

Department of Computer Science and Engineering

BENGALURU, KARNATAKA, INDIA.

B. TECH. (CSE) - V SEMESTER

Aug. – Dec. 2024

UE22CS351A – Database Management System PROJECT REPORT

ON

Online Vehicle Booking Management System

**Submitted by:**

**PES1UG23CS846 - Sanket Muttur**

**PES1UG23CS847 - Veeresh Amaragatti**

Abstract:

The **Online Vehicle Booking Management System** (OVBMS) is a web-based platform designed to facilitate vehicle rentals, offering users a streamlined process to search, book, and manage vehicle reservations. The system provides a user-friendly interface for customers to register, browse available vehicles, make bookings, and receive confirmations, while administrators are empowered with tools to manage vehicle listings, user accounts, and bookings. The OVBMS leverages a MySQL database for data management, with RESTful APIs ensuring smooth data communication. Secure data transmission is maintained via HTTPS. This platform addresses the need for a reliable, secure, and scalable vehicle rental solution that enhances user experience and operational efficiency for both customers and administrators.

Table of Contents

[1. User Requirement Specification 1](#_Toc183698891)

[1.1. Purpose of the Project 1](#_Toc183698892)

[1.2. Scope of the Project 1](#_Toc183698893)

[1.3. Detailed Description of the Project 2](#_Toc183698894)

[2. List of Softwares/Tools/Programming languages used 3](#_Toc183698895)

[3. ER Diagram 4](#_Toc183698896)

[4. Relational Schema 4](#_Toc183698897)

[5. DDL Commands for creating Tables in OVBMS 5](#_Toc183698898)

[6. CRUD operation Screenshots 7](#_Toc183698899)

[6.1. CREATE: Adding Vehicle 7](#_Toc183698900)

[6.2. READ: Vehicle Listing 10](#_Toc183698901)

[6.3. UPDATE: Updating Vehicle Information 11](#_Toc183698902)

[6.4. DELETE: Deleting the Vehicle 15](#_Toc183698903)

[7. List of functionalities/features of the application 17](#_Toc183698904)

[8. Triggers, Procedures/Functions, Nested query, Join, Aggregate queries 26](#_Toc183698905)

[8.1. Procedure 1 26](#_Toc183698906)

[8.2. Procedure 2 27](#_Toc183698907)

[8.3. Trigger 28](#_Toc183698908)

[8.4. Event 29](#_Toc183698909)

[8.5. Nested Query & Aggregate Query 30](#_Toc183698910)

[8.6. Join Query 31](#_Toc183698911)

[9. Code snippets for invoking the Procedures/Functions/ Trigger 32](#_Toc183698912)

[9.1. Invoking “BookVehicle” Procedure: 32](#_Toc183698913)

[9.2. Invoking “HandleBookingRequest” Procedure: 32](#_Toc183698914)

[10. SQL queries (Create, Insert, Triggers, Procedures/ Functions, Nested query, Join, Aggregate queries) used in the project in the form of .sql file 33](#_Toc183698915)

[11. GitHub Repo Link 41](#_Toc183698916)

# 1. User Requirement Specification

## 1.1. Purpose of the Project

The purpose of this project is to design and implement a Vehicle Rental Management System aimed at providing an integrated platform for customers to reserve and rent vehicles efficiently. The system will allow customers to browse available vehicles, make reservations, while enabling managers to maintain the vehicle inventory, approve or reject booking requests.

This system will automate the vehicle rental process, making it more accessible for customers and easier to manage for rental service providers. By replacing manual booking, the system enhances both operational efficiency and customer satisfaction.

## 1.2. Scope of the Project

The scope of the **Vehicle Rental Management System** includes the development and deployment of a web-based application that facilitates the end-to-end vehicle rental process. It covers the functionalities required by both customers and managers, ensuring that all aspects of the rental service are effectively managed.

#### Key Features:

* **Customer Functions**:
  + **Account Registration**: Customers can create accounts, enter personal details such as name, password, email, and license ID.
  + **Vehicle Reservation**: Customers can view available vehicles, reserve vehicles for specific dates, and track their bookings.
* **Manager Functions**:
  + **Vehicle Management**: Managers can register new vehicles into the system, including specifications like seating capacity, fuel type, model year, and rental price.
  + **Booking Management**: Managers are responsible for approving or rejecting customer booking requests based on vehicle availability and service conditions.

The system is built to accommodate a wide range of rental operations, from small fleets to large-scale car rental companies. It is designed to be scalable and extendable, with potential future additions such as online payment integration, vehicle maintenance tracking, and automated notifications.

## 1.3. Detailed Description of the Project

The **Vehicle Rental Management System** consists of several interrelated components, designed to ensure smooth and efficient operation of a vehicle rental service. Below is a detailed breakdown of the system’s entities and processes:

#### The Entity-Relationship (ER) Diagram illustrates the core entities in the system and their interactions. These include:

* **Manager**:  
   The manager is responsible for overseeing vehicle registrations, approving or rejecting customer bookings. Managers interact with the system through functionalities such as:
  + **Vehicle Registration**: Add and manage vehicles in the rental fleet.
  + **Booking Request Approval**: Approve or reject booking requests from customers.
* **Customer**:  
   Customers can register accounts, view vehicle availability, make bookings. The customer interacts with the system via:
  + **Registration**: Provide personal details, including name, email, password, and license ID.
  + **Vehicle Reservation**: Browse available vehicles and place reservations by specifying booking dates.
* **Vehicle**:  
   The system keeps detailed records of each vehicle in the fleet, including its name, model year, seating capacity, fuel type, price per day, and other relevant attributes.
* **Booking Request**:  
   Customers place booking requests for specific vehicles, specifying a date range. Each request is reviewed and either approved or rejected by a manager. The booking process includes:
  + **Request Date**: The date when the booking request was submitted.
  + **Rental Period**: The start and end dates of the booking.
  + **Approval Status**: Indicates whether the request has been approved or rejected by the manager.

#### Functional Requirements for the Online Vehicle Booking Management System:

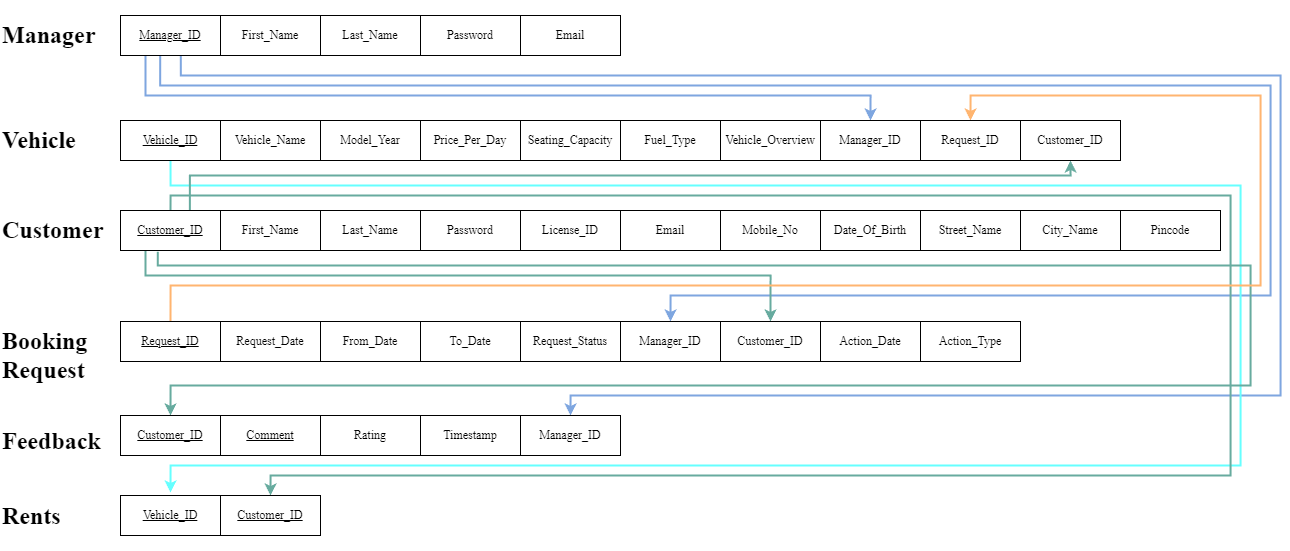
1. **Customer Registration and Login**
   * Customers can create accounts and log in using credentials, including personal details like name, email, password, and license ID.
2. **Vehicle Browsing and Availability Check**
   * Customers can browse the available vehicles and check their availability for specific dates.
3. **Vehicle Reservation**
   * Customers can reserve vehicles for specific dates, selecting from the available fleet.
4. **Booking Management**
   * Managers can view booking requests, approve or reject them based on vehicle availability and service conditions.
5. **Vehicle Inventory Management**
   * Managers can add, update, or remove vehicle details, including attributes like seating capacity, fuel type, model year, and rental price.

# 2. List of Softwares/Tools/Programming languages used

1. **Frontend**: ReactJS
   * For building the user interface of the web application.
2. **Backend**: Node.js, Express.js
   * For handling server-side logic and APIs.
3. **Database**: MySQL
   * To store and manage data such as user information, vehicle details, and bookings.
4. **IDE**: Visual Studio Code (Visual Studio Code)
   * Used for coding and project development.
5. **Version Control**: GitHub
   * For version control and collaborative development.

