



**Module Code & Module Title**

CS6PO5NT Final Year Project

**Assessment Weightage & Type**

25% FYP Interim Report

**Year and Semester**

2022-23 Autumn

**Student Name:** Susan Shrestha

**London Met ID:** 20048536

**College ID:** NP05CP4S210004

**Internal Supervisor:** Mr. Rabi Rouniyar

**External Supervisor:** Mr. Shekhar Timisina

**Assignment Due Date:** 2022-12-28

**Assignment Submission Date:** 2022-12-28

**Project Title:** Ride Hero - Ride Sharing Web App

*I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.*

**Abstract**

Nepal's transportation sector is faced with a range of challenges that can make it difficult for people to access essential services and opportunities. One significant issue is the poor transportation infrastructure. This can be a major barrier for individuals trying to access healthcare, education, and other essential services. Additionally, Nepal lacks a robust and efficient public transportation system, particularly in urban areas, which can further limit access to essential services. The cost of transportation can also be high, which can be a financial burden for many, particularly those living in poverty. Traffic congestion, particularly in the capital city of Kathmandu, can also be a major inconvenience for people trying to get around. Lastly, the transportation sector in Nepal is a major contributor to air pollution and greenhouse gas emissions, which poses environmental challenges. Overall, these challenges can make it difficult for people in Nepal to access essential services. To address these issues, it will be important to focus on developing a more flexible and adaptable transportation system that is able to respond to the needs of the population and the changing environment.

This project is all about developing a ride-sharing web app which is an online platform that connects drivers and passengers, providing a convenient and cost-effective alternative to traditional transportation methods. This web app allows passengers to request a ride from their location, and this web app will match them with a nearby driver who can accept the ride request and pick up the passenger. This web app also handles payment for the ride, making the entire process seamless and easy for both parties. This app can be accessed on web browsers, allowing passengers to request and offer rides from anywhere. This type of platform is particularly useful for people who do not own a car, or for those who want a more convenient and cost-effective way to get around. It also provides an opportunity for drivers to earn extra income by offering rides to others. Overall, this project offers a convenient and efficient solution for both passengers and drivers, making it a popular choice for transportation.

## Table of Contents

1. Introduction .....	1
1.1. Project Introduction .....	1
1.2. Problem Scenario .....	2
1.3. Project as a Solution .....	2
1.4. Aims and Objectives .....	3
2. Background .....	4
2.1. Technology used .....	4
2.1.1. Web Browser .....	4
2.1.2. Server Side Scripting Language – PHP .....	4
2.1.3. IDE – Visual Studio Code .....	4
2.1.4. Web Design .....	5
2.1.5. Framework – Laravel .....	5
2.1.6. Web Server – XAMPP .....	5
2.1.7. Database Management System – MYSQL .....	6
2.2. Similar Systems .....	7
2.2.1. Similar System Consideration .....	7
2.2.2. Similar System Comparison .....	10
3. Development .....	11
3.1. Methodology .....	11
3.1.1. Methodology Consideration .....	12
3.1.2. Selected Methodology .....	15
3.1.3. Methodology Section Justification .....	16
4. Work To Date .....	18
4.1. Requirement Gathering .....	18

4.1.1. Conducted Online Survey .....	18
4.2. Use Case Diagram .....	23
4.2.1. Individual Use Case Diagram.....	23
4.2.2. System Use Case Diagram .....	27
4.3. High Level Use Case .....	28
4.4. Activity Diagram.....	30
4.4.1. Passenger Activity Diagram .....	30
4.4.2. Driver Activity Diagram.....	33
4.5. Sequence Diagram .....	36
4.6. System Architecture Diagram .....	37
4.7. Class Diagram .....	38
4.8. Wireframes .....	39
4.8.1. Login Page .....	39
4.8.2. Register Page .....	40
4.8.3. Home Page .....	41
4.8.4. Passenger Interface Page.....	42
4.8.5. Driver Interface Page .....	43
4.8.6. Admin Dashboard .....	44
4.9. Entity Relationship Diagram .....	45
4.10. Data Flow Diagram (DFD) .....	46
4.11. Milestone Chart.....	47
4.12. Work Breakdown Structure (WBS) .....	48
4.13. Gantt Chart .....	49
4.14. Analysis of Progress .....	50
4.14.1. Progress Table.....	51

4.14.2. Progress Review .....	52
4.14.3. Project Timeline .....	53
4.14.4. Action Plan .....	54
5. Future Work .....	55
6. Conclusion .....	57
7. References.....	58
8. Appendix .....	61
8.1. Appendix – A: Web Application Features .....	61
8.2. Appendix – B: Phases of RUP Methodology .....	63
8.3. Appendix – C: Survey Result.....	65
8.4. Appendix – D: Individual Use Case .....	68
8.5. Appendix – E: Activity Diagram.....	72
8.6. Appendix – F: Sequence Diagrams .....	76
8.7. Appendix – G: Architecture Diagram .....	77
8.8. Appendix – H: Class Diagram.....	77
8.9. Appendix – I: Wireframes .....	78
8.10. Appendix – J: ERD Details .....	88
8.11. Appendix – K: DFD details.....	90
8.12. Appendix – L: Milestone Chart.....	91
8.13. Appendix – M: WBS.....	92
8.14. Appendix – N: Gantt Chart.....	93
8.15. Appendix – O: SRS.....	94
1. Introduction .....	94
2. Overall Description.....	95

## List of Figures

Figure 1: Uber .....	7
Figure 2: OLA.....	8
Figure 3: Lyft .....	9
Figure 4: Scrum Methodology .....	12
Figure 5: DSDM Methodology .....	13
Figure 6: Extreme Programming Methodology .....	14
Figure 7: RUP Methodology .....	15
Figure 8: Survey 1 .....	18
Figure 9: Survey 2.....	19
Figure 10: Survey 3.....	20
Figure 11: Survey 4.....	21
Figure 12: Survey 5.....	22
Figure 13: Individual Use Case of Admin Manage Users .....	23
Figure 14: Individual Use Case of Admin Payment Management .....	23
Figure 15: Individual Use Case of Driver Ride History .....	24
Figure 16: Individual Use Case of KYC Verification .....	24
Figure 17: Individual Use Case of Driver Manage Ride.....	25
Figure 18: Individual Use Case of Passenger Ride .....	26
Figure 19: Individual Use Case of Passenger Ride History .....	26
Figure 20: System Use Case Diagram .....	27
Figure 21: Passenger Ride Request Activity Diagram.....	30
Figure 22: Passenger Location Detection Activity Diagram.....	31
Figure 23: Final Passenger Activity Diagram .....	32
Figure 24: Driver KYC Verification Activity Diagram.....	33
Figure 25: Driver Manage Ride Activity Diagram .....	34
Figure 26: Final Driver Activity Diagram .....	35
Figure 27: Sequence Diagram .....	36
Figure 28: System Architecture Diagram.....	37
Figure 29: Class Diagram.....	38

Figure 30: Login Page .....	39
Figure 31: Register Page .....	40
Figure 32: Home Page .....	41
Figure 33: Passenger Interface Page.....	42
Figure 34: Driver Interface Page .....	43
Figure 35: Admin Dashboard .....	44
Figure 36: Entity Relationship Diagram .....	45
Figure 37: Data Flow Diagram .....	46
Figure 38: Milestone Chart .....	47
Figure 39: Work Breakdown Structure .....	48
Figure 40: Gantt Chart.....	49
Figure 41: Survey 6.....	65
Figure 42: Survey 7 .....	65
Figure 43: Survey 8.....	66
Figure 44: Survey 9.....	66
Figure 45: Survey 10 .....	67
Figure 46: Survey 11 .....	67
Figure 47: Individual Use Case of Admin Login .....	68
Figure 48: Individual Use Case of Admin Logout .....	68
Figure 49: Individual Use Case of Driver Registration.....	69
Figure 50: Individual Use Case of Driver Login .....	69
Figure 51: Individual Use Case Diagram of Driver Logout .....	70
Figure 52: Individual Use Case of Passenger Registration .....	70
Figure 53: Individual Use Case of Passenger Login .....	71
Figure 54: Individual Use Case of Passenger Logout .....	71
Figure 55: Passenger Register Activity Diagram .....	72
Figure 56: Passenger Login Activity Diagram .....	73
Figure 57: Driver Register Activity Diagram .....	74
Figure 58: Driver Login Activity Diagram .....	75
Figure 59: Login Sequence Diagram.....	76
Figure 60: Reset Password Page.....	78

Figure 61: Account Recovery Page.....	79
Figure 62: Change Password Page.....	80
Figure 63: MY KYC Page .....	81
Figure 64: My Vehicle Page .....	82
Figure 65: My Rides Page.....	83
Figure 66: My Earnings Page.....	84
Figure 67: My Settings Page .....	85
Figure 68: My Ride History Page .....	86
Figure 69: My Settings Page .....	87
Figure 70: Previous Milestone Chart .....	91
Figure 71: Previous Work Breakdown Structure .....	92
Figure 72: Previous Gantt Chart.....	93



## Table of Tables

Table 1: Comparison between similar projects.....	10
Table 2: Justification 1.....	16
Table 3: Justification 2.....	16
Table 4: Justification 3.....	17
Table 5: High Level Use Case.....	28
Table 6: Progress Table .....	51
Table 7: Progress Table Timeline .....	53
Table 8: Future Work Table.....	55

## 1. Introduction

### 1.1. Project Introduction

Ride-sharing refers to the use of a digital platform to connect passengers with drivers for the purpose of transporting passengers from one location to another. Ride-sharing services have become increasingly popular in recent years as a convenient and cost-effective alternative to traditional taxi services. A ride-sharing web app is a software application that allows users to request and pay for rides through their internet browser. These apps typically use GPS technology to match passengers with nearby drivers and to track the progress of the ride. Passengers can usually rate their experience and leave reviews for drivers. This feedback system helps to ensure and maintain the overall quality of the service. Ride-sharing web apps offer a number of benefits to both passengers and drivers. For passengers, they provide an easy way to get around without the need to own a vehicle or deal with the inconvenience of public transportation. For drivers, they offer the opportunity to earn money by providing rides to people in their community.

This project is a web-based app through which is a popular alternative to traditional modes of transportations, such as taxis and public transportation. This web app provides on-demand ride-sharing services through several modes of transportation such as cars, bikes and tuk-tuks. This web app is a revolutionary transportation solution that connects passengers with drivers in a convenient and cost-effective manner through an online platform. This web app offers a quick and easy way for passengers and drivers to connect and coordinate rides. When a passenger requests a ride, drivers can choose to accept or decline the fare, and if accepted, the passenger will receive a confirmation message. With just a single click, both passengers and drivers can request or cancel rides, making the booking process simple and efficient. The admin has full control over the web application, including the ability to add, delete, and update passenger and driver information, as well as authenticate personal information as needed. To ensure safety, the admin also verifies that drivers have valid licenses before allowing them to work on the platform. All payment transactions are handled digitally and monitored by the admin, ensuring a secure and streamlined experience for all users. Overall, this web app is a reliable and convenient transportation solution that benefits both passengers and drivers.

## 1.2. Problem Scenario

The following are the problem scenarios:

- Limited transportation options can make it difficult for people to access necessary services.
- Transportation can be costly, especially for those with a limited income or who need to travel frequently.
- Transportation may not be accessible to everyone, including those with disabilities or mobility challenges.
- Public transportation may experience congestion, leading to delays and reduced reliability.

## 1.3. Project as a Solution

This project is considered as problem-solving web app because this web app address a variety of problems related to transportation, convenience, cost, the environment, and income generation.

This project can solve a number of problems, including:

- This project can be alternative to car ownership or public transportation.
- This project allows easy request and payment for rides.
- This project can be more affordable, especially for long distances or multiple passengers.
- This project enables income generation through driving for the app.
- This project improves accessibility for those with disabilities or mobility issues.

## 1.4. Aims and Objectives

The main aim of this project is to provide a convenient, affordable and safe ride for people to get from one place to another. This web app generally works by connecting passengers with drivers who are willing to provide a ride. Passengers can request and pay for their ride, allowing them to track their driver in real-time, while at the same time the driver can see the exact location of the passenger and have access to the payment once the ride is complete.

In order to achieve this aim, the following objectives have been set up which are listed below:

### Objectives

- To Conduct an analysis of the major terms and resources needed for the web app development.
  - To develop a complete web application using the agile methodology.
  - To create a proper format for the project documentation.
  - To build wireframes and GUI for the web app.
  - To facilitate a convenient way for users to request rides.
  - To evaluate the impact of the web app on users.
  - To learn about web platforms and development tools.
- To document the development and execution of the web app.

Refer to [Appendix – A](#) for Web Application Features of Admin, Passenger and Driver Features

## **2. Background**

### **2.1. Technology used**

To create my project “Ride Hero – Ride Sharing Web App”, I would use a combination of front-end technologies such as HTML, CSS, and JavaScript, as well as back-end technologies like PHP and MySQL. I might also use a JavaScript library like React JS to build user interfaces and interactive features. These technologies would be used with web browsers like Chrome or Edge to create a functional web application that meets the needs of users and provides a seamless ride-requesting experience.

In this section of the report, I will provide a brief overview of the technologies that have been used in the development of my project.

#### **2.1.1. Web Browser**

##### **a) Google Chrome**

Google Chrome is a fast, secure, and stable web browser that can sync with a user's Google account to access bookmarks, history, and settings across devices (Moreau, 2022).

##### **b) Microsoft Edge**

Microsoft Edge is a fast web browser with a clean interface and support for modern web standards, included in the Windows operating system as a replacement for Internet Explorer.

#### **2.1.2. Server Side Scripting Language – PHP**

PHP is a server-side language used to create dynamic websites. It is open-source and integrates with HTML to display different content based on user input or variables. PHP can perform various tasks for modern web development, such as connecting to databases and processing forms (Jackson, 2022).

#### **2.1.3. IDE – Visual Studio Code**

Visual Studio Code (VS Code) is a free, open-source code editor with fast performance, extensive features, and support for various languages and platforms. It includes an integrated debugger, code completion, syntax highlighting, and support for version control systems (Mustafeez, 2022).

#### **2.1.4. Web Design**

##### **a) HTML**

HTML is a markup language used to create web pages by specifying their layout, including headings, paragraphs, lists, links, images, and other elements. It can also create forms for user data and interaction (javatpoint, 2022).

##### **b) CSS**

CSS is a stylesheet language used to control the layout, color, font, and other design elements of an HTML document. It separates content from presentation and can create responsive designs that adapt to different screens and devices (techopedia, 2018).

##### **c) JavaScript**

JavaScript is a programming language used to create interactive and dynamic web pages. It is essential for modern web development and is supported by all major browsers. JavaScript adds functionality to web pages and can create games, applications, and other interactive experiences (javatpoint, 2022).

##### **d) React JS**

React is a JavaScript library for building user interfaces. It helps developers create reusable, scalable, and maintainable components for complex web applications. React uses a virtual DOM to optimize component rendering and improve performance (tutorialspoint, 2022).

#### **2.1.5. Framework – Laravel**

Laravel is an open-source PHP framework for web application development. It provides tools and libraries for tasks like routing, authentication, and database management, and has a clean and expressive syntax (Laravel, 2022).

#### **2.1.6. Web Server – XAMPP**

XAMPP is a free, open-source cross-platform web server stack including Apache HTTP Server, MariaDB database, and PHP and Perl interpreters. It is designed for easy installation and use as a local development environment for web applications (javatpoint, 2022).

**2.1.7. Database Management System – MYSQL**

MySQL is a widely used open-source database system for web development, capable of storing, organizing, and retrieving data. It is fast, reliable, and easy to use, often paired with PHP and other web technologies for dynamic websites. It supports various data types for small and large projects (Moore, 2022).

## 2.2. Similar Systems

### 2.2.1. Similar System Consideration

#### a) System 1

System Name: Uber

URL: <https://www.uber.com/>

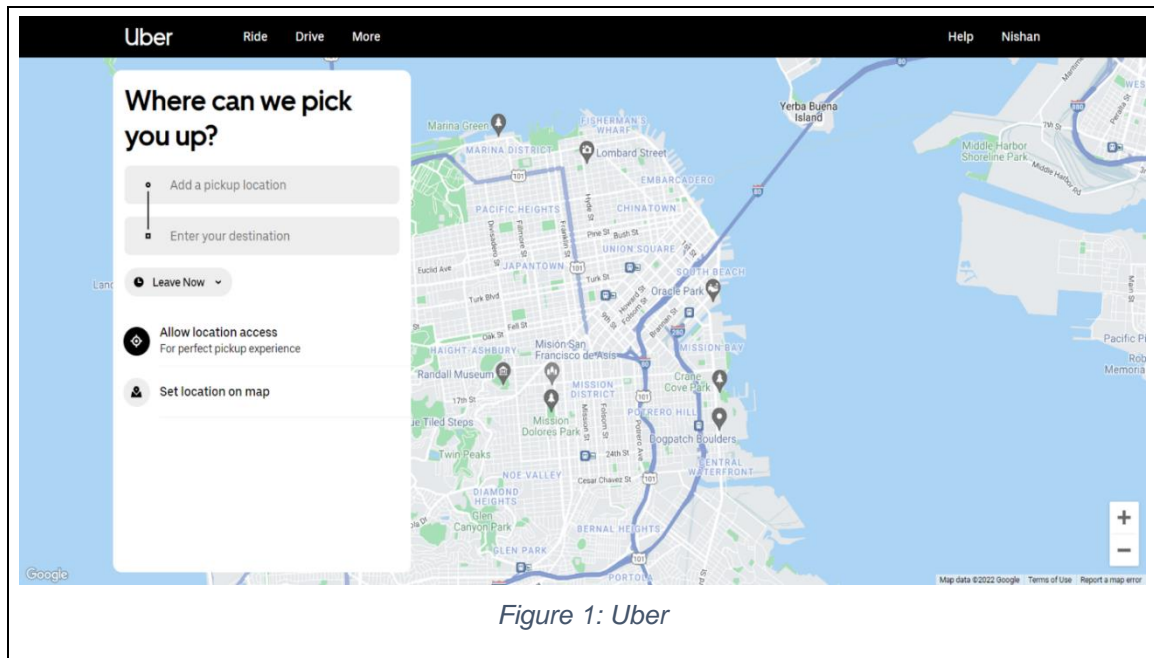


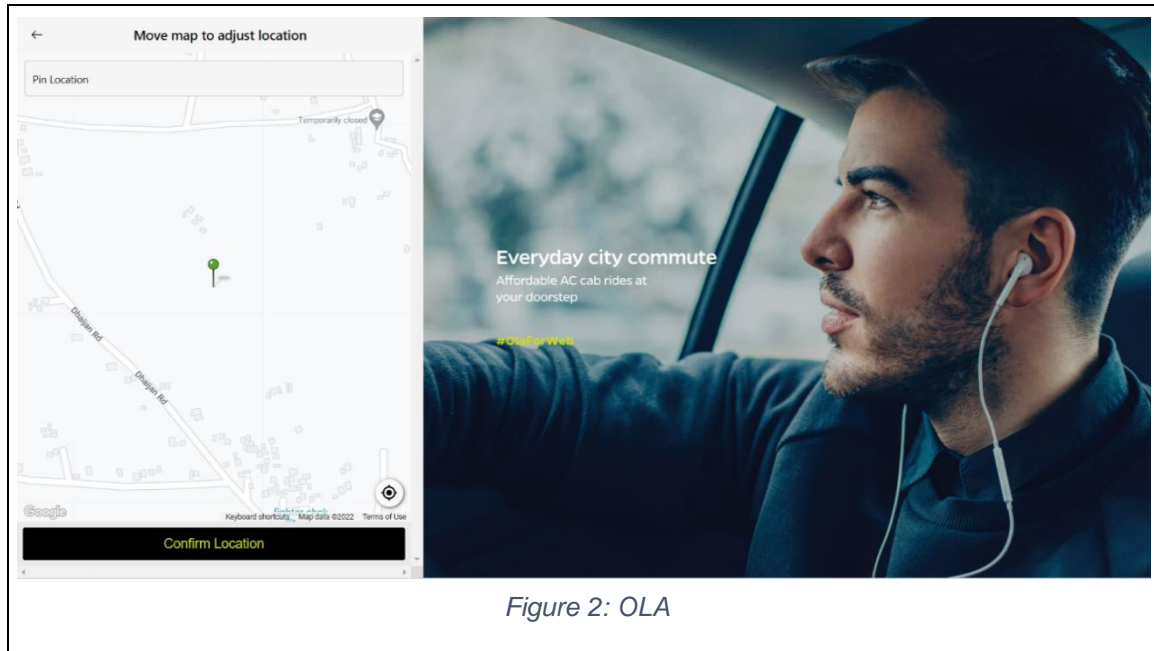
Figure 1: Uber

Uber is a technology company that offers a range of transportation services to consumers around the world. The company was founded in 2009 and has since become a major player in the sharing economy, operating in more than 10,000 cities in over 65 countries. To use Uber's ride-hailing service, users download the Uber app on their smartphone and create an account. They can then request a ride from a nearby Uber driver by entering their destination into the app. Payment is automatically charged to the user's credit card or PayPal account, and users can rate their driver and provide feedback about their experience (BLYSTONE, 2022).



**b) System 2**

System Name: OLA

URL: <https://www.olacabs.com/>*Figure 2: OLA*

Ola is a technology company that offers a range of transportation services, including ride-hailing, auto-rickshaw hailing, and food delivery. The company was founded in 2011 in Mumbai, India and has since expanded to operate in over 250 cities in India and a number of other countries including Australia, New Zealand, and the United Kingdom. To use Ola's ride-hailing service, users download the Ola app on their smartphone and create an account. They can then request a ride from a nearby Ola driver by entering their destination into the app. Payment is automatically charged to the user's credit card or linked bank account, and users can rate their driver and provide feedback about their experience on the ride (engineeringforchange, 2022).

**c) System 3**

System Name: Lyft

URL: <https://www.lyft.com/>

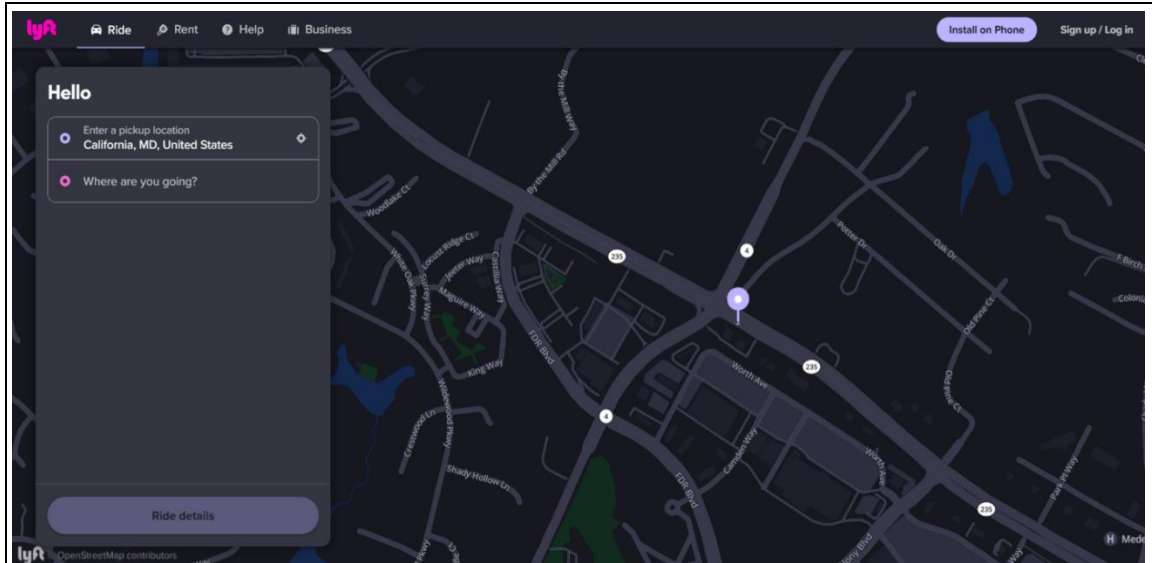


Figure 3: Lyft

Lyft is a technology company that offers a ride-hailing service to consumers in the United States and Canada. The company was founded in 2012 and has since become a major player in the North American ride-hailing market, operating in more than 600 cities across the two countries. To use Lyft's service, users download the Lyft app on their smartphone and create an account. They can then request a ride from a nearby Lyft driver by entering their destination into the app. Payment is automatically charged to the user's credit card or linked bank account, and users can rate their driver and provide feedback about their experience (DAVIS, 2022).

### 2.2.2. Similar System Comparison

There are several ride-sharing system that have similar models to my FYP project which I have mentioned above. Here is the comparison of my system to the above mentioned system are shown in the table below:

*Table 1: Comparison between similar projects*

S.N	Features	Uber	Ola	Lyft	My project
1	Registration and login	✓	✓	✓	✓
2	Tuk-tuk service	✗	✗	✗	✓
3	Online payment	✓	✓	✓	✓
4	Offline payment	✓	✓	✓	✓
5	Rating and feedback	✓	✓	✓	✓
6	Bargain option	✗	✗	✗	✓

In conclusion, my project as ride-sharing web app, "**Ride Hero**", offers a unique and attractive feature not available on other platforms: the ability to negotiate fare prices. This, along with the option to request rides on traditional tuk-tuk vehicles, sets our app apart and makes it a cost-effective and culturally relevant option for passengers. With advanced features and efficient system for matching passengers with drivers, my project offers a better experience for both parties. If all features are implemented successfully, "Ride Hero" is sure to be a successful project.

### **3. Development**

#### **3.1. Methodology**

Methodology in software refers to the systematic approach to the development and maintenance of software systems. It includes the methods and techniques used to plan, design, implement, test, and maintain software, as well as the principles and frameworks that guide the software development process. The choice of methodology depends on the goals and constraints of the project, and may vary widely depending on the size, complexity, and intended audience of the software (Jr., 1995).

Some benefits of using a software methodology are as follows:

- It provides structure and predictability to the software development process.
- It improves communication and collaboration within the development team.
- It reduces the risk of project failure.
- It increase the quality of the software.
- It increases efficiency and optimizes workflows.
- It helps to manage scope, schedule, and budget.

### 3.1.1. Methodology Consideration

When considering a software methodology for a project, it is essential to consider a number of factors in order to select the methodology that is best suited to the project. The nature of the software being produced, the preferences and demands of the development team, the resources available, the software's specifications, and the needs of the end users are all important factors to consider. The ideal software methodology will be selected based on a combination of these and other criteria, and it is necessary to thoroughly assess all of these factors in order to select the best methodology for the project.

