CAR SELLING WEBSITE

A Project Report
Submitted in partial fulfilment of the
Requirements for the award of the Degree of

BACHELOR OF SCIENCE (INFORMATION TECGNOLOGY)

By

Kharat Deepesh Dinkar

Seat No: _____

Under the esteemed guidance of
Mr. Manoj Kumar C. Yadav
Assistant Professor

DEPARTMENT OF DATA SCIENCE

N.G Acharya & D.K. Marathe

College Of Arts, Science & Commerce

(Affiliated to University of Mumbai)

NAAC Accredited "A" Grade

Shri. N. G. Acharya Marg, Chembur, Mumbai-71

MAHARASHTRA

2024-25

PROFORMA FOR THE APPROVAL PROJECT PROPOSAL

PNR NO: -…… Roll No:____28___

1. Name of the Student:

KHARAT DEEPESH DINKAR

2. Title of the Project:

CAR SELLING WEBSITE

3. Name of the Guide:

Mr. Manoj Kumar. C. Yadav

4. Teaching experience of the Guide:

5. Is this your first submission? Yes No

Signature of the Student Signature of the Guide Date

Date: - _____ Date: -_____

Signature of the Coordinator

Date: -_____

N.G. Acharya & D.K. Marathe College
Of Arts, Science & Commerce
(Affiliated to University of Mumbai)
NAAC Accredited "A" Grade
Shri. N. G. Acharya Marg, Chembur, Mumbai-71
MAHARASHTRA
2024-25

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the project entitled, **"CAR SELLING WEBSITE"**, is bonafide work of **KHARAT DEEPESH DINKAR** bearing Seat No: _____ submitted in partial fulfilment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY from University of Mumbai.

Internal Guide **Coordinator**

External Examiner

Date: **College Seal**

Acknowledgement

Acknowledgement for Car Selling Website Project

We would like to express our sincere gratitude to everyone who contributed to the successful development of our Car Selling Website project.

First and foremost, we thank our project supervisor for their invaluable guidance and support. Their insights and expertise greatly aided us in navigating the complexities of web development.

We extend our appreciation to our team members for their collaboration, dedication, and creative ideas that shaped the project. The synergy within our group was key to overcoming challenges and achieving our goals.

Special thanks to our peers for providing constructive feedback and encouragement throughout the process. We also acknowledge the resources and tools that facilitated our work, enhancing the project's quality.

Finally, we are grateful to our families for their unwavering support and motivation, which inspired us to bring our vision to life.

This project is a testament to teamwork, innovation, and the pursuit of excellence in creating a user-friendly platform for car selling.

DECLARATION

I hereby declare that the project entitled, **"Car Selling Website"** done at **N.G. Acharya & D.K. Marathe College of Arts, Science & Commerce**, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfilment of the requirements for the award of degree of BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY) to be submitted as final semester project as part of our curriculum.

Deepesh Dinkar Kharat

ABSTRACT

The **Car Seller Website** is an innovative online platform designed to streamline the process of buying and selling vehicles. Targeting both individual sellers and potential buyers, the website offers a user-friendly interface that simplifies vehicle listings, search functionalities, and transaction processes.

By bridging the gap between buyers and sellers, the Car Seller Project Website aims to create a seamless, efficient, and enjoyable automotive marketplace. Whether you're looking to sell your car or find your next vehicle, our platform is designed to meet your needs with ease and convenience

.

The Car Seller Project Website is a cutting-edge digital platform designed to facilitate the buying and selling of vehicles, catering to both individual sellers and prospective buyers. By integrating advanced features and a user-centric approach, the website aims to transform the traditional automotive marketplace into a streamlined, efficient, and enjoyable experience.

TABLE OF CONTENTS

INTRODUCTION

INTRODUCTION

Welcome to the Car Seller Project Website, a comprehensive online platform designed to revolutionize the way individuals buy and sell vehicles. In an ever-evolving automotive market, we recognize the challenges that both sellers and buyers face. Our mission is to create a seamless, efficient, and user-friendly experience that addresses these challenges head-on.

- **Overview:**

The Car Seller Project Website serves as a one-stop marketplace where users can list their cars for sale or browse a diverse inventory of vehicles. With features tailored to enhance user engagement and streamline transactions, our platform prioritizes accessibility, security, and community trust.

- **Key Objectives:**

1. **Empower Sellers:** We provide an easy-to-use interface that allows sellers to create detailed listings, upload images, and communicate directly with potential buyers, maximizing their chances of a successful sale.
2. **Simplify Buying:** For buyers, we offer advanced search and filtering options, enabling them to find the perfect vehicle that fits their preferences and budget without hassle.
3. **Enhance Security:** Our platform incorporates secure payment options and communication tools, ensuring that transactions are safe and reliable for all users.
4. **Foster Community Trust:** Through a robust review and rating system, we aim to build a community of trust where users can make informed decisions based on the experiences of others.
5. **Educational Resources:** We provide a wealth of information, including buying guides, maintenance tips, and industry news, empowering users to make knowledgeable choices throughout their automotive journey.

1.1 Background

The process of buying and selling vehicles has traditionally been conducted through classified advertisements, dealership visits, and word-of-mouth recommendations. However, these methods are often time-consuming, inefficient, and lack transparency.

With the advancement of digital platforms, online marketplaces for vehicles have gained popularity. Websites like **CarTrade, OLX Autos, and Cars24** have transformed how people buy and sell cars. These platforms offer user-friendly interfaces, advanced search filters, and easy communication between buyers and sellers.

The **Car Seller Project Website** is designed to further improve this experience by integrating modern web technologies. It allows individual sellers and dealerships to list their cars effortlessly while providing buyers with an intuitive search mechanism to find the perfect vehicle based on their needs.

1.2 Objectives

The main objective of this project is to develop a **feature-rich** and **user-friendly** platform for buying and selling vehicles. The key objectives are:

- **Simplified Vehicle Listing** – Sellers can easily upload details of their vehicles, including specifications, images, pricing, and contact information.
- **Advanced Search & Filters** – Buyers can search for cars based on brand, model, price, location, fuel type, and other parameters to find the best match.
- **User Authentication & Profile Management** – Secure login and profile management for buyers and sellers to track listings and inquiries.
- **Efficient Communication** – Integrated messaging or contact options to enable buyers to connect with sellers.
- **Responsive & Scalable Design** – The website will be mobile-friendly and scalable to handle a large number of listings and users.
- **Secure Data Management** – The system will ensure user data privacy and prevent fraudulent activity.

These objectives ensure that the project delivers a **seamless, reliable, and efficient** experience for all users.

1.3 Purpose, Scope, and Applicability

1.3.1 Purpose

The primary purpose of this project is to create a **dedicated online marketplace** for car buyers and sellers, offering a seamless and structured process. It addresses issues such as:

- The **lack of transparency** in offline transactions.
- **Time-consuming** traditional methods of buying and selling.
- The **need for a centralized** platform to compare and list vehicles.

This project enhances the current system by integrating features like a **user-friendly interface, powerful search options, and secure authentication mechanisms**.

1.3.2 Scope

The scope of this project includes:

- **Website Development** – The platform will be built using **Laravel (PHP framework) for backend** and **MySQL for database management**.
- **User Roles** – Two main user types: **sellers** (who list vehicles) and **buyers** (who search for vehicles).
- **Vehicle Listings** – Users can upload car details, including model, price, location, and images.
- **Search & Filters** – Advanced filters to refine search results based on different criteria.
- **User Authentication** – Secure login and account management for users.

Limitations:

- The project will not include **a payment gateway** or **auction-based selling** in its initial phase.
- **Vehicle verification** and **ownership transfer processes** will need to be handled offline.

1.3.3 Applicability

This project has a broad application and serves multiple stakeholders:

- **For Individual Sellers:** Enables quick and easy listing of their cars without relying on dealerships.
- **For Car Dealerships:** Provides a digital platform to showcase their inventory to a larger audience.
- **For Buyers:** Offers a structured way to explore cars with advanced filters and reliable seller information.

Additionally, this platform can be extended for use in **rental vehicle businesses** and **fleet management companies** in the future.

1.4 Achievements

Upon successful completion of this project, the following achievements will be realized:

- **Technical Skills Gained:**
  - Development of a fully functional website using **Laravel and MySQL**.
  - Implementation of **responsive UI/UX design** for a smooth user experience.
  - Integration of **secure authentication** and **database management** techniques.

- **Project Contributions:**
  - A structured and efficient **car-selling marketplace** that simplifies the vehicle trading process.
  - A digital solution that reduces **dependency on offline transactions**.
  - A practical demonstration of **real-world web development concepts**.

This project will serve as a **strong foundation** for further enhancements, such as integrating **online payments, AI-based price predictions, and automated verification systems**.

SURVEY OF TECHNOLOGY

SURVEY OF TECHNOLOGY

The **Car Seller Project Website** is built using a combination of frontend, backend, and database technologies to ensure smooth functionality, security, and scalability. The technologies used include HTML, CSS, JavaScript (for the frontend), PHP with Laravel (for the backend), and MySQL (for database management). Each technology plays a crucial role in the development of the website.

2.1 Frontend Technologies

1. HTML (HyperText Markup Language)

HTML is used to structure web pages by defining elements such as headings, paragraphs, forms, buttons, and images. It provides the basic framework on which the website is built.

2. CSS (Cascading Style Sheets)

CSS is responsible for styling and designing the website. It controls colours, fonts, layouts, and responsiveness, ensuring that the website is visually appealing and works well on different devices. CSS helps create a modern and user-friendly interface.

3. JavaScript

JavaScript adds interactivity and dynamic behaviour to the website. It enables features such as real-time search filters, form validation, and content updates without reloading the page. This improves user experience and makes the website more engaging.

2.2 Backend Technologies

1. PHP (Hypertext Preprocessor)

PHP is a server-side scripting language used to handle requests from the frontend, process data, and generate dynamic content. It manages tasks such as user authentication, form processing, and data retrieval from the database. PHP ensures smooth communication between the frontend and backend.

2. Laravel (PHP Framework)

Laravel is a powerful PHP framework that follows the Model-View-Controller (MVC) architecture. It simplifies backend development by providing built-in tools for:

- User authentication (login, registration).
- Routing (handling different web pages and requests).
- Database migrations and queries (interacting with MySQL easily).
- Security features (protection against SQL injections and CSRF attacks).

Laravel ensures structured, secure, and efficient backend development.

2.3 Database Technology

1. MySQL

MySQL is a relational database management system (RDBMS) used to store and manage structured data such as:

- User accounts and login details.
- Vehicle listings (make, model, price, images, etc.).
- Transaction and inquiry records.

MySQL is fast, secure, and scalable, making it suitable for handling large amounts of data. Laravel integrates seamlessly with MySQL, ensuring smooth database operations.

REQUIRMENT AND ANALYSIS

REQUIRMENT AND ANALYSIS

3.1 Problem Definition: -

The Car Seller Project Website aims to address the inefficiencies of traditional car-buying and selling methods. Currently, many individuals and dealerships rely on classified ads, word-of-mouth, or third-party agents, which can be time consuming, lack transparency, and lead to potential fraud.