# 3. ER Diagram

# 4. Relational Schema



# 5. DDL Commands for creating Tables in OVBMS

#### Customer Table:

CREATE TABLE `customer` (

  `email` varchar(45) NOT NULL,

  `name` varchar(45) NOT NULL,

  `password` varchar(255) NOT NULL,

  `license\_id` varchar(16) NOT NULL,

  `mobile\_no` varchar(10) NOT NULL,

  `dob` date NOT NULL,

  PRIMARY KEY (`email`)

)

#### Booking Request Table:

CREATE TABLE `booking\_requests` (

  `request\_id` varchar(255) NOT NULL,

  `request\_date` datetime NOT NULL,

  `from\_date` date NOT NULL,

  `to\_date` date NOT NULL,

  `request\_status` varchar(15) NOT NULL DEFAULT 'Pending',

  `customer\_id` varchar(45) NOT NULL,

  `vehicle\_id` varchar(10) NOT NULL,

  `manager\_id` varchar(45) DEFAULT NULL,

  `action\_type` varchar(45) DEFAULT NULL,

  `action\_date` datetime DEFAULT NULL,

  PRIMARY KEY (`request\_id`),

  KEY `fk\_manager\_id\_idx` (`manager\_id`),

  KEY `fk\_br\_manager\_id\_idx` (`manager\_id`),

  KEY `fk\_customer\_id\_idx` (`customer\_id`),

  KEY `fk\_vehicle\_id\_idx` (`vehicle\_id`),

  CONSTRAINT `fk\_br\_manager\_id` FOREIGN KEY (`manager\_id`) REFERENCES `manager` (`email`) ON UPDATE CASCADE,

  CONSTRAINT `fk\_customer\_id` FOREIGN KEY (`customer\_id`) REFERENCES `customer` (`email`) ON UPDATE CASCADE,

  CONSTRAINT `fk\_vehicle\_id` FOREIGN KEY (`vehicle\_id`) REFERENCES `vehicles` (`license\_no`) ON UPDATE CASCADE

)

#### Vehicles Table:

CREATE TABLE `vehicles` (

  `license\_no` varchar(10) NOT NULL,

  `vehicle\_name` varchar(45) NOT NULL,

  `model\_year` year NOT NULL,

  `price\_per\_day` decimal(10,0) NOT NULL,

  `seating\_capacity` int NOT NULL,

  `fuel\_type` varchar(10) NOT NULL,

  `vehicle\_image` longtext,

  `vehicle\_overview` varchar(225) DEFAULT NULL,

  `manager\_id` varchar(45) NOT NULL,

  `availability` varchar(45) NOT NULL DEFAULT 'Available',

  PRIMARY KEY (`license\_no`),

  KEY `fk\_manager\_id\_idx` (`manager\_id`),

  CONSTRAINT `fk\_manager\_id` FOREIGN KEY (`manager\_id`) REFERENCES `manager` (`email`) ON UPDATE CASCADE

)

#### Manager Table:

CREATE TABLE `manager` (

  `email` varchar(45) NOT NULL,

  `name` varchar(45) NOT NULL,

  `password` varchar(255) NOT NULL,

  PRIMARY KEY (`email`)

)

