CAR RENTAL SYSTEM

A MINI PROJECT REPORT

Submitted by

RANJITH R 220701218

RANJITH KUMAR R 220701219

PRINCE PERINBARAJ 220701205

In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE

RAJALAKSHMI ENGINEERING COLLEGE (AUTONOMOUS)

THANDALAM

CHENNAI-602105

2023 - 24

BONAFIDE CERTIFICATE

Certified that this project report "**REC CAR RENTAL SYSTEM**" is the bonafide work of **"RANJITH(220701218),RANJITH KUMAR R(220701219),PRINCE PERINBARAJ S(220701205) "** who carried out the project work under my supervision.

Submitted for the Practical Examination held on _________________

SIGNATURURE

Mrs.K.Mahesmeena,
Assistant Proffesor(SG),
Computer Science and Engineering,
Rajalakshmi Engineering College,
(Autonomous),
Thandalam, Chennai -602105

Abstract

The Car Rental System (CRS) is a robust and efficient platform designed to facilitate the hiring of vehicles, providing a streamlined experience for users, administrators, drivers, and staff. This project report outlines the development and functionalities of the CMS, which encompasses four main modules: Admin, User, Driver, and Staff.

The Admin Module allows administrators to manage vehicle categories, add new vehicles, and create driver accounts, while also providing capabilities to monitor booking statuses in real-time. The User Module offers a user-friendly interface for users to register, log in, view available vehicles, make bookings, and track their booking status. The Driver Module enables drivers to manage ride requests and update ride statuses throughout the journey. The Staff Module provides staff members with similar functionalities to the admin, excluding the ability to add new vehicle categories.

Designed using a combination of frontend technologies (HTML, CSS, JavaScript), backend frameworks (PHP or Node.js), and databases (MySQL or MongoDB), the CRS ensures secure authentication and efficient data management. Rigorous testing methods, including unit, integration, system, and user acceptance testing, were employed to ensure the system's reliability and effectiveness.

Future enhancements, such as integrating online payment options, implementing a rating system, incorporating GPS tracking, and developing mobile applications, are proposed to further enhance the system's functionality and user experience. This project demonstrates the successful implementation of a car rental system solution that meets the needs of its diverse user base while ensuring operational efficiency and user satisfaction.

Table of Contents

1.INTRODUCTION

1.1. INTRODUCTION

The Car Rental System (CRS) is a comprehensive application designed to streamline the process of hiring vehicles. This system provides a seamless experience for users to hire vehicles, allows administrators to manage vehicle categories and details, and enables drivers and staff to manage bookings and ride statuses efficiently.

1.2. OBJECTIVES

- To develop a user-friendly interface for vehicle hiring and management
- To implement a database system for storing and retrieving vehicle and booking information
- To provide functionalities for adding and managing vehicle categories and driver assignments
- To enable real-time updates and tracking of booking statuses
- To ensure scalability and security for handling growth and protecting

1.3. MODULES

- Admin module
- User module
- Staff module
- Backend services module
- Database module

2.SURVEY OF TECHNOLOGIES

2.1. SOFTWARE DESCRIPTION

The Car Rental system project utilizes various software technologies to ensure a robust and scalable system. The core technologies include PHP for server-side scripting, SQL for database management, and Bootstrap for responsive design.

2.2. LANGUAGES

The Car Rental system project leveraged several programming languages and technologies to build the system. Each language was chosen for its specific strengths and contributions to different aspects of the project

2.2.1 PHP

PHP (Hypertext Preprocessor) is a widely-used open-source scripting language suited for web development. It was employed for server-side scripting to handle data processing, database interactions, and dynamic content generation. Key features of PHP utilized in this project include

- Server-side scripting
- Form handling
- Database connectivity using MySQL
- Session management

2.2.2 SQL

SQL (Structured Query Language) is used for managing and manipulating relational databases. The project used SQL to interact with the MySQL database for various CRUD (Create, Read, Update, Delete) operations. Key SQL functionalities include:

- Data definition and manipulation (DDL and DML)
- Querying the database

- Data normalization and integrity constraints

2.2.3 HTML

HTML (Hypertext Markup Language) is the standard language for creating web pages. It was used to structure the content of the car rental system. Key features of HTML utilized in this project include:

- Page structuring with elements like headings, paragraphs, lists, and forms
- Embedding images and media
- Creating links and navigation

2.2.4 CSS

CSS (Cascading Style Sheets) is used to style and layout web pages.CSS was employed to ensure that the car rental system had an attractive and responsive design. Key CSS features utilized in this project include:

- Layout design using Flexbox and Grid
- Styling of HTML elements (e.g., colors, fonts, spacing)
- Responsive design techniques for various devices

2.2.5 JAVASCRIPT

JavaScript is a versatile programming language used for adding interactivity to web pages. In this project, JavaScript was used for client-side scripting to enhance user experience. Key JavaScript features utilized include:

- DOM manipulation for dynamic content updates
- Event handling (e.g., form validation, button clicks)
- AJAX for asynchronous data loading

2.2.6 JQUERY

jQuery is a fast, small, and feature-rich JavaScript library. It simplifies tasks like HTML document traversal and manipulation, event handling, and AJAX interactions. Key jQuery functionalities used in this project include:

- Simplified AJAX requests
- DOM manipulation and traversal
- Enhancing user interface elements (e.g., animations, effects)

2.2.7 BOOTSTRAP

Bootstrap is a popular front-end framework for developing responsive and mobile-first web projects. It was used to design and develop a responsive layout for the car rental system. Key Bootstrap features utilized in this project include:

- Grid system for responsive design
- Pre-designed components like navbars, buttons, and forms
- Utility classes for spacing, alignment, and typography

These languages and technologies were chosen for their robustness, community support, and ability to deliver a seamless user experience. The integration of these technologies resulted in a functional and efficient car rental system.

