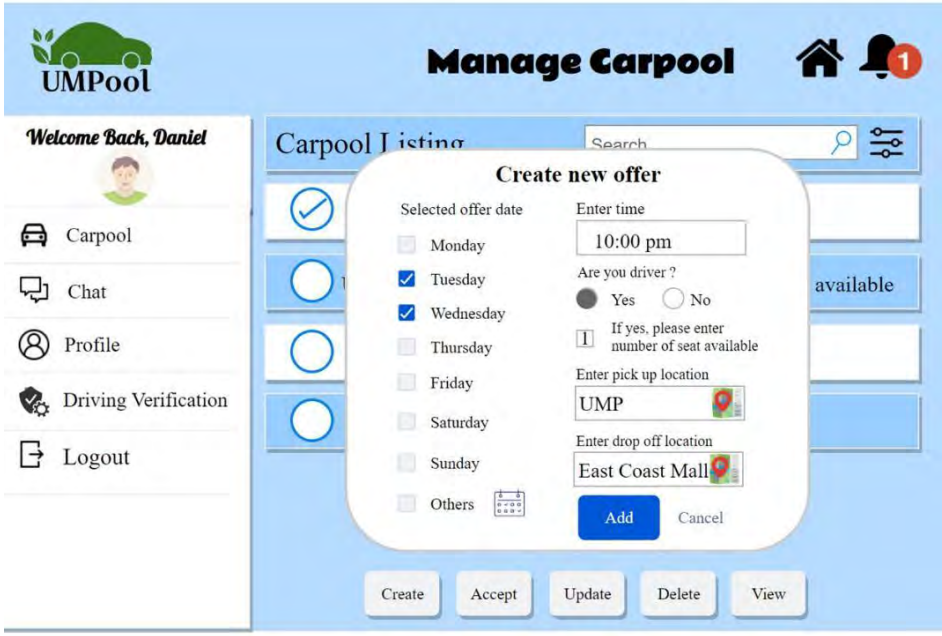


Figure 3.14 Create Carpool Offer Page

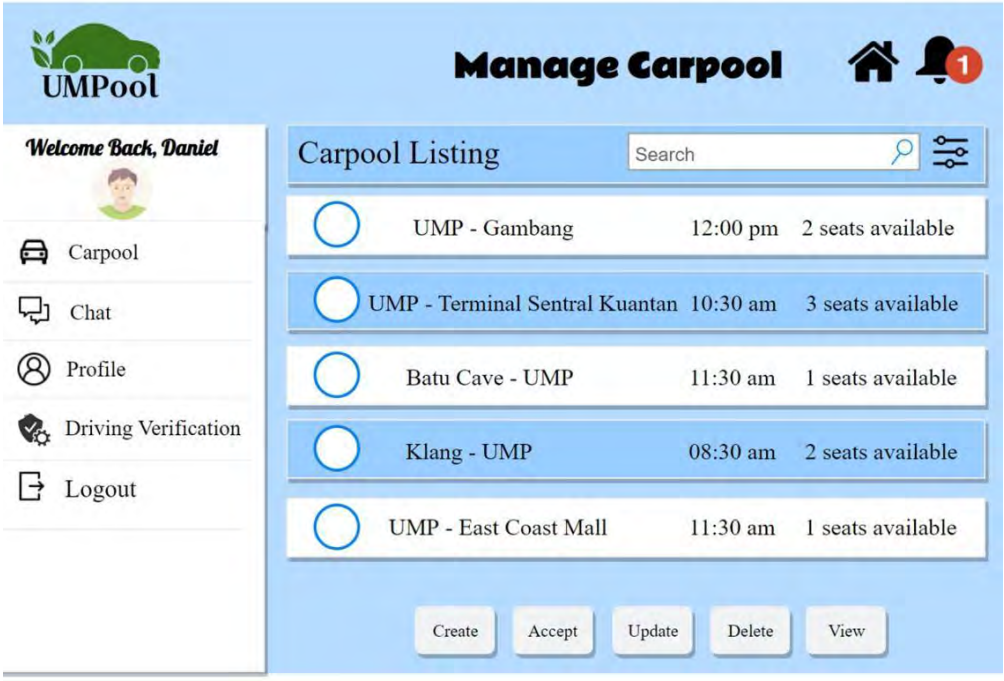


Figure 3.15 Latest Carpool Listing

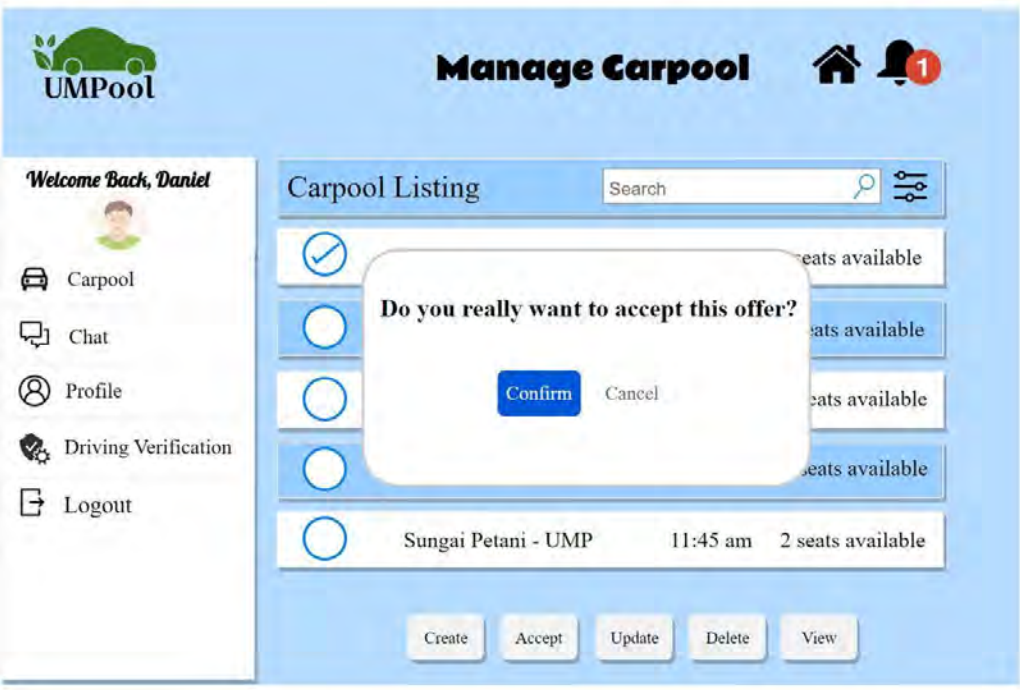


Figure 3.16 Accept Offer Confirmation

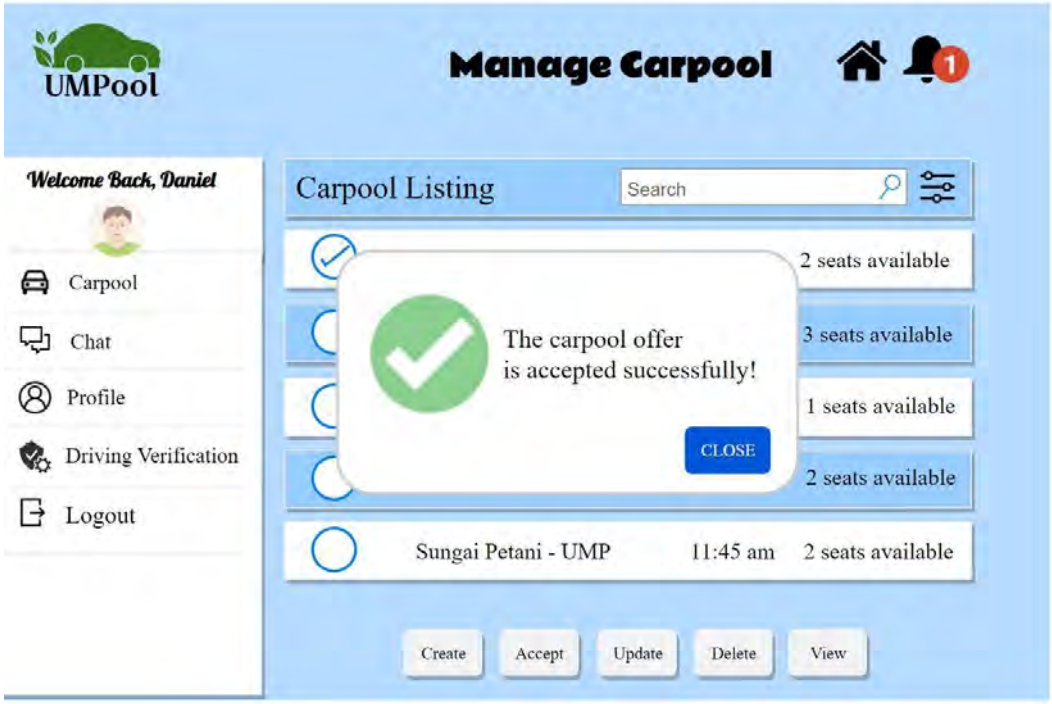


Figure 3.17 Accept Carpool Offer Successful Message

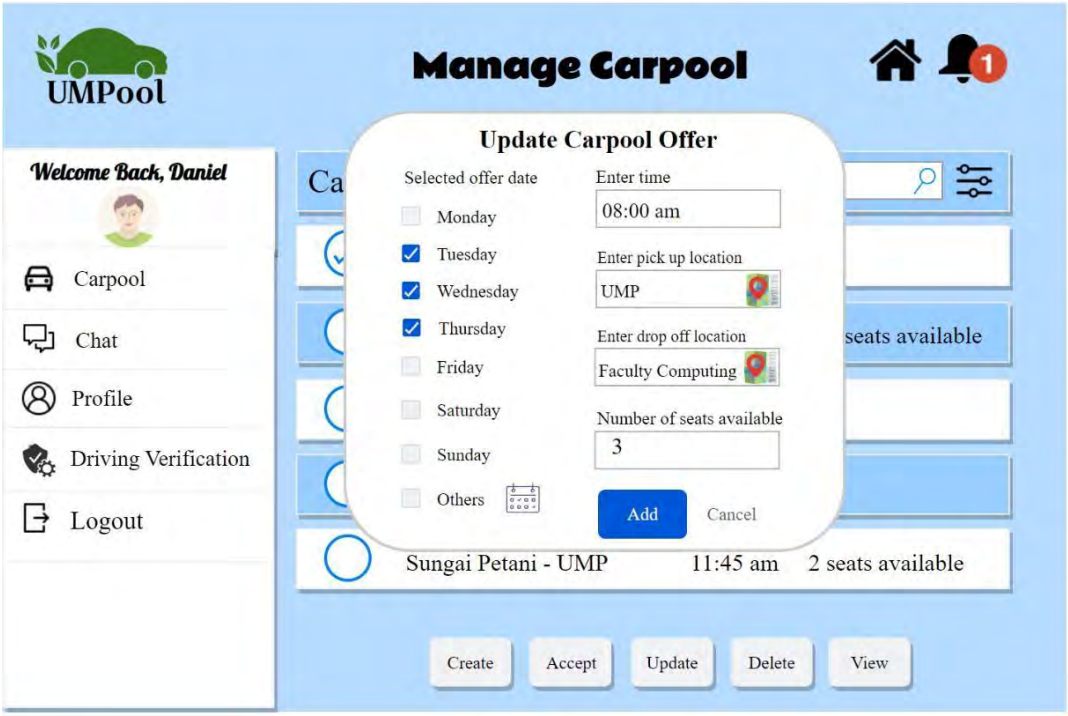


Figure 3.18 Update Offer Page

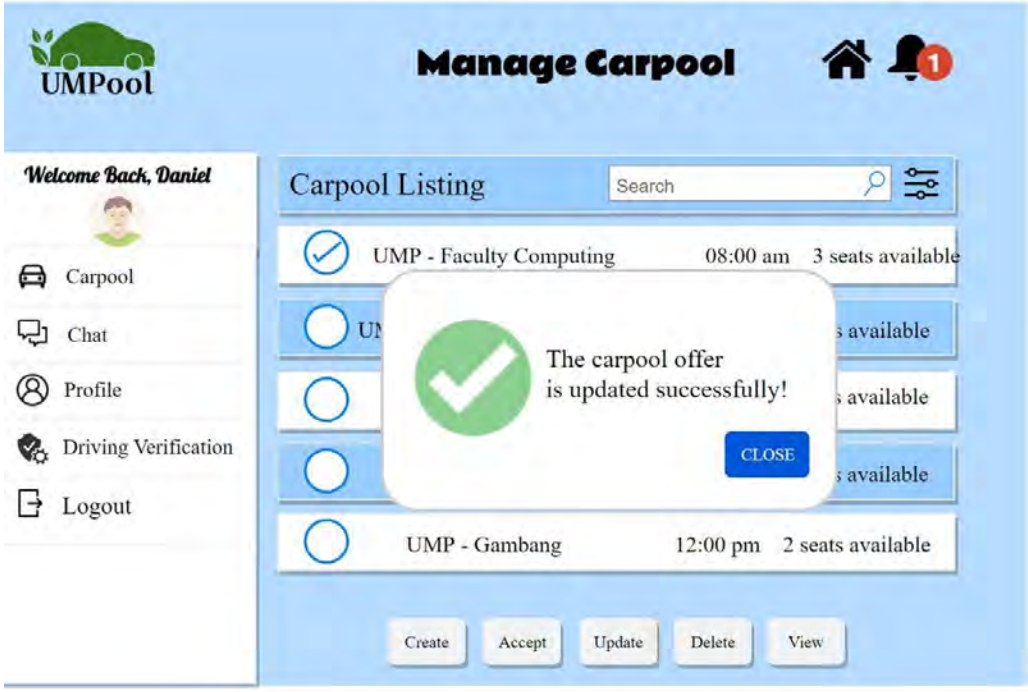


Figure 3.19 Update Offer Message

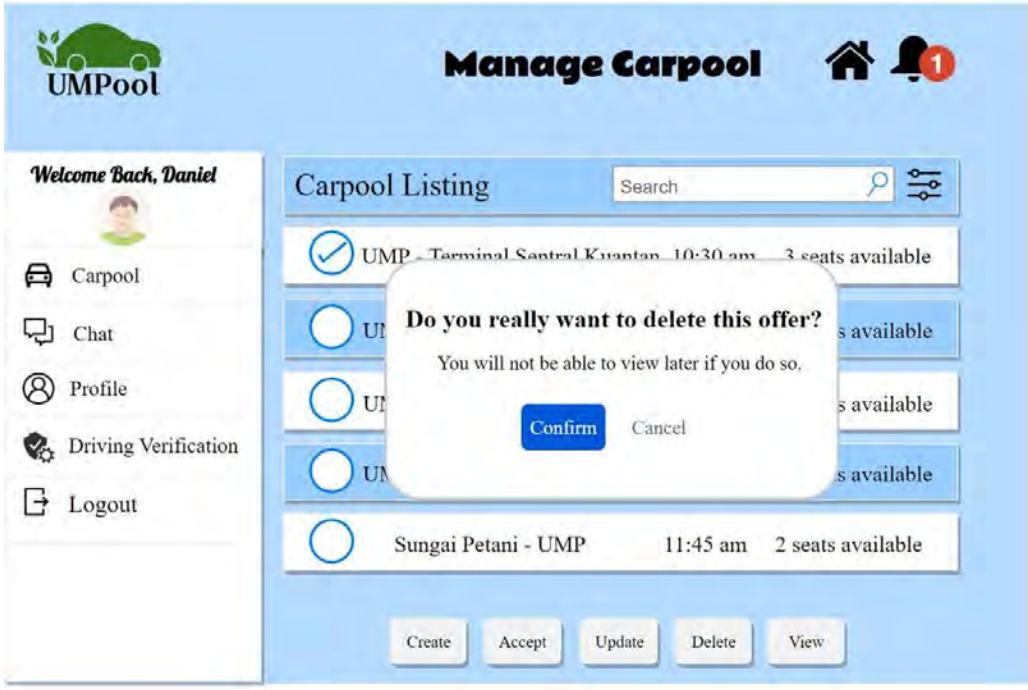


Figure 3.20 Delete Offer Confirmation

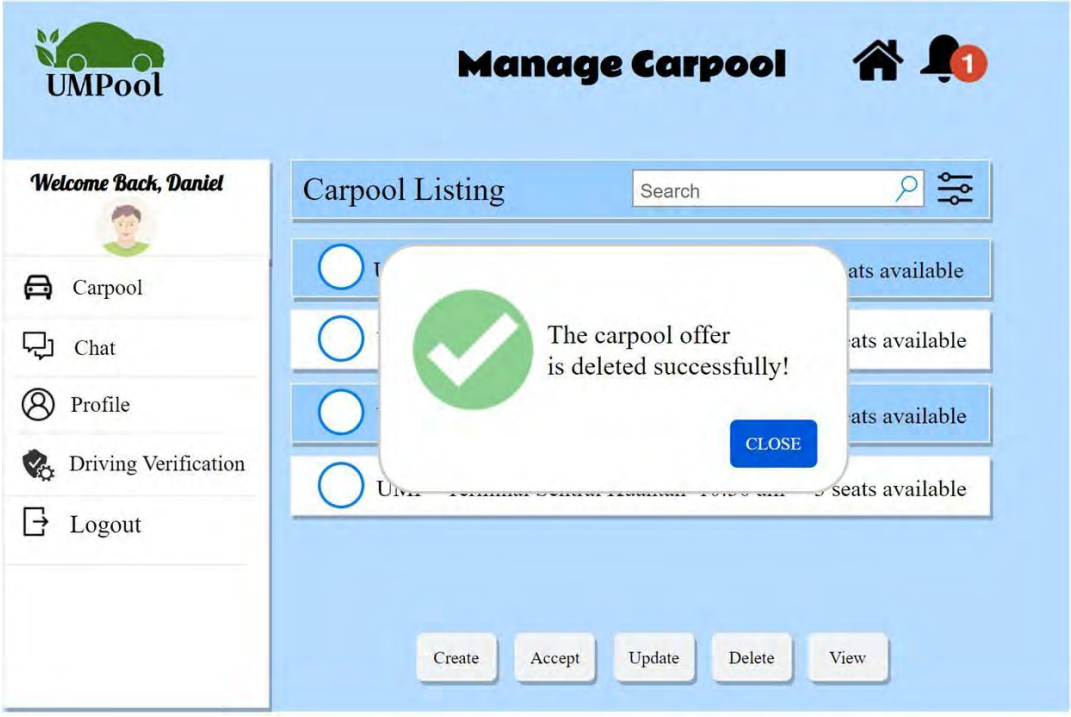


Figure 3.21 Delete Carpool Offer Message

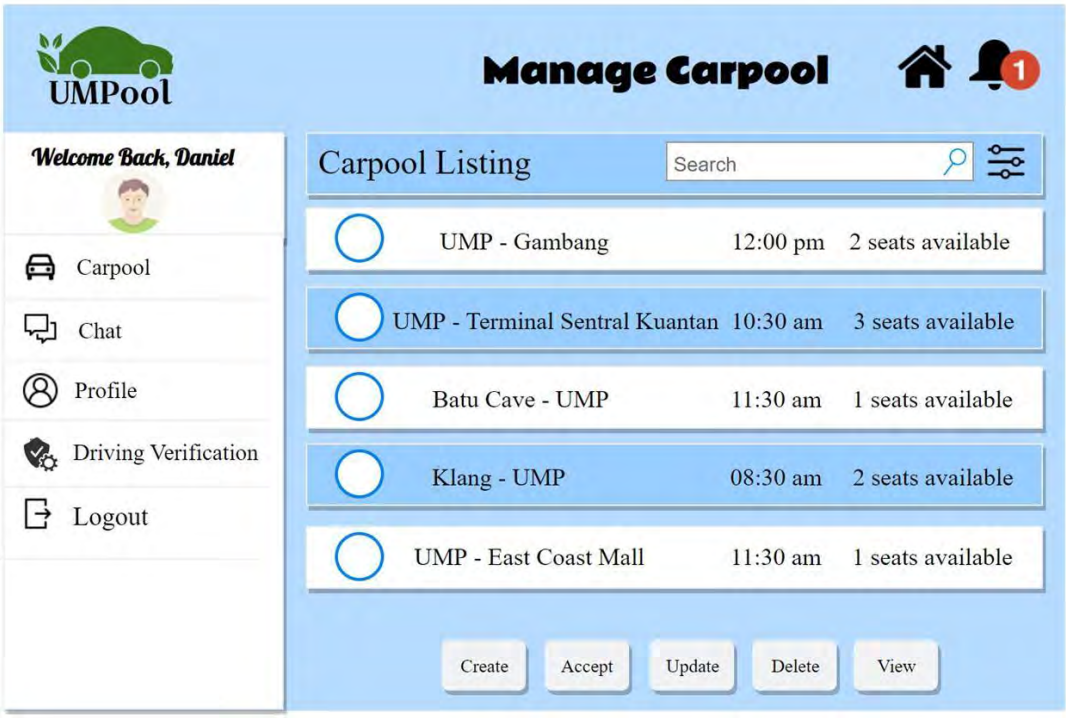


Figure 3.22 Latest Carpool Offer

Figure 4.23 View Carpool Offer

Module 4: Manage Payment

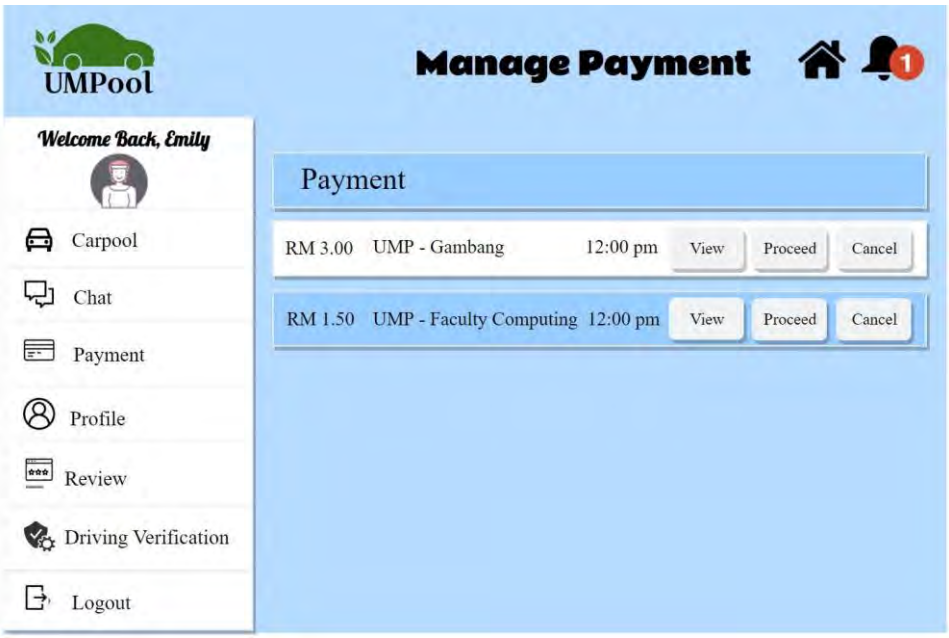


Figure 3.24 Payment Listing Interface

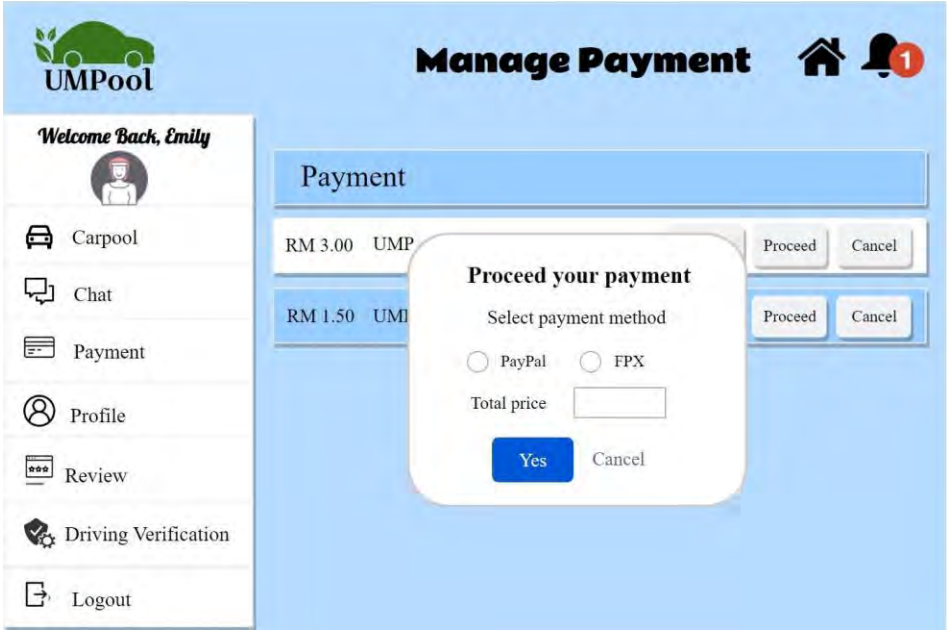


Figure 3.25 Accept Payment Interface

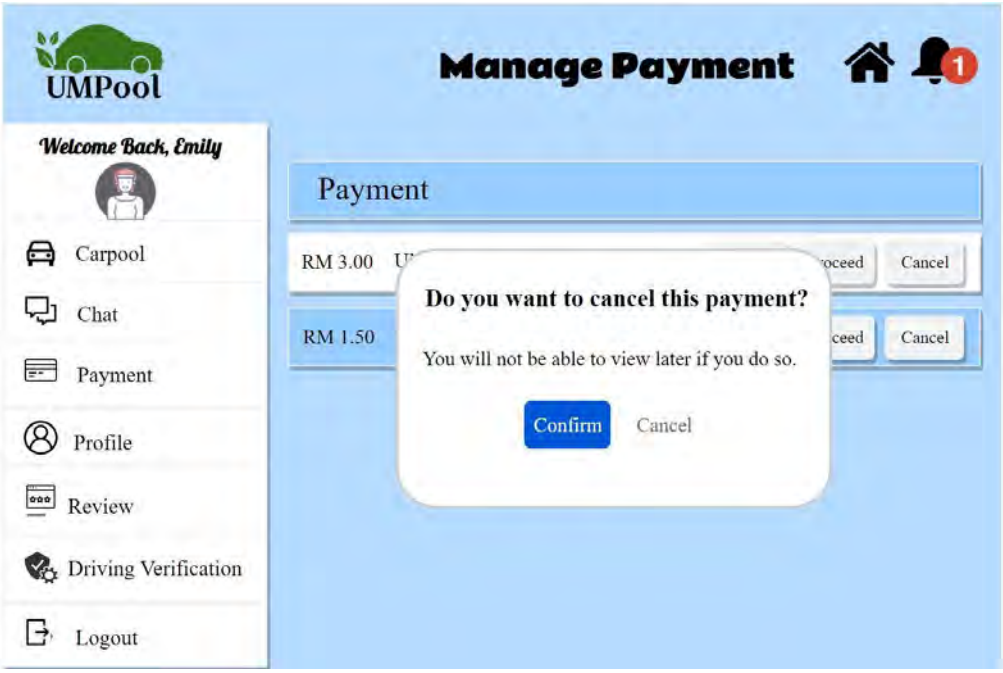


Figure 3.26 Cancel Payment Interface

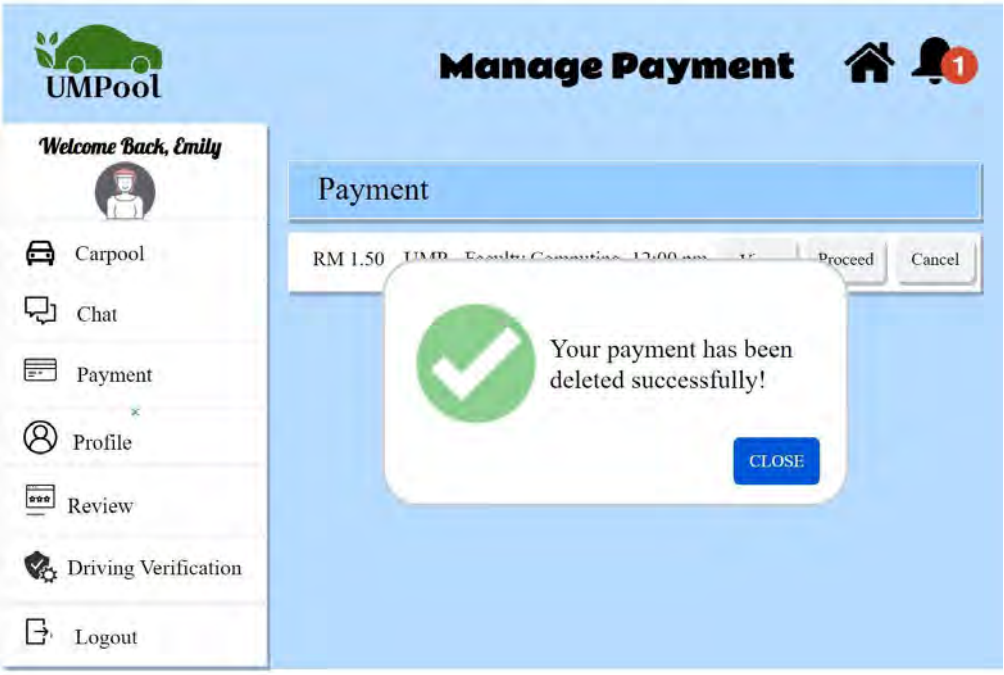


Figure 3.27 Delete Payment Message

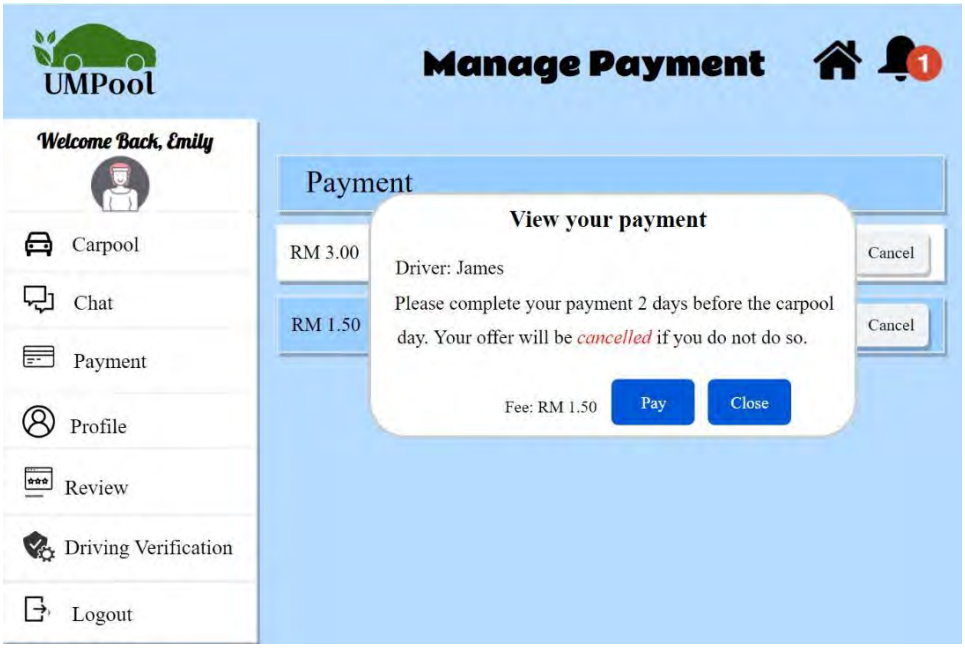


Figure 3.28 View Payment Page

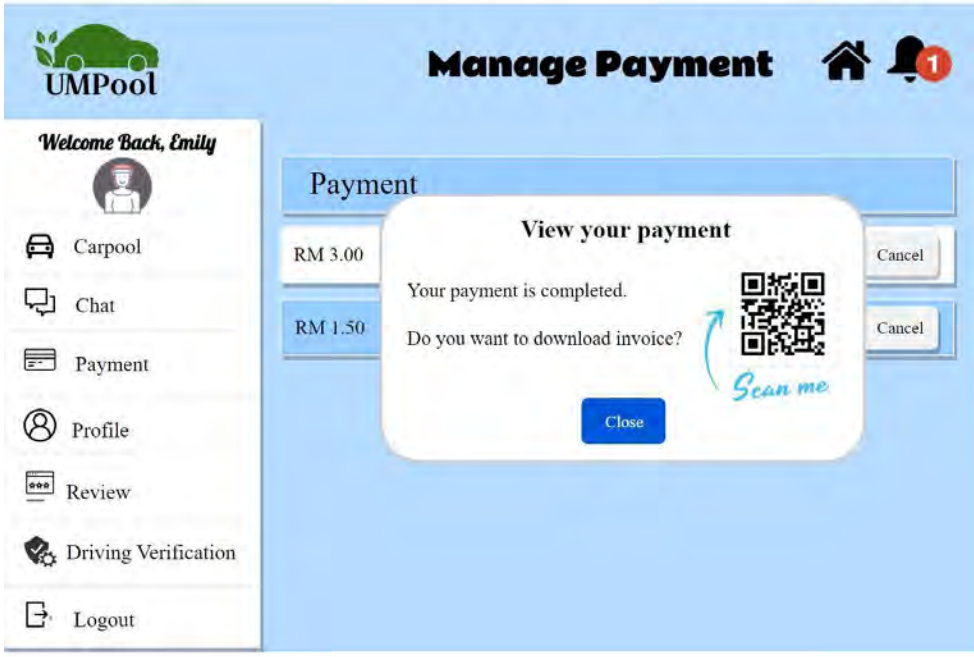


Figure 3.29 View Completed Payment

Module 5: Manage Review

Figure 3.30 Manage Review Page

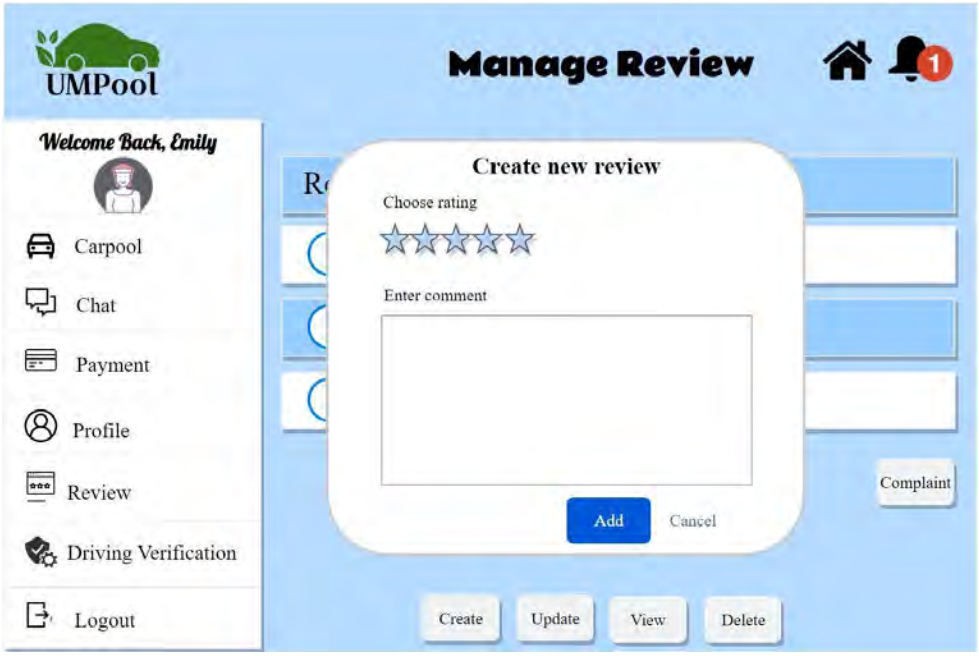