Identified Problems in the Existing System:

- **Lack of a centralized platform** – Buyers and sellers struggle to find a reliable place to list and purchase cars.
- **Limited search and filter options** – Users face difficulty in finding vehicles that match their specific requirements.
- **Lack of direct communication** – Sellers and buyers often rely on intermediaries, leading to additional costs.
- **Security concerns** – Many existing systems lack proper user verification, increasing the risk of fraud.

Sub-Problems:

- **Vehicle Listing Issues:** Users need an efficient way to list cars with images, descriptions, and pricing.
- **Search & Filtering Challenges:** The system must allow users to search for vehicles based on model, price, fuel type, and location.
- **User Authentication & Verification:** To prevent fraud, the system should verify user details before allowing transactions.
- **Scalability & Performance:** The platform must handle multiple listings without performance issues.

3.2Requirements Specification Functional Requirements:

 User Registration & Login:

- Users can create an account and log in securely.
- Authentication via email verification.

 Vehicle Listing Management:

- Users can upload car details (model, price, images, features).
- Listings should be editable and removable.

 Search and Filter System:

- Users can search based on price range, model, brand, fuel type, etc.

 User-to-User Communication:

- Buyers can contact sellers via email or in-system messaging.

 Admin Panel:

- Admins can manage user accounts and monitor listings for fraud.

 Responsive Design:

- The website must be accessible on mobile and desktop devices.

Non-Functional Requirements:

 Security:

- Secure authentication and data protection measures.

 Performance:

- The website should load quickly and handle multiple requests efficiently.

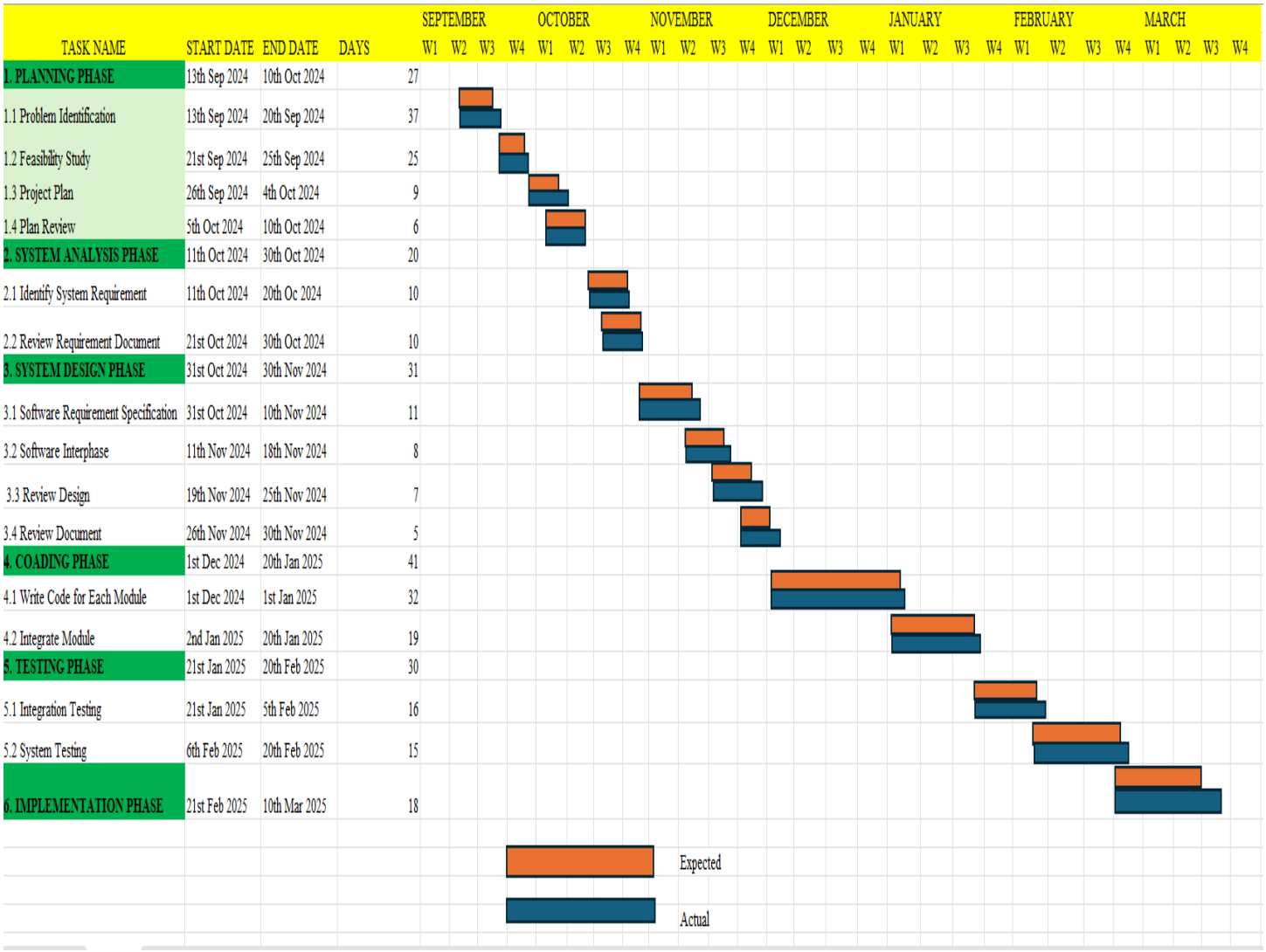
 Scalability:

- The system should support an increasing number of users and listings.

3.3 Planning and Scheduling Project Planning:

The development is divided into several stages, including requirement gathering, UI/UX design, development, testing, and deployment. Agile methodology will be used for continuous feedback and improvements.

Project Timeline (Gantt Chart Representation



TASK NAME	START DATE	END DATE	DAYS
1. PLANNING PHASE	13th Sep 2024	10th Oct 2024	27
1.1 Problem Identification	13th Sep 2024	20th Sep 2024	37
1.2 Feasibility Study	21st Sep 2024	25th Sep 2024	25
1.3 Project Plan	26th Sep 2024	4th Oct 2024	9
1.4 Plan Review	5th Oct 2024	10th Oct 2024	6
2. SYSTEM ANALYSIS PHASE	11th Oct 2024	30th Oct 2024	20
2.1 Identify System Requirement	11th Oct 2024	20th Oc 2024	10
2.2 Review Requirement Document	21st Oct 2024	30th Oct 2024	10
3. SYSTEM DESIGN PHASE	31st Oct 2024	30th Nov 2024	31
3.1 Software Requirement Specification	31st Oct 2024	10th Nov 2024	11
3.2 Software Interphase	11th Nov 2024	18th Nov 2024	8
3.3 Review Design	19th Nov 2024	25th Nov 2024	7
3.4 Review Document	26th Nov 2024	30th Nov 2024	5
4. COADING PHASE	1st Dec 2024	20th Jan 2025	41
4.1 Write Code for Each Module	1st Dec 2024	1st Jan 2025	32
4.2 Integrate Module	2nd Jan 2025	20th Jan 2025	19
5. TESTING PHASE	21st Jan 2025	20th Feb 2025	30
5.1 Integration Testing	21st Jan 2025	5th Feb 2025	16
5.2 System Testing	6th Feb 2025	20th Feb 2025	15
6. IMPLEMENTATION PHASE	21st Feb 2025	10th Mar 2025	18

3.4 Software and Hardware Requirements

3.4.1 Hardware Requirements:

- **Processor:** Intel i5 or higher
- **RAM:** Minimum 8GB
- **Storage:** Minimum 100GB HDD/SSD
- **Graphics Card:** Integrated graphics or higher
- **Internet Connection:** Required for hosting and database access

3.4.2 Software Requirements:

- **Operating System:** Windows/Linux/Mac
- **Development Tools:** Visual Studio Code, Sublime Text, PHPStorm
- **Backend:** PHP (Laravel Framework)
- **Frontend:** HTML, CSS, JavaScript
- **Database**: MySQL
- **Web Server:** Apache (XAMPP/LAMP)
- **Testing Tools:** Postman (for API testing), Browser Developer Tools

3.5 Preliminary Product Description

The Car Seller Project Website provides a platform where users can:

- Register and manage their profiles.
- List cars for sale with detailed specifications.
- Search for vehicles using advanced filters.
- Contact sellers directly for inquiries.
- Use a secure and scalable system for smooth transactions.

The system ensures ease of use, security, and efficiency, making the car buying and selling process seamless.

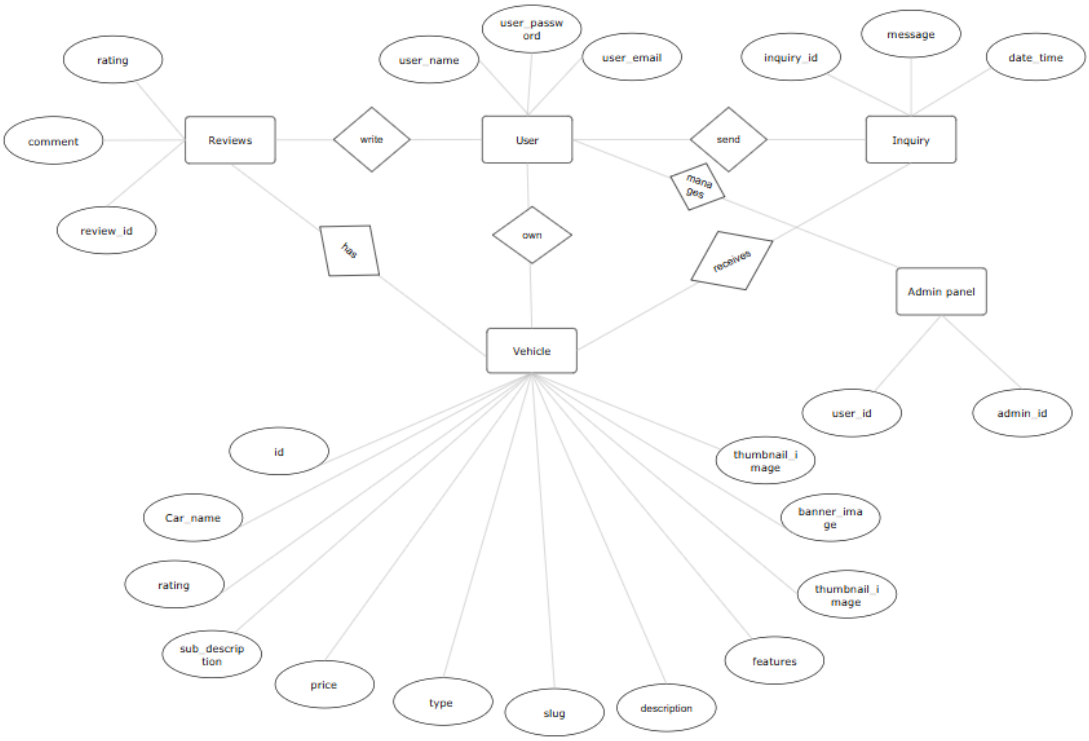
3.6 Conceptual Models

3.6.1 ER Diagram: -

This ER Diagram structures the Car Seller Project Website to handle:

1.User registrations, vehicle listings, inquiries, reviews, and admin monitoring. 2. Data relationships between users, cars, inquiries, and reviews.