3.REQUIREMENTS AND ANALYSIS

3.1 REQUIREMENT SPECIFICATION

- User Requirements:

  The system should be accessible via web browsers and mobile devices.

- System Requirements:

  The system should ensure data integrity and provide a secure environment for user information.

3.2. HARDWARE AND SOFTWARE REQUIREMENTS

- Hardware:

  i. Server with minimum 4GB RAM and 100GB storage.

  ii. Client devices with internet access.

- Software:

  i. Operating System: Linux/Windows

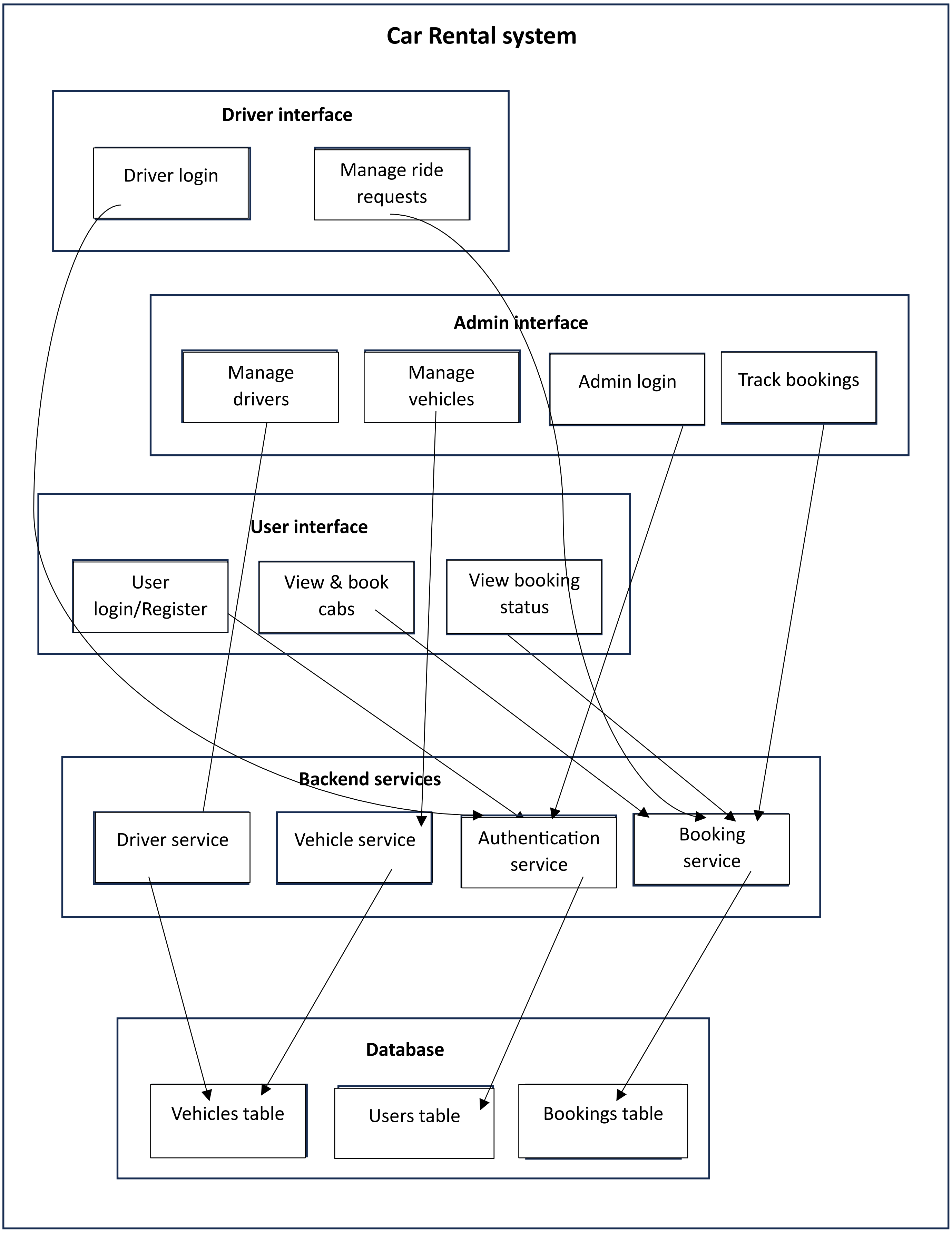
  ii. Web Server: Apache

  iii. Database: MySQL

  iv. Languages: PHP, SQL, HTML, CSS, JavaScript

  v. Frameworks: Bootstrap

3.3 ARCHITECTURAL DIAGRAM



EXPLANATION:

Car Rental System:

- User Interface: Simplified to core actions.
    - User Login/Register
    - View & Book Cabs
    - View Booking Status

- Admin Interface: Simplified to core admin actions.
    - Admin Login
    - Manage Vehicles
    - Manage Drivers
    - Track Bookings

- Driver Interface: Simplified to core driver actions.
    - Driver Login
    - Manage Ride Requests

- Backend Services: Core backend services.
    - Authentication Service
    - Booking Service
    - Vehicle Service
    - Driver Service

- Database: Core tables.
    - Users Table
    - Vehicles Table
    - Bookings Table

INTERACTIONS:

Car Rental system:

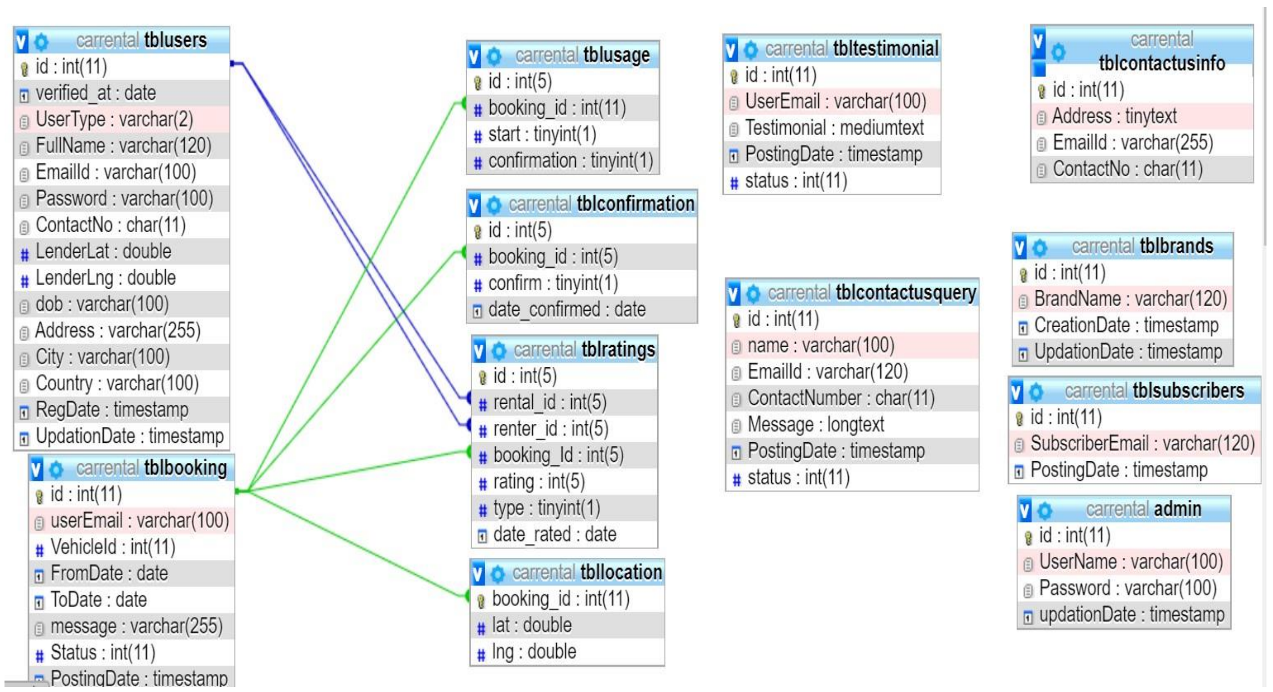
- User Interface
    - User Login/Register interacts with the Authentication Service.
    - View & Book Cabs interacts with the Booking Service.
    - View Booking Status interacts with the Booking Service.

- Admin Interface
    - Admin Login interacts with the Authentication Service.
    - Manage Vehicles interacts with the Vehicle Service.
    - Manage Drivers interacts with the Driver Service.
    - Track Bookings interacts with the Booking Service.

- Driver Interface
    - Driver Login interacts with the Authentication Service.
    - Manage Ride Requests interacts with the Booking Service.

- Backend Services
    - Authentication Service interacts with the Users Table.
    - Booking Service interacts with the Bookings Table.
    - Vehicle Service interacts with the Vehicles Table.
    - Driver Service interacts with the Vehicle Table.

3.4 ER DIAGRAM



EXPLANATION

Caar Rental System

The Entity-Relationship (ER) diagram provides a visual representation of the database structure for the Car Rental System (CRS). This diagram includes five main entities: Users, Vehicles, Categories, Bookings, and Staff. Below is a detailed explanation of each entity, their attributes, and the relationships between them.

Entities and Attributes

1.Users

- user_id: INT, Unique identifier for each user.
- username: VARCHAR, The name used by the user for login.
- password: VARCHAR, The password for user authentication.
- email: VARCHAR, The email address of the user.
- mobile_number: VARCHAR, The contact number of the user.