Figure 3.31 Create review page

Figure 3.32 Update review Page

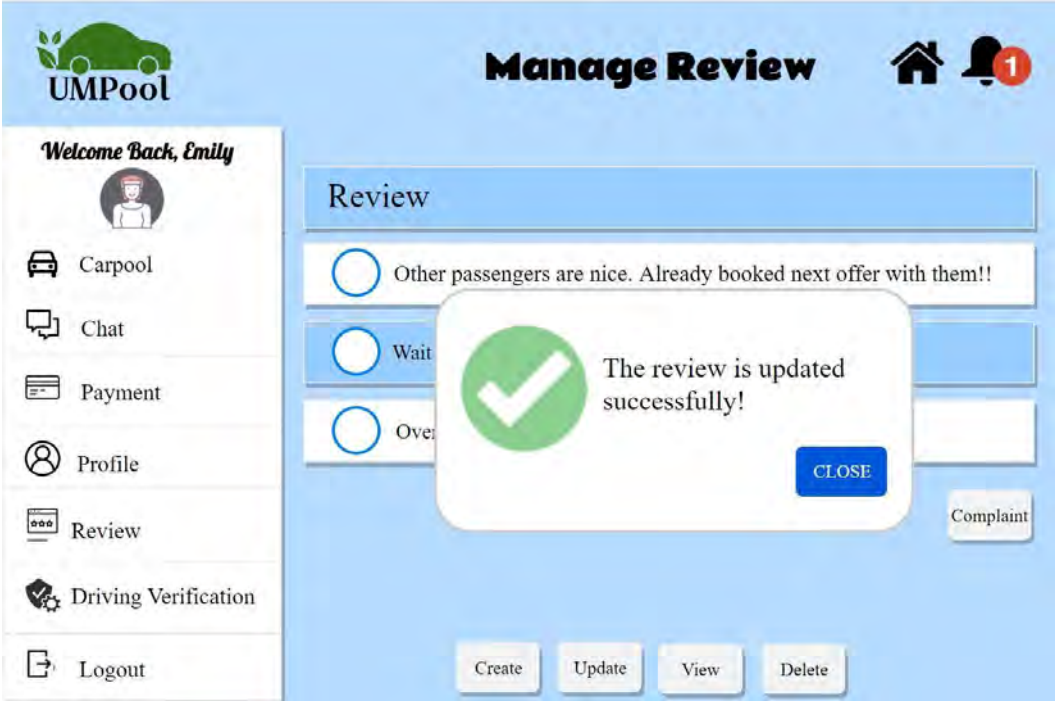


Figure 3.33 Update Successful Message

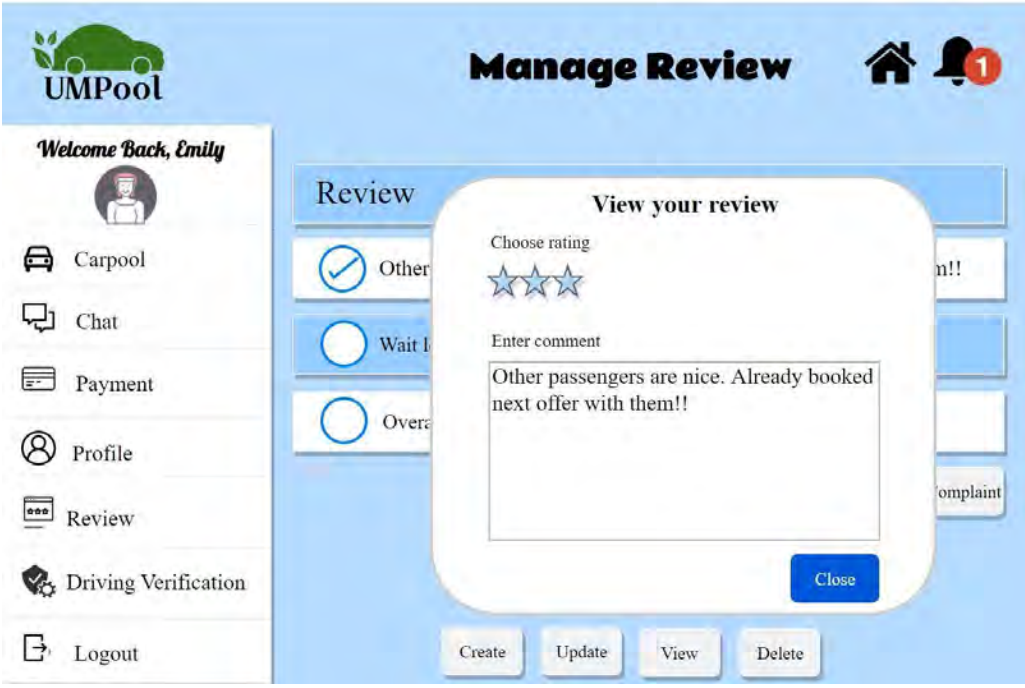


Figure 3.34 View Review Page

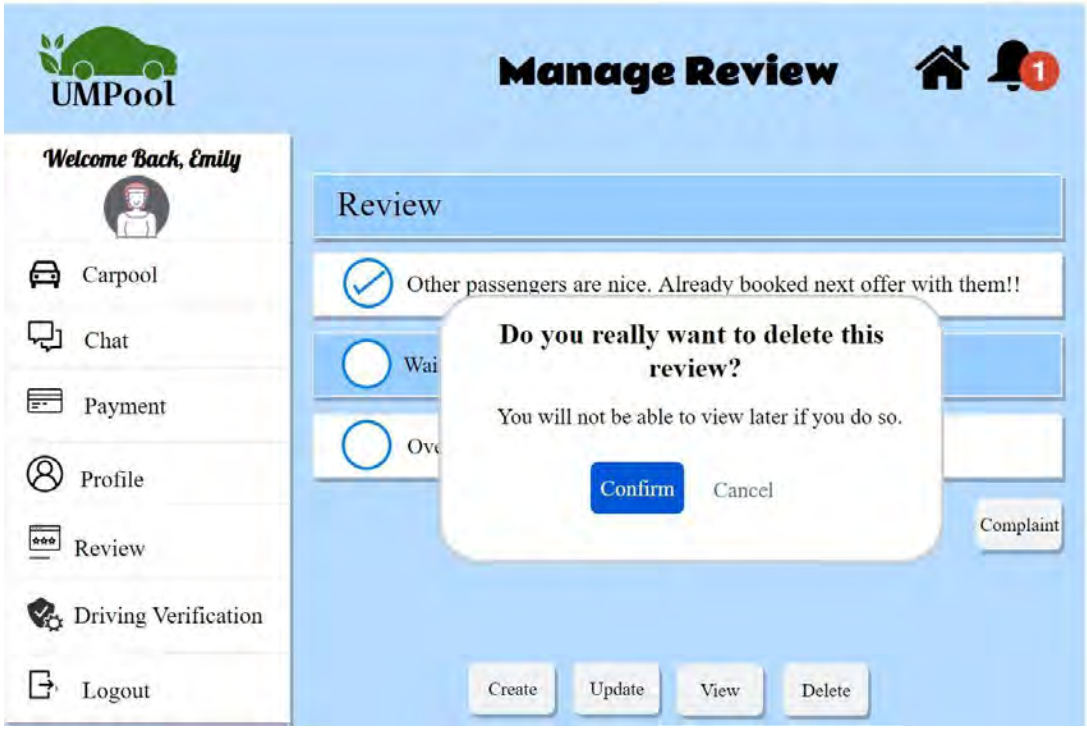


Figure 3.35 Delete Review Page

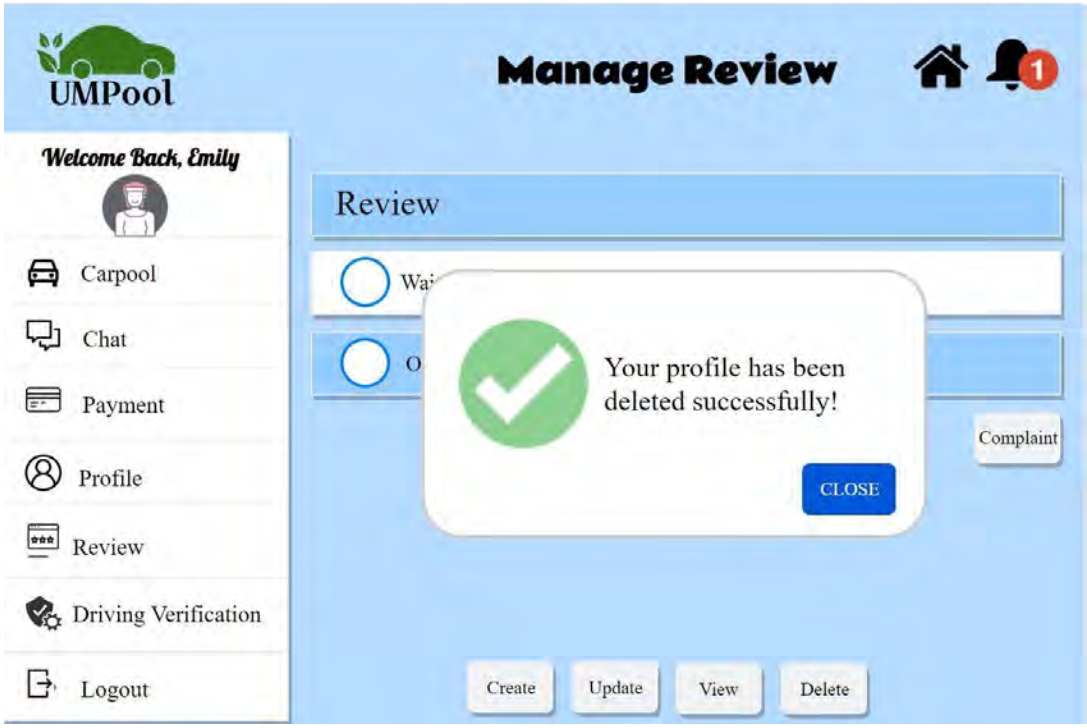


Figure 3.36 Delete Review Success Message

Figure 3.37 Latest review

Module 6: Manage Driving Verification

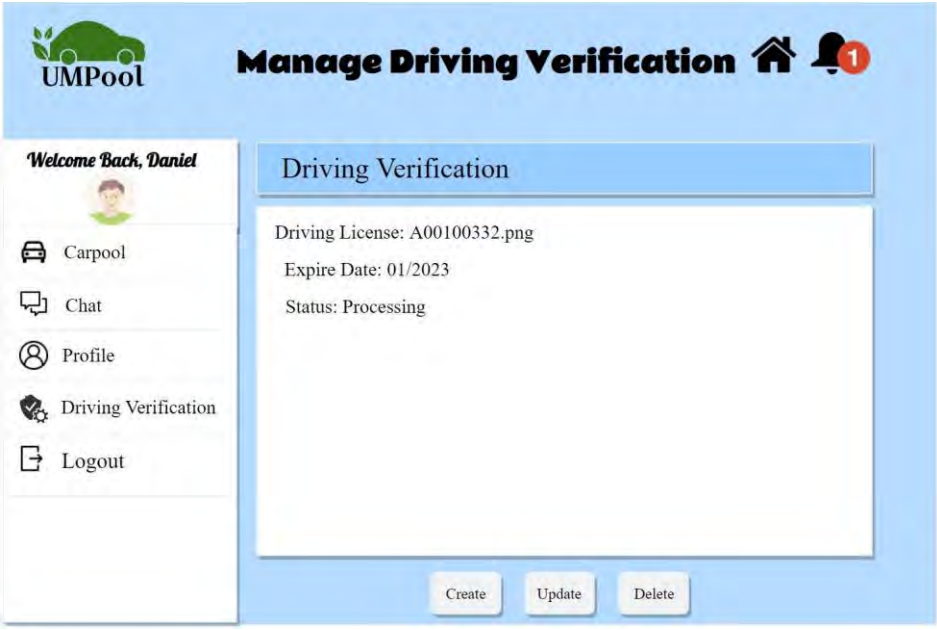


Figure 3.38 View Driving Verification

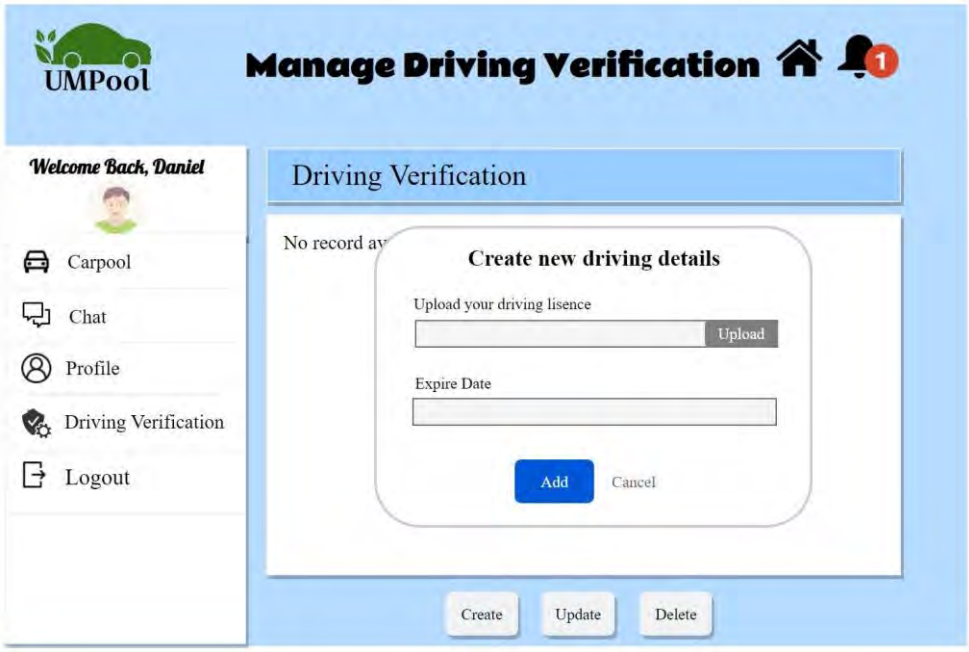


Figure 3.39 Create new driving details

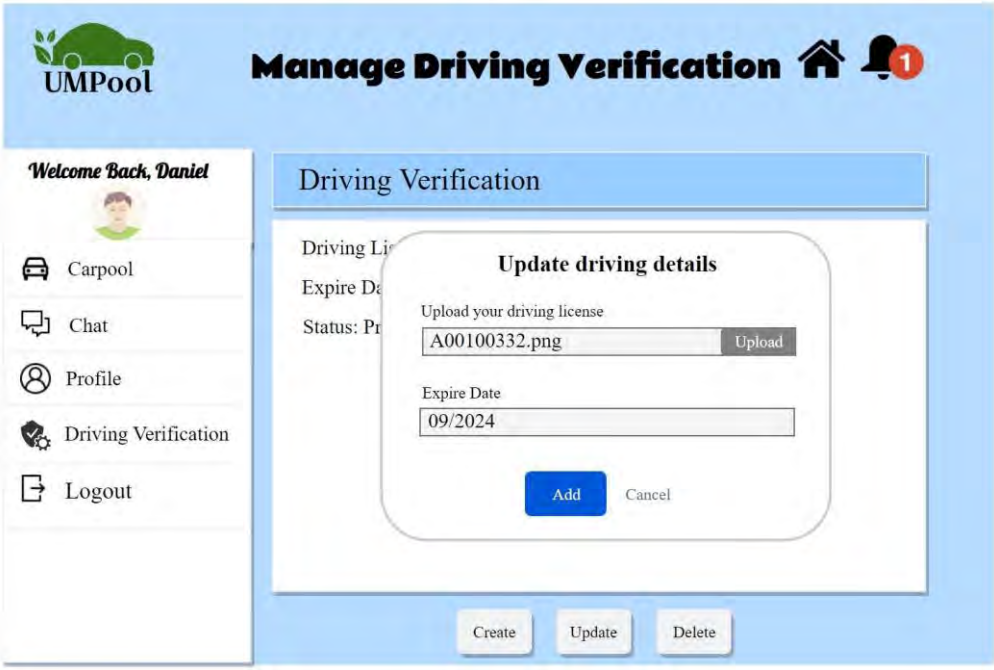


Figure 3.40 Update driving details

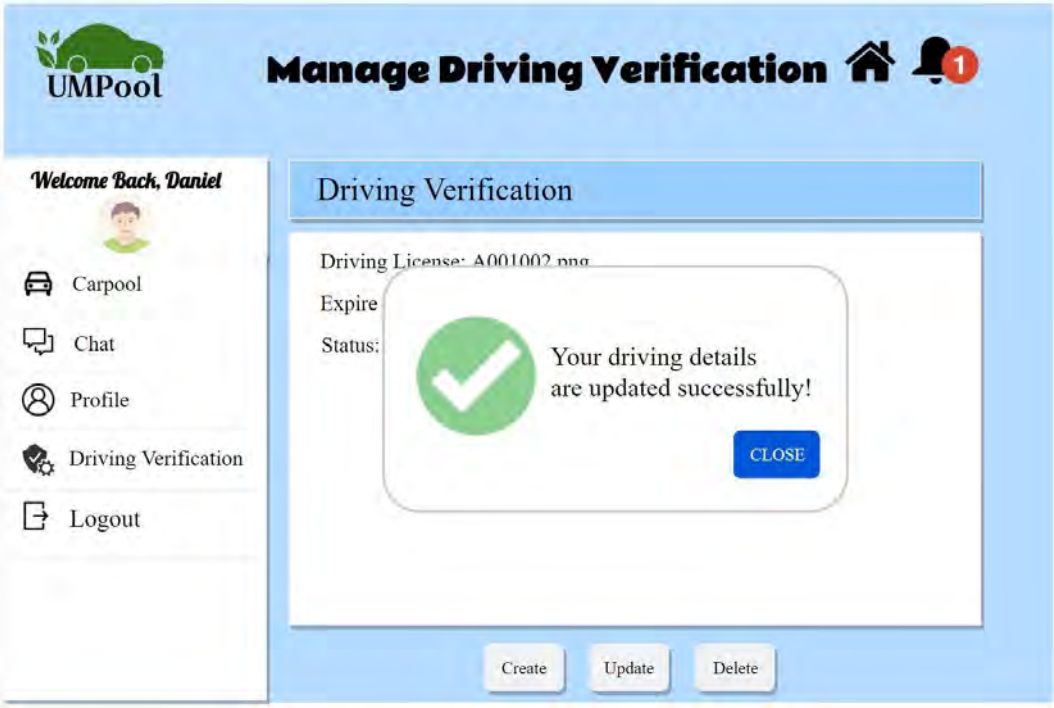


Figure 3.41 Update driving verification success message

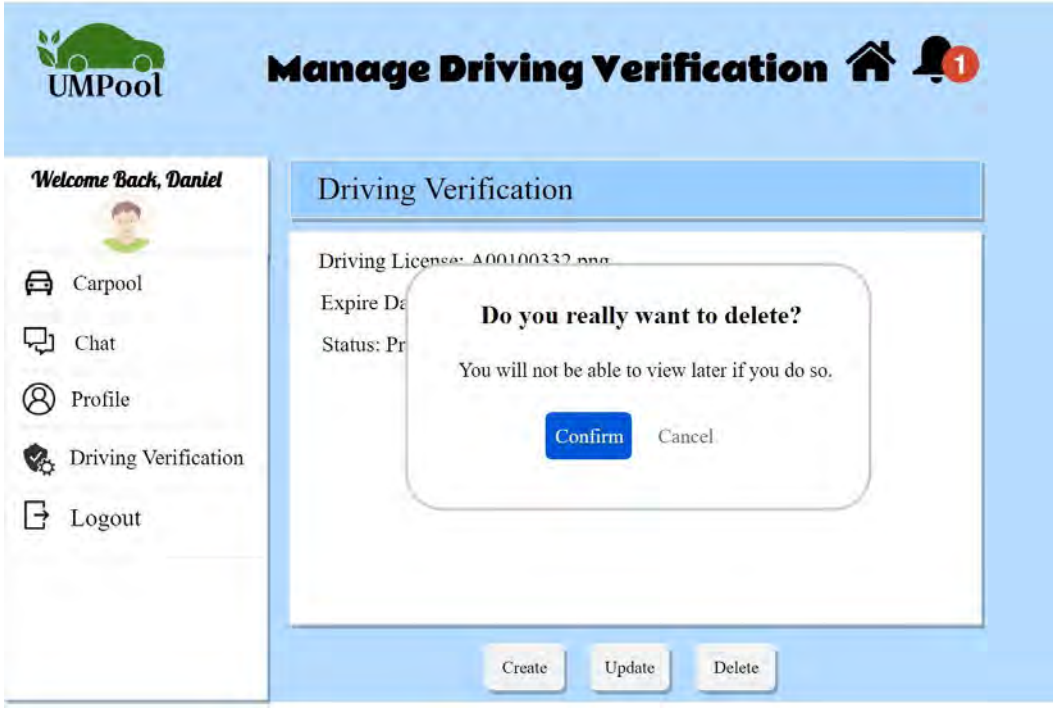


Figure 3.42 Confirmation Delete Driving Details

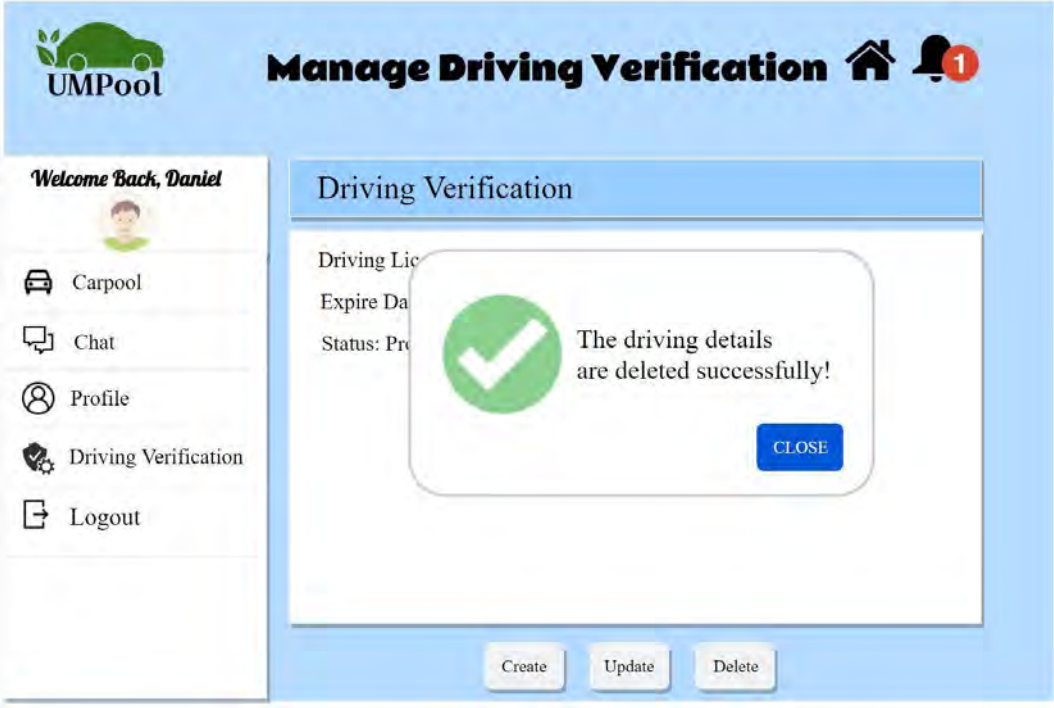


Figure 3.43 Delete driving details success message

Figure 3.44 Latest driving details

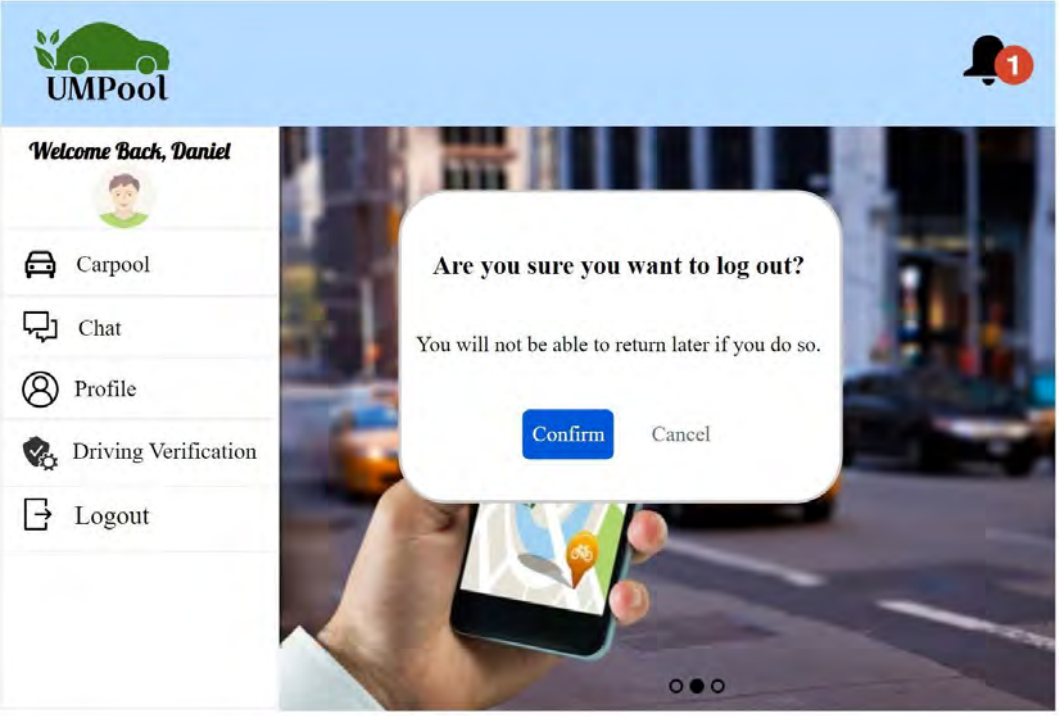


Figure 3.45 Log Out

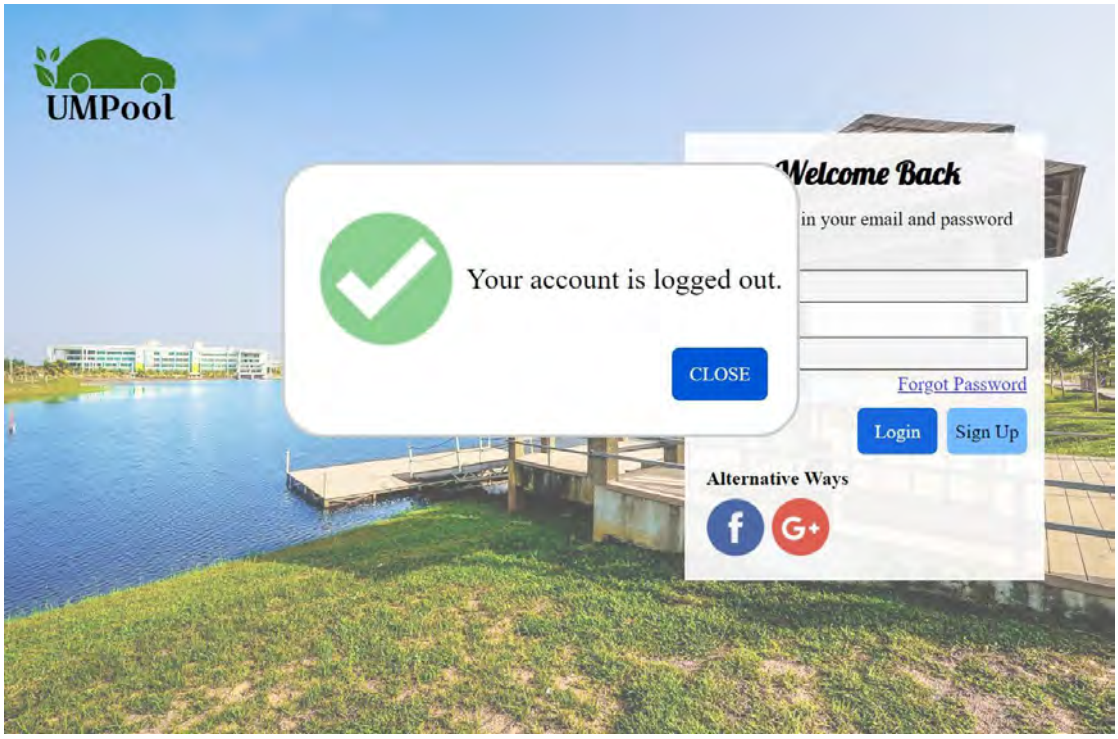


Figure 3.46 Log Out Success Message

3.2 HARDWARE AND SOFTWARE SPECIFICATION

Table 3.3 Hardware Specification

Hardware	Specification	Purpose
