st20274562 CIS6003 WRIT1.docx

by Hasini Ravihansi Hewa Nadungodage

Submission date: 14-Mar-2025 07:30AM (UTC+0000)

Submission ID: 252603681

File name:
122083_Hasini_Ravihansi_Hewa_Nadungodage_st20274562_CIS6003_WRIT1_2517131_869413437.docx
(878.51K)

Word count: 1534

Character count: 8867

Table of Contents

Table of Figures

Acknowledgment

The successful completion of this assignment would not have been possible without my effort and the valuable support of many individuals, whose contributions I deeply appreciate and sincerely acknowledge.

First, I would like to express my heartfelt gratitude to Mr. Bhagya Rathnayake, the module lecturer for Advanced Programming, for his continuous guidance, insightful feedback, and encouragement throughout this assignment. His expertise and support have been instrumental in helping me understand key concepts and improve the overall quality of my work.

I would also like to extend my appreciation to my parents for their unwavering support, patience, and encouragement, which kept me motivated during this challenging process. And, to my classmates for their valuable discussions, constructive feedback, and suggestions that helped refine my ideas and enhance my understanding of the subject matter.

Lastly, I would like to acknowledge all those who, in one way or another, contributed to the successful completion of this assignment. Their support, whether direct or indirect, has been immensely valuable, and I am sincerely grateful for their assistance.

Introduction

Mega City Cab is a well-known Cab service operating in Colombo, handling thousands of customer bookings every month. Currently, all bookings, customer details, and fare calculations done manually, which makes the process slow and inefficient. To solve these issues, this project aims to develop a web-based Cab Management System that will allow the company to handle bookings, manage customer and driver records, and calculate fares in an organized and automated way.

The system built using Java and will have a user-friendly menu-driven interface that ensures smooth navigation. Employees will be able to register new customers, record bookings, assign drivers, view booking details and maintain car records**.** The system will also include a secure login feature to ensure only authorized users can access it.

1. Purpose

The primary objective of this Online Cab Service System is to eliminate the inefficiencies associated with manual booking management by providing a digital, user-friendly, and error-free platform for handling cab bookings, driver assignments, and billing. It ensures data security, reliability, and accessibility, enabling the company to improve customer service and operational efficiency.

2. Scope

The Mega City Cab Management System designed to automate the cab booking process, replacing the current manual system. It will allow customers to book rides, drivers to manage their assigned rides, and administrators to oversee operations efficiently.

This will be a web-based system that enables secure user authentication, ride booking, driver and vehicle management, and fare calculation. The system will ensure smooth coordination between customers, drivers, and administrators.

Overall Description

The Mega City Cab Management System is a web-based application developed to replace the current manual process of managing cab bookings. It provides a simple and efficient way for customers, drivers, and administrators to handle cab reservations. Customers can register, make bookings, and check their ride status, while drivers can view assigned bookings, accept or reject rides, and update trip details. Administrators manage customer and driver records, vehicle details, branches, and generate reports on total bookings and sales. The interface designed to be user-friendly, making it easy for all users to navigate and perform their tasks. With this system, Mega City Cab Company can offer a more reliable and organized service, ensuring a smooth experience for both customers and drivers.

Requirement and Features

Functional Requirement

- User Authentication: Customers and Admins must be able to log in securely with a username and password. And, admin should login with his/her given user credentials.
- Customer Booking System: Customers must be able to book a cab by selecting a pickup and drop-off location. Customers should get a confirmation number booking. Customers should be able to view, update, or cancel bookings.
- Vehicle & Driver Management: Admins must be able to add, update, or remove vehicles and view vehicle details in the system.
- Billing & Payment Calculation: The system should calculate the fare based on distance, time, and any additional charges.
- Logout and Session Management: Users should be able to log out securely from the system.

Non-Functional Requirements

- Performance: The system should be responsive and process requests efficiently.
- Security: Passwords should be securely stored and encrypted. Only authorized users (Admin) should be able to access vehicle and driver management.

Task A

UML Diagrams for Mega City Cab System

The Mega City Cab System is design to streamline the cab booking process by automating customer orders, driver management, and billing. To effectively model the system, UML (Unified Modeling Language) diagrams are used to represent different aspects of its structure and functionality. This section presents the Use Case Diagram, Class Diagram, and Sequence Diagram**,** along with detailed explanations of the design decisions.

- **Use Case Diagram**

A Use Case Diagram in UML is a visual representation how users, known as actors, interact with a system. It highlights the system's functional requirements by illustrating the various ways in which different users engage with specific functionalities or use cases. (geeksforgeeks, 2025)

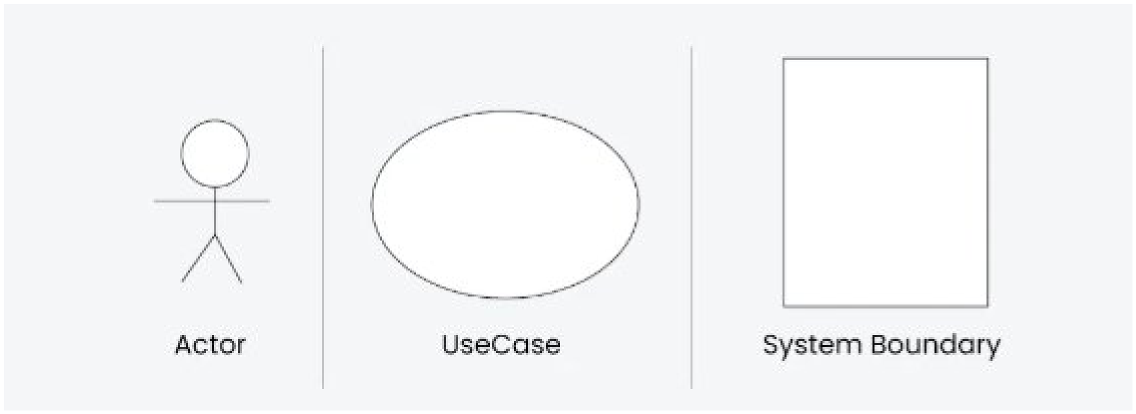


Figure 1: Use Case Diagram Notations

The Use Case Diagram for the Mega City Cab visually represents how different users interact with the system's core functionalities. The system involves three primary actors: Customer, Driver, and Administrator**,** each performing distinct roles.

Actors:

1. Customer - Registers, logs in, books a ride, views booking details, makes payments.
2. Driver - Logs in, views available bookings, accepts/rejects rides, updates ride status.
3. Admin - Manages customers, drivers, vehicles, vehicle category and booking records.

Use Cases:

Customer Use Cases:

1. Login – Customer logs in.
2. Register - Customer creates an account and logs in.
3. Make Payment – Pays after ride completion.
4. View Booking Details – Checks the status of ongoing and completed bookings.

Driver Use Cases:

1. Login – Authenticates into the system.
2. Accept/Reject Ride – Chooses to take or decline a ride.
3. Update Ride Status – Marks ride as "Picked Up" or "Completed."
4. View Available Bookings – Sees pending ride requests.

Admin Use Cases:

1. Login – Admin logs in.
2. View & Manage Bookings – Monitors all ride requests and statuses.
3. Manage Drivers, vehicles and vehicle category - Adds, removes, or modifies

