A close-up of a assessment brief

AI-generated content may be incorrect.

A close-up of a document

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.

**Assignment Cover Sheet**

|  |  |  |
| --- | --- | --- |
| **Qualification** | | **Module Number and Title** |
| Top up - BSc in Software Engineering (CMU) | | CIS6003 Advanced Programming |
| **Student Name & No.** | | **Assessor** |
| R.G Pramod Sandakelum  Registration Number GM/BSCSD/04/08  University Number ST20261236 | | Mrs. Vijini Mekala |
| **Hand out date** | | **Submission Date** |
|  | | 14.03.2025 – before 2.00pm |
| **Assessment type**  WRIT1-Coursework | **Duration/Length of**  **Assessment Type** | **Weighting of Assessment**  100% |

|  |  |
| --- | --- |
| **Learner declaration** | |
| I, R.G Pramod Sandakelum GM/BSCSD/04/08 - ST20261236 certify that the work submitted for this assignment is my own and research sources are fully acknowledged. | |
| |  |  |  |  | | --- | --- | --- | --- | | **Marks Awarded** | | | | | First assessor | |  | | | IV marks | |  | | | Agreed grade | |  | | | Signature of the assessor |  | Date |  | |

Contents

[Mega City Cabs 6](#_Toc192859674)

[Class Diagram For the System 6](#_Toc192859675)

[Use Case Diagram 7](#_Toc192859676)

[Sequence Diagram 8](#_Toc192859677)

[Justification For the System Design 10](#_Toc192859678)

[System Nature Processes and Operations Justification 12](#_Toc192859679)

[Coding Screenshots for the System 19](#_Toc192859680)

[GitHub Repository of the System 21](#_Toc192859681)

[User Manual for The System 23](#_Toc192859682)

[Future Recommendations For the system 32](#_Toc192859683)

[Test Plan 33](#_Toc192859684)

[References 39](#_Toc192859685)

[Figure 1 class diagram 7](#_Toc192859709)

[Figure 2 Login Sequence 9](#_Toc192859710)

[Figure 3 user add Sequence 9](#_Toc192859711)

[Figure 4 add Vehicle Sequence 10](#_Toc192859712)

[Figure 5 booking and billing sequence 11](#_Toc192859713)

[Figure 6 login screen 14](#_Toc192859714)

[Figure 7 signup screen 15](#_Toc192859715)

[Figure 8 admin panel 16](#_Toc192859716)

[Figure 9 driver panel 16](#_Toc192859717)

[Figure 10 customer panel 16](#_Toc192859718)

[Figure 11 user management 17](#_Toc192859719)

[Figure 12 vehicle management 17](#_Toc192859720)

[Figure 13 Destinations Management 18](#_Toc192859721)

[Figure 14 distance and fare calculation 18](#_Toc192859722)

[Figure 15 book a trip 19](#_Toc192859723)

[Figure 16 driver and vehicle selection 19](#_Toc192859724)

[Figure 17 locked out driver 20](#_Toc192859725)

[Figure 18 locked out vehicle 20](#_Toc192859726)

[Figure 19 Backend Structure 21](#_Toc192859727)

[Figure 20 Front End Structure 22](#_Toc192859728)

[Figure 21 Git hub Repository 23](#_Toc192859729)

[Figure 22 main login 25](#_Toc192859730)

[Figure 23 user role selection 25](#_Toc192859731)

[Figure 24 booking 26](#_Toc192859732)

[Figure 25 Trips Screen 26](#_Toc192859733)

[Figure 26 bill table 27](#_Toc192859734)

[Figure 27 bill View 27](#_Toc192859735)

[Figure 28 bill print 28](#_Toc192859736)

[Figure 29 driver view 29](#_Toc192859737)

[Figure 30 trip end 29](#_Toc192859738)

[Figure 31 bill generated 29](#_Toc192859739)

[Figure 32 Admin View 30](#_Toc192859740)

[Figure 33 User Management 30](#_Toc192859741)

[Figure 34 Add New User 31](#_Toc192859742)

[Figure 35 edit user 31](#_Toc192859743)

[Figure 36 manage vehicle 32](#_Toc192859744)

[Figure 37 edit vehicle 33](#_Toc192859745)