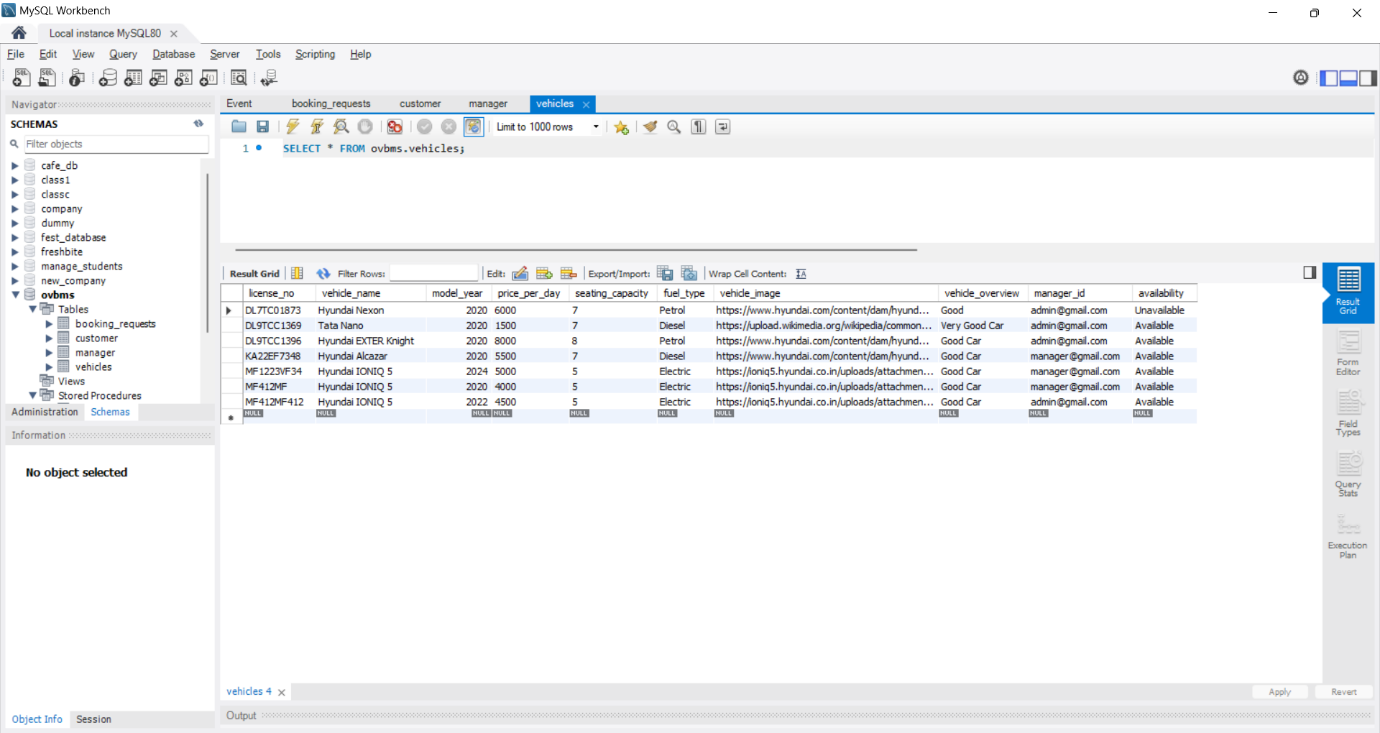
# 6. CRUD operation Screenshots

## 6.1. CREATE: Adding Vehicle

#### 6.1.1. Adding Vehicle code

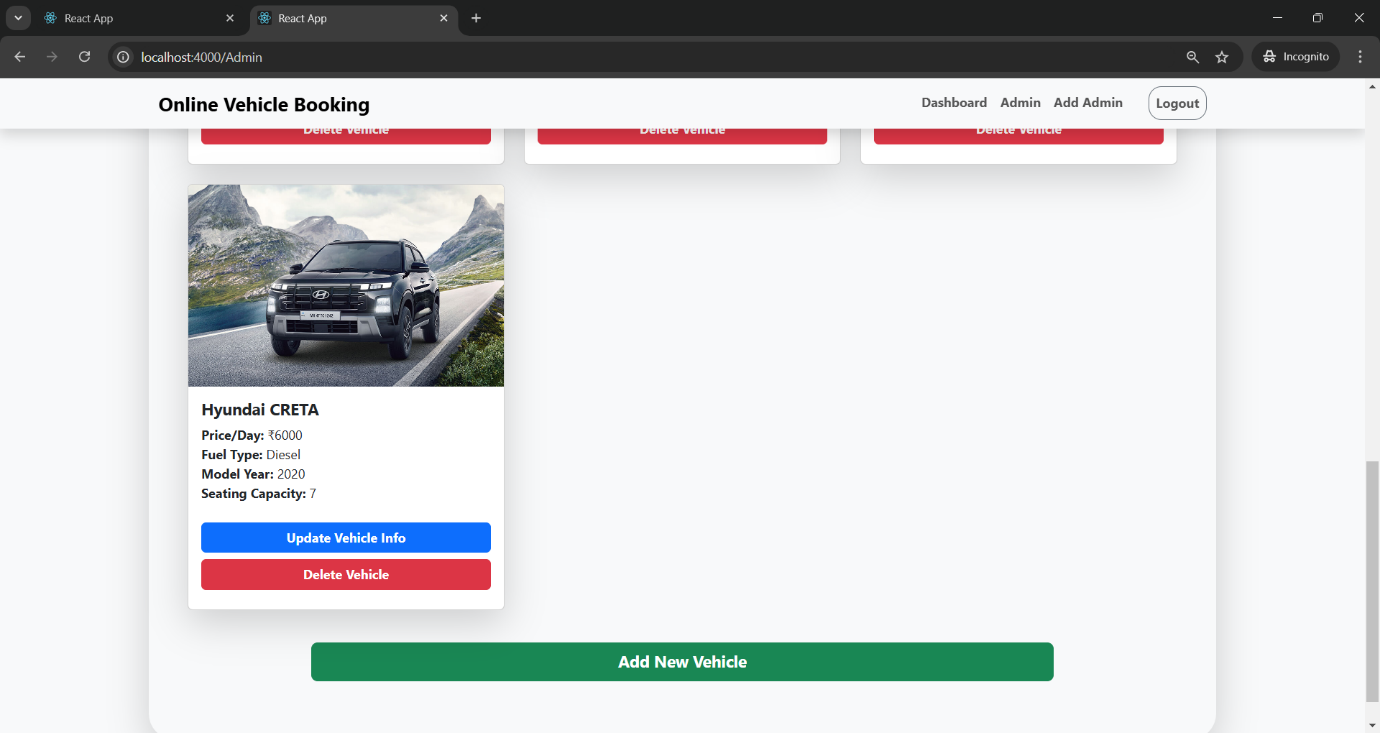


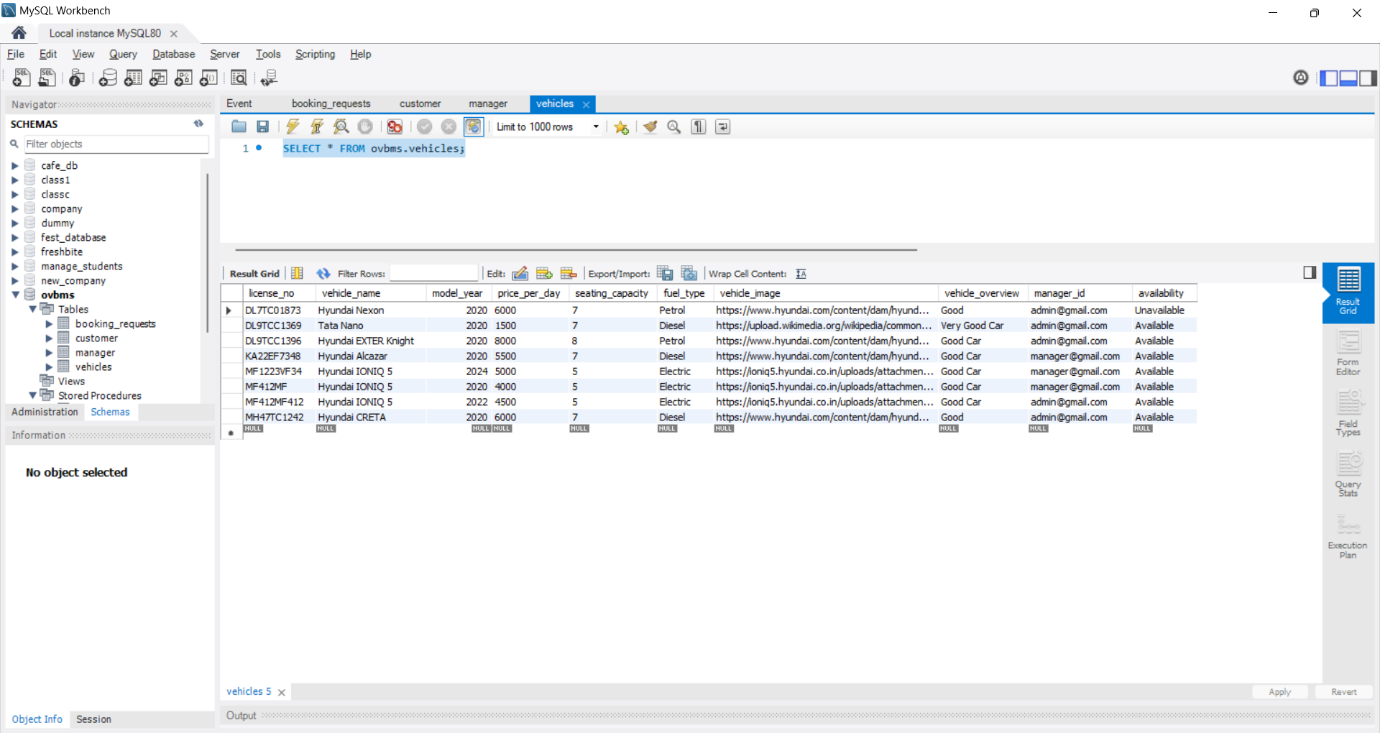
#### 6.1.2. Before Adding



#### 6.1.3. Adding the Vehicle

#### 6.1.4. After Adding the Vehicle





## 6.2. READ: Vehicle Listing

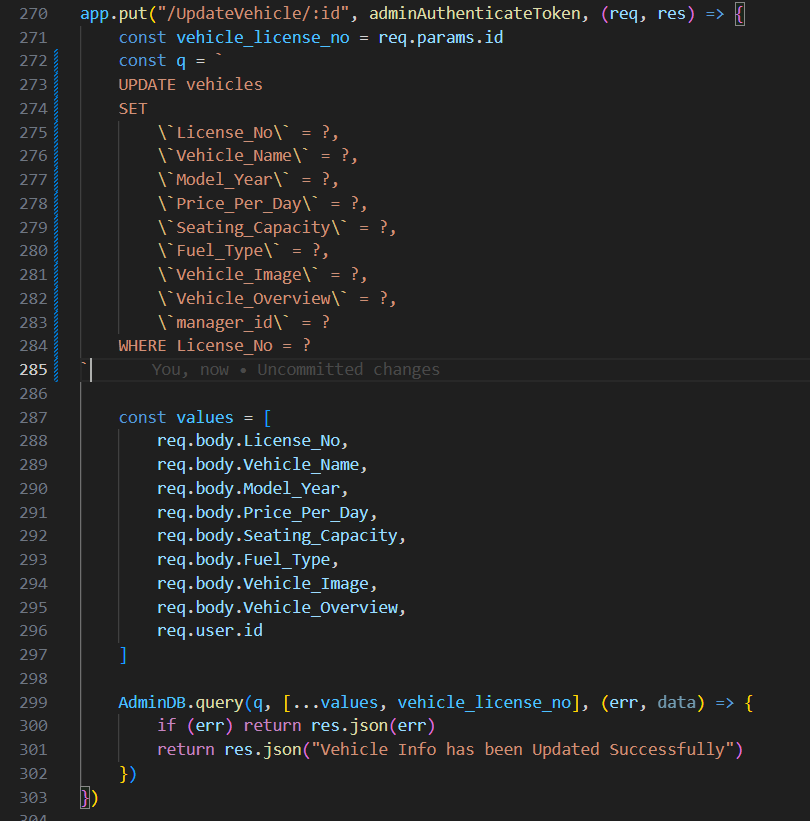
#### 6.2.1. Vehicle Listing Code

#### 6.2.2. Vehicle Table

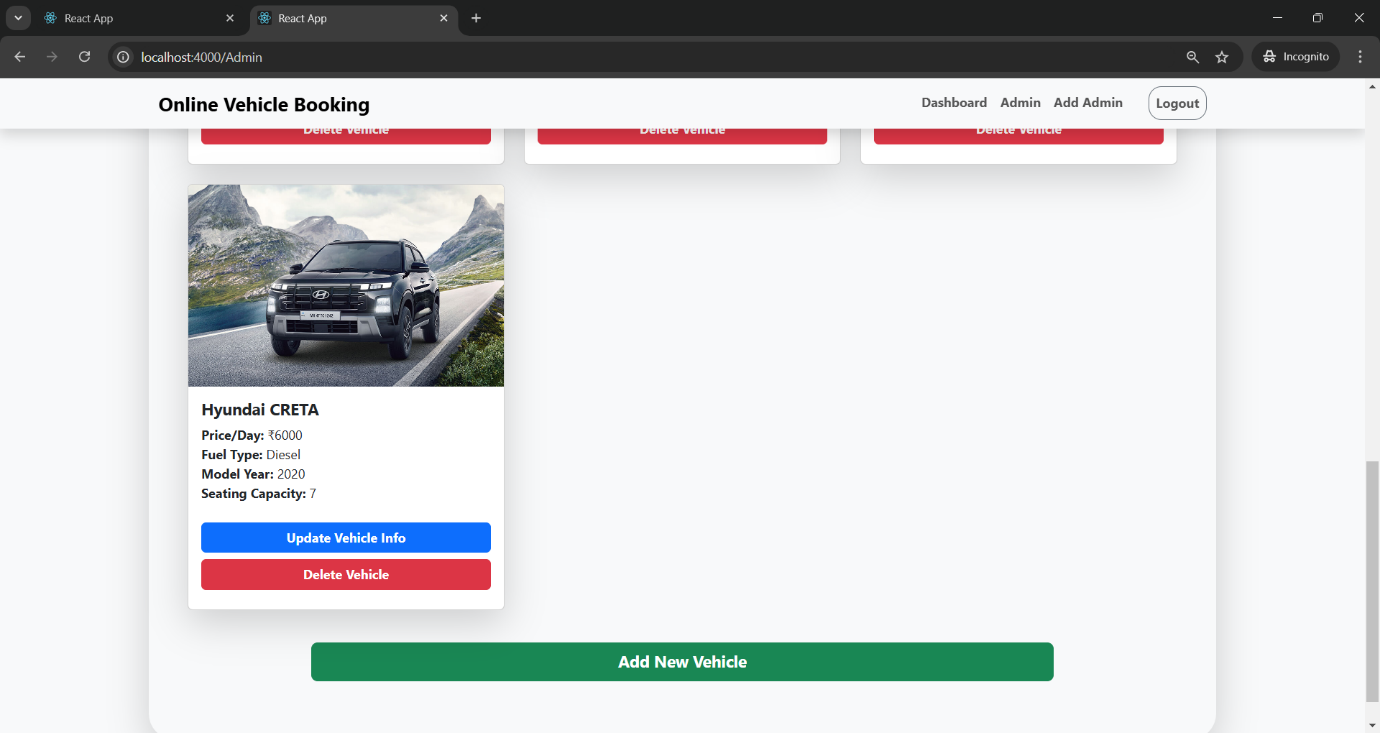
#### 6.2.3. Vehicle Listing in UI

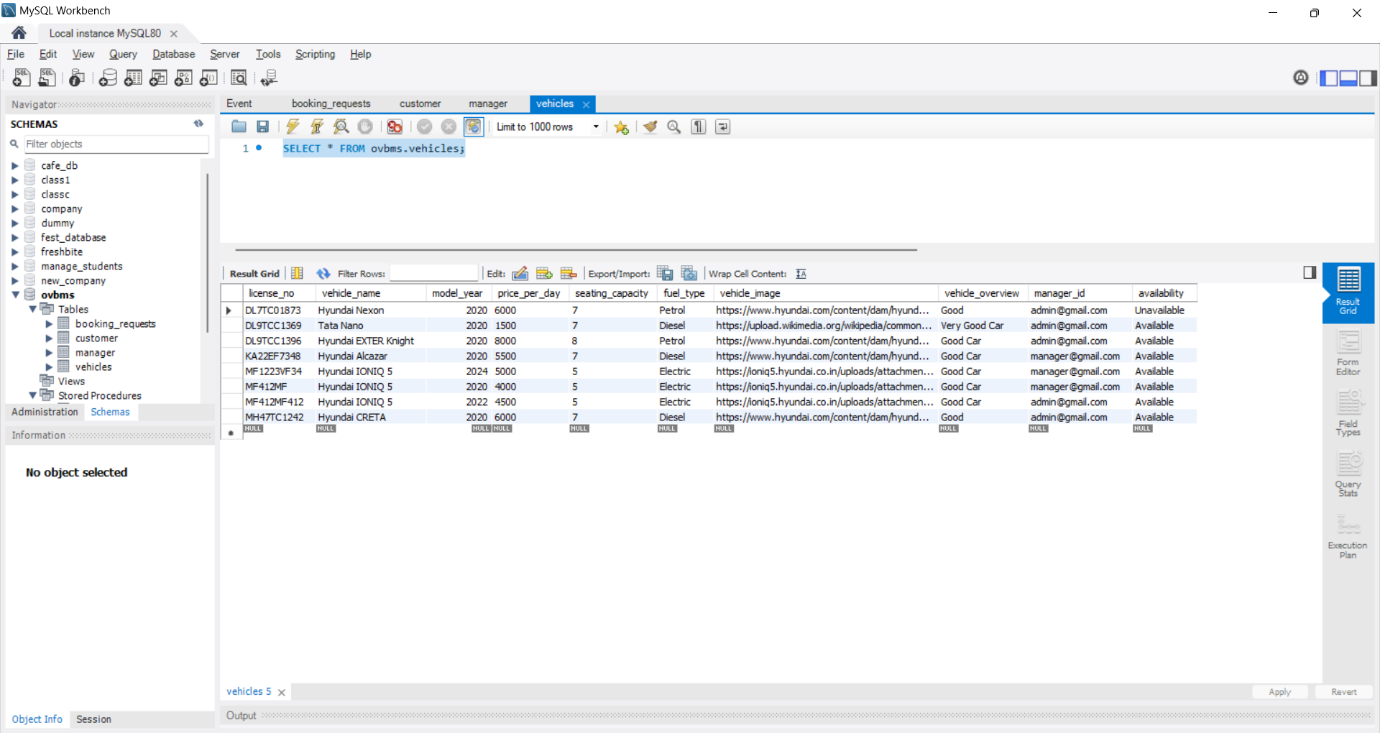
## 6.3. UPDATE: Updating Vehicle Information