#### a) Scrum Methodology

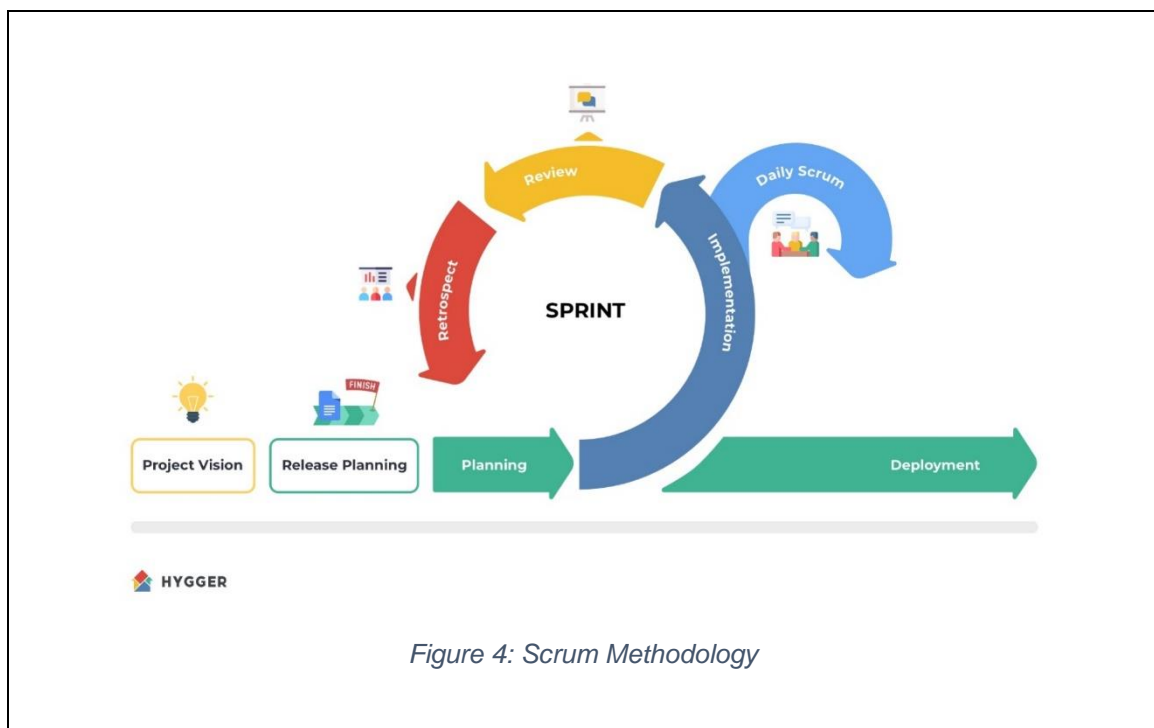
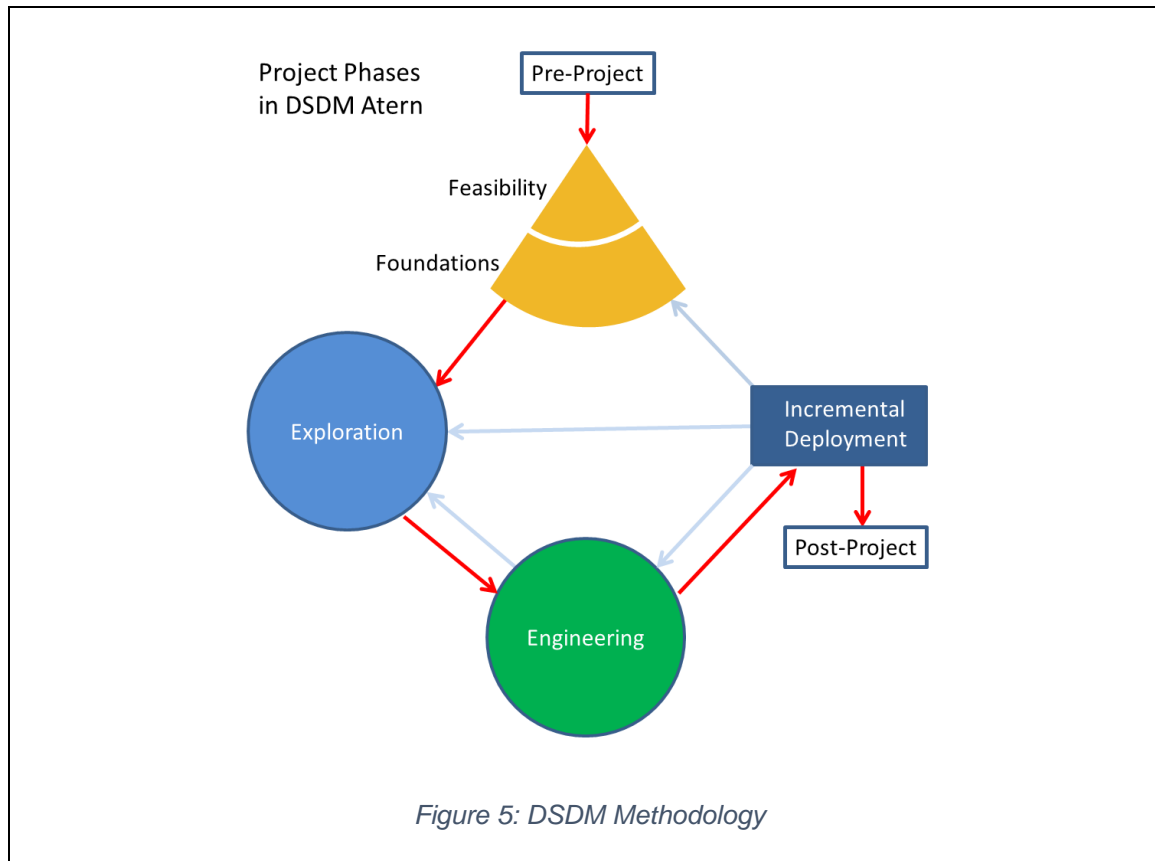


Figure 4: Scrum Methodology

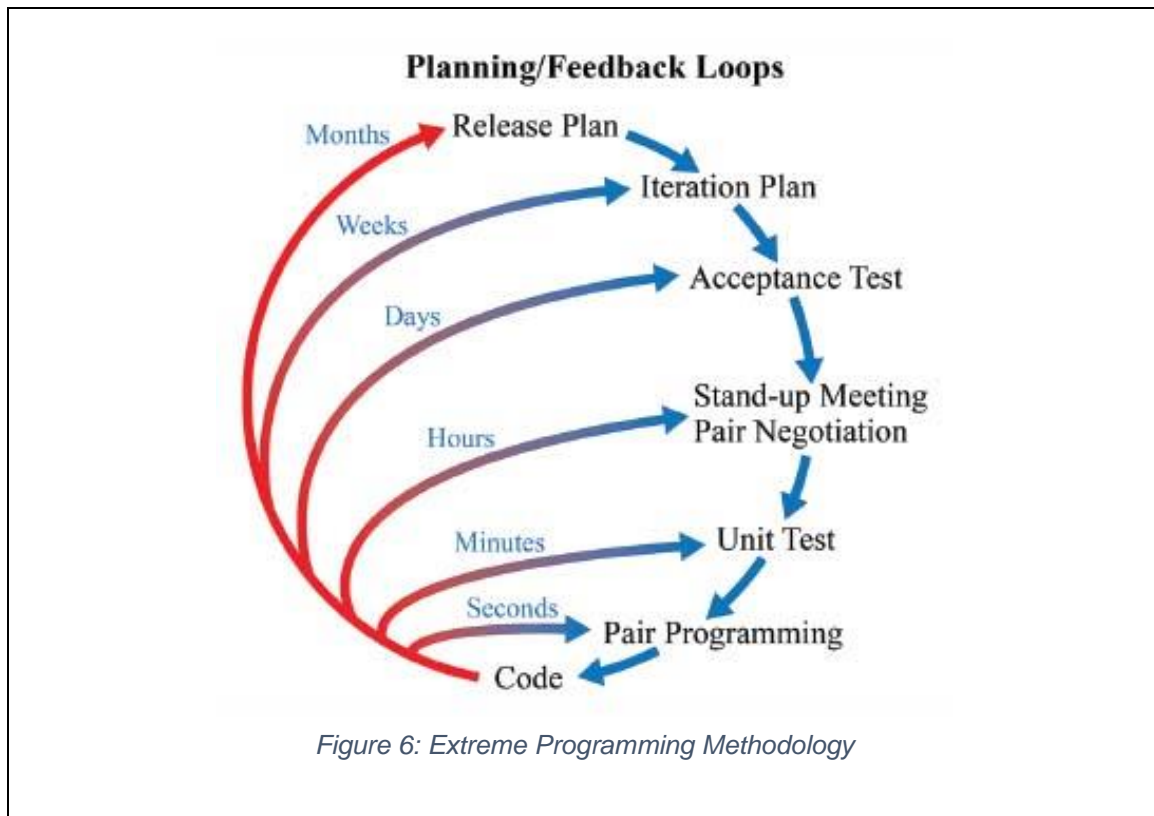
Scrum is an agile methodology for managing large projects, particularly in software development. It involves working in short, iterative cycles called "sprints" to develop functional software. The approach emphasizes transparency, inspection, and adaptability and encourages frequent communication and collaboration among team members. In Scrum, the team meets daily to check in with each other and ensure that the task is on track. At the end of each sprint, the team holds a

feedback session to present completed work and address any issues that have arisen (Schwaber, 2001).

## b) DSDM

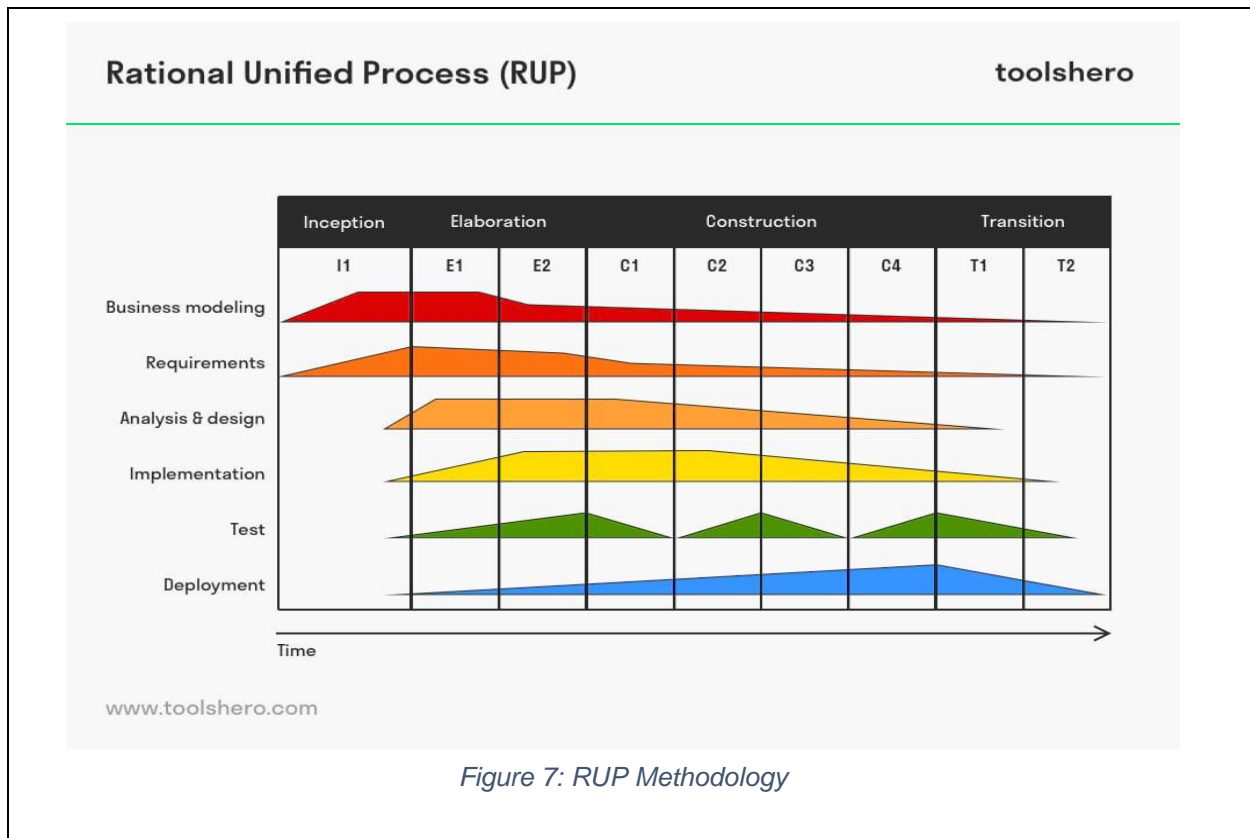


DSDM (Dynamic Systems Development Method) is an agile methodology for managing the development of software and other systems. It emphasizes active user involvement, frequent delivery, and focus on the business need and encourages the delivery of working software quickly and getting frequent feedback from stakeholders. Teams work in short, iterative cycles called "timeboxes" and hold regular progress reviews and review meetings to ensure the work is on track. This methodology is well-suited for managing complex projects (Bentley, 2010).

**c) Extreme Programming (XP)**

Extreme Programming (XP) is an agile software development process that prioritizes simplicity, collaboration, and feedback to deliver high-quality products on time. It emphasizes continuous testing, rapid delivery of working software, and frequent communication between developers, customers, and users. XP highlights the importance of communication and collaboration within the development team and with partners to ensure everyone is working towards the same goal (AgileAlliance, 2022).

### 3.1.2. Selected Methodology



RUP (Rational Unified Process) is a software development methodology that is based on the principles of iterative and incremental development. It was developed by Rational Software (now owned by IBM) and is designed to help organizations manage the complexity of software development projects by providing a structured process and a set of tools and templates. RUP is a process framework that can be customized to meet the specific needs of an organization or project. It is based on the idea of delivering incremental releases of software, with each release building on the previous one and adding additional functionality (Grady Booch, 1999).

Refer to [Appendix – B](#) for the reasons of choosing and phases of RUP Methodology



### 3.1.3. Methodology Section Justification

The justification for the selection of the RUP methodology are as follows:

#### Justification 1: For not choosing Scrum Methodology

*Table 2: Justification 1*

Point	This methodology lacks a clear and well-defined set of requirements, which can make it difficult to effectively plan and execute projects
Business Case	It can be difficult to plan and manage a project utilizing the Scrum approach, which is meant to handle well-defined needs, without a clear and well-defined set of requirements.
Feature	The app must be capable of accommodating new requirements and adapting to changing business needs.
Justification	Scrum methodology is not appropriate for this type of project since it may not be able to handle quickly changing requirements effectively.

#### Justification 2: For not choosing DSDM Methodology

*Table 3: Justification 2*

Point	This methodology is complex to implement, as involves a number of different roles, processes, and techniques.
Business Case	It can be difficult to implement in a project as it requires a number of different roles, processes, and techniques.
Features	The app involves a high level of complexity or rapidly changing requirements.
Justification	DSDM methodology is not appropriate for this project since it may not be handled by a limited people.

**Justification 3: For not choosing Extreme Programming (XP) Methodology***Table 4: Justification 3*

Point	This methodology lacks rapidly changing requirements.
Business Case	It can be difficult to plan and manage a project utilizing the XP methodology, which is meant to small, well-defined projects with simple requirements.
Feature	The app involves a high level of complexity or rapidly changing requirements.
Justification	XP methodology is not appropriate for this project since it may not be handled complexity and quickly changing requirements.

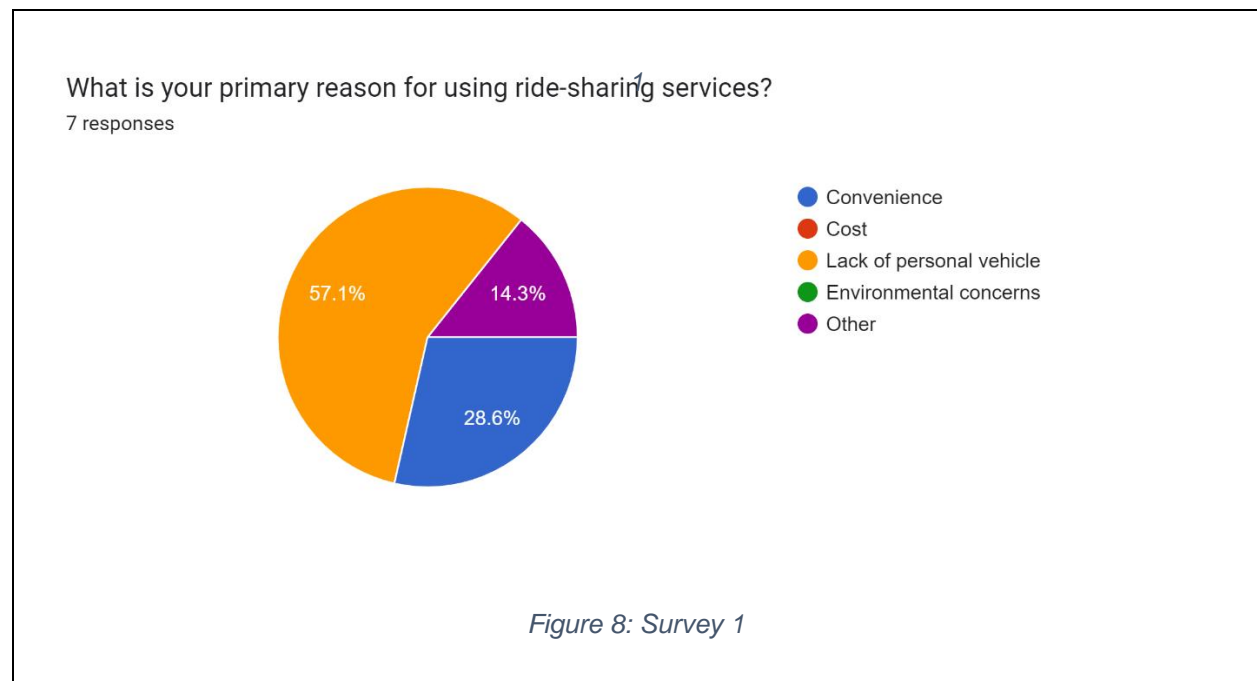
## 4. Work To Date

### 4.1. Requirement Gathering

The main aim of this system is to provide a convenient, affordable and safe ride for people to get from one place to another. This web app generally works by connecting passengers with drivers who are willing to provide a ride. Passengers can request and pay for their ride, allowing them to track their driver in real-time, while at the same time the driver can see the exact location of the passenger and have access to the payment once the ride is complete. Key requirements for a successful ride-sharing app include market research, user experience, reliability, and integration.

#### 4.1.1. Conducted Online Survey

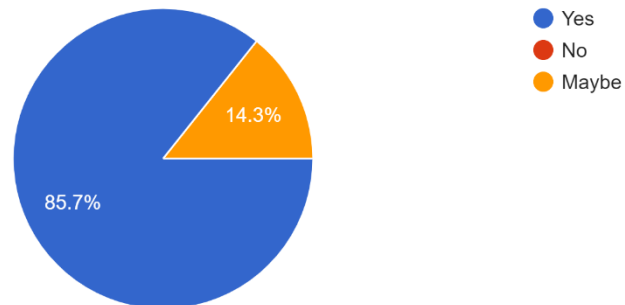
I conducted an online survey form to comprehend and understand the current situation of the ride-sharing web app in Nepal and to collect the feedback from the people to make better decisions and implementation while developing the web app.



Among the 7 participants in the pre-survey, 57.1% said that the main reason for using ride-sharing services is the lack of a personal vehicle, 28.6% cited convenience as the main reason, and 14.3% selected other.

Do you think requesting a vehicle online will save your time?

7 responses

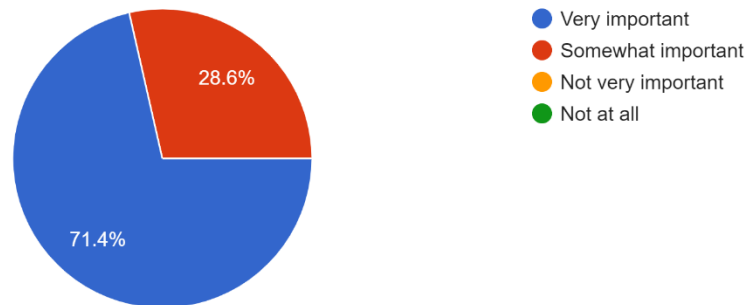


*Figure 9: Survey 2*

In the pre-survey, participants were asked if they believed that requesting a vehicle online would save them time. Of the 7 participants, 85.7% responded 'yes', while 14.3% responded 'maybe'.

How important is it for you to have a choice of vehicle options (e.g. car, bike, tuktuk)?

7 responses

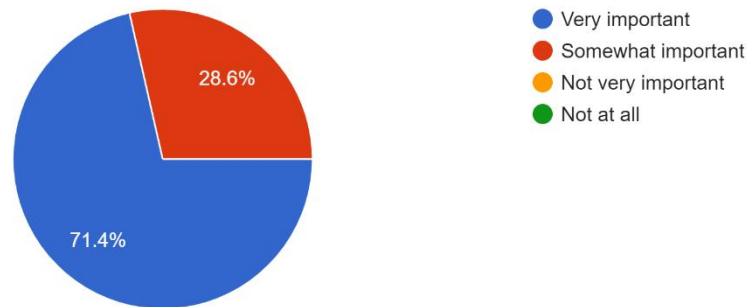


*Figure 10: Survey 3*

In the pre-survey, participants were asked whether they thought it was important to have a choice of vehicle options (e.g. car, bike, tuktuk). Of the 7 participants, 71.4% responded 'very important', while 28.6% responded 'somewhat important'.

How important is it for you to have a rating system for drivers?

7 responses



*Figure 11: Survey 4*

In the pre-survey, participants were asked whether they thought it was important to have a rating system for drivers. Of the 7 participants, 71.4% responded 'very important', while 28.6% responded 'somewhat important'.

What kind of payment method do you prefer?

7 responses

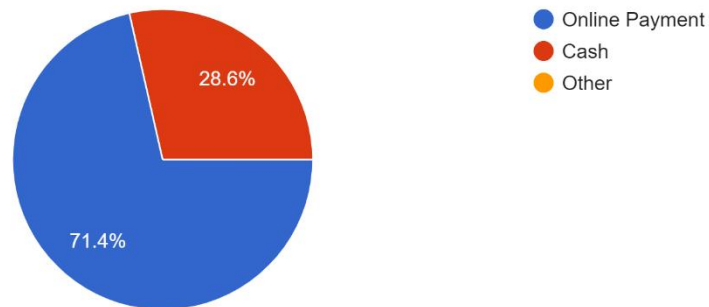


Figure 12: Survey 5

In the pre-survey, participants were asked about their preferred method of payment. Of the 7 participants, 71.4% responded 'online payment', while 28.6% responded 'cash'.

Refer [Appendix – C](#) for other remaining survey

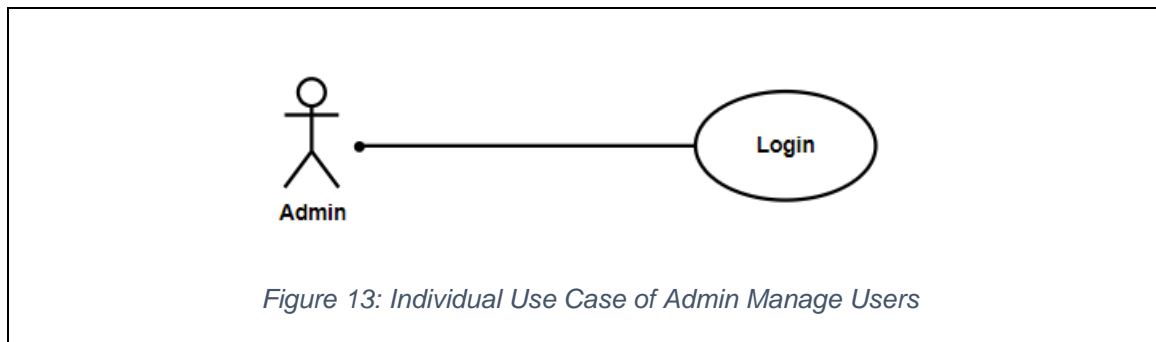
## 4.2. Use Case Diagram

A UML use case diagram is the principal form of system/software specifications for an undeveloped software program. Use cases define the intended behavior (what) rather than the exact technique of achieving it (how). A key concept of use case modelling is that it allows us to build a system from the perspective of the end user. It is an effective technique for explaining system behavior in user terms (Doe, 2019).

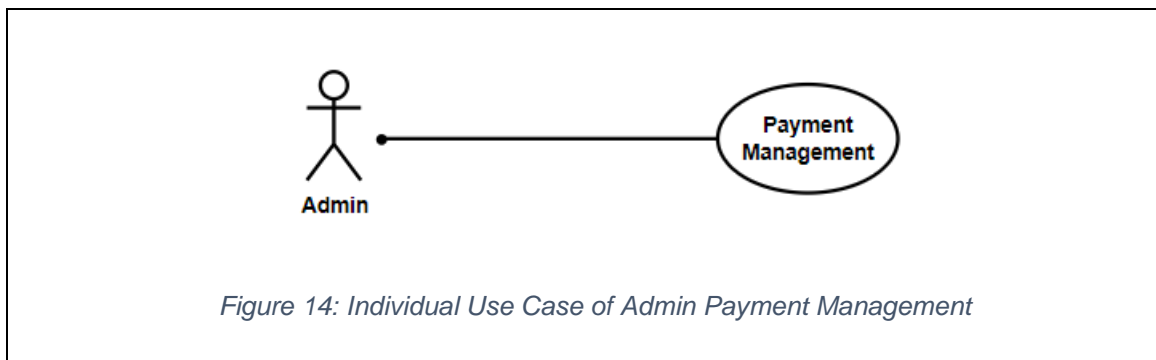
### 4.2.1. Individual Use Case Diagram

#### a) Admin

##### 1) Manage Users



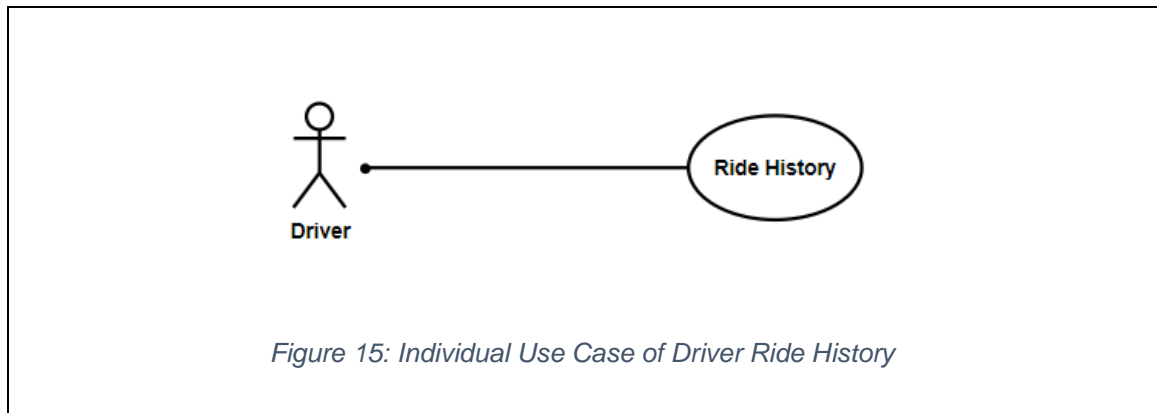
##### 2) Payment Management



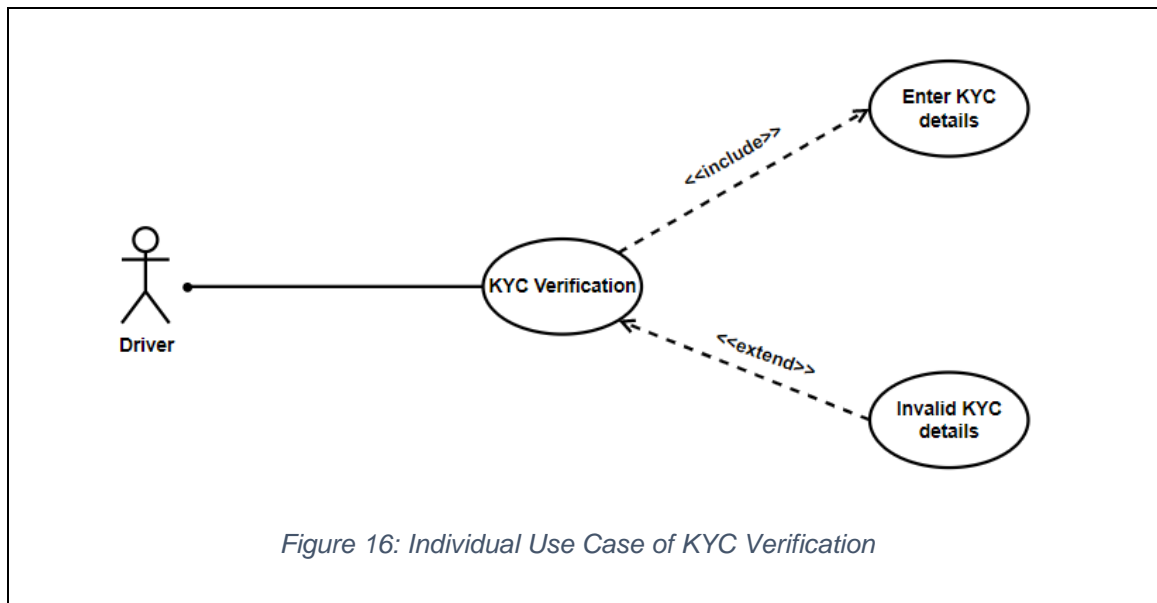


**b) Driver**

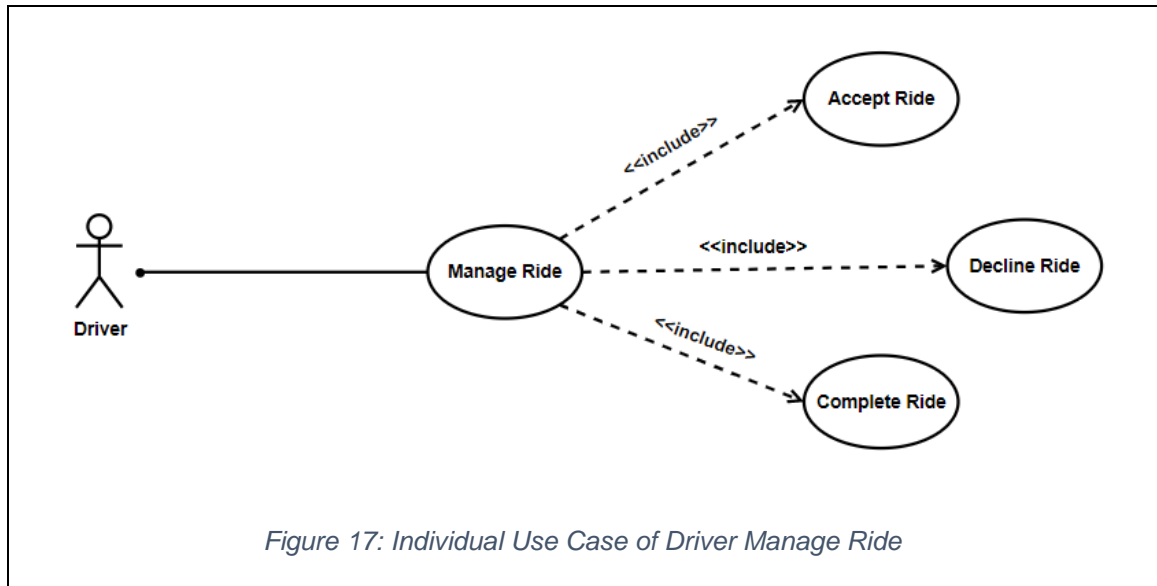
## 1) Ride History

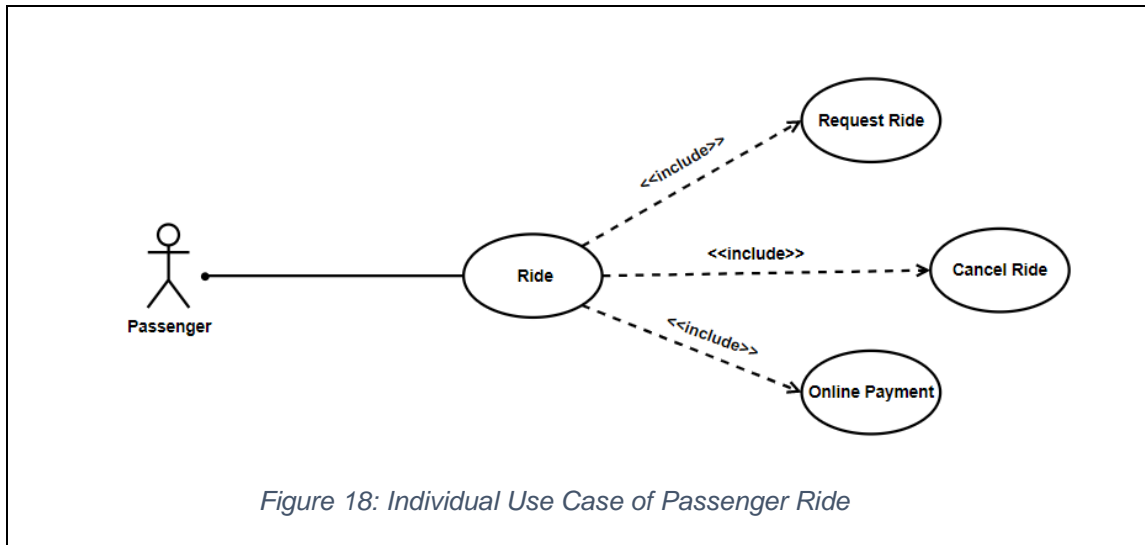
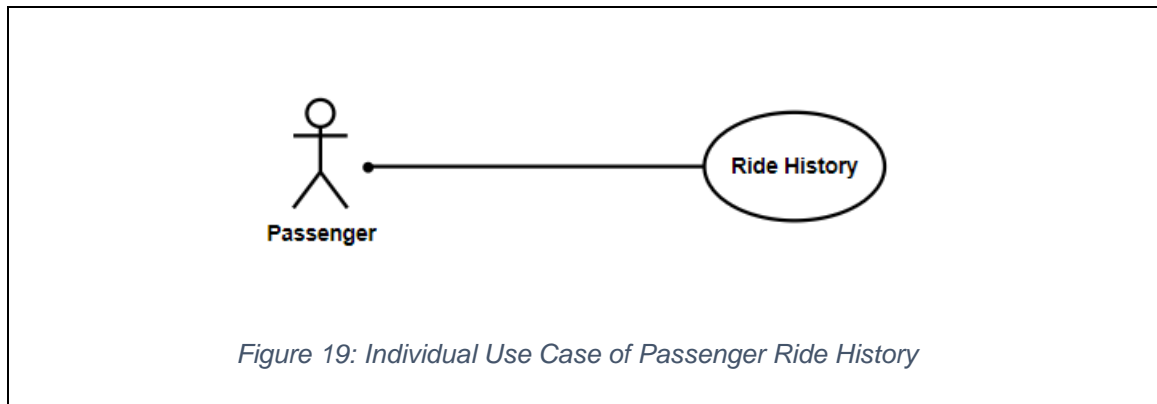


