RENTER

REPORT TO OUR RENTER SYSTEM

Table of Contents

[Introduction: 3](#_Toc156689389)

[About Renter: 3](#_Toc156689390)

[Chapter 1: Application Architecture 3](#_Toc156689391)

[Database: 3](#_Toc156689392)

[App Service: 4](#_Toc156689393)

[User Interface: 4](#_Toc156689394)

[Chapter 2: Database Model 4](#_Toc156689395)

[Vehicle table: 4](#_Toc156689396)

[Car table: 4](#_Toc156689397)

[Bike table: 4](#_Toc156689398)

[User table: 4](#_Toc156689399)

[Permission table: 4](#_Toc156689400)

[User Permission: 5](#_Toc156689401)

[Vehicle Rental: 5](#_Toc156689402)

[Reports table: 5](#_Toc156689403)

[ER-Diagram: 6](#_Toc156689404)

[Physical Relational Model: 7](#_Toc156689405)

[Database Functions, Procedures, and Triggers 7](#_Toc156689406)

[Functions: 7](#_Toc156689407)

[These are all per day but if there is additional days like for instance a delay for returning the vehicle for 3 days and he initially rented it for 5 days what would happen is 8](#_Toc156689408)

[The user pays the bill for the 5 days: costPerday \*5 8](#_Toc156689409)

[The user pays the penalty fee bill for the 3 days: costPerDay \* 2 \*3 8](#_Toc156689410)

[Procedures: 8](#_Toc156689411)

[Triggers: 8](#_Toc156689412)

[Chapter 3: User Interface 9](#_Toc156689413)

[Dashboard (Home Page) 9](#_Toc156689414)

[Rented Page 11](#_Toc156689415)

[Add Vehicles 12](#_Toc156689416)

[User Stats 13](#_Toc156689417)

[Reports 13](#_Toc156689418)

[Update Vehicle Age 14](#_Toc156689419)

[Chapter 4: Implementation and Testing 14](#_Toc156689420)

[Conclusion 15](#_Toc156689421)

# Introduction:

Welcome to RENTER a comprehensive system designed to streamline the management of car and motorbike rentals. This user guide intends to provide you with a detailed overview of the features and functionalities offered by our system, helping you navigate through the various aspects of renting and managing vehicles effortlessly.

## About Renter:

RENTER is user-friendly and efficient for businesses operating in the rental industry. This system is designed to simplify the rental process, enhance customer experience, and optimize overall management, whether you operate a rental agency or manage a fleet of vehicles.

# Chapter 1: Application Architecture

In this chapter we will talk about the application architecture that consist of 3 main components

* Database
* App Server
* User Interface

## Database:

The database is structured with tables for vehicle information, car details, bike details, user info, permission table, user permission (specify the users that gain special permissions), and vehicle rental table that contain all the booking details . This relational database ensures efficient data retrieval and integrity through primary and foreign key constraints.

## App Service:

This component acts as an intermediary between the database and the client interface, facilitating seamless communication and data processing. It hosts business logic and handles user requests.

## User Interface:

The client interface provides a user-friendly interaction point between users and the institution. It facilitates various operations such as searching for wanted vehicle , adding and updating vehicles, viewing available and unavailable vehicles ,and view the history of old rentals.

# Chapter 2: Database Model

## Vehicle table:

This table carry the details about the vehicle such as vehicle id, registration number, brand, model, age availability, and an image for the vehicle. The id is unique and acts as a primary key.

## Car table:

This table is extended from the vehicle table and provide more details about vehicles of type ‘Car’ such as number of seats and number of doors, in addition to the car id that acts as a primary key and a foreign key that reference the vehicle table

## Bike table:

This table is also extended from the vehicle table that covers the `Bike ` type of vehicles in that table. It provides more details about the bike such as engine size and number of wheels, and it contains an id that act as a primary key and a foreign key that is referenced from vehicle.

## User table:

Here the table is focused on the dealers in our system (the employees). This table holds user id that acts as a primary key, full name, email, and password.

## Permission table:

This table includes the permissions that are available in this system, it only have an id as a primary key and permission name.

## User Permission:

This table acts as a link between the user and the permission, permission is provided for the user by setting the permission id and the user id in the same row ,both columns are primary keys and foreign keys referencing their tables.

