Car Rental Management System

Project Guide

Table of Contents

1. INTRODUCTION

The MoveZ car rental website is a web-based application developed to simplify the process of renting vehicles online using ASP.NET Core MVC with Microsoft SQL Server, the system integrates multiple user roles (Admin, Staff, Driver, Customer) with distinct dashboards and access rights. The application combines ease of use for customers with powerful management tools for administrators. The system allows browsing available cars, viewing details, making booking, and managing reservation. it is designed with a user –friendly interface, ensuring both customers and administrators can efficiently interact with the system.

Unlike a simple car rental demo, this extended version of CRMS includes advanced features such as Google login, OTP-based payments, notifications, seasonal offers, car comparison, automated booking validations, and FAQs chatbot. The aim is to not only meet academic requirements but to deliver a system that closely mirrors real-world applications in terms of usability, security, and scalability.

2. OBJECTIVES

The objectives of the Car Rental Management System are as follows:

1. Provide a role-based secure authentication system.
2. Enable guests to browse, filter, and compare cars.
3. Allow customers to register, manage profiles, book cars, and make payments.
4. Provide multiple payment options including OTP-secured methods.
5. Give administrators full control over cars, bookings, payments, drivers, and staff.
6. Ensure staff and drivers can perform restricted tasks according to their roles.
7. Introduce customer support features like FAQs and notifications.
8. Apply professional UI/UX design principles using Bootstrap.
9. Ensure the system is robust, scalable, and adaptable for future enhancements.

3. SYSTEM DESIGN & ARCHITECTURE

3.1 MVC + Service + Repository Architecture

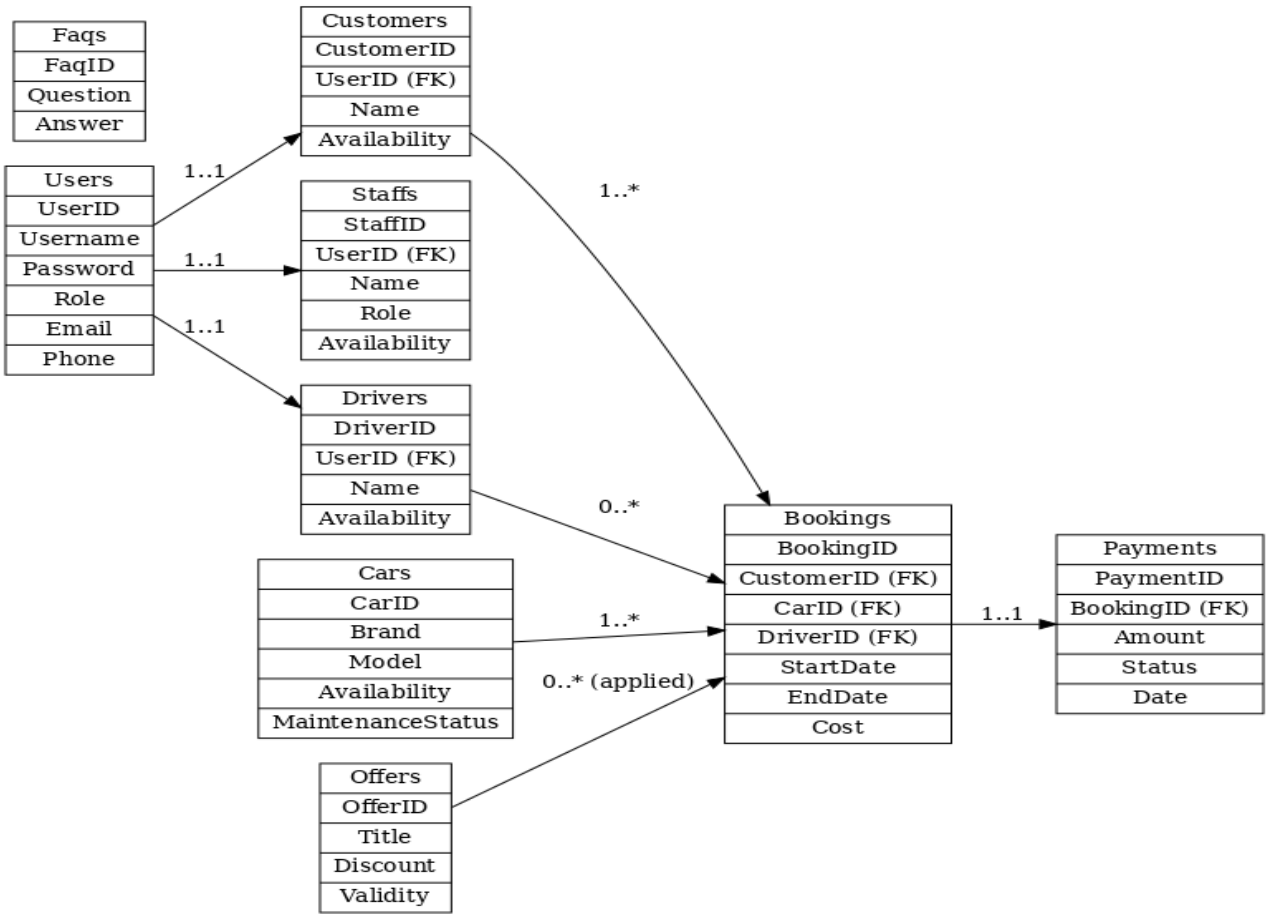
The car rental website is implemented using ASP.NET core MVC combined with service and repository patterns to ensure scalability, and clean separation of concern.