## 2) KYC Verification



## 3) Manage Ride



**c) Passenger****1) Ride****2) Ride History**

Refer to [Appendix – D](#) for remaining Admin, Driver and Passenger Individual Use Case

### 4.2.2. System Use Case Diagram

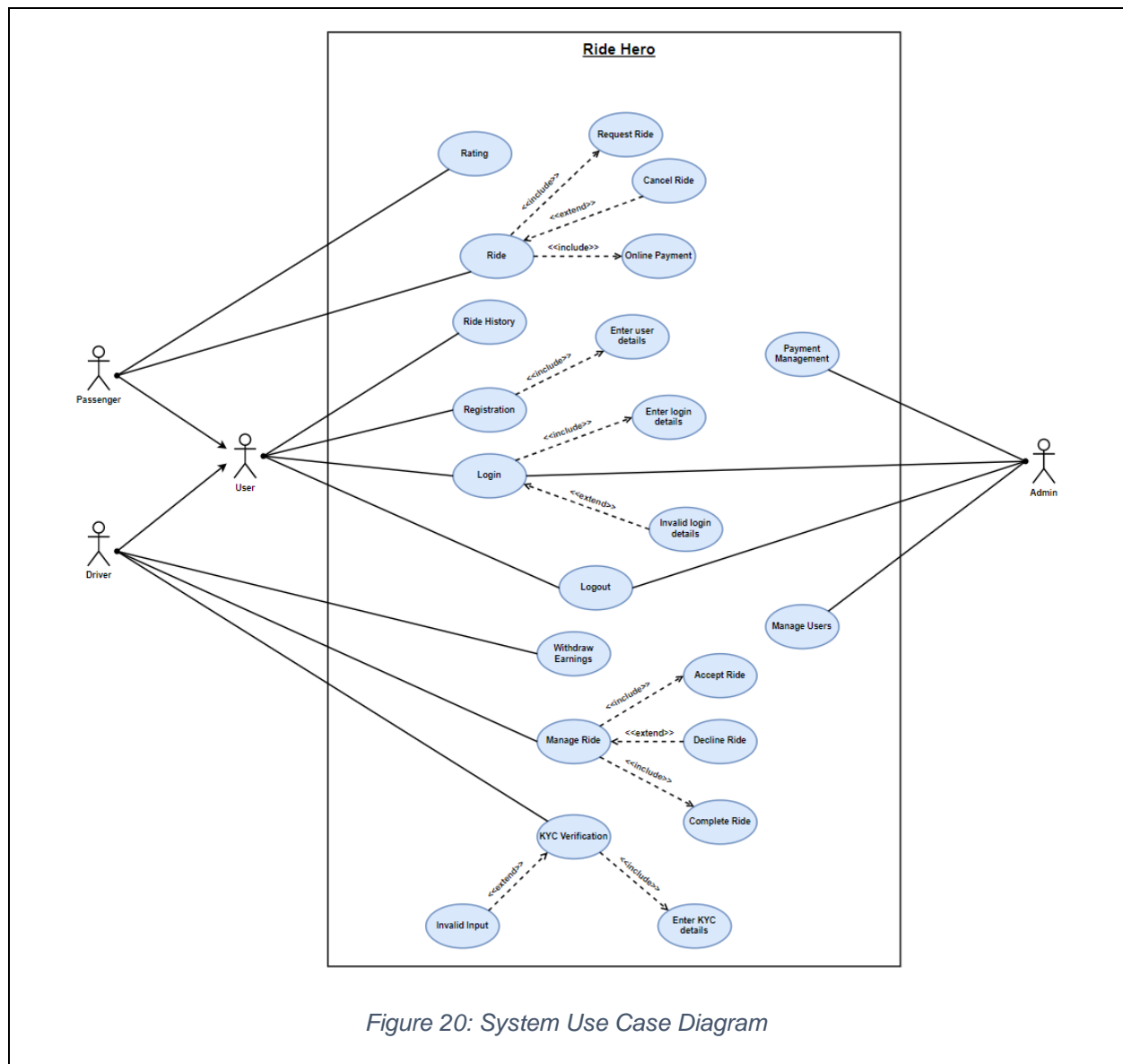


Figure 20: System Use Case Diagram

### 4.3. High Level Use Case

Here is the high level use case of each actors with their description:

*Table 5: High Level Use Case*

S.N.	Use Cases	Actors	Description
1.	Registration	Passenger, Driver	Passenger and driver provides necessary credentials requested by the web app to register into the web app.
2.	Login	Admin, Passenger, Driver	Admin, Passenger, and Driver must login with their registered email address and password.
3.	Logout	Admin, Passenger, Driver	Admin, Passenger, and Driver can logout from the web app.
4.	Manage Users (Passengers and Drivers)	Admin	Admins can manage passenger and driver accounts, including creating and deleting accounts, reviewing user information, and resetting passwords.
5.	Payment Management	Admin	Admins can view and manage the overall payment transaction.
6.	Ride (Accept, Cancel and Online Payment)	Passenger	Passengers can enter their pickup and drop-off locations, select a type of vehicle, and request a ride through the web app and can pay the fare digitally.
7.	Rating	Passenger	Passenger can rate the driver after the ride is completed.

<b>8.</b>	Ride History	Passenger, Driver	Passenger and Driver both can view their ride history.
<b>9.</b>	Withdraw Earnings	Driver	Driver can view and withdraw their earnings.
<b>10.</b>	Manage Ride (Accept, Decline, and Complete)	Driver	Driver can accept, decline or complete a ride.
<b>11.</b>	KYC Verification	Driver	Driver must provide the KYC details to be a verified driver.

## 4.4. Activity Diagram

### 4.4.1. Passenger Activity Diagram

#### a) Passenger Ride Request Activity Diagram

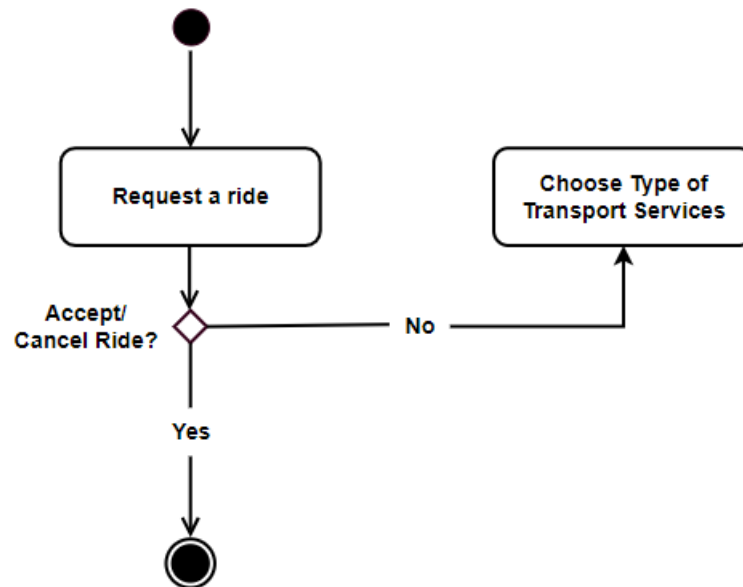
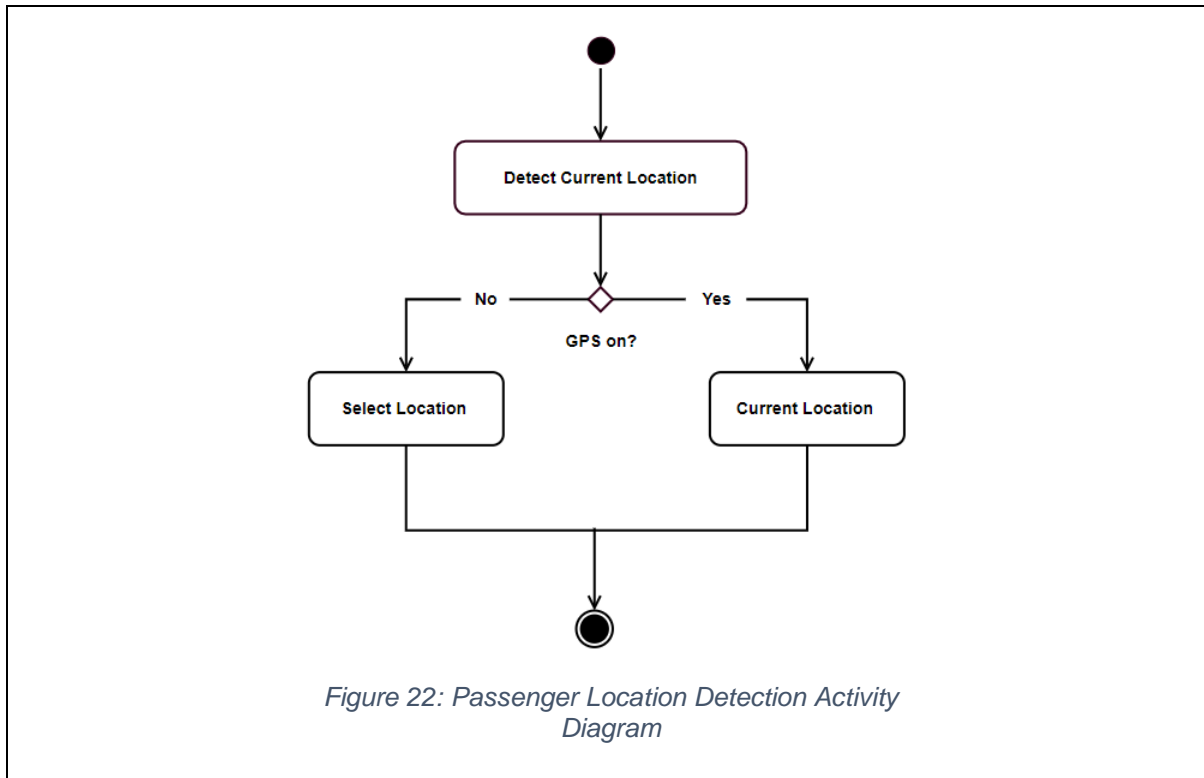


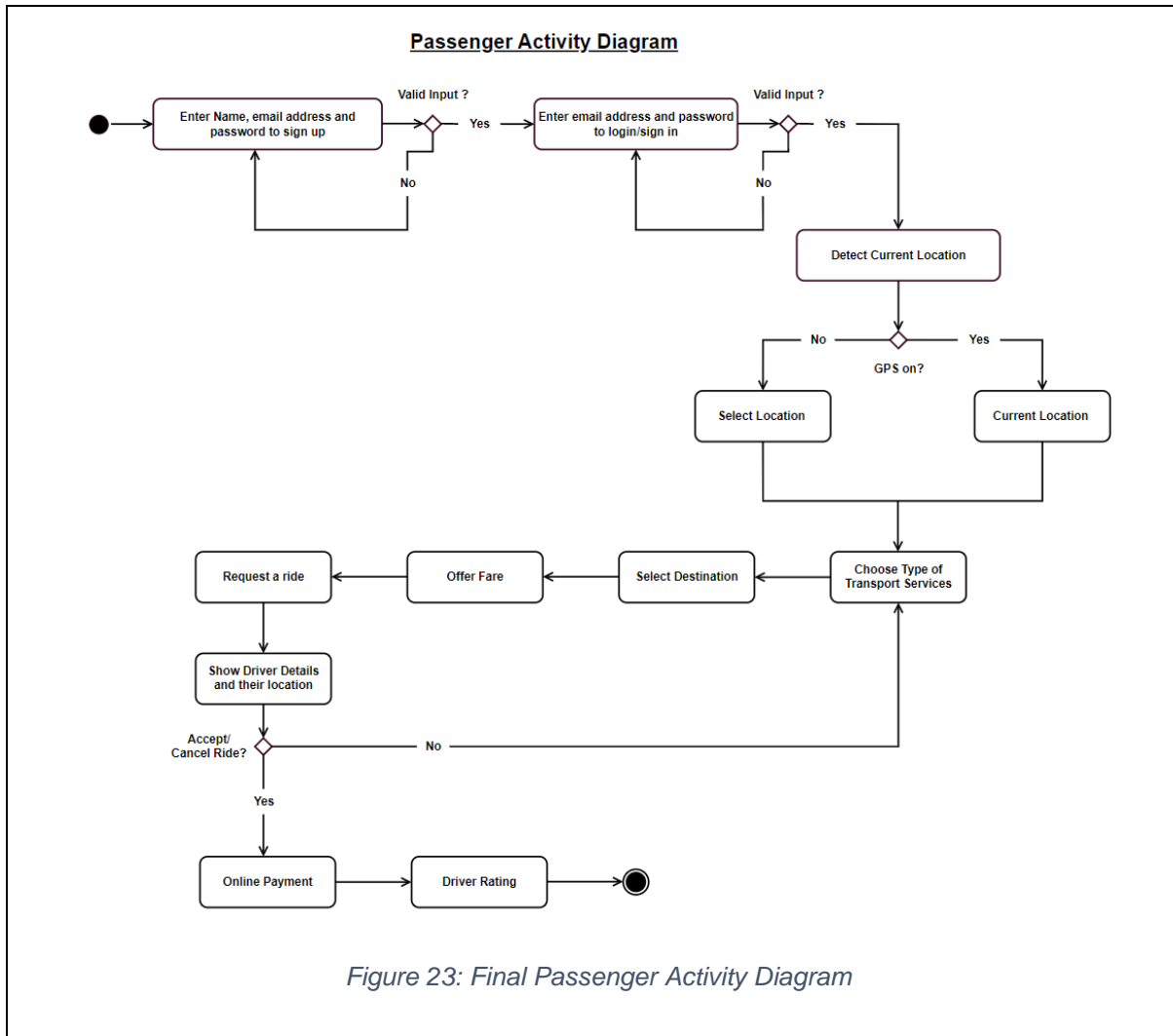
Figure 21: Passenger Ride Request Activity Diagram

## b) Passenger Location Detection Activity Diagram





## c) Final Passenger Activity Diagram



#### 4.4.2. Driver Activity Diagram

##### a) Driver KYC Verification Activity Diagram

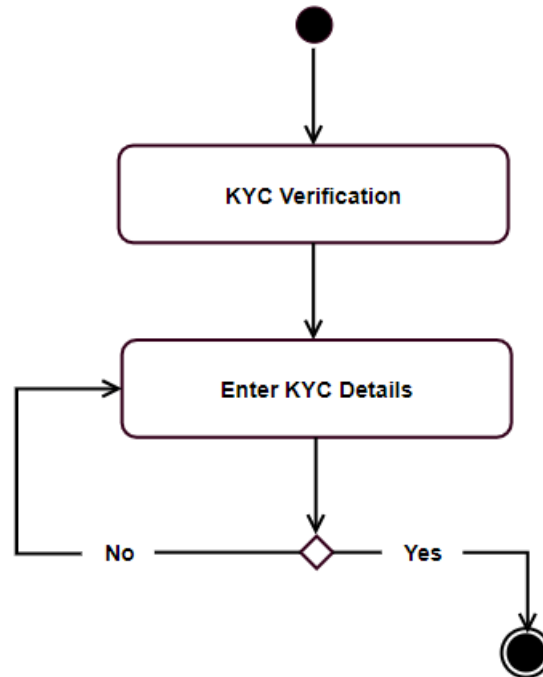
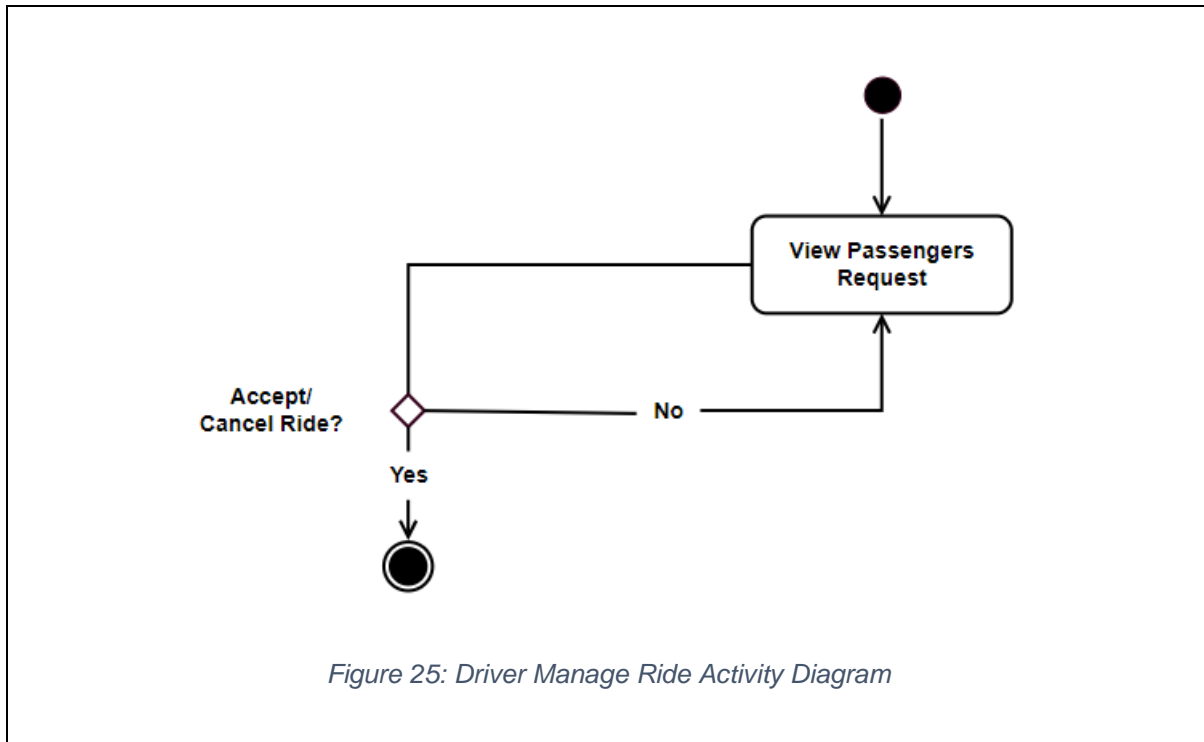
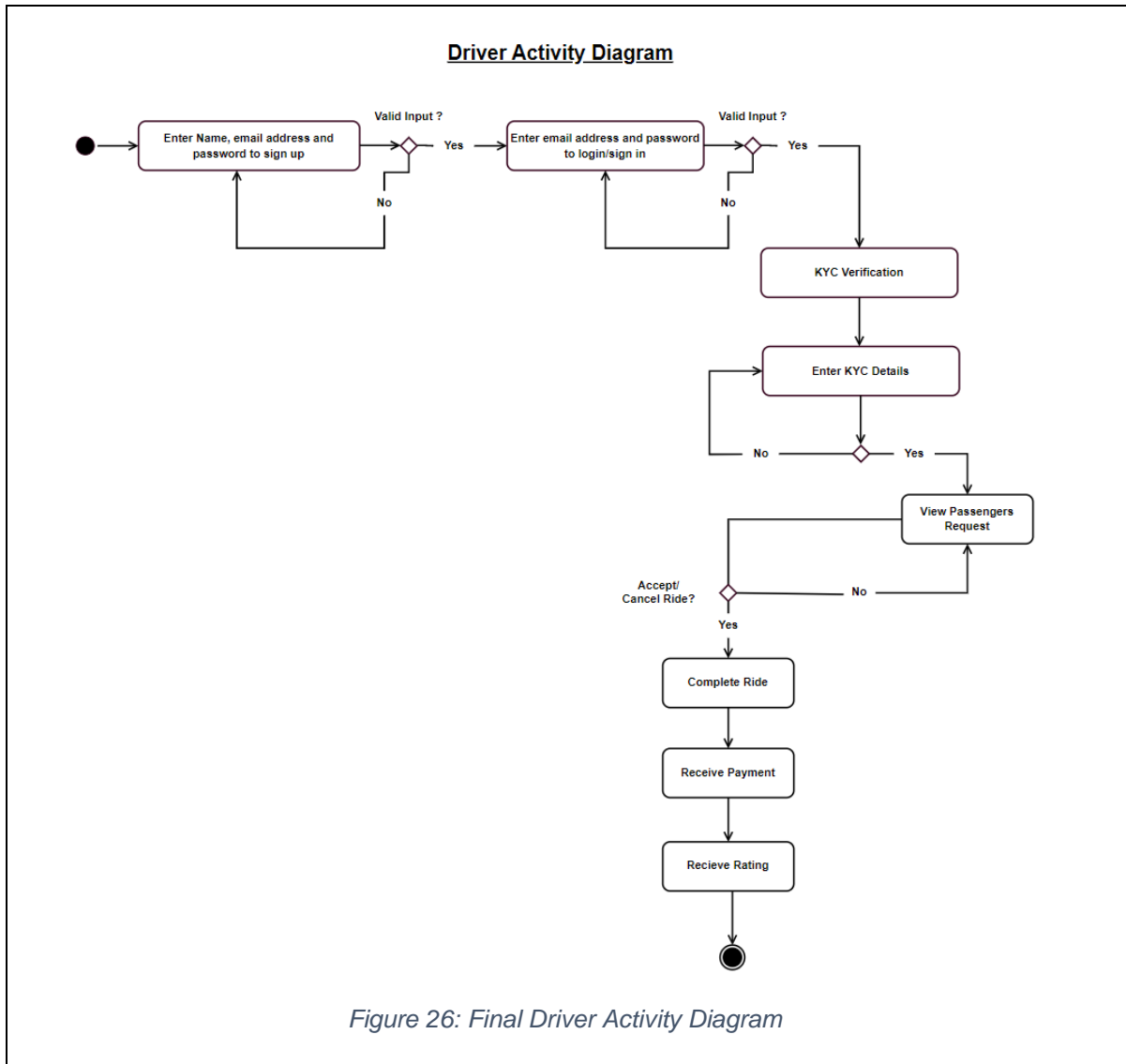


Figure 24: Driver KYC Verification Activity Diagram

## b) Driver Manage Ride Activity Diagram

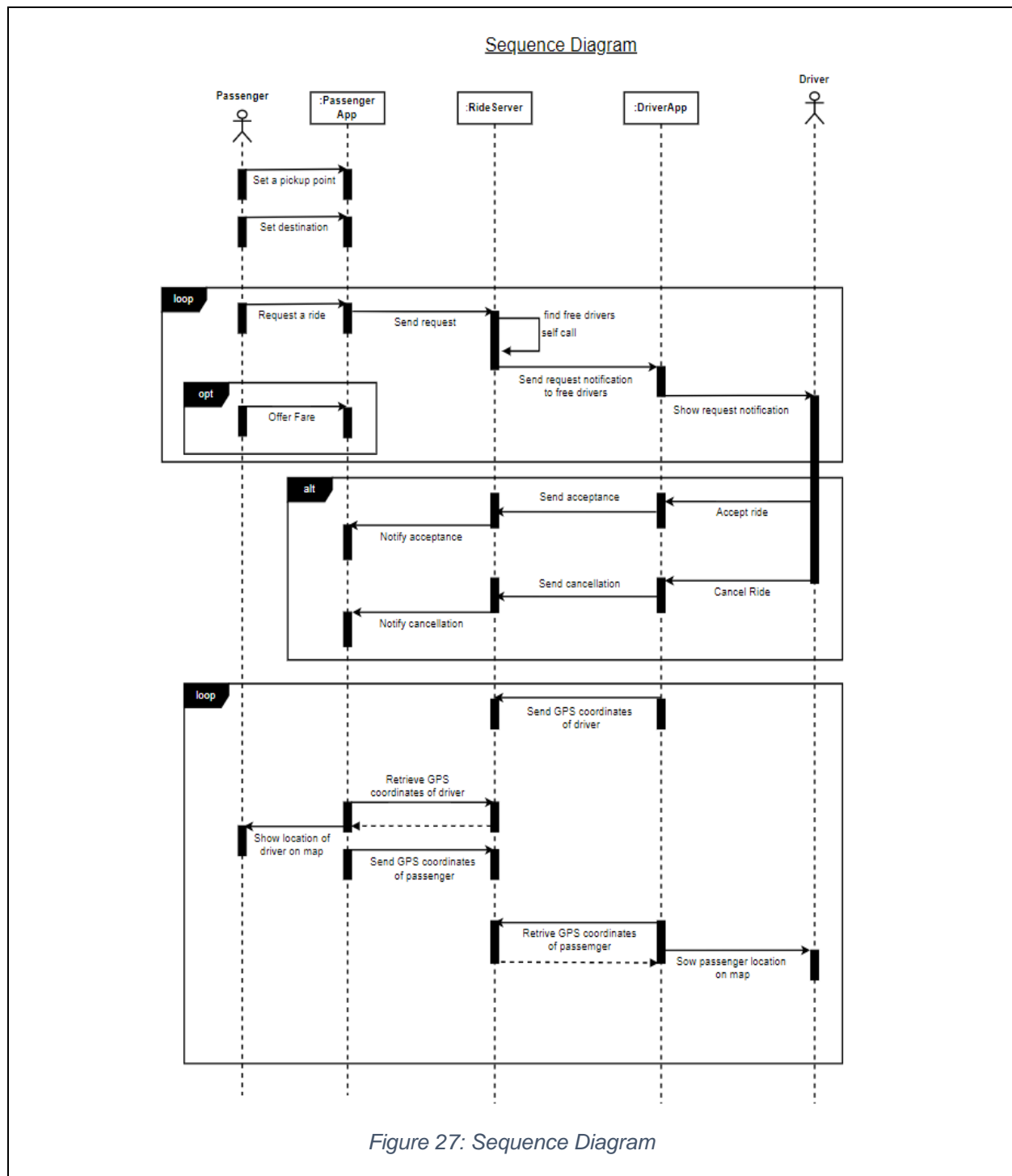


## c) Final Driver Activity Diagram



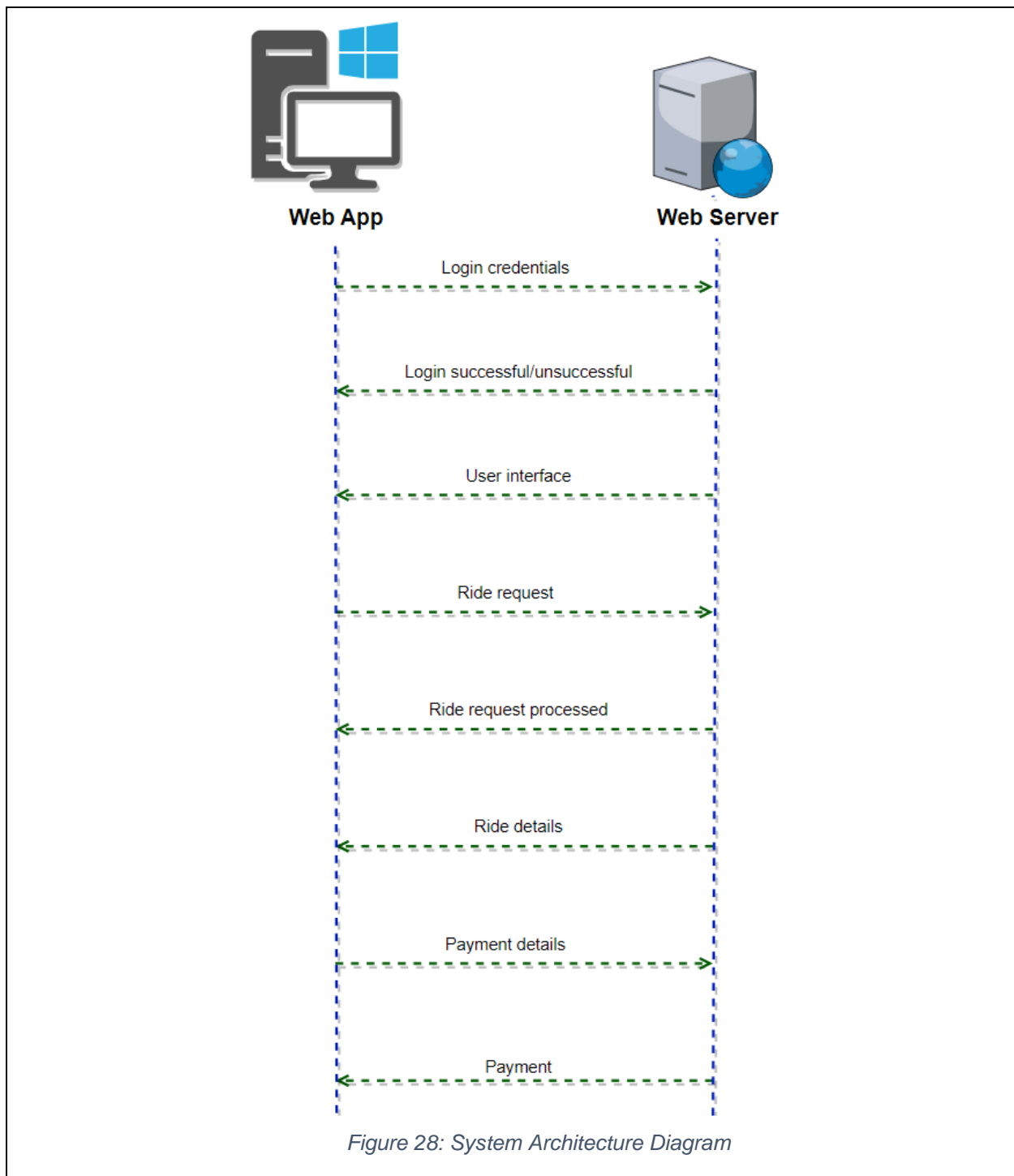
Refer to [Appendix – E](#) for more activity diagram of Passenger and Driver