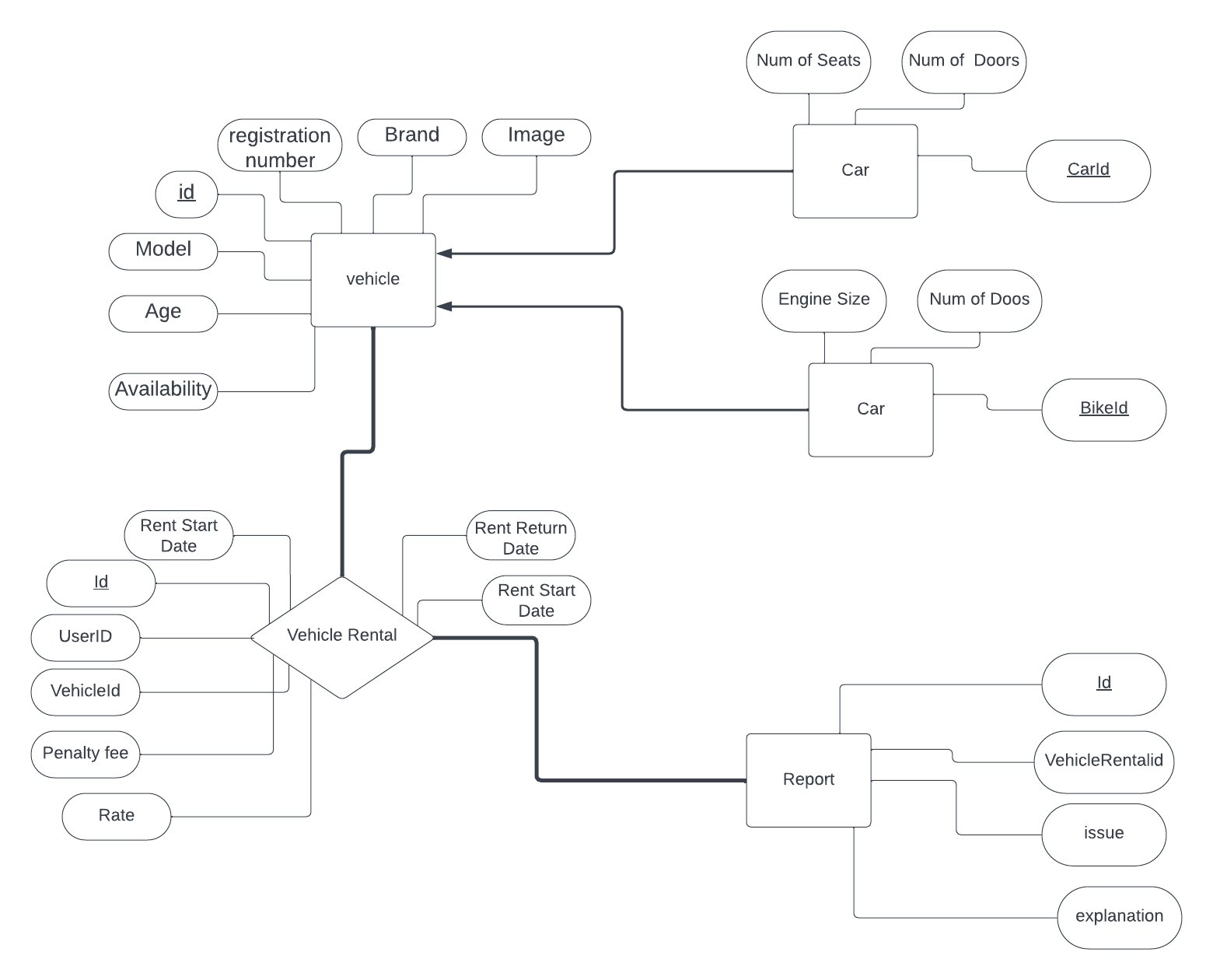
## Vehicle Rental:

This table cover the relation between the renter and the vehicle he want to rent. The table carry the id of the order as a primary key, user id and user id act as a foreign key for their tables, rent start date, rent end date, rent receive date, phone number, penalty fees, and rate.

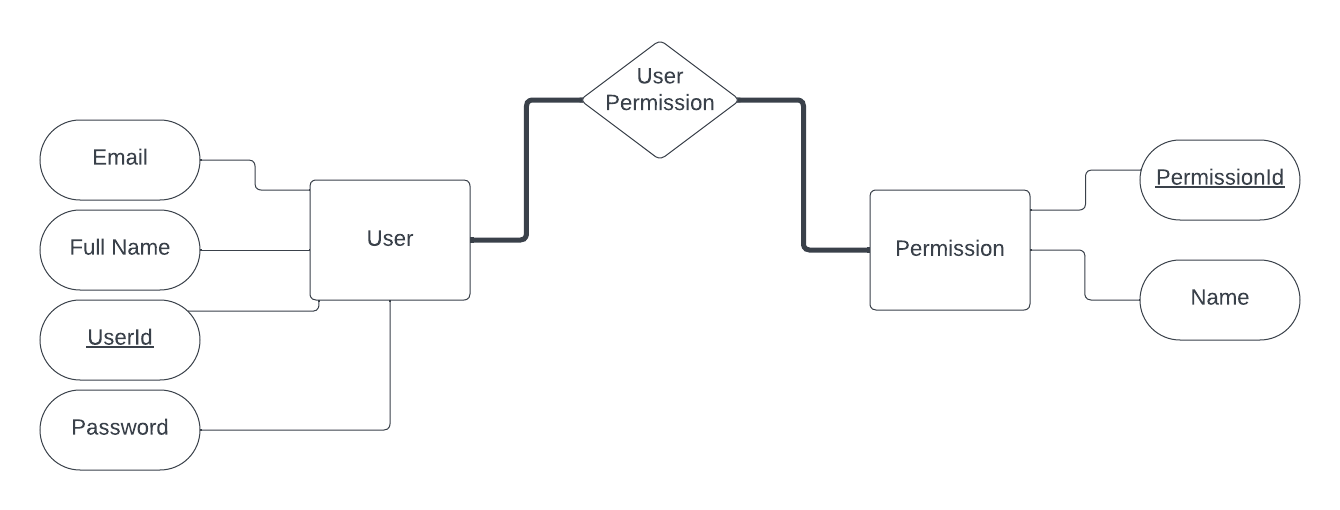
## Reports table:

This table carry the vehicle report ,it hold an id as a primary key , vehicle id as a foreign key referenced from vehicles , issue type, and an explanation.