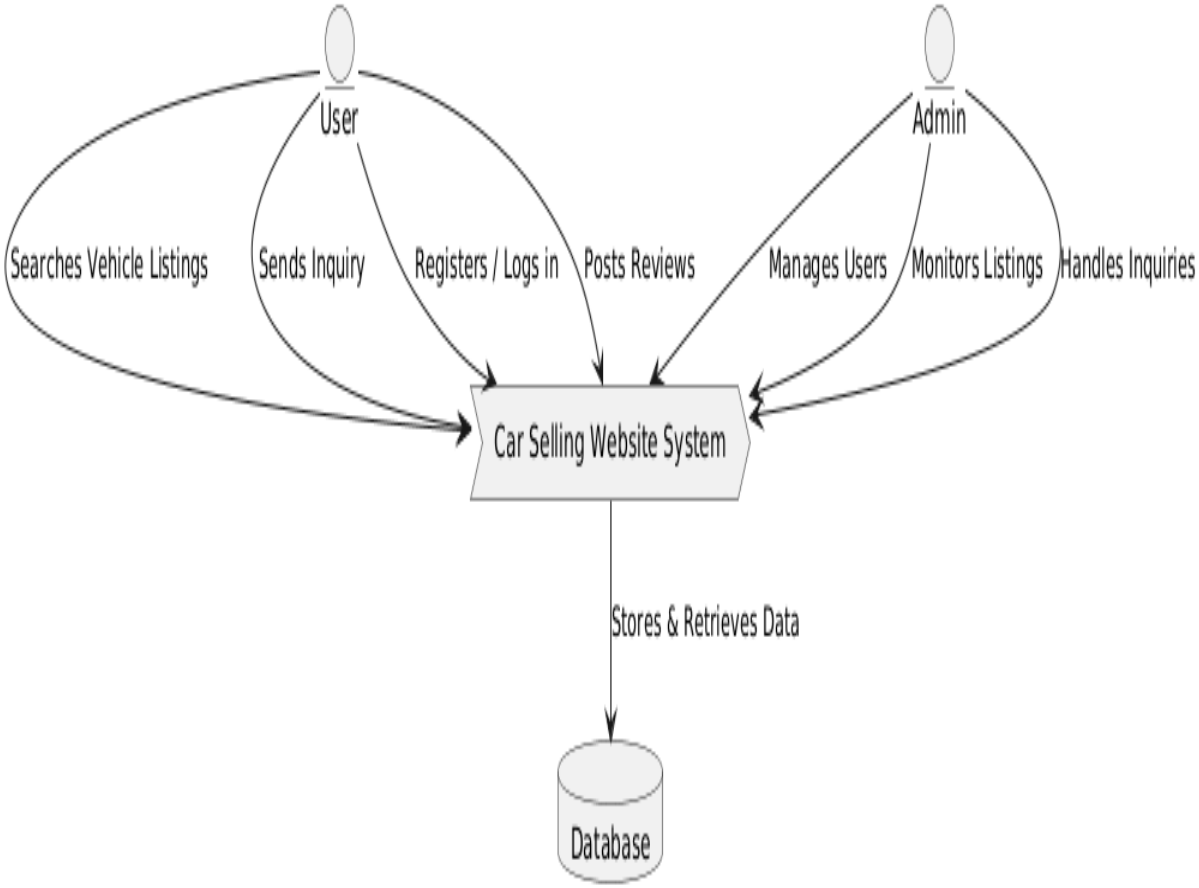
3. A scalable and efficient database design for managing the platform.



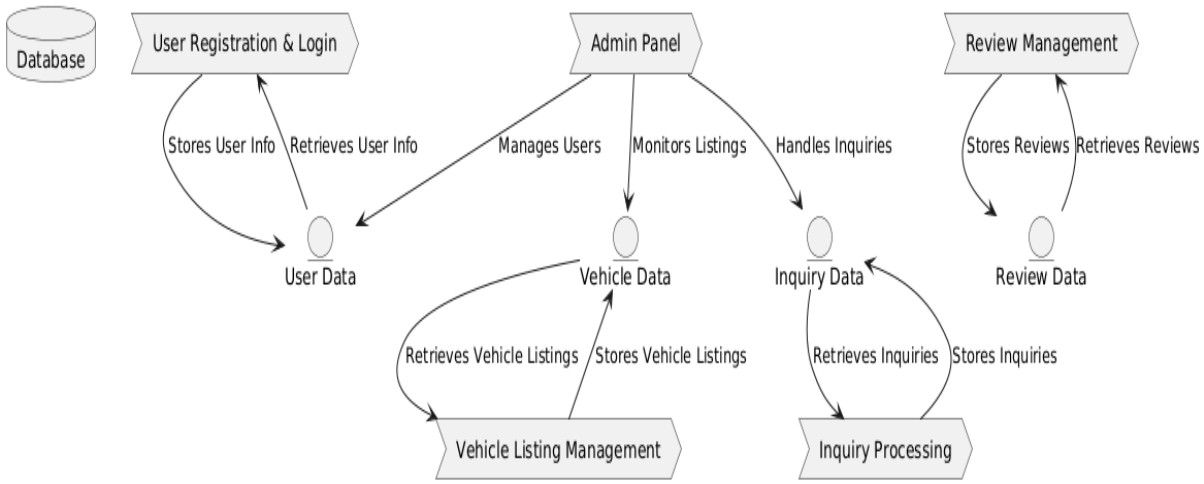
3.6.2 Data Flow Diagram: -

- The User interacts with the Car Seller Website.
- The Website communicates with the Database to store and fetch vehicle listings, inquiries, and user data.
- The Admin Panel manages users, listings, and system activities.

Level 0:-


User
Admin
Searches Vehicle Listings
Sends Inquiry
Registers / Logs in
Posts Reviews
Manages Users
Monitors Listings
Handles Inquiries
Car Selling Website System
Stores & Retrieves Data
Database

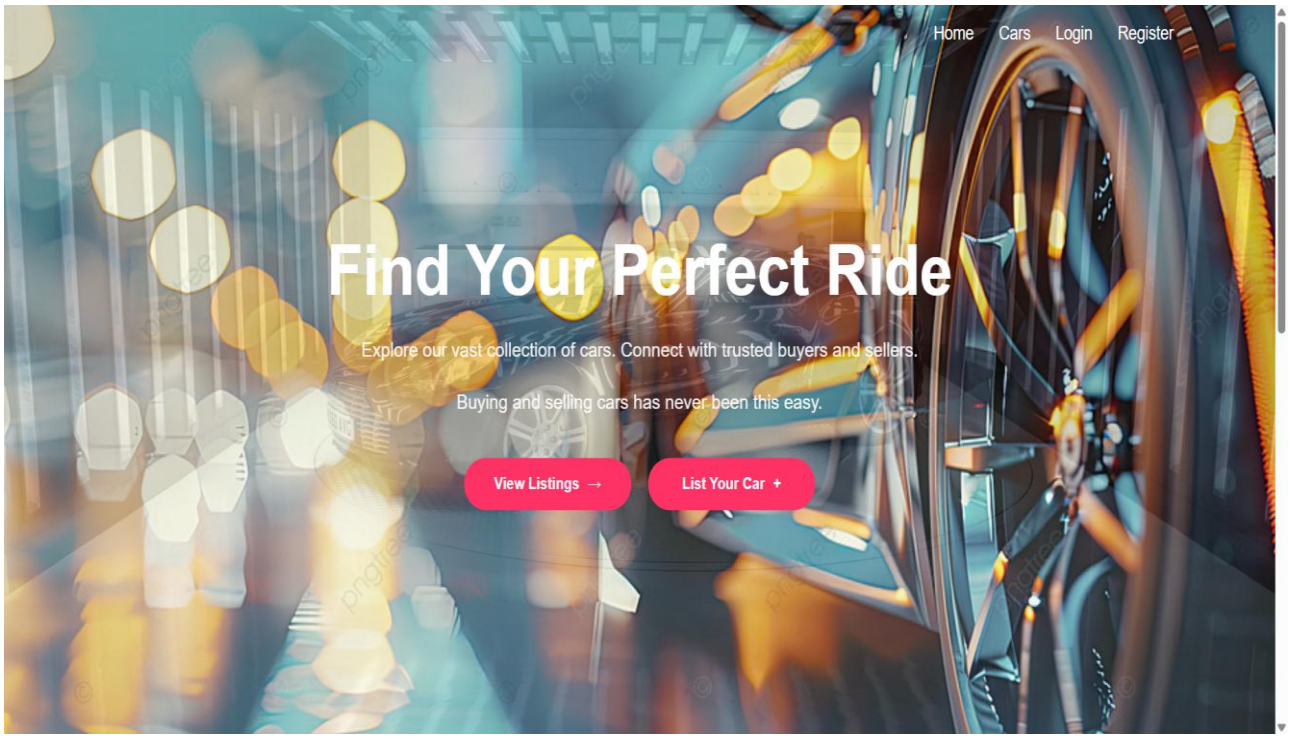
Level 1

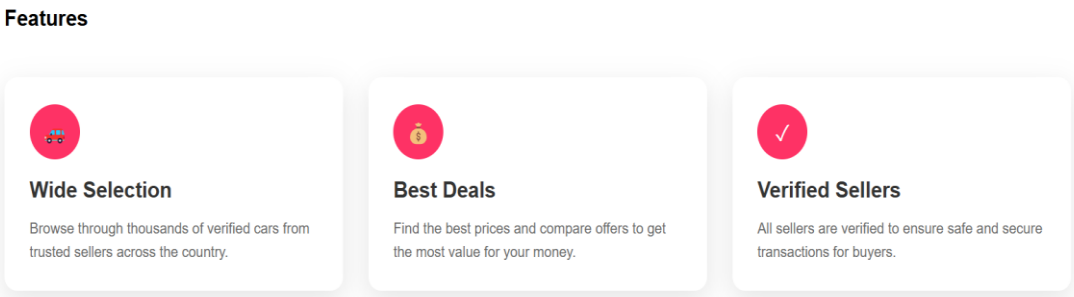

Database
User Registration & Login
Admin Panel
Review Management
Stores User Info
Retrieves User Info
Manages Users
Monitors Listings
Handles Inquiries
Stores Reviews
Retrieves Reviews
User Data
Vehicle Data
Inquiry Data
Review Data
Retrieves Vehicle Listings
Stores Vehicle Listings
Retrieves Inquiries
Stores Inquiries
Vehicle Listing Management
Inquiry Processing