Lenovo Xiaoxin Pro 14	Retina IPS LCD, 625 nits (typ)Windows 11 AMD Ryzen 7-5800H 8-Cores, 16-Threads (3.20 GHz, up to 4.40 GHz with Max Boost, 4MB Cache L2/16 MB Cache L3)	It allows the development of systems and documentation.
Smartphone (iPhone 8 plus)	Retina IPS LCD, 625 nits (typ) iOS 11	It allows users to test the functionality of scanning WhatsApp QR contacts.

Table 3.4 Software Specification

Software	Specification	Purpose
Microsoft Word	Type: Microsoft Word 365 for Enterprise	Used in documentation.
Visual Studio Code	Version: 1.67.2	Used in the development of system.
Draw.io	N/A	Used in visualise diagram.
XAMPP	Version: 3.3.0	Used in testing web pages.

APPENDIX D
SOFTWARE DESIGN DOCUMENT

2022

SOFTWARE DESIGN DESCRIPTION (SDD)

[UMPOOL: A CARPOOLING SYSTEM]

DOCUMENT APPROVAL

	Name	Date
Authenticated by: Name: Lee Lye Eng	Lee Lye Eng	20/2/2023
Approved by:	Dr Ahmad Fakhri Bin Ab. Nasir	20/2/2023

_____ Client		

Software : Microsoft Office 2016

Archiving Place : Google Drive

TABLE OF CONTENT

LIST OF FIGURES

1.1 LIST OF TABLES

LIST OF APPENDICES

CHAPTER 1

1.1 PROJECT DESCRIPTION

UMPool: A Carpooling System is a web-based application is a system that facilitates carpooling services for students and staff at Universiti Malaysia Pahang. There are three users in the system which are admin, driver, and passenger. Users are allowed to create and accept other carpool offer when needed. When the offer is accepted by passengers, passengers