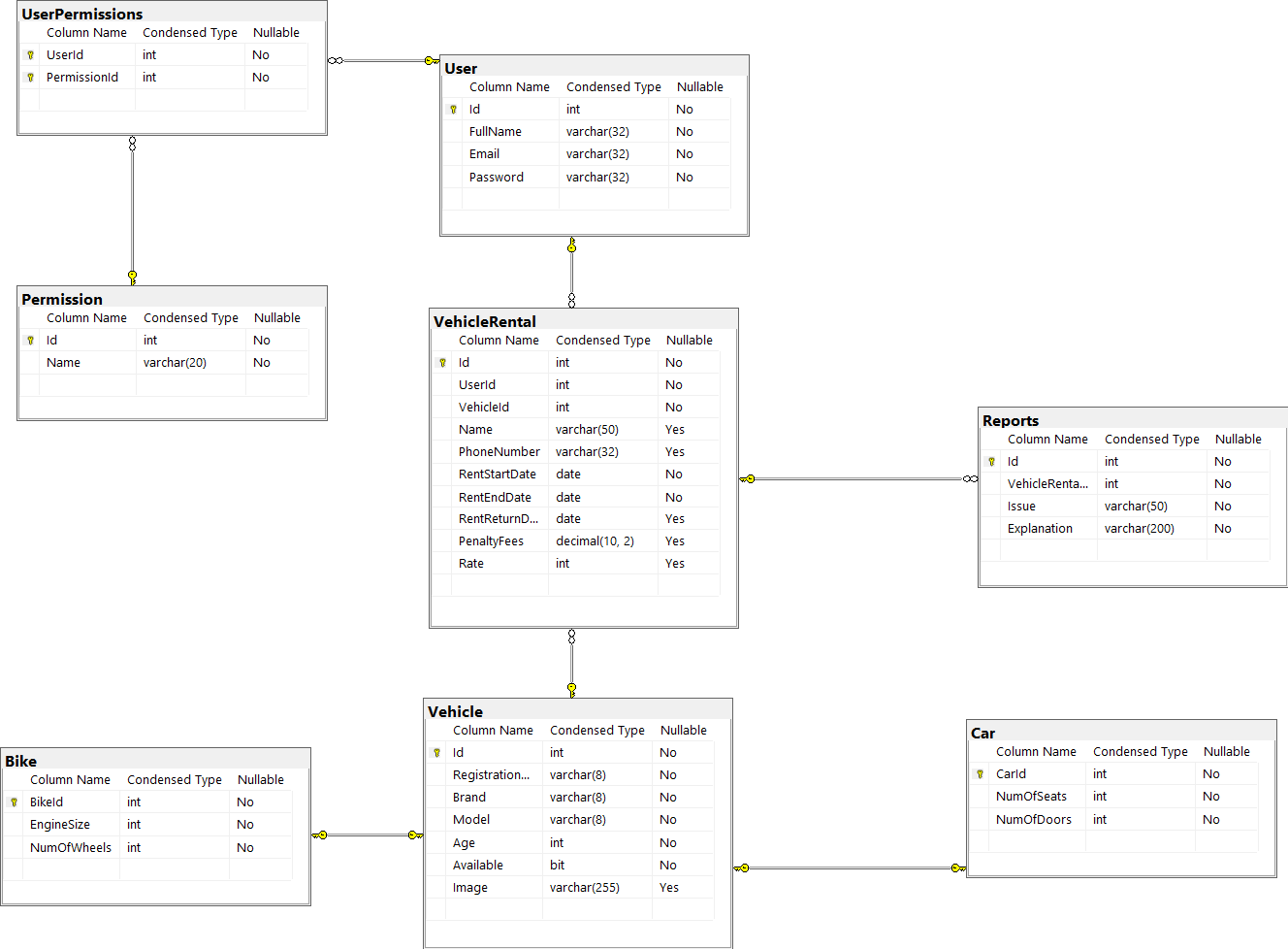
## ER-Diagram:



Phone nb



## Physical Relational Model:



## Database Functions, Procedures, and Triggers

### Functions:

* + CarCost (@CarId int): this function returns the car cost according to its age and number of doors
  + BikeCost (@BikeId int): this function returns the Bike cost according to its engine size
  + VehicleCost (@VehicleId): This helper function determine whether the vehicle is a bike or a car and depending on its type, it calls CarCost or BikeCost function.
  + VehicleRating (@id int): this function returns the average rating of the vehicle that have the determined id.

Remark:: the cost is calculated for a normal rental process without any delay. In case of the delay, the car cost per day will be doubled for each day of delay and added to the normal cost.

* Car cost per day = 30 - (1.5\*age of car) – (num of doors \* 2)
* Bike cost per day = 15 + (engine size/100)

### These are all per day but if there is additional days like for instance a delay for returning the vehicle for 3 days and he initially rented it for 5 days what would happen is

### The user pays the bill for the 5 days: costPerday \*5

### The user pays the penalty fee bill for the 3 days: costPerDay \* 2 \*3

### Procedures:

* + AddCar: this procedure takes as parameters the attributes of the vehicle and car table and inserts these data in a new row.
  + AddBike: this procedure takes as parameters the attributes of the vehicle and Bike table and inserts these data in a new row
  + UpdateYears: this procedure is responsible for incrementing the age of all vehicles by 1
  + GetUserPermissions: returns all the permissions that have been granted for a given user
  + GetRentedVehicles: shows the vehicles that have been rented on a given day
  + GetVehicleReports: lists the reports of each vehicle
  + GenerateRentalReport: generates a rental report for each user/month
  + RentVehicle: rents the given vehicle if it was available