2.Vehicles

- vehicle_id: INT, Unique identifier for each vehicle.
- vehicle_name: VARCHAR, The name of the vehicle.
- registration_number: VARCHAR, The registration number of the vehicle.
- chassis_number: VARCHAR, The chassis number of the vehicle.
- driver_id: INT, Foreign key linking to the driver associated with the vehicle.
- category_id: INT, Foreign key linking to the vehicle category.

3.Categories

- category_id: INT, Unique identifier for each vehicle category.
- category_name: VARCHAR, The name of the category (e.g., Sedan, SUV, Van).

4.Bookings

- booking_id**: INT, Unique identifier for each booking.
- user_id**: INT, Foreign key linking to the user who made the booking.
- vehicle_id**: INT, Foreign key linking to the booked vehicle.
- pickup_location**: VARCHAR, The starting location for the booking.
- dropoff_location**: VARCHAR, The destination for the booking.
- status**: VARCHAR, The current status of the booking (e.g., Requested, Accepted, Picked Up, Dropped Off).

5.Staff

- staff_id**: INT, Unique identifier for each staff member.
- name**: VARCHAR, The name of the staff member.
- password**: VARCHAR, The password for staff authentication.
- mobile_number**: VARCHAR, The contact number of the staff member.

RELATIONSHIPS

1. Users and Bookings

- Relationship: Users make Bookings.
- Description: A User can make multiple Bookings, but each Booking is made by only one User. This is represented by the relationship where a User has a one-to-many relationship with Bookings.

2.Bookings and vehicles

- Relationship: Bookings involve Vehicles.
- Description: A Booking involves one Vehicle, and each Vehicle can be involved in multiple Bookings over time. This is represented by the relationship where a Booking has a many-to-one relationship with Vehicles.

3.Vehicles and categories

- Relationship: Vehicles belong to Categories.
- Description: Each Vehicle belongs to one Category, but each Category can include multiple Vehicles. This is represented by the relationship where a Vehicle has a many-to-one relationship with Categories.

4.Staff and vehicles

- Relationship: Staff manages Vehicles.
- Description: Staff manage the details of Vehicles. Each Staff member can manage multiple Vehicles, and each Vehicle can be managed by multiple Staff members. This is represented by the relationship where Staff and Vehicles have a many-to-many relationship.

4.PROGRAM CODE

```
<?php require_once('config.php'); ?>
<!DOCTYPE html>
<html lang="en">
<?php require_once('inc/header.php') ?>
<body>
<?php $page = isset($_GET['p']) ? $_GET['p'] : 'home';  ?>
<?php require_once('inc/topBarNav.php') ?>
     <?php if($_settings->chk_flashdata('success')): ?>
      <script>
        alert_toast("<?php echo $_settings->flashdata('success') ?>",
        'success')
      </script>
<?php endif;?>
<?php
    if(!file_exists($page.".php") && !is_dir($page)){
        include '404.html';
    }else{
      if(is_dir($page))
        include $page.'/index.php';
      else
        include $page.'.php';
    }
?>
<?php require_once('inc/footer.php') ?>

  <div class="modal fade" id="uni_modal" role='dialog'>
    <div class="modal-dialog   rounded-0 modal-md modal-dialog-centered"
    role="document">
      <div class="modal-content  rounded-0">
        <div class="modal-header">
        <h5 class="modal-title"></h5>
      </div>
      <div class="modal-body">
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-primary" id='submit'
        onclick="$('#uni_modal form').submit()">Save</button>
        <button type="button" class="btn btn-secondary"
        data-dismiss="modal">Cancel</button>
        </div>
        </div>
      </div>
    </div>
    <div class="modal fade" id="uni_modal_right" role='dialog'>
      <div class="modal-dialog  rounded-0 modal-full-height  modal-md"
      role="document">
        <div class="modal-content rounded-0">
          <div class="modal-header">
          <h5 class="modal-title"></h5>
          <button type="button" class="close" data-dismiss="modal"
          aria-label="Close">
            <span class="fa fa-arrow-right"></span>
          </button>
        </div>
        <div class="modal-body">
        </div>
        </div>
      </div>
    </div>
    <div class="modal fade" id="viewer_modal" role='dialog'>
      <div class="modal-dialog modal-md" role="document">
        <div class="modal-content">
                <button type="button" class="btn-close"
                data-dismiss="modal"><span class="fa fa-times"></span></
                button>
                <img src="" alt="">
        </div>
      </div>
    </div>
    <div class="modal fade" id="confirm_modal" role='dialog'>
      <div class="modal-dialog modal-md modal-dialog-centered"
      role="document">
        <div class="modal-content">
          <div class="modal-header">
          <h5 class="modal-title">Confirmation</h5>
        </div>
        <div class="modal-body">
          <div id="delete_content"></div>
        </div>
        <div class="modal-footer">
          <button type="button" class="btn btn-primary" id='confirm'
          onclick="">Continue</button>
          <button type="button" class="btn btn-secondary"
          data-dismiss="modal">Close</button>
        </div>
        </div>
      </div>
    </div>

  </body>
  </html>
```

5.RESULTS AND DISCUSSION

5.1 FUNCTIONALITY OF THE PROJECT

The Car Rental System (CRS) is designed to facilitate the efficient hiring and management of vehicles. Below are the detailed functionalities of the project, categorized by user roles:

Functionalities of the Car Rental System

1.Admin Module

Features:

- Admin login:
    - Secure login for administrators to access the system.