[Figure 38 Destination Manage 33](#_Toc192859746)

### Mega City Cabs

### Class Diagram For the System

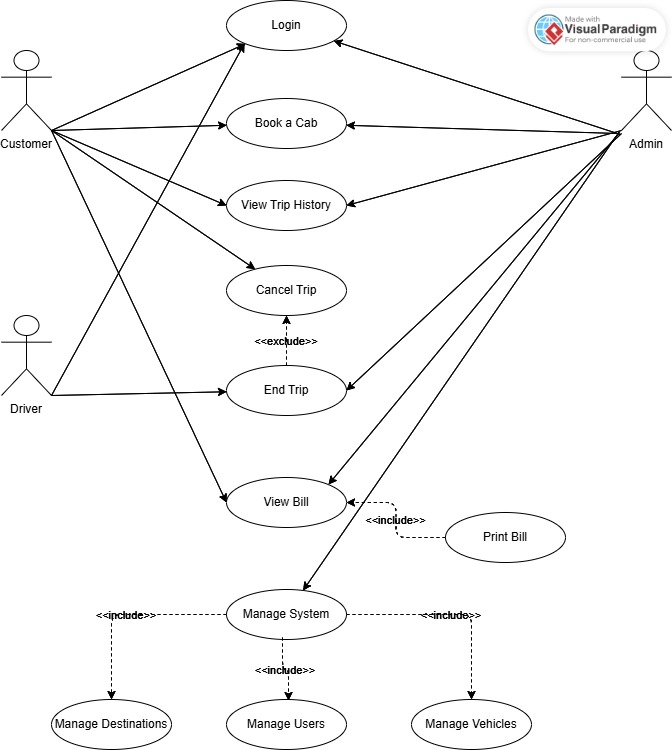
Several blue screens with text

AI-generated content may be incorrect.

Figure 1 class diagram

(Original Image can be found Here <https://github.com/pramodsandakelum/vehiclesystem/blob/main/Diagrams/Class%20Diagram1.jpg>)

### Use Case Diagram



### Sequence Diagram

A diagram of a login process

AI-generated content may be incorrect.

Figure 2 Login Sequence

A diagram of a program

AI-generated content may be incorrect.

Figure 3 user add Sequence

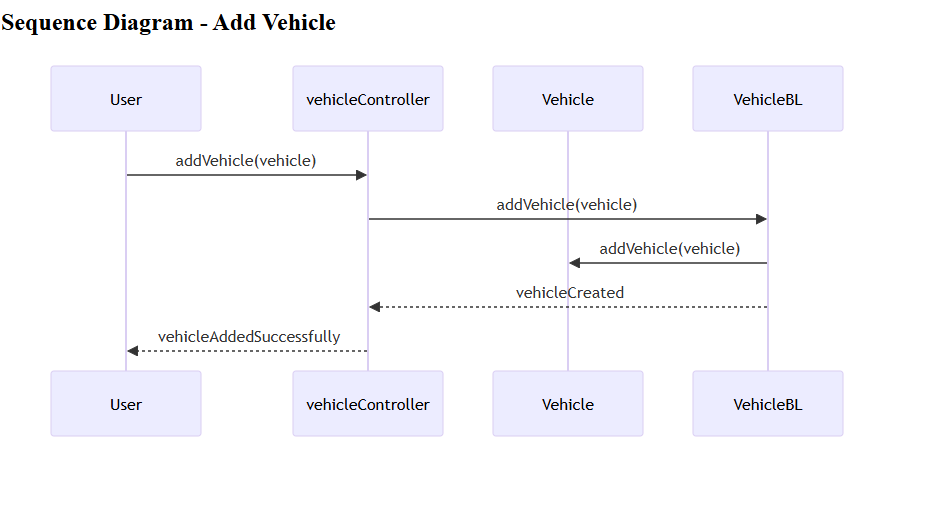


Figure 4 add Vehicle Sequence

A diagram of a company

AI-generated content may be incorrect.