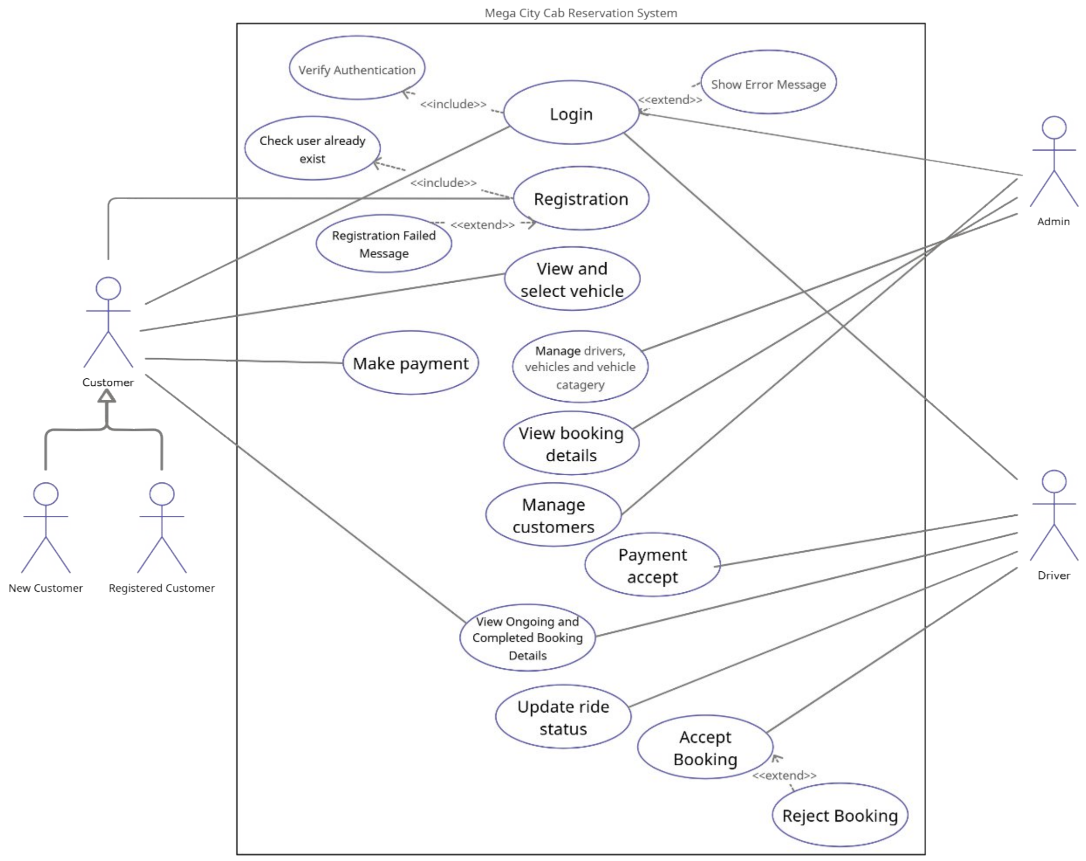


Figure 2: Use Case Diagram of Mega City Cab

The diagram shows how different actors interact with the Online Vehicle Reservation System. The three primary user roles are Customer, Driver, and Admin and each of have distinct functionalities mapped out in the diagram. The Customer begins by registering and logging in. Verify Authentication is an included use case within the Login use case. Every time a Customer or Driver attempts to log in, the system will verify their credentials before granting access. The system verifies authentication and checks for existing users. If an error occurs, a failure message displayed. After successful authentication, the customer can view and select a vehicle, proceed with booking, check booking details, and make payments. The Admin has a broader scope, managing customers, vehicles, and bookings. They oversee payments and system operations, ensuring smooth functionality.

- **Class Diagram**

A class diagram is a visual tool that represents the structure of a system by showing its classes, attributes, methods, and the relationships between them. It helps everyone involved in a project like developers and designers to understand how the system is organized and how its components interact with each one.

In object-oriented programming (OOP), a class is a blueprint or template for creating objects. Objects are instances of classes, and each class defines a set of attributes (data members) and methods (functions or procedures) that the objects created from that class will possess. The attributes represent the characteristics or properties of the object, while the methods define the behaviors or actions that the object can perform. (visual-paradigm, 2025)