### Triggers:

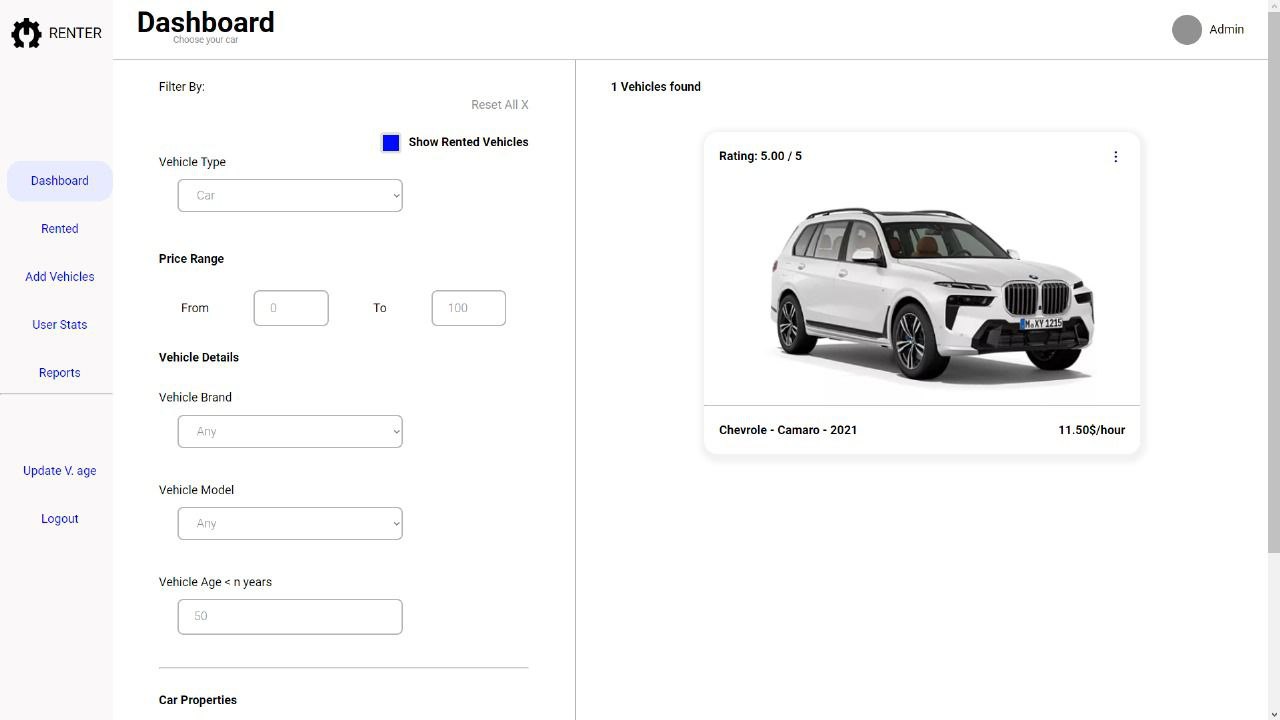
* + PreventVehicleDeletion: prevent the deletion of a vehicle in case it was rented
  + LateReturnPenalty: calculates the penalty fees and set them in their columns in Vehicle Rental table after update
  + UpdateVehicleAvailability: change the availability of a vehicle from available to unavailable and vice-versa depending if it was returned or rented

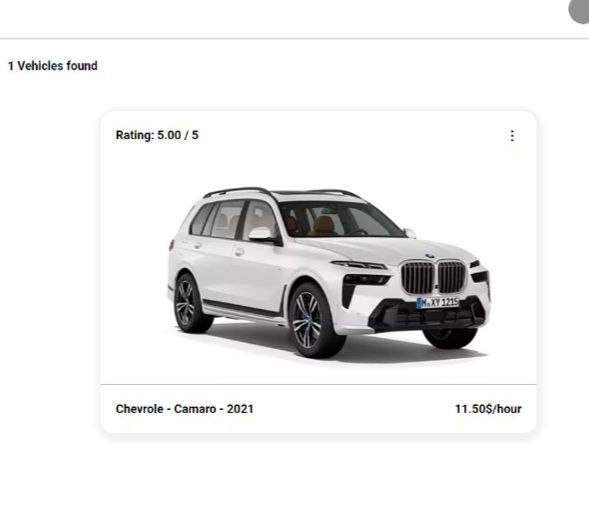
# Chapter 3: User Interface

In this chapter, we will discuss the user interface and how to use it to work efficiently. We will provide many sketches that shows the interface to be familiar with it.

## Dashboard (Home Page)

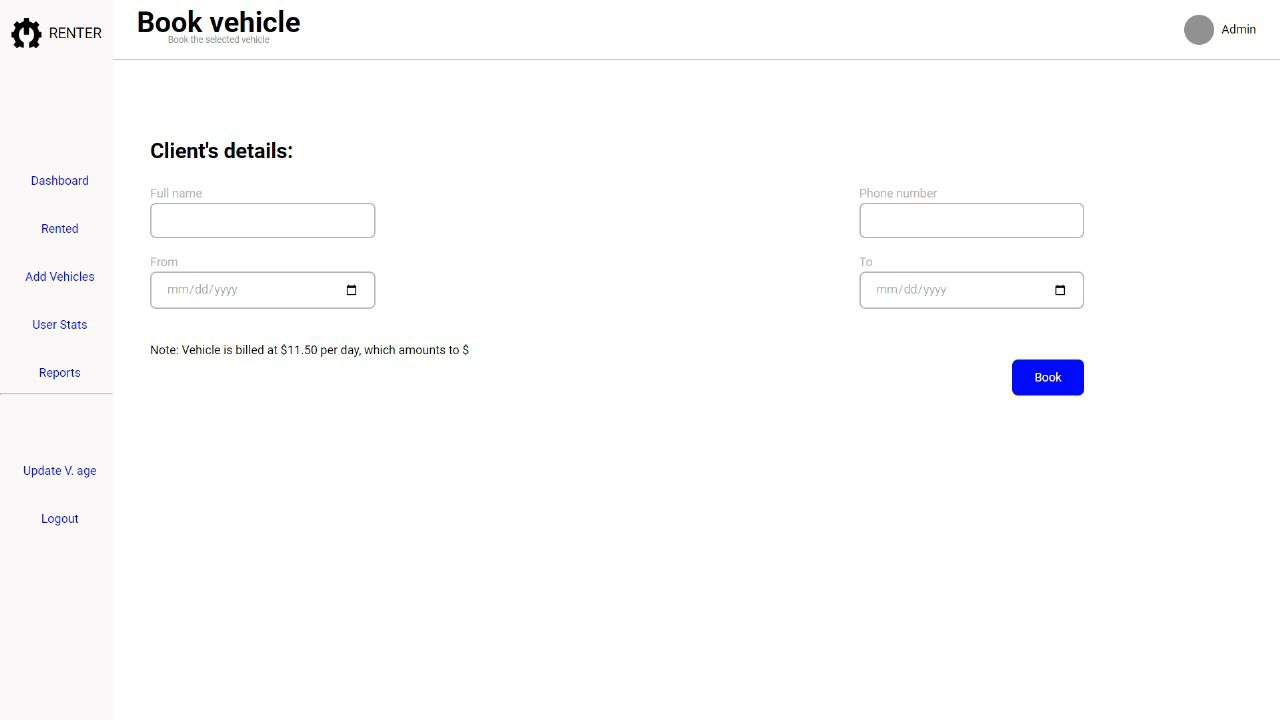
This is the home page where you can browse and search for the vehicle you want.



