Online Car Rental Platform - Project Report

Objective:

To develop an online car rental platform using Python and Object-Oriented Programming (OOP) principles, allowing customers to rent cars hourly, daily, or weekly, view available inventory, and receive auto-generated bills.

Problem Statement:

A car rental company requires an online platform for managing car rentals. The platform should allow customers to:
1. View the inventory of available cars.
2. Rent cars based on hourly, daily, or weekly rates.
3. Return rented cars and generate a detailed bill.

Key requirements:
- Customers can rent cars if the requested quantity is available.
- The system updates the inventory and tracks rental details for billing.

Tools and Technologies Used:
- Programming Language: Python
- Environment: Jupyter Notebook
- Modules:
  - `datetime` (for time tracking)
  - Custom modules for `CarRental` and `Customer`

Implementation Details

1. CarRental Module
The `CarRental` class is responsible for managing car inventory, handling rental operations, and calculating bills.

Methods:
- `__init__(self, stock=0)`: Initializes the car inventory.
- `display_cars()`: Returns the number of available cars.
- `rent_hourly(num_of_cars)`: Allows renting cars on an hourly basis.
- `rent_daily(num_of_cars)`: Allows renting cars on a daily basis.
- `rent_weekly(num_of_cars)`: Allows renting cars on a weekly basis.
- `return_car(rental_time, rental_mode, num_of_cars)`: Calculates the rental period, updates the stock, and generates a bill.

2. Customer Module

The `Customer` class manages customer interactions, such as requesting and returning cars.

Methods:
- `__init__(self)`: Initializes rental details.
- `request_car(num_of_cars)`: Validates and processes car rental requests.
- `return_car()`: Provides rental details for processing returns.

3. Main Script
The main script integrates the modules and provides a user interface.

Features:
- Displays available cars.
- Handles rental requests for hourly, daily, and weekly modes.
- Processes returns and calculates bills.

User Interaction:
- The script prompts users to choose actions (e.g., view cars, rent cars, return cars).
- Accepts inputs for the number of cars and rental mode.
- Displays messages for successful operations and errors.

Sample Workflow

1. Viewing Inventory:
   - User selects the option to view available cars.
   - The system displays the current stock.

2. Renting Cars:
   - User specifies the number of cars and rental mode.
   - The system validates the request and updates the inventory.

3. Returning Cars:
   - User provides rental details (time, mode, and quantity).
   - The system calculates the rental duration and generates a bill.

Billing System

- Hourly Rate: $5 per hour.
- Daily Rate: $20 per day.
- Weekly Rate: $60 per week.

The bill is calculated based on rental duration, mode, and the number of cars rented.

Conclusion:

This project demonstrates the effective use of Python and OOP principles in building a functional car rental platform. The modular design ensures code reusability and maintainability, while the user-friendly interface simplifies customer interactions.

Future Enhancements

- Add a database for persistent inventory and transaction records.
- Implement user authentication for better security.
- Include a web-based interface for improved accessibility.
- Integrate additional features like discounts and loyalty programs.