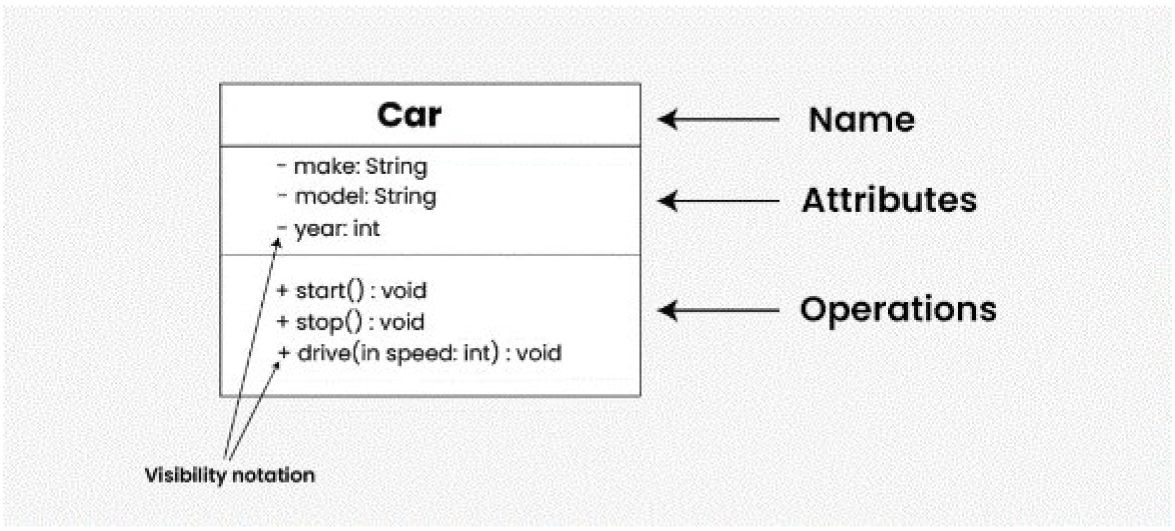


Figure 3: Class Notation

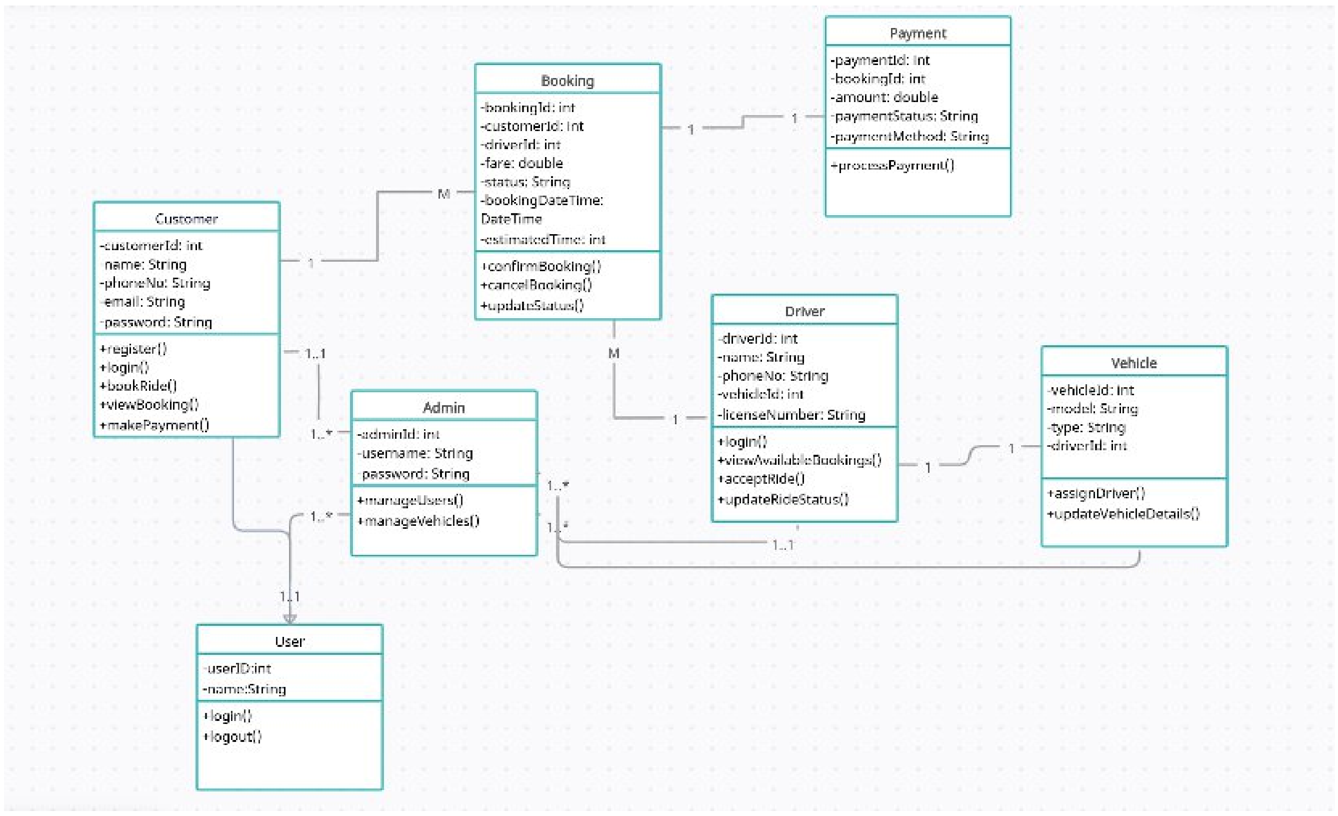


Figure 4: Class Diagram of Mega City Cab

Sequence Diagram

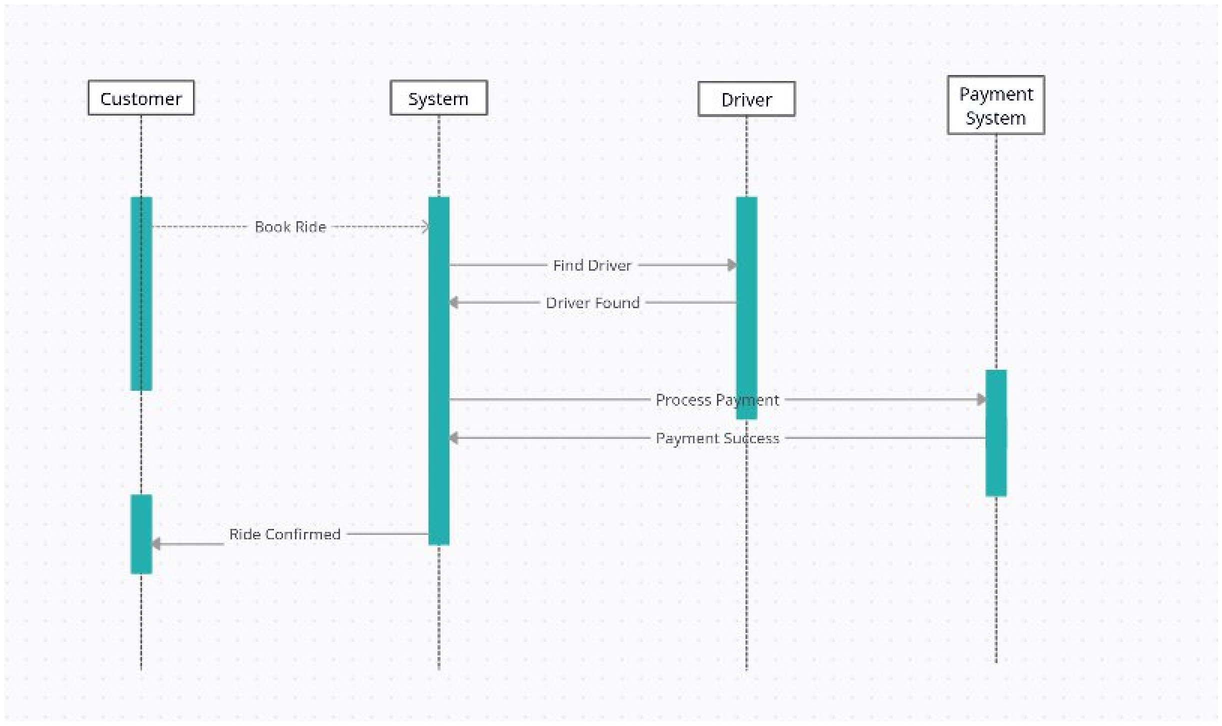


Figure 5: Sequence Diagram for

Task B

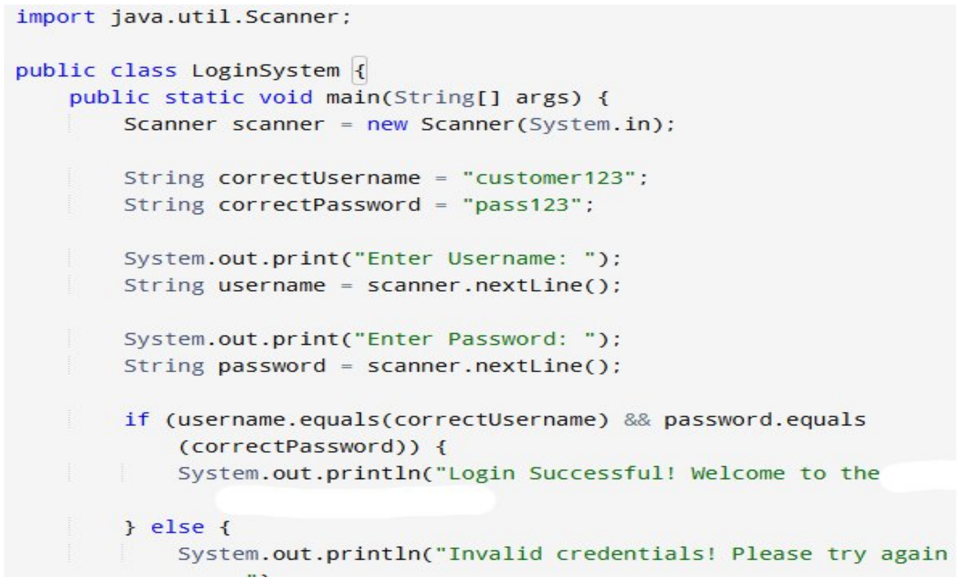
<u>System Implementation</u>

The Online Cab Booking System designed as a web-based application where customers can book rides and admins can manage vehicles and drivers. The system built using Java for backend processing, handling all business logic and user requests. The frontend consists of HTML, CSS, and JavaScript, ensuring a simple and user-friendly interface. The system follows a structured approach, separating the user interface, business logic, and data management. When a customer books a ride, the request processed by the backend, which checks for available vehicles, calculates the fare, and assigns a driver. The system then stores the booking details, ensuring that both the customer and the admin can access the information when needed. The admin has control over vehicle management, allowing them to add, remove, or update available cars and assign drivers accordingly.

User Interface Development

The system features a simple yet effective interface that allows users to interact seamlessly. The login page ensures secure access for both customers and admins, preventing unauthorized users from making changes to the system. Once logged in, customers presented with a booking page where they can select their pickup and drop-off locations, choose a preferred vehicle, and confirm their ride.

The admin dashboard provides an overview of active bookings, available vehicles, and registered drivers. Admins can manage vehicles and driver assignments through an intuitive panel that allows them to update information as needed. The system also includes a billing page where customers can review their total fare before confirming the ride.

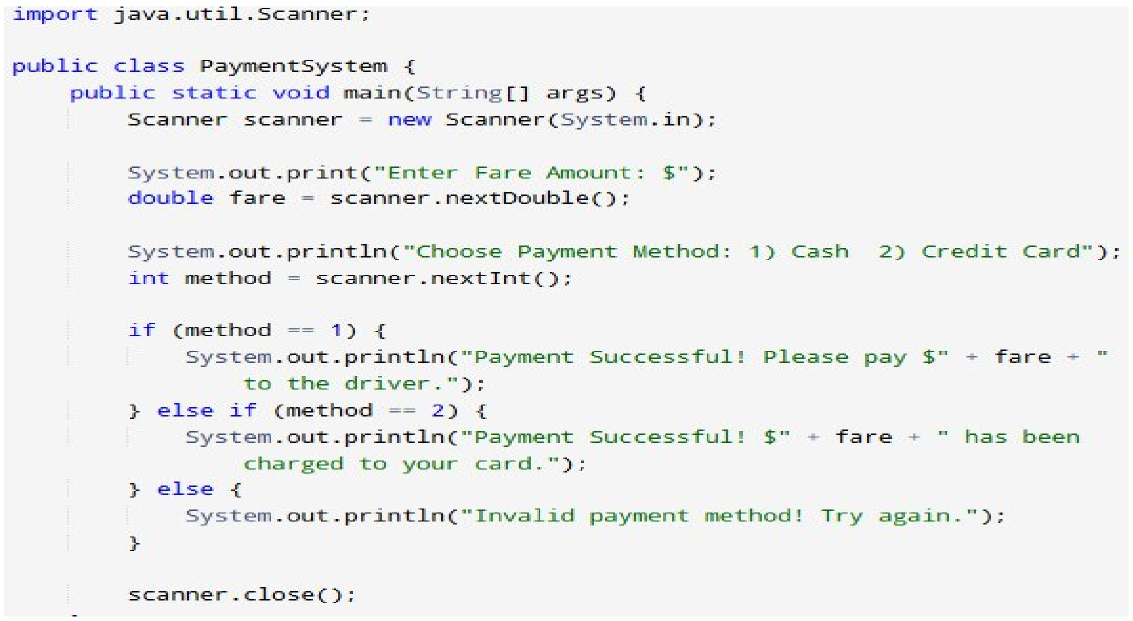


```
import java.util.Scanner;

public class LoginSystem {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        String correctUsername = "customer123";
        String correctPassword = "pass123";

        System.out.print("Enter Username: ");
        String username = scanner.nextLine();

        System.out.print("Enter Password: ");
        String password = scanner.nextLine();

        if (username.equals(correctUsername) && password.equals
            (correctPassword)) {
            System.out.println("Login Successful! Welcome to the

        } else {
            System.out.println("Invalid credentials! Please try again
```

Figure 6: Login Page

Backend Implementation

The backend developed in Java, ensuring efficient processing of all user interactions. The system built using Object-Oriented Programming principles, with separate classes for managing users, bookings, vehicles, and payments. Each class is responsible for specific tasks, allowing the system to remain modular and maintainable. When a customer books a ride, the booking class processes the request by checking vehicle availability, calculating the fare, and assigning a driver. The system ensures that the information is stored correctly and updates the status of the booking accordingly. Admin functionalities handled through dedicated classes that manage vehicles, drivers, and reports.

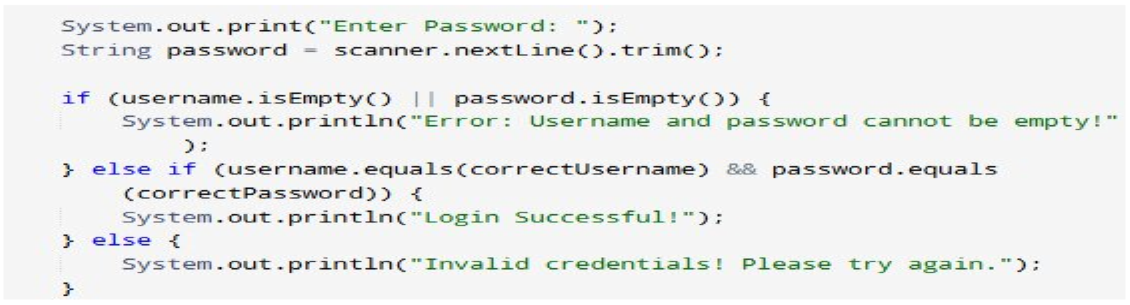


```
import java.util.Scanner;

public class PaymentSystem {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter Fare Amount: $");
        double fare = scanner.nextDouble();

        System.out.println("Choose Payment Method: 1) Cash  2) Credit Card");
        int method = scanner.nextInt();

        if (method == 1) {
            System.out.println("Payment Successful! Please pay $" + fare + "
                to the driver.");
        } else if (method == 2) {
            System.out.println("Payment Successful! $" + fare + " has been
                charged to your card.");
        } else {
            System.out.println("Invalid payment method! Try again.");
        }

        scanner.close();
```

Figure 7: Payment Calculation

Error Handling and Validation

To ensure a smooth user experience, the system incorporates error handling and validation at multiple levels. Input validation prevents incorrect or incomplete data from being processed. For instance, during login, the system checks whether the username and password fields are empty before verifying the credentials. Similarly, when booking a ride, the system ensures that pickup and drop-off locations are not the same, avoiding invalid bookings. Security measures also implemented, ensuring that unauthorized users cannot access admin functionalities. Passwords validated correctly, and session management handled securely to prevent unauthorized access after logout. These measures contribute to making the system more robust and reliable.



```
System.out.print("Enter Password: ");
String password = scanner.nextLine().trim();

if (username.isEmpty() || password.isEmpty()) {
    System.out.println("Error: Username and password cannot be empty!"
        );
} else if (username.equals(correctUsername) && password.equals
    (correctPassword)) {
    System.out.println("Login Successful!");
} else {
    System.out.println("Invalid credentials! Please try again.");
}
```

Figure 8: Input Validation & Error Handling

<u>Designing Pattern</u>

The designing pattern is way of technique which provides solutions to the issues or prevalent error in the process of software development. Designing pattern helps the software developer or people who works on software development fields to evaluate their way of coding in effective and efficient way. The designing pattern gives clear view of the ways which coding or template of coding to tackle the problems that comes in development phase. In developing environment designing pattern coverup all the common issues which will occur while developing a software.

