**Car Rental System Project**

**Overview**

This is a simple **Car Rental System** using **Object-Oriented Programming (OOP)** principles in **Java**. The system allows customers to rent cars from a rental agency. The system supports:

* Adding cars and customers to the rental agency.
* Renting a car to a customer based on availability.
* Calculating rental costs based on the number of rental days.

**Components of the System**

**1. Car Class:**

* Represents a car available for rent.
* Contains properties like make, model, rental price per day, and availability status.
* Can calculate rental costs based on the number of days.

**2. Customer Class:**

* Represents a customer who rents cars from the agency.
* Contains details such as customer name, customer ID, and phone number.

**3. RentalAgency Class:**

* Manages the list of cars and customers.
* Allows the addition of cars and customers to the system.
* Handles the process of renting cars to customers based on availability.

**4. Main Class:**

* This is the entry point for the application.
* It creates objects for cars, customers, and the rental agency.
* Demonstrates the rental process by renting cars to customers.

**How the Program Works**

1. **Create Cars**: We define cars with their make, model, and rental price.
2. **Add Customers**: Customers are created with their names, customer IDs, and phone numbers.
3. **Rent Cars**: Customers can rent cars from the agency by specifying the car’s make, model, and rental period.
4. **Availability Check**: The system checks whether the requested car is available for rent.

**Code Implementation**

**Car.java**

java

Copy

// Car class

class Car {

private String make;

private String model;

private double rentalPricePerDay;

private boolean isAvailable;

public Car(String make, String model, double rentalPricePerDay) {

this.make = make;

this.model = model;

this.rentalPricePerDay = rentalPricePerDay;

this.isAvailable = true;

}

public boolean isAvailable() {

return isAvailable;

}

public void setAvailability(boolean isAvailable) {

this.isAvailable = isAvailable;

}

public double calculateRentalCost(int days) {

return rentalPricePerDay \* days;

}

@Override

public String toString() {

return make + " " + model + " | Price per day: $" + rentalPricePerDay;

}

}

**Customer.java**

java

Copy

// Customer class

class Customer {

private String name;

private String customerId;

private String phoneNumber;

public Customer(String name, String customerId, String phoneNumber) {

this.name = name;

this.customerId = customerId;

this.phoneNumber = phoneNumber;

}

public String getCustomerDetails() {

return "Customer: " + name + " | ID: " + customerId + " | Phone: " + phoneNumber;

}

}

**RentalAgency.java**

java

Copy

// RentalAgency class

import java.util.ArrayList;

import java.util.List;

class RentalAgency {

private List<Car> carsAvailable;

private List<Customer> customers;

public RentalAgency() {

carsAvailable = new ArrayList<>();

customers = new ArrayList<>();

}

public void addCar(Car car) {

carsAvailable.add(car);

}

public void addCustomer(Customer customer) {

customers.add(customer);

}

public Car rentCar(String customerId, String make, String model, int days) {

Customer customer = findCustomerById(customerId);

if (customer != null) {

for (Car car : carsAvailable) {

if (car.isAvailable() && car.toString().contains(make) && car.toString().contains(model)) {

car.setAvailability(false);

System.out.println(customer.getCustomerDetails() + " rented " + car.toString() + " for " + days + " days.");

return car;

}

}

}

System.out.println("Car not available for rent.");

return null;

}

private Customer findCustomerById(String customerId) {

for (Customer customer : customers) {

if (customerId.equals(customer.customerId)) {

return customer;

}

}

return null;

}

}

**Main.java**

java

Copy

// Main class

public class Main {

public static void main(String[] args) {

// Create Rental Agency

RentalAgency agency = new RentalAgency();

// Create Cars

Car car1 = new Car("Toyota", "Camry", 50.0);

Car car2 = new Car("Honda", "Civic", 45.0);

agency.addCar(car1);

agency.addCar(car2);

// Create Customers

Customer customer1 = new Customer("Ella Sifah", "C128", "123-456-7890");

Customer customer2 = new Customer("Ryan Joel", "C129", "987-654-3210");

agency.addCustomer(customer1);

agency.addCustomer(customer2);

// Rent Cars to Customers

agency.rentCar("C128", "Toyota", "Camry", 8);

agency.rentCar("C129", "Honda", "Civic", 11);

}

}

Top of Form

Bottom of Form