VEHICLE PARKING SYSTEM

CODING :

```
import java.sql.*;
import java.util.Scanner;


public class ParkingSystemApp {
    // Database connection details
    private static final String DB_URL = "jdbc:mysql://localhost:3306/parking_system";
    private static final String DB_USER = "root";
    private static final String DB_PASSWORD = "karthi@123";

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        ParkingSystem parkingSystem = new ParkingSystem();

        while (true) {
            System.out.println("\nVehicle Parking System");
            System.out.println("1. Enter Parking");
            System.out.println("2. Exit Parking");
            System.out.println("3. Exit Program");
            System.out.print("Choose an option: ");
            int choice = scanner.nextInt();
            scanner.nextLine(); // Consume newline

            try {
                if (choice == 1) {
                    System.out.print("Enter License Plate: ");
                    String licensePlate = scanner.nextLine();
                    parkingSystem.enterParking(licensePlate);
                } else if (choice == 2) {
                    System.out.print("Enter License Plate: ");
                    String licensePlate = scanner.nextLine();
```

```
                parkingSystem.exitParking(licensePlate);
            } else if (choice == 3) {
                System.out.println("Exiting program.");
                break;
            } else {
                System.out.println("Invalid option. Please try again.");
            }
        } catch (SQLException e) {
            System.out.println("Database error: " + e.getMessage());
        }
    }

    scanner.close();
}

// Database connection method
private static Connection getConnection() throws SQLException {
    return DriverManager.getConnection(DB_URL, DB_USER, DB_PASSWORD);
}

// Inner class for the Parking System
public static class ParkingSystem {

    // Method to enter parking (i.e., park a vehicle)
    public void enterParking(String licensePlate) throws SQLException {
        int availableSpotId = findAvailableSpot();
        if (availableSpotId == -1) {
            System.out.println("No available parking spots. Please try again later.");
            return;
        }

        // Reserve the spot and register the vehicle
```

```
            reserveSpot(availableSpotId);
            registerVehicle(licensePlate, availableSpotId);
            System.out.println("Vehicle " + licensePlate + " has been successfully parked at spot " +
availableSpotId + ".");
        }

        // Method to exit parking (i.e., vehicle exits)
        public void exitParking(String licensePlate) throws SQLException {
            String query = "UPDATE Vehicles SET exit_time = NOW() WHERE license_plate = ? AND
exit_time IS NULL";
            try (Connection conn = getConnection();
                 PreparedStatement stmt = conn.prepareStatement(query)) {
                stmt.setString(1, licensePlate);
                int rowsUpdated = stmt.executeUpdate();

                if (rowsUpdated > 0) {
                    // Free the spot once the vehicle exits
                    freeSpot(licensePlate);
                    System.out.println("Vehicle " + licensePlate + " has exited the parking lot. Spot is now
free.");
                } else {
                    System.out.println("Vehicle " + licensePlate + " not found or already exited.");
                }
            }
        }

        // Method to find an available parking spot
        private int findAvailableSpot() throws SQLException {
            String query = "SELECT spot_id FROM ParkingSpots WHERE is_available = TRUE LIMIT
1";
            try (Connection conn = getConnection();
                 PreparedStatement stmt = conn.prepareStatement(query);
                 ResultSet rs = stmt.executeQuery()) {
                if (rs.next()) {
```

```
            return rs.getInt("spot_id");
        }
        return -1; // No available spots
    }
}

// Method to reserve a parking spot
private void reserveSpot(int spotId) throws SQLException {
    String query = "UPDATE ParkingSpots SET is_available = FALSE WHERE spot_id = ?";
    try (Connection conn = getConnection();
         PreparedStatement stmt = conn.prepareStatement(query)) {
        stmt.setInt(1, spotId);
        stmt.executeUpdate();
    }
}

// Method to free a parking spot when a vehicle exits
private void freeSpot(String licensePlate) throws SQLException {
    String query = "UPDATE ParkingSpots SET is_available = TRUE WHERE spot_id =
(SELECT spot_id FROM Vehicles WHERE license_plate = ? AND exit_time IS NOT NULL)";
    try (Connection conn = getConnection();
         PreparedStatement stmt = conn.prepareStatement(query)) {
        stmt.setString(1, licensePlate);
        stmt.executeUpdate();
    }
}

// Method to register a vehicle in the system
private void registerVehicle(String licensePlate, int spotId) throws SQLException {
    String query = "INSERT INTO Vehicles (license_plate, spot_id, entry_time) VALUES (?, ?,
NOW())";
    try (Connection conn = getConnection();
         PreparedStatement stmt = conn.prepareStatement(query)) {
```