- Add Vehicle Category:
    - Allows the admin to create new categories of vehicles (e.g., Sedan, SUV, Van).
    - Category details include the category name.

- Add Staff:
    - Enables the admin to add new staff members to assist in managing the system.
    - Staff details include name, contact information, and login credentials.

- Add New Vehicles:
    - Admin can add new vehicles under specific categories.
    - Vehicle details include:
        1. Vehicle Name
        2. Vehicle Registration Number

3. Chassis Number
4. Driver Name
5. Password for Driver Account
6. Driver Mobile Number

- Create Driver Account:
    - Automatically creates a driver account using the provided driver details.

- Register Vehicle Number:
    - Generates a unique registration number for each vehicle.

- Track Booking Status:
    - Admin can monitor the status of all bookings in real-time.
    - View details such as current ride progress and interactions between drivers and users.

2.User module

Features:

- User Registration:
    - Allows new users to register by providing their details such as name, email, mobile number, and password.

- User Login:
    - Secure login for registered users.

- View Available Cabs:
  - Users can see a list of all available vehicles for hire, categorized by vehicle type.

- Book a Vehicle:
  - Users can select a vehicle from the list of available vehicles.
  - Enter pickup and drop-off locations to make a booking request.

- Booking Requests:
  - Sends the booking request to the driver associated with the selected vehicle.

- View Booking Status:
  - Users can view the status of their current and past bookings, including accepted, picked up, and dropped statuses.

3.Driver module

Features:

- Driver Login:
  - Secure login for drivers.

- Manage Ride Requests:
  - Drivers can view and manage ride requests from users.
  - Options to accept or reject ride requests.

- Update Ride Status:
  - Drivers can update the status of the ride at various stages, including:
    1. Request Accepted

2. Picked Up Passenger
3. Dropped Passenger

4.Staff module

Features:

- Staff Login:
    - Secure login for staff members.

- Manage Vehicle Details:
    - Staff can manage vehicle details, including updating vehicle information.
    - Similar functionality to the admin for managing vehicles but without the ability to add new vehicle categories.

- Track Booking Status:
    - Staff can monitor the status of all bookings and assist in operational tasks.

Database design

The CRS database consists of several tables to manage users, drivers, vehicles, categories, bookings, and staff information. Key tables include:

- Users Table:
    - Stores user details such as user ID, username, password, email, and mobile number.

- Vehicles Table:
    - Stores vehicle details such as vehicle ID, name, registration number, chassis number, driver ID, driver name, driver address and category ID.

- Categories Table:
  - Stores vehicle category details such as category ID and category name.

- Bookings Table:
  - Stores booking details such as booking ID, user ID, vehicle ID, pickup location, drop-off location, and status.

- Staff Table:
  - Stores staff details such as staff ID, name, password, and mobile number.

5.2 DISCUSSION

The development and deployment of the CRS highlighted several important insights and potential areas for improvement:

1.User Experience:

- The user interface (UI) is critical for ensuring ease of use for all user roles. Feedback from initial testing indicated that while the basic functionality was accessible, the UI could be enhanced for better navigation and user satisfaction. Future iterations could incorporate more intuitive design elements and user guides.

2. Scalability:

- As the number of users and vehicles increases, the system needs to maintain performance and responsiveness. The current implementation with SQLite is suitable for initial stages, but migrating to a more robust database system like PostgreSQL or MySQL might be necessary for scalability.

3.Security:

- Security measures such as password hashing, secure session management, and role-based access control were implemented. Continuous security assessments are essential to protect user data and prevent unauthorized access.

4. Real-Time Updates:

- Implementing real-time updates for booking statuses using WebSockets or similar technologies would significantly enhance the user experience. This feature would allow users and drivers to receive instant notifications about booking status changes.

5. Integration with Mapping Services:

- Integrating with mapping services like Google Maps or Mapbox could provide enhanced functionality for specifying pickup and drop-off locations, estimating travel times, and improving route planning.

6. Feedback Mechanism:

- Incorporating a feedback mechanism where users can rate drivers and provide comments on their ride experience could help improve service quality and provide valuable insights for management.

7. Automation and AI:

- Future enhancements could include automation of certain administrative tasks and the use of AI for predictive analytics, such as predicting demand for vehicles based on historical data and optimizing resource allocation.

5.3 CHALLENGES FACED DURING DEVELOPMENT

1. Complex User Role Management:

- Challenge: Implementing distinct functionalities and permissions for different user roles (admin, staff, driver, user) required careful planning and robust access control mechanisms.
- Solution: Role-based access control (RBAC) was implemented to ensure that each user type could only access and perform actions relevant to their role.

2. Real-Time Booking Updates:

- Challenge: Ensuring that booking status updates (e.g., accepted, picked up passenger, dropped passenger) were reflected in real-time for all relevant users.
- Solution: Incorporating technologies like WebSockets for real-time communication between the server and clients to provide instant updates on booking statuses.