will proceed to payment to get carpool offer confirmation. After the payment is completed, users (drivers and passengers) can communicate with one another by scanning the WhatsApp QR code provided at the upcoming carpool listing. At the same time, admin will hold the payment until driver and passengers have completed the carpool.

There are six modules in the system which including manage user login, manage profile, manage carpool, manage payment, manage review, and manage driver verification. For manage user login, users are allowed to login by email or multi-factor authentications (MFA) which are Facebook Login and Google Sign-in. Users do not have account requires to register and then proceed to login.

For the manage profile module, all users enable to update their profile information and change their password. After user make changes on their profile, the system will display the latest profile page.

For the manage carpool module, both driver and passenger can create, accept, update, delete carpool offer. The latest carpool offer displays when users made changes to the carpool offer. In addition, this module displays the carpool listing for user to search, and filter based on their preference.

For manage payment, passenger is required to complete the payment to get confirmation of the offer. There are two (2) payment method provide in the system which are Paypal and FPX. Users are not allowed to cancel their offer after payment is completed. Admin will hold the payment and release to driver when the carpool offer status is changed from upcoming to completed.

For manage review, passengers are able to create, edit, delete, and view their reviews once their trip has ended.

For manage driver verification, driver can create, update, delete, and view their driving details including driving license and driving period. Admin will verify the driver verification by approving and rejecting the driver application.

1.2 SYSTEM IDENTIFICATION

Format: XXX-CB19092-XXXXX-XXXX-VX

System ID: SDD-CB19092-UACS-2022-V1

Document: Software Design Document

Developer ID: CB19092

System Name: UMPool: A Carpooling System

System Abbreviation: UACS

Development Year: 2022

Version: 1

1.3 ARCHITECTURE / BLUE PRINT

This section will provide details about the system architecture, which includes the application layer, the business service layer, and the middleware layer.

1.1.1 3-Tier Architecture Layer

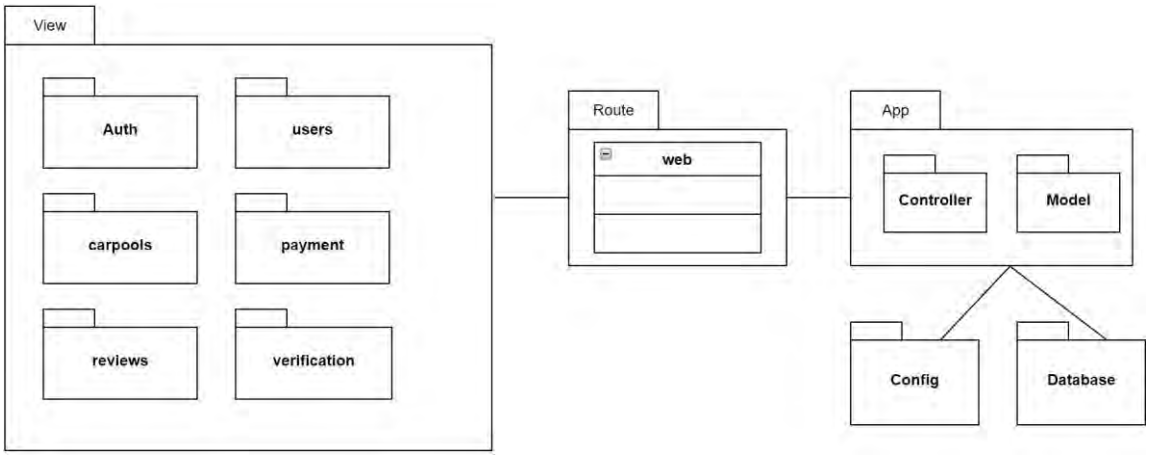


Figure 1 3-Tier Architecture Layer of UACS

1.1.2 Middleware Layer

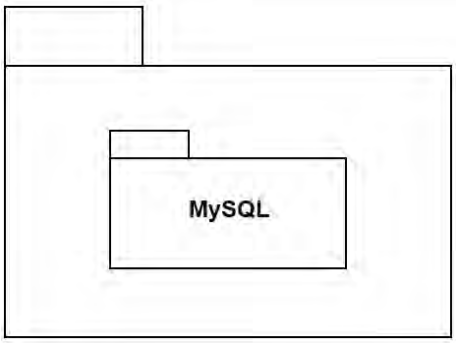


Figure 2 Middleware Layer of UACS

Package Name	Description
MySQL	An open source database is used to provide high flexibility on data structures.

