UNIVERSITI MALAYSIA TERENGGANU
FACULTY OF COMPUTER SCIENCE AND MATHEMATICS

CSA3023
WEB-BASED APPLICATION DEVELOPMENT

System Proposal
E- HAILING SYSTEM

Prepared by:
GROUP 3

Prepared for:
SIR FAKHRUL ADLI BIN MOHD ZAKI

BACHELOR OF COMPUTER SCIENCE WITH MARITIME INFORMATICS
SEMESTER II 2024/2025

Group Members

No	Matric No	Name	Job Description
1	S72248	AINAA MARDHIAH BINTI JABRI	SOFTWARE DEVELOPER
2	S70639	SITI AISYAH BINTI MOHD ZAHIR	OPERATION MANAGER
3	S71451	WAN NUR AIN SHAFIQAH BINTI RANI	BACKEND DEVELOPER

Table of Contents

1. Project Overview

The e-hailing system project is a technology-enabled platform that seeks to connect passengers with drivers through a mobile or web app, making it easy to book transportation services. The system could consist of user and driver applications, backend management dashboard, GPS ride matching, fare estimation, real-time tracking, in-app payments, and rating systems. The goal is to enhance convenience, safety and efficiency in mobility within cities by offering an alternative to traditional taxi services and enabling drivers to earn flexibly. This project involves cross-functional collaboration on software development, UX design, operations and data analysis to enable a seamless transportation experience.

The goal of this project is to create a dynamic, web-based application utilizing the Model-View-Controller (MVC) architecture. Each team member will be in charge of a particular module that handles fundamental Create, Retrieve, Update, and Delete (CRUD) tasks, such as handling user profiles, driver data, ride bookings and payment records.

The project will place a strong emphasis on creating a straightforward but practical system with straightforward database structures, user-friendly interfaces and necessary backend procedures to effectively support the entire e-hailing service.

