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, …………………………………………. <name of the student and registration number>, 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 |  | |

### Mega City Cabs

### Class Diagrams For the System

Several blue screens with text

AI-generated content may be incorrect.

Figure 1 class diagram

### Use Case Diagram

### Sequence Diagram

### 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 2 login screen

Signup

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3 signup screen

Admin Panel

A screenshot of a computer

AI-generated content may be incorrect.

Figure 4 admin panel

Driver

A screenshot of a computer

AI-generated content may be incorrect.

Figure 5 driver panel

Customer

A screenshot of a computer

AI-generated content may be incorrect.

Figure 6 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 7 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 8 vehicle management

A screenshot of a computer

AI-generated content may be incorrect.

Figure 9 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 10 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 11 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 12 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 13 locked out driver

A screenshot of a computer

AI-generated content may be incorrect.

Figure 14 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 15 Backend Structure

**Front End Structure**

**A screenshot of a computer

AI-generated content may be incorrect.**

Figure 16 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

<https://github.com/pramodsandakelum/vehiclesystem>

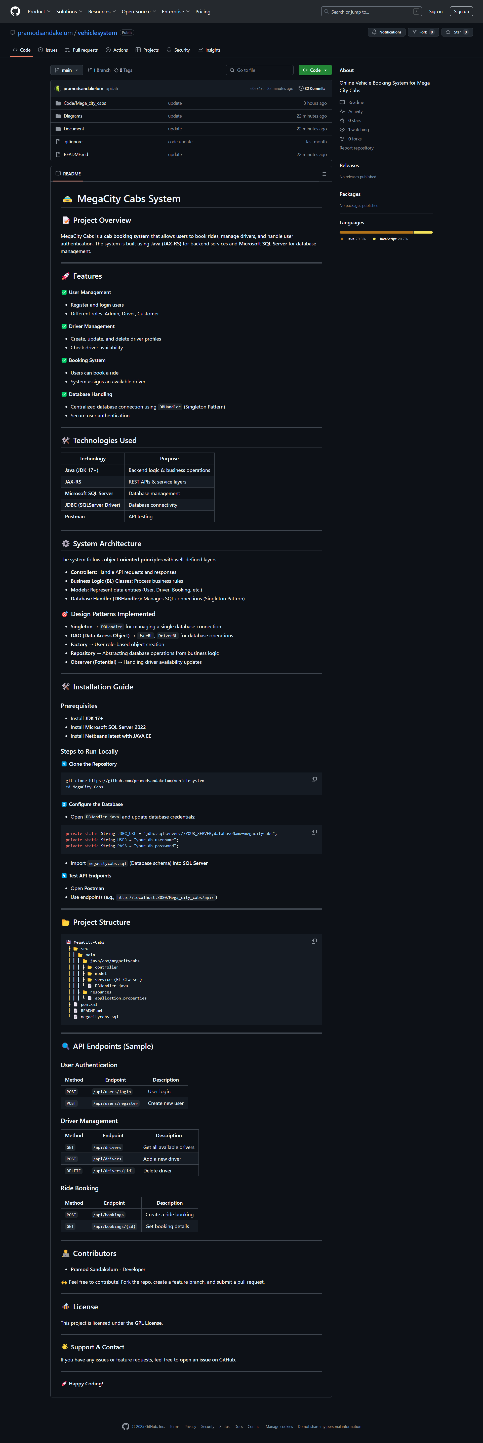


Figure 17 Git hub Repository

### 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 18 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 19 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 cant submit without filling the necessary details

A screenshot of a computer

AI-generated content may be incorrect.

Figure 20 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 21 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 22 bill table

A screenshot of a computer

AI-generated content may be incorrect.

Figure 23 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 24 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 25 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 26 trip end

Generated Bill

A screenshot of a computer

AI-generated content may be incorrect.

Figure 27 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 28 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

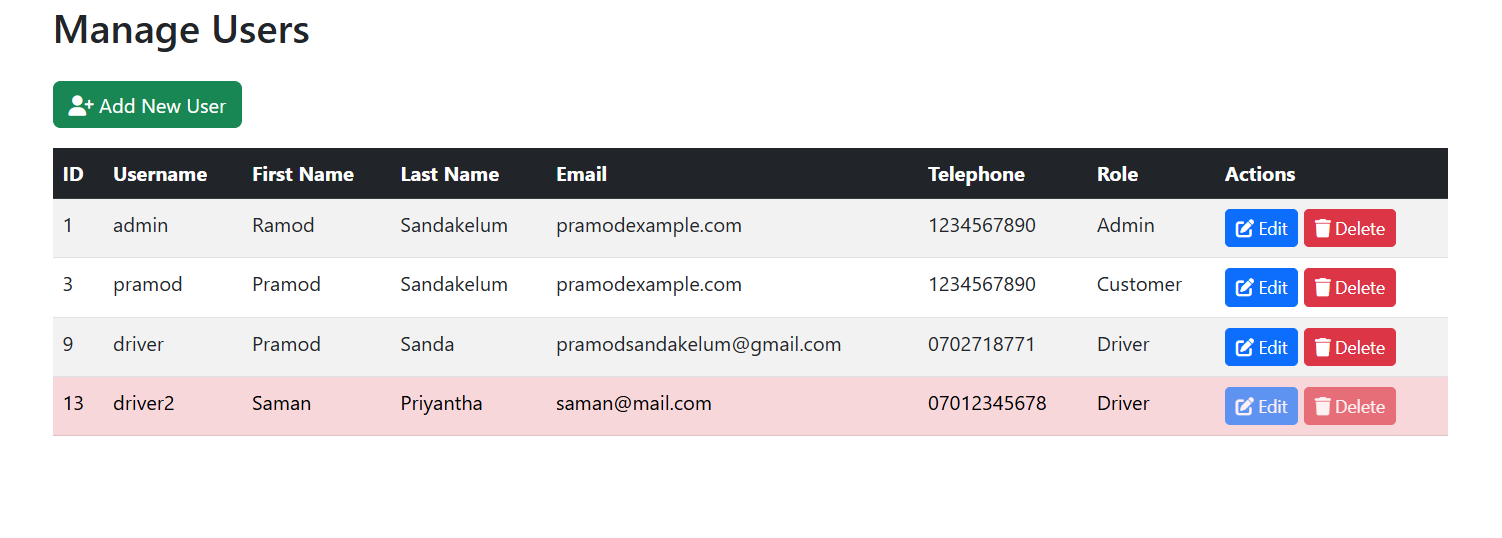


Figure 29 User Management

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 30 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 31 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 32 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 33 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 34 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.