1.4 ARCHITECTURE / BLUEPRINT DESCRIPTION

1.4.1 Package Module

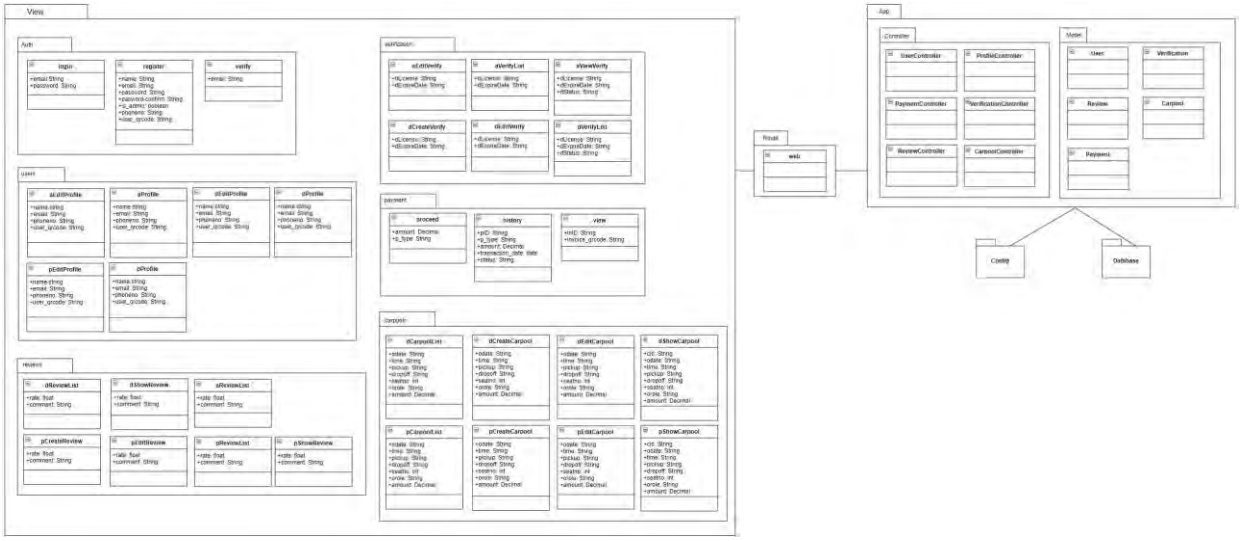


Figure 3 Package Module of UACS

1.4.2 Application Layer

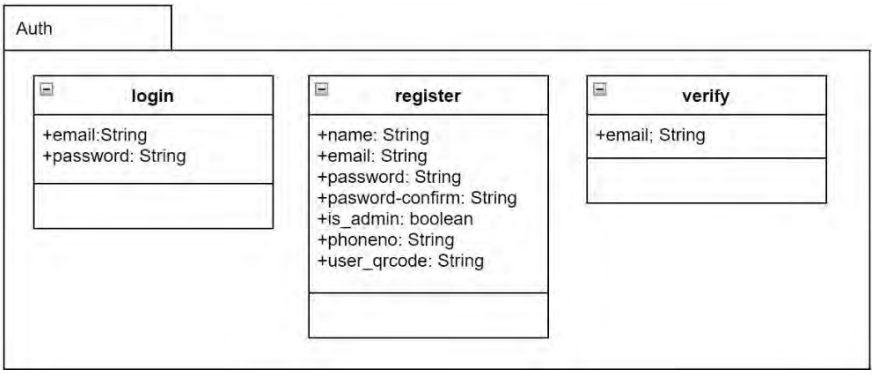


Figure 4 Auth

Interface	Description
login	Interface that allows users to log in with their credentials.
register	Interface that allows users (driver and passenger) to create an account and their details will be stored at user database.
verify	Interface that allows users to enter their email address and a reset password link will be sent to their email.

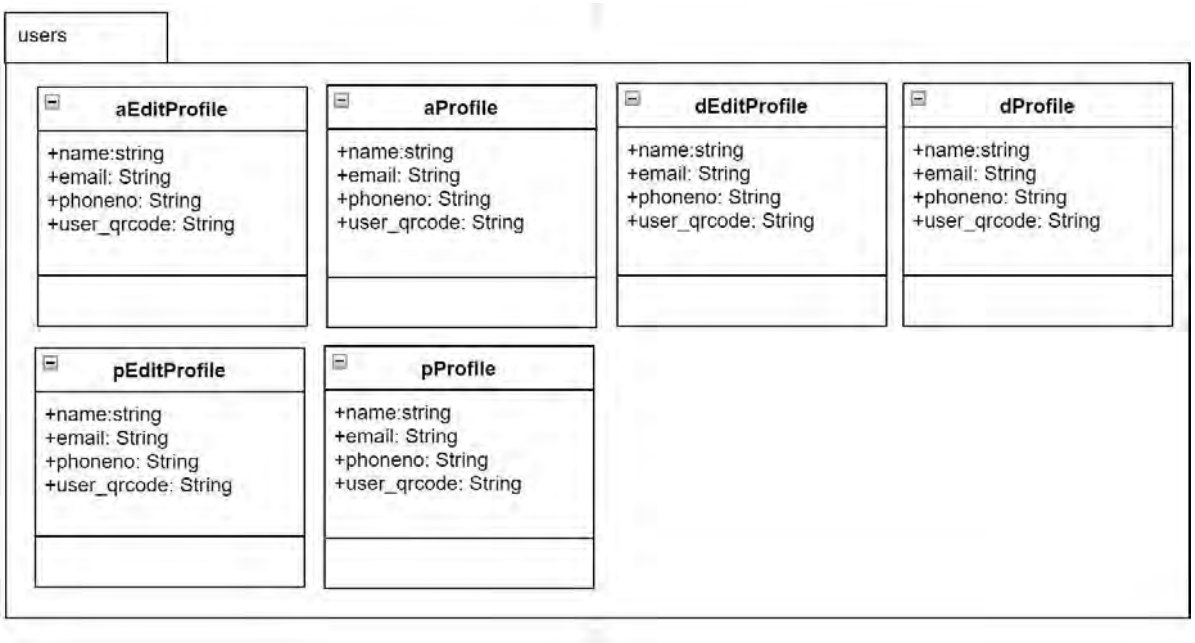


Figure 5 users

Interface	Description
aEditProfile	Interface that allows admin to update their profile by selecting fields and clicking submit.
aProfile	Interface that allows admin to view their profiles.
dEditProfile	Interface that allows driver to update their profile by selecting fields and clicking submit.
dProfile	Interface that allows driver to view their profiles.
pEditProfile	Interface that allows passenger to update their profile by selecting fields and clicking submit.
pProfile	Interface that allows passenger to view their profiles.

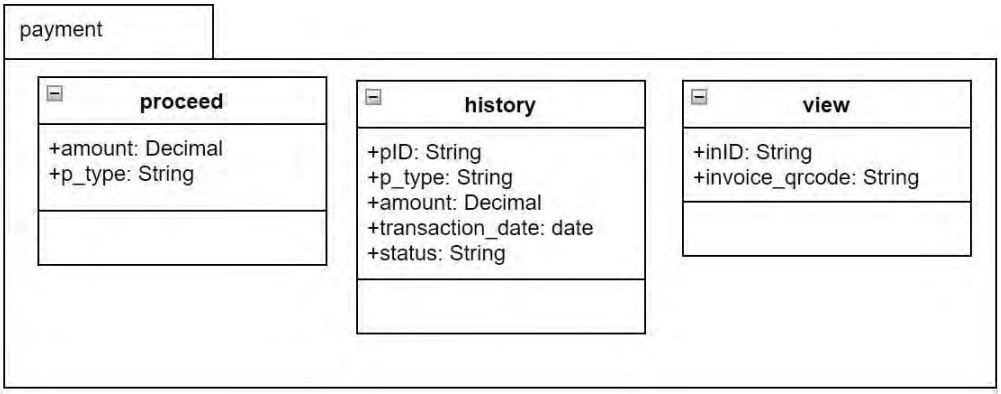


Figure 6 payment

Interface	Description
proceed	Interface that allows them to choose their payment method and check their carpool fee total.
history	Interface that allows users to view their transaction history.
view	Interface that allows users to generate invoice by scanning the qr code provided.

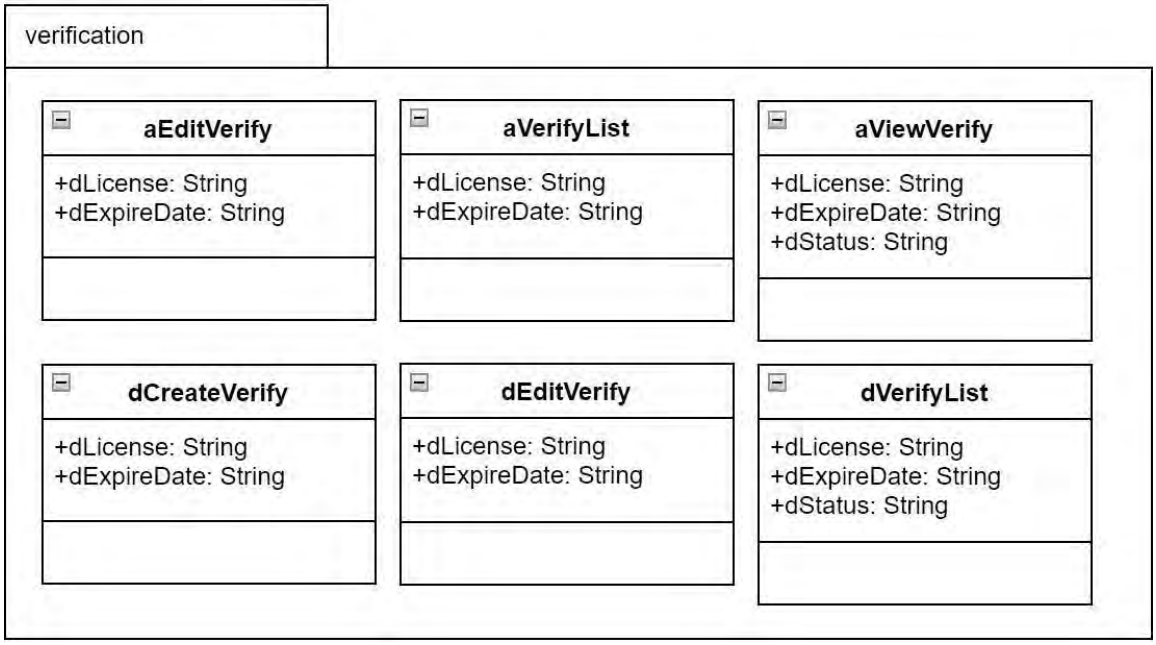


Figure 7 verification

Interface	Description
aEditVerify	Interface that allows admins to approve or reject drivers' driving applications.
aVerifyList	Interface that displays all the drivers' driving applications.
aViewVerify	Interface that allows admin to view the details of drivers' verification.
dCreateVerify	Interface that allows drivers to enter their driving license information and expiration date for verification purposes.

dEditVerify	Interface that allows drivers to update their application by entering latest driving details.
dVerifyList	Interface that displays list of application created by driver.

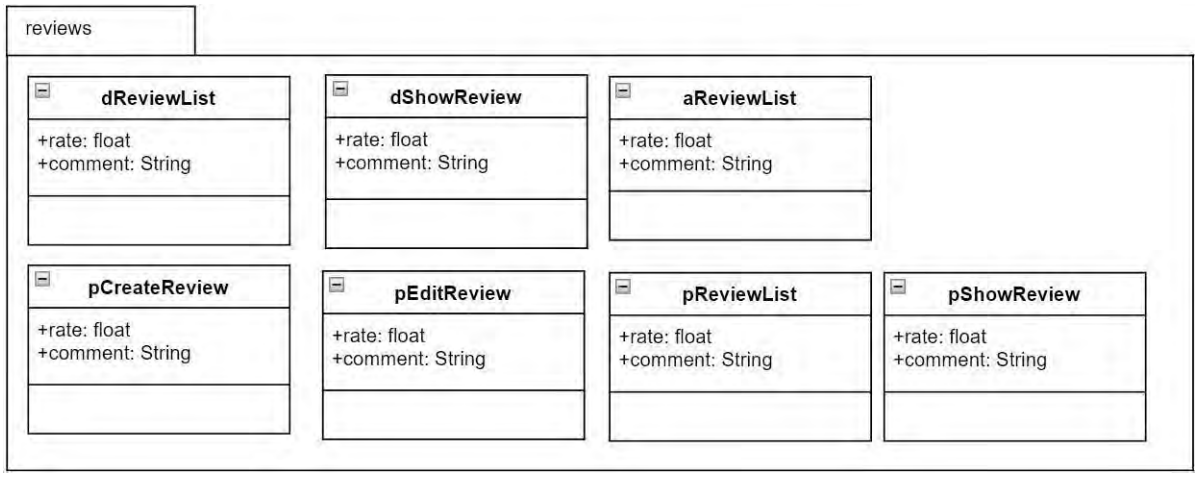


Figure 8 reviews

Interface	Description
dReviewList	Interface that allows driver to view the review listing.
dShowReview	Interface that allows driver to view the review details.
aReviewList	Interface that allows admin to view the review listing.

pCreateReview	Interface that allows passenger to give review when the carpool is completed.
pEditReview	Interface that allows passenger to update their review.
pReviewList	Interface that allows passenger to view the review listing.
pShowReview	Interface that allows passengers to view the review details.

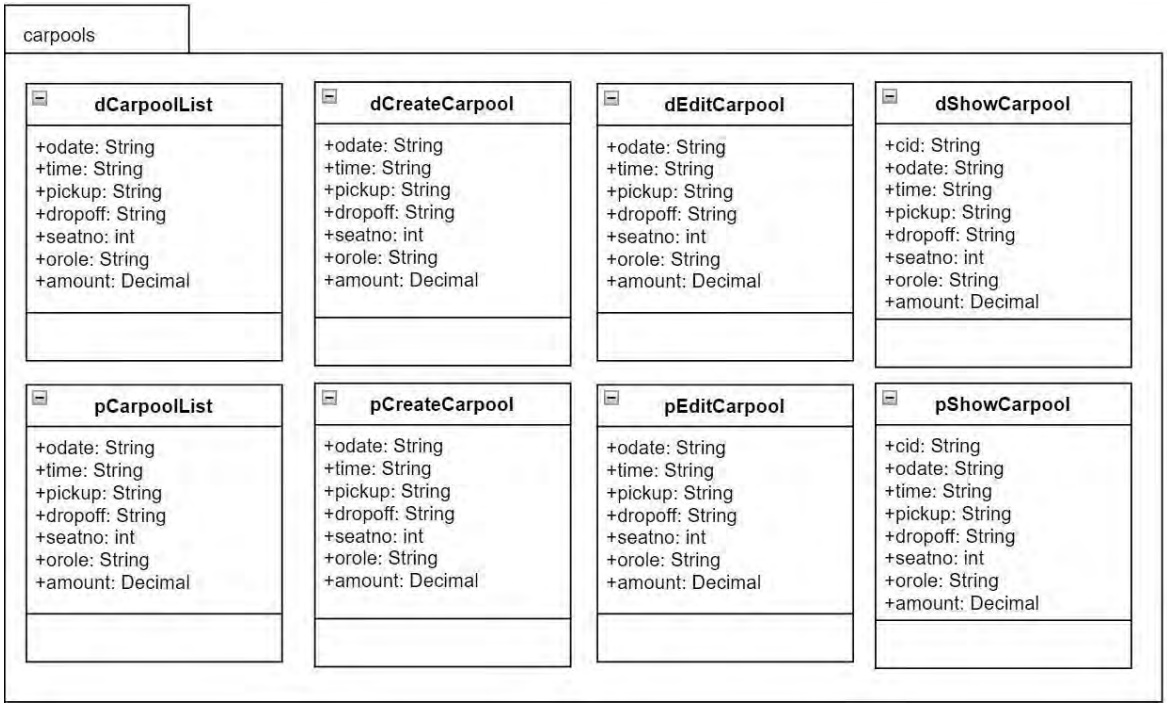


Figure 9 carpools

Interface	Description
dCarpoolList	Interface that allows driver to view the carpool listing.
dCreateCarpool	Interface that allows driver to create a carpool offer.
dEditCarpool	Interface that allows driver to update their carpool information.
dShowCarpool	Interface that allows driver to view specific carpool details.
pCarpoolList	Interface that allows passenger to view the carpool listing.
pCreateCarpool	Interface that allows passenger to create a carpool offer.
pEditCarpool	Interface that allows passenger to update their carpool information.
pShowCarpool	Interface that allows passenger to view specific carpool details.