- **Model**
  Represents entities such as Users, Cars, Bookings, Payments, and Drivers.
- **View**
  User interface built with Razor Pages, Bootstrap, and JavaScript.
- **Controller**
  Handles HTTP requests, communicates with service layer, and manage responses to the user interface, controllers never directly access the database.
- **Service Layer**
  Encapsulates the business logic of the appilication.it acts as a bridge between controllers and the repository layer.
- **Repository Layer**
  Responsible for interacting with the database. CRUD operations(Create,Read,Update,Delete) are implemented here using Entity Framework Core.
- **DTOs(Data Transfer Objects)**
  Used to transfer data between layers securely , preventing over-posting or exposing database entities directly
- **View Models**
  Provide data shaping for razor views, ensuring only necessary fields are passed to the UI
- **Mappings**
  The system uses mapping between ViewModel – model-DTO ensure data integrity and maintain a clear boundary layers. Mapping Function is used to handle this transformation.

3.2 DATABASE DESIGN

The system uses Microsoft SQL Server as the database backend. Tables include:

- **Users**: Stores user credentials, roles, and profile details.
- **Cars**: Stores car details such as brand, model, availability, and maintenance status.
- **Bookings**: Records reservations with dates, costs, and car assignments.
- **Payments**: Tracks transactions and their status.
- **Customers**: Stores Customer profiles and availability
- **Staffs**: Stores Admin and Staff profiles and availability
- **Drivers**: Stores driver profiles and availability.
- **Offers**: Stores Seasonal Offer details and validity
- **Faqs**: Stores Frequent question and answers.



3.3 User Role Diagram

The system supports multiple roles with distinct privileges:

- Admin: Full access to all modules.
- Staff: Access to Manage bookings and payments. Do not have access to staff user creation and offer management.
- Driver: Restricted to viewing assigned bookings only.
- Customer: Can view/edit profile, book cars, and manage payments.

4. FEATURES & MODULES

4.1 General Features

1. Separate modules for Guest Section, Customer Dashboard, and Admin Dashboard.
2. Role-based authentication with support for standard login and Google login.
3. Customers access their dedicated dashboard, while Admins access the Admin Dashboard.
    - The first Admin logs in using a hardcoded username and password, then can create Staff and additional Admin accounts.
4. Password management, including secure password change and prevention of duplicate email registrations.
5. Role-specific access control:
    - Customer: Manage own bookings, payments, and profile.
    - Admin: Full access to all modules.
    - Staff: Access Admin Dashboard except the Staff Management module.
    - Driver: Access assigned booking module only.

4.2 Guest Users Features

Guests are unregistered users who can access general pages like Home, About Us, Services, and Contact Us. They can browse cars with filters, view details, and compare cars. Attempting to book redirects them to the login/register page. A FAQs chatbot is also available.

1. Access to **Home**, **About Us**, **Services**, and **Contact Us** pages.
2. Browse cars with **filters** (name, brand, model, price).
3. **Car comparison** feature: compare two cars in a single view with a summarized comparison.
4. View detailed information of a car, including per-day rental cost.
5. When attempting to book, guests are redirected to **login/register** or can use **Google login**.
6. Integrated **FAQs chatbot** on the homepage for instant support.
7. Contact form available for guest inquiries.