## 4.5. Sequence Diagram



Refer to [Appendix – F](#) for more sequence diagrams

## 4.6. System Architecture Diagram



Refer to [Appendix – G](#) for details of system architecture diagram

## 4.7. Class Diagram

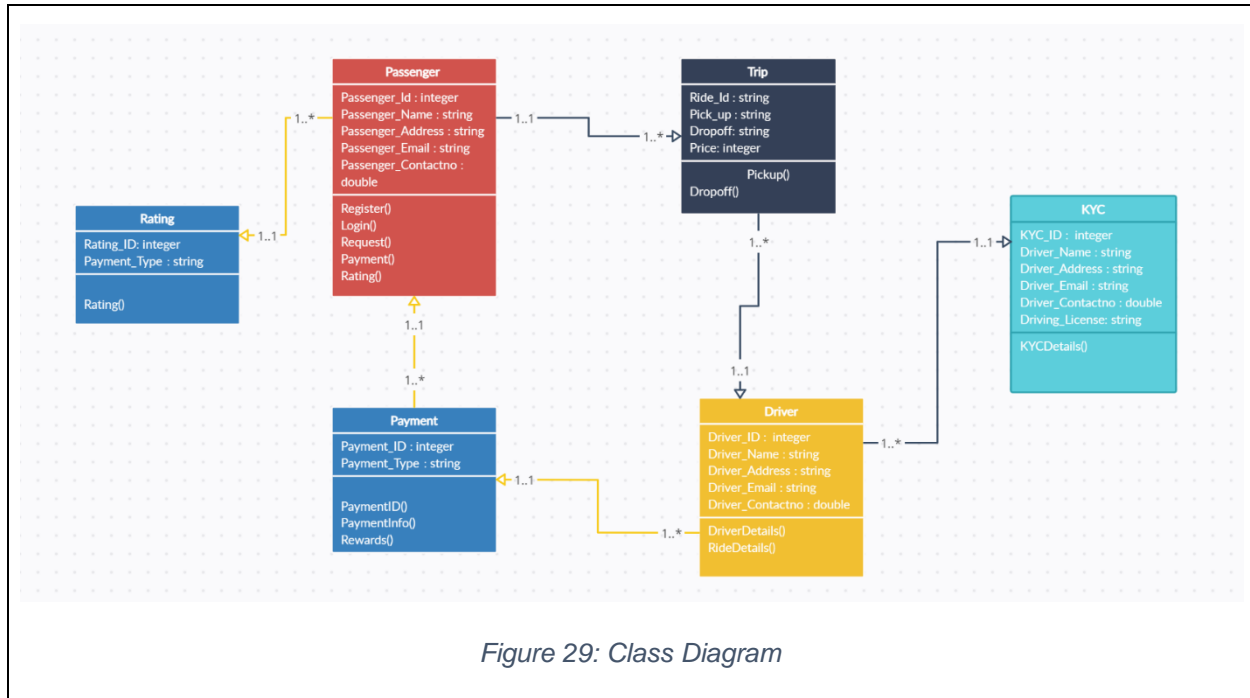
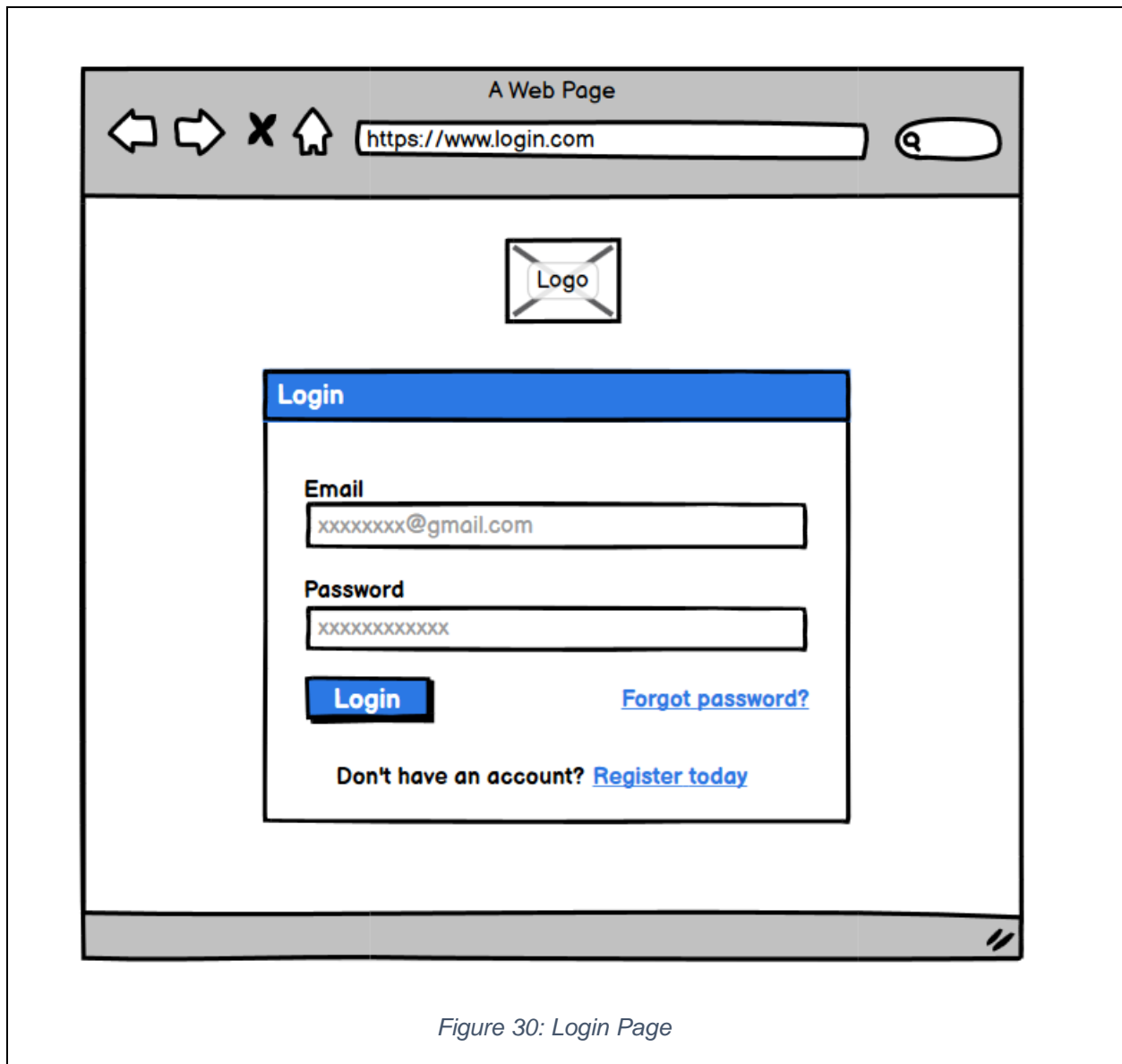


Figure 29: Class Diagram

Refer to [Appendix – H](#) for description of class diagram

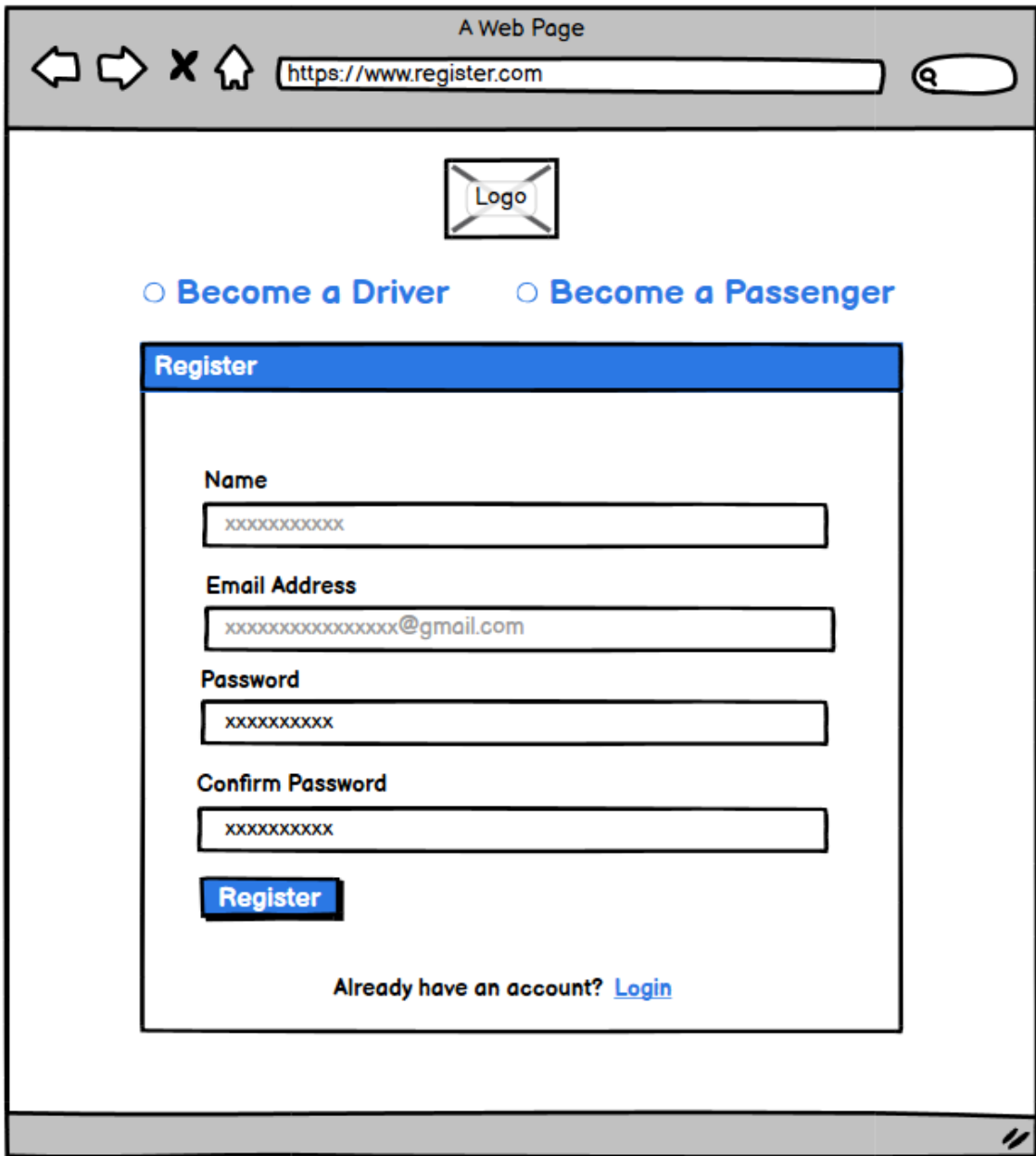
## 4.8. Wireframes

### 4.8.1. Login Page





#### 4.8.2. Register Page



A Web Page

https://www.register.com

Logo

☐ Become a Driver ☐ Become a Passenger

**Register**

Name  
xxxxxxxxxx

Email Address  
xxxxxxxxxxxxxxxxxx@gmail.com

Password  
xxxxxxxxxx

Confirm Password  
xxxxxxxxxx

**Register**

Already have an account? [Login](#)

Figure 31: Register Page

## 4.8.3. Home Page

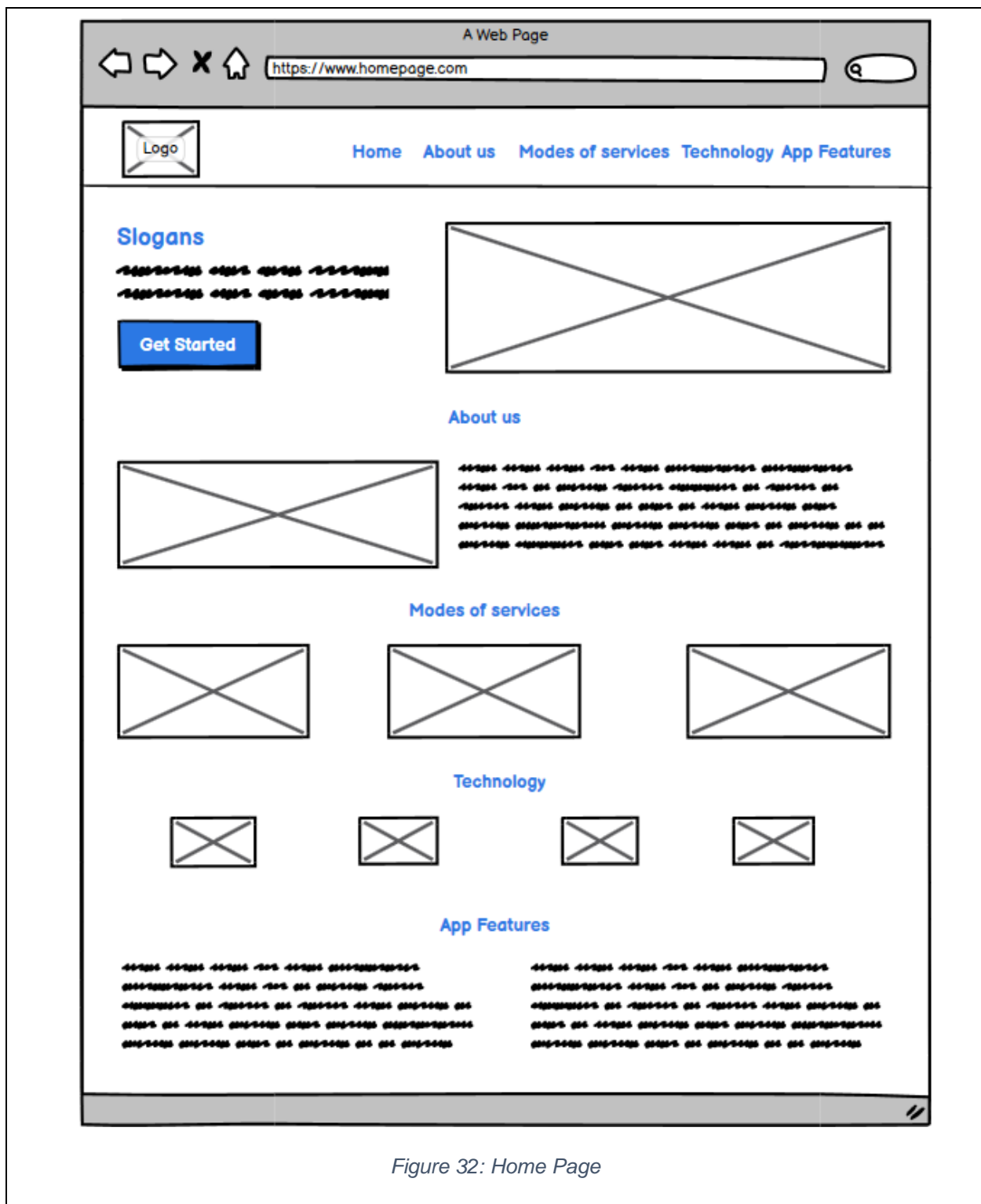


Figure 32: Home Page

#### 4.8.4. Passenger Interface Page

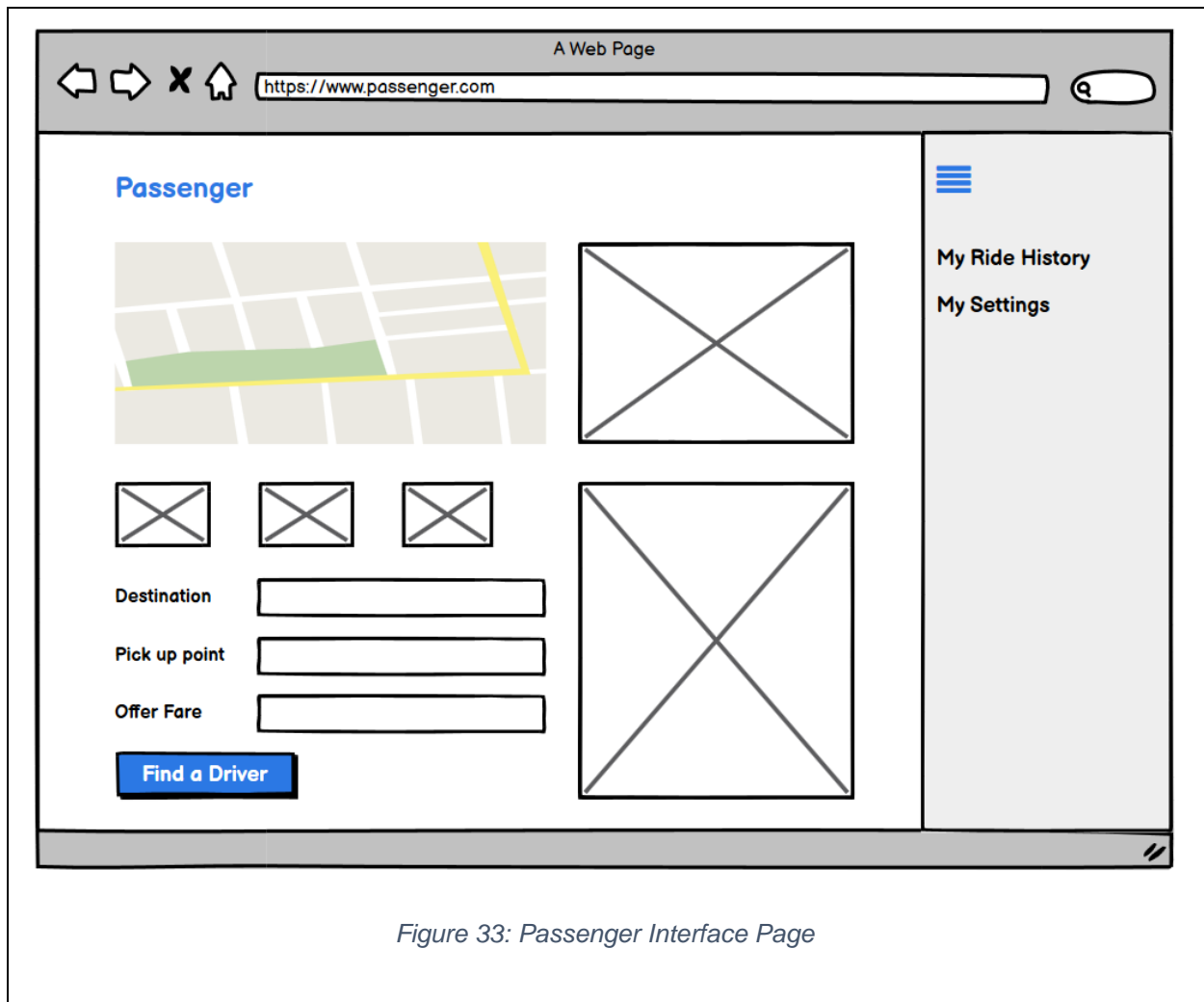


Figure 33: Passenger Interface Page

#### 4.8.5. Driver Interface Page

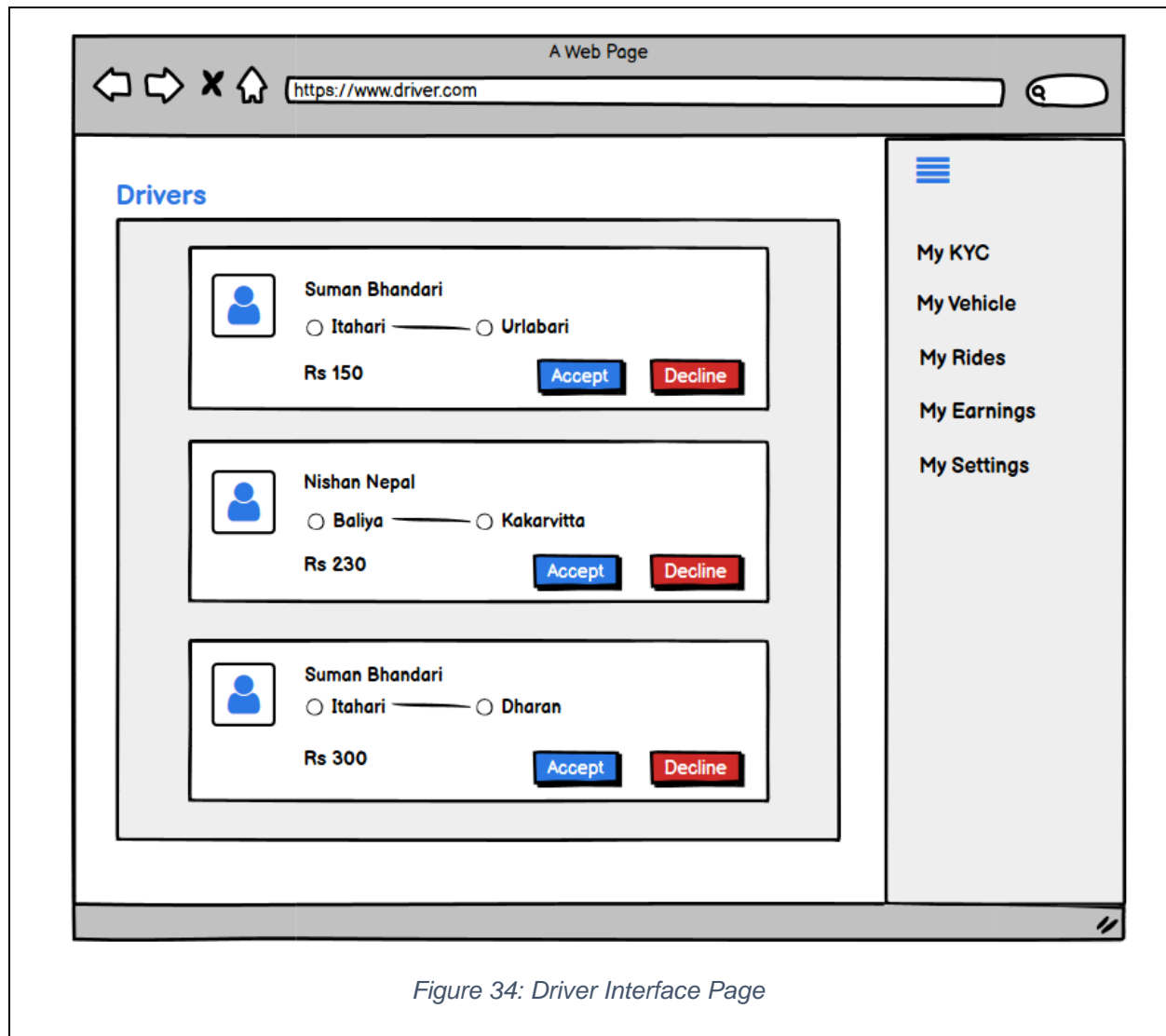


Figure 34: Driver Interface Page

#### 4.8.6. Admin Dashboard

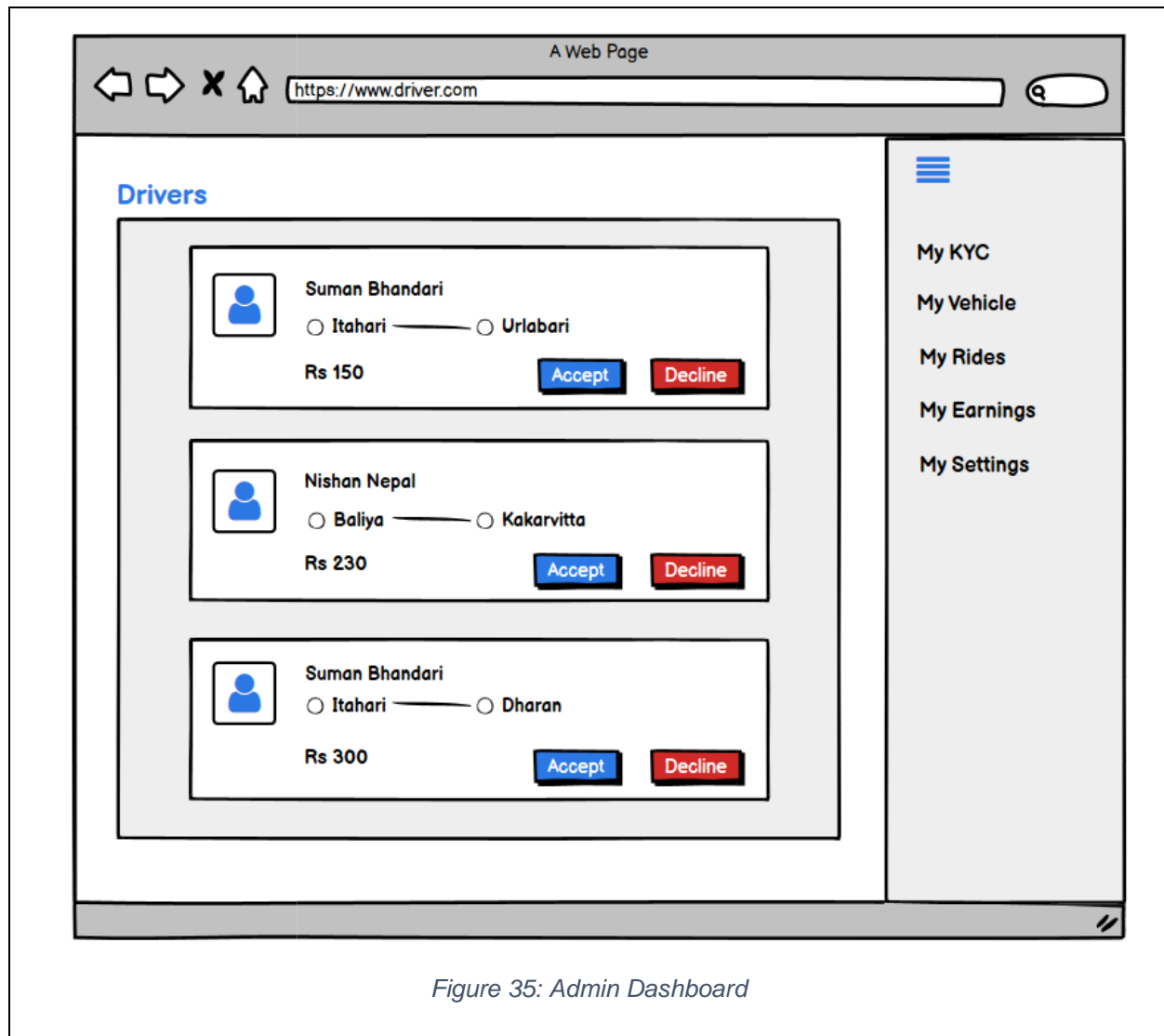
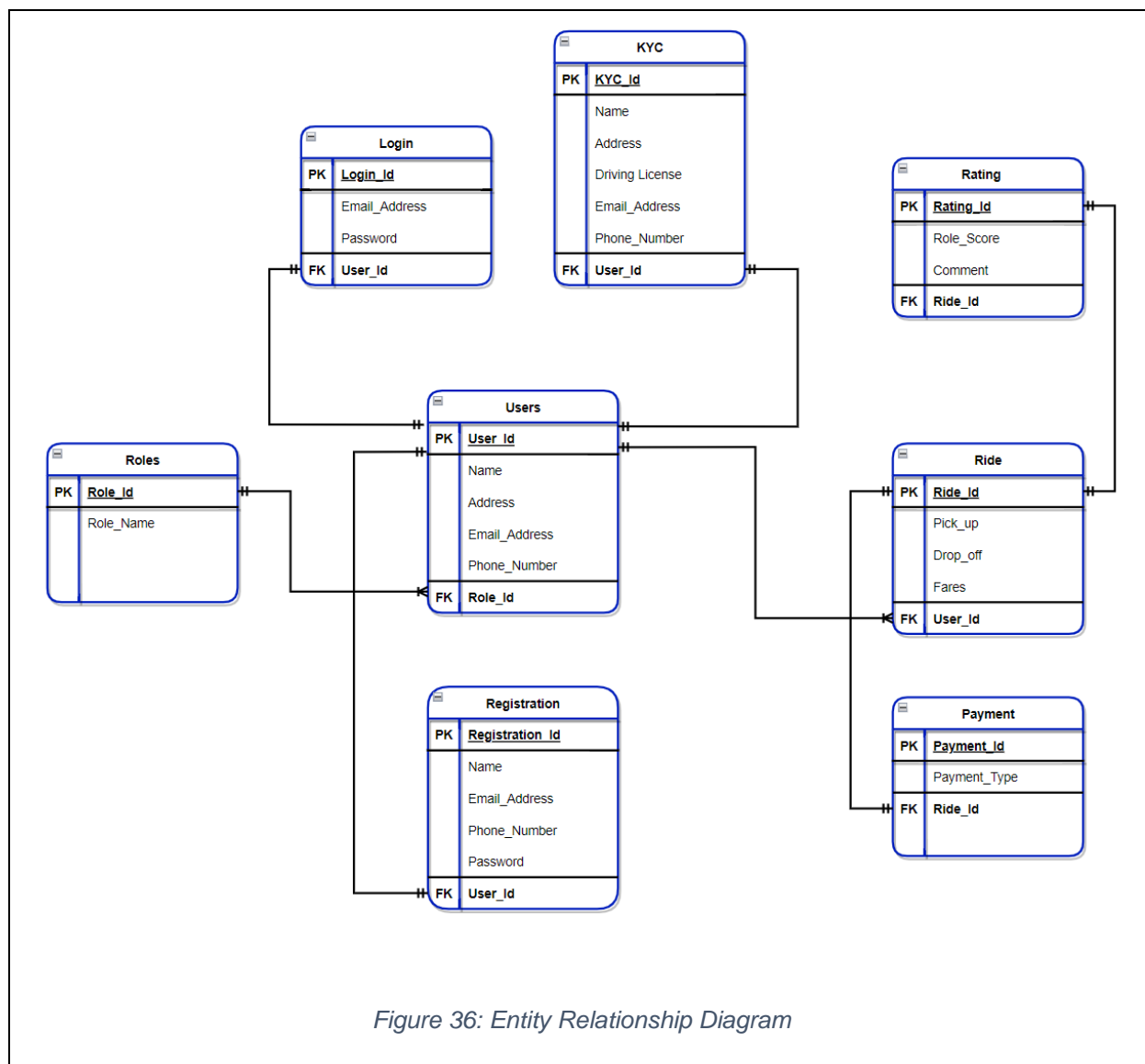