<u>What is Software development Designing pattern?</u>

Reusable solutions for typical software design challenges are known as design patterns. Expert object-oriented software engineers use these best practices to write more structured, manageable, and scalable code. Design patterns provide a standard terminology and are specific to particular scenarios and problems. Design patterns are not finished code but templates or blueprints only.

Design patterns can be further categorized into three main groups: creation, structural, and behavioral.

1. Creation patterns

Creation design patterns are the oldest and most well-known. They include patterns like the factory method, the observer pattern, and the singleton pattern. These patterns are often used to solve fundamental design problems, such as how to create objects, how to communicate between objects, and how to manage object lifetimes.

2. Structural patterns

Structural design patterns are used to create larger, more complex systems. They include patterns like the composite pattern, the decorator pattern, and the facade pattern. These patterns are often used to organize code, to make it more modular and reusable, and to hide internal complexity.

3. Behavioural patterns

Behavioural design patterns are used to define interactions between objects. They include patterns like the strategy pattern, the template method pattern, and the state pattern. These patterns are often used to encapsulate logic, to make code more flexible, and to make it easier to change behavior without changing code.

Software Design Patterns used

- Singleton Pattern Is used for Database Connection and it ensures only one instance of the database connection is created. And it helps prevents multiple unnecessary database connections and improving performance.

Task C

Software testing is a crucial phase in the development process that ensures the system functions as expected, meets requirements, and is free of defects. It involves systematically evaluating the software through different testing techniques to identify and fix potential errors before deployment. In this project, testing is essential to verify that the Mega City Cab Booking System operates correctly, handles user inputs properly, and maintains data integrity. By using a structured test plan, the system's functionalities including login authentication, booking management, billing, and data storage tested.

Testing types

- Testing of unit
- Testing of system
- Testing of integration
- Testing of user acceptance
- Testing of black box
- Testing of white box

Test Driven Development

Test-driven development (TDD) is a process on software development which uses to test application to get error free application or to minimize the test coverage from the start. The Test-Driven Development generate test for every function which are executed in application, if automated test gives fail or error, developer should be inform to change the failed function or write the code again to prevent any kind of fail.

- Run all the test cases and make sure that the new test case fails.
- **Red –** Create a test case and make it fail, Run the test cases
- **Green –** Make the test case pass by any means.
- **Refactor –** Change the code to remove duplicate/redundancy.Refactor code – This is done to remove duplication of code.
- Repeat the above-mentioned steps again and again. This ensures that every piece of functionality tested before it is implemented.

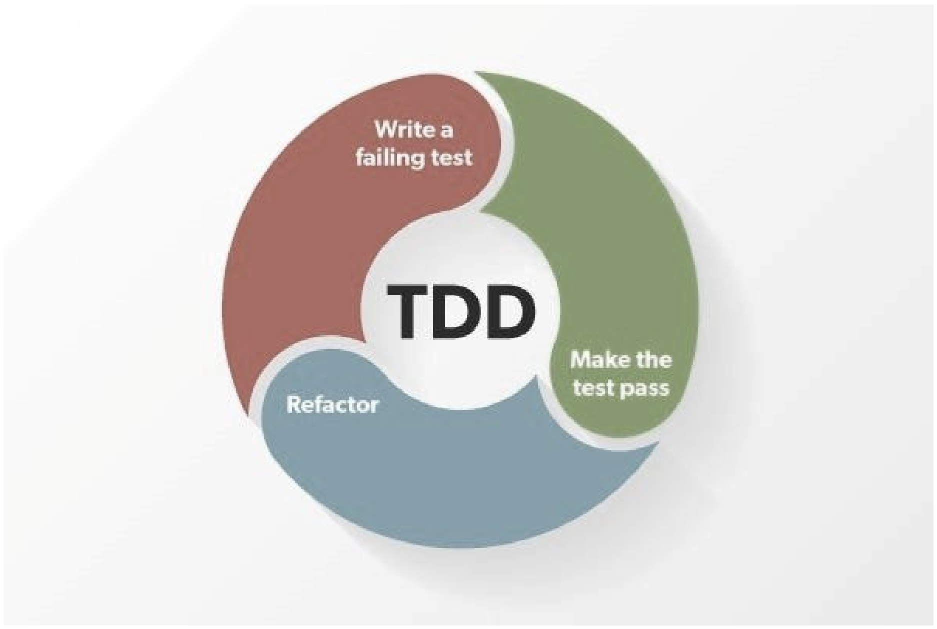


Figure 9: Test Driven Development (TDD)

Steps Followed in TDD for the System

- Step 1: Write a test case for a function before writing its implementation.
- Step 2: Run the test and confirm it fails since the function not implemented yet.
- Step 3: Implement the function with minimal code to pass the test.
- Step 4: Run the test again and confirm it passes.
- Step 5: Refactor the function while ensuring the test still passes.

How do the Test-Driven Development process conduct in Mega City Cab Booking Service?

User Authentication before implementing login and registration, test cases written to check:

- If a user can register successfully.
- If a user can log in with correct credentials.
- If a user cannot log in with incorrect credentials.

Payment Processing, before implementing payment logic, tests checked:

- If a payment is processed correctly.
- If invalid payments are rejected.
- If a receipt is generated after a successful payment.

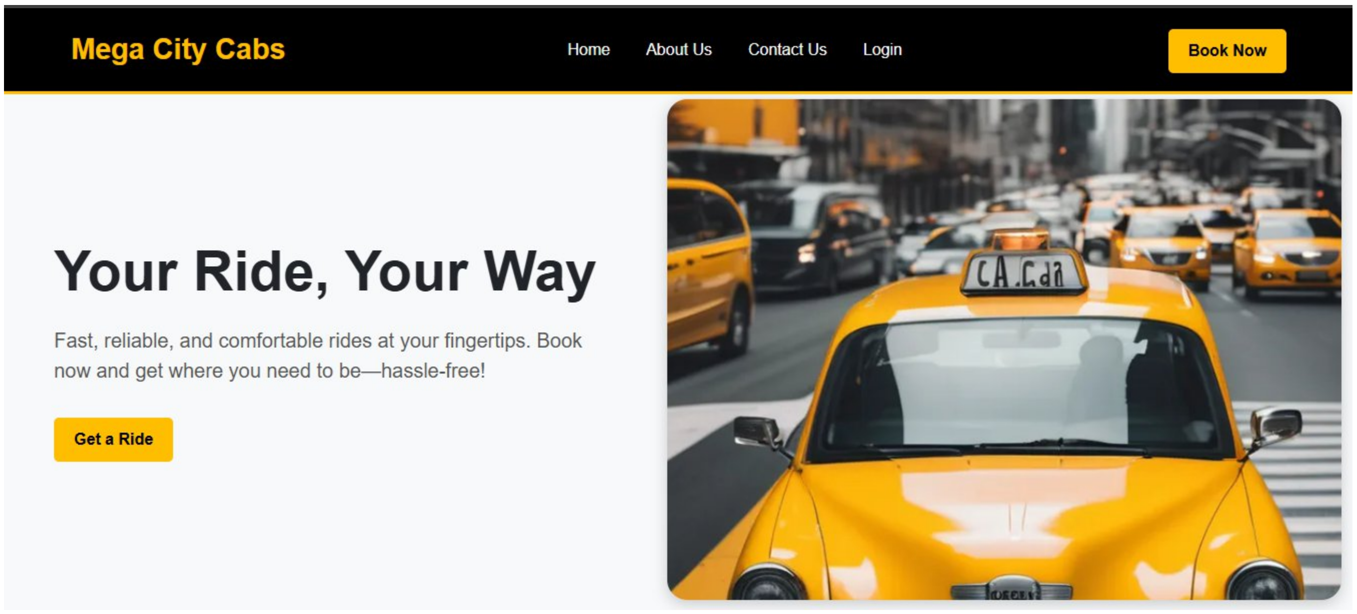
Test Case

Test Case	Test Scenario	Expected Results	Actual Resulted	Status
Customer Registration	Fill the fields and click on the “Register” button	Should register and redirect to customer login page	As expected, the customer registered to system and redirected to login	Pass
	without fill the fields	Should show the error message to fill data for relevant field	As expected, the error message has been displayed	Pass
Customer Login	User logs in successfully with valid username & password	Login is successful	As expected, login is successful	Pass
	Fill incorrect login details	Display error message “User Credential incorrect” in login page	As expected, displayed the error message	Pass
Customer Cab Booking	Select pickup location, drop location, vehicle type and date field then click on book now button	Booking is confirmed	As expected, customer booking is confirmed	Pass
Driver Booking	Shows the driver the booking list of customers	Display booking of customer	As expected, booking list is displayed	Pass
	Click the Accept button	Redirect to driver dashboard page	As expected, booking has been accepted redirected and to driver-dashboard	Pass
	Click the Reject button	-	-	Fail