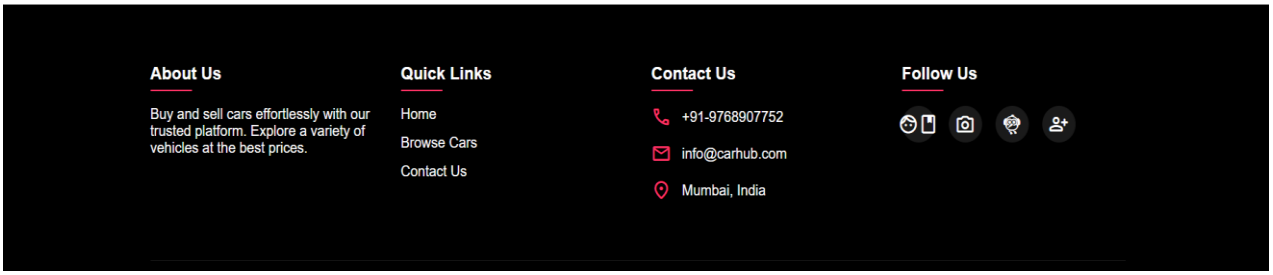
SYSTEM DESIGN

BASIC MODELS

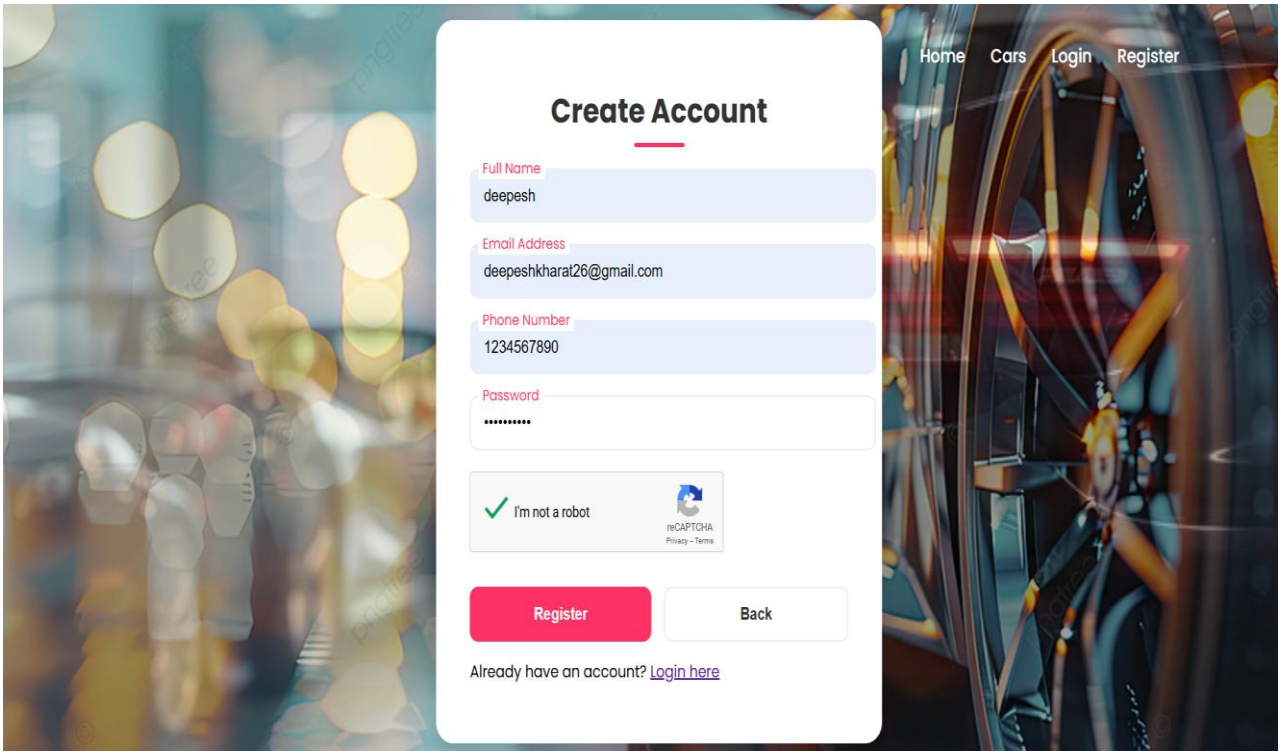
4.1.1 Home Page: -



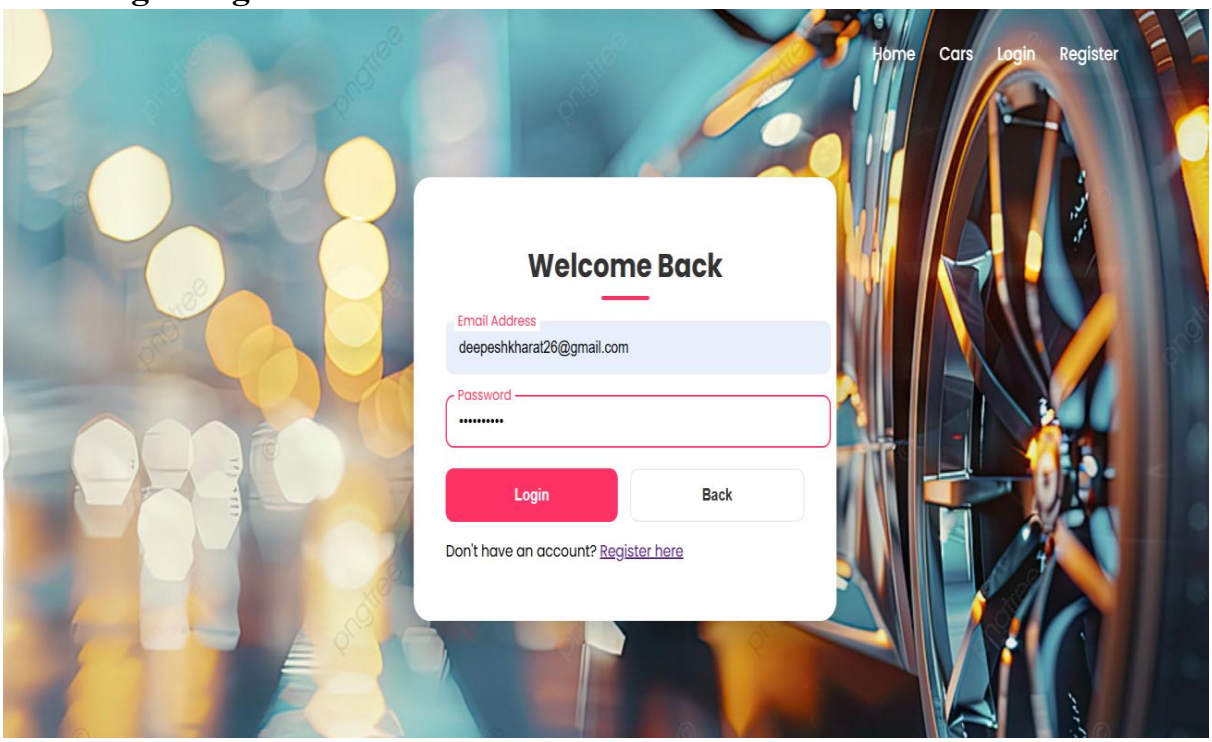




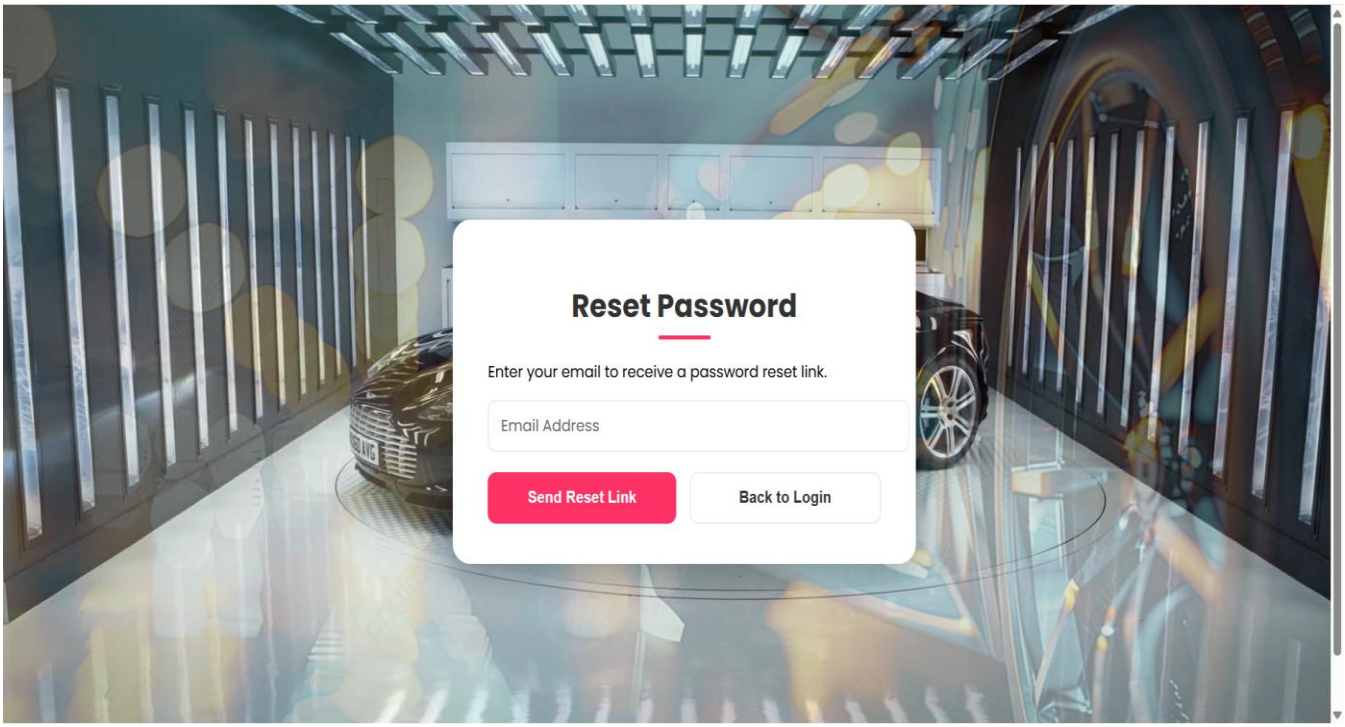
4.1.2 Register Page: -



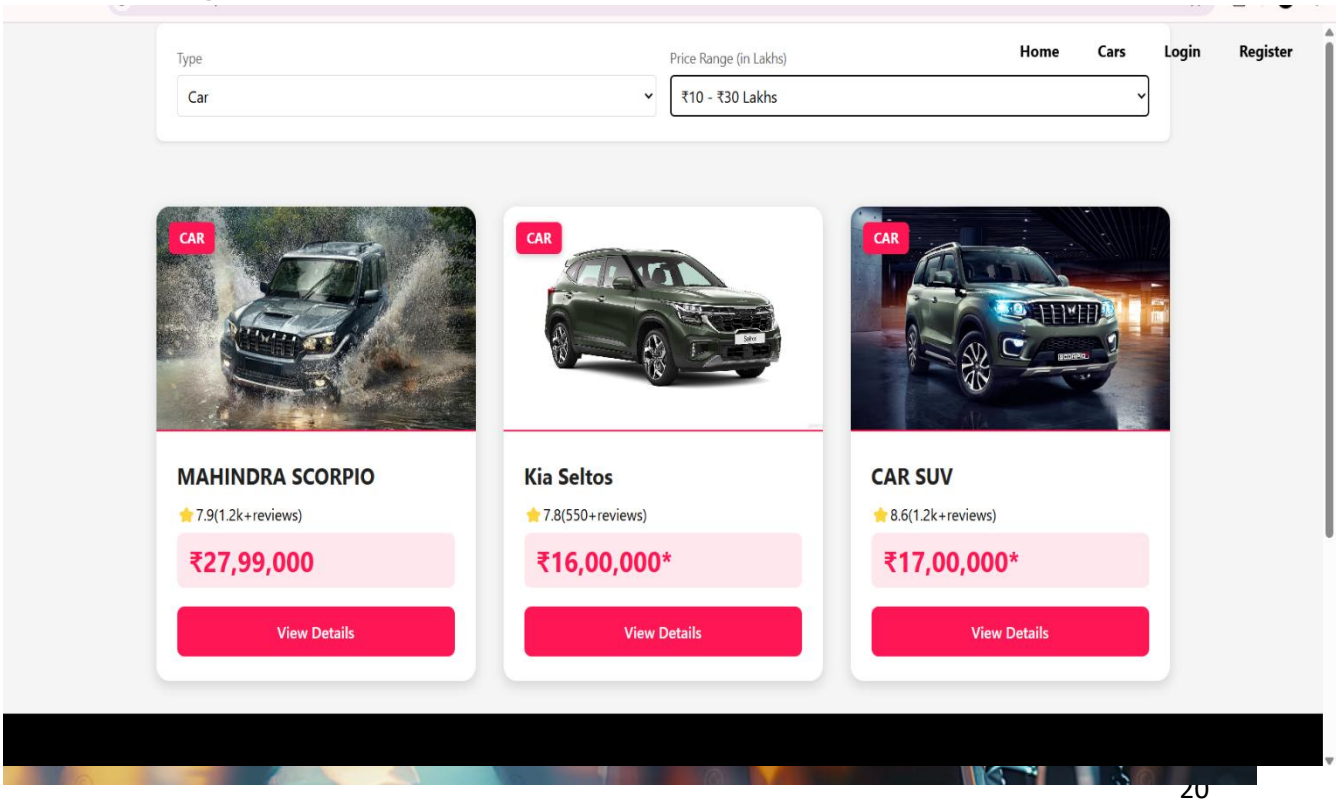
4.1.3 Login Page: -



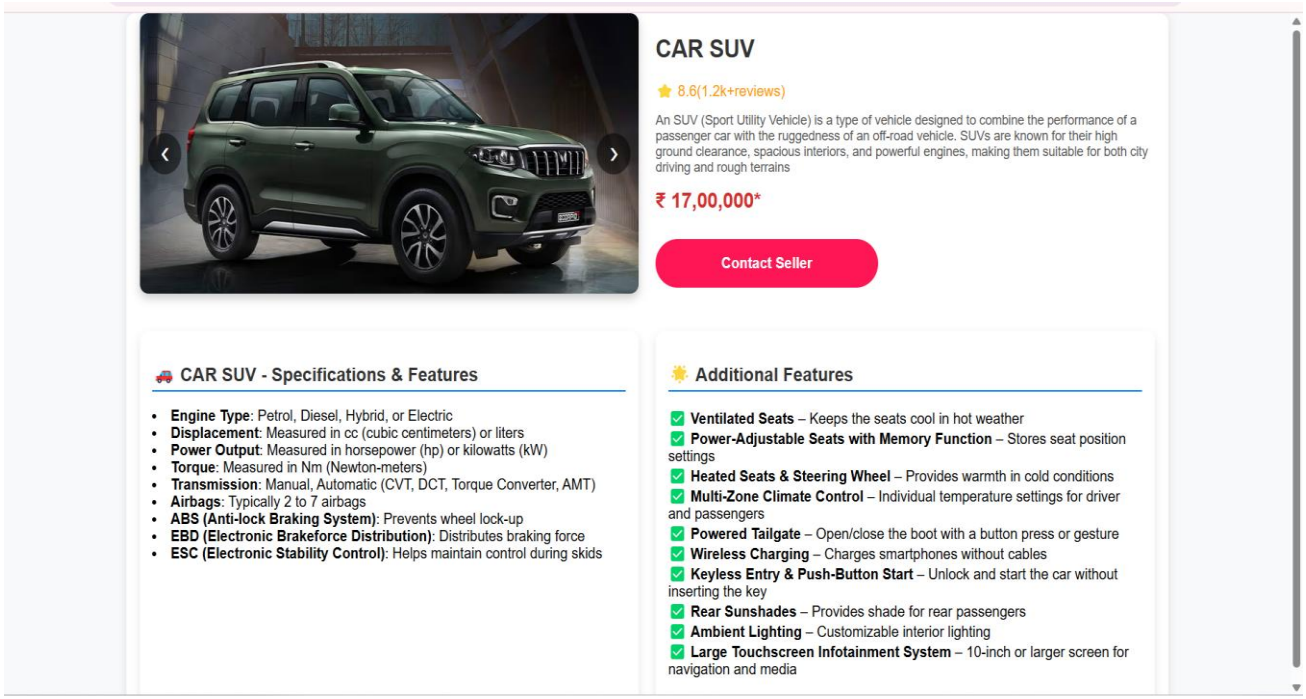
4.1.4 Forgot Password Page: -



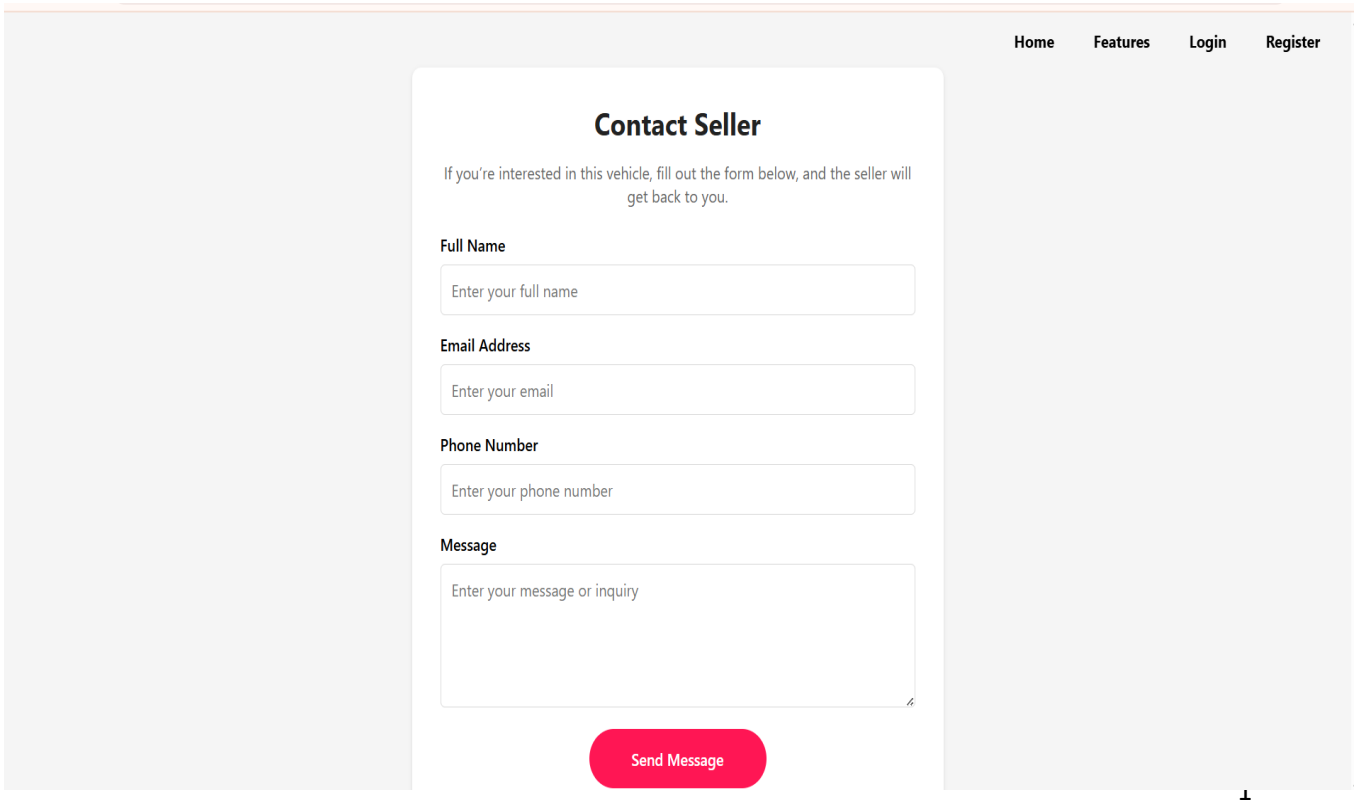
4.1.5 Car Page: -



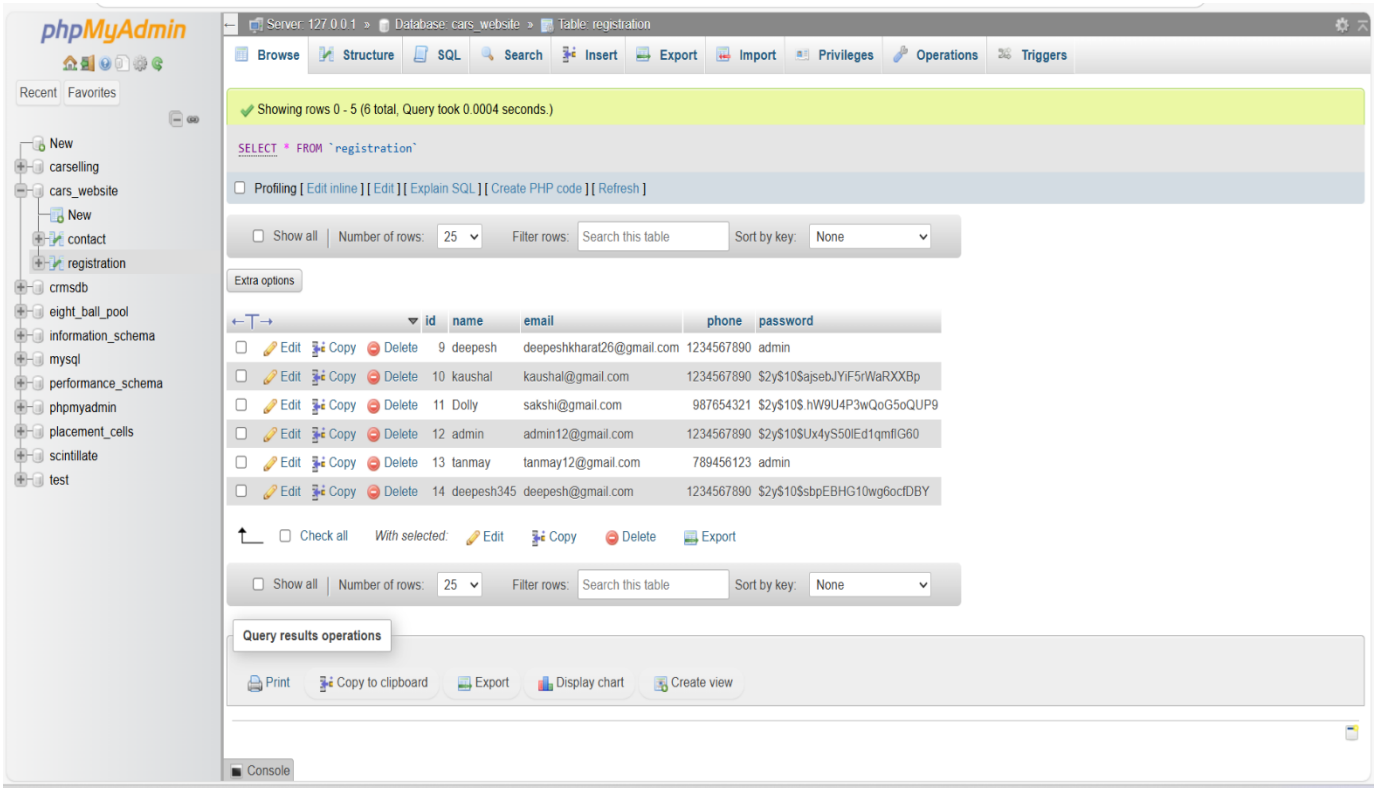
4.1.6 Car details Page: -



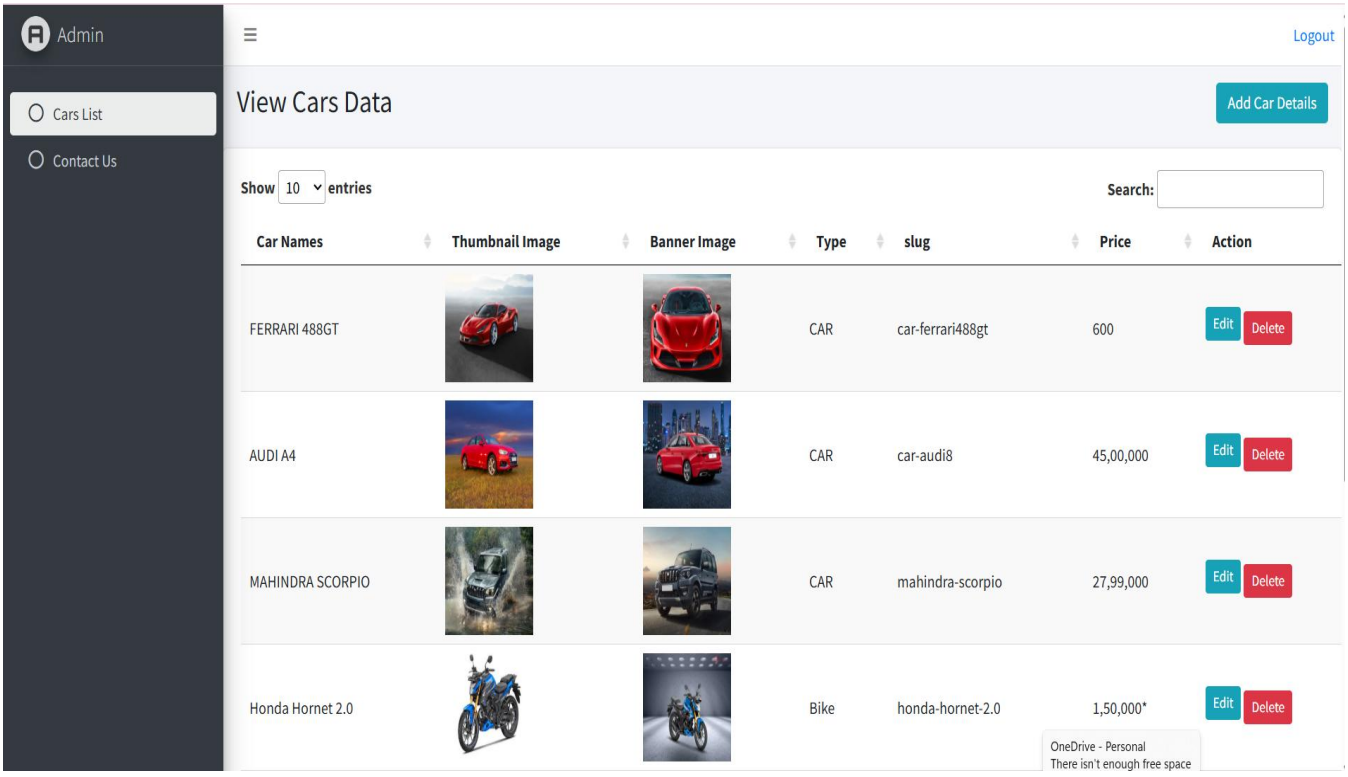
4.1.7 Contact Page: -



