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 and every operations

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

Users can Signup 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