Test Case	Test Scenario	Expected Results	Actual Resulted	Status
Driver Login	Fill username password and click login	Login is successful and redirect to driver dashboard page	As expected, Login is successful	Pass
	Fill incorrect Username password and login	Display error message “User Credential incorrect” in login page	As expected, the error message displayed	Pass
Administrator Login	Fill the correct username, password and click login	Login is successful and redirect to	As expected, login is successful and redirected to	Pass
	Fill incorrect username, password and click on login	Display error message “User Credential incorrect” in login page	As expected, the error message displayed	Pass
Booking Details	Go to booking details	Display all the booking completed history	As expected, all the booking history completed	Pass
Manage Driver	Fill the fields on add Driver page and click on the “Add” button	add Driver details and redirect to the Driver list page.	As expected, Driver data has been added	Pass
	Fill the fields on update Driver page with relevant data	Data should be filled to relevant fields in update	As expected, data are filled relevant field	Pass
Add Vehicle Category	Fill the fields on add Vehicle Category page	add Vehicle Category details	As expected, data has been added	Pass

Technical Documentation

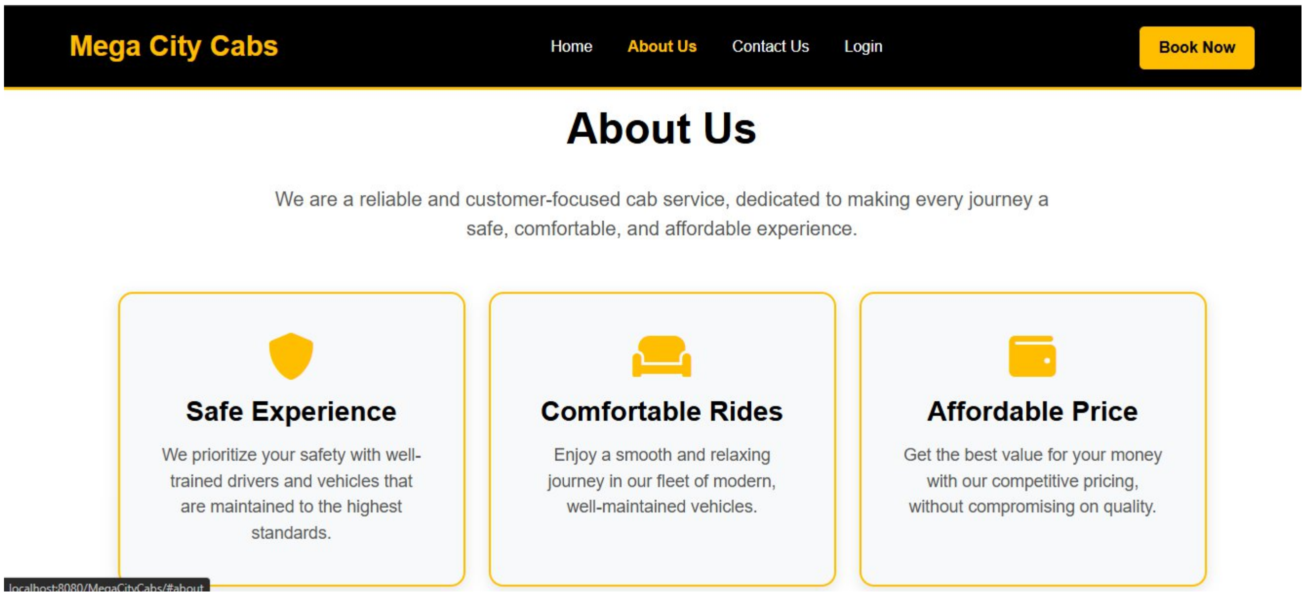
Homepage



Your Ride, Your Way

Fast, reliable, and comfortable rides at your fingertips. Book now and get where you need to be—hassle-free!

About Us



About Us

We are a reliable and customer-focused cab service, dedicated to making every journey a safe, comfortable, and affordable experience.

Safe Experience

We prioritize your safety with well-trained drivers and vehicles that are maintained to the highest standards.

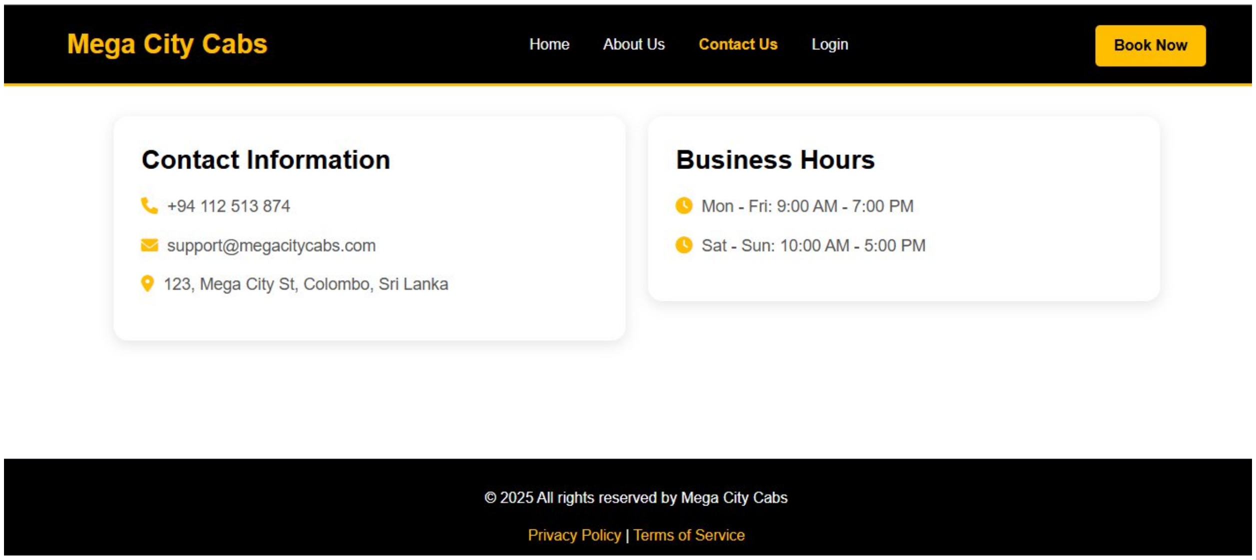
Comfortable Rides

Enjoy a smooth and relaxing journey in our fleet of modern, well-maintained vehicles.

Affordable Price

Get the best value for your money with our competitive pricing, without compromising on quality.