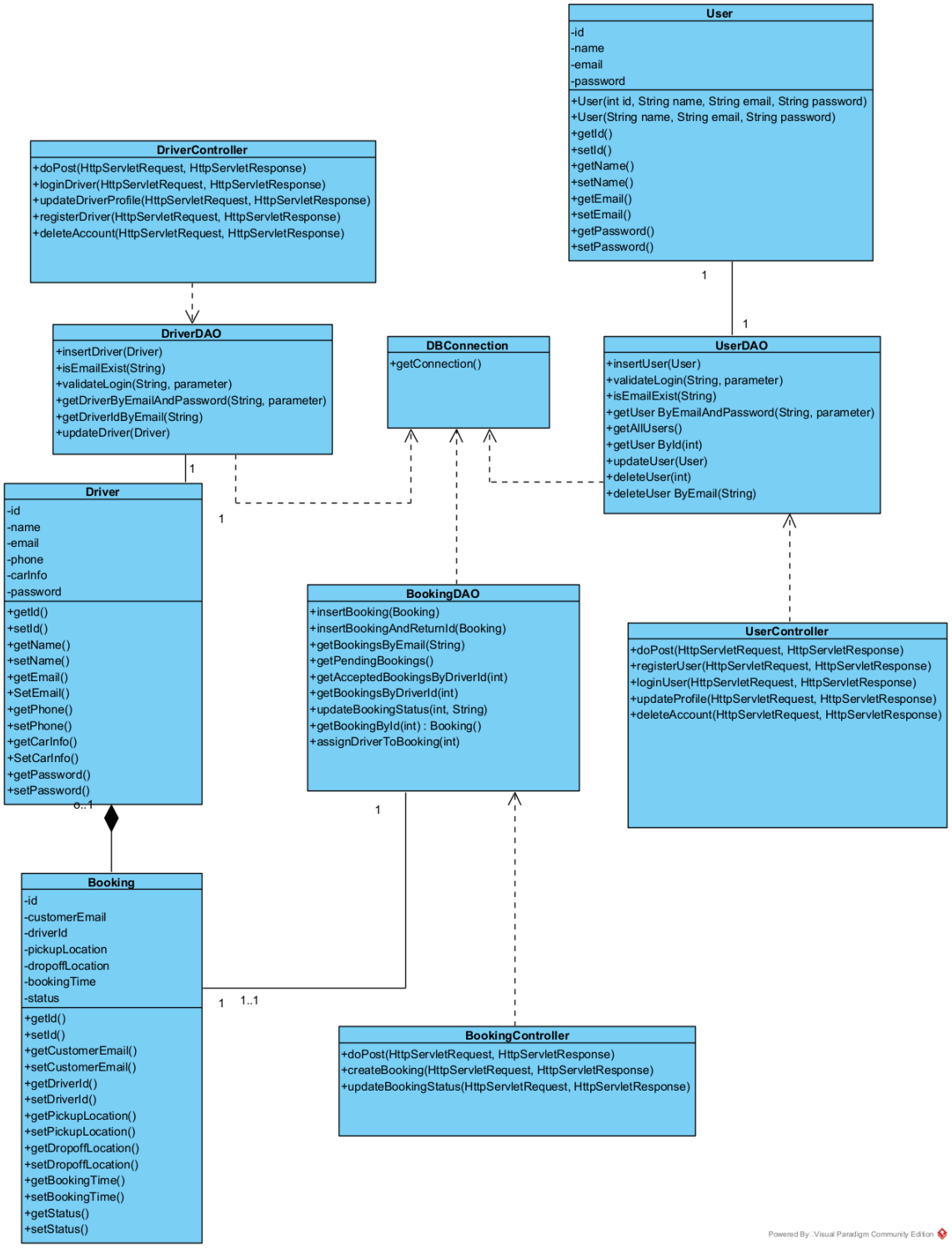
2. Task Distribution

AINAA MARDHIAH BINTI JABRI – Manage User Accounts
(CRUD: Create new users, view user profiles, update user info, delete user accounts)

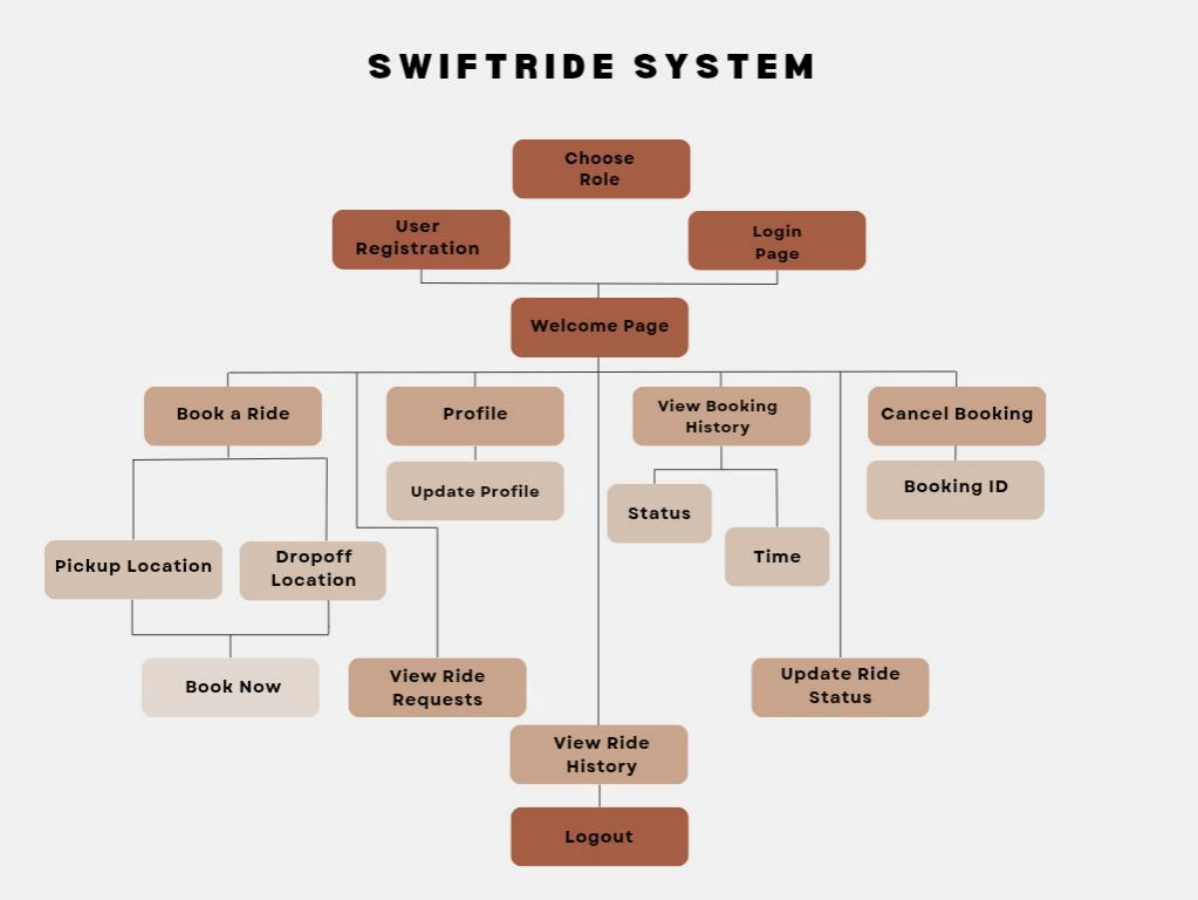
SITI AISYAH BINTI MOHD ZAHIR - Manage Driver Profiles
(CRUD: Create driver profiles, view driver info, update driver status, delete driver accounts)