#### 6.3.1. Vehicle Update Code

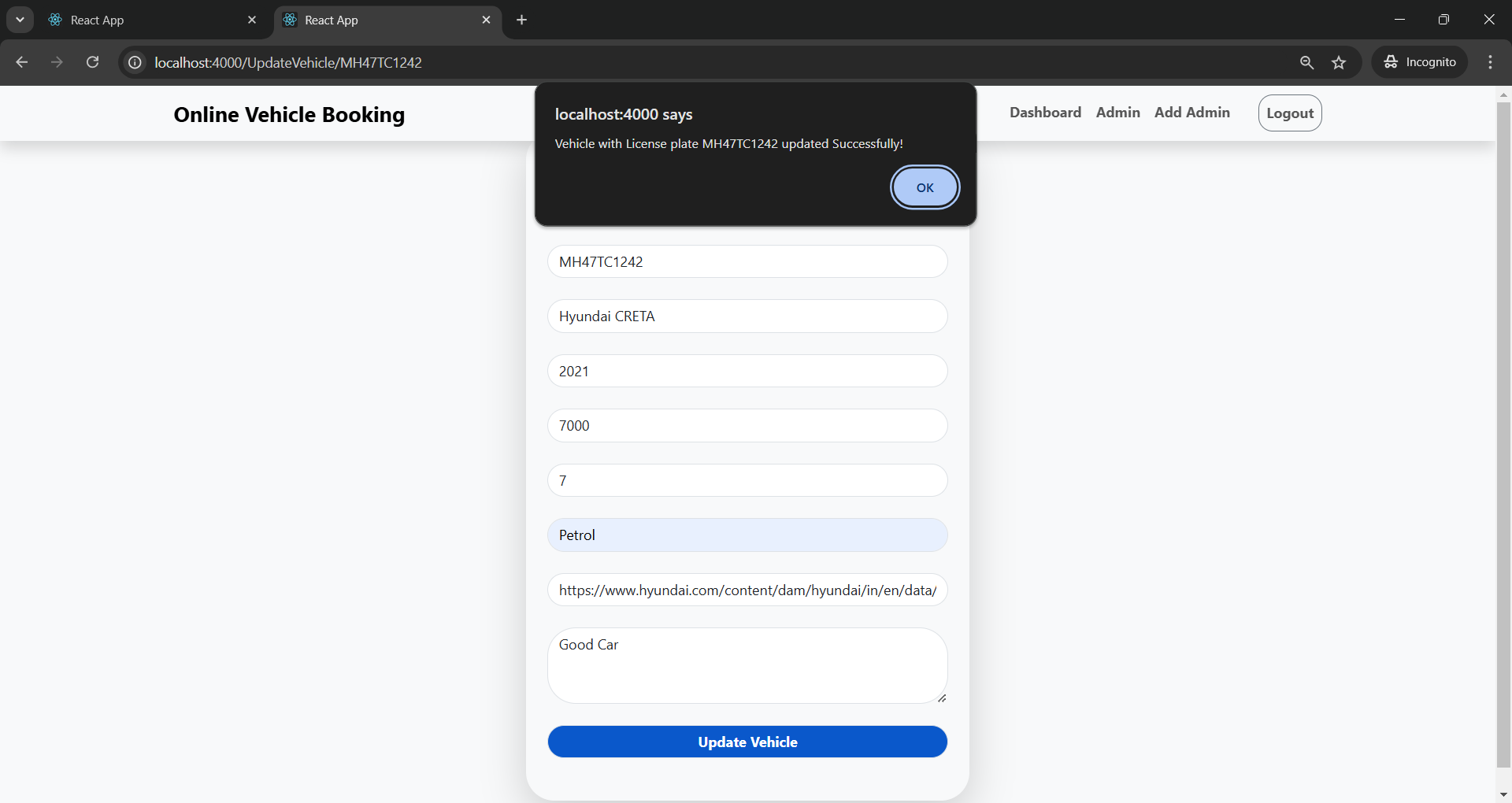


#### 6.3.2. Before Update

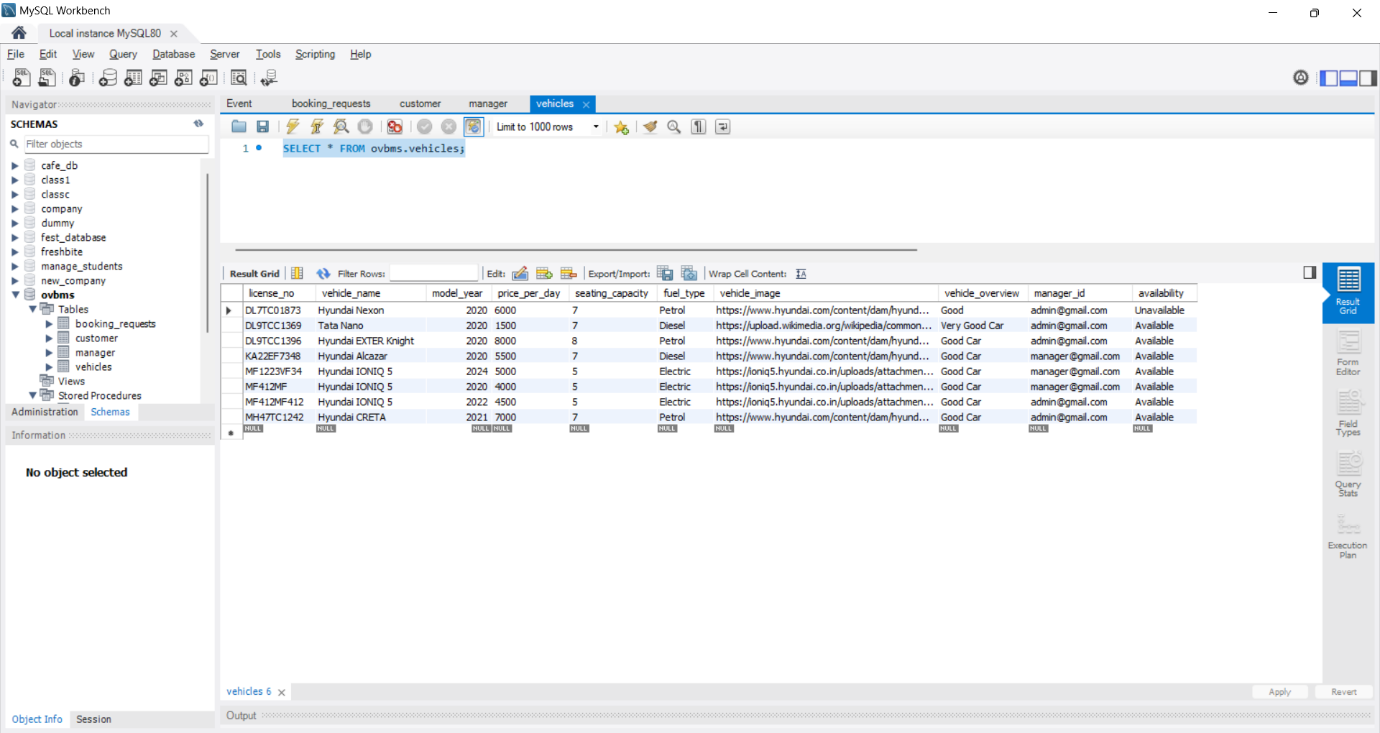




#### 6.3.3. Updating Vehicle Info



#### 6.3.4. After Update



## 6.4. DELETE: Deleting the Vehicle

#### 6.4.1. Vehicle Deletion Code

#### 6.4.2. Before Deletion

#### 6.4.3. Deleting the Vehicle

#### 6.4.4. After Deletion

# 7. List of functionalities/features of the application

#### 7.1. Admin Sign In

#### 7.2. Vehicle Listing at Admin Side

#### 7.3. Adding Vehicle at Admin side

#### 7.4. Updating Vehicle Info at Admin side

#### 7.5. Deleting Vehicle at Admin side

#### 7.6. Admin Dashboard

#### 7.7. Adding different Admin at Admin side

#### 7.8. Vehicle Listing at User side before Sign In

#### 7.9. User Sign Up

#### 7.10. User Sign In

#### 7.11. Vehicle Listing at User side after Sign In

#### 7.12. Vehicle Booking

#### 7.13. Vehicle Listing after Booking

#### 7.14. Vehicle Booking Requests in Admin Dashboard

#### 7.15. Vehicle Booking Requests in Admin Dashboard after handling the Booking Request

#### 7.16. Vehicle Listing at Admin side after handling the Booking Request

#### 7.17. Vehicle Listing at User side after handling the Booking Request

# 8. Triggers, Procedures/Functions, Nested query, Join, Aggregate queries

## 8.1. Procedure 1

#### Procedure for Vehicle Booking:

CREATE DEFINER=`root`@`localhost` PROCEDURE `HandleBookingRequest`(

in requestId varchar(255),

in action varchar(15),

in managerId varchar(45))

BEGIN

if action = 'Approve' then

update booking\_requests set

request\_status = 'Approved',

action\_type = 'Approve',

action\_date = curdate(),

manager\_id = managerId

where request\_id = requestId;

elseif action = 'Reject' then

update booking\_requests set

request\_status = 'Rejected',

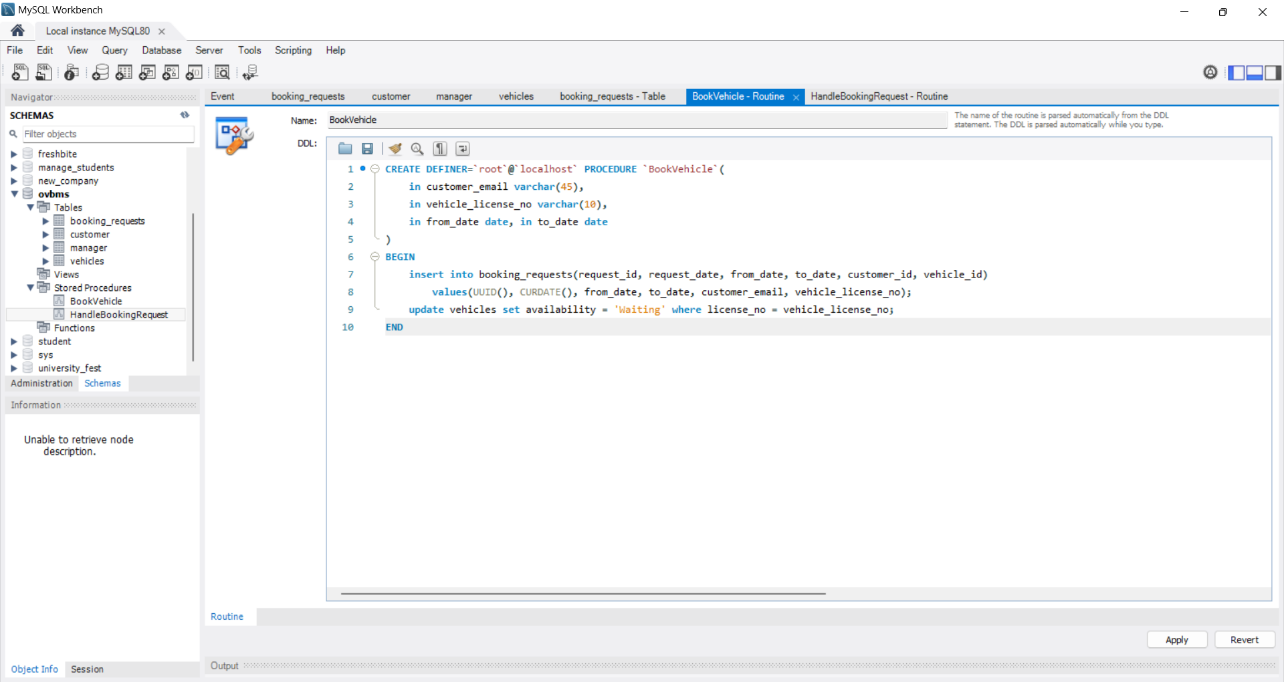
action\_type = 'Reject',

action\_date = curdate(),

manager\_id = managerId where request\_id = requestId;