Contact Information & Footer



Mega City Cabs

Home About Us Contact Us Login

Book Now

Contact Information

+94 112 513 874

support@megacitycabs.com

123, Mega City St, Colombo, Sri Lanka

Business Hours

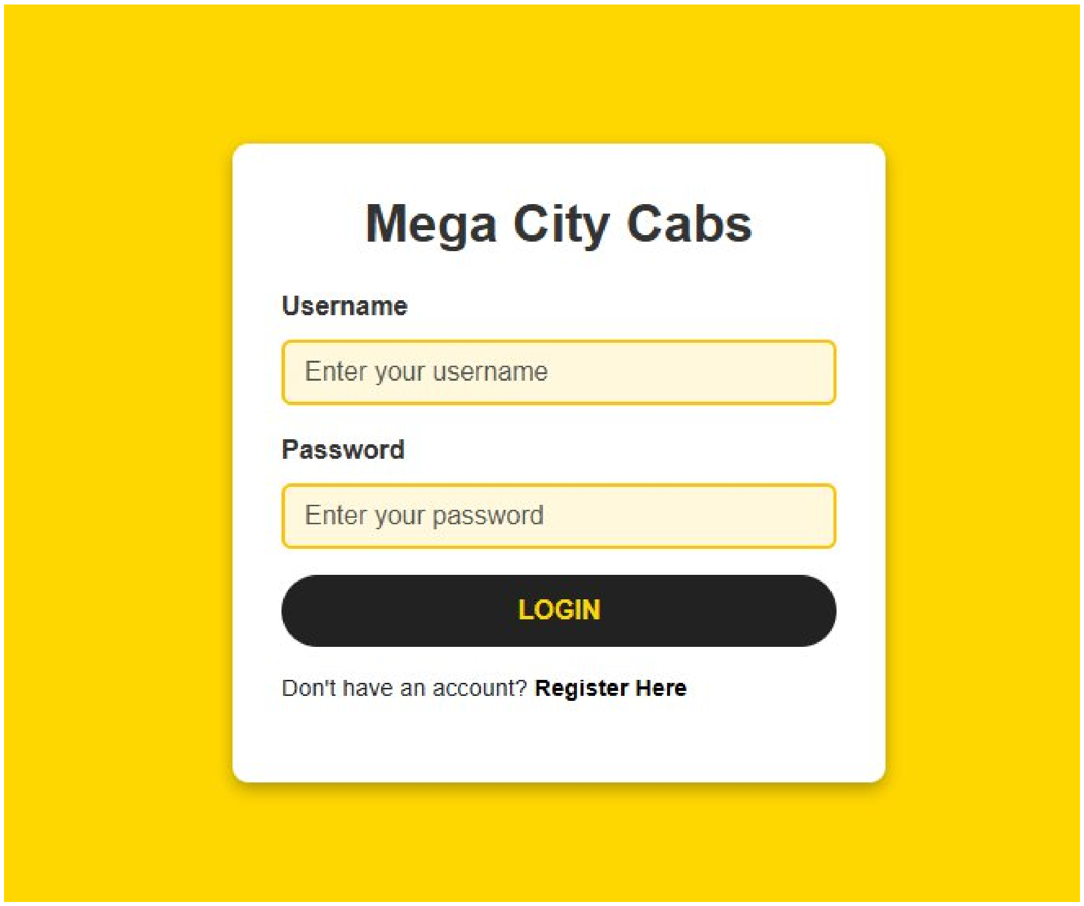
Mon - Fri: 9:00 AM - 7:00 PM

Sat - Sun: 10:00 AM - 5:00 PM

© 2025 All rights reserved by Mega City Cabs

Privacy Policy | Terms of Service

Customer Login



Mega City Cabs

Username

Enter your username

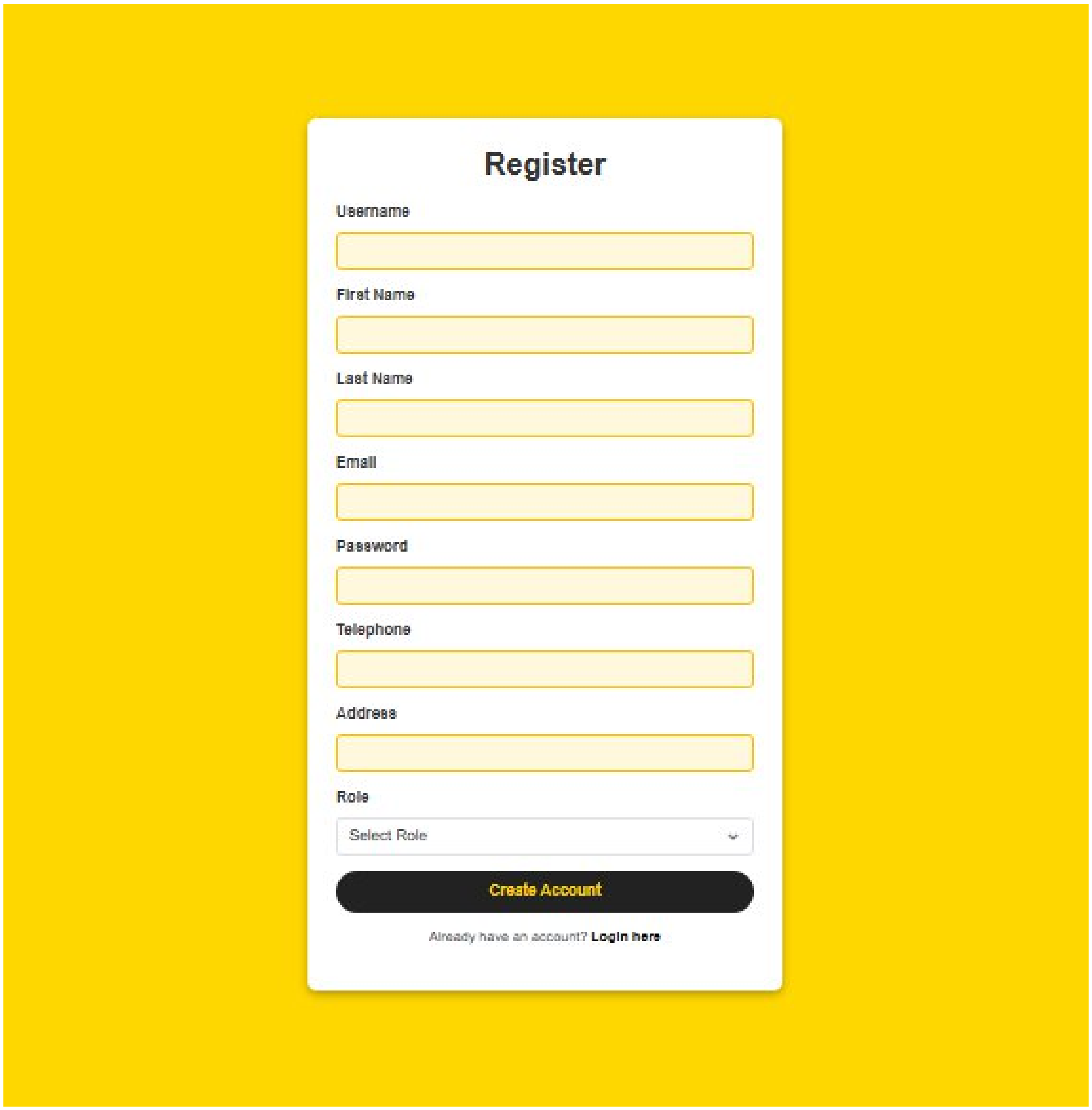
Password

Enter your password

LOGIN

Don't have an account? **Register Here**

New Customer Registration



Register

Username

First Name

Last Name

Email

Password

Telephone

Address

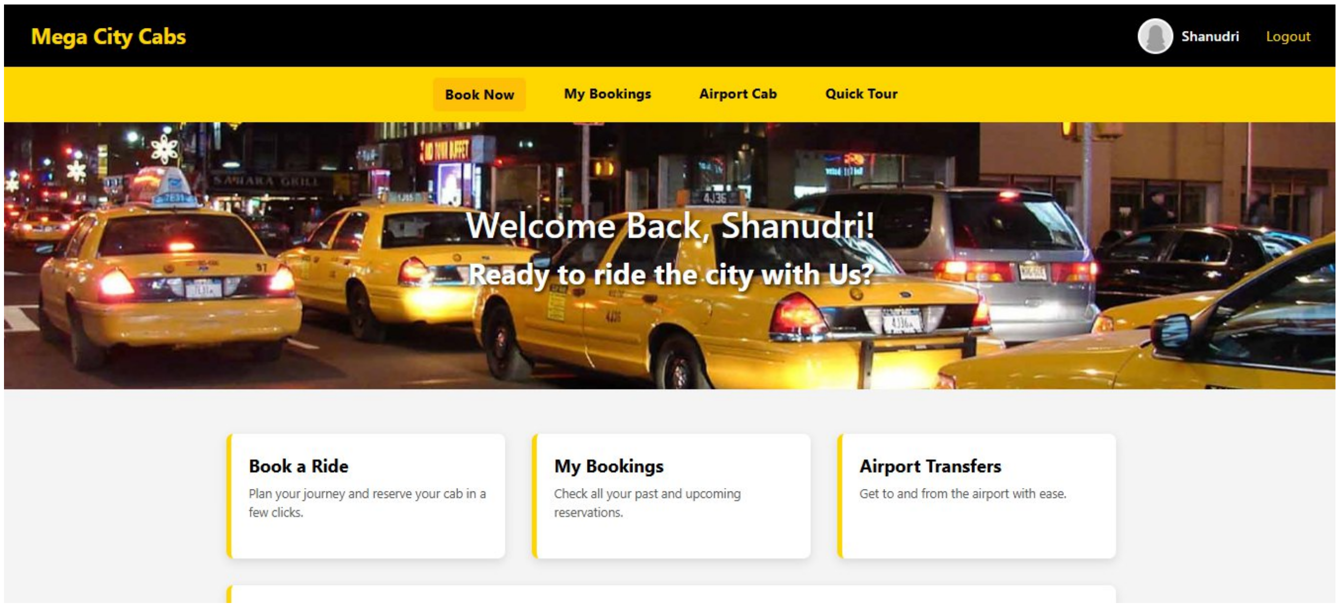
Role

Select Role

Create Account

Already have an account? **Login here**

Customer Homepage (Dashboard)



Welcome Back, Shanudri!

Ready to ride the city with Us?

Book a Ride

Plan your journey and reserve your cab in a few clicks.