Figure 5 booking and billing sequence

### Justification For the System Design

The system is designed following the MVC (Model-View-Controller) architecture and Singleton Design Pattern with a layered approach to separate concerns, improve scalability, and ensure modular design. Below is the justification for each component:

**Controller Layer**

The four controller classes

1. UserController
2. DriverController
3. VehicleController
4. BookingController

act as intermediaries between the client (Front end) and the business logic layer. Their primary responsibilities are:

* Handling HTTP requests (e.g., user authentication, booking creation, insert update delete operations for the whole system).
* Communicating with the Business Logic Layer (BL) to process data and return responses.
* Ensuring data validation and input sanitization before passing it to the BL.

**Reason**

* This ensures that all business rules are implemented in the BL, keeping controllers lightweight.
* Improves maintainability by making controllers independent of database operations.

**Business Logic (BL) Layer**

The four BL classes

1. UserBL
2. DriverBL
3. VehicleBL
4. BookingBL

serve as the core processing units of the system. Their primary functions include:

* Processing data received from the controllers.
* Applying business rules such as fare calculation, trip assignment, and status updates.
* Interfacing with the DBHandler for database operations.

**Reason**

* Separates business logic from controllers, making it easier to modify logic without affecting external APIs.
* Encapsulates core operations, allowing future enhancements like Google maps based location feeding to the system and driver allocation using gps eg.- Uber.

**Model Layer**

The nine model classes

1. User
2. Driver
3. Vehicle
4. Booking
5. Bill
6. Destination
7. userCredentialDTO
8. bookingdetailDTO
9. billcalculateDTO

represent real-world entities and database structures. Their purpose is to:

* Define data attributes and enforce object structure.
* Facilitate communication between the BL and DB.
* Ensure encapsulation by providing getters and setters.

Reason

* Promotes reusability across multiple system layers.
* Enhances data consistency by enforcing well-defined attributes.
* Faster Transactions rather than a direct database connection

**Database Handler (DBHandler)**

The DBHandler class is a **singleton** that manages database connectivity in a Java application. It ensures that only **one instance of the database connection exists** throughout the application lifecycle.

* Uses a **singleton pattern** to ensure that only **one connection instance** exists.
* Providing database connectivity and handling transactions.
* Executing CRUD operations for all entities.

Reason

* Prevents direct DB access from multiple classes, reducing redundancy.
* Enhances security and scalability by centralizing query execution.

### System Nature Processes and Operations Justification

This system Operates as a Backend API there are API end points for each operation

And the Frontend Part Operates Using JSP Pages and JavaScript functions using JSON for in between data transactions

Users can Sign up and Login to the System

And the Based on the User Roles the functionality is different

Below are Some Screenshots

Login

A sign with a login box and a blue and yellow sign

AI-generated content may be incorrect.

Figure 6 login screen

Signup

A screenshot of a computer

AI-generated content may be incorrect.

Figure 7 signup screen

Admin Panel

A screenshot of a computer

AI-generated content may be incorrect.

Figure 8 admin panel

Driver

A screenshot of a computer

AI-generated content may be incorrect.

Figure 9 driver panel

Customer

A screenshot of a computer

AI-generated content may be incorrect.

Figure 10 customer panel

**Registration Process**

System Already Has a built-in Super Admin Account

Rest of the users can register into the system using sign up portal as customers or drivers

And if the User is a driver, he doesn’t need to create a separate driver profile based on the role selection the system will automatically generate the driver profile.

Other than that, the administrator can change the user profiles to admin customer or user

Below is the Admin Panel for User Management

A screenshot of a computer

AI-generated content may be incorrect.

Figure 11 user management

Like wise the admin can manage Vehicles and Destinations also

A screenshot of a car registration form

AI-generated content may be incorrect.

Figure 12 vehicle management

A screenshot of a computer

AI-generated content may be incorrect.

Figure 13 Destinations Management

**Destinations Management**

When inserting Destinations the admin can enter the corresponding location latitude and longitude

**Reason**

This is Because the system uses the Haversine Formula to calculate the distance between the two points of pickup and drop location providing the user with accurate information like taxi fare and the total distance

The Haversine formula is used to calculate the great-circle distance (shortest distance) between two points on a sphere, given their latitudes and longitudes. It is commonly used in geographical applications like GPS and navigation systems. (SimonKettle, 2017)

A screenshot of a computer

AI-generated content may be incorrect.

