RYDON Vehicle Rental System - Project Report

GROUP 8

Submitted By,

Agnus Christopher,07

Aleena Santhosh,11

Athira S Kumar,21

Gangothri Ganesh,29

RYDON Vehicle Rental System - Project Report

1. Introduction

The RYDON Vehicle Rental System is a Java-based desktop application developed to simplify vehicle rentals. It allows customers to view, book, and manage vehicle rentals while enabling administrators to oversee bookings and maintain vehicle data efficiently. This system uses a MySQL database for data storage and follows an MVC architecture for modular and scalable design.

2. Objectives

• Develop a vehicle rental management application with separate modules for customers and administrators.
• Simplify the vehicle booking process with an intuitive graphical user interface.
• Enable efficient vehicle data management, customer tracking, and booking records.
• Implement MySQL as the backend database for persistent data storage.
• Follow object-oriented design and MVC architecture principles for clarity and maintainability.

3. Requirement Specification

Hardware Requirements

• Processor: Intel i3 or higher
• RAM: Minimum 4GB
• Hard Disk: Minimum 500MB free space
• Display: 1366x768 resolution or higher

Software Requirements

• Operating System: Windows 10 or later
• Programming Language: Java (JDK 17 or above)
• IDE: IntelliJ IDEA / Eclipse / VS Code
• Database: MySQL 8.0 or above
• Build Tool: Apache Maven

4. Design Document

The **RYDON Vehicle Rental System** consists of four primary classes that represent the key entities in the system — *Vehicle, Customer, Booking,* and *Maintenance.*
Each class defines essential fields used to manage the rental process, maintenance records, and customer information. The classes interact through unique identifiers to maintain data integrity and relationships.

Class Design

Vehicle

This class defines all vehicles available for rent.
It stores details such as vehicle ID, registration number, type (Car or Bike), make, model, year, odometer reading, daily rental rate, per-kilometer rate, and availability status.
Additional attributes include photo, insurance details, and pollution test information, ensuring that only valid and roadworthy vehicles are listed for rental.

Customer

The Customer class holds customer-related data such as ID, name, phone number, email, and address.
This information is used to identify renters and associate them with their respective bookings. A single customer can have multiple bookings over time.

Booking

The Booking class manages the details of each rental transaction.
It includes booking ID, customer ID, vehicle ID, start and end dates, odometer readings, and total charge.
It also tracks whether the customer requires a driver and maintains the booking's status (Active, Completed, or Cancelled).
This class links customers to vehicles and records the duration and distance of each rental.

Maintenance

This class keeps records of all vehicle maintenance activities.
It contains maintenance ID, vehicle ID, date, maintenance type, vendor name, cost, and additional notes.
This ensures that vehicles are properly serviced and tracked throughout their lifecycle.

Database Design

The database design includes four primary tables:

- user(user_id, name, email, password)
- vehicle(vehicle_id, make, model, type, availability)

• booking(booking_id, user_id, vehicle_id, booking_date, return_date)

The relationships are established through foreign keys linking users and vehicles to bookings.

5. Source Code

The source code follows an MVC structure:
• model/ – Contains data models such as Vehicle, User, and Booking.
• dao/ and dao/impl/ – Data Access Objects and their implementations for CRUD operations.
• service/ – Business logic layer for bookings and vehicles.
• controller/ – Handles interactions between UI and backend logic.
• ui/ – Java Swing components forming the front-end interface.

6. User Manual

1. Launching the Application:
   - Open the RYDON Vehicle Rental System project in your IDE.
   - Ensure the MySQL server is running.
   - Compile and run the main Java file in the 'ui' package.

2. Login Process:
   - The login screen provides options for Admin and Customer.
   - Enter valid credentials to access the respective dashboard.

3. Admin Features:
   - Add, edit, or delete vehicles in the system.
   - View all registered users and their bookings.
   - Manage booking records and availability status.

4. Customer Features:
   - View the list of available vehicles with brand, model, and type.
   - Select a vehicle and create a booking.
   - View booking history and update user details.

5. Database Configuration:
   - Modify connection settings in 'MySQLConnection.java' (URL, username, password).
   - Import 'rydon_schema.sql' in MySQL Workbench to create tables.

6. Exit and Data Persistence:
   - On exiting, all data is retained in the MySQL database.
   - Next session resumes with existing data.

7. Test Cases

• TC01 – Login Functionality:
 - Input: Valid and invalid credentials.
 - Expected Result: User with valid credentials logs in successfully; invalid login shows error message.

• TC02 – Vehicle Addition by Admin:
 - Input: Admin adds a new vehicle with details (brand, model, type).
 - Expected Result: Vehicle appears in the available vehicle list; record is saved in the database.

• TC03 – Booking Creation by Customer:
 - Input: Customer selects an available vehicle and provides booking details.
 - Expected Result: Booking record created; vehicle marked unavailable; confirmation displayed.

• TC04 – View and Manage Bookings:
 - Input: Admin or user views bookings.
 - Expected Result: Booking list loads with correct user-vehicle mapping.

• TC05 – Vehicle Availability Update:
 - Input: Vehicle booked or cancelled.
 - Expected Result: Availability field updates automatically in the database.

• TC06 – Data Persistence:
 - Input: Application closed and reopened.
 - Expected Result: All data remains intact; no data loss.

• TC07 – Error Handling:
 - Input: Missing or invalid entries in forms.
 - Expected Result: Proper validation messages displayed; no crash occurs.

• TC08 – UI Functionality:
 - Input: User navigates through menus and dashboards.
 - Expected Result: All buttons, forms, and navigation work smoothly without lag or errors.

8. Conclusion

The RYDON Vehicle Rental System effectively streamlines vehicle rental management using a clean MVC architecture. It provides efficient booking management, secure database connectivity, and an intuitive user experience. Future improvements could include adding online payment integration, mobile app support, and enhanced reporting features.