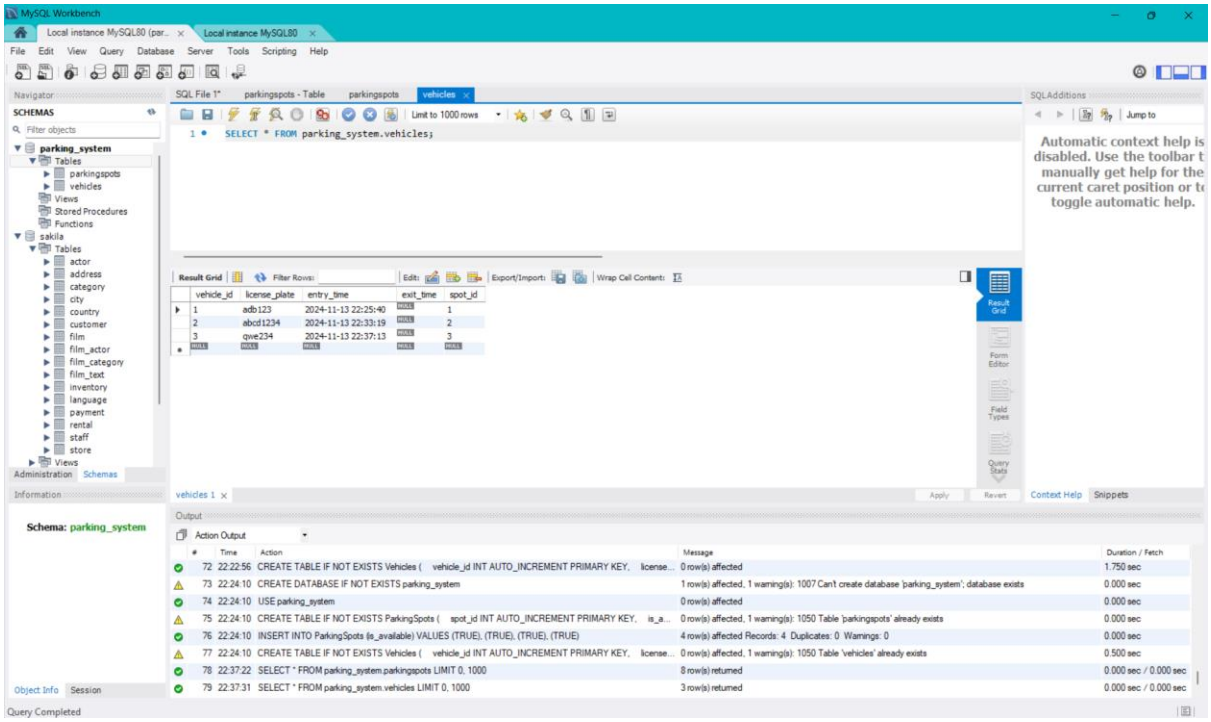
```
            stmt.setString(1, licensePlate);
            stmt.setInt(2, spotId);
            stmt.executeUpdate();
        }
    }
  }
}
```