END IF;

END ;;



## 8.2. Procedure 2

#### Procedure for handling the Booking Requests:

CREATE DEFINER=`root`@`localhost` PROCEDURE `HandleBookingRequest`(

in requestId varchar(255),

in action varchar(15),

in managerId varchar(45))

BEGIN

if action = 'Approve' then

update booking\_requests set

request\_status = 'Approved',

action\_type = 'Approve',

action\_date = curdate(),

manager\_id = managerId

where request\_id = requestId;

elseif action = 'Reject' then

update booking\_requests set

request\_status = 'Rejected',

action\_type = 'Reject',

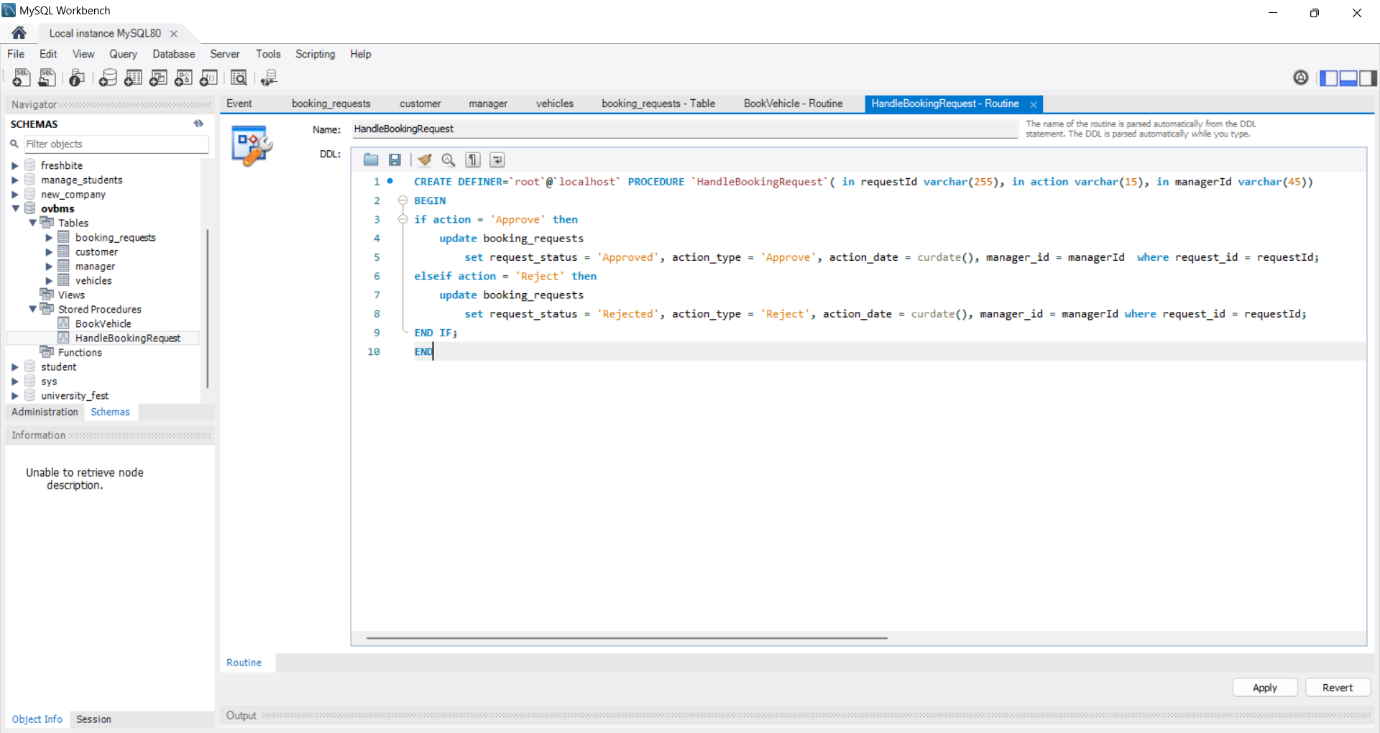
action\_date = curdate(),

manager\_id = managerId

where request\_id = requestId;

END IF;

END ;;



## 8.3. Trigger

#### Trigger for handling the data in “Vehicles” tables when the action is taken on Booking Request (after update on “Booking\_Requests” table):

CREATE DEFINER=`root`@`localhost` TRIGGER `booking\_requests\_AFTER\_UPDATE`

AFTER UPDATE ON `booking\_requests`

FOR EACH ROW

BEGIN

declare other\_bookings int;

if new.request\_status = 'Rejected'

and old.request\_status = 'Pending' then

update vehicles set availability = 'Available'

where license\_no = new.vehicle\_id;

  END if;

if new.request\_status = 'Approved'

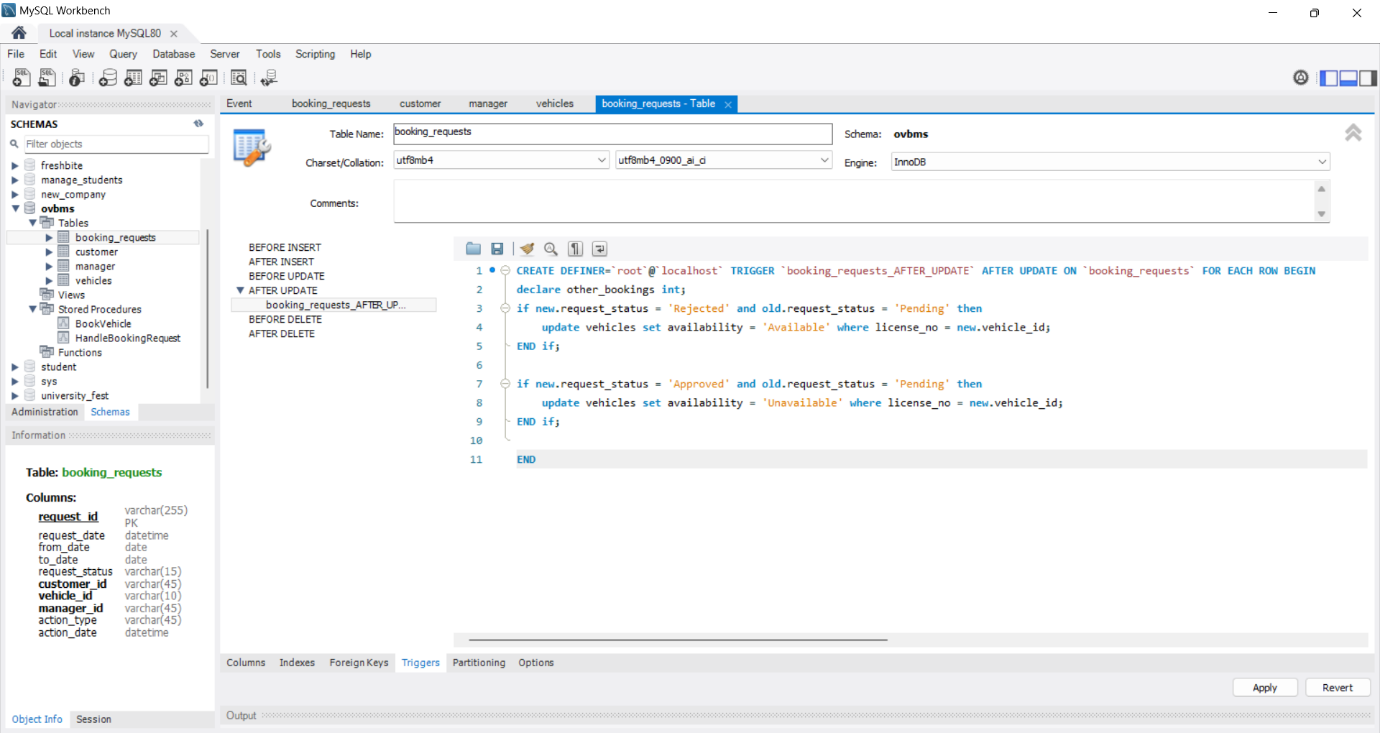
and old.request\_status = 'Pending' then

update vehicles set availability = 'Unavailable'

where license\_no = new.vehicle\_id;

END if;

END



## 8.4. Event

#### Event for handling the data in “Booking\_Requests” table and “Vehicles” table whenever the booking span of the vehicle is completed (checks everyday):

delimiter $$

CREATE DEFINER=`root`@`localhost` EVENT `CheckCompletedBookings`

ON SCHEDULE EVERY 1 DAY

STARTS CURRENT\_TIMESTAMP

DO

BEGIN

UPDATE vehicles SET availability = 'Available'

WHERE license\_no IN (

SELECT vehicle\_id

FROM booking\_requests

WHERE request\_status = 'Approved'

AND to\_date < CURDATE());

UPDATE booking\_requests

SET request\_status = 'Completed'

WHERE request\_status = 'Approved'

AND to\_date < CURDATE();

END $$

delimiter ;