Figure 14 distance and fare calculation

**Booking a Trip**

The Customer can easily book a trip using below simple interface and after the booking is done the related driver and the car is locked out in the system until the relevant trip is ended or cancelled this prevents accidental allocation of a unavailable driver or a car for another ride

A screenshot of a computer

AI-generated content may be incorrect.

Figure 15 book a trip

Even the administrator is locked out from editing or deleting the driver or car from a active trip

This is an example booking demo to show the system capability

A screenshot of a computer

AI-generated content may be incorrect.

Figure 16 driver and vehicle selection

View for the Admin locked out user and car

A screen shot of a computer

AI-generated content may be incorrect.

Figure 17 locked out driver

A screenshot of a computer

AI-generated content may be incorrect.

Figure 18 locked out vehicle

Automatic Logout functionality

The System will automatically log out the user after 15 minutes of inactivity to prevent the wastage of server resources and for extra security.

### Coding Screenshots for the System

**Project Structure**

Backend

A screenshot of a computer

AI-generated content may be incorrect.

Figure 19 Backend Structure

**Front End Structure**

**A screenshot of a computer

AI-generated content may be incorrect.**

Figure 20 Front End Structure

**Backend has three main packages**

1. Controller

JAX-RS API End points to communicate with the front end

1. Models

Model Classes For the entities inside the system that are mapped with the database tables to maintain inter communication

1. Service

Business Logic Layer of the System That will perform CRUD Operations for entire system

**Front End**

Front end consists of JSP Pages and JavaScript files that are linked them JSP Pages will provide the view for the user while JavaScript files will maintain the transactions between frontend an the backend using JSON objects for increased reliability JS files and JSP files are coded separately for better troubleshooting.

### GitHub Repository of the System

**Version Control and GitHub Repository**

To ensure efficient tracking of project changes, a public GitHub repository has been created for version control and deployment. The repository is updated daily with new features, bug fixes, and enhancements. Below is the link to the repository:

GitHub Desktop Client Was Used for Version Control and To Push the Code into Server