3. User Interface Design:

- Challenge: Designing an intuitive and user-friendly interface that caters to the needs of different user roles, each with distinct functionalities.
- Solution: Iterative design and user testing were conducted to refine the UI, ensuring it was easy to navigate and met the requirements of each user type.

4. Database Design and Management:

- Challenge: Creating a robust database schema to efficiently store and retrieve complex relationships between users, vehicles, bookings, and categories.
- Solution: Normalized database design was implemented to minimize redundancy and ensure efficient query performance, with SQLAlchemy ORM for handling database interactions.

5. Scalability Concerns:

- Challenge: Ensuring the system could scale to accommodate a growing number of users, vehicles, and bookings without performance degradation.
- Solution: Initial development used SQLite for simplicity, with plans for migration to more scalable databases like PostgreSQL as the system grows. Optimization techniques and load testing were also employed to ensure scalability.

6. Security Measures:

- Challenge: Protecting user data and ensuring secure access control to prevent unauthorized access and data breaches.
- Solution: Implemented security best practices such as password hashing, secure session management, HTTPS, and regular security audits to safeguard the system.

7. Driver and Vehicle Assignment Logic:

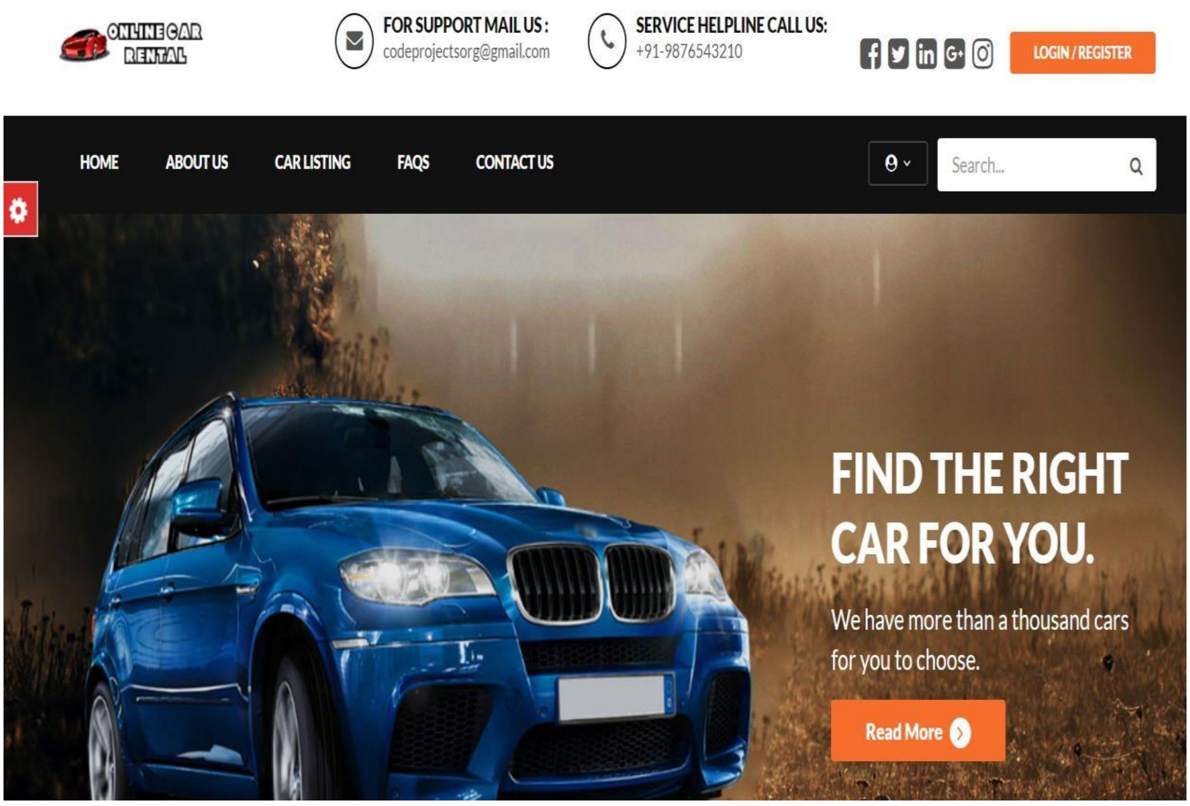
- Challenge: Developing an efficient logic for assigning drivers to vehicles and handling booking requests based on availability and proximity.
- Solution: Algorithms were designed to match booking requests with available drivers and vehicles, taking into account factors like current location and booking history.

8. Handling Concurrent Bookings:

- Challenge: Managing concurrent booking requests to avoid conflicts and ensure accurate vehicle availability.
- Solution: Transaction management and locking mechanisms were implemented to handle concurrent access to booking records, ensuring data consistency.