Figure 35: Admin Dashboard

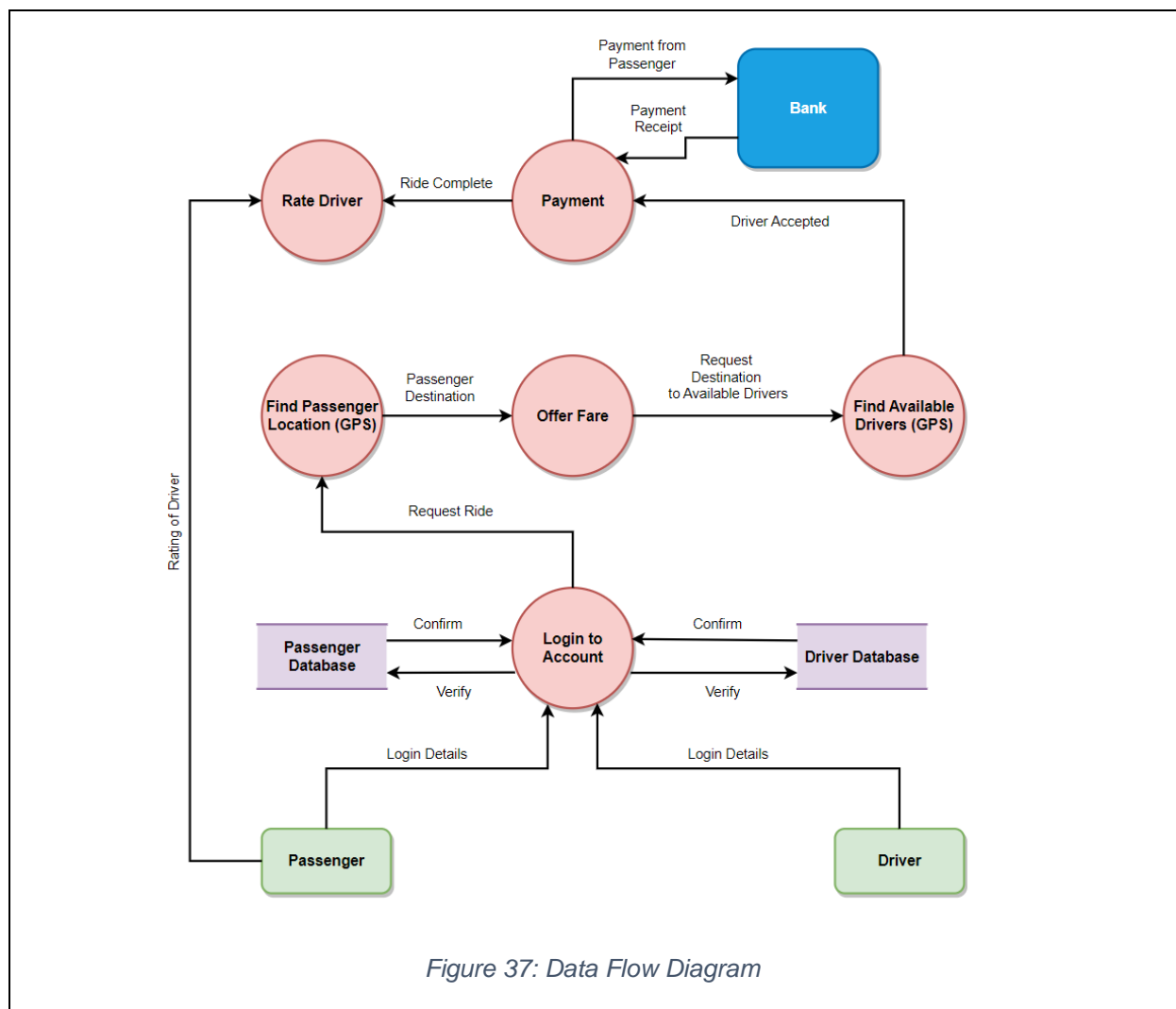
Refer to [Appendix – I](#) for more wireframes

## 4.9. Entity Relationship Diagram



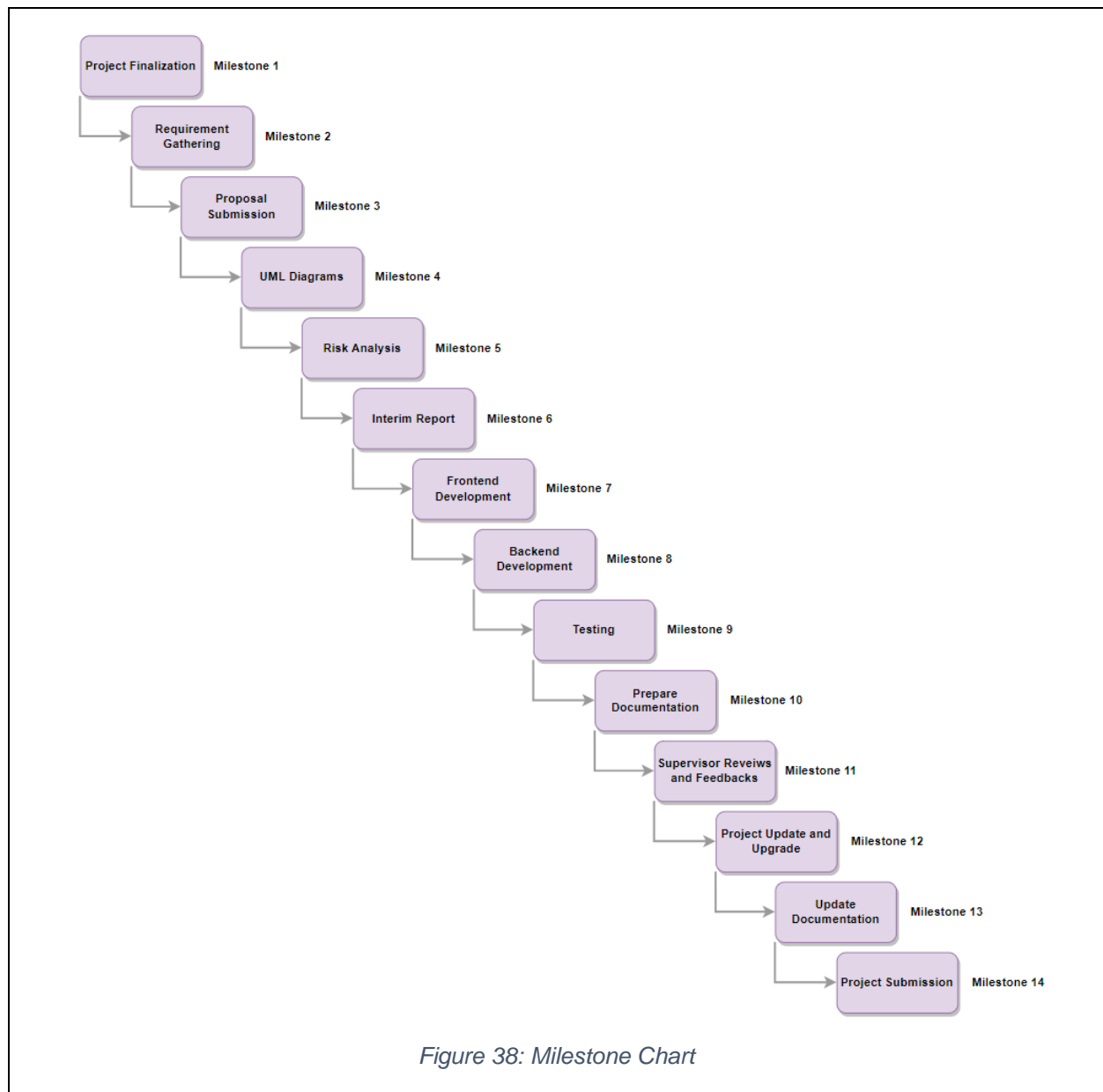
Refer to [Appendix – J](#) for more details of ERD

#### 4.10. Data Flow Diagram (DFD)



Refer to [Appendix – K](#) for more details on DFD

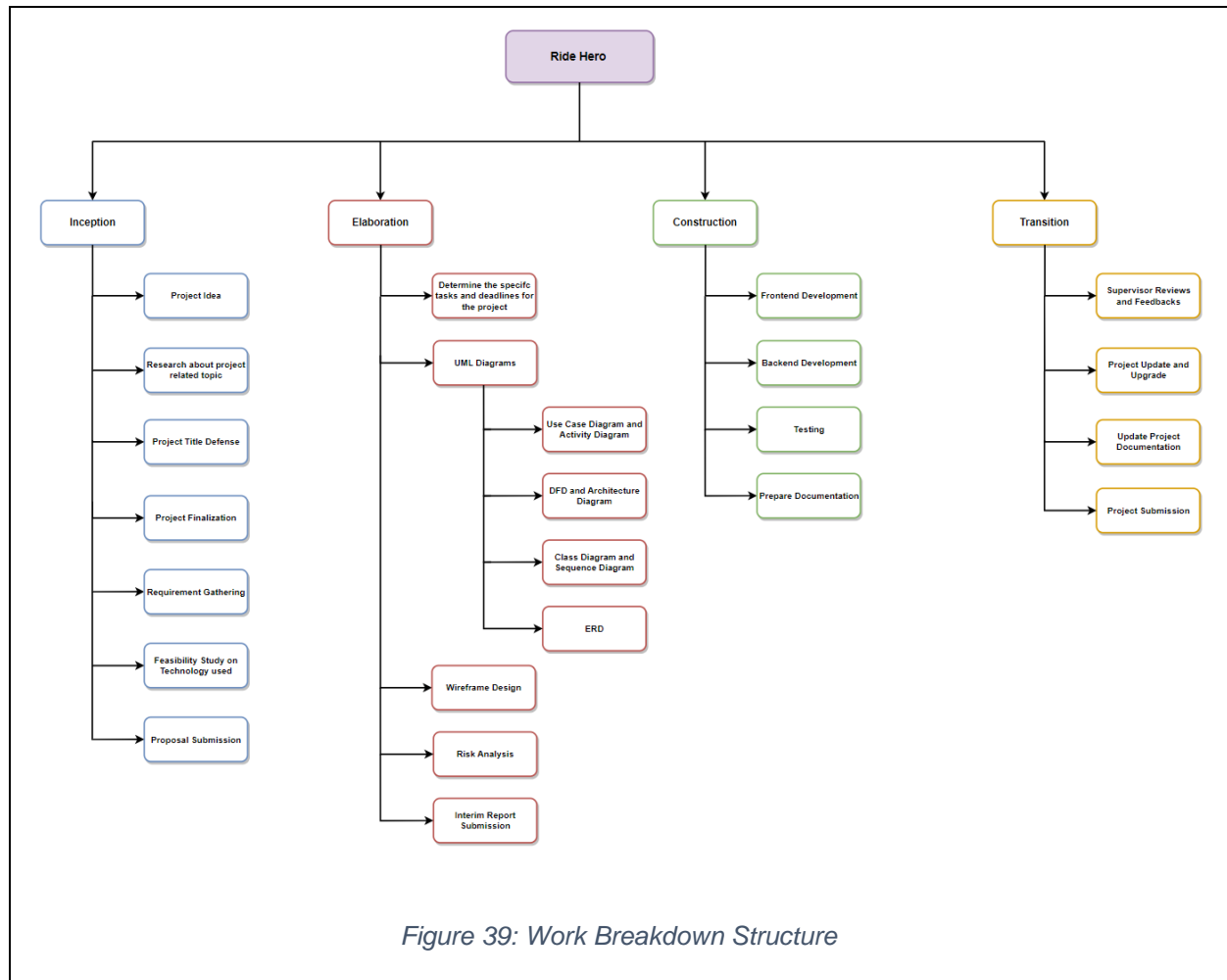
## 4.11. Milestone Chart



Refer to [Appendix – L](#) for previous milestone chart



## 4.12. Work Breakdown Structure (WBS)



Refer to [Appendix – M](#) for previous WBS

### 4.13. Gantt Chart

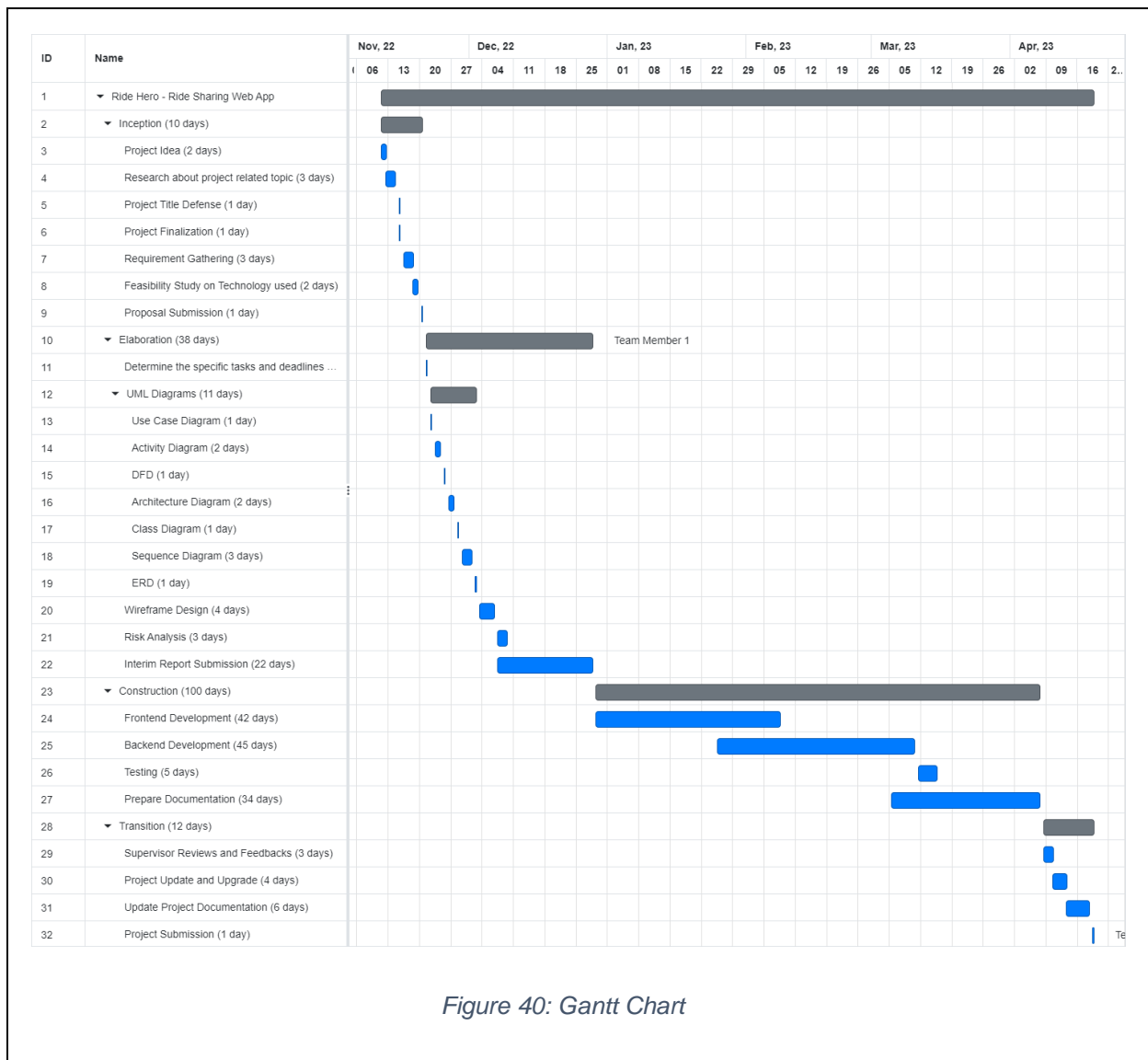


Figure 40: Gantt Chart

Refer to [Appendix – N](#) for previous Gantt Chart

#### 4.14. Analysis of Progress

In this progress review, I have analyzed the current state of my project and found that the inception and elaboration phases have been completed successfully. These phases included the overall design of the application, as well as some additional features, and the creation of a time estimation Gantt chart and a Work Breakdown Structure (WBS) to plan and sequence the work efficiently. I am now in the construction and transition phases, which involve building and testing the application, including implementing the code, integrating various components, and conducting thorough testing to ensure that the application meets the requirements and is of high quality. I am making good progress on the project and am confident that I will be able to complete it on time, although I will continue to monitor my progress closely and make any necessary adjustments to stay on track,

Refer to [Appendix – O](#) for SRS

#### 4.14.1. Progress Table

In this section, the progress of individual task of the project is tracked and also determine the areas of tasks that need to be completed. The following table shows the progress of the task:

*Table 6: Progress Table*

S.N.	Tasks	Progress	Progress %
1.	Research about project topic	Completed	100%
2.	Requirement Gathering	Completed	100%
3.	Feasibility Study on Technology used	Completed	100%
4.	Proposal Submission	Completed	100%
5.	Use Case Diagram	Completed	100%
6.	Activity Diagram	Completed	100%
7.	Architecture Diagram	Completed	100%
8.	Sequence Diagram	Completed	100%
9.	Class Diagram	Completed	100%
10.	Data Flow Diagram (DFD)	Completed	100%
11.	Entity Relationship Diagram (ERD)	Completed	100%
12.	Wireframe Design	Completed	100%
13.	Interim Report	Completed	100%

**4.14.2. Progress Review**

The progress of the project so far has been very good, with important tasks such as research about the project topic, requirement gathering, feasibility study on the technology used, and proposal submission all completed successfully. The use case diagram, activity diagram, architecture diagram, sequence diagram, class diagram, data flow diagram (DFD), and entity relationship diagram (ERD) as well as wireframes have also been completed, providing a clear understanding of the project's requirements and design.

Overall, the project is on track and making good progress. However, it will be important to continue to monitor progress and make any necessary adjustments to stay on track and ensure the success of the project.

#### 4.14.3. Project Timeline

The following progress timeline table shows the tasks that have been completed and the tasks that are currently in progress:

*Table 7: Progress Table Timeline*

S.N.	Tasks	Progress	Progress %
1.	Research about project topic	Completed	100%
2.	Requirement Gathering	Completed	100%
3.	Feasibility Study on Technology used	Completed	100%
4.	Proposal Submission	Completed	100%
5.	Use Case Diagram	Completed	100%
6.	Activity Diagram	Completed	100%
7.	Architecture Diagram	Completed	100%
8.	Sequence Diagram	Completed	100%
9.	Class Diagram	Completed	100%
10.	Data Flow Diagram (DFD)	Completed	100%
11.	Entity Relationship Diagram (ERD)	Completed	100%
12.	Wireframe Design	Completed	100%
13.	Interim Report	Completed	100%
14.	Frontend Development	Incomplete	0%
15.	Backend Development	Incomplete	0%
16.	Testing	Incomplete	0%
17.	Review	Incomplete	0%
18.	Documentation	Incomplete	0%

**4.14.4. Action Plan**

An action plan is a detailed list of the tasks and activities that need to be completed in order to achieve goals of the project. It typically includes a timeline for when each task should be completed, a description of the tasks and the resources that are required to complete it. The task of the project is followed according to a new Gantt chart created for the interim report as reference for the tasks that must be accomplished and the deadlines for each activity. The action plan includes a description of each activity, the resources required to perform it, and the work's expected length. The Gantt chart can be used to track each task's progress and highlight areas where work is lagging. It is also a good idea to review and update the action plan and Gantt chart on a frequent basis to make sure that they remain precise and current as the project moves forward. I can effectively handle the project by following the action plan and referring to the Gantt chart.

## 5. Future Work

There is still work to be done on my project in order to make it a success. This may involve development, testing, implementation, and documentation. These tasks will help to ensure that my project is fully functional, meets the needs of the users, and is well-documented for future reference.

*Table 8: Future Work Table*

Work	Process	Description
<b>Construction</b>	Frontend Development	This involves designing the user interface of a web app using HTML, CSS, JS and React JS.
	Backend Development	This involves creating and maintaining the database in XAMPP server using MYSQL with the help of Laravel.
	Testing	This involves verifying functionality and user requirements of a website or app through test case development, defect identification and resolution, and documentation maintenance.
	Prepare Documentation	This involves creating project documentation that describe the purpose, functionality, and usage of a web app.
<b>Transition</b>	Supervisor Reviews & Feedbacks	This involves obtaining input and guidance from a supervisor on project progress, incorporating feedback, and regularly communicating progress and issues.
	Project Updates & Upgrades	This involves implementing changes and improvements to a project, including new features and functionality, defect resolution, and maintenance.
	Update Documentation	This involves revising and expanding documentation for a website or app, including maintenance, improvement, and ensuring accuracy.



In conclusion, it is important to prioritize these tasks and create a plan for completing them efficiently in order to bring your project to a successful completion. By focusing on these key areas and following a structured plan, you can ensure that your project meets all of its goals and is a success.

## 6. Conclusion

In conclusion, the ride-sharing web app project has completed the elaboration phase, which means that the project goals and scope have been defined and the necessary resources have been identified. The remaining phases of the project, including development, testing, implementation, and documentation, will be critical for ensuring the success of the app. It is important to allocate sufficient time and resources for each of these phases in order to bring the app to completion and ensure that it meets the needs of the end users. By following a structured development process and conducting thorough testing, you can help ensure that the app is of high quality and performs as expected. The implementation and documentation phases will also be important for ensuring that the app is deployed and supported effectively. By following best practices and staying on track with the project plan, you can help ensure the success of the ride-sharing web app project.

## 7. References

AgileAlliance, 2022. *Extreme Programming (XP)*. [Online] Available at:

[https://www.agilealliance.org/glossary/xp/#q=~\(infinite~false~filters~\(postType~\(~'post~'aa book~'aa event session~'aa experience report~'aa glossary~'aa research paper~'aa video\)\)~tags~\(~'xp\)\)~searchTerm~'~sort~false~sortDirection~'asc~page~1\)](https://www.agilealliance.org/glossary/xp/#q=~(infinite~false~filters~(postType~(~'post~'aa book~'aa event session~'aa experience report~'aa glossary~'aa research paper~'aa video))~tags~(~'xp))~searchTerm~'~sort~false~sortDirection~'asc~page~1))

[Accessed 16 December 2022].

Bentley, C., 2010. The DSDM Agile Project Management Handbook: For DSDM Atern Projects. In: s.l.:TSO (The Stationery Office).

BLYSTONE, D., 2022. *The Story of Uber*. [Online] Available at: <https://www.investopedia.com/articles/personal-finance/111015/story-uber.asp>

[Accessed 16 December 2022].

DAVIS, L., 2022. *Lyft vs. Uber: What's the Difference?*. [Online] Available at: <https://www.investopedia.com/articles/personal-finance/010715/key-differences-between-uber-and-lyft.asp>

[Accessed 16 December 2022].

Doe, J., 2018. Designing User-Friendly Wireframes. *Journal of User Experience Design*, 7(3), pp. 56-65.

Doe, J., 2019. Designing Efficient Use Case Diagrams. *Best Practices in Software Design*, Pearson(Jane Smith), pp. 82-94.

engineeringforchange, 2022. *Ola Cabs*. [Online] Available at: <https://www.engineeringforchange.org/solutions/product/ola-cabs/>

[Accessed 16 December 2022].

Grady Booch, J. R. I. J., 1999. The Unified Software Development Process. In: Boston: Addison-Wesley.

Jackson, P., 2022. *What is PHP? Write your first PHP Program*. [Online]  
Available at: <https://www.guru99.com/what-is-php-first-php-program.html>  
[Accessed 16 December 2022].

javatpoint, 2022. *Learn JavaScript Tutorial*. [Online]  
Available at: <https://www.javatpoint.com/javascript-tutorial>  
[Accessed 16 December 2022].

javatpoint, 2022. *What is HTML*. [Online]  
Available at: <https://www.javatpoint.com/what-is-html>  
[Accessed 16 December 2022].

javatpoint, 2022. *XAMPP TUTORIAL*. [Online]  
Available at: <https://www.javatpoint.com/xampp>  
[Accessed 16 December 2022].

Jr., F. P. B., 1995. *The Mythical Man-Month: Essays on Software Engineering*. In: s.l.:Addison-Wesley Professional.

Laravel, 2022. *Introduction*. [Online]  
Available at: <https://laravel.com/docs/4.2/introduction>  
[Accessed 16 December 2022].

Moore, L., 2022. *MySQL*. [Online]  
Available at: <https://www.techtarget.com/searchoracle/definition/MySQL>  
[Accessed 16 December 2022].

Moreau, E., 2022. *What Is the Google Chrome Browser?*. [Online]  
Available at: <https://www.lifewire.com/what-is-google-chrome-4687647>  
[Accessed 16 December 2022].

Mustafeez, A. Z., 2022. *What is Visual Studio Code?*. [Online]  
Available at: <https://www.educative.io/answers/what-is-visual-studio-code>  
[Accessed 16 December 2022].

Schwaber, K., 2001. *Agile Software Development with Scrum*. In: s.l.:Prentice Hall.

techopedia, 2018. *Cascading Style Sheet (CSS)*. [Online]  
Available at: <https://www.techopedia.com/definition/26268/cascading-style-sheet-css>  
[Accessed 16 December 2022].

tutorialspoint, 2022. *ReactJS - Overview*. [Online]  
Available at: [https://www.tutorialspoint.com/reactjs/reactjs\\_overview.htm](https://www.tutorialspoint.com/reactjs/reactjs_overview.htm)  
[Accessed 16 December 2022].

## 8. Appendix

### 8.1. Appendix – A: Web Application Features

#### 1. Admin

- Login to the app using their email address and password.
- Set up and manage their password if necessary.
- View and manage all users (passengers and drivers) and their profiles, including personal information, ride history, ratings and reviews, and any other relevant details.
- Set policies and rules for drivers and verify the driver based on their driver KYC details which includes driving license as a necessary credentials.
- Monitor and analyze ride data.
- Manage payment processing and billing for rides.
- Set pricing and pricing policies for rides, including dynamic pricing based on factors such as demand, time of day, and distance travelled.
- View and manage driver ratings and reviews to ensure that drivers are meeting the required standards of service and safety.
- Log out of the app when they are finished using it to protect their account and personal information and prevent unauthorized access.

#### 1. Passenger

- Login to the app using their email address and password.
- Register for a new account by providing their personal information, such as name, and email address.
- Set up and manage their password if necessary.
- Create and manage their personal profile.
- Request rides by specifying a pickup and drop-off location, and view quotes and estimated arrival times for available drivers.
- View a map with real-time location tracking of their assigned driver and the estimated time of arrival at their destination.
- Rate and review drivers after a ride to provide feedback and help other passengers make informed decisions.

- View and manage their ride history.
- Accept or cancel a ride request if needed.
- Set up and manage digital payment options.
- View and manage their preferences or special requests, such as preferred vehicle types or accessibility options.
- Log out of the app when they are finished using it to protect their account and personal information and prevent unauthorized access.

## 2. Driver

- Login to the app using their email address and password
- Register for a new account by providing their personal information and vehicle details.
- Set up and manage their password if necessary.
- Create and manage their personal profile.
- View a map with real-time updates on ride requests and their current location.
- Accept or decline ride requests.
- View and update their availability status to let passengers know when they are available to pick up a ride.
- View and update their earnings and payout information, including a summary of their total earnings.
- View and manage their ride history.
- Log out of the app when they are finished using it to protect their account and personal information and prevent unauthorized access.

[Back to Previous Content](#)

## 8.2. Appendix – B: Phases of RUP Methodology

### Reasons for Choosing RUP Methodology

Here are the several reasons why I selected to use the RUP methodology for my project:

- It is easy to follow and understand due to its well-documented and well-structured nature.
- It is flexible and adaptable, allowing customization based on project requirements.
- It is based on software development best practices and provides tools and templates for tracking performance.
- It is widely used and has a large practitioner community for easy access to information and support.
- It is suitable for a variety of software development projects.

### Phases of RUP Methodology

The RUP methodology has four phases throughout its lifecycle:

#### a) Inception

This phase is the preliminary or the first phase of the development process that focuses on clarifying the project's initial boundaries and highlighting critical stakeholders. It entails collecting and documenting the software's requirements as well as developing a high-level strategy for the development effort.

The following are the key process that takes place during the initiation phase:

- Project Idea
- Project Finalization
- Research about project related topics in depth
- Requirement Gathering
- Research about the technology used

#### b) Elaboration

This phase is the second phase of the development process that entails fine-tuning the initial software design and developing a more thorough development strategy. It consists of actions such as prototyping, risk assessment, and the creation of a thorough project timeline.



The following are the key process that takes place during the elaboration phase:

- Identify project tasks and its completion date
- Project Architecture
- Project UML Diagrams
- Risk Analyzation
- Proposal Confirmation

### **c) Construction**

This phase is the third phase of the development process that includes the actual software development, which includes coding, testing, and integration.

The following are the key process that takes place during the elaboration phase:

- Development
- Testing
- Prepare Documentation

### **d) Transition**

This phase is the last or final phase of the development process that deals with deploying the software and delivering it to the production environment. User acceptance, testing, training, and deployment are all part of the process.

The following are the key process that takes place during the elaboration phase:

- Supervisor Reviews and Feedbacks
- Updates and Upgrades
- Update Project Documentation
- Project Submission

[Back to Previous Content](#)

### 8.3. Appendix – C: Survey Result

Have you ever used a ride-sharing service before?

7 responses

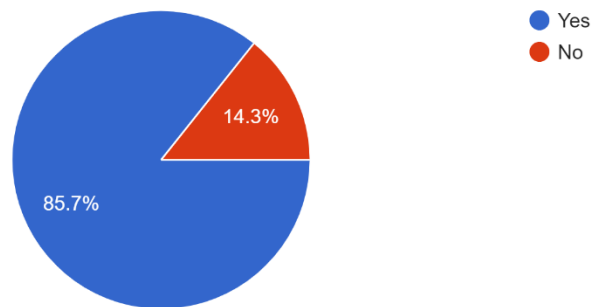


Figure 41: Survey 6

How often do you use ride-sharing services?

7 responses

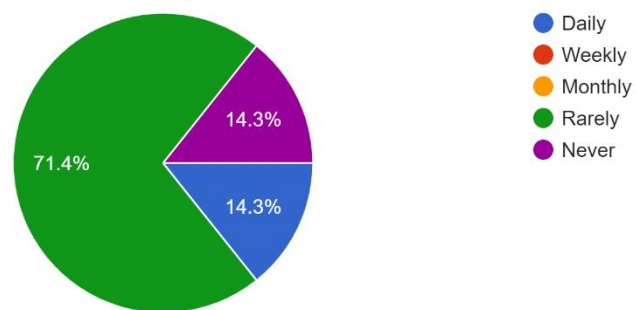


Figure 42: Survey 7

Have you ever had a negative experience while using a ride-sharing service?