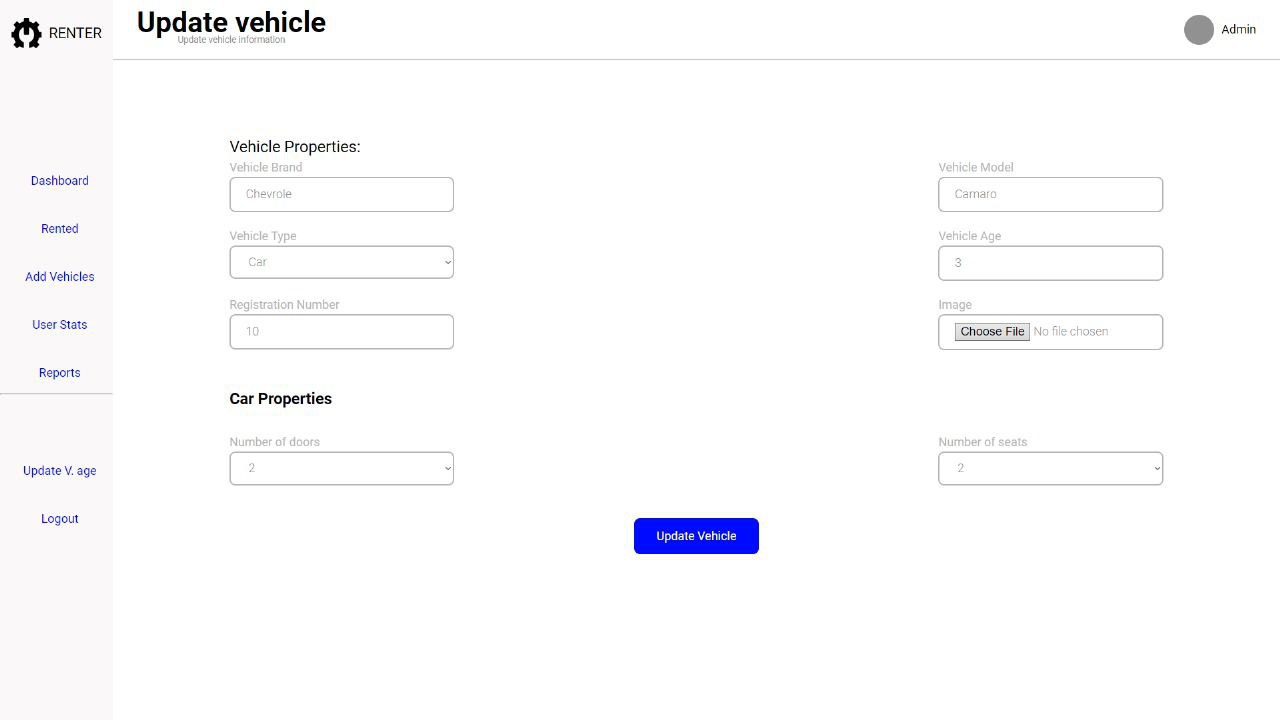


This button opens 2 choices for you, either book the vehicle or you update the vehicle (permission needed for this).

This is where you book the vehicle you want by placing your name, phone and the duration by setting the receiving and returning dates