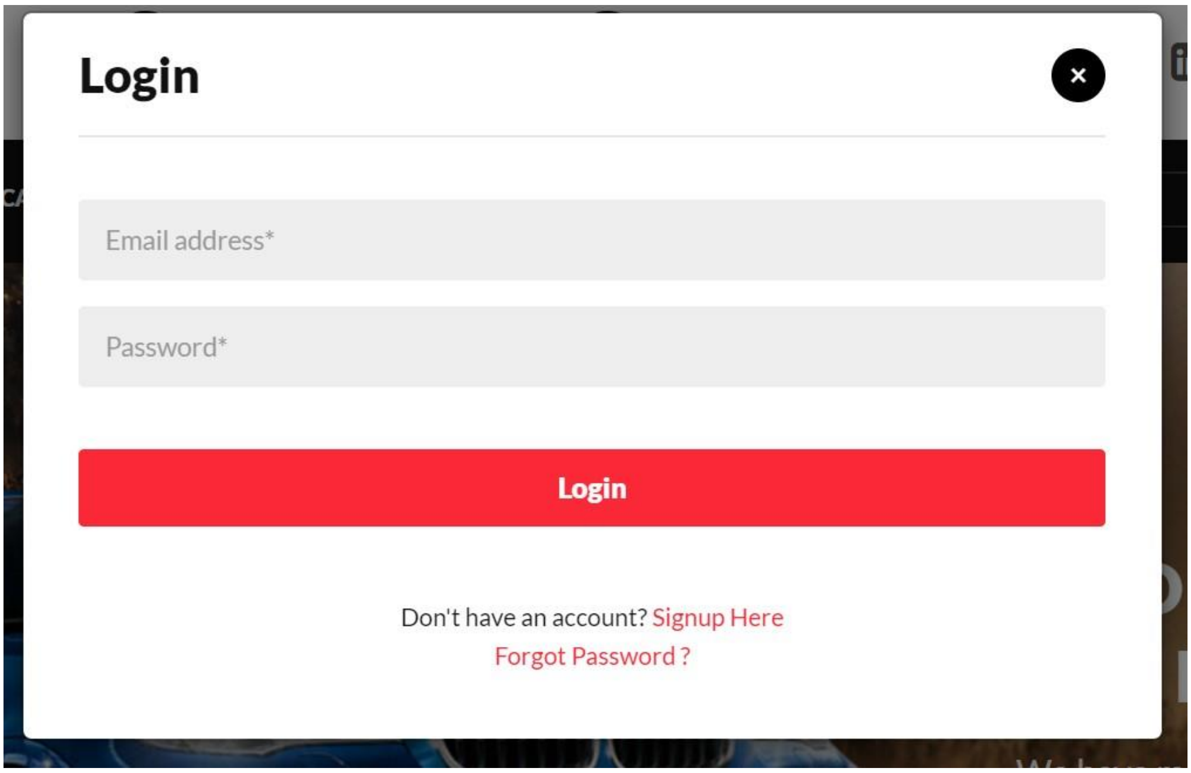
Application's Output

Frontend


ONLINE CAR RENTAL
FOR SUPPORT MAIL US :
codeprojectsorg@gmail.com
SERVICE HELPLINE CALL US:
+91-9876543210
LOGIN / REGISTER
HOME
ABOUT US
CAR LISTING
FAQS
CONTACT US
Search...
FIND THE RIGHT CAR FOR YOU.
We have more than a thousand cars for you to choose.
Read More