7 responses

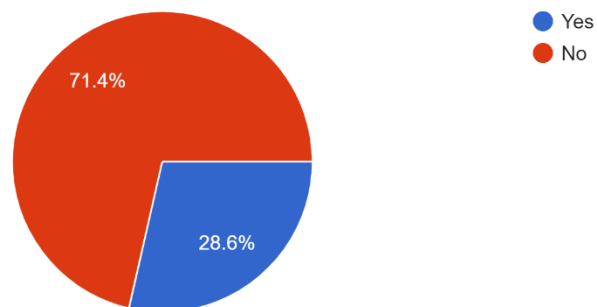


Figure 43: Survey 8

If yes, share your negative experience:

7 responses

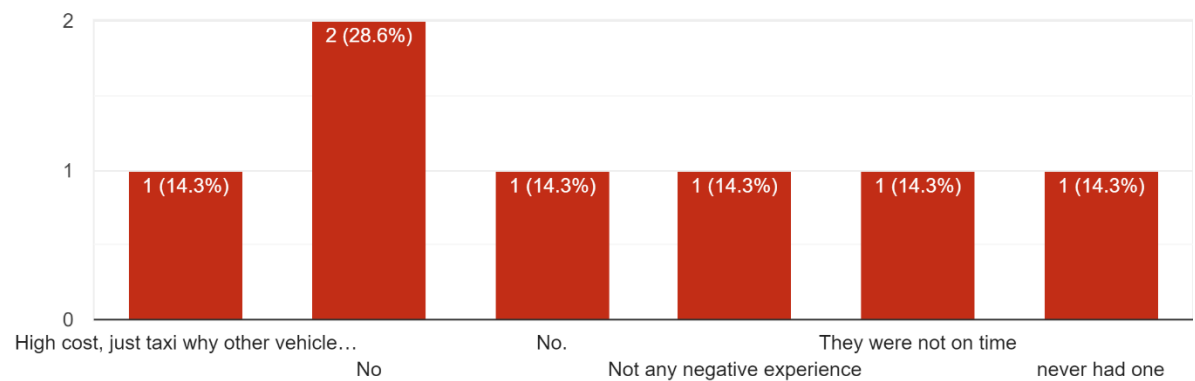


Figure 44: Survey 9

Would you recommend this web app to others?

7 responses

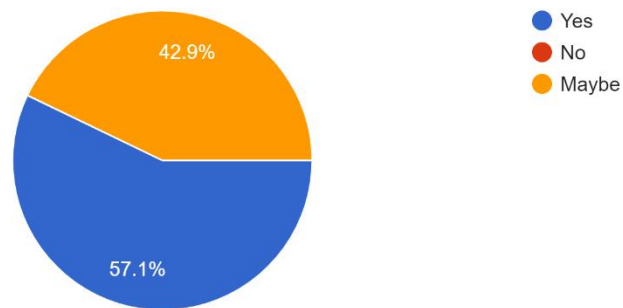


Figure 45: Survey 10

On the scale of 1 to 5, rate this web app

7 responses

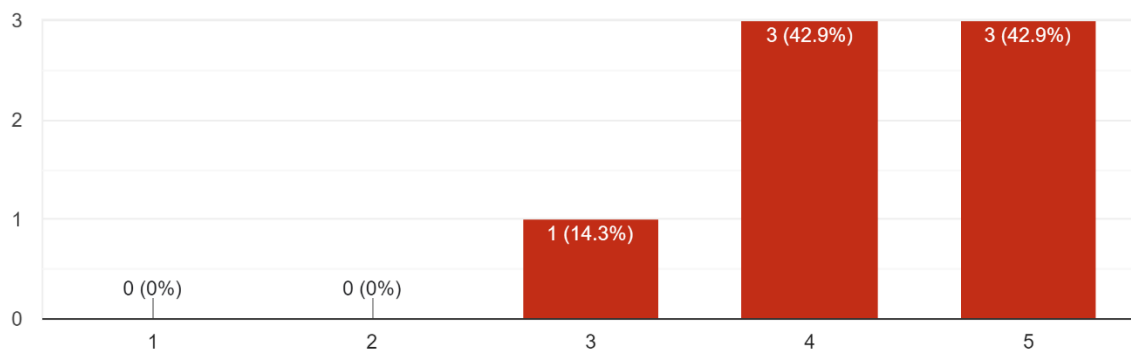


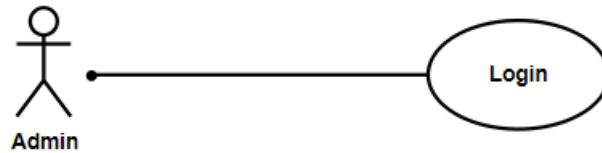
Figure 46: Survey 11

[Back to Previous Content](#)

## 8.4. Appendix – D: Individual Use Case

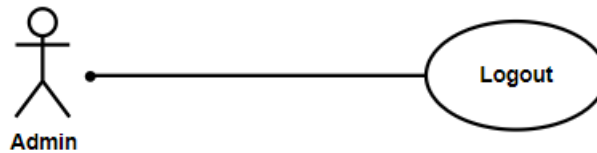
### 1. Admin

#### a) Login



*Figure 47: Individual Use Case of Admin Login*

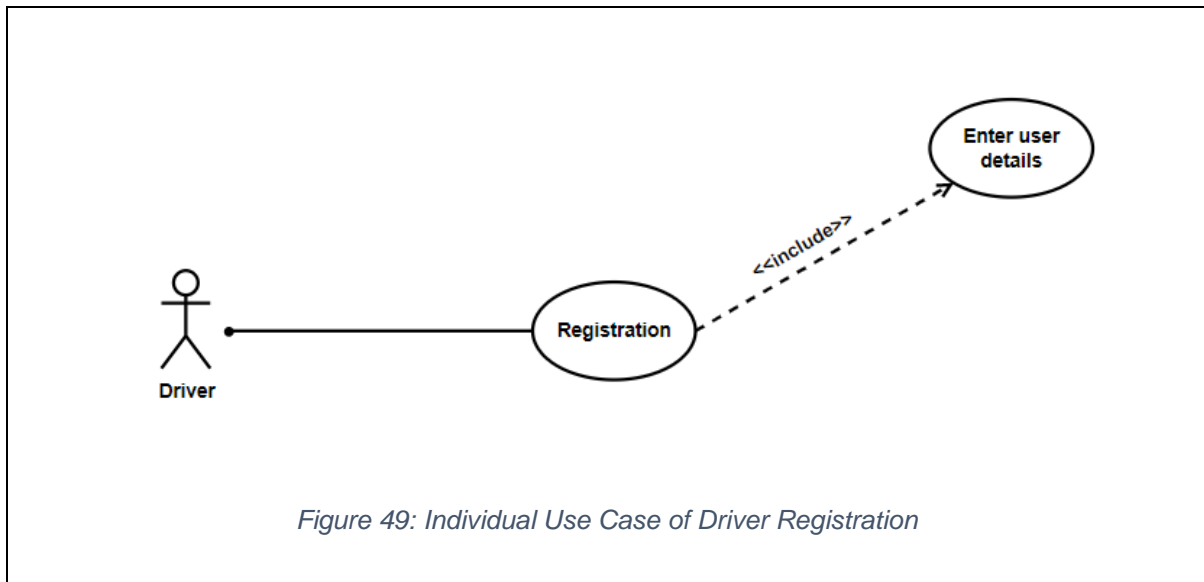
#### b) Logout



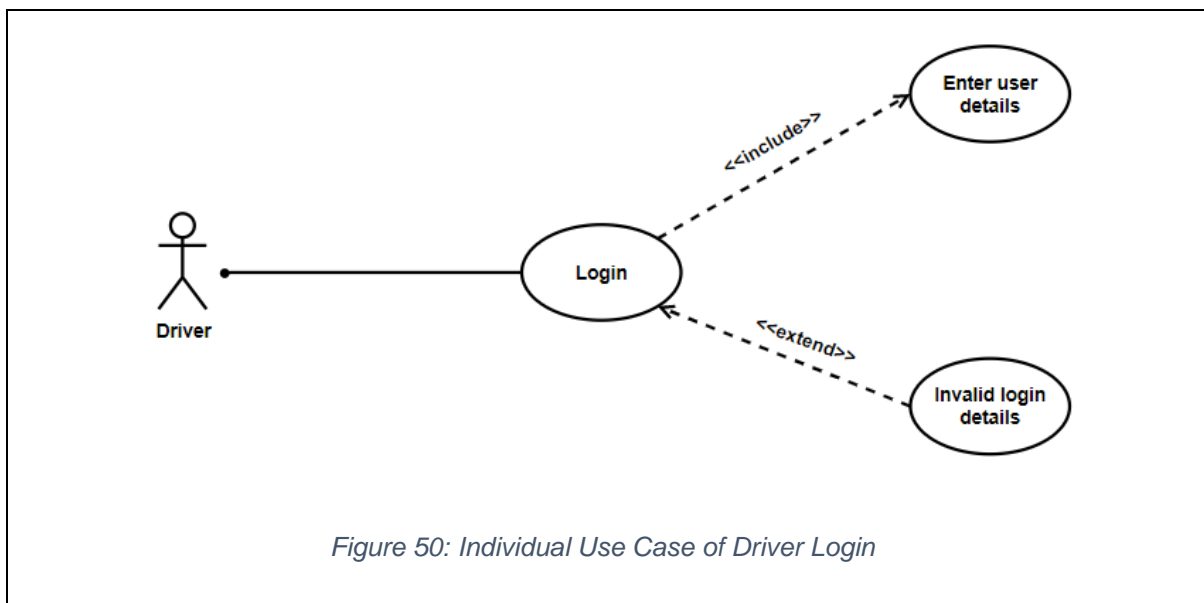
*Figure 48: Individual Use Case of Admin Logout*

## 2. Driver

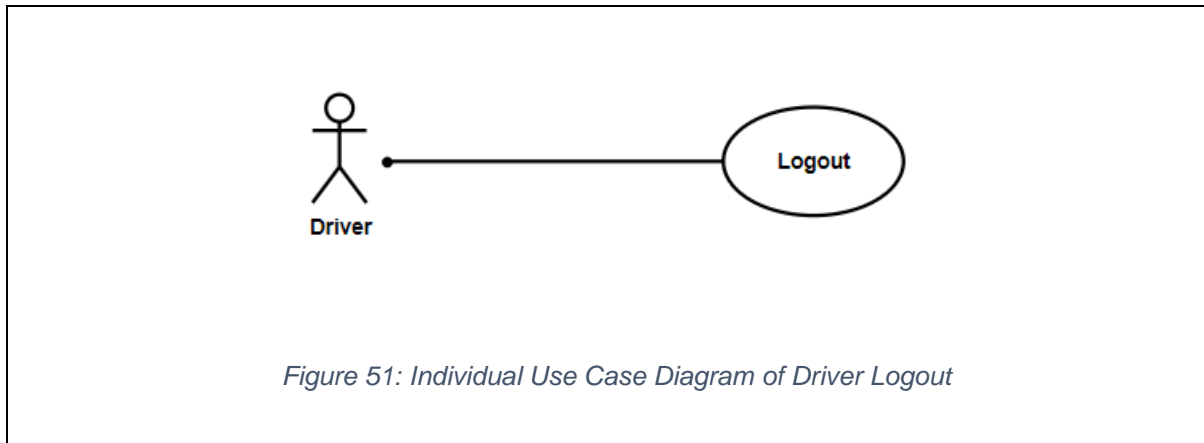
### a) Registration



### b) Login

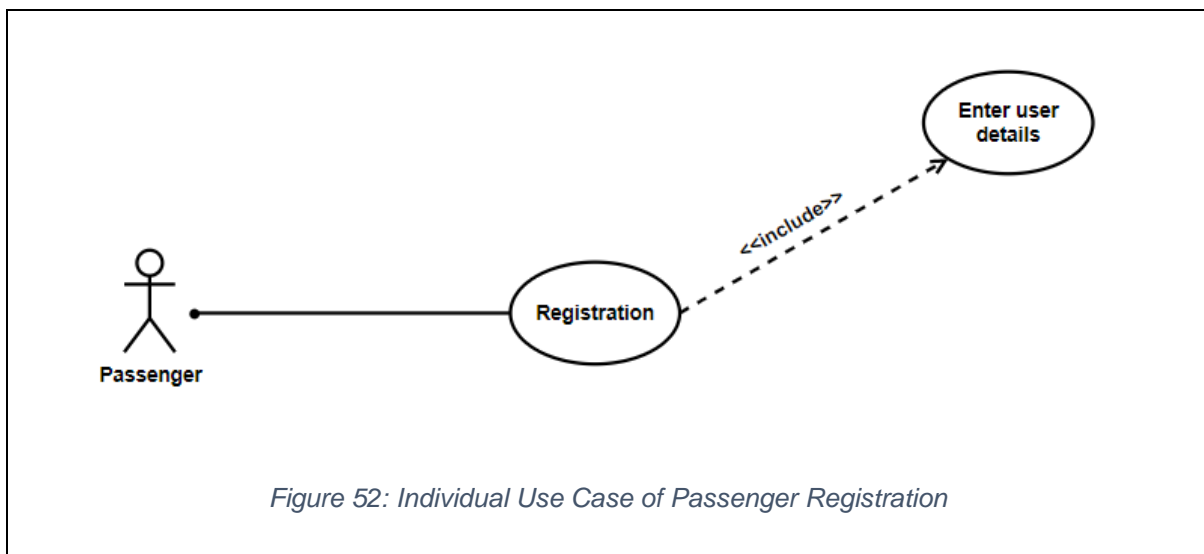


## c) Logout

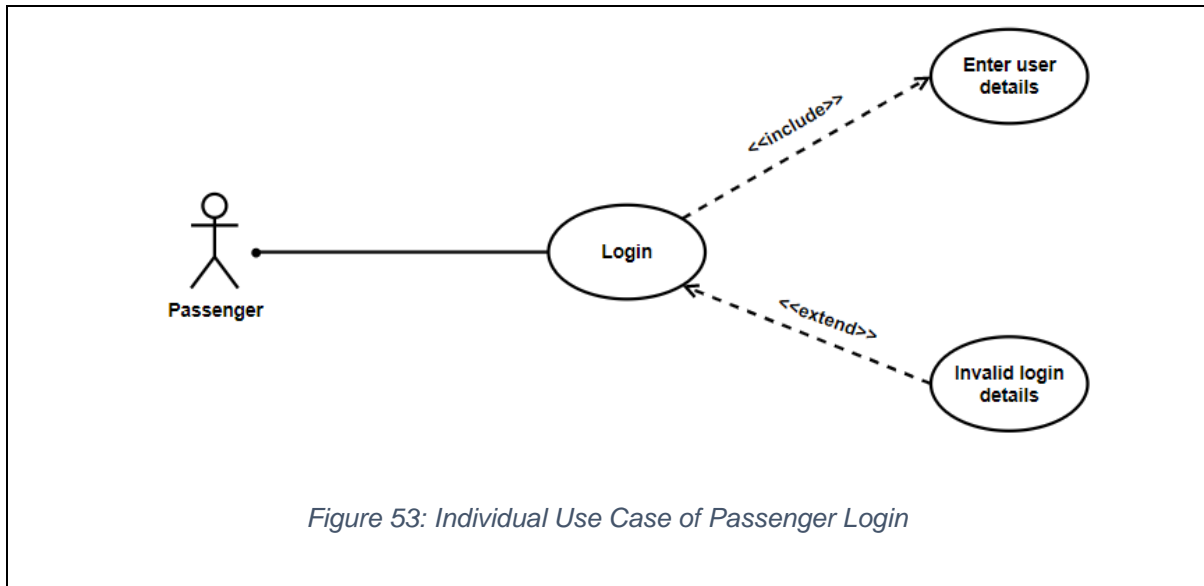


## 3. Passenger

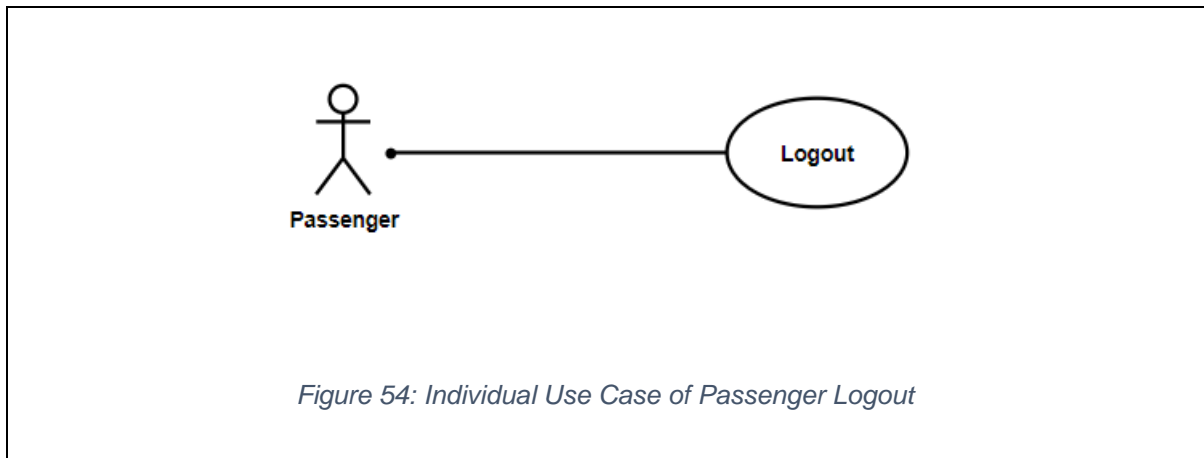
## a) Registration



## b) Login



## c) Logout

[Back to Previous Content](#)

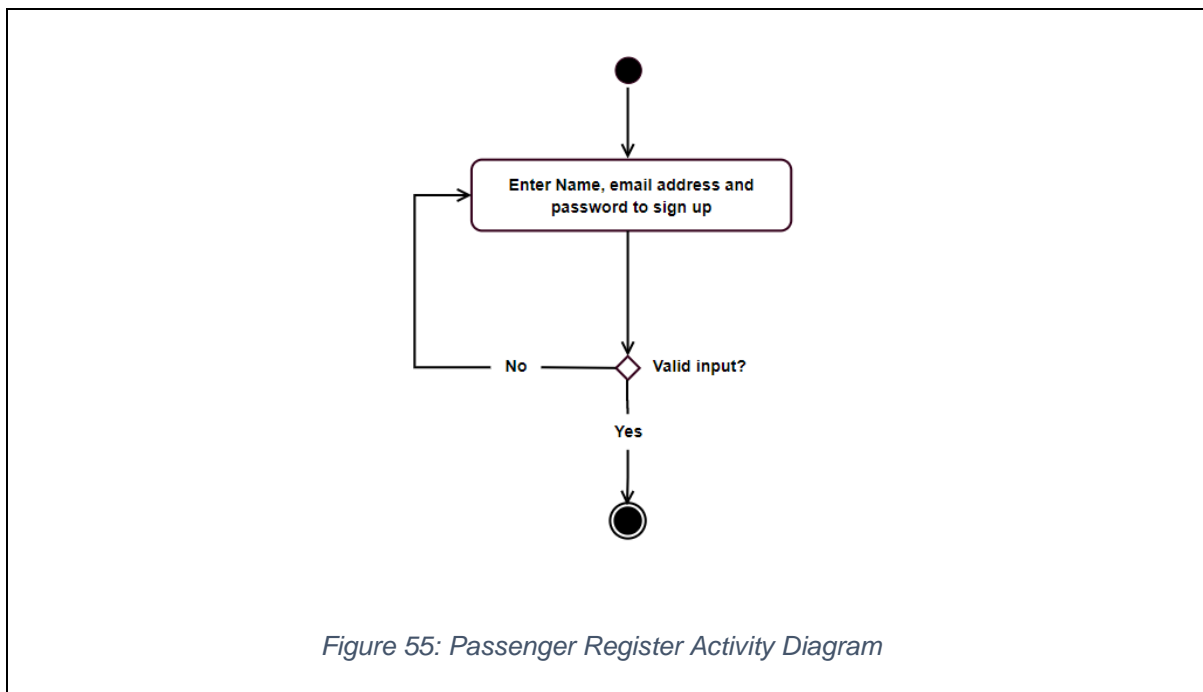


## 8.5. Appendix – E: Activity Diagram

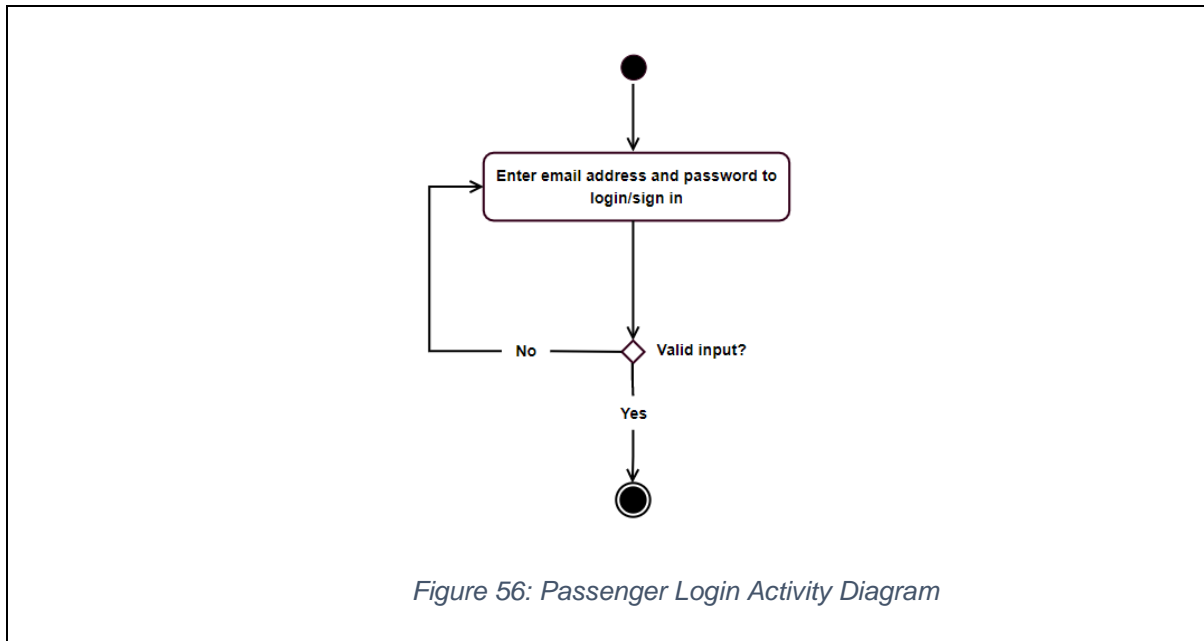
An activity diagram is a graphical representation of the flow of activities or actions within a system. It is typically used to model the behavior of a system or process and can be used to represent the flow of control in a software system, the processing of a request or order, or the steps involved in a business process. Activity diagrams are created using a set of standardized symbols and notation, which are connected by arrows to indicate the flow of control from one step to the next.

### 1. Passenger Activity Diagram

#### a) Passenger Register Activity Diagram

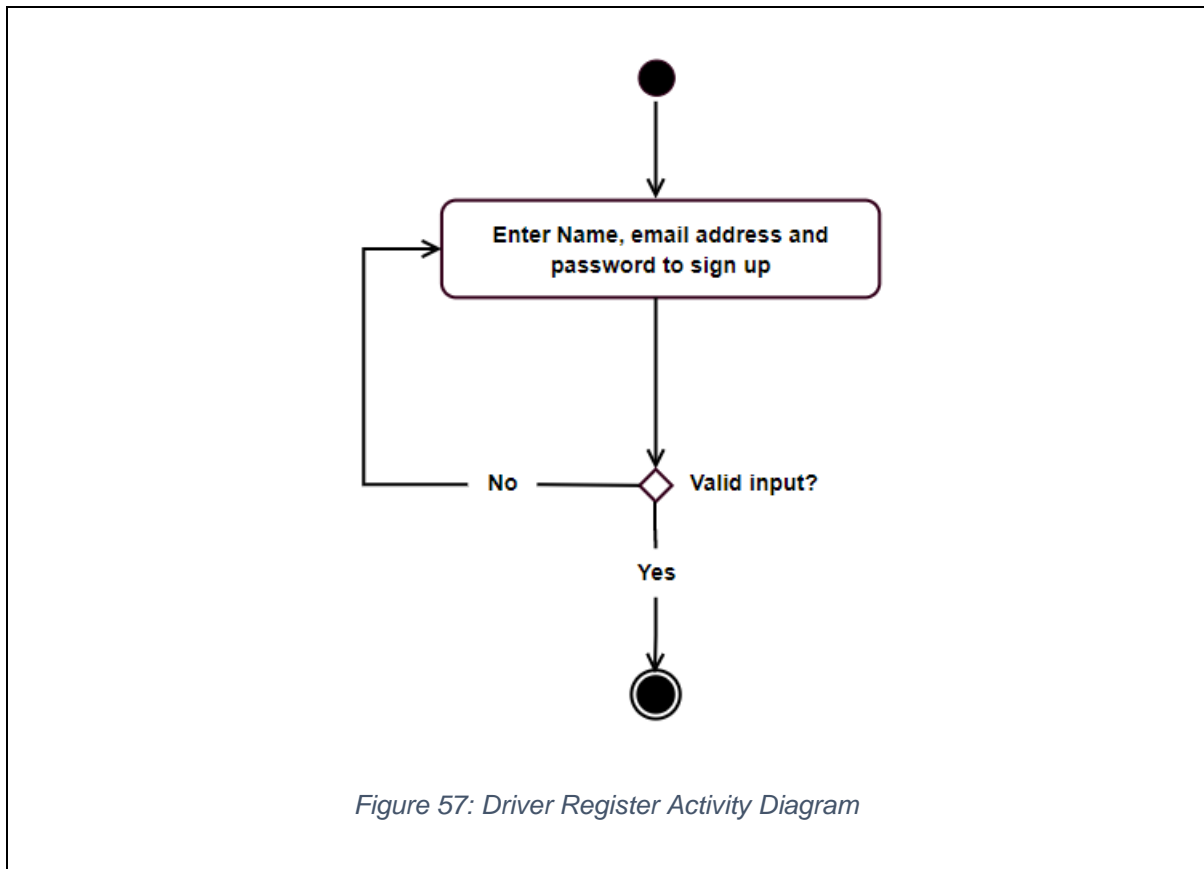


## b) Passenger Login Activity Diagram

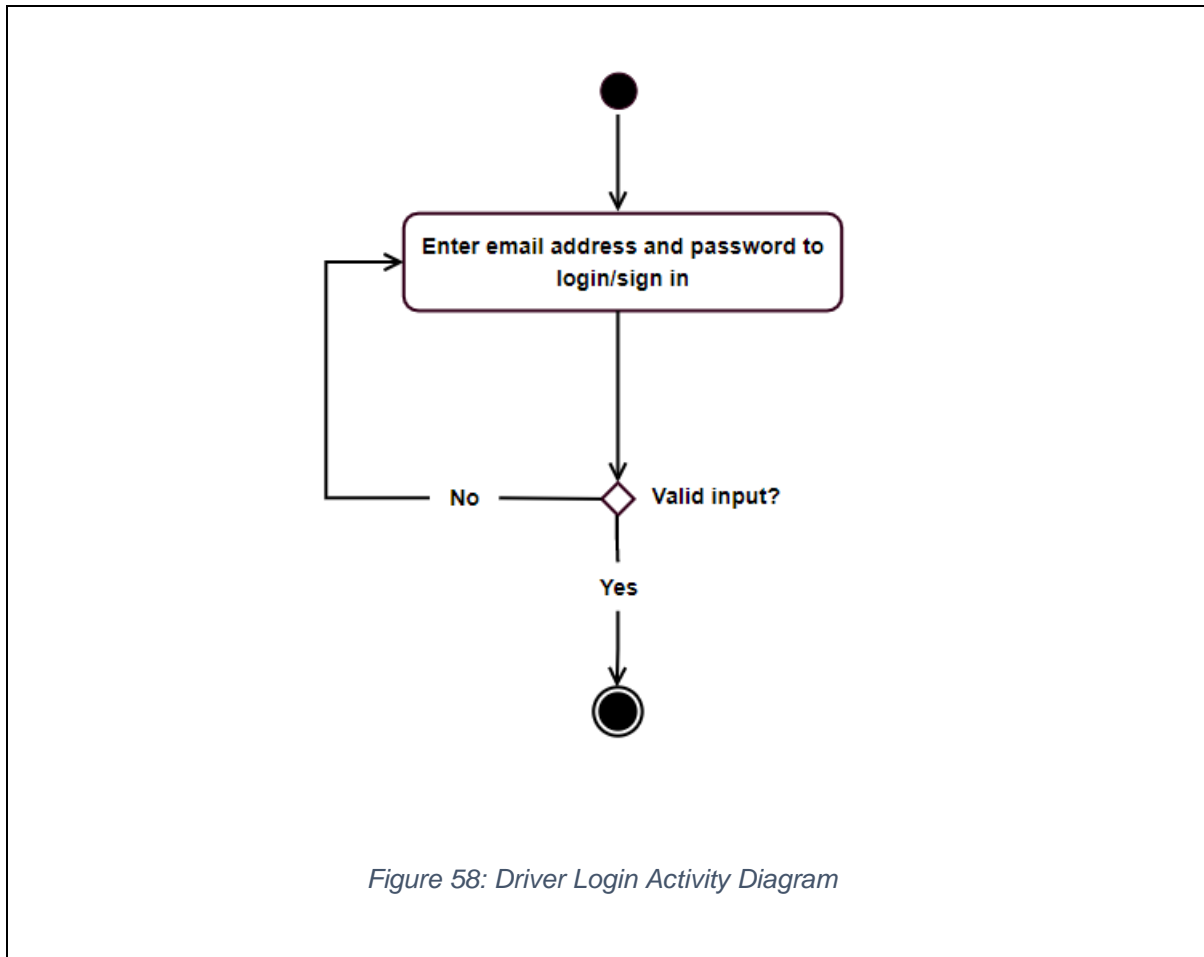


## 2. Driver

### a) Driver Register Activity Diagram

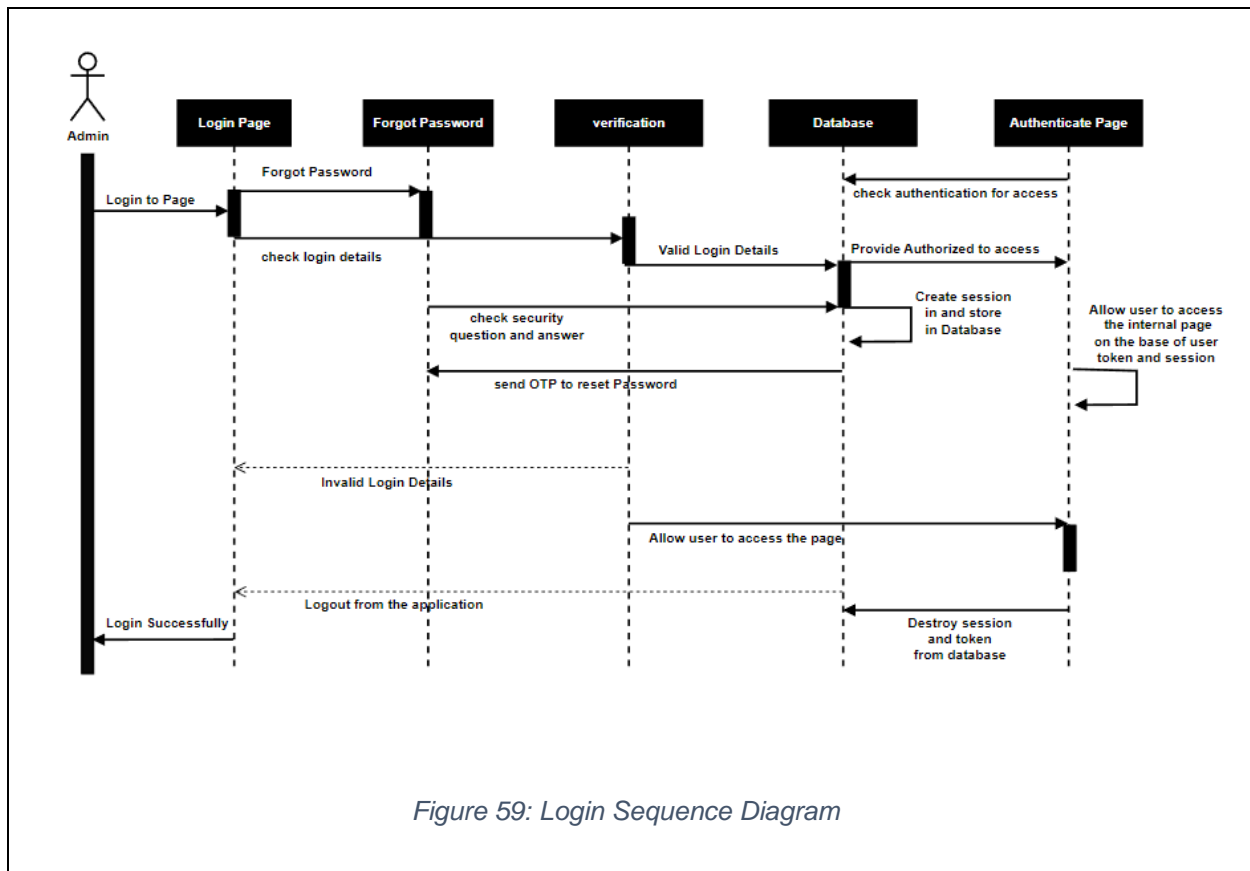


## b) Driver Login Activity Diagram

[Back to Previous Content](#)

## 8.6. Appendix – F: Sequence Diagrams

A sequence diagram is a graphical representation that demonstrates the interactions of objects or components in a system through time. It shows the chain of messages or events exchanged between objects, as well as the order in which they occur. In software development, sequence diagrams are frequently used to visualize the flow of messages and data between objects or components in a system, as well as to facilitate in the design and debugging of complicated systems.