WAN NUR AIN SHAFIQAH BINTI RANI – Manage Ride Bookings
(CRUD: Create bookings, view booking history, update booking status, cancel bookings)

3. Class Diagrams Related to the Design of Your Web Application



4. Graphical User Interface Design



5. Description of the MVC architecture used in the project

The e-Hailing system is developed using the Model-View-Controller (MVC) architectural pattern, which separates the application into three interconnected components: Model, View, and Controller. This design promotes organized code, ease of maintenance, and scalability.

1. Model

The Model component consists of Java classes that represent the core business data and the logic to interact with the database. It is further divided into:

- **Entity classes (POJOs)**: These are simple Java classes located in the model package, such as User, Driver, and Booking. They represent real-world data structures like user profiles, driver information, and ride bookings.
- **DAO (Data Access Object) classes**: Located in the dao package, these classes handle all database operations (e.g., insert, update, delete, select). Examples include UserDAO, DriverDAO, and BookingDAO.

These model components encapsulate the application's data and business rules.

2. View

The View component comprises JavaServer Pages (JSP) located in the jsp directory. These files are responsible for displaying the user interface and rendering data to users.

Examples of JSP files:

- user_login.jsp
- driver_register.jsp
- book_ride.jsp

These pages use HTML and embedded Java (JSP tags) to present information and forms to users. They do not contain business logic, ensuring separation of concerns.

3. Controller

The Controller components are implemented as Java Servlet classes found in the controller package. Controllers act as intermediaries between the view and the model.

Each controller handles HTTP requests, processes user input, communicates with the DAO classes, and forwards the response to appropriate JSP pages.

Examples:

- UserController.java
- DriverController.java
- BookingController.java

Controllers manage the flow of the application and control what the user sees based on the logic executed.

MVC Workflow in the Project

1. The user interacts with the JSP page (e.g., submits a login form).
2. The Controller servlet receives the request, processes input, and calls the appropriate DAO to retrieve or modify data.
3. The Model (DAO + POJOs) handles database interaction and returns the result.
4. The Controller forwards the processed data to a JSP page for display.
5. The View renders the response to the user.

Conclusion

This MVC architecture ensures a clean separation between data management, user interface, and application logic, making the e-Hailing system easier to develop, test, and maintain.

6. **GitHub Repository**

   https://github.com/Ainaa2003/Group3-e-Hailing.git

7. **Deployment URL**

   http://localhost:8080/ehailing/