🔗 **GitHub Repository:** <https://github.com/pramodsandakelum/vehiclesystem>

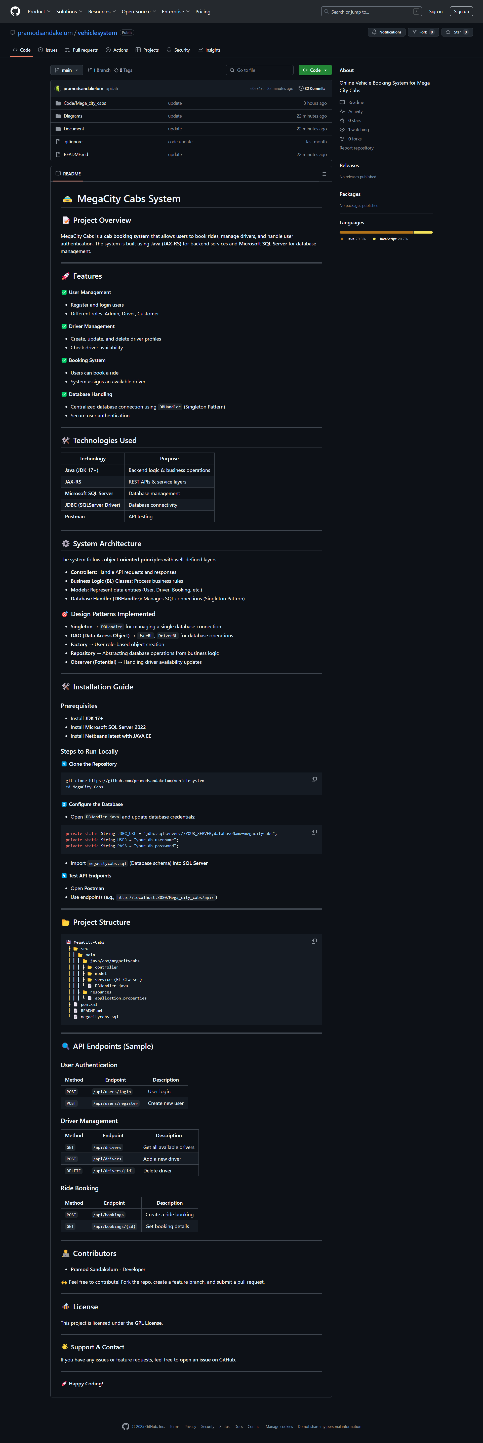


Figure 21 Git hub Repository

**Version Control Practices Implemented:**

* **Commit Messages:** Clear and descriptive commit messages are used for tracking changes.
* **Pull Requests & Code Reviews:** Changes are reviewed before merging to maintain code quality.
* **Tagging and Releases:** Each major update is tagged for easy tracking.
* **Continuous Integration:** Automated builds and tests are triggered upon every push to the repository.

**Workflow Demonstration:**

1. **Clone the Repository:**

git clone [<https://github.com/pramodsandakelum/vehiclesystem>]

1. **Create a New Branch for Features:**

git checkout -b feature-branch

1. **Stage and Commit Changes:**

git add

git commit -m "Added new feature"

1. **Push Changes and Create a Pull Request:**

git push origin feature-branch

1. **Merge Approved Changes:**

git checkout main

git merge feature-branch

### User Manual for The System

**Login and Registration Process**

If The User already has an account user can login directly using the username and password

A sign with a login and a taxi sign

AI-generated content may be incorrect.

Figure 22 main login

Otherwise, the user has to register in the system during registration user must select the correct role Customer or Driver if the user failed to do so they will have to contact the administrator to change the user role to relevant one.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 23 user role selection

**Customer – Book A Trip**

To add a booking user must click on the book a cab link on the side bar then on the booking screen user must fill the details correctly after selecting the drop and pickup location the user can calculate the actual fare and distance to travel between the two points by clicking calculate fare button all fields are validated user can’t submit without filling the necessary details

A screenshot of a computer

AI-generated content may be incorrect.

Figure 24 booking

**Customer – View Trips**

Customer can view all the trips under his id by visiting the Trip History page ongoing trip can be cancelled by clicking the Cancel Trip button

A table with text and images

AI-generated content may be incorrect.

Figure 25 Trips Screen

**Customer -Billing**

From Bills Screen customer can view the bills and print them and if the customer has any promo codes, they can apply it prior to billing to obtain a discount by applying the code to the bill.

A close-up of a computer screen

AI-generated content may be incorrect.

Figure 26 bill table

A screenshot of a computer

AI-generated content may be incorrect.

Figure 27 bill View

**Bill Print View**

By clicking on the print bill button, the customer can print or save the bill as a pdf document.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 28 bill print

**Driver – Trip History**

Login process is same for the driver after logging into the system a driver can see this screen

A table with text on it

AI-generated content may be incorrect.

Figure 29 driver view

Finish a Trip

Driver Can end the trip after arriving at the destination after ending the trip the bill will be generated automatically for the customer so he can pay it after

A screenshot of a computer

AI-generated content may be incorrect.

Figure 30 trip end

Generated Bill

A screenshot of a computer

AI-generated content may be incorrect.

Figure 31 bill generated

**Admin – Managing User Vehicles Destinations**

Admin uses the same login for the system this is the view for an admin

A white background with black and white text

AI-generated content may be incorrect.

Figure 32 Admin View

Managing Users

Admin can manage users in this screen add update delete or change the role of a user

If a driver is on a booked trip the driver is not available to edit or delete until the trip ends