[Back to Previous Content](#)

## 8.7. Appendix – G: Architecture Diagram

An architecture diagram is a visual representation of the structure and components of a system or application. It is a high-level view of the overall design and organization of a system and is used to communicate the design and structure to stakeholders and team members. An architecture diagram is a helpful tool for comprehending a system's general design and is an important aspect of the design and development process.

### Above Architecture Diagram Description

In the diagram, the web app and server are connected over a network, such as the Internet. The user starts by logging into the web app by entering their login credentials. The server authenticates the credentials and sends a response back to the web app indicating whether the login was successful or not. If the login was successful, the web app displays the main menu to the user. From the main menu, the user can request a ride by entering their pickup location and destination. The server processes the ride request and sends a response back to the web app with the ride details, including the estimated pickup time and price. The user can then enter their payment details and the server processes the payment.

[Back to Previous Content](#)

## 8.8. Appendix – H: Class Diagram

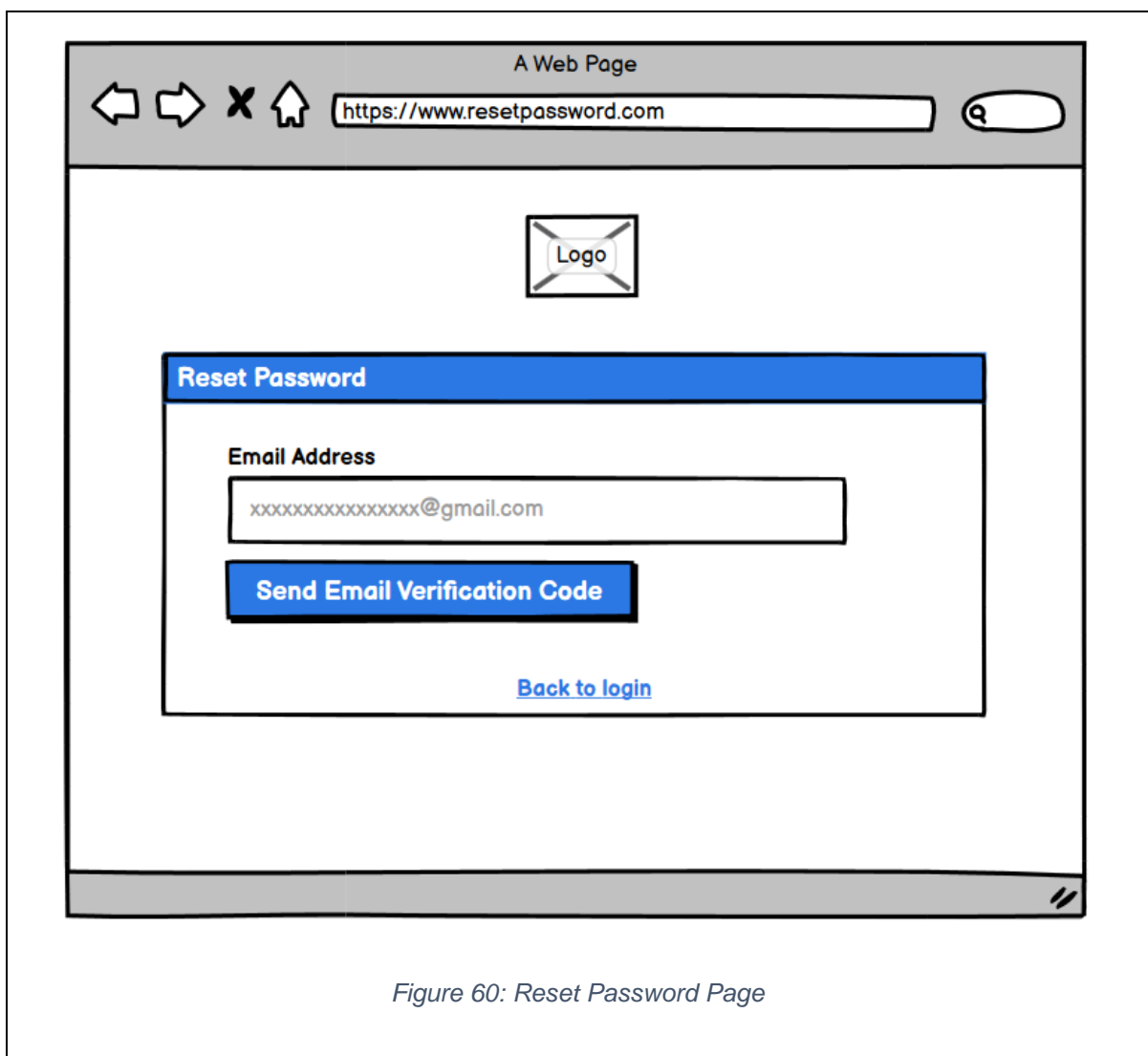
Class diagram is a type of Unified Modeling Language (UML) diagram that depicts the structure of a system by displaying the classes, properties, operations, and relationships between them. Class diagrams are used to represent the static aspects of a system, and they are highly useful for comprehending and explaining the design of object-oriented systems.

[Back to Previous Content](#)

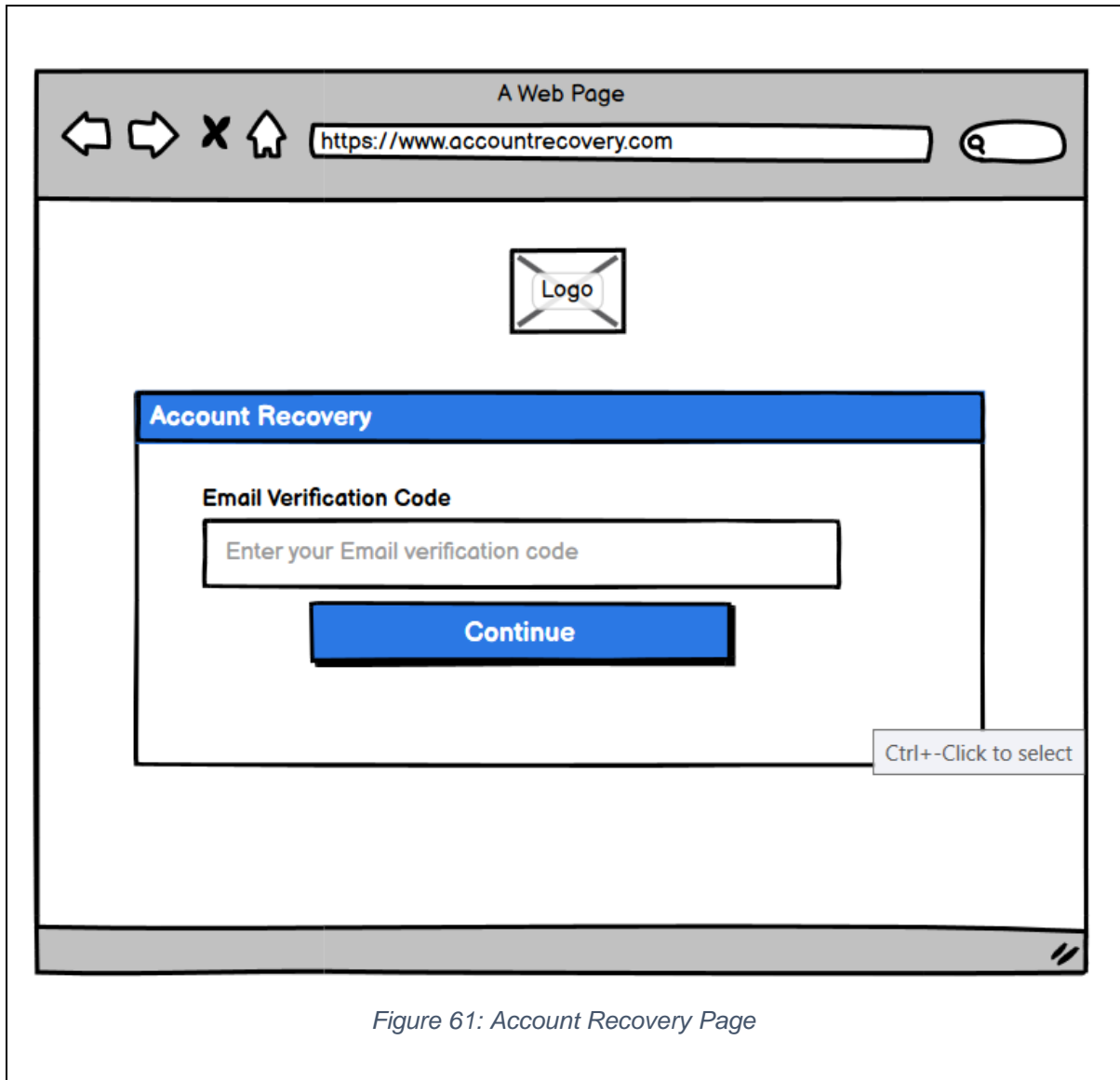
## 8.9. Appendix – I: Wireframes

A wireframe is a visual representation of the structure of a website or application's user interface design. It is a user experience blueprint that shows how the interface will look and perform. Wireframes are often developed early in the design process and are used to demonstrate to stakeholders the structure, hierarchy, and functionality of a design. This allows designers and stakeholders to focus on the user experience and functionality of the interface rather than the visual aspects (Doe, 2018).

### a) Reset Password Page



## b) Account Recovery Page





## c) Change Password

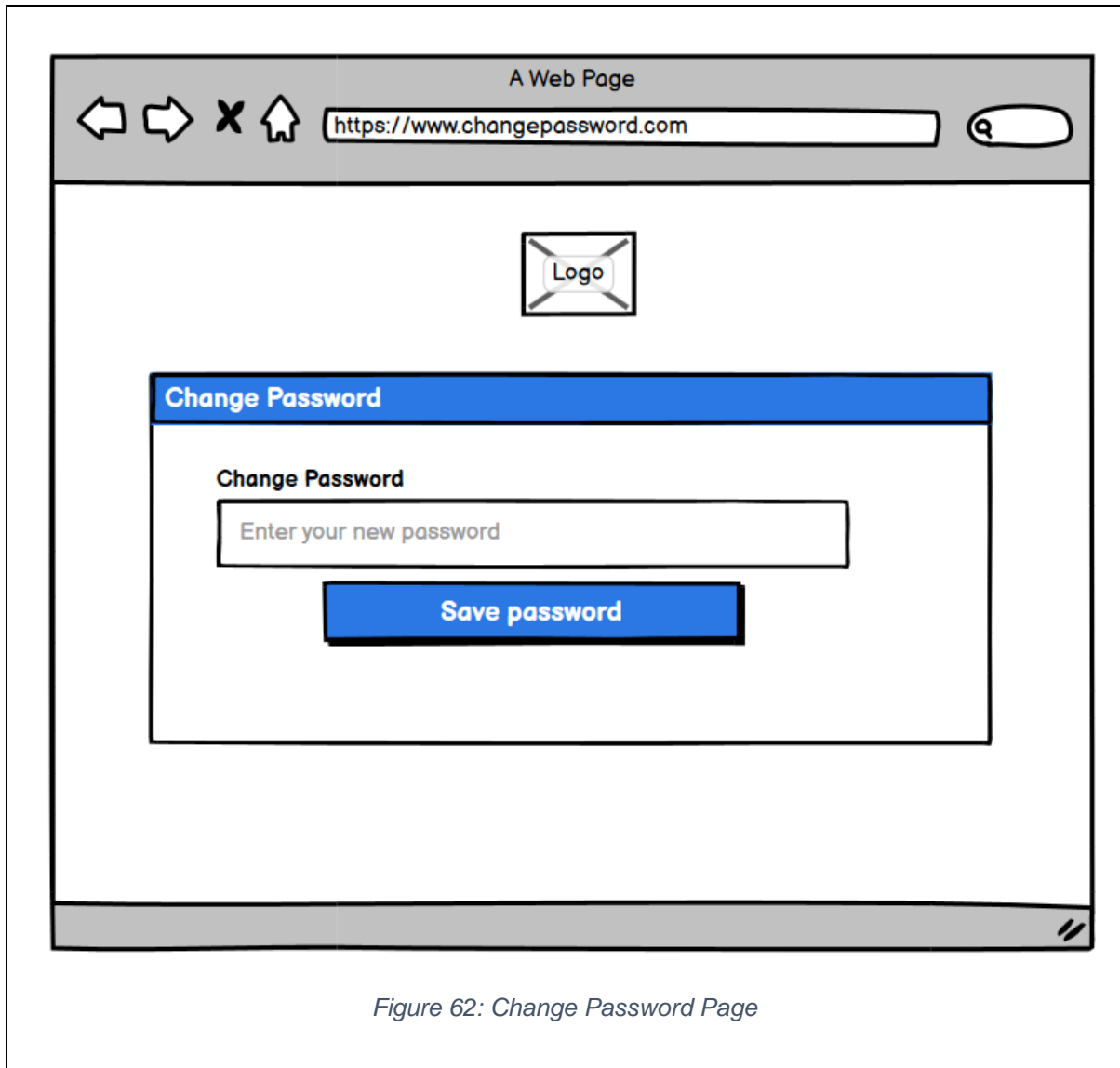


Figure 62: Change Password Page

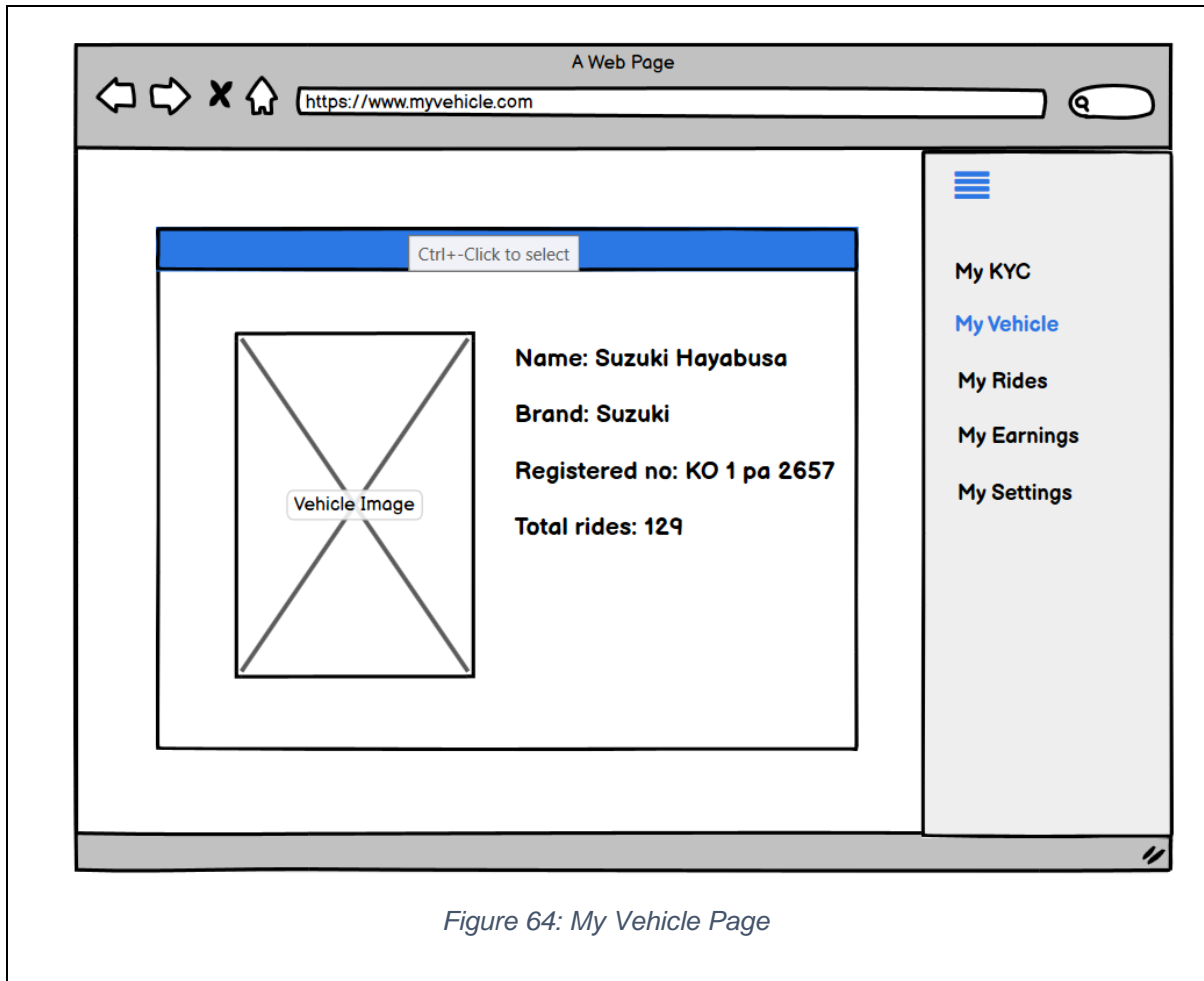
## d) My KYC Page: Driver

The image shows a web browser window titled "A Web Page" with the address bar displaying "https://www.mykyc.com". The page content is divided into a main area and a sidebar. The sidebar on the right contains a menu with the following items: "My KYC", "My Vehicle", "My Rides", "My Earnings", and "My Settings". The main content area features a form titled "My KYC" with the following fields and controls:

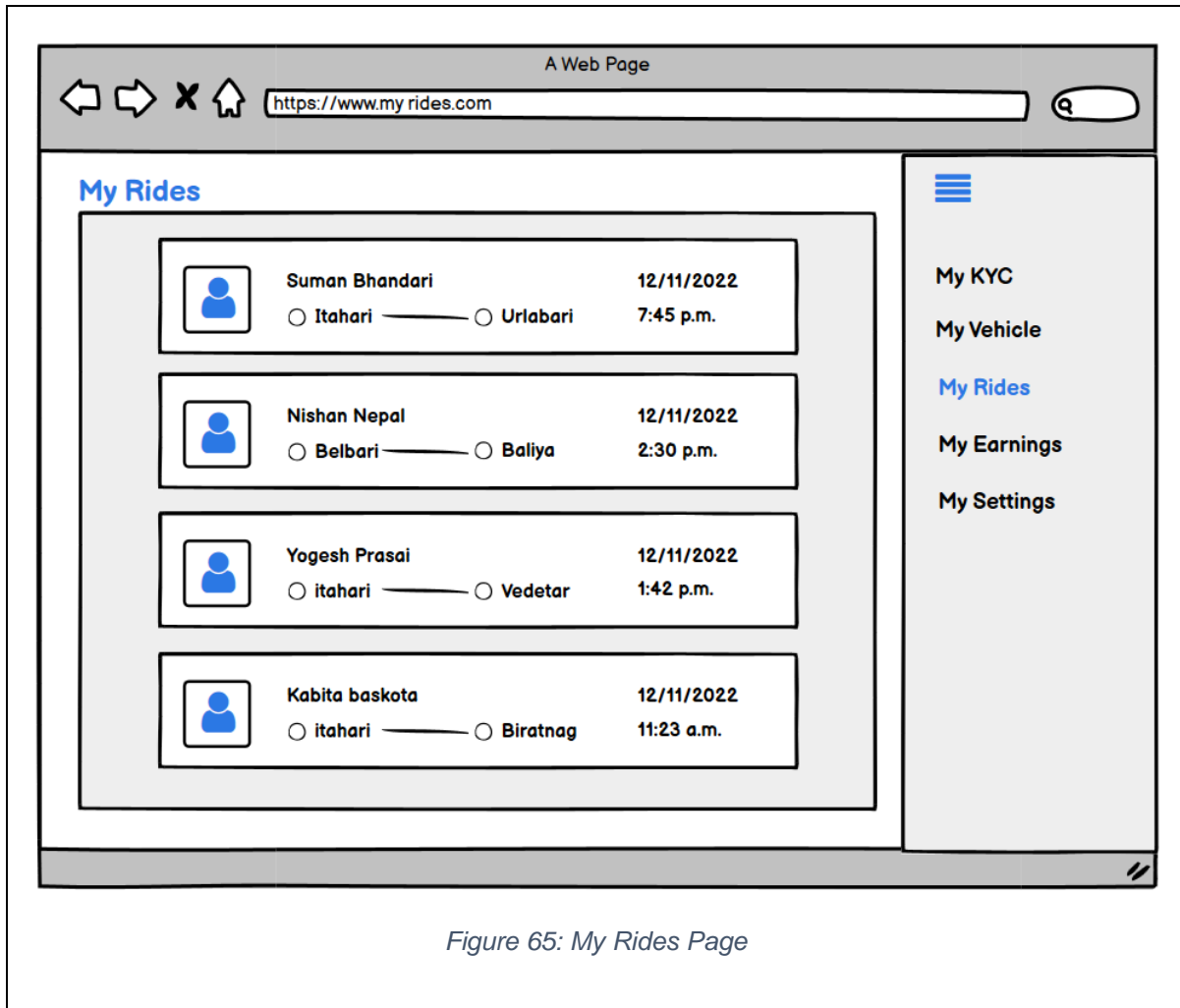
- First name**: A text input field containing "xxxxxxxxxx".
- Last name**: A text input field with a placeholder "Ctrl+Click to select".
- DOB**: A text input field with a placeholder "--/--/----
- Address**: A text input field.
- License**: A text input field with a placeholder "Choose from file".
- License Category**: A dropdown menu.
- Submit**: A blue button.

