import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

class Car {

private String carId;

private String brand;

private String model;

private double basePricePerDay;

private boolean isAvailable;

public Car(String carId, String brand, String model, double basePricePerDay) {

this.carId = carId;

this.brand = brand;

this.model = model;

this.basePricePerDay = basePricePerDay;

this.isAvailable = true;

}

public String getCarId() {

return carId;

}

public String getBrand() {

return brand;

}

public String getModel() {

return model;

}

public double calculatePrice(int rentalDays) {

return basePricePerDay \* rentalDays;

}

public boolean isAvailable() {

return isAvailable;

}

public void rent() {

isAvailable = false;

}

public void returnCar() {

isAvailable = true;

}

}

class Customer {

private String customerId;

private String name;

public Customer(String customerId, String name) {

this.customerId = customerId;

this.name = name;

}

public String getCustomerId() {

return customerId;

}

public String getName() {

return name;

}

}

class Rental {

private Car car;

private Customer customer;

private int days;

public Rental(Car car, Customer customer, int days) {

this.car = car;

this.customer = customer;

this.days = days;

}

public Car getCar() {

return car;

}

public Customer getCustomer() {

return customer;

}

public int getDays() {

return days;

}

}

class CarRentalSystem {

private List<Car> cars;

private List<Customer> customers;

private List<Rental> rentals;

public CarRentalSystem() {

cars = new ArrayList<>();

customers = new ArrayList<>();

rentals = new ArrayList<>();

}

public void addCar(Car car) {

cars.add(car);

}

public void addCustomer(Customer customer) {

customers.add(customer);

}

public void rentCar(Car car, Customer customer, int days) {

if (car.isAvailable()) {

car.rent();

rentals.add(new Rental(car, customer, days));

} else {

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

}

}

public void returnCar(Car car) {

car.returnCar();

Rental rentalToRemove = null;

for (Rental rental : rentals) {

if (rental.getCar() == car) {

rentalToRemove = rental;

break;

}

}

if (rentalToRemove != null) {

rentals.remove(rentalToRemove);

} else {

System.out.println("Car was not rented.");

}

}

public void menu() {

Scanner scanner = new Scanner(System.in);

while (true) {

System.out.println("===== Car Rental System =====");

System.out.println("1. Rent a Car");

System.out.println("2. Return a Car");

System.out.println("3. Exit");

System.out.print("Enter your choice: ");

int choice = scanner.nextInt();

scanner.nextLine(); // Consume newline

if (choice == 1) {

System.out.println("\n== Rent a Car ==\n");

System.out.print("Enter your name: ");

String customerName = scanner.nextLine();

System.out.println("\nAvailable Cars:");

for (Car car : cars) {

if (car.isAvailable()) {

System.out.println(car.getCarId() + " - " + car.getBrand() + " " + car.getModel());

}

}

System.out.print("\nEnter the car ID you want to rent: ");

String carId = scanner.nextLine();

System.out.print("Enter the number of days for rental: ");

int rentalDays = scanner.nextInt();

scanner.nextLine(); // Consume newline

Customer newCustomer = new Customer("CUS" + (customers.size() + 1), customerName);

addCustomer(newCustomer);

Car selectedCar = null;

for (Car car : cars) {

if (car.getCarId().equals(carId) && car.isAvailable()) {

selectedCar = car;

break;

}

}

if (selectedCar != null) {

double totalPrice = selectedCar.calculatePrice(rentalDays);

System.out.println("\n== Rental Information ==\n");

System.out.println("Customer ID: " + newCustomer.getCustomerId());

System.out.println("Customer Name: " + newCustomer.getName());

System.out.println("Car: " + selectedCar.getBrand() + " " + selectedCar.getModel());

System.out.println("Rental Days: " + rentalDays);

System.out.printf("Total Price: $%.2f%n", totalPrice);

System.out.print("\nConfirm rental (Y/N): ");

String confirm = scanner.nextLine();

if (confirm.equalsIgnoreCase("Y")) {

rentCar(selectedCar, newCustomer, rentalDays);

System.out.println("\nCar rented successfully.");

} else {

System.out.println("\nRental canceled.");

}

} else {

System.out.println("\nInvalid car selection or car not available for rent.");

}

} else if (choice == 2) {

System.out.println("\n== Return a Car ==\n");

System.out.print("Enter the car ID you want to return: ");

String carId = scanner.nextLine();

Car carToReturn = null;

for (Car car : cars) {

if (car.getCarId().equals(carId) && !car.isAvailable()) {

carToReturn = car;

break;

}

}

if (carToReturn != null) {

Customer customer = null;

for (Rental rental : rentals) {

if (rental.getCar() == carToReturn) {

customer = rental.getCustomer();

break;

}

}

if (customer != null) {

returnCar(carToReturn);

System.out.println("Car returned successfully by " + customer.getName());

} else {

System.out.println("Car was not rented or rental information is missing.");

}

} else {

System.out.println("Invalid car ID or car is not rented.");

}

} else if (choice == 3) {

break;

} else {

System.out.println("Invalid choice. Please enter a valid option.");

}

}

System.out.println("\nThank you for using the Car Rental System!");

}

}

public class Main{

public static void main(String[] args) {

CarRentalSystem rentalSystem = new CarRentalSystem();

Car car1 = new Car("C001", "Toyota", "Camry", 60.0); // Different base price per day for each car

Car car2 = new Car("C002", "Honda", "Accord", 70.0);

Car car3 = new Car("C003", "Mahindra", "Thar", 150.0);

rentalSystem.addCar(car1);

rentalSystem.addCar(car2);

rentalSystem.addCar(car3);

rentalSystem.menu();

}

}