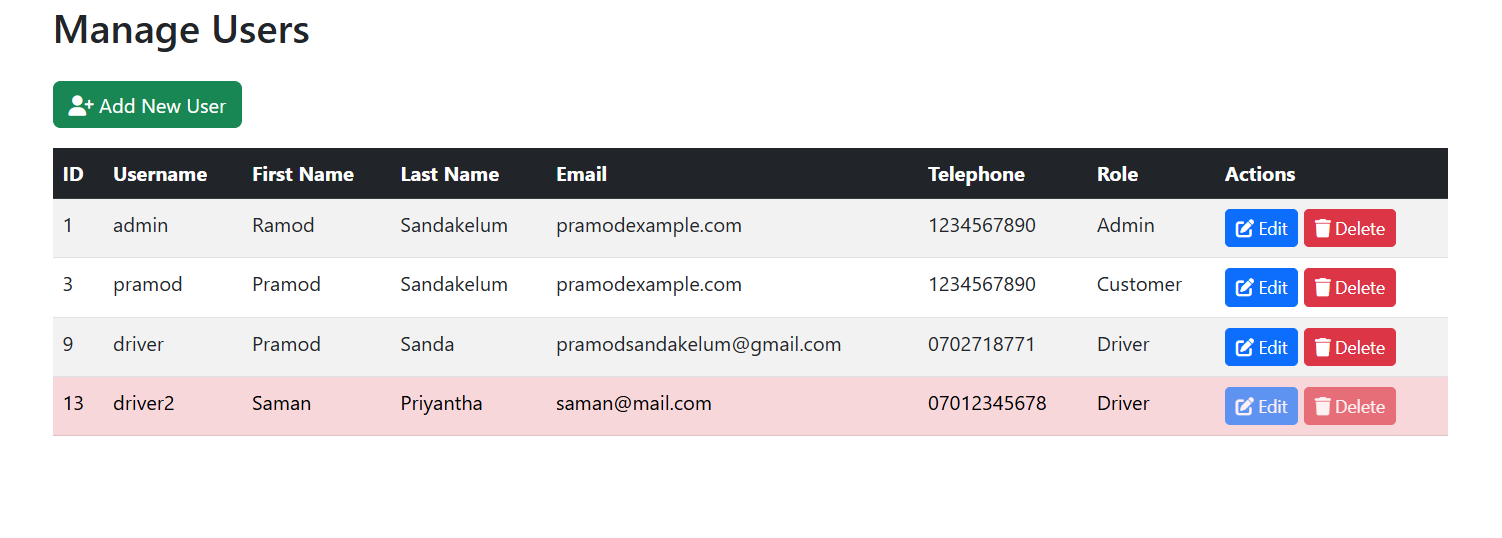


Figure 33 User Management

Add new User

A screenshot of a login form

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 34 Add New User

Edit Existing User

Admin can click Edit Button in front of each user to edit them

A screenshot of a computer

AI-generated content may be incorrect.

Figure 35 edit user

Delete User

To delete a user simply click delete button and confirm the message

A screenshot of a computer

AI-generated content may be incorrect.

**Admin – Manage Vehicles**

Managing vehicles also same as above process Vehicles that are already booked cannot be edited or deleted until the trip is finished.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 36 manage vehicle

Edit Vehicle

When editing a vehicle, the booking status cannot be edited because it is decided by the booking generation because if mistakenly a vehicle state was changed it will be unavailable for bookings

A screenshot of a computer

AI-generated content may be incorrect.

Figure 37 edit vehicle

**Admin – Manage Destinations**

Destinations can be added to the system, so the users are able to select pickup and drop locations

When booking a trip when adding the location coordinates for the location should be inserted because these coordinates are used to calculate distance and trip fares

A screenshot of a computer

AI-generated content may be incorrect.

Figure 38 Destination Manage

Edit And Delete is same process as the users and vehicles.

**Other Tasks for Admin**

The admin can view trip history and end trips for any user or driver and the admin can view bills for any user.

### Future Recommendations For the system

* 1. Google maps Api can be integrated into the system, so the users don’t need to insert the locations into the system
  2. Vehicles can be equipped with GPS trackers to monitor the movement Realtime
  3. A mobile version can be developed for more compatibility.

### Test Plan

This document outlines the test plan for the Mega City Cab system, ensuring all core functionalities are tested systematically. Each test case is structured professionally with clear steps, expected results, and placeholders for screenshots. The focus remains on validating key functionalities to maintain a seamless user experience.

**2. Scope**

The testing will cover the following key functionalities:

* **User Login Test**
* **Add New User**
* **Display Trip Details**
* **Calculate and Print Bill**
* **Add a Trip Booking**
* **Auto Logout Test**

**Test Types and Approach**

* **Manual Testing:**
  + Conducted for exploratory, usability, and UI testing scenarios.
  + Each test case in the plan (e.g., login functionality, booking entry) is manually executed to verify functionality and user experience.
* **System Testing:**
* Comprehensive testing of the entire application to validate overall system behavior.