Home Page

Login Page



Login

Email address*

Password*

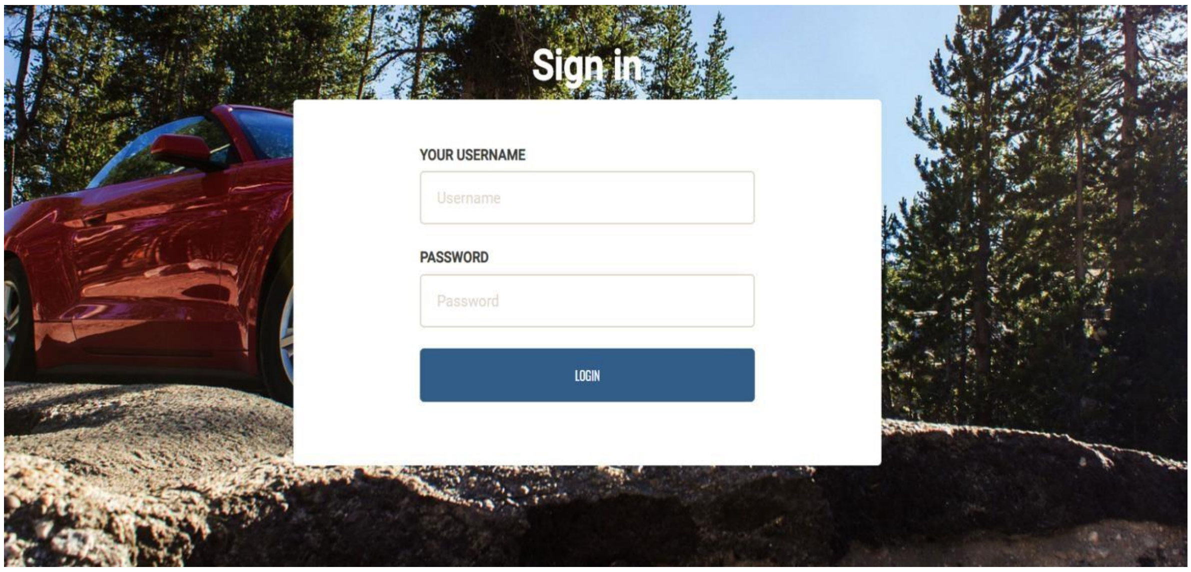
Login

Don't have an account? Signup Here

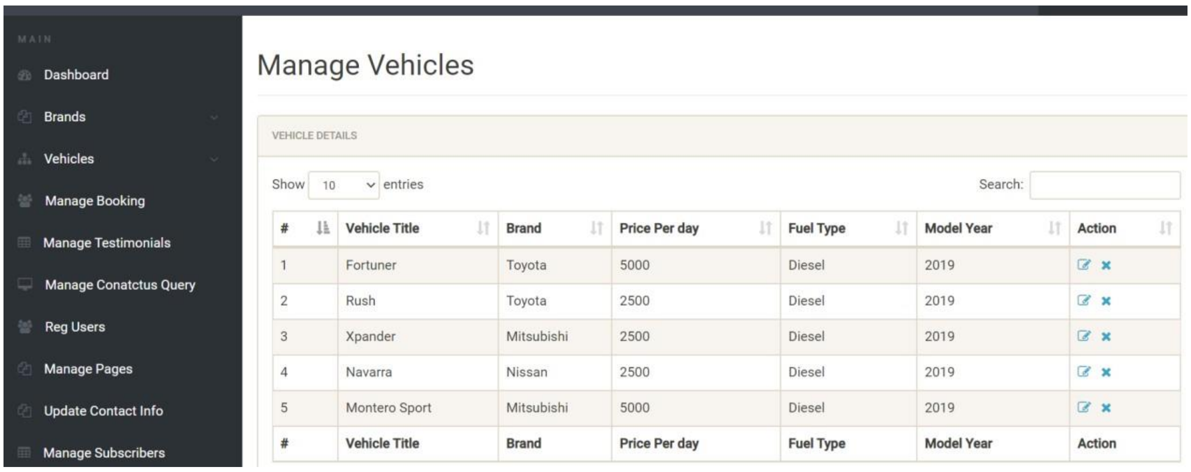
Forgot Password ?

Backend

Admin Login page



Manage Vehicles



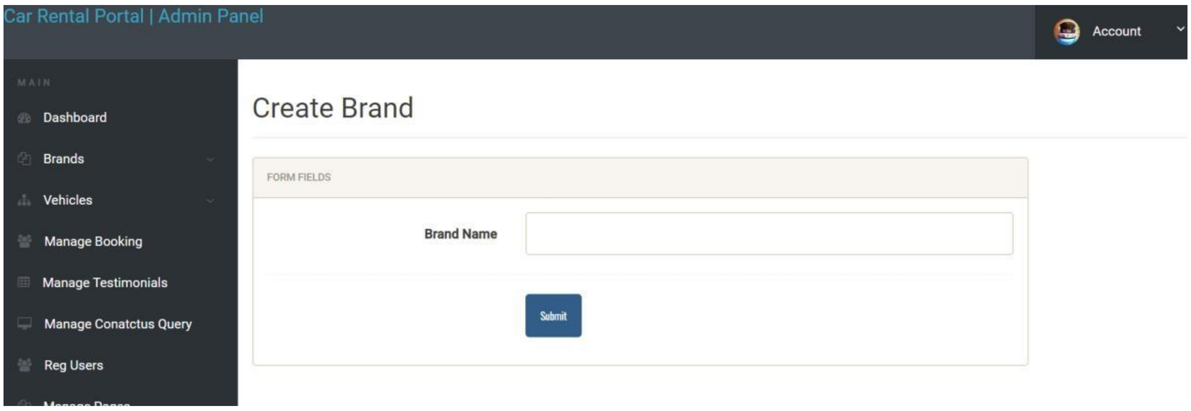
Manage Vehicles

VEHICLE DETAILS

Show 10 entries Search:

#	Vehicle Title	Brand	Price Per day	Fuel Type	Model Year	Action
1	Fortuner	Toyota	5000	Diesel	2019	
2	Rush	Toyota	2500	Diesel	2019	
3	Xpander	Mitsubishi	2500	Diesel	2019	
4	Navarra	Nissan	2500	Diesel	2019	
5	Montero Sport	Mitsubishi	5000	Diesel	2019	
#	Vehicle Title	Brand	Price Per day	Fuel Type	Model Year	Action

Add Brand



Create Brand

FORM FIELDS

Brand Name

CONCLUSION

In conclusion, a well-designed car rental system offers numerous benefits for both customers and rental companies alike. For customers, it provides convenience, flexibility, and access to a wide range of vehicles to suit their needs. It also offers competitive pricing and streamlined reservation processes through online platforms or mobile apps. On the other hand, rental companies benefit from increased efficiency in managing their fleet, maximizing utilization rates, and enhancing customer satisfaction through improved service delivery. Additionally, the integration of advanced technologies such as GPS tracking, automated check-in/out systems, and predictive maintenance further enhances the overall rental experience. Overall, a robust car rental system is essential for meeting the evolving needs of modern consumers while driving operational excellence for rental businesses.