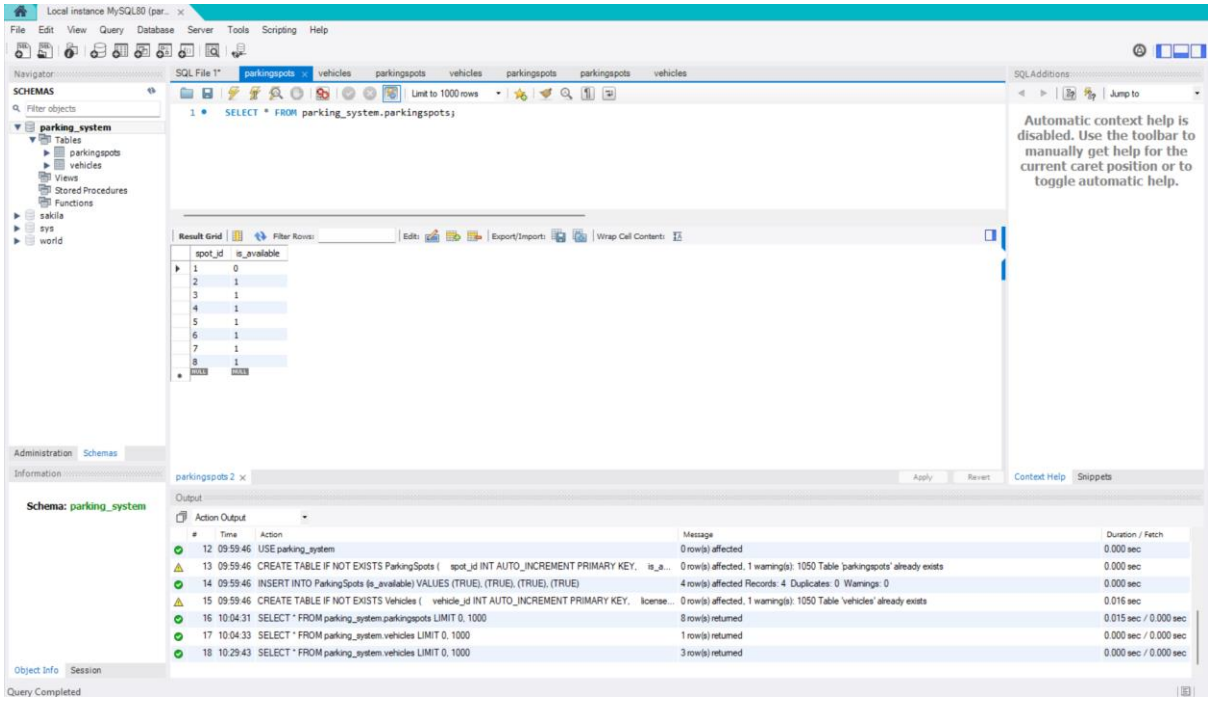
DATABASE DESIGN:

```
CREATE DATABASE IF NOT EXISTS parking_system;
DROP DATABASE parking_system;
CREATE DATABASE parking_system;
USE parking_system;
CREATE TABLE IF NOT EXISTS ParkingSpots (
   spot_id INT PRIMARY KEY AUTO_INCREMENT,
   is_available BOOLEAN DEFAULT TRUE
);
CREATE TABLE IF NOT EXISTS Vehicles (
   vehicle_id INT PRIMARY KEY AUTO_INCREMENT,
   license_plate VARCHAR(20) NOT NULL,
   entry_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
   exit_time TIMESTAMP NULL,
   spot_id INT,
   FOREIGN KEY (spot_id) REFERENCES ParkingSpots(spot_id)
);
```

OUTPUT:

DATABASE DESIGN:





```
PS D:\project> java -cp ".;C:\Users\Karthiyayini\Downloads\mysql-connector-j-9.1.0\mysql-co

Vehicle Parking System
1. Enter Parking
2. Exit Parking
3. Exit Program
Choose an option: 1
Enter License Plate: wer43
Spot 4 has been reserved.
Vehicle wer43 has been registered.
Vehicle wer43 has been successfully parked at spot 4.

Vehicle Parking System
1. Enter Parking
2. Exit Parking
3. Exit Program
Choose an option: 1
Enter License Plate: TN93A4295
Spot 5 has been reserved.
Vehicle TN93A4295 has been registered.
Vehicle TN93A4295 has been successfully parked at spot 5.

Vehicle Parking System
1. Enter Parking
2. Exit Parking
3. Exit Program
Choose an option: 2
Enter License Plate: TN93A4295
Spot has been freed.
Vehicle TN93A4295 has exited the parking lot. Spot is now free.
```

```
PS D:\project> java -cp ".;C:\Users\Karthiyayini\Downloads\mysql-connector-j-9.1.0\mysql-connector-j-9.1.0\mysql-connector-j-9.1.0.jar" ParkingSystemApp

Vehicle Parking System
1. Enter Parking
2. Exit Parking
3. Exit Program
Choose an option: 1
Enter License Plate: wer43
Spot 4 has been reserved.
Vehicle wer43 has been registered.
Vehicle wer43 has been successfully parked at spot 4.

Vehicle Parking System
1. Enter Parking
2. Exit Parking
3. Exit Program
Choose an option: 1
Enter License Plate: TN93A4295
Spot 5 has been reserved.
Vehicle TN93A4295 has been registered.
Vehicle TN93A4295 has been successfully parked at spot 5.

Vehicle Parking System
1. Enter Parking
2. Exit Parking
3. Exit Program
Choose an option: 2
Enter License Plate: TN93A4295
Spot has been freed.
Vehicle TN93A4295 has exited the parking lot. Spot is now free.

Vehicle Parking System
1. Enter Parking
2. Exit Parking
3. Exit Program
Choose an option: 3
Exiting program.
```