4.2 Customer Features

Customers can register/login using email or Google authentication. Their dashboard allows profile editing, viewing bookings, making new bookings, processing payments, and applying promo codes. OTP verification is required for online payments. They can cancel bookings within 24 hours of the start date.

1. Secure login with registered credentials.
2. Customer Dashboard includes:
    - Profile management (view & edit).
    - View personal bookings and payments.
    - Editable account details.
3. Booking system pre-fills details from the customer's profile, requiring minimal input.
4. **Date validation** during booking:
    - Past dates not allowed.
    - End date must be after start date.
    - Already-booked dates hidden from availability.
    - Cancelled dates reappear in availability.
5. Option to book cars **with or without a driver**.
6. **Automatic cost calculation** based on duration, per-day car cost, and optional driver fee.
7. **Payment methods**: Card, Wallet, or Cash.
    - Card/Wallet payments secured with **OTP via email** → status set to *Paid*.
    - Cash payments initially *Pending* until confirmed at the counter → status updated to *Paid*.
8. **Promo codes** can be applied for discounts.

9. Customers can monitor booking and payment history, including **Paid, Pending, and Refunded** transactions (dynamically updated).
10. Bookings can be cancelled up to **24 hours before the start date**; after that, the cancel option disappears.
11. **Real-time notifications** for all booking and payment actions.
12. **Search and filter options** for bookings and payments by status or keyword.

4.3 Admin Features

Admins manage all aspects of the system. Their dashboard provides an overview of bookings, payments, and available cars. They can add/edit/remove cars, manage bookings, assign drivers, verify payments, process refunds, manage staff, and create seasonal offers. Admins can also manage FAQs to support customers.

1. **Admins and Staffs log in:** via role-based authentication; Drivers only access assigned bookings.
2. **Notifications** for all customer actions.
3. **Dashboard Overview**:
    - Total payments received.
    - Total available and maintenance cars.
    - Hourly bookings summary.
    - Booking status summary.
    - Quick-access links.
4. **Car Management**: Add, view, edit, or remove cars. Cars under maintenance are automatically hidden from the website and reappear when marked available again.
5. **Booking Management**: View all bookings; confirmed only after payment. Admin can assign drivers once payment is verified.
6. **Payment Management**:
    - View all transactions.
    - Update *Pending* cash payments to *Paid* upon confirmation.
    - Refunds processed with OTP verification for added security.
7. **Staff & Admin Management**: Create additional staff/admin accounts, with list and role management.
8. **Driver Management**: Add, view, edit, or remove drivers.
    - Drivers appear in booking assignment lists.
    - Assigned drivers remain unavailable until their current duty is completed or a booking is cancelled.
9. **Offers Management**: Create and manage seasonal offers displayed on the website.
10. **FAQs Management**: Add and update frequently asked questions for customer convenience.
11. **Notifications** for every key admin action.
12. **Search and filter functionality** in all admin modules by status and keyword.

4.4 Staff Features

Staff members have restricted access to the admin dashboard. They can manage bookings and payments but cannot create or manage other staff and offers.

4.5 Driver Features

Drivers can only access assigned bookings. They remain unavailable while assigned to a booking and become available once duties are completed.

5. TECHNOLOGY STACK

- **Frontend**: HTML5, CSS3, Bootstrap, Razor Pages, JavaScript.
- **Backend**: ASP.NET Core MVC, C#.
- **Database**: Microsoft SQL Server.
- **ORM**: Entity Framework Core.
- **Authentication**: Identity Framework with Google OAuth integration.
- **Tools**: Visual Studio, GitHub, SMTP for OTP emails.

6. SECURITY & AUTHENTICATION

- The system ensures security through:
- Role-based access restrictions.
- Google OAuth login integration.
- OTP verification for sensitive actions like payments.
- Prevention of duplicate email registration.
- Password change facilities.