1.4.3 Business Services Layer

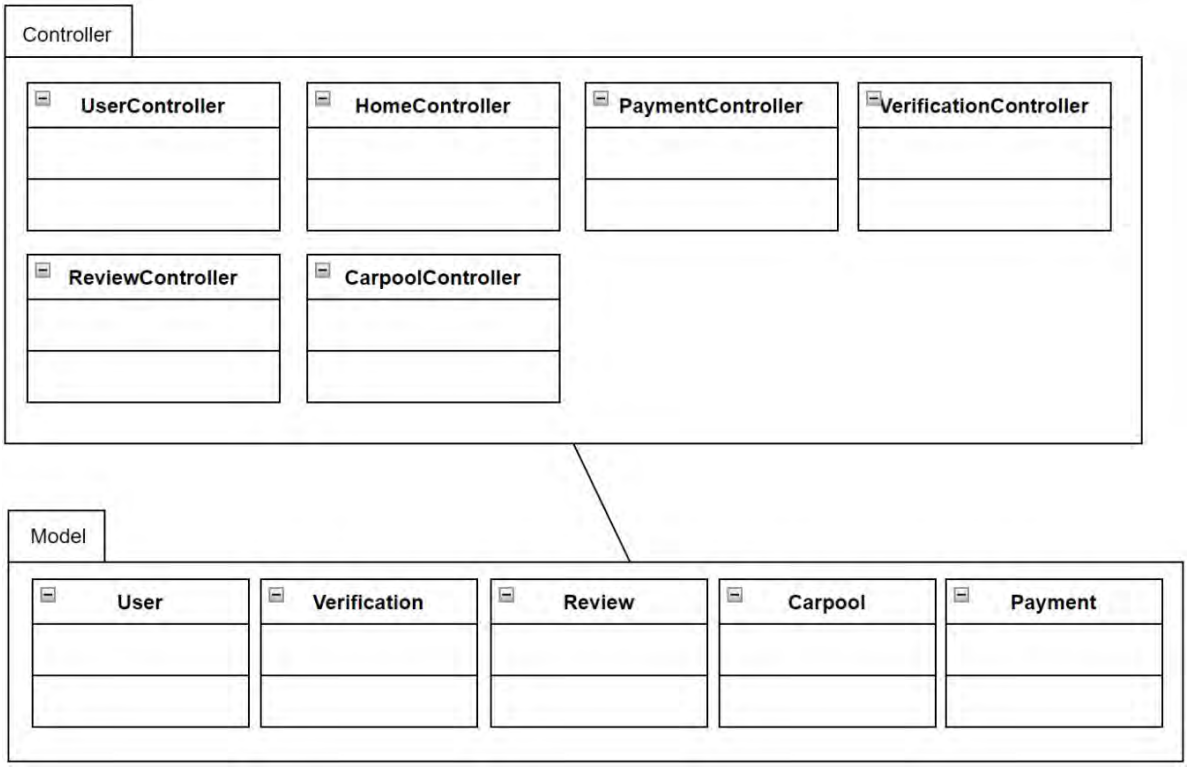


Figure 10 Business Services Layer of UACS

1.4.4 Controller

Class Name	Description
UserController	Controller is used to manage communication between the users module, user interface and the User Model.
HomeController	Controller is used to manage communication between the auth interface

	and the User Model.
PaymentController	Controller is used to manage communication between the payment interface and the Payment Model.
VerificationController	Controller is used to manage communication between the verification interface and the Verification Model.
ReviewController	Controller is used to manage communication between the reviews interface and the Review Model.
CarpoolController	Controller is used to manage communication between the carpools interface and the Carpool Model.

1.4.5 Model

Class Name	Description
User	This class connects the table of users on system database for storing or retrieving data as requested by the controller.
Verification	This class connects the table of Verification on system database for storing or retrieving data as requested by the controller.

Review	This class connects the table of review on system database for storing or retrieving data as requested by the controller.
Carpool	This class connects the table of Carpool on system database for storing or retrieving data as requested by the controller.
Payment	This class connects the table of payment on system database for storing or retrieving data as requested by the controller.

CHAPTER 2

2.1 DETAILED DESCRIPTION

2.1.1 Auth module

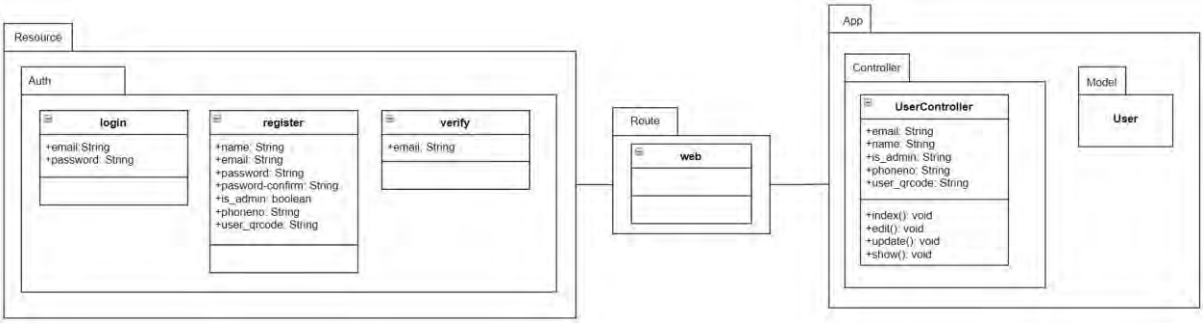


Figure 11 Detailed Architecture of auth module

Table 2.1 LoginPage Class Diagram Description

Class Type	Boundary Class	
Responsibility	This class is responsible for user to enter their account credentials or login using multi-factor authentication.	
Input	Email address and password.	
Output	User is logged in to the system and the system displays the main menu page.	
Attributes	**Attributes Name**	**Attributes Type**

	email password	String String
Methods	**Method Name**	**Description**
	N/A	N/A
Algorithm	N/A	

Table 2.2 Register Class Diagram Description

Class Type	Boundary Class	
Responsibility	This class is responsible for user to enter their account information or register using multi-factor authentication.	
Input	Username, password, email, contact number, role, and WhatsApp qr code.	
Output	An account is successfully created, and the user is asked to log in to their account.	
Attributes	**Attributes Name**	**Attributes Type**
	name email password is_admin	String String String String

	phoneno user_qrcode	String String
Methods	**Method Name**	**Description**
	N/A	N/A
Algorithm	N/A	

Table 2.3 Verify Class Diagram Description

Class Type	Boundary Class	
Responsibility	This class is responsible for user to enter their email address to send the verification message to their email.	
Input	email	
Output	Password is reset successfully, and the user is asked to log in to their account.	
Attributes	**Attributes Name**	**Attributes Type**
	email	String
Methods	**Method Name**	**Description**
	N/A	N/A
Algorithm	N/A	

Table 2.4 UserController Class Diagram Description

Class Type	Controller Class	
Responsibility	This class is responsible for managing user logins that communicates between the interface and model classes.	
Input	Email and password	
Output	User is logged in successfully.	
Attributes	**Attributes Name**	**Attributes Type**
	email password	String String
Methods	**Method Name**	**Description**
	index()	
	edit()	
	update()	
	show()	
Algorithm	BEGIN IF register() is called THEN create object of user model to connect with user table. Insert new user details in user table.	

	Display to **Register Success** message Return to User Login page. ELSE IF resetpass() is called THEN create object of user model to connect with user table. Update the new password in the user table. Display Reset Password Success message. Return to User Login page. ELSE IF login() is called THEN create object of user model to connect with user table. Verify the account credential in user table. Display result. Return to Main Menu page. END IF END

Table 2.5 User Model Class Diagram Description

Class Type	Model Class
Responsibility	This class connects the table of users on system database for storing

	or retrieving data as requested by the controller.	
Attributes	**Attributes Name**	**Attributes Type**
	Id email username password role phoneno user_qrcode Npassword ConfirmNpassword	String String String String String String String String String
Methods	**Method Name**	**Description**
	N/A	N/A
Algorithm	N/A	

2.1.2 Manage Profile

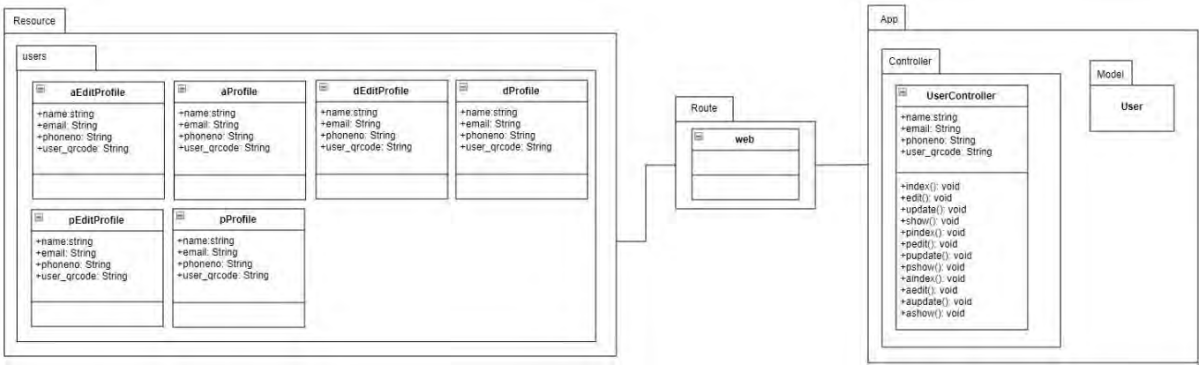


Figure 12 Manage Profile

Table 2.6 UpdateProfile Class Diagram Description

Class Type	Boundary Class	
Responsibility	This class is responsible for updating user profile that communicates between the interface and model classes.	
Input	Email, contact number and WhatsApp QR code	
Output	The system will display the latest user profile.	
Attributes	**Attributes Name**	**Attributes Type**
	email phoneno user_qrcode	String String String
Methods	**Method Name**	**Description**

	N/A	N/A
Algorithm	N/A	

Table 2.7 ViewUserProfilePage Class Diagram Description

Class Type	Boundary Class	
Responsibility	This class is responsible for prompting the latest profile that communicates between the interface and model classes.	
Input	Email, username, role, contact number and WhatsApp QR code	
Output	The system will display the latest profile.	
Attributes	**Attributes Name**	**Attributes Type**
	email username role phoneno user_qrcode	String String String String String
Methods	**Method Name**	**Description**
	N/A	N/A
Algorithm	N/A	

Table 2.8 ProfileController Class Diagram Description

Class Type	Controller Class	
Responsibility	This class is responsible for managing profile that communicates between the interface and model classes.	
Input	Email, username, role, contact number and WhatsApp QR code	
Output	The system will display the latest user profile.	
Attributes	**Attributes Name**	**Attributes Type**
	email username role phoneno user_qrcode	String String String String String
Methods	**Method Name**	**Description**
	update(Id)	To update the user profile information into database.
	viewUserProfile()	To remove the user profile information from database.
Algorithm	BEGIN	

	IF update() is called
	
	THEN create object of user model to connect with user table.
	
	Update the user data in the user table.
	
	Return to the latest user profile.
	
	ELSE
	
	THEN create object of user model to connect with user table.
	
	Return to the latest user profile.
	
	END IF
	
	END

2.1.3 Manage Carpool

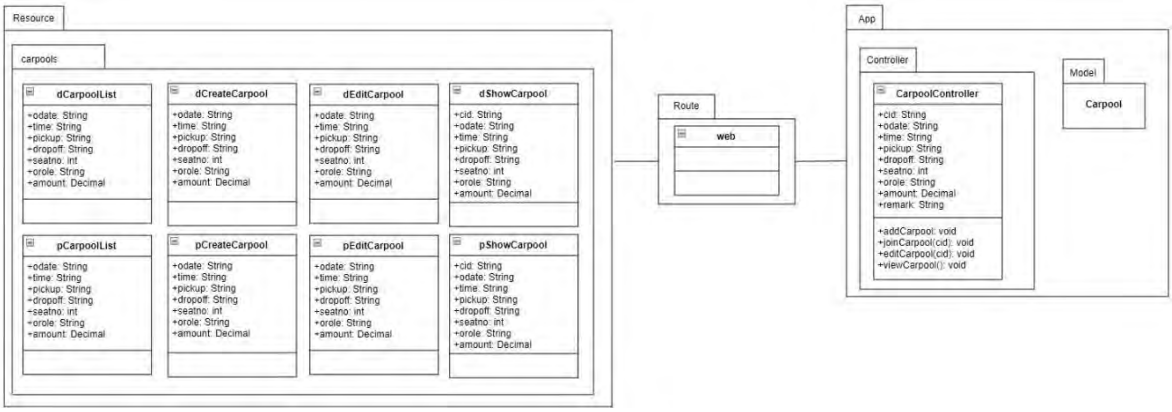


Table 2.9 AddCarpool Class Diagram Description

Class Type	Boundary Class	
Responsibility	This class is responsible for creating carpool that communicates between the interface and model classes.	
Input	odate, time, pickup, dropoff, seatno, orole and amount	
Output	The system will display the carpool listing that matches the keyword.	
Attributes	**Attributes Name**	**Attributes Type**
	odate	String
	time	String
	pickup	String
	dropoff	String

	seatno orole amount	Int String Decimal
Methods	**Method Name**	**Description**
	N/A	N/A
Algorithm	N/A	

Table 2.12 JoinCarpoolPage Class Diagram Description

Class Type	Boundary Class	
Responsibility	This class is responsible for accepting carpool confirmation that communicates between the interface and model classes.	
Input	remark	
Output	The system will display the latest carpool status.	
Attributes	**Attributes Name**	**Attributes Type**
	remark	String
Methods	**Method Name**	**Description**
	N/A	N/A
Algorithm	N/A	

Table 2.13 EditCarpoolPage Class Diagram Description

Class Type	Boundary Class	
Responsibility	This class is responsible for updating carpool offer that communicates between the interface and model classes.	
Input	odate, time, pickup, dropoff, seatno, orole and amount	
Output	The system will display the accept carpool offer successful message.	
Attributes	**Attributes Name**	**Attributes Type**
	odate	String
	time	String
	pickup	String
	dropoff	String
	seatno	Int
	orole	String
	amount	Decimal
Methods	**Method Name**	**Description**
	N/A	N/A
Algorithm	N/A	