## 8.5. Nested Query & Aggregate Query

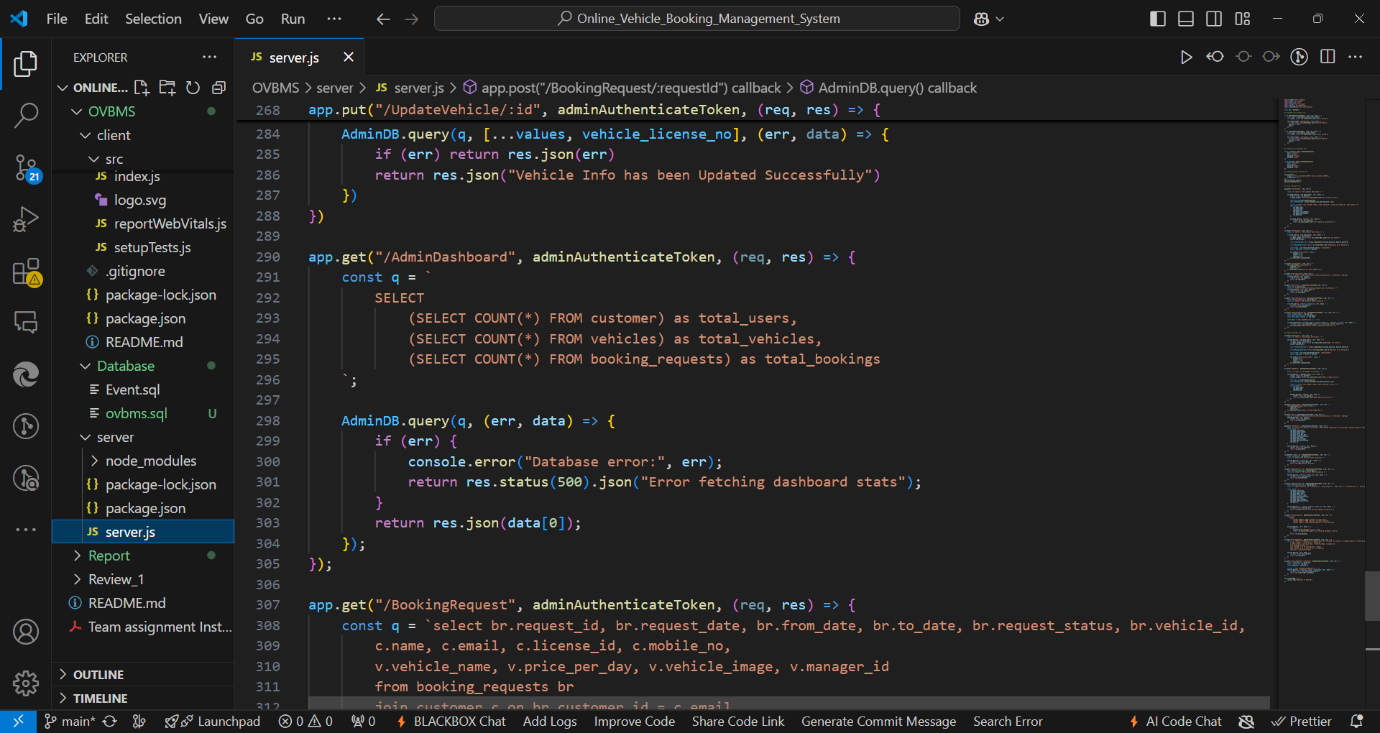
#### Nested Query & Aggregate Query in the Backend used for Admin Dashboard

SELECT

(SELECT COUNT(\*) FROM customer) as total\_users,

(SELECT COUNT(\*) FROM vehicles) as total\_vehicles,

(SELECT COUNT(\*) FROM booking\_requests) as total\_bookings;



## 8.6. Join Query

#### Join Query for Listing the Booking Requests in Admin Dashboard with Customer details and Vehicle details

select

br.request\_id, br.request\_date, br.from\_date, br.to\_date, br.request\_status, br.vehicle\_id,

c.name, c.email, c.license\_id, c.mobile\_no,

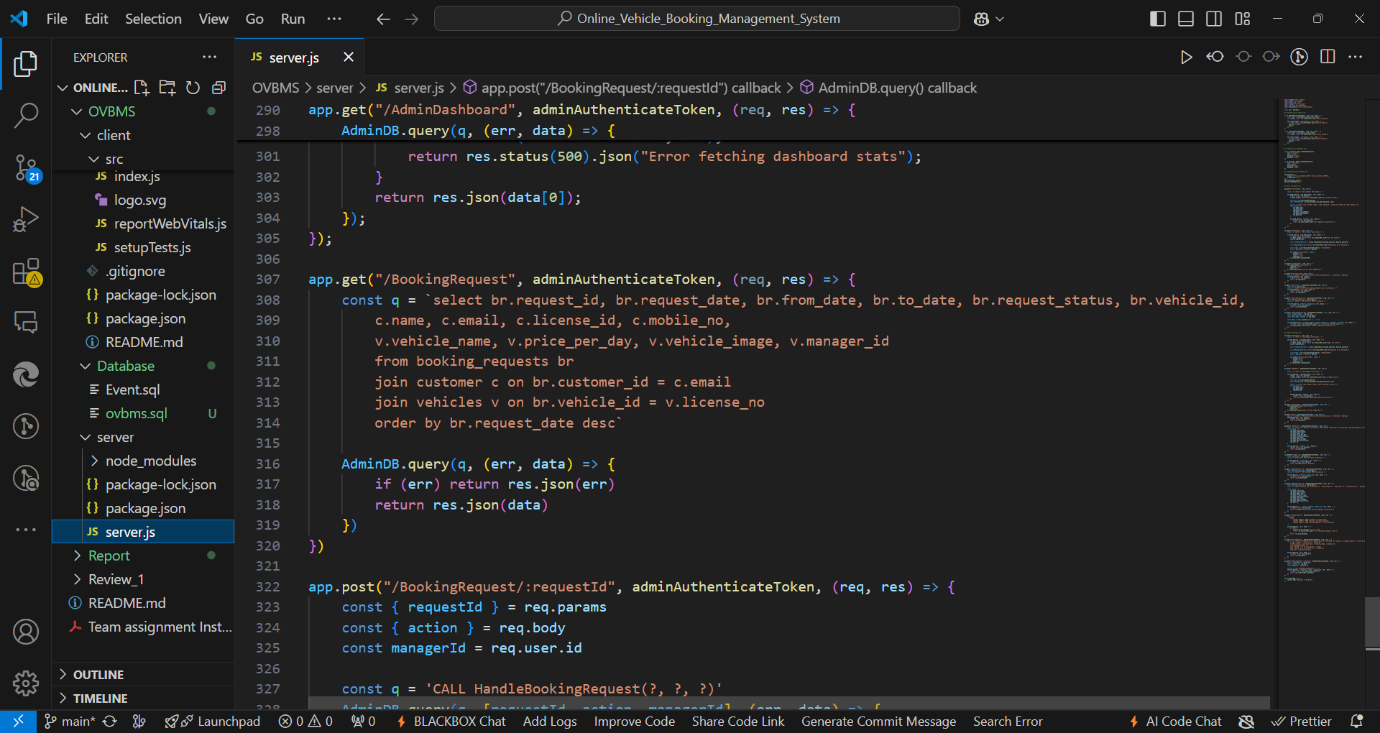
v.vehicle\_name, v.price\_per\_day, v.vehicle\_image, v.manager\_id

from booking\_requests br

join customer c on br.customer\_id = c.email

join vehicles v on br.vehicle\_id = v.license\_no

order by br.request\_date desc;

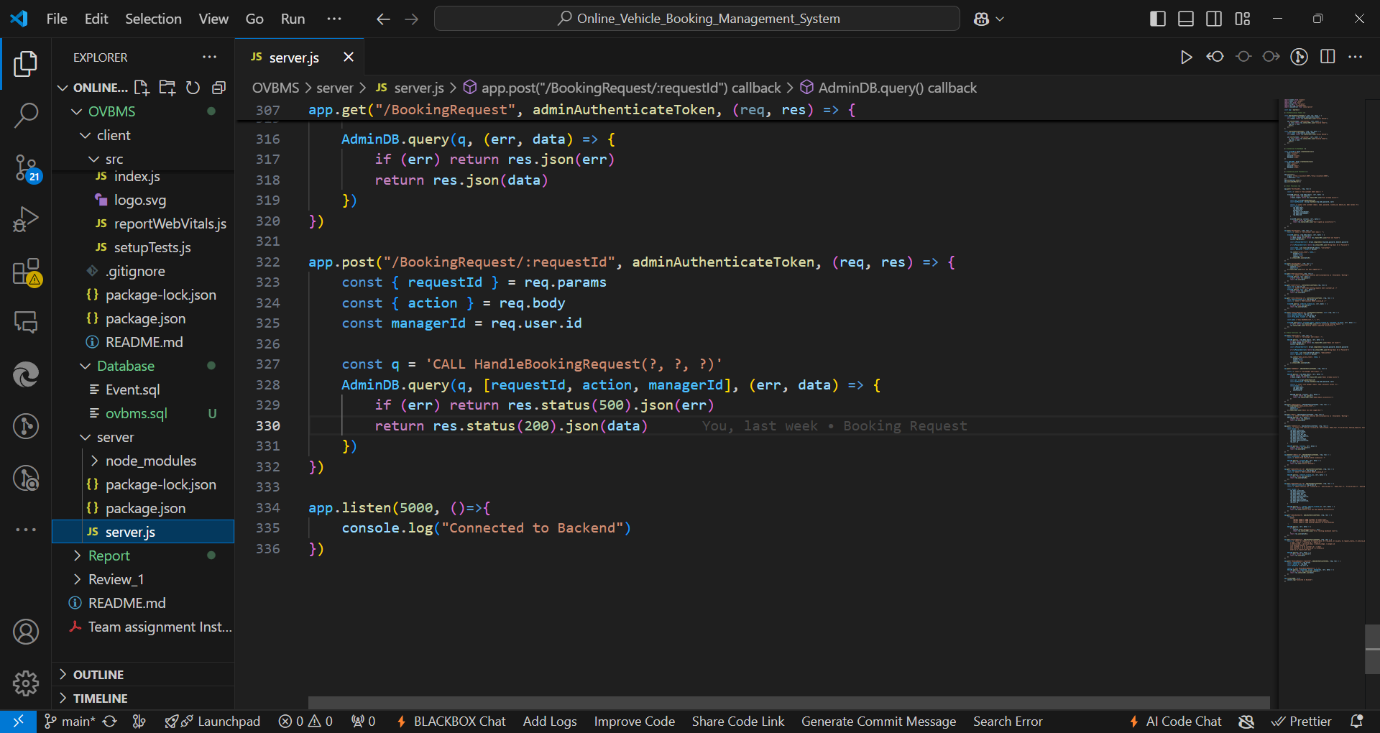


# 9. Code snippets for invoking the Procedures/Functions/ Trigger

## 9.1. Invoking “BookVehicle” Procedure:

## 9.2. Invoking “HandleBookingRequest” Procedure:

#### Note: Trigger is also invoked at this part only to handle the data in other tables



# 10. SQL queries (Create, Insert, Triggers, Procedures/ Functions, Nested query, Join, Aggregate queries) used in the project in the form of .sql file