7. BOOKING & PAYMENT WORKFLOW

1. Workflow:
2. Customer selects a car using filtration and cooperation.
3. Provide essential details Location to pick up , driver expectation and dates profile details come automatically.
4. System validates dates against availability.
5. Cost is automatically calculated.
6. Customer selects payment method:
   - Card/Wallet with OTP.
   - Cash with pending status until verified.
7. Confirmation and notifications are sent.
8. Customer can cancel booking before a day with refund policy

7. MANAGE BOOKING & PAYMENT WORKFLOW

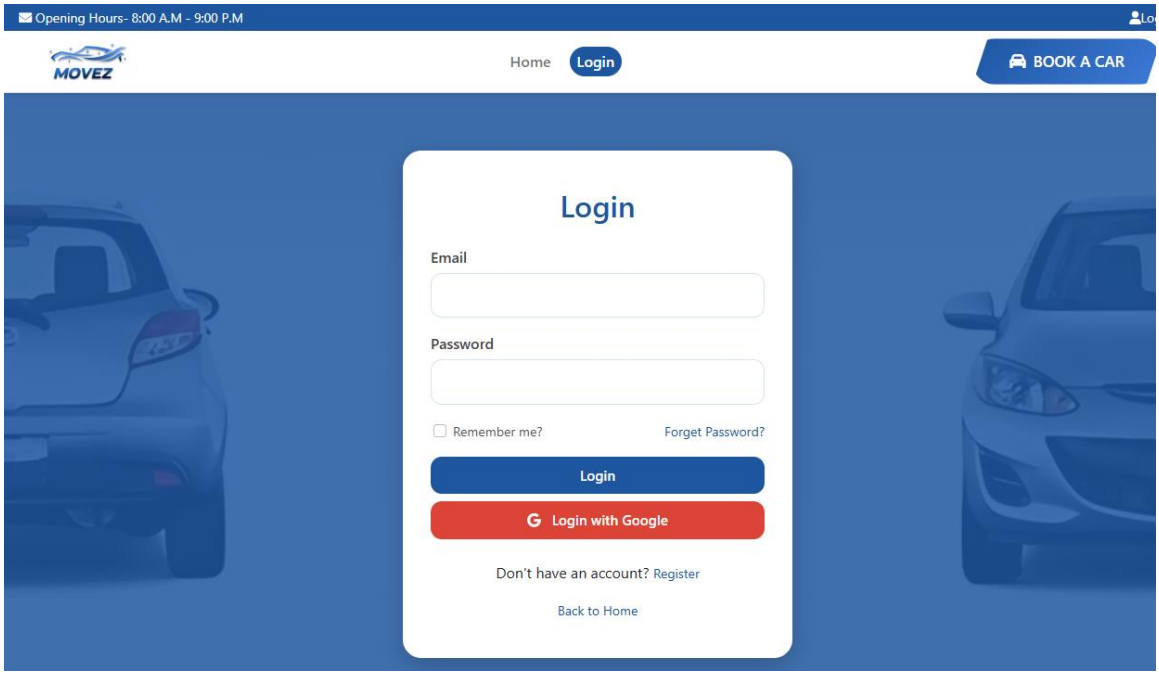
1. **Booking and Payment Notifications**
   - The Admin Dashboard receives real-time notifications for all bookings and payment activities.
2. **Booking Assignment Rules**
   - Bookings cannot be assigned until payment is received.
   - For **online payments (Card/Wallet)**, the system automatically marks the status as *Paid*, allowing direct booking assignment without verification.
   - For **cash payments**, the booking remains *Pending* until confirmed at the counter.
3. **Cash Payment Verification**
   - The cashier receives physical payments at the counter.
   - An **OTP is sent to the customer's registered email** for verification.
   - Once the customer enters the OTP, the system updates the status to *Paid*.
4. **Driver Assignment**
   - After payment confirmation, bookings can be assigned to drivers (if the customer requested the "With Driver" option).
   - For driver-less bookings, assignment is completed directly without driver allocation.
5. **Booking Cancellations and Refunds**
   - If a customer cancels a booking **after payment**, the system automatically updates the payment status to *Pending Refund*.
   - The cashier initiates the refund process, generating an **OTP sent to the company's official email**.
   - Upon OTP verification, the refund is processed and credited to the customer's account.