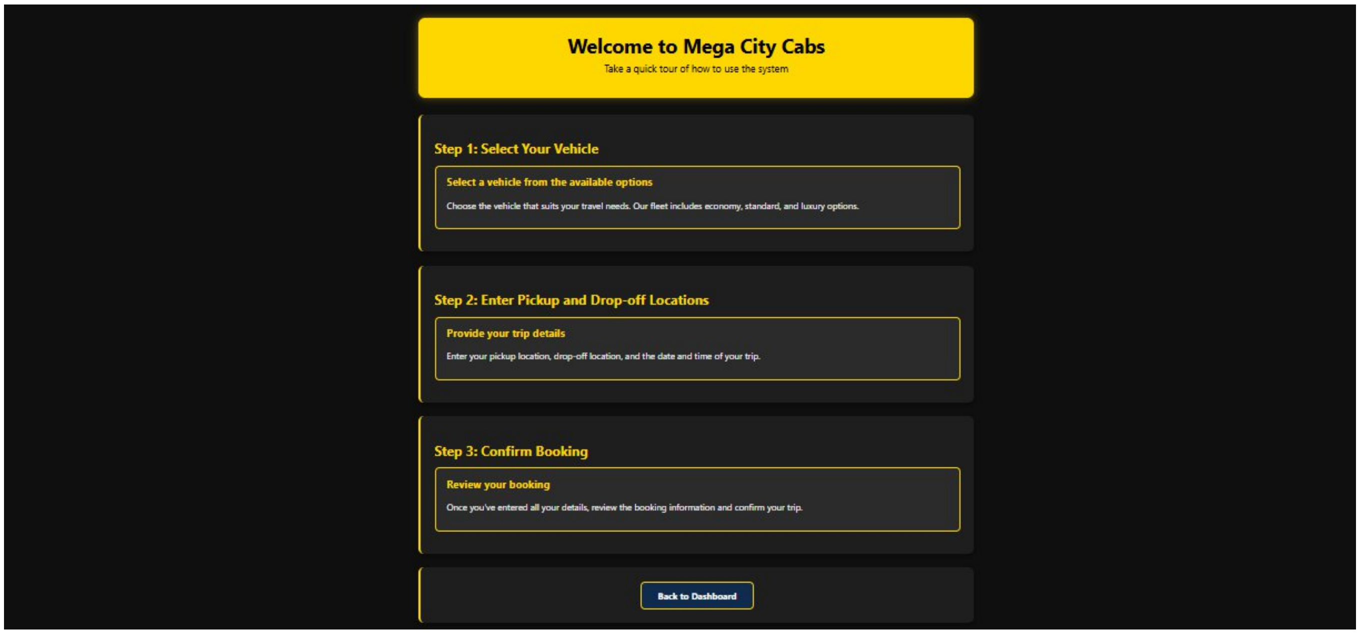
My Bookings

Check all your past and upcoming reservations.

Airport Transfers

Get to and from the airport with ease.

Quick Guidance for New Customers



Welcome to Mega City Cabs

Take a quick tour of how to use the system

Step 1: Select Your Vehicle

Select a vehicle from the available options

Choose the vehicle that suits your travel needs. Our fleet includes economy, standard, and luxury options.

Step 2: Enter Pickup and Drop-off Locations

Provide your trip details

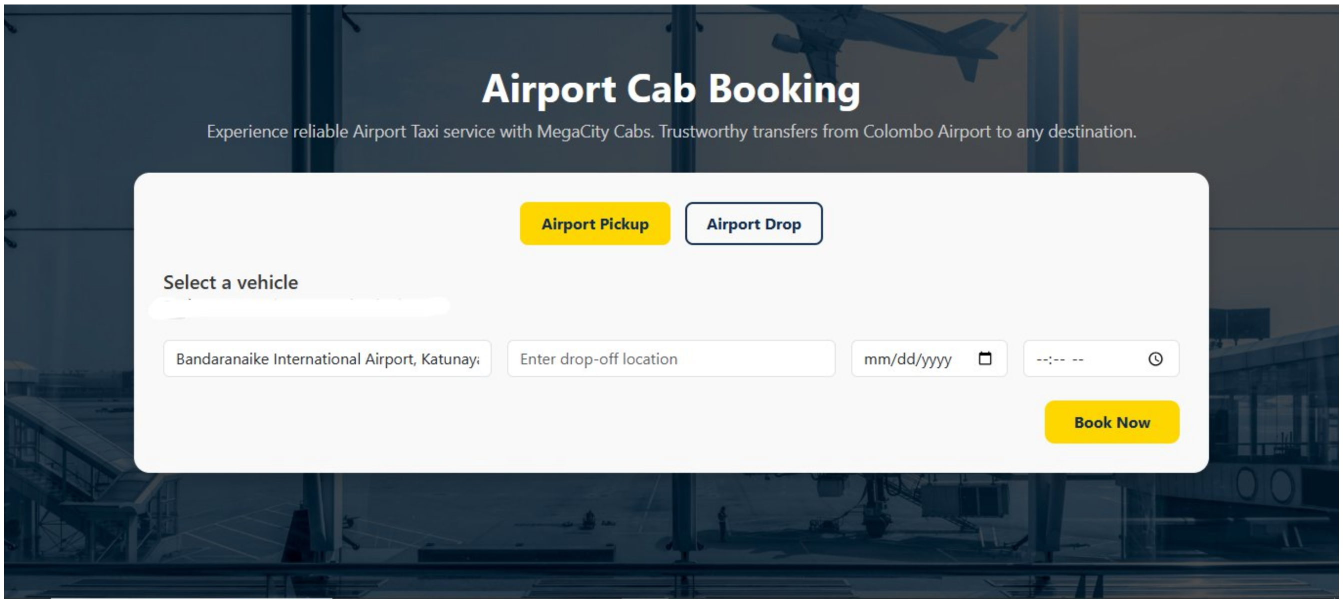
Enter your pickup location, drop-off location, and the date and time of your trip.

Step 3: Confirm Booking

Review your booking

Once you've entered all your details, review the booking information and confirm your trip.

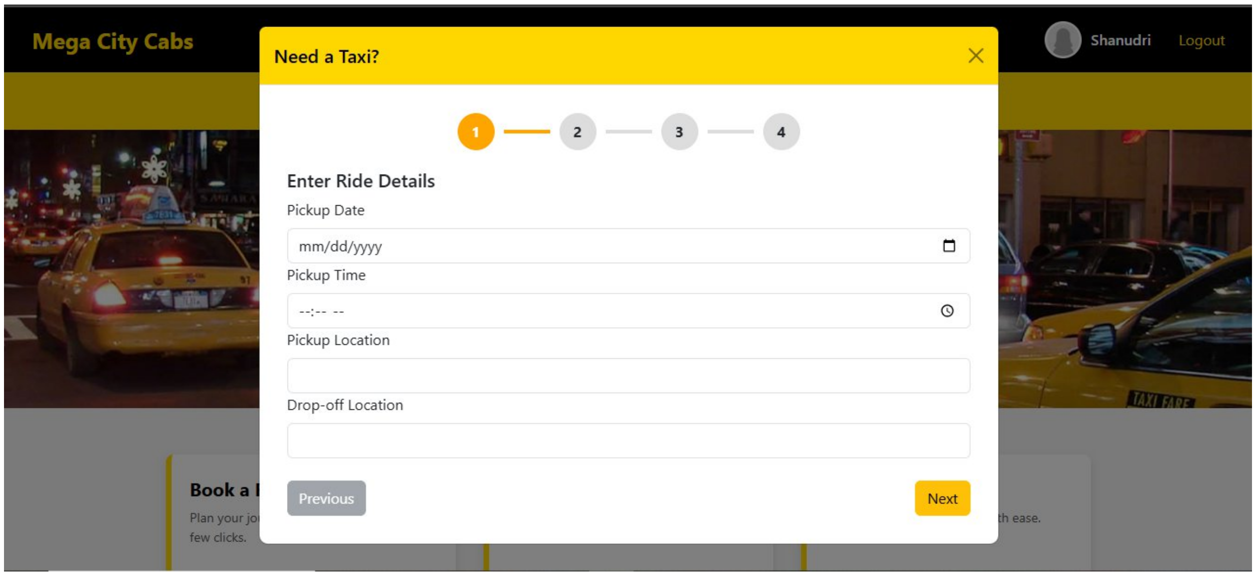
Airport Cab Service



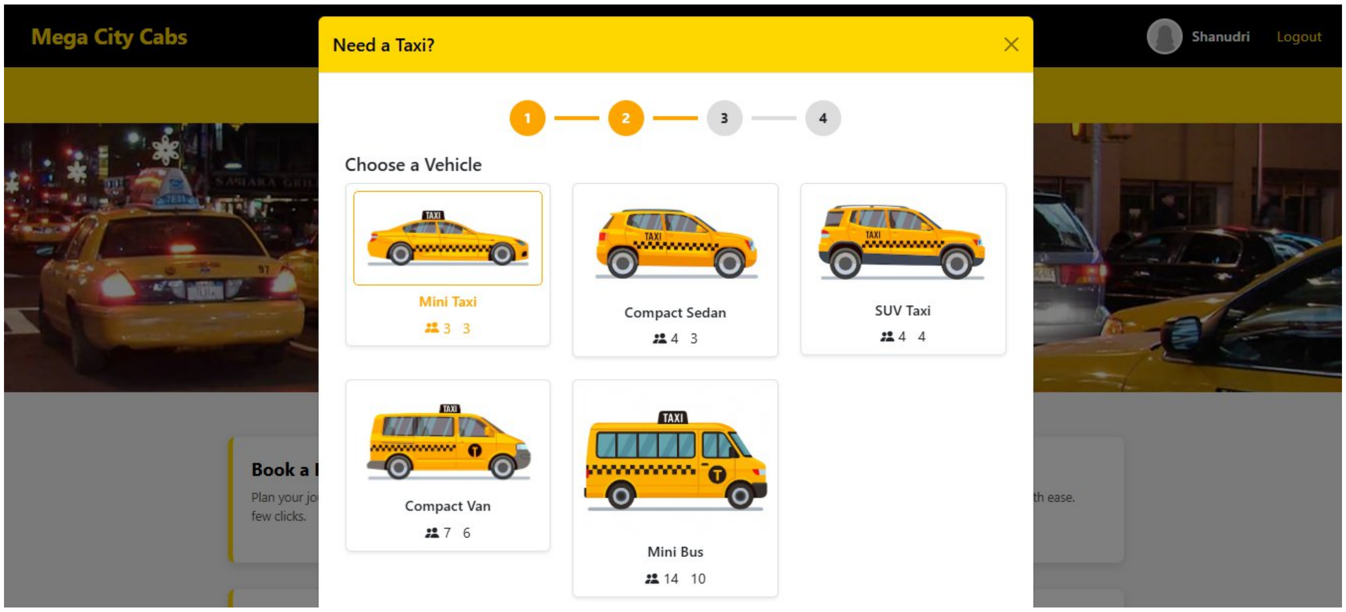
Airport Cab Booking
Experience reliable Airport Taxi service with MegaCity Cabs. Trustworthy transfers from Colombo Airport to any destination.
Airport Pickup
Airport Drop
Select a vehicle
Bandaranaike International Airport, Katunaya
Enter drop-off location
mm/dd/yyyy
--:-- --
Book Now