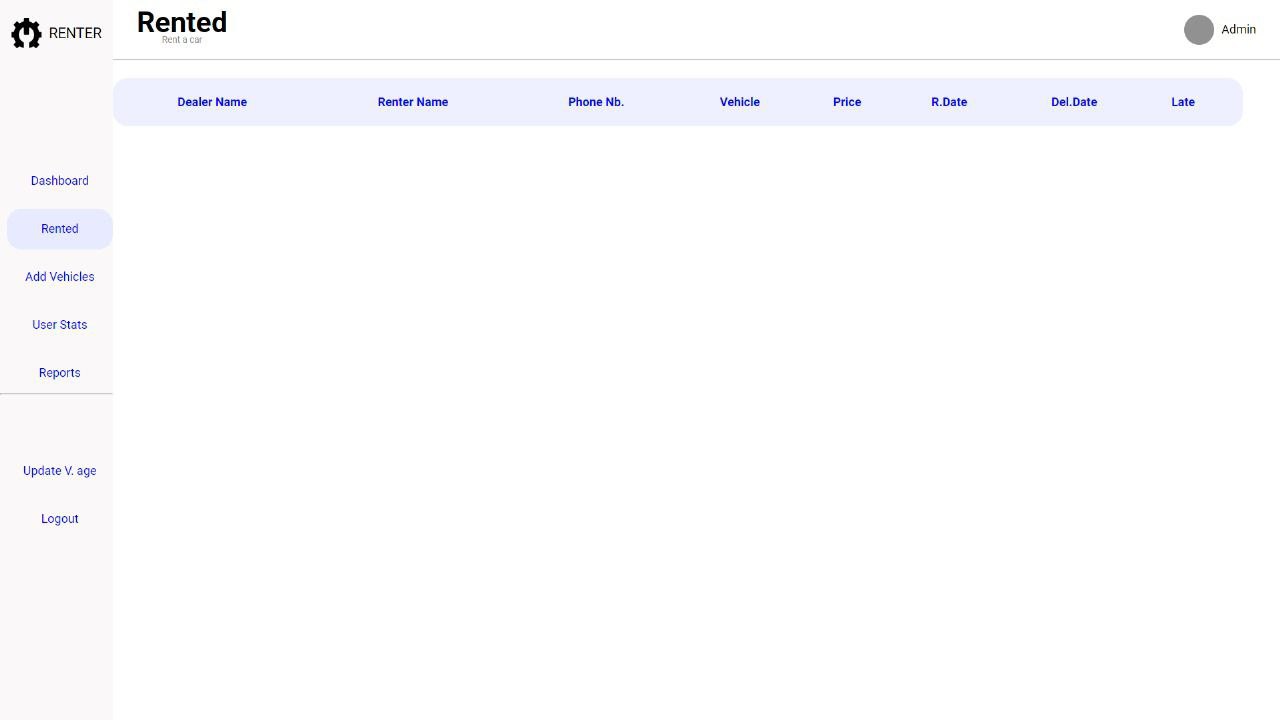


This update vehicle page is accessed with permission, where you can modify vehicle info

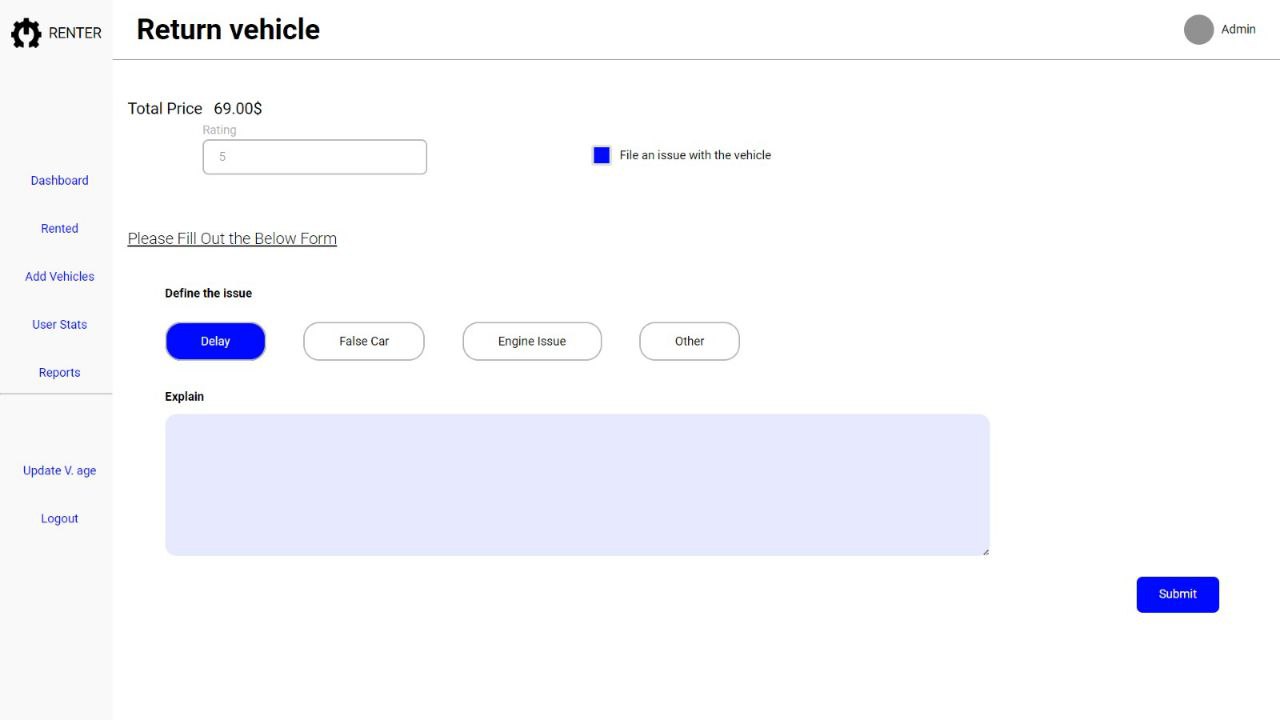


## Rented Page

This is the rented page where the rented but not returned vehicles are displayed here with a return button