CREATE DATABASE  IF NOT EXISTS `ovbms` /\*!40100 DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci \*/ /\*!80016 DEFAULT ENCRYPTION='N' \*/;

USE `ovbms`;

-- MySQL dump 10.13  Distrib 8.0.36, for Win64 (x86\_64)

--

-- Host: localhost    Database: ovbms

-- ------------------------------------------------------

-- Server version 8.0.37

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

/\*!50503 SET NAMES utf8 \*/;

/\*!40103 SET @OLD\_TIME\_ZONE=@@TIME\_ZONE \*/;

/\*!40103 SET TIME\_ZONE='+00:00' \*/;

/\*!40014 SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0 \*/;

/\*!40014 SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0 \*/;

/\*!40101 SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='NO\_AUTO\_VALUE\_ON\_ZERO' \*/;

/\*!40111 SET @OLD\_SQL\_NOTES=@@SQL\_NOTES, SQL\_NOTES=0 \*/;

--

-- Table structure for table `booking\_requests`

--

DROP TABLE IF EXISTS `booking\_requests`;

/\*!40101 SET @saved\_cs\_client     = @@character\_set\_client \*/;

/\*!50503 SET character\_set\_client = utf8mb4 \*/;

CREATE TABLE `booking\_requests` (

  `request\_id` varchar(255) NOT NULL,

  `request\_date` datetime NOT NULL,

  `from\_date` date NOT NULL,

  `to\_date` date NOT NULL,

  `request\_status` varchar(15) NOT NULL DEFAULT 'Pending',

  `customer\_id` varchar(45) NOT NULL,

  `vehicle\_id` varchar(10) NOT NULL,

  `manager\_id` varchar(45) DEFAULT NULL,

  `action\_type` varchar(45) DEFAULT NULL,

  `action\_date` datetime DEFAULT NULL,

  PRIMARY KEY (`request\_id`),

  KEY `fk\_manager\_id\_idx` (`manager\_id`),

  KEY `fk\_br\_manager\_id\_idx` (`manager\_id`),

  KEY `fk\_customer\_id\_idx` (`customer\_id`),

  KEY `fk\_vehicle\_id\_idx` (`vehicle\_id`),

  CONSTRAINT `fk\_br\_manager\_id` FOREIGN KEY (`manager\_id`) REFERENCES `manager` (`email`) ON UPDATE CASCADE,

  CONSTRAINT `fk\_customer\_id` FOREIGN KEY (`customer\_id`) REFERENCES `customer` (`email`) ON UPDATE CASCADE,

  CONSTRAINT `fk\_vehicle\_id` FOREIGN KEY (`vehicle\_id`) REFERENCES `vehicles` (`license\_no`) ON UPDATE CASCADE

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `booking\_requests`

--

LOCK TABLES `booking\_requests` WRITE;

/\*!40000 ALTER TABLE `booking\_requests` DISABLE KEYS \*/;

INSERT INTO `booking\_requests` VALUES ('03a7e591-a65b-11ef-a411-b44506948d64','2024-11-19 00:00:00','2024-11-20','2024-11-21','Rejected','user@gmail.com','KA22EF7348','admin@gmail.com','Reject','2024-11-19 00:00:00'),('12013110-acbc-11ef-b89b-b44506948d64','2024-11-27 00:00:00','2024-11-28','2024-11-29','Approved','user3@gmail.com','DL7TC01873','admin@gmail.com','Approve','2024-11-27 00:00:00'),('127b808c-a63c-11ef-a411-b44506948d64','2024-11-19 00:00:00','2024-11-19','2024-11-19','Completed','user@gmail.com','MF412MF412','admin@gmail.com','Approve','2024-11-19 00:00:00'),('27b53886-a63c-11ef-a411-b44506948d64','2024-11-19 00:00:00','2024-11-19','2024-11-19','Rejected','user2@gmail.com','MF412MF412','admin@gmail.com','Reject','2024-11-19 00:00:00'),('377f7cc6-acbc-11ef-b89b-b44506948d64','2024-11-27 00:00:00','2024-11-29','2024-11-30','Rejected','user@gmail.com','DL9TCC1369','admin@gmail.com','Reject','2024-11-27 00:00:00'),('67acae30-a6f4-11ef-a411-b44506948d64','2024-11-20 00:00:00','2024-11-21','2024-11-22','Rejected','user@gmail.com','DL9TCC1396','admin@gmail.com','Reject','2024-11-25 00:00:00'),('6e544a2c-a65e-11ef-a411-b44506948d64','2024-11-19 00:00:00','2024-11-20','2024-11-21','Completed','user@gmail.com','DL9TCC1369','admin@gmail.com','Approve','2024-11-19 00:00:00'),('80ac4895-a63a-11ef-a411-b44506948d64','2024-11-19 00:00:00','2024-11-19','2024-11-19','Rejected','user@gmail.com','MF1223VF34','admin@gmail.com','Reject','2024-11-19 00:00:00'),('85e8f85d-a643-11ef-a411-b44506948d64','2024-11-19 00:00:00','2024-11-19','2024-11-20','Completed','user2@gmail.com','MF1223VF34','admin@gmail.com','Approve','2024-11-19 00:00:00'),('8a5055b5-a63a-11ef-a411-b44506948d64','2024-11-19 00:00:00','2024-11-19','2024-11-19','Rejected','user@gmail.com','MF412MF412','admin@gmail.com','Reject','2024-11-19 00:00:00'),('add803bf-ab30-11ef-a411-b44506948d64','2024-11-25 00:00:00','2024-11-26','2024-11-27','Rejected','user@gmail.com','DL9TCC1369','admin@gmail.com','Reject','2024-11-25 00:00:00'),('ba0871d1-a64b-11ef-a411-b44506948d64','2024-11-19 00:00:00','2024-11-19','2024-11-20','Rejected','user2@gmail.com','DL9TCC1369','admin@gmail.com','Reject','2024-11-19 00:00:00'),('c9da1da0-a534-11ef-a8a4-b44506948d64','2024-11-18 00:00:00','2024-11-18','2024-11-19','Completed','user@gmail.com','MF412MF','admin@gmail.com','Approve','2024-11-18 00:00:00'),('e6568d9d-a309-11ef-a8a4-b44506948d64','2024-11-15 00:00:00','2024-11-17','2024-11-18','Rejected','user@gmail.com','DL9TCC1369','admin@gmail.com','Reject','2024-11-15 00:00:00'),('e728937a-a669-11ef-a411-b44506948d64','2024-11-19 00:00:00','2024-11-21','2024-11-22','Rejected','user@gmail.com','KA22EF7348','admin@gmail.com','Reject','2024-11-19 00:00:00'),('e7f8db69-a534-11ef-a8a4-b44506948d64','2024-11-18 00:00:00','2024-11-19','2024-11-22','Rejected','user@gmail.com','MF1223VF34','admin@gmail.com','Reject','2024-11-18 00:00:00'),('fa54d88c-a363-11ef-a8a4-b44506948d64','2024-11-15 00:00:00','2024-11-18','2024-11-18','Completed','user@gmail.com','DL9TCC1369','admin@gmail.com','Approve','2024-11-15 00:00:00');

/\*!40000 ALTER TABLE `booking\_requests` ENABLE KEYS \*/;

UNLOCK TABLES;

/\*!50003 SET @saved\_cs\_client      = @@character\_set\_client \*/ ;

/\*!50003 SET @saved\_cs\_results     = @@character\_set\_results \*/ ;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;

/\*!50003 SET character\_set\_client  = utf8mb4 \*/ ;

/\*!50003 SET character\_set\_results = utf8mb4 \*/ ;

/\*!50003 SET collation\_connection  = utf8mb4\_0900\_ai\_ci \*/ ;

/\*!50003 SET @saved\_sql\_mode       = @@sql\_mode \*/ ;

/\*!50003 SET sql\_mode              = 'ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION' \*/ ;

DELIMITER ;;

/\*!50003 CREATE\*/ /\*!50017 DEFINER=`root`@`localhost`\*/ /\*!50003 TRIGGER `booking\_requests\_AFTER\_UPDATE` AFTER UPDATE ON `booking\_requests` FOR EACH ROW BEGIN

declare other\_bookings int;

if new.request\_status = 'Rejected' and old.request\_status = 'Pending' then

  update vehicles set availability = 'Available' where license\_no = new.vehicle\_id;

END if;

if new.request\_status = 'Approved' and old.request\_status = 'Pending' then

  update vehicles set availability = 'Unavailable' where license\_no = new.vehicle\_id;

END if;

END \*/;;

DELIMITER ;

/\*!50003 SET sql\_mode              = @saved\_sql\_mode \*/ ;

/\*!50003 SET character\_set\_client  = @saved\_cs\_client \*/ ;

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ ;

/\*!50003 SET collation\_connection  = @saved\_col\_connection \*/ ;

--

-- Table structure for table `customer`

--

DROP TABLE IF EXISTS `customer`;

/\*!40101 SET @saved\_cs\_client     = @@character\_set\_client \*/;

/\*!50503 SET character\_set\_client = utf8mb4 \*/;

CREATE TABLE `customer` (

  `email` varchar(45) NOT NULL,

  `name` varchar(45) NOT NULL,

  `password` varchar(255) NOT NULL,

  `license\_id` varchar(16) NOT NULL,

  `mobile\_no` varchar(10) NOT NULL,

  `dob` date NOT NULL,

  PRIMARY KEY (`email`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `customer`

--

LOCK TABLES `customer` WRITE;

/\*!40000 ALTER TABLE `customer` DISABLE KEYS \*/;

INSERT INTO `customer` VALUES ('user@gmail.com','User','$2a$10$r1RUIFDnQZ4yFKrkPs5glepf/itAsIfCuirP325S.lgIqcA/Z/j0e','DL-1420110012345','9876543210','2004-04-01'),('user2@gmail.com','User2','$2a$10$p3zZnsiKqKIdLQOk2RXrWOgfx18cq9PLIGk9yJlhGtLwl.p1Gm5zW','DL-2420110012345','9276543210','2004-02-15'),('user3@gmail.com','User3','$2a$10$aDhirlcY3P0DCvEJjpph0.gkqX5/V714NuTqRn21StOpcddSgko8e','DL-1420110012346','9276543211','2000-02-17');