4.1.8 MySQL Database: -



4.1.9 Admin Panel: -



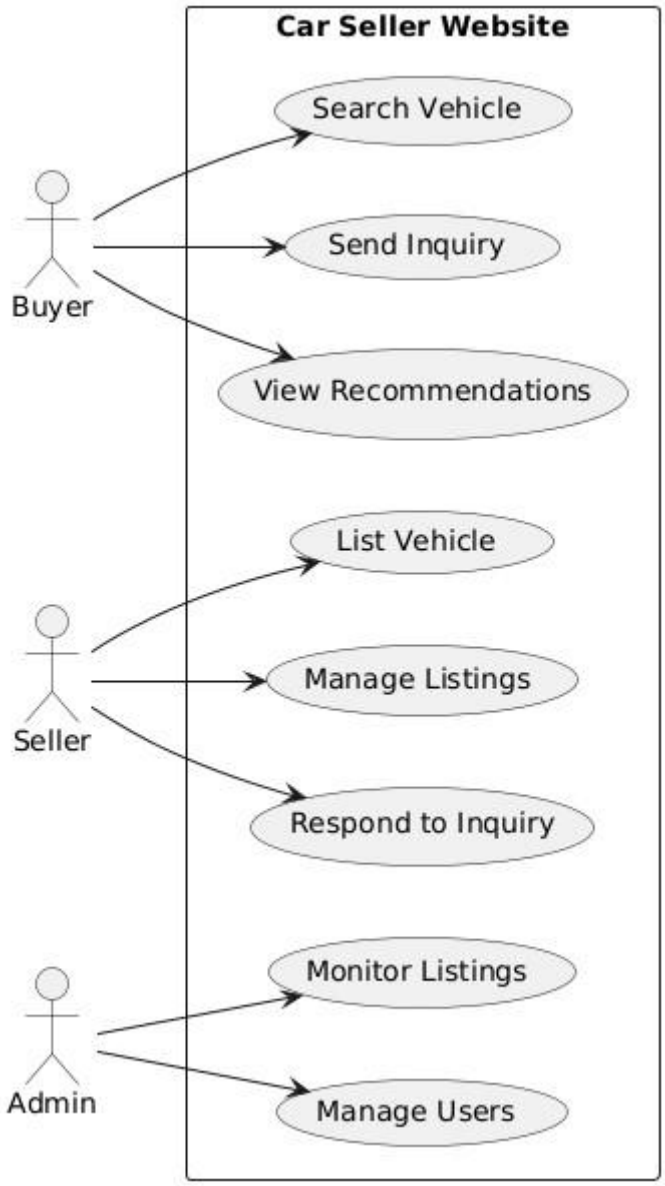
4.2 Data Design

This section describes the data architecture, including schema design and integrity constraints.

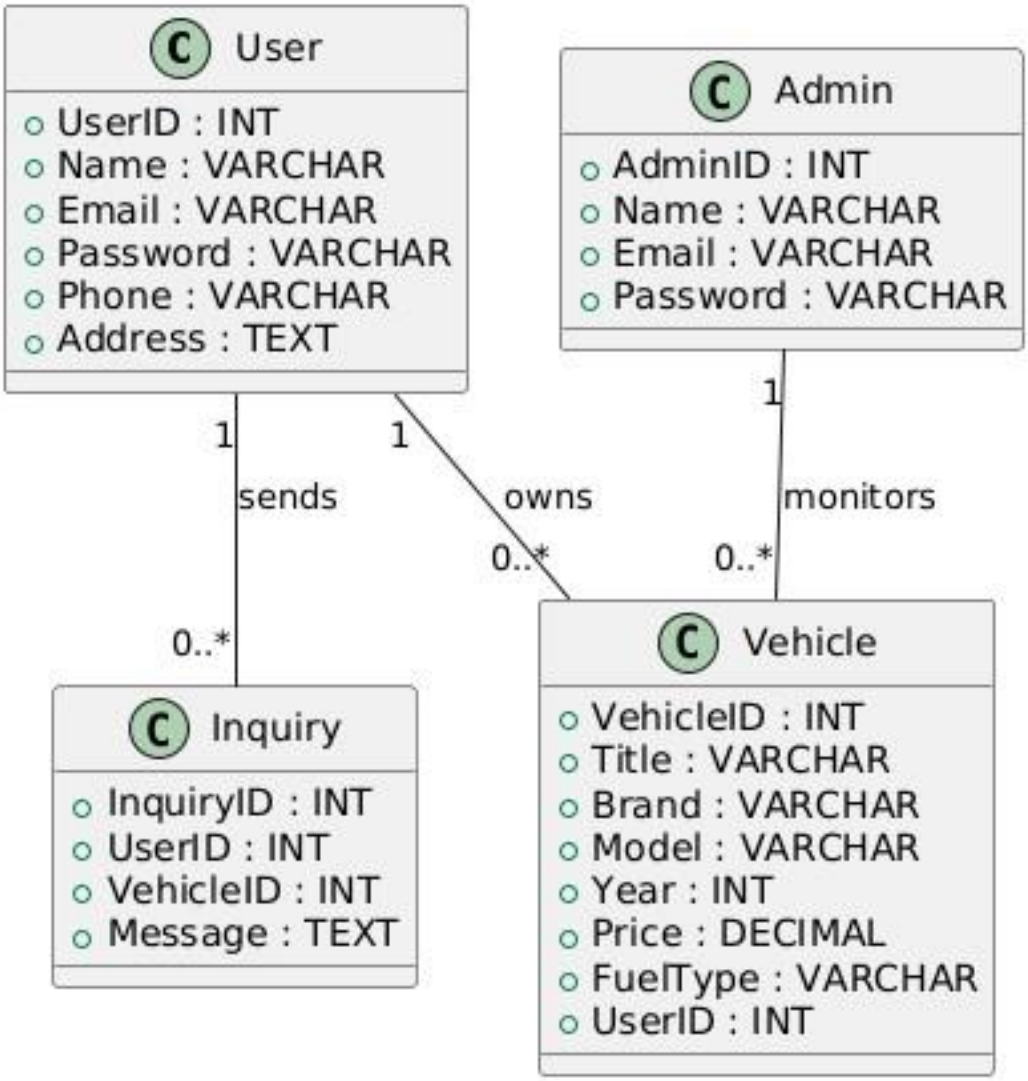
4.2.1 Schema Design:

- Description: Outline of the database schema for the Car Seller Website, detailing tables (collections) and attributes (documents).
- Tables and Attributes:
  - Users Table: Stores user profiles with relevant details.
  - Attributes: UserID (PK), Name, Email (Unique), Password, Phone, Address.
  - Vehicles Table: Stores car listings created by users.
  - Attributes: VehicleID (PK), Title, Brand, Model, Year, Price, FuelType, Transmission, Mileage, Description, Image, UserID (FK).
  - Inquiries Table: Stores messages from buyers to sellers.
  - Attributes: InquiryID (PK), UserID (FK), VehicleID (FK), Message, InquiryDate.
  - Reviews Table: Holds ratings and reviews for vehicles.
  - Attributes: ReviewID (PK), UserID (FK), VehicleID (FK), Rating, Comment, ReviewDate.
  - Admins Table: Stores admin details for monitoring and managing the platform.
  - Attributes: AdminID (PK), Name, Email (Unique), Password.