Upon clicking the return button the following page is displayed

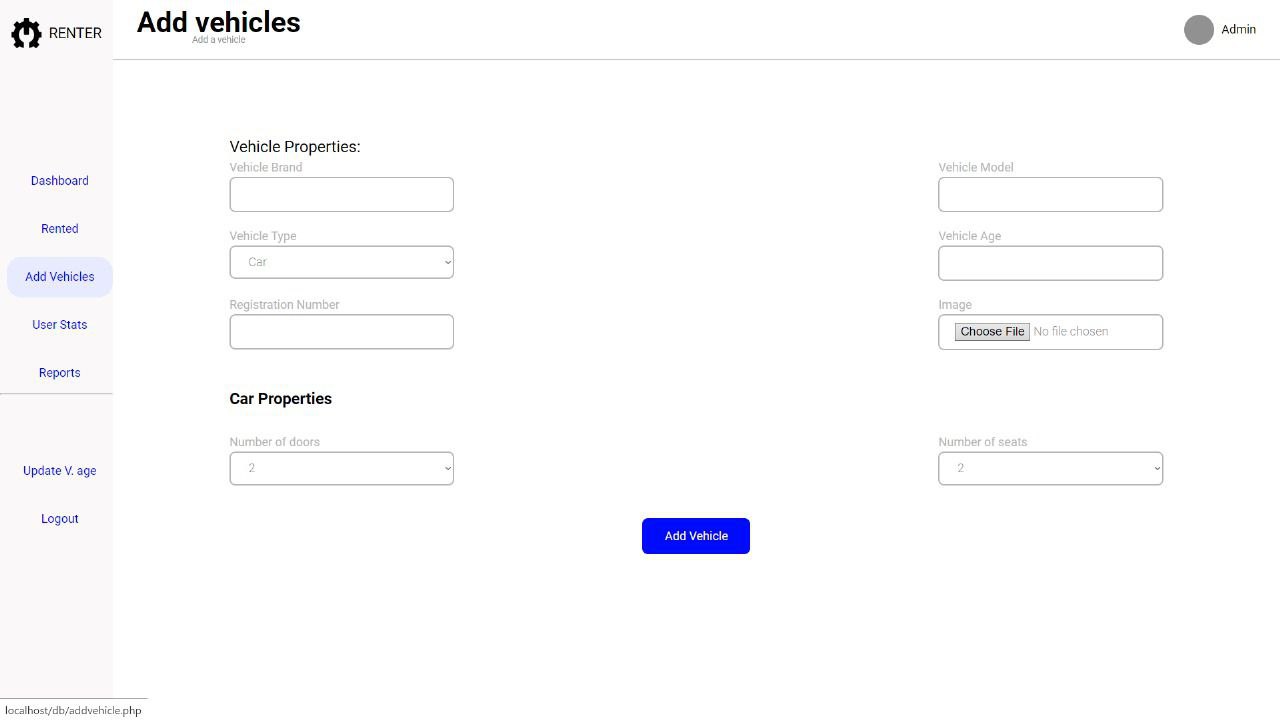


This page shows the total price if the vehicle was returned immediately. If there is any issues, click the checkbox to appear the report form and provide the type and el description of the issue faced

In case of delay, an alert will appear showing number of delayed days and the new total cost.