/\*!40000 ALTER TABLE `customer` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `manager`

--

DROP TABLE IF EXISTS `manager`;

/\*!40101 SET @saved\_cs\_client     = @@character\_set\_client \*/;

/\*!50503 SET character\_set\_client = utf8mb4 \*/;

CREATE TABLE `manager` (

  `email` varchar(45) NOT NULL,

  `name` varchar(45) NOT NULL,

  `password` varchar(255) NOT NULL,

  PRIMARY KEY (`email`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `manager`

--

LOCK TABLES `manager` WRITE;

/\*!40000 ALTER TABLE `manager` DISABLE KEYS \*/;

INSERT INTO `manager` VALUES ('admin@gmail.com','Admin','$2a$10$JZHSUXa2KJ52oNB.8QaeXucRZ0n5OpGCLFXH0Ort8NkAh5a2BDTxi'),('admin2@gmail.com','Admin 2','$2a$10$8uDDK7B87GcQDlTm1c7U8enTAZf1KlMu7UHOCn8yLU8LLVAbxwZFe'),('manager@gmail.com','Manager','$2a$10$AhkVm6h3LCwiHr936ONwy.HOsV./g.f/3cdrCjfhAWW9Am4nM.bJy');

/\*!40000 ALTER TABLE `manager` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `vehicles`

--

DROP TABLE IF EXISTS `vehicles`;

/\*!40101 SET @saved\_cs\_client     = @@character\_set\_client \*/;

/\*!50503 SET character\_set\_client = utf8mb4 \*/;

CREATE TABLE `vehicles` (

  `license\_no` varchar(10) NOT NULL,

  `vehicle\_name` varchar(45) NOT NULL,

  `model\_year` year NOT NULL,

  `price\_per\_day` decimal(10,0) NOT NULL,

  `seating\_capacity` int NOT NULL,

  `fuel\_type` varchar(10) NOT NULL,

  `vehicle\_image` longtext,

  `vehicle\_overview` varchar(225) DEFAULT NULL,

  `manager\_id` varchar(45) NOT NULL,

  `availability` varchar(45) NOT NULL DEFAULT 'Available',

  PRIMARY KEY (`license\_no`),

  KEY `fk\_manager\_id\_idx` (`manager\_id`),

  CONSTRAINT `fk\_manager\_id` FOREIGN KEY (`manager\_id`) REFERENCES `manager` (`email`) ON UPDATE CASCADE

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `vehicles`

--

LOCK TABLES `vehicles` WRITE;

/\*!40000 ALTER TABLE `vehicles` DISABLE KEYS \*/;

INSERT INTO `vehicles` VALUES ('DL7TC01873','Hyundai Nexon',2020,6000,7,'Petrol','https://www.hyundai.com/content/dam/hyundai/in/en/data/find-a-car/Verna/Highlights/vernapoolside1.jpg','Good','admin@gmail.com','Unavailable'),('DL9TCC1369','Tata Nano',2020,1500,7,'Diesel','https://upload.wikimedia.org/wikipedia/commons/thumb/3/3c/Tata\_-\_Nano\_-\_Kolkata\_2011-09-15\_5184.JPG/420px-Tata\_-\_Nano\_-\_Kolkata\_2011-09-15\_5184.JPG','Very Good Car','admin@gmail.com','Available'),('DL9TCC1396','Hyundai EXTER Knight',2020,8000,8,'Petrol','https://www.hyundai.com/content/dam/hyundai/in/en/data/find-a-car/Exter/Highlights/smallimageexter2\_5.jpg','Good Car','admin@gmail.com','Available'),('KA22EF7348','Hyundai Alcazar',2020,5500,7,'Diesel','https://www.hyundai.com/content/dam/hyundai/in/en/data/find-a-car/Alcazar/Gallery/alcazargallerybig1.jpg','Good Car','manager@gmail.com','Available'),('MF1223VF34','Hyundai IONIQ 5',2024,5000,5,'Electric','https://ioniq5.hyundai.co.in/uploads/attachments/cluqpl0yg4jiktasahp8ats4c-front-image.png','Good Car','manager@gmail.com','Available'),('MF412MF','Hyundai IONIQ 5',2020,4000,5,'Electric','https://ioniq5.hyundai.co.in/uploads/attachments/cluqpl0yg4jiktasahp8ats4c-front-image.png','Good Car','manager@gmail.com','Available'),('MF412MF412','Hyundai IONIQ 5',2022,4500,5,'Electric','https://ioniq5.hyundai.co.in/uploads/attachments/cluqpl0yg4jiktasahp8ats4c-front-image.png','Good Car','admin@gmail.com','Available');

/\*!40000 ALTER TABLE `vehicles` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Dumping events for database 'ovbms'

--

/\*!50106 SET @save\_time\_zone= @@TIME\_ZONE \*/ ;

/\*!50106 DROP EVENT IF EXISTS `CheckCompletedBookings` \*/;

DELIMITER ;;

/\*!50003 SET @saved\_cs\_client      = @@character\_set\_client \*/ ;;

/\*!50003 SET @saved\_cs\_results     = @@character\_set\_results \*/ ;;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;;

/\*!50003 SET character\_set\_client  = utf8mb4 \*/ ;;

/\*!50003 SET character\_set\_results = utf8mb4 \*/ ;;

/\*!50003 SET collation\_connection  = utf8mb4\_0900\_ai\_ci \*/ ;;

/\*!50003 SET @saved\_sql\_mode       = @@sql\_mode \*/ ;;

/\*!50003 SET sql\_mode              = 'ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION' \*/ ;;

/\*!50003 SET @saved\_time\_zone      = @@time\_zone \*/ ;;

/\*!50003 SET time\_zone             = 'SYSTEM' \*/ ;;

/\*!50106 CREATE\*/ /\*!50117 DEFINER=`root`@`localhost`\*/ /\*!50106 EVENT `CheckCompletedBookings` ON SCHEDULE EVERY 1 DAY STARTS '2024-11-19 11:42:39' ON COMPLETION NOT PRESERVE ENABLE DO BEGIN

UPDATE vehicles

SET availability = 'Available'

WHERE license\_no IN (

  SELECT vehicle\_id

  FROM booking\_requests

  WHERE request\_status = 'Approved'

  AND to\_date < CURDATE()

    );

UPDATE booking\_requests

SET request\_status = 'Completed'

WHERE request\_status = 'Approved'

AND to\_date < CURDATE();

END \*/ ;;

/\*!50003 SET time\_zone             = @saved\_time\_zone \*/ ;;

/\*!50003 SET sql\_mode              = @saved\_sql\_mode \*/ ;;

/\*!50003 SET character\_set\_client  = @saved\_cs\_client \*/ ;;

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ ;;

/\*!50003 SET collation\_connection  = @saved\_col\_connection \*/ ;;

DELIMITER ;

/\*!50106 SET TIME\_ZONE= @save\_time\_zone \*/ ;

--

-- Dumping routines for database 'ovbms'

--

/\*!50003 DROP PROCEDURE IF EXISTS `BookVehicle` \*/;

/\*!50003 SET @saved\_cs\_client      = @@character\_set\_client \*/ ;

/\*!50003 SET @saved\_cs\_results     = @@character\_set\_results \*/ ;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;

/\*!50003 SET character\_set\_client  = utf8mb4 \*/ ;

/\*!50003 SET character\_set\_results = utf8mb4 \*/ ;

/\*!50003 SET collation\_connection  = utf8mb4\_0900\_ai\_ci \*/ ;

/\*!50003 SET @saved\_sql\_mode       = @@sql\_mode \*/ ;

/\*!50003 SET sql\_mode              = 'ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION' \*/ ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` PROCEDURE `BookVehicle`(in customer\_email varchar(45), in vehicle\_license\_no varchar(10), in from\_date date, in to\_date date)

BEGIN

insert into booking\_requests(request\_id, request\_date, from\_date, to\_date, customer\_id, vehicle\_id) values(UUID(), CURDATE(), from\_date, to\_date, customer\_email, vehicle\_license\_no);

update vehicles set availability = 'Waiting' where license\_no = vehicle\_license\_no;

END ;;

DELIMITER ;

/\*!50003 SET sql\_mode              = @saved\_sql\_mode \*/ ;

/\*!50003 SET character\_set\_client  = @saved\_cs\_client \*/ ;

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ ;

/\*!50003 SET collation\_connection  = @saved\_col\_connection \*/ ;

/\*!50003 DROP PROCEDURE IF EXISTS `HandleBookingRequest` \*/;

/\*!50003 SET @saved\_cs\_client      = @@character\_set\_client \*/ ;

/\*!50003 SET @saved\_cs\_results     = @@character\_set\_results \*/ ;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;

/\*!50003 SET character\_set\_client  = utf8mb4 \*/ ;

/\*!50003 SET character\_set\_results = utf8mb4 \*/ ;

/\*!50003 SET collation\_connection  = utf8mb4\_0900\_ai\_ci \*/ ;

/\*!50003 SET @saved\_sql\_mode       = @@sql\_mode \*/ ;

/\*!50003 SET sql\_mode              = 'ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION' \*/ ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` PROCEDURE `HandleBookingRequest`( in requestId varchar(255), in action varchar(15), in managerId varchar(45))

BEGIN

if action = 'Approve' then

update booking\_requests set request\_status = 'Approved', action\_type = 'Approve', action\_date = curdate(), manager\_id = managerId  where request\_id = requestId;

elseif action = 'Reject' then

update booking\_requests set request\_status = 'Rejected', action\_type = 'Reject', action\_date = curdate(), manager\_id = managerId where request\_id = requestId;

END IF;

END ;;

DELIMITER ;

/\*!50003 SET sql\_mode              = @saved\_sql\_mode \*/ ;

/\*!50003 SET character\_set\_client  = @saved\_cs\_client \*/ ;

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ ;

/\*!50003 SET collation\_connection  = @saved\_col\_connection \*/ ;

/\*!40103 SET TIME\_ZONE=@OLD\_TIME\_ZONE \*/;

/\*!40101 SET SQL\_MODE=@OLD\_SQL\_MODE \*/;

/\*!40014 SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS \*/;

/\*!40014 SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS \*/;

/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;

/\*!40111 SET SQL\_NOTES=@OLD\_SQL\_NOTES \*/;

-- Dump completed on 2024-11-28 15:16:12

# 11. GitHub Repo Link

GitHub [*link*](https://github.com/SanketMuttur/Online_Vehicle_Booking_Management_System) for [*Online Vehicle Booking Management System*](https://github.com/SanketMuttur/Online_Vehicle_Booking_Management_System)