8. NOTIFICATIONS & COMMUNICATION SYSTEM

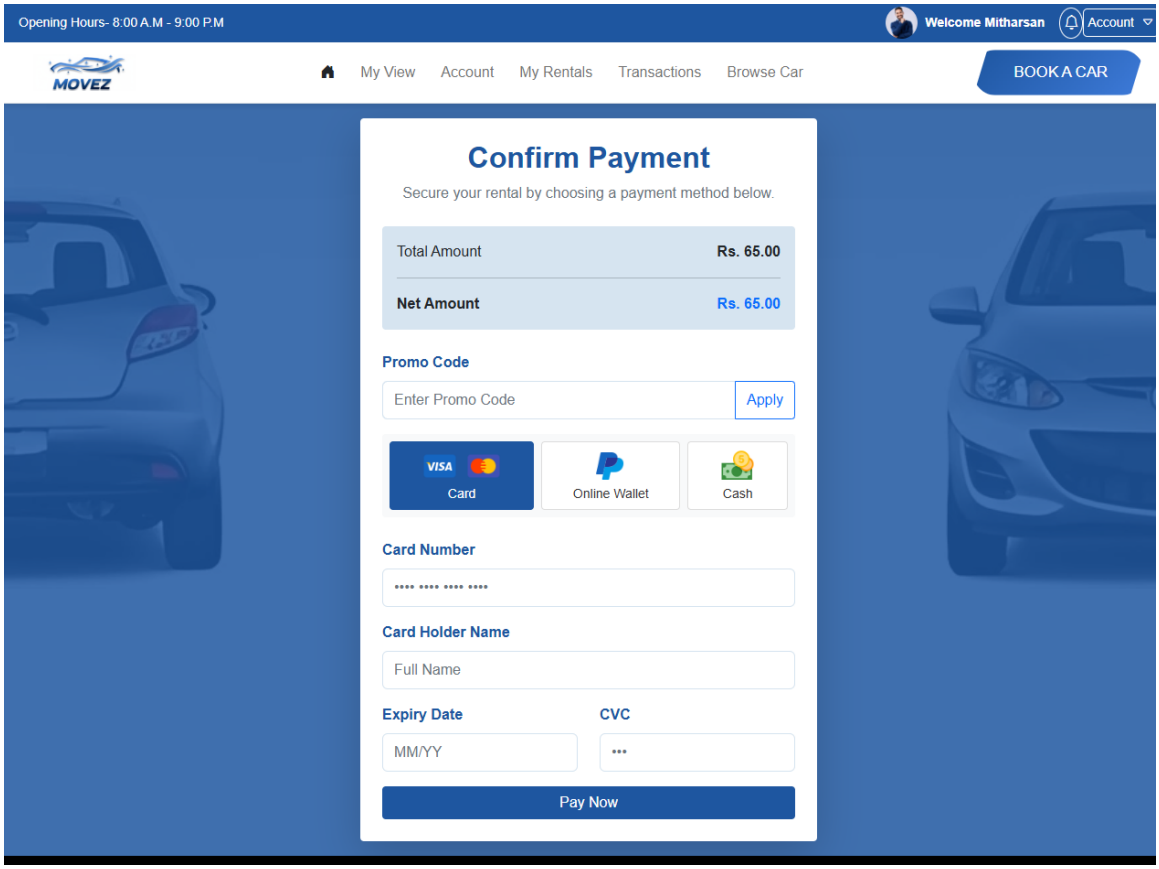
Notifications are triggered for all major events like bookings, cancellations, payments, and driver assignments. Customers receive notifications via dashboard and email, while admins get alerts about customer actions.