**3. Test Cases**

**3.1. Test Case: User Login Test - TC/LOGIN/MCB/001**

|  |  |
| --- | --- |
| **Test Case ID** | TC/LOGIN/MCB/001 |
| **Description** | Validate the login function with both valid and invalid credentials. |
| **Preconditions** | The application should be running. |
| **Test Steps** | 1. Enter a valid username and password.  2. Click the Login button.  3. Verify that the system grants access to the dashboard.  4. Logout From System  4. Enter an invalid username or password.  6. Click Login. |
| **Expected Results** | ✅ Valid credentials: Successful login and redirection to the main dashboard.  ❌ Invalid credentials: Display of an error message and no access granted. |
| **Screenshot** |  |

**3.2. Test Case: Add New User - TC/USER/MCB/002**

|  |  |
| --- | --- |
| **Test Case ID** | TC/USER/MCB/002 |
| **Description** | Validate that a new customer can be added, capturing all required details. |
| **Preconditions** | Admin is logged into the system. |
| **Test Steps** | 1. Navigate to the User Management section.  2. Enter the following details:  - username - password - first name - last name - email - phone  – address – user role  3. save. |
| **Expected Results** | ✅ The booking is saved successfully, and a user is generated.  ✅ A confirmation message is displayed. |
| **Screenshot** |  |

**3.3. Test Case: Display Trip Details - TC/TRIP/MCB/003**

|  |  |
| --- | --- |
| **Test Case ID** | TC/TRIP/MCB/003 |
| **Description** | Validate that Trip details are displayed correctly. |
| **Preconditions** | At least one Trip detail exists in the system. |
| **Test Steps** | 1. Navigate to the Trip Details section.  2. View Is loaded with existing Details. |
| **Expected Results** | ✅ All details related to the Trip Booking (Code, Date, Customer, Driver etc.) are displayed accurately. |
| **Screenshot** | **A black and white table with text  AI-generated content may be incorrect.** |

**3.4. Test Case: Calculate and Print Bill - TC/BILL/MCB/004**

|  |  |
| --- | --- |
| **Test Case ID** | TC/BILL/MCB/004 |
| **Description** | Validate the functionality for calculating the bill and printing it. |
| **Preconditions** | A valid booking must exist. |
| **Test Steps** | 1. Open Bill Section.  2. Click the View Bill button.  3.Apply a Promo Code to see discounts are applied correctly pay bill  4.Print Bill |
| **Expected Results** | ✅ The bill amount is calculated correctly with taxes/discounts applied.  ✅ The printed bill output reflects the accurate total. |
| **Screenshot** |  |

**3.5. Test Case: Add a Trip Booking - TC/BOOK/MCB/005**

|  |  |
| --- | --- |
| **Test Case ID** | TC/BOOK/MCB/005 |
| **Description** | Validate A Correct Trip Booking is added. |
| **Preconditions** | A Customer or Admin is logged in to the system. |
| **Test Steps** | 1. Open Book a Cab Section.  2. Fill the Form with Correct Details like Pickup And Drop Location.  3.Click Confirm Booking.  4.Check Trip History Section for a Ongoing Trip |
| **Expected Results** | ✅ The Trip Is Inserted Correctly with A Booking Code.  ✅ The printed bill output reflects the accurate total. |
| **Screenshot** |  |

**3.6. Test Case: Auto Logout Test - TC/LOGIN/MCB/006**

|  |  |
| --- | --- |
| **Test Case ID** | TC/LOGIN/MCB/006 |
| **Description** | Test Session Timeout and Logout Process. |
| **Preconditions** | A Customer or Admin is logged in to the system. |
| **Test Steps** | 1. Default Session Timeout is for the System Is 15 Minutes.  2. Wait 15 Minutes Without Activity to check |
| **Expected Results** | ✅ User is Logged out of system after 15 minutes. |
| **Screenshot** |  |

# References

SimonKettle, 2017. *Distance on a sphere: The Haversine Formula.* [Online]   
Available at: https://community.esri.com/t5/coordinate-reference-systems-blog/distance-on-a-sphere-the-haversine-formula/ba-p/902128  
[Accessed 25 January 2025].