Table 2.17 CarpoolController Class Diagram Description

Class Type	Boundary Class	
Responsibility	This class is responsible for manage carpool that communicates between the interface and model classes.	
Input	oId, odate, time, pickup, dropoff, seatno and orole	
Output	The system will display the latest carpool offers status and carpool listing.	
Attributes	**Attributes Name**	**Attributes Type**
	oId	String
	odate	Date
	time	Time
	pickup	String
	dropoff	String
	seatno	Int
	orole	String
Methods	**Method Name**	**Description**
	addCarpool()	To create new carpool offer in latest carpool listing.
	joinCarpool(oID)	To accept the carpool offer made

		by driver or passenger.
	editCarpool(oID)	To update the carpool offer created by user.
	viewCarpool()	To remove the carpool offer created by user.
Algorithm	BEGIN IF addCarpool() is called THEN create object of carpool model to connect with CarpoolDetails table. Insert new carpool details in CarpoolDetails table. Return to latest carpool offers status page. ELSE IF joinCarpool(oID) is called THEN create object of carpool model to connect with CarpoolDetails table. Insert carpool details in CarpoolDetails table. Display accept success message. Return to latest carpool offers status page. ELSE IF editCarpool(oID) is called THEN create object of carpool model to connect with	

	CarpoolDetails table. Update the carpool data in the CarpoolDetails table. Display update success message. Return to latest carpool offers status page. ELSE IF viewCarpool() is called Return to latest carpool offers status page. END IF END

2.1.4 Manage Payment

Payment Listing Interface

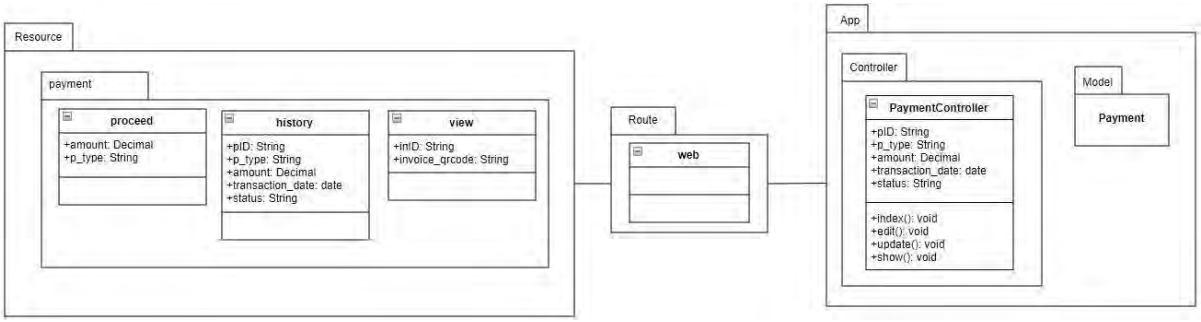


Table 2.18 Class Diagram of Payment Listing Interface

Class Type	Boundary Class
Responsibility	This class is responsible for displaying all pending payment that

	communicates between the interface and model classes.	
Input	Price, pickup, dropoff and time	
Output	The system will display all pending payment.	
Attributes	**Attributes Name**	**Attributes Type**
	price time pickup dropoff	Decimal Time String String
Methods	**Method Name**	**Description**
	N/A	N/A
Algorithm	N/A	

Accept Payment Interface

Table 2.19 Class Diagram of Accept Payment Interface

Class Type	Boundary Class
Responsibility	This class is responsible for accepting payment that communicates between the interface and model classes.
Input	p_type and price.

Output	The system will display the payment success message.	
Attributes	**Attributes Name**	**Attributes Type**
	p_type price	String Decimal
Methods	**Method Name**	**Description**
	N/A	N/A
Algorithm	N/A	

Cancel Payment Interface

Table 2.20 Class Diagram of Cancel Payment Interface

Class Type	Boundary Class	
Responsibility	This class is responsible for cancel payment that communicates between the interface and model classes.	
Input	N/A	
Output	The system will display the payment cancel success message	
Attributes	**Attributes Name**	**Attributes Type**
	N/A	N/A
Methods	**Method Name**	**Description**

	N/A	N/A
Algorithm	N/A	

View Payment Interface

Table 2.21 Class Diagram of View Payment Interface

Class Type	Boundary Class	
Responsibility	This class is responsible for displaying respective payment that communicates between the interface and model classes.	
Input	N/A	
Output	The system will display the latest carpool listing.	
Attributes	**Attributes Name**	**Attributes Type**
	odate time pickup dropoff seatno	Date Time String String int
Methods	**Method Name**	**Description**
	N/A	N/A

Algorithm	N/A

Manage Payment Controller

Table 2.22 Class Diagram of Manage Payment Controller

Class Type	Boundary Class	
Responsibility	This class is responsible for updating carpool that communicates between the interface and model classes.	
Input	odate, time, pickup, dropoff and seatno	
Output	The system will display all pending payment.	
Attributes	**Attributes Name**	**Attributes Type**
	remaindate	Int
Methods	**Method Name**	**Description**
	proceed()	To allows users to pay their carpool payment.
	cancel()	To allow users to cancel their carpool.
Algorithm	BEGIN IF proceed() is called THEN create object of payment model to connect with payment table.	

	Update the payment data in the payment table. Display payment success message. Return to the manage payment interface. ELSE IF delete() is called THEN create object of payment model to connect with payment table. Remove the payment data in the payment table. Return to the manage payment interface. END IF END

2.1.5 Manage Review

Manage Review Interface

Table 2.23 Class Diagram of Manage Review Interface

Class Type	Boundary Class
Responsibility	This class is responsible for managing review that communicates between the interface and model classes.
Input	N/A
Output	The system will display the latest review status.

Attributes	**Attributes Name**	**Attributes Type**
	rid rate comment	Int String String
Methods	**Method Name**	**Description**
	N/A	N/A
Algorithm	N/A	

Create Review Interface

Table 2.24 Class Diagram of Create Review Interface

Class Type	Boundary Class	
Responsibility	This class is responsible for creating review that communicates between the interface and model classes.	
Input	Rate and comment	
Output	The system will display the create success message.	
Attributes	**Attributes Name**	**Attributes Type**
	rate comment	Float String
Methods	**Method Name**	**Description**

	N/A	N/A
Algorithm	N/A	

Update Review Interface

Table 2.25 Class Diagram of Update Review Interface

Class Type	Boundary Class	
Responsibility	This class is responsible for updating review that communicates between the interface and model classes.	
Input	Rate and comment	
Output	The system will display update review success message.	
Attributes	**Attributes Name**	**Attributes Type**
	rate comment	Float String
Methods	**Method Name**	**Description**
	N/A	N/A
Algorithm	N/A	

View Review Interface

Table 2.26 Class Diagram of View Review Interface

Class Type	Boundary Class

Responsibility	This class is responsible for displaying respective review that communicates between the interface and model classes.	
Input	N/A	
Output	The system will display the latest review status.	
Attributes	**Attributes Name**	**Attributes Type**
	N/A	N/A
Methods	**Method Name**	**Description**
	N/A	N/A
Algorithm	N/A	

Delete Review Interface

Table 2.27 Class Diagram of Delete Review Interface

Class Type	Boundary Class	
Responsibility	This class is responsible for deleting review that communicates between the interface and model classes.	
Input	N/A	
Output	The system will display the delete review success message.	
Attributes	**Attributes Name**	**Attributes Type**
	N/A	N/A

Methods	**Method Name**	**Description**
	N/A	N/A
Algorithm	N/A	

Manage Review Controller

Table 2.28 Class Diagram of Manage Review Controller

Class Type	Boundary Class	
Responsibility	This class is responsible for managing review that communicates between the interface and model classes.	
Input	Rate and comment	
Output	The system will display the create success message.	
Attributes	**Attributes Name**	**Attributes Type**
	N/A	N/A
Methods	**Method Name**	**Description**
	create()	To create new review when the carpool is over.
	update()	To update review when changes are needed.
	view()	To display the list of review created by users

	delete()	To remove the respective review created by users.
Algorithm	BEGIN IF create() is called THEN create object of review model to connect with review table. Insert new review data in the review table. Display create success message. Return to the manage review interface. ELSE IF update() is called THEN create object of review model to connect with review table. Update the review data in the review table. Display update review success message. Return to the manage review interface. ELSE IF view() is called THEN create object of review model to connect with review table. Display the respective review.	

	Return to the manage review interface. ELSE IF delete() is called THEN create object of review model to connect with review table. Remove the review data in the review table. Display delete review success message. Return to the manage review interface. END IF END

2.1.6 Manage Driver Verification

Latest Driving Details Interface

Table 2.29 Class Diagram of Latest Driving Details Interface

Class Type	Boundary Class
Responsibility	This class is responsible for displaying driving details for verification that communicates between the interface and model classes.
Input	carno, Ncapacity, d_license, dExpDate and status
Output	The system will display the current record of driving details.

Attributes	**Attributes Name**	**Attributes Type**
	carno Ncapacity d_license dExpDate status	String Int String Date String
Methods	**Method Name**	**Description**
	N/A	N/A
Algorithm	N/A	

Confirmation Delete Driving Details Interface

Table 2.30 Class Diagram of Confirmation Delete Driving Details Interface

Class Type	Boundary Class	
Responsibility	This class is responsible for displaying confirmation of deleting driving details that communicates between the interface and model classes.	
Input	N/A	
Output	The system will display delete driving details success message	
Attributes	**Attributes Name**	**Attributes Type**

	N/A	N/A
Methods	**Method Name**	**Description**
	N/A	N/A
Algorithm	N/A	

Update Driving Details Interface

Table 2.31 Class Diagram of Update Driving Details Interface

Class Type	Boundary Class	
Responsibility	This class is responsible for updating driving details that communicates between the interface and model classes.	
Input	carno, Ncapacity, d_license, dExpDate and status	
Output	The system will display update driving verification success message.	
Attributes	**Attributes Name**	**Attributes Type**
	carno Ncapacity d_license dExpDate status	String Int String Date String

Methods	**Method Name**	**Description**
	N/A	N/A
Algorithm	N/A	

Create Driving Details Interface

Table 2.32 Class Diagram of Create Driving Details Interface

Class Type	Boundary Class	
Responsibility	This class is responsible for creating driving details that communicates between the interface and model classes.	
Input	Carno, Ncapacity, d_license, dExpDate and status.	
Output	The system will display view driver verification.	
Attributes	**Attributes Name**	**Attributes Type**
	carno Ncapacity d_license dExpDate status	String Int String Date String
Methods	**Method Name**	**Description**

	N/A	N/A
Algorithm	N/A	

Manage Driving Verification Controller

Table 2.33 Class Diagram of Manage Driving Verification Controller

Class Type	Boundary Class	
Responsibility	This class is responsible for managing the driving verification that communicates between the interface and model classes.	
Input	carno, Ncapacity, d_license, dExpDate and status.	
Output	The system will display manage driver verification interface.	
Attributes	**Attributes Name**	**Attributes Type**
	carno Ncapacity d_license dExpDate status	String Int String Date String
Methods	**Method Name**	**Description**
	create()	To allow users to create new driving details.

	update()	To allow users to update their driving details.
	delete()	To allow users to remove their driving details.
Algorithm	BEGIN IF create() is called THEN create object of VerificationDetailsmodel to connect with VerificationDetails table. Insert new driving data in the VerificationDetails table. Display create success message. Return to the manage driving verification interface. ELSE IF update() is called THEN create object of VerificationDetails model to connect with VerificationDetails table. Update the driving data in the review table. Display update success message. Return to the manage driver verification interface. ELSE IF delete() is called THEN create object of VerificationDetails model to connect	

	with VerificationDetails table. Remove the driving data in the VerificationDetails table. Display delete success message. Return to the manage driver verification interface. END IF END

2.2 DATA DICTIONARY

2.2.1 User

Table 2.34 Data Dictionary of User

Data Name	Data Type	Description	Module	Constraint
Id	varchar(15)	User identifier	Manage User Login Manage Profile Manage Driver Verification	PK
username	varchar(15)	User Name	Manage User Login	-
email	varchar(30)	Email of user	Manage User Login Manage Profile	-
password	varchar(12)	User password	Manage User Login	-

role	varchar(10)	Role of user (driver / passenger)	Manage User Login	-
phoneno	varchar(15)	Contact information	Manage User Login Manage Profile	-
qrcode_1	varchar(30)	WhatsApp QR code	Manage User Login Manage Profile	-

2.2.2 CarpoolDetails

Table 2.35 Data Dictionary of CarpoolDetails

Data Name	Data Type	Description	Module	Constraint
oId	varchar(15)	Offer identifier	Manage Carpool	PK
odate	varchar(15)	Offer date	Manage Carpool	-
time	varchar(30)	Time to pick up (24-hour clock)	Manage Carpool	-
pickup	varchar(12)	Pick up location	Manage Carpool	-
dropoff	varchar(10)	Drop off location	Manage Carpool	-
seatno	varchar(15)	Number of seat available	Manage Carpool	-
orole	varchar(30)	Role of offer	Manage Carpool	-

2.2.3 Payment

Table 2.36 Data Dictionary of Payment

Data Name	Data Type	Description	Module	Constraint
pID	varchar(15)	Payment identifier	Manage Payment	PK
p_type	varchar(10)	Payment method	Manage Payment	-
price	decimal (10,2)	Total split per passenger in carpool	Manage Payment Manage Carpool	-
remaindate	date	The number of remaining date to cancel automatically	Manage Payment	-
qrcode_2	varchar(30)	Invoice QR code	Manage Payment	-

2.2.4 Review

Table 2.39 Data Dictionary of Review

Data Name	Data Type	Description	Module	Constraint
rId	varchar(12)	Review Identifier	Manage Review	PK
pID	varchar(12)	Payment Identifier	Manage Payment Manage Review	FK
rate	float	Rating	Manage Review	-
comment	varchar(12)	Comment of Carpool	Manage Review	-

2.2.5 VerificationDetails

Table 2.40 Data Dictionary of VerificationDetails

Data Name	Data Type	Description	Module	Constraint
d_license	varchar(30)	Driving license	Manage Driver Verification	PK
Id	varchar(15)	User identifier	Manage User Login Manage Profile Manage Driver Verification	FK
dExpDate	date	Driving lisence expire date	Manage Driver Verification	-
status	varchar(15)	Status of verification	Manage Driver Verification	-