9. ADVANCE SYSTEM FEATURES

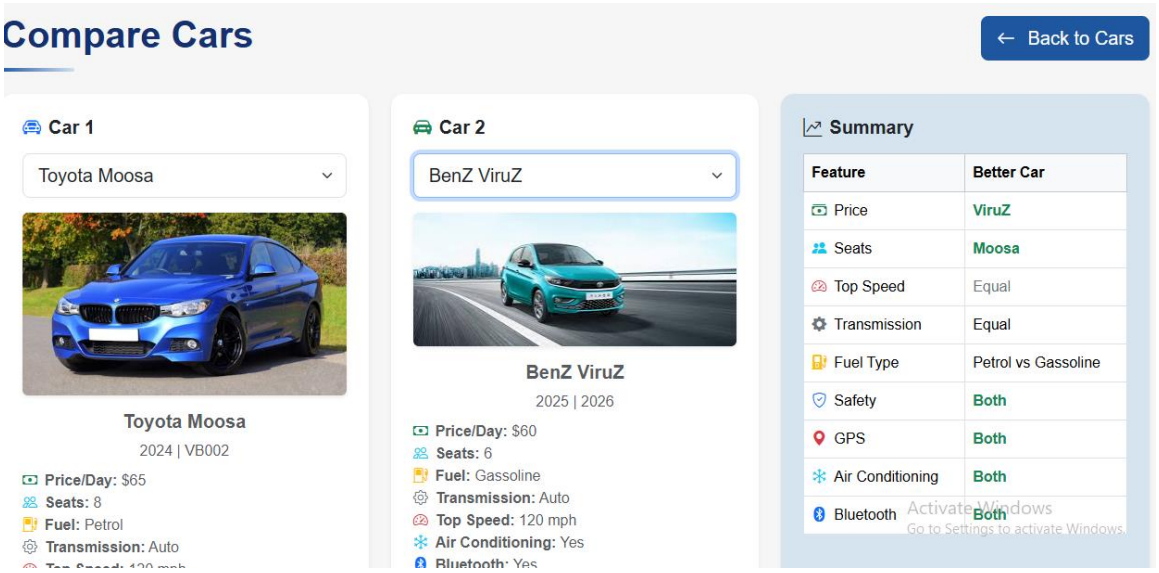
Google Login facility:



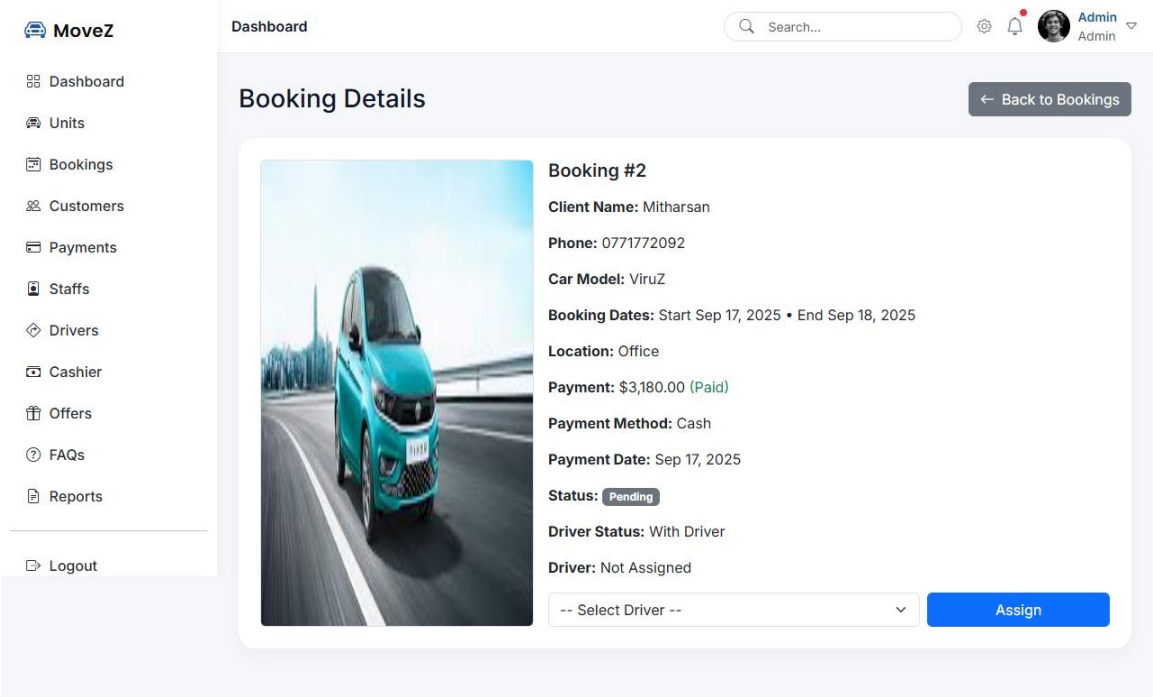
Different Payment Methods:



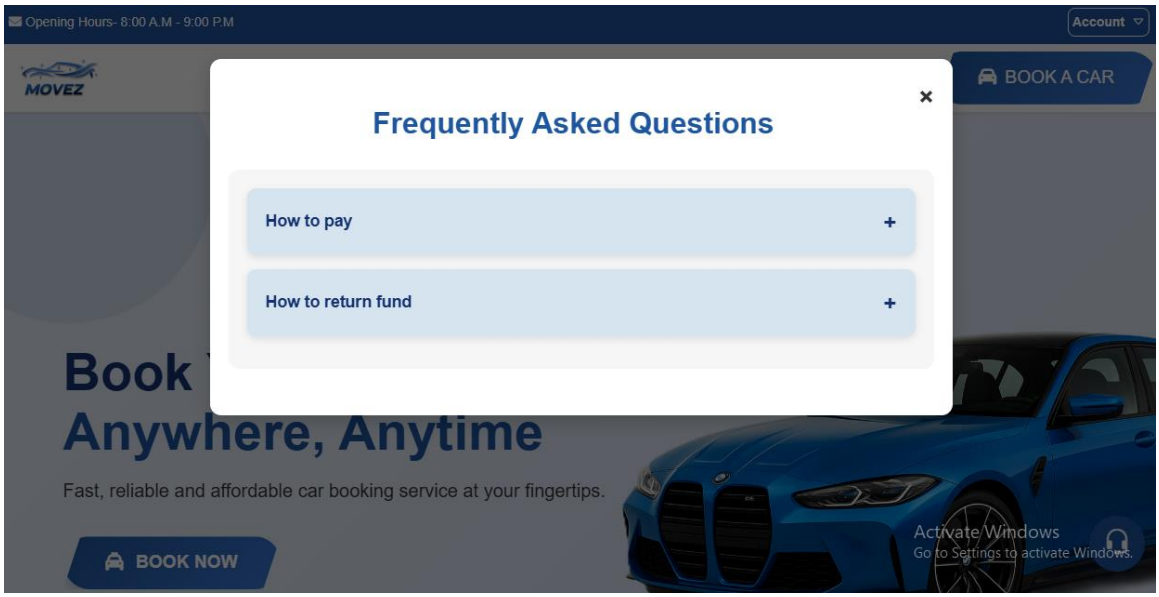
Compare two cars facilities



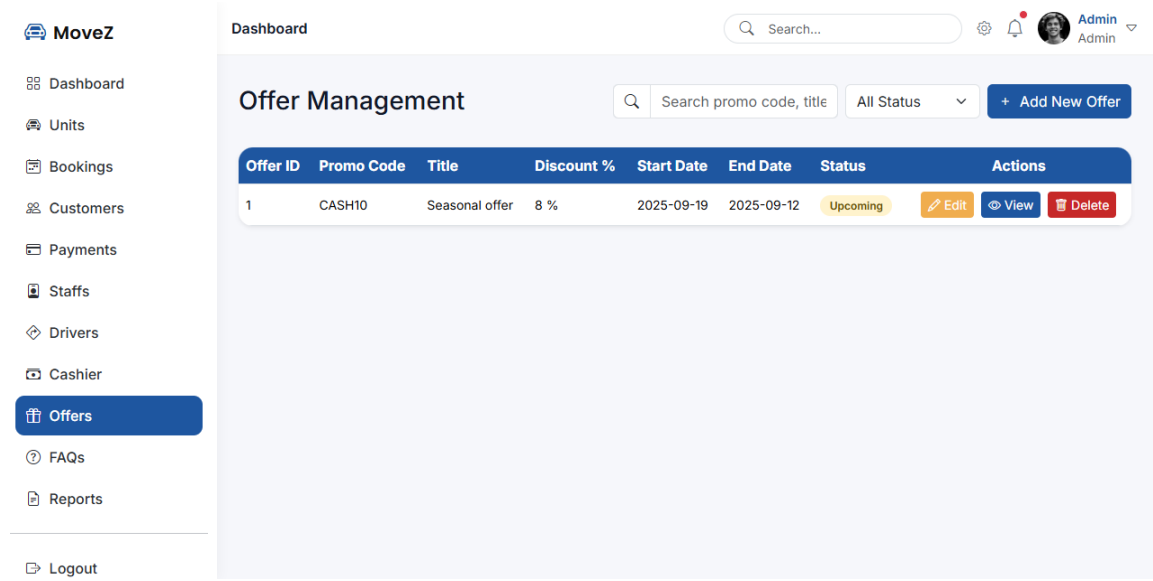
Assign Booking for drivers:



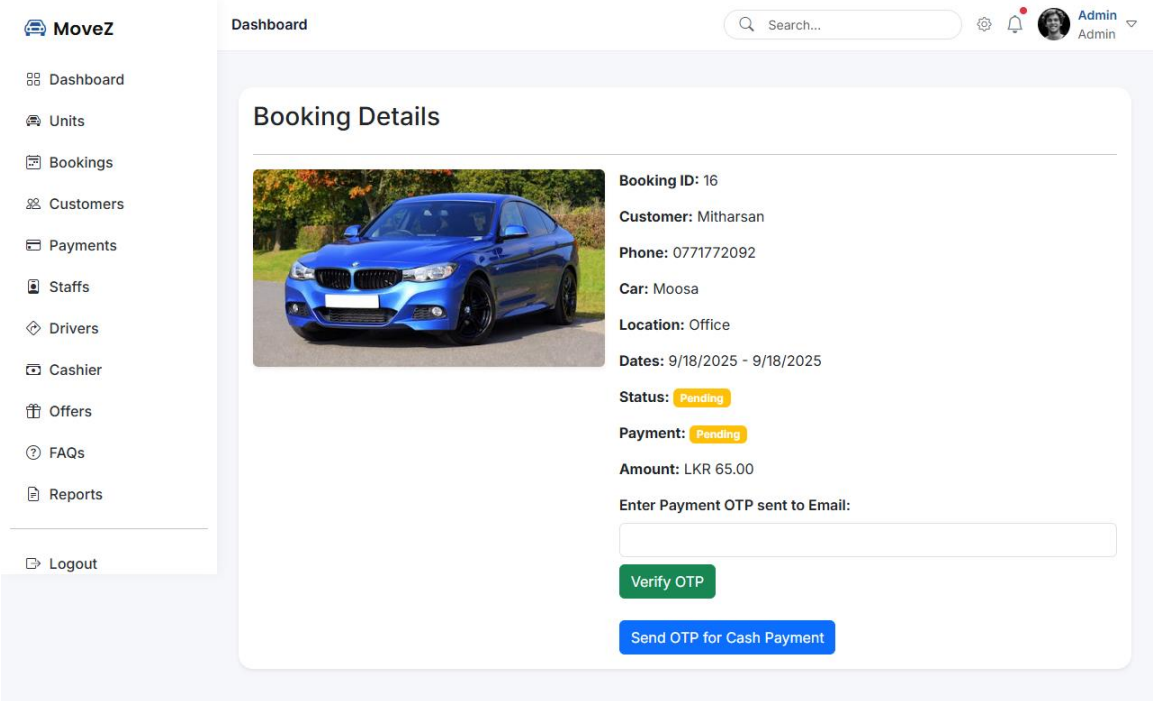
Chat Box for FAQs



Seasonal Offer Creation



OTP Verification:



10. CHALLENGES & SOLUTIONS

- Role-based routing required middleware customization.
- Booking conflicts solved by validating overlapping dates.
- OTP configuration required SMTP integration.
- Ensuring UI responsiveness required Bootstrap grid layouts.

11. TESTING & VALIDATION

Testing involved both manual and automated approaches:

- Unit testing for booking validations and payment processing.
- Integration testing for authentication and role-based access.
- Manual testing of UI responsiveness and cross-browser compatibility.

12. CONCLUSION & FUTURE ENHANCEMENTS

The Car Rental Management System successfully integrates multiple modules, user roles, and advanced features. It demonstrates practical application of ASP.NET Core MVC and SQL Server in building secure and scalable web applications.

Future Enhancements:

- Mobile application for Android/iOS.
- AI-powered chatbot for support.
- Integration with Stripe/PayPal.
- Advanced data analytics and reporting.

13. DEFAULT CREDENTIALS

- **Initial Admin Login**
  UserName-admin@movez.com
  PassWord-Admin@123

- **Company Email for Cashier refund**
  UserName-carrentalmovez@gmail.com
  PassWord-Admin@123

14. REFERENCES

- ASP.NET Core MVC Documentation: https://learn.microsoft.com/aspnet/core
- Microsoft SQL Server Documentation: https://learn.microsoft.com/sql
- Bootstrap: https://getbootstrap.com
- Entity Framework Core: https://learn.microsoft.com/ef/core
- Google OAuth Documentation: https://developers.google.com/identity