Customer Cab Booking

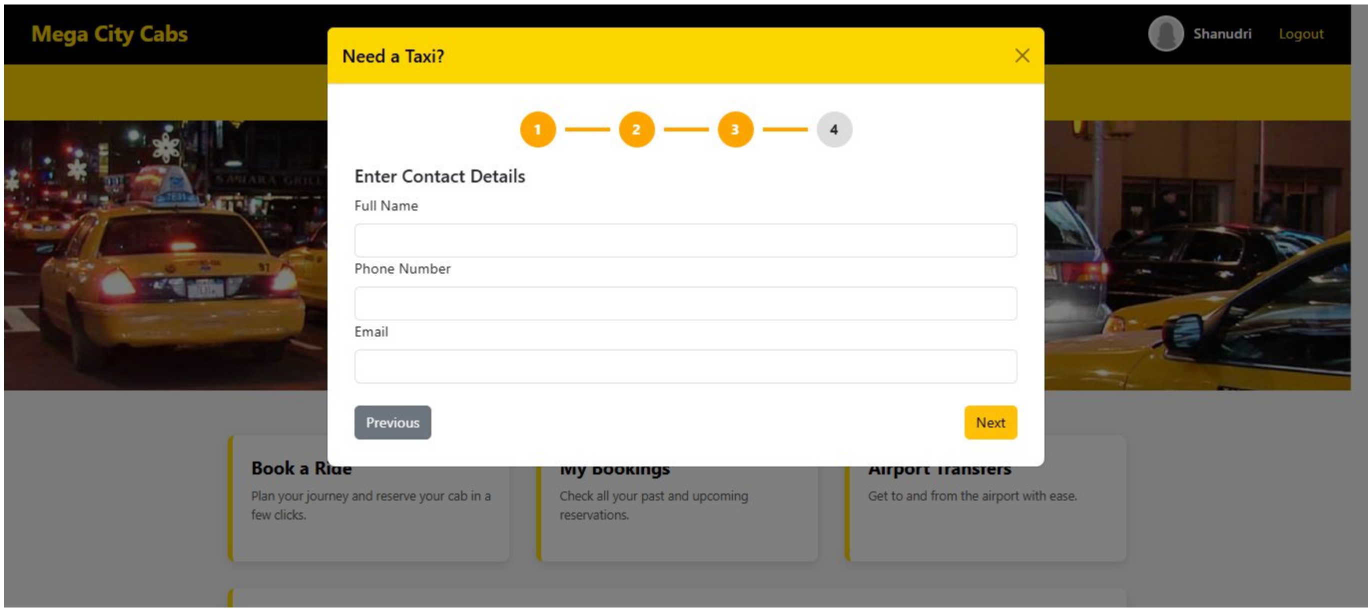
First step: Entering Ride details



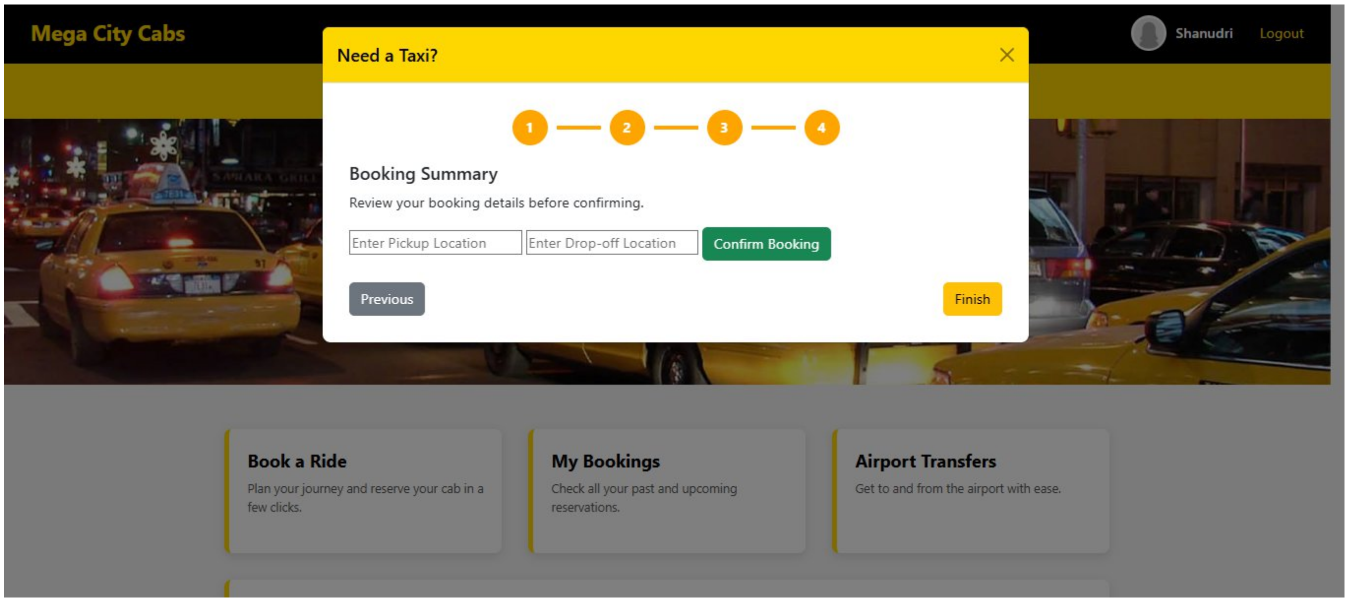
Second Step: Choosing a suitable cab



Third Step: Entering Customer Contact details



Last Step: Confirm the Booking entering pickup location and drop-off location.



Booking Summary

Review your booking details before confirming.

Book a Ride

Plan your journey and reserve your cab in a few clicks.

My Bookings

Check all your past and upcoming reservations.

Airport Transfers

Get to and from the airport with ease.

Task D

<u>GitHub Repositories</u>

Link for the Mega City Cab reservation system:

https://github.com/HRavihansii/CabReservationSystem.git

Conclusion

In conclusion, the Mega City Cab system project has successfully covered all key requirements from system design to implementation and testing. The detailed UML diagrams, including the use case, class, and sequence diagrams, have laid a strong foundation for understanding the system's architecture and functionality. With the interactive Java-based application developed, users can securely authenticate, manage customer bookings, calculate bills, and perform other necessary operations. Through careful validation and error-checking mechanisms, the system ensures smooth interactions for all users.

References

Browserstack, 2024. *browserstack.* [Online]
Available at: https://www.browserstack.com/guide/what-is-test-driven-development
[Accessed 25 2 2025].

creately, 2025. *creately.* [Online]
Available at: https://app.creately.com/d/start/dashboard

geeksforgeeks, 2025. *geeksforgeeks.* [Online]
Available at: https://www.geeksforgeeks.org/use-case-diagram/
[Accessed 23 February 2025].

lucid.app, 2025. *lucid.app.* [Online]
Available at: https://lucid.app/lucidchart/056aadb8-840b-403d-bbf9-0ecb2eee1704/edit?beaconFlowId=ECD3333C73BCCCEA&page=0_0#

Steinfeld, G., 2024. *IBM.* [Online]
Available at: https://developer.ibm.com/articles/5-steps-of-test-driven-development/
[Accessed 2025].

visual-paradigm, 2025. *visual-paradigm.* [Online]
Available at: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/
[Accessed 2025].

st20274562 CIS6003 WRIT1.docx

ORIGINALITY REPORT

8%	4%	0%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to University of Wales Institute, Cardiff Student Paper	3%
2	www.softwareadvice.com Internet Source	1%
3	Submitted to American InterContinental University Student Paper	1%
4	Submitted to University of Bedfordshire Student Paper	1%
5	pubmed.ncbi.nlm.nih.gov Internet Source	1%
6	businessdocbox.com Internet Source	<1%
7	www.coursehero.com Internet Source	<1%
8	Kennis Chan. "Testing and Measurement: Techniques and Applications - Proceedings of the 2015 International Conference on Testing and Measurement Techniques (TMTA 2015), 16-17 January 2015, Phuket Island, Thailand", CRC Press, 2015 Publication	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		