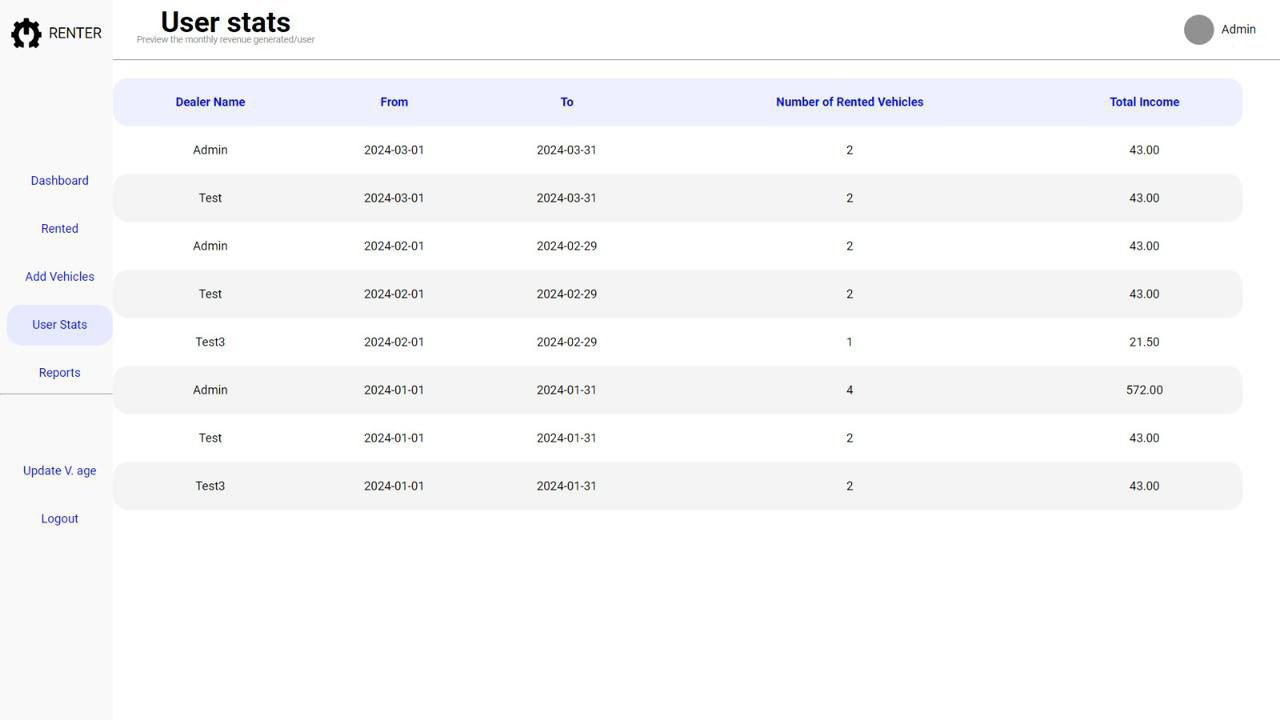
## Add Vehicles

This page require permission to be accessed where admin can add new vehicles



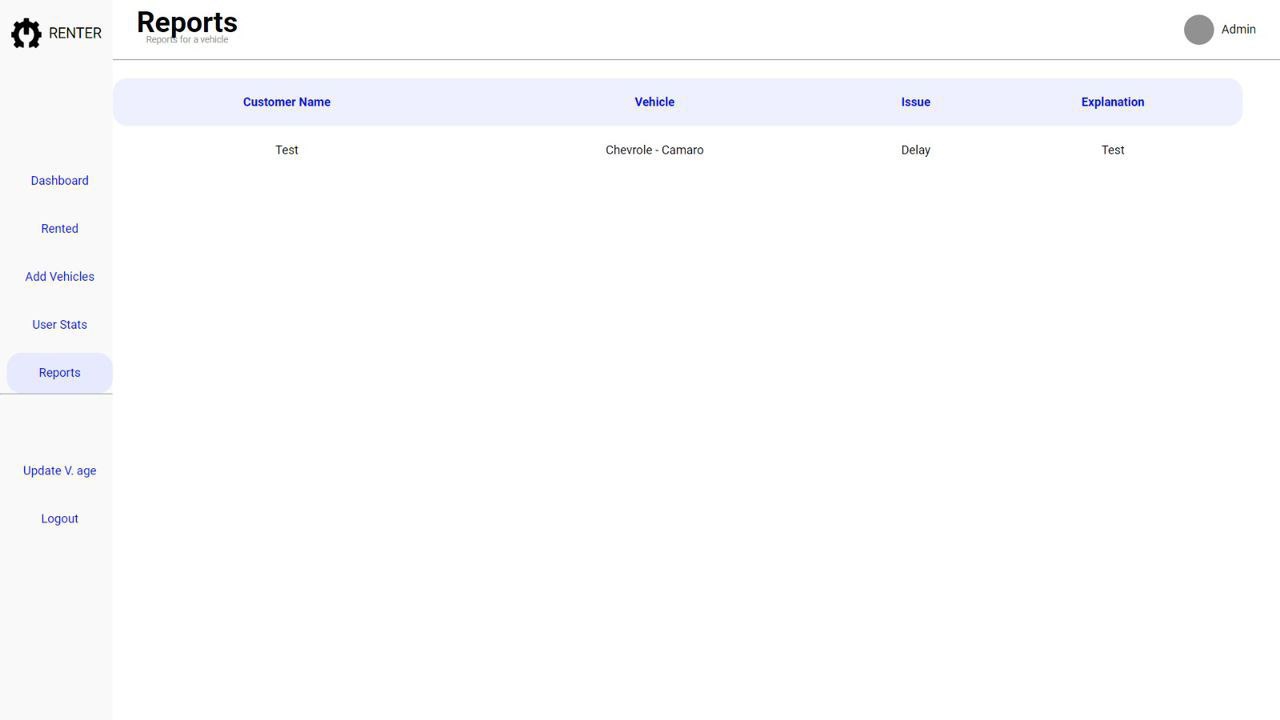
## User Stats

This page shows detail about the users and some stats



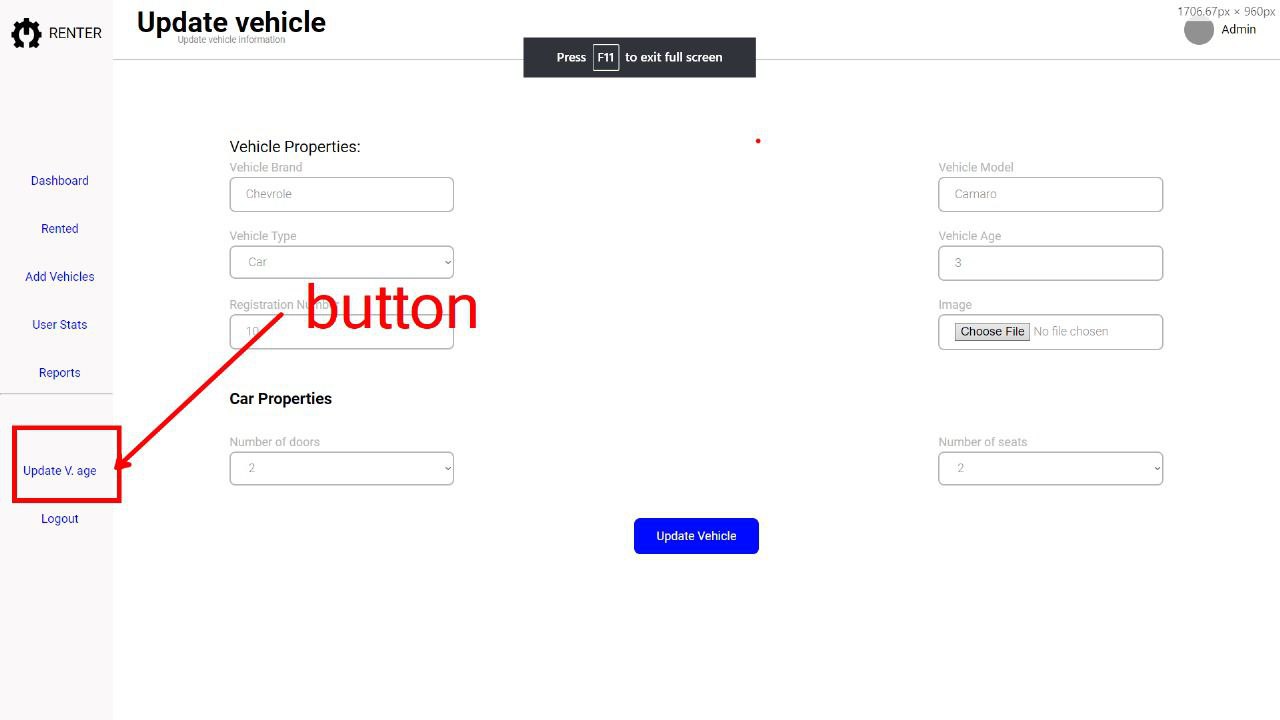
## Reports

This page shows the reports submitted on the rented vehicles



## Update Vehicle Age

Upon clicking on then shown button bellow, all the vehicle ages will increment by 1 year(be 1 year older)



# Chapter 4: Implementation and Testing

During the implementation phase, we undertook tasks such as creating the database tables, establishing relationships, and formulating the application logic. To guarantee the precision of individual components, unit testing was conducted, while integration testing was employed to confirm the smooth communication between these components. Challenges encountered in the implementation process, such as ensuring data validation and implementing security measures, were tackled to fortify the system's resilience.

link to view the implementation 🡺 [Database.sql](GenerationScripts/VehicleRental.sql)

# Conclusion

In conclusion, RENTER stands as a comprehensive system accuratly crafted to enhance the efficiency of car and motorbike rental management. This report has aimed to furnish you with a comprehensive understanding of the countless features and functionalities inherent in our system, facilitating seamless navigation through the diverse facets of vehicle rental and management.