Figure 63: MY KYC Page

## e) My Vehicle Page: Driver



## f) My Rides Page: Driver



## g) My Earnings Page: Driver

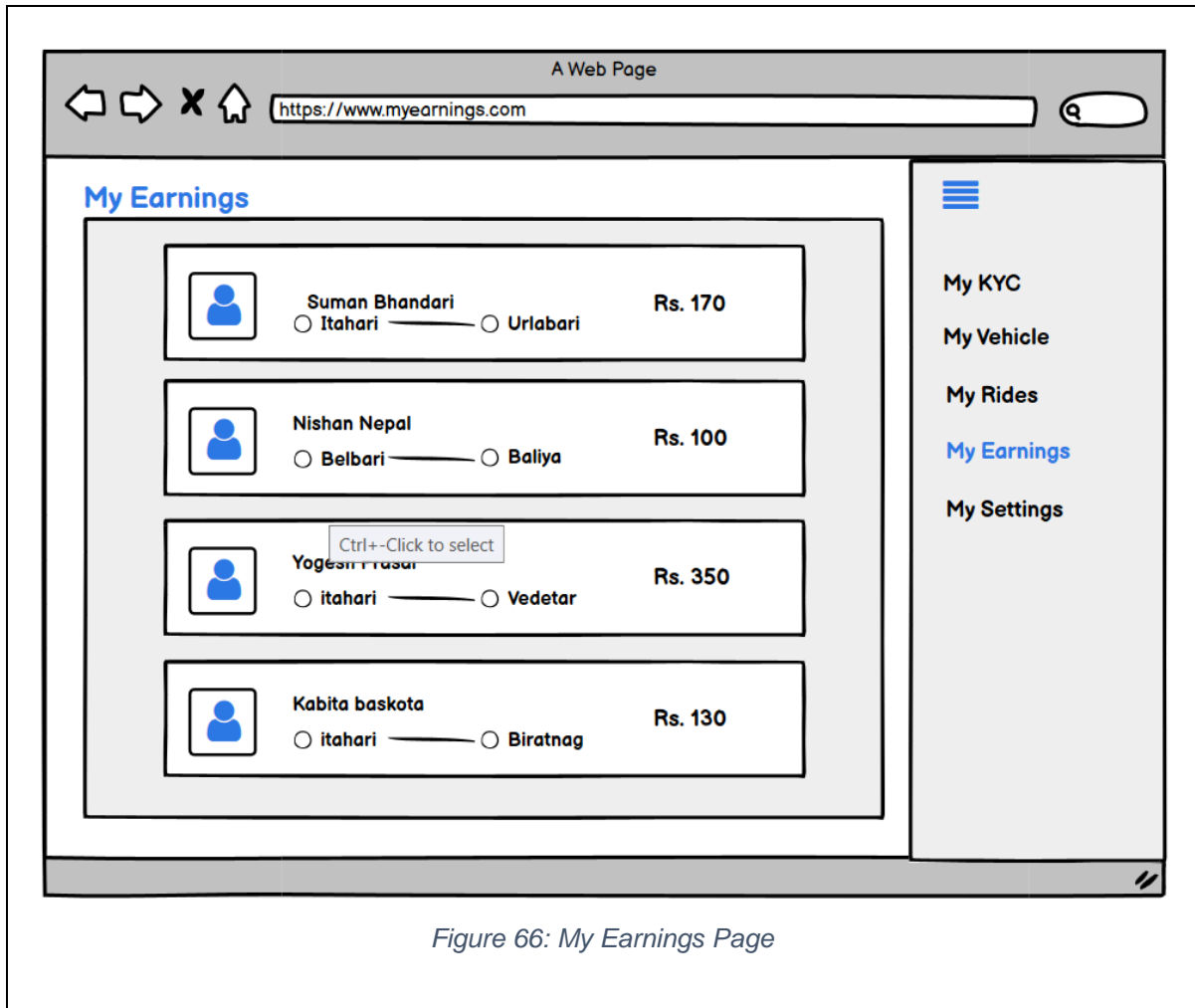


Figure 66: My Earnings Page

## h) My Settings Page: Driver

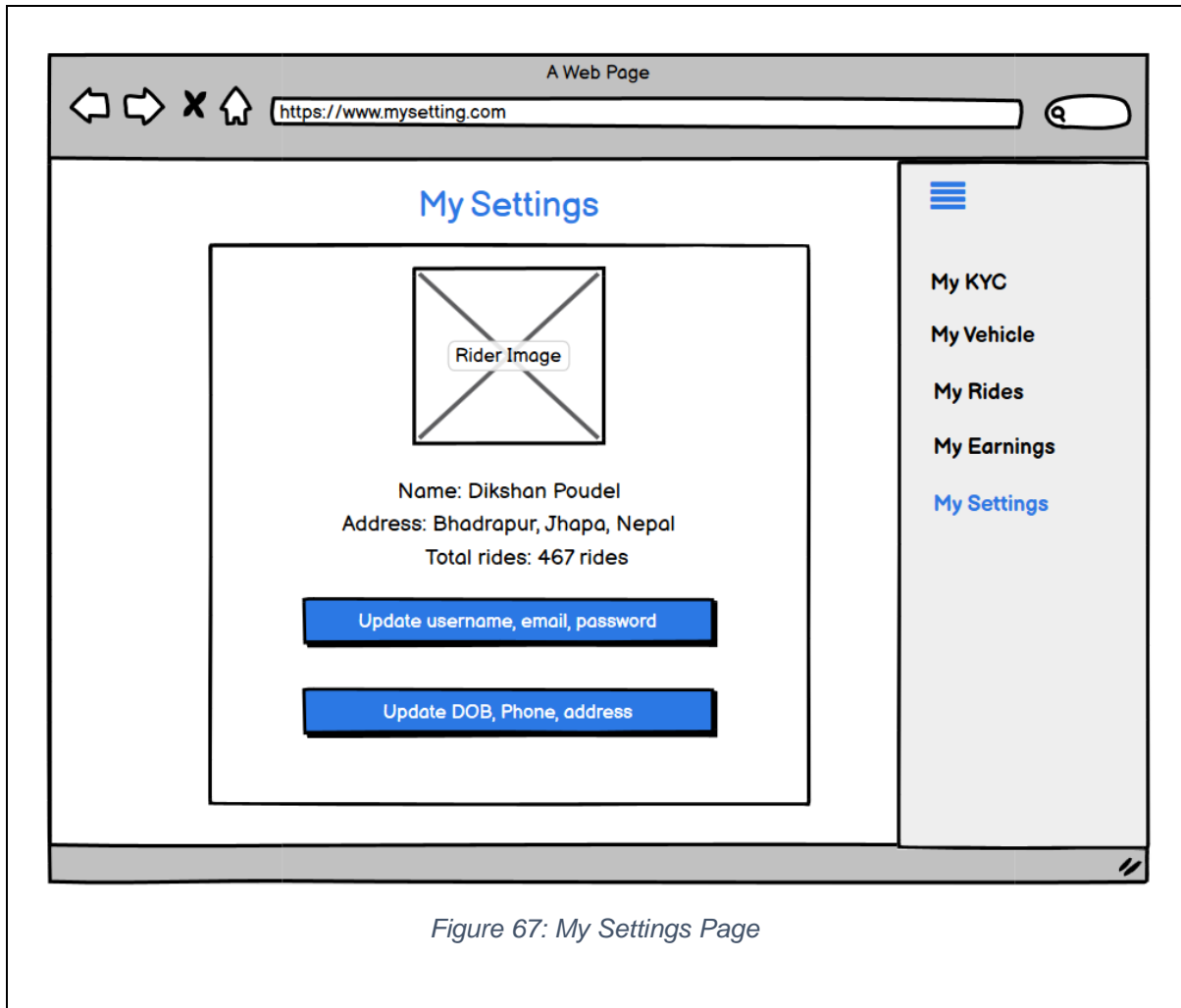


Figure 67: My Settings Page

## i) My Ride History Page: Passenger

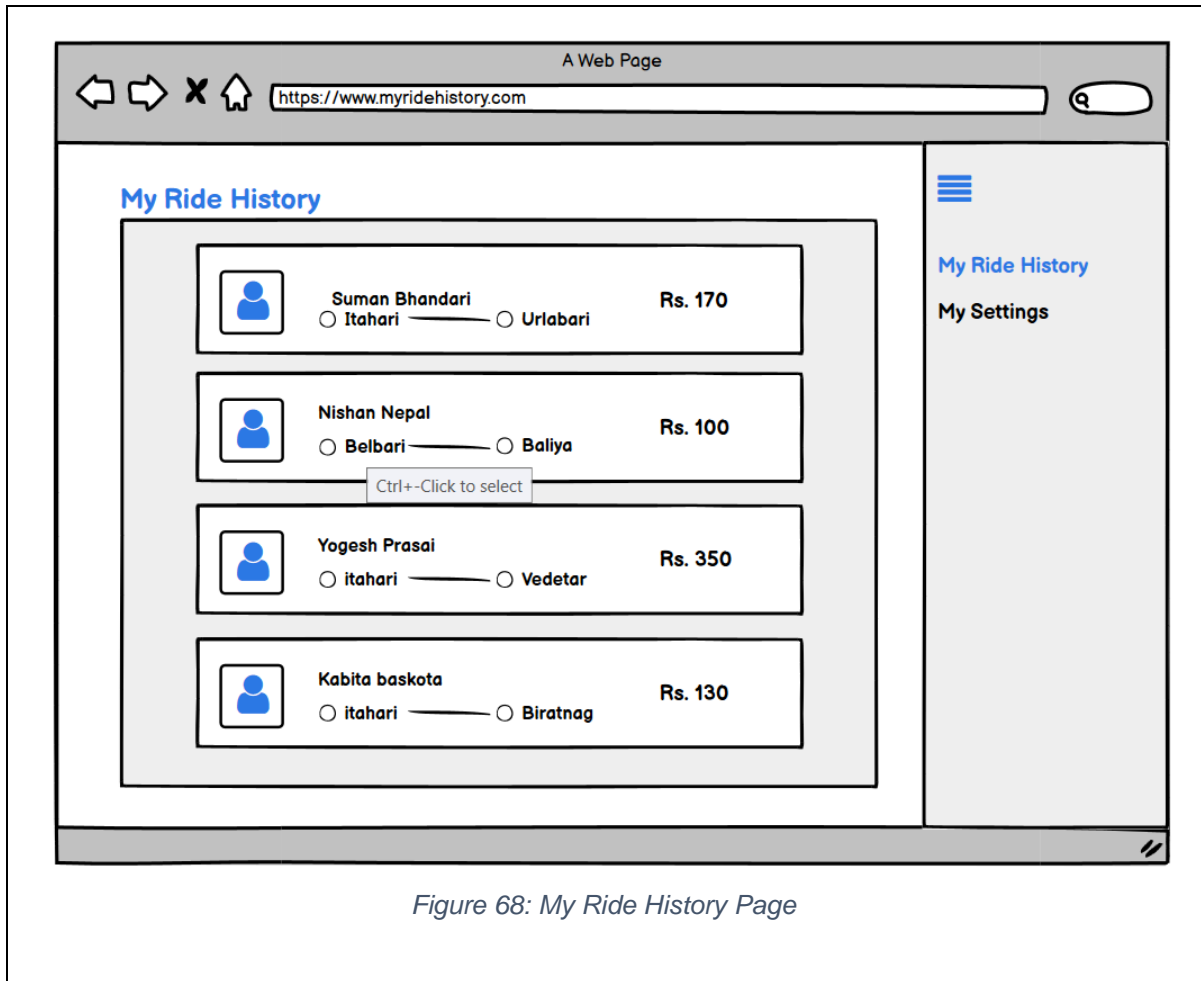
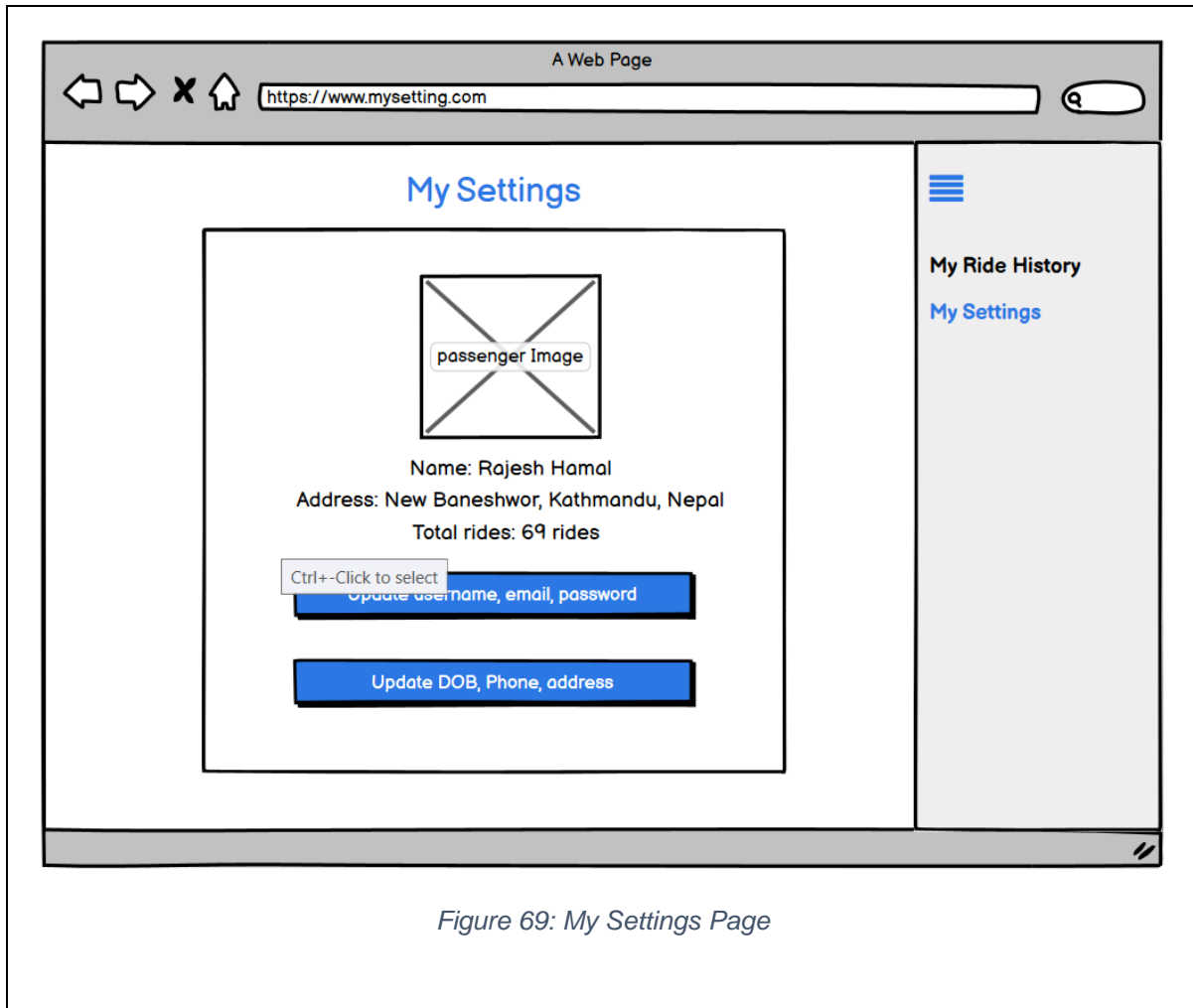


Figure 68: My Ride History Page

## j) My Settings Page: Passenger

[Back to Previous Content](#)



## 8.10. Appendix – J: ERD Details

An Entity Relationship Diagram (ERD) is a visual representation of the relationships between entities in a database. It is used to model and design the structure of a database and is an essential tool for database designers and developers. An ERD is a graphical representation of the data relationships and attributes and is used to model the data requirements and design the database structure.

### Above ERD description

In the above ERD, the entities are:

- **User Entity:** It represents a user of the app, who can be an admin, driver, or passenger. The attributes for this entity includes name, email address, phone number, and password. The primary key for this entity is the "User\_Id" attribute. The "Role\_Id" attribute is a foreign key that references the "Role" entity.
- **Login Entity:** It represents a login session for a user. The attributes for this entity includes email address and password. The primary key for this entity is the "Login\_Id" attribute. The "User\_Id" attribute is a foreign key that references the "User" entity.
- **Registration Entity:** It represents the process of a user signing up for the app. The attributes for this entity includes name, email address, phone number, and password. The primary key for this entity is the "Registration\_Id" attribute. The "User\_Id" attribute is a foreign key that references the "User" entity.
- **Role Entity:** It represents the role of a user. The attributes for this entity include role name. The primary key for this entity is the "Role\_Id" attribute. The "User\_Id" attribute is a foreign key that references the "User" entity.
- **Ride Entity:** It represents a ride that is requested and completed through the app. The attributes for this entity includes pick up, drop off and fares. The primary key for this entity is the "Ride\_Id" attribute. The "User\_Id" attribute is a foreign key that references the "User" entity.
- **Rating Entity:** It represents a rating given by a passenger to a driver after a ride. The attributes for this entity includes rating score and comments. The primary key

for this entity is the "Rating \_Id" attribute. The "Ride \_Id" attribute is a foreign key that references the "Ride" entity.

- **Payment Entity:** It represents a payment made by a passenger to a driver for a ride. The attributes for this entity include payment type. The primary key for this entity is the "Payment \_Id" attribute. The "Ride \_Id" attribute is a foreign key that references the "Ride" entity.
- **KYC Entity:** It represents the process of verifying a driver's identity. The attributes for this entity include name, address, email address, driving license and phone number. The primary key for this entity is the "KYC \_Id" attribute. The "User \_Id" attribute is a foreign key that references the "User" entity.

The relationship between these entities are as follows:

- A "User" can have many "Login" (one-to-many relationship).
- A "User" can have one "Registration" (one-to-one relationship).
- A "User" can be the driver for many "Rides" and can be a passenger for many "Rides" (many-to-many relationship). This relationship is represented by the "Driver" and "Passenger" entities, which have a one-to-many relationship with the "Ride" entity.
- A "Ride" can have one "Payment" (one-to-one relationship).
- A "Ride" can have one "Rating" (one-to-one relationship).
- A "User" can have one "KYC" record (one-to-one relationship).

[Back to Previous Content](#)

### 8.11. Appendix – K: DFD details

A Data Flow Diagram (DFD) is a graphical depiction of data flow in a system. It is used to understand and convey how data is processed and altered as it goes through a system, and it is especially useful for recognizing and modeling the processes, data storage, and external entities involved. DFDs are commonly used in software development and systems analysis.

#### Above DFD description

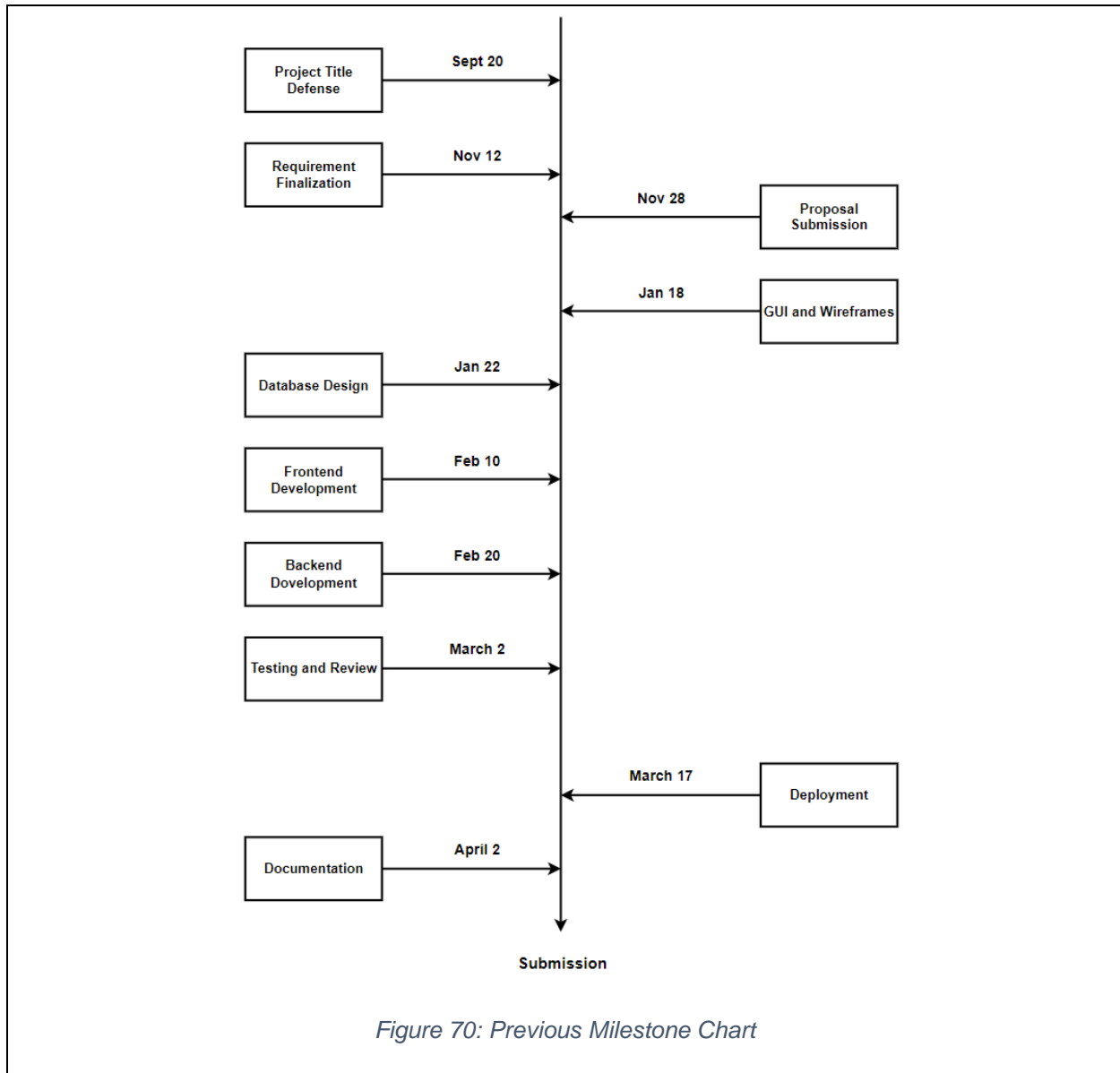
The above DFD in the context of a passenger and driver service could refer to an individual who is either a passenger or a driver using the service. The processes associated with a passenger and driver service may include:

- **Login account:** This process involves creating an account with the passenger and driver service, which may require providing personal information such as an email address and phone number.
- **Find passenger location:** This process involves using GPS technology to locate the passenger's current location so that a driver can pick them up.
- **Offer fare:** This process involves presenting the passenger with a fare estimate for their ride based on the distance and duration of the trip.
- **Find available driver:** This process involves using the passenger and driver service's platform to match a passenger with a nearby driver who is available to provide a ride.
- **Payment:** This process involves the passenger paying for the ride which may can be offline or digital payment.
- **Rate driver:** This process involves the passenger providing feedback on their experience with the driver, which may include rating the driver on a scale and writing a review. This feedback can be used by the passenger and driver service to improve the quality of service provided to passengers and to help drivers improve their performance.

[Back to Previous Content](#)

## 8.12. Appendix – L: Milestone Chart

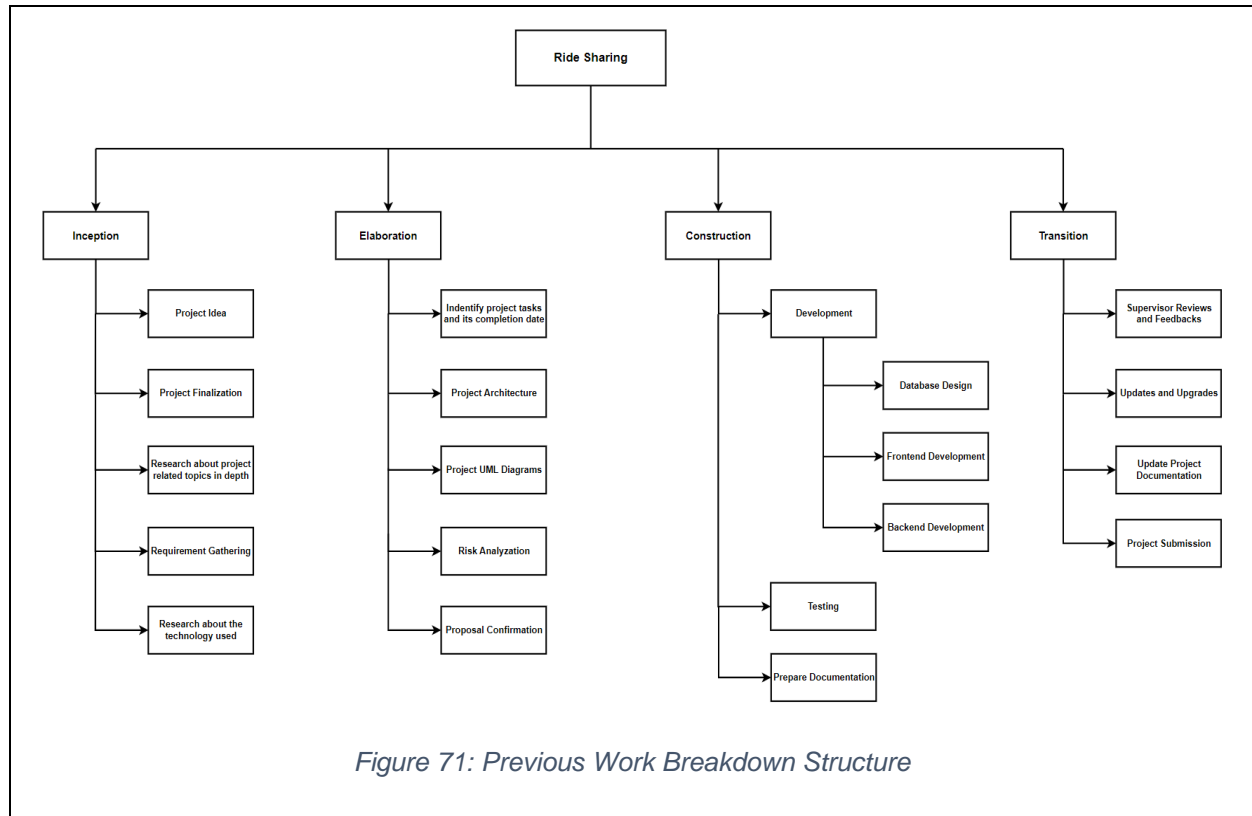
A milestone chart is a graphical representation of the progress of a project or process over time. It is often comprised of a timeline indicating the planned and actual completion dates for a sequence of major milestones or events. Milestones are often visualized as points on a timeline, with the expected completion date displayed as a solid line and the actual completion date represented as a dotted line.



[Back to Previous Content](#)

### 8.13. Appendix – M: WBS

A Work Breakdown Structure (WBS) is a hierarchical illustration of the project work that must be accomplished. It is used to break down a project into smaller, more manageable units of tasks, as well as to organize and coordinate the project team's work. A work breakdown structure (WBS) generally consists of a top-level overview of the project, with lower-level tasks and activities stacked below.



[Back to Previous Content](#)

## 8.14. Appendix – N: Gantt Chart

A Gantt chart is a graphical representation of the schedule and progress of a project or process. It is commonly used to illustrate the tasks and activities associated with a project, as well as to manage and coordinate the project team's activity. A Gantt chart is composed of a horizontal timeline that specifies the duration of each task or activity, as well as vertical bars that specify the start and end dates of each task.

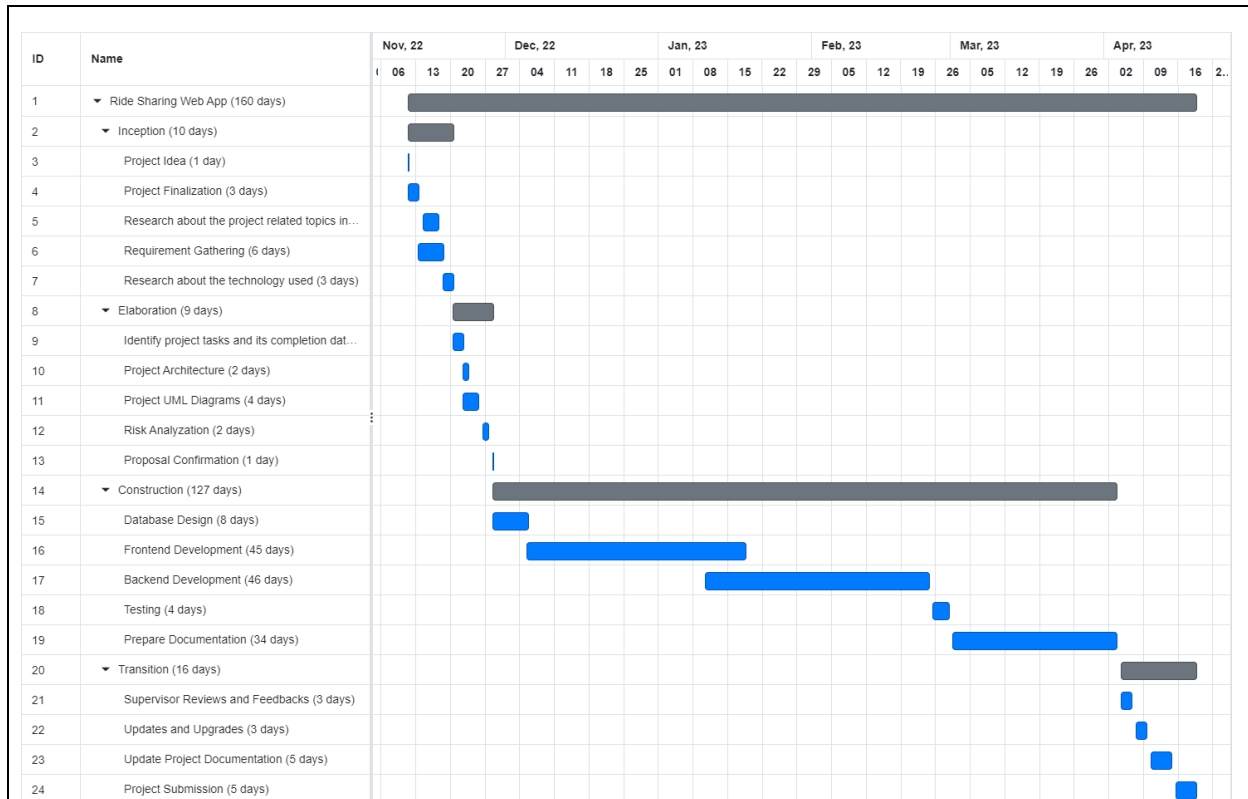


Figure 72: Previous Gantt Chart

[Back to Previous Content](#)

## **8.15. Appendix – O: SRS**

### **1. Introduction**

#### **1.1. Purpose**

The main purpose of creating an SRS document for the ride-sharing web app, "Ride Hero", is to define the requirements clearly and accurately for the system. This document will provide a detailed description of the features of the app and how they operate to enhance the user experience. It will also explain the functionality of the system to facilitate better interaction with the end users.

#### **1.2. Project Scope**

This project can be viewed as a problem-solving project that aims to address the challenge of providing convenient and affordable transportation options for individuals and group of individuals. This project aims to develop a platform that allows users to easily request and pay for rides from drivers using the web browsers and provides drivers with a reliable source of income. The project scope typically includes the development and deployment of a user (passenger) app and a driver app, as well as various features and functionalities that enhance the user experience and improve the efficiency and reliability of the ride-sharing service. By addressing the problem of providing convenient and affordable transportation, this project has the potential to solve real-world challenges and make a positive impact on people's lives.

#### **1.3. Overview**

This document provides a detailed and comprehensive overview of the proposed software system, including its overall description, functional and non-functional requirements, data and behavioral model. It is structured into three main sections: the first section introduces the system, the second section covers the overall description, product perspectives, functions, user characteristics, constraints, assumptions, and dependencies, and the final section presents the specific requirements, data and behavioral model description, and the functional and non-functional requirements of the system. This document aims to ensure that all stakeholders have a clear and shared understanding of the system's capabilities and requirements.

## 2. Overall Description

### 2.1. Product Perspective

From a product perspective, the ride-sharing web app is an online platform that allows passengers to request and pay for rides from drivers using their web browsers. This software is intended to give people and groups with a quick and affordable transportation service, as well as to provide drivers with a persistent source of income. This web app should have a user-friendly interface that allows passengers to easily search for and choose a driver based on their location, vehicle type, and availability, and to receive real-time updates on the status of their ride. This web app should also provide drivers with the tools they need to efficiently accept and manage ride requests, track their earnings, and communicate with passengers.

### 2.2. Specific Requirements

#### a) Interface Requirements

The interface requirements of this project are as follows:

1. **Login/Registration:** Users should be able to create an account and log in to the app using their email address and password.
2. **Profile:** Users should be able to view and edit their profile, including their name, contact information, and payment methods.
3. **Request a ride:** Passengers should be able to enter their pickup and drop-off locations and request a ride. The web app should display the estimated cost and time of the ride.
4. **View available vehicles:** Passengers should be able to view a list of available vehicles in their area, including the type of vehicle, the driver's name and rating, and the estimated time of arrival.
5. **Request a ride:** Once the passenger has selected a vehicle, they should be able to request the ride and view the details of their ride.
6. **Cancel a ride:** Passengers should be able to cancel a ride if they change their mind or their plans change.
7. **Rate a driver:** After a ride is completed, passengers should be able to rate the driver and leave feedback.



8. **Payment:** Passengers should be able pay the fare for their rides through the app digitally.

#### b) Functional Requirements

The functional requirements of this project are as follows:

1. **Geolocation:** The app should be able to determine the passenger's location in order to provide them with nearby ride options and to track the progress of their ride.
2. **Matching:** The app should be able to match riders with nearby drivers based on their pickup and drop-off locations, as well as the type of vehicle they have requested.
3. **Routing:** The app should be able to calculate the optimal route for the driver to take based on real-time traffic conditions and the rider's desired pickup and drop-off locations.
4. **Fare calculation:** The app should be able to calculate the fare for the ride based on the distance traveled, duration of the trip, and any applicable surcharges or discounts.
5. **Payment processing:** The app should be able to process payments from the rider to the driver, including handling any necessary transactions with third-party payment processors.
6. **Driver availability:** The app should be able to track the availability of drivers and display only those who are currently able to accept rides.
7. **Ratings and feedback:** The app should be able to track and display ratings and feedback from riders to help users make informed decisions about which driver to choose.
8. **Vehicle tracking:** The app should be able to track the location of vehicles in real-time to provide accurate estimates of arrival times to riders and to facilitate the routing of rides.

### c) Non-Functional Requirements

The non-functional requirements of this project are as follows:

1. **Performance:** The app should be able to handle high volume traffic and large amounts of data without experiencing delays or performance issues.
2. **Scalability:** The app should be able to scale up or down in response to changes in demand without disrupting service.
3. **Robustness:** The app should be able to handle unexpected events or errors without crashing or losing data.
4. **Security:** The app should protect user data and transactions by implementing secure authentication and encryption methods. This includes protecting against cyber threats such as malware, hacking, and phishing attacks.
5. **Privacy:** The app should respect users' privacy and protect their personal information. This includes complying with relevant laws and regulations such as the General Data Protection Regulation (GDPR).
6. **Usability:** The app should be easy for users to navigate and use, with a clear and intuitive interface. This includes providing clear instructions and error messages to help users understand how to use the app.
7. **Accessibility:** The app should be accessible to users with disabilities and meet relevant accessibility standards, such as the Web Content Accessibility Guidelines (WCAG).
8. **Maintainability:** The app should be easy to maintain and update over time, with a clear and well-organized codebase. This includes providing documentation and support for developers working on the app.

[Back to Previous Content](#)