Vehicle Rental System

Project Title: Vehicle Rental System
Submitted By: Rayyan Aamir (24K-0687), Hammad Haider (24K-0634), Usaid Khan (24K-0832)
Course: OOP Lab
Instructor: Shafique-ur-Rehman
Submission Date: 11th May, 2025

---

1. Executive Summary

Project Overview:
This project simulates a console-based Vehicle Rental System using Object-Oriented Programming (OOP) principles in C++. It allows customers to rent and return various types of vehicles (Car, Bike, Truck) while providing administrative functionality to manage the vehicle inventory. The rental system demonstrates robust design through class relationships, encapsulation, inheritance, and file handling.

---

2. Introduction

Background:
Traditional vehicle rental processes involve manual handling of records and bookings. This project was chosen to demonstrate how a digital alternative can simplify and automate rental management. The system introduces a menu-driven interface with organized vehicle categorization and user-level privileges for administrators and customers.

Objectives of the Project:

- Enable vehicle rental and return functionality for customers.
- Allow admins to manage vehicle inventory.
- Store persistent data using file handling.
- Apply core OOP principles effectively.

---

3. System Description

Original System Concept:
Inspired by real-world vehicle rental platforms like Uber Rentals and local rental

agencies, this system simulates a miniature yet functional terminal-based vehicle rental application.

Innovations and Modifications:

- Different user roles: Admin and Customer.
- Admin privileges to add or remove vehicles.
- Customer bookings are stored individually.
- Real-time vehicle availability and booking status.
- Colored terminal interface for better visuals.

---

4. Programming Approach and Methodology

OOP Techniques Used:

- **Inheritance:** Car, Bike, Truck inherit from Vehicle & Admin, Customer inherit from User
- **Polymorphism:** Display methods are overridden for custom vehicle output.
- **Encapsulation:** Data members are private/protected with access via public methods.
- **Operator Overloading:** << operator is overloaded for displaying vehicles.
- **Friend Functions:** Provide controlled access to private data.
- **Static Members:** Used to track object counts and ID counters.
- **Composition:** Each Customer contains a vector of their own Booking objects.
- **File Handling:** CSV files used for persistent vehicle, booking, admin and customer data.

Methodology:

- Modular design for maintainability.
- Color-coded UI for terminal navigation.
- Keyboard navigation (arrow/WASD).

---

5. System Rules and Flow

Modified System Rules:

- Admins can add or delete vehicles.
- Each vehicle has a type, ID, rate, and availability.
- Customers can only rent available vehicles and return their rented vehicles.
- Booking history is user-specific.

Turn-based Mechanics:

- User selects role (Admin or Customer).
- Admins access inventory management menu.
- Customers navigate booking system.

Completion Conditions:

- Vehicle is rented and marked unavailable.
- Return updates status and removes booking.

---

6. Implementation and Development

Development Process:

- Implemented in C++ with classes for Vehicle, Booking, Admin, and Customer.
- Applied CSV file handling for storing data.
- Console UI built with utility functions and color ANSI codes.

Programming Languages and Tools:

- **Language:** C++
- **Libraries:** iostream, fstream, conio.h, cstdlib.h, iomanip, sstream, vector, string
- **Tools:** Visual Studio Code, GitHub, Git

Challenges Encountered:

- Designing polymorphic vehicle structure.
- Managing file reading/writing for objects.
- Handling input/output without UI libraries.
- Keyboard navigation using _getch().

---

7. Team Contributions

- **Rayyan Aamir:** Developed admin functionalities, customer booking and main function.
- **Usaid Khan:** Implemented vehicle management logic and file handling via CSV files.
- **Hammad Haider:** Designed colored terminal interface and formatted output, and finalized project report. Also implemented bookings class.

---

8. Results and Discussion

The system successfully allows users to book and manage vehicles through a terminal interface. OOP principles were applied throughout, ensuring scalability and clarity in the codebase.The structured logic and UI design provided a robust learning experience in C++ development.

---

9. References

- C++ Primer (5th Edition)
- www.cplusplus.com
- GitHub for code collaboration and version control
- Project Source Code : https://github.com/Rayyan-2704/Vehicle-Rental-System