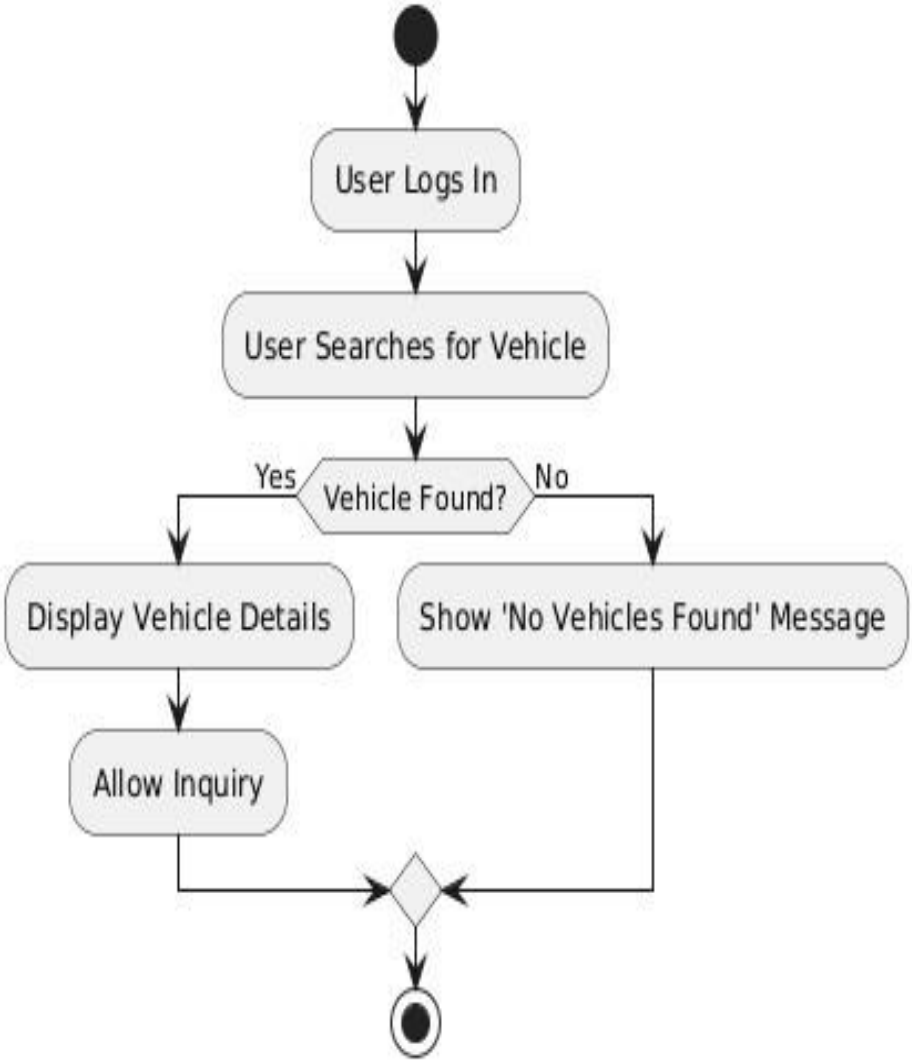
4.2.1.1 Use Case Diagram



4.2.1.2 Class Diagram

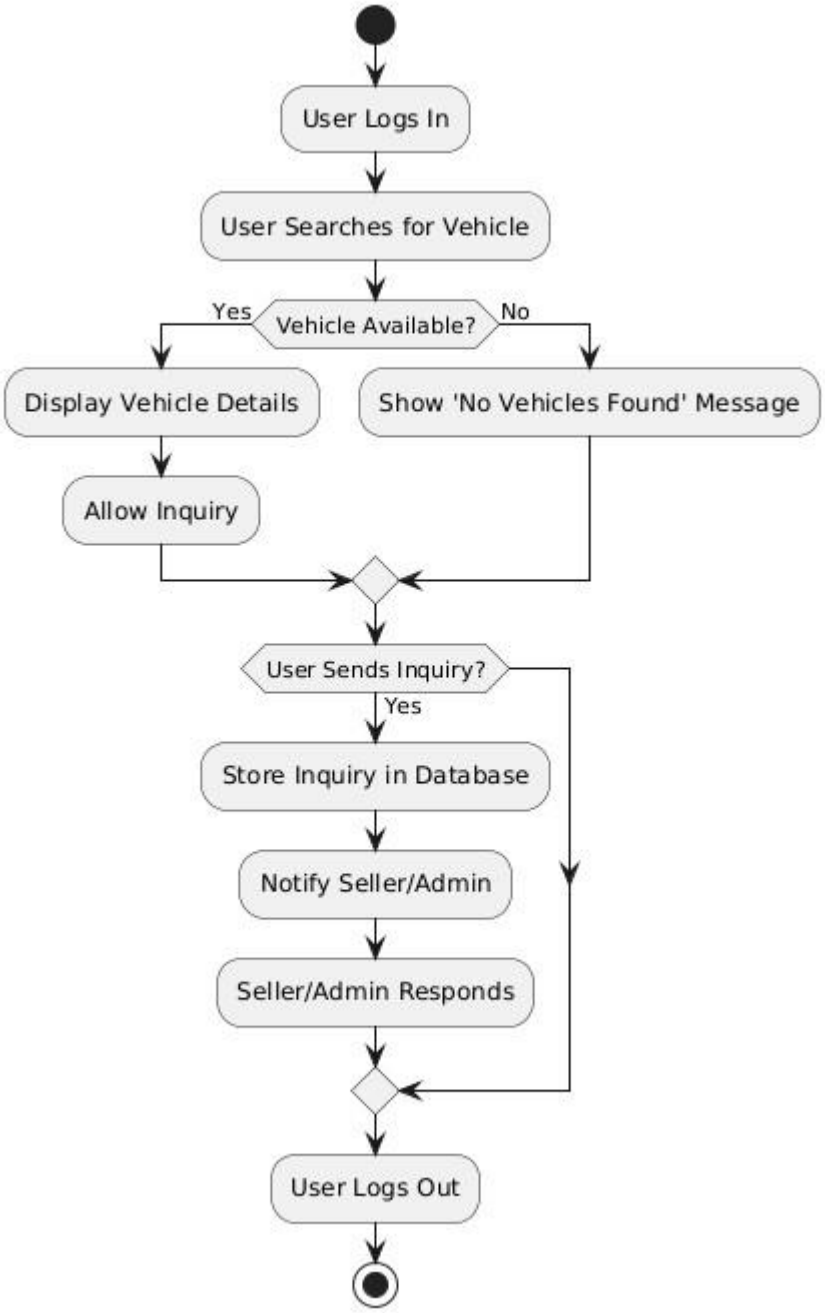


4.2.1.3 Activity Diagram

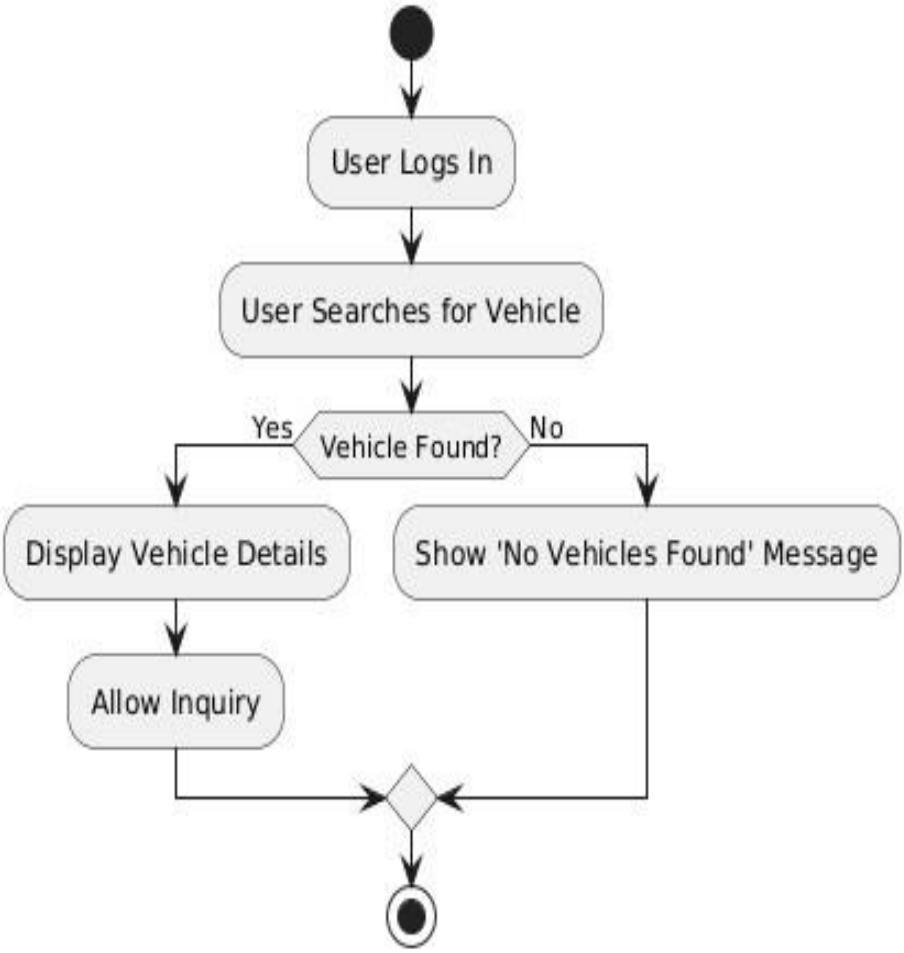


4.3 Logic Design

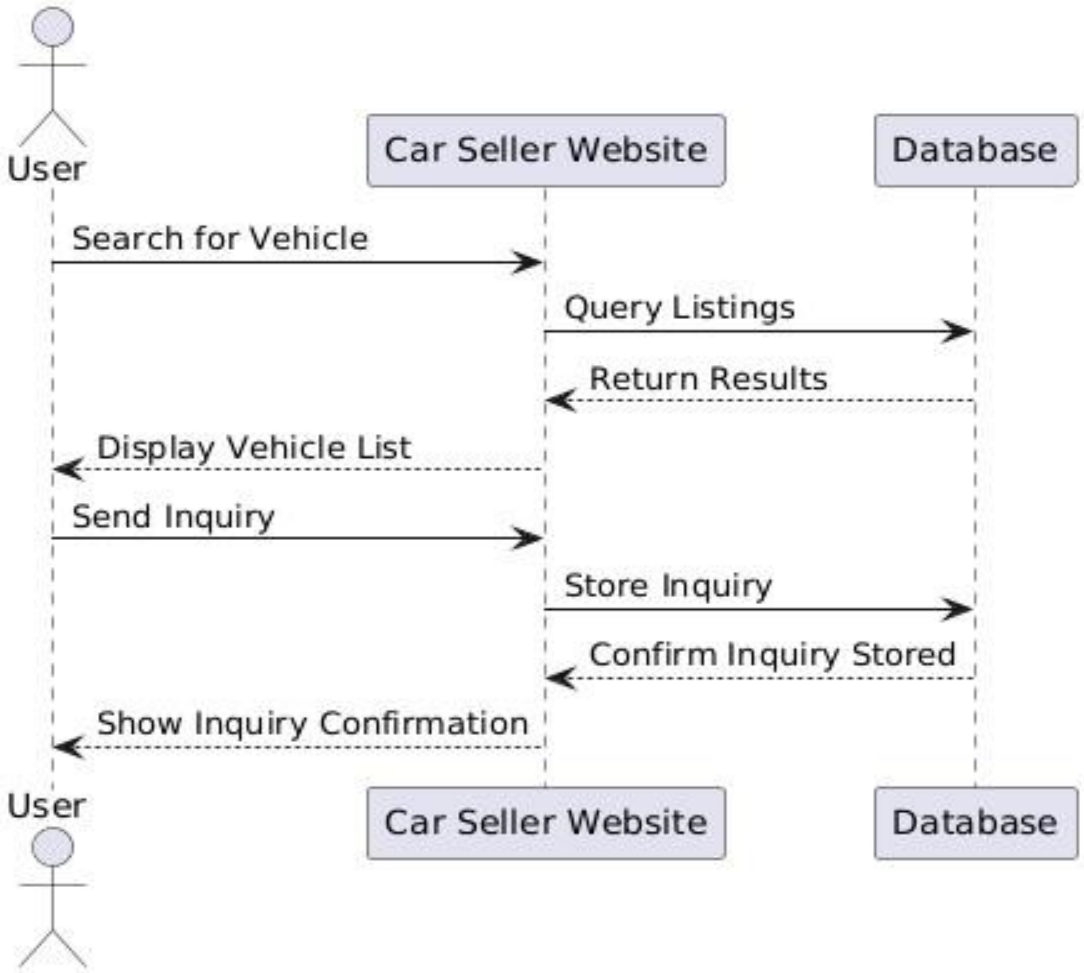
4.3.1 Logic Diagram



4.3.2 Flowchart Diagram



4.3.3 Sequence Diagram



4.4 Test Case Design

Test Case ID	Test Scenario	Test Steps	Expected Result	Successful/Unsuccessful
TC001	User Login	1.Enter valid email and password 2.Click "Login"	User is redirected to the dashboard	
TC002	Invalid User Login	1.Enter incorrect email/password 2. Click "Login"	"Invalid credentials" message is displayed	Pass
TC003	Search for Vehicle	1. Enter search criteria (brand, price) 2. Click "Search"	Relevant vehicle listings are displayed	Pass
TC004	No Vehicle Found	1. Click on a vehicle from the search results	Vehicle details page opens with specifications	Pass
TC005	View Vehicle Details	1. Click on a vehicle from the search results	Vehicle details page opens with specifications	Pass
TC006	Send Inquiry	1. Click "Contact Seller" 2. Enter message 3. Click "Send"	Inquiry is sent successfully	Pass

TC007	List a Vehicle (Seller)	1. Click "List Your Car" 2. Fill details 3. Submit	Vehicle is added to listings	Pass
TC008	Edit Vehicle Listing	1. Seller logs in 2. Go to "My Listings" 3. Click "Edit"	Listing updates successfully	Pass
TC009	Delete Vehicle Listing	1. Seller logs in 2. Go to "My Listings" 3. Click "Delete"	Vehicle listing is removed	Pass
TC010	Approve Listing (Admin)	1. Admin logs in 2. View pending listings 3. Click "Approve"	Vehicle listing is marked as active	Pass
TC011	Remove Listing (Admin)	1. Admin logs in 2. Click "Remove" on a listing	Vehicle is removed from the platform	Pass
TC012	View Recommendations	1. Log in 2. Go to "Recommended Vehicles" 1. Log in 2. Go to "Recommended Vehicles"	Suggested vehicles appear based on user activity	Pass
TC013	Logout	1. Click "Logout" button	User is logged out and	Pass

			redirected to the login page	

IMPLEMENTATION AND TESTING

IMPLEMENTATION AND TESTING

In this chapter, we will detail the implementation of the Car Seller Project Website, covering both frontend and backend development along with testing strategies. The system is built using Laravel (PHP) for the backend, MySQL for database management, and HTML, CSS, and JavaScript for the frontend.

5.1 Implementing Approach

1. **Frontend (HTML, CSS, JavaScript)**
    - The user interface where buyers and sellers can interact with the system.
    - Users can search for vehicles, list vehicles, send inquiries, and manage their accounts.

2. **Backend (Laravel - PHP Framework)**
    - Handles business logic, database management, authentication, and API requests.
    - It processes user interactions, stores listings, and manages user inquiries.

3. **Database (MySQL)**
    - Stores data such as:
    - User details (buyers, sellers, and admins).
    - Vehicle listings (make, model, year, price, images, etc.).
    - Inquiries (messages sent from buyers to sellers).

Implementation Steps

1. **Frontend Implementation (HTML, CSS, JavaScript)**
    - **Develop UI** – Create pages for home, login, registration, vehicle listing, and search.
    - **Vehicle Listing & Search** – Allow sellers to list cars and buyers to search with filters.
    - **Inquiry System** – Enable buyers to send inquiries to sellers.

2. **Backend Implementation (Laravel, MySQL)**
    - **User Authentication** – Implement login, registration, and account management.
    - **Vehicle Management** – Develop CRUD operations for vehicle listings.
    - **Inquiry Handling** – Store inquiries in the database and notify sellers.

3. **Database (MySQL)**
    - Store user details, vehicle listings, and inquiries.
    - Ensure data integrity and relationships between entities.

5.2 Coding Details and Code Efficiency

1 Clean and Readable Code

- Used meaningful variable, class, and function names.
- Added comments to explain complex logic and workflows.
- Maintained consistent code formatting and indentation across files.

2 Code Optimization for Efficiency

- Avoided duplicate code using reusable components and helper functions.
- Implemented optimized search and filtering using indexed queries.
- Used Laravel's Eloquent relationships and eager loading to improve performance.

3 Performance Improvement

- Minimized calculations inside loops and optimized data processing.
- Applied caching to reduce database load for frequently accessed data.
- Limited unnecessary API calls and used pagination for large result sets.

4. Memory Management

- Properly managed file uploads and closed resources after use.
- Chose memory-efficient data structures to handle large datasets.
- Utilized Laravel's automatic garbage collection features.

5. Security and Error Handling

- Validated all user input to prevent SQL injection and XSS attacks.
- Used try-catch blocks for robust error handling.
- Stored sensitive data securely and implemented encryption practices.

5.3 Beta Testing

1. Real-User Testing

- Involved real users in testing core modules like vehicle posting and inquiry system.
- Collected feedback on usability, navigation, and performance.

2.Bug Detection and Fixing

- Identified and fixed issues related to image uploads, mobile responsiveness, and broken filters.
- Logged and resolved backend exceptions and frontend glitches.

3.Performance Evaluation

- Performed load testing under simulated user traffic.

- Measured and optimized page load times and query execution.

4. Usability Testing

- Conducted task-based testing to evaluate ease of use.
- Made UI/UX improvements based on tester feedback.

5.Security and Compatibility Checks

- Verified secure user sessions and input validation.
- Tested system on multiple browsers and mobile devices for compatibility.

Coding

Login.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Login - Premium Auto Marketplace</title>
                                                                                    <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500;600;700&
display=swap" rel="stylesheet">
   <link rel="stylesheet" href="style.css">
</head>
<body>
   <div class="header"></div>
      <nav>
         <ul>
            <li><a href="index.html">Home</a></li>
            <li><a href="http://127.0.0.1:8000/cars">Cars</a></li>
            <li><a href="login.html">Login</a></li>
            <li><a href="register.html">Register</a></li>
         </ul>
      </nav>
   </div>

   <div class="container">
      <div class="form-box">
         <h1>Welcome Back</h1>
         <!-- <form id="loginForm" novalidate>
            <div class="input-field">
               <input type="email" id="email" placeholder=" " required>
               <label for="email">Email Address</label>
               <div class="error-message">Please enter a valid email address</div>
            </div>
            <div class="input-field">
               <input type="password" id="password" placeholder=" " required>
               <label for="password">Password</label>
               <div class="error-message">Password is required</div>
            </div>
            <div class="forgot-password">
               <a href="#">Forgot Password?</a>
            </div>
            <div class="btn-field">
               <button type="submit" class="btn btn-primary">
                  <span>Login</span>
               </button>
```

```
                    <button type="button" onclick="window.location.href='index.html'"
class="btn btn-secondary">
                        <span>Back</span>
                    </button>
                </div>
                <p class="register-link">
                    Don't have an account? <a href="register.html">Register here</a>
                </p>
            </form> -->
            <form id="loginForm" action="login_process.php" method="POST" novalidate>
                <div class="input-field">
                    <input type="email" id="email" name="email" placeholder=" " required>
                    <label for="email">Email Address</label>
                    <div class="error-message" style="display:none;">Please enter a valid email
address</div>
                </div>
                <div class="input-field">
                    <input type="password" id="password" name="password" placeholder=" "
required>
                    <label for="password">Password</label>
                        <div class="error-message" style="display:none;">Password is
required</div>
                </div>
                <!-- <div class="forgot-password">
                    <a href="#">Forgot Password?</a>
                </div> -->
                <div class="btn-field">
                    <button type="submit" class="btn btn-primary">
                        <span>Login</span>
                    </button>
                        <button type="button" onclick="window.location.href='index.html'"
class="btn btn-secondary">
                        <span>Back</span>
                    </button>
                </div>
                <p class="register-link">
                    Don't have an account? <a href="register.html">Register here</a>
                </p>
            </form>
        </div>
    </div>

    <!-- <script>
        const loginForm = document.getElementById('loginForm');
        const emailInput = document.getElementById('email');
```

```
const passwordInput = document.getElementById('password');

function validateEmail(email) {
    return /^[^\s@]+@[^\s@]+\.[^\s@]+$/.test(email);
}

function showError(input, message) {
    const field = input.parentElement;
    field.classList.add('error');
    field.querySelector('.error-message').textContent = message;
}

function clearError(input) {
    input.parentElement.classList.remove('error');
}

loginForm.addEventListener('submit', async (e) => {
    e.preventDefault();
    let isValid = true;

    clearError(emailInput);
    clearError(passwordInput);

    if (!emailInput.value || !validateEmail(emailInput.value)) {
        showError(emailInput, 'Please enter a valid email address');
        isValid = false;
    }

    if (!passwordInput.value) {
        showError(passwordInput, 'Password is required');
        isValid = false;
    }

    if (isValid) {
        const submitButton = loginForm.querySelector('button[type="submit"]');
        const buttonText = submitButton.querySelector('span');

        submitButton.classList.add('loading');
        buttonText.textContent = 'Loading...';
        try {
            const formData = new FormData(loginForm);
            const response = await fetch('login_process.php', {
                method: 'POST',
                body: formData
            });
```

```
            const result = await response.json();  // Expecting JSON response from PHP

            if (result.success) {
                window.location.href = 'index.php';
            } else {
                showError(emailInput, result.message || 'Login failed. Please try again.');
            }
        } catch (error) {
            showError(emailInput, 'An error occurred. Please try again.');
        } finally {
            submitButton.classList.remove('loading');
            buttonText.textContent = 'Login';
        }
    }
});
emailInput.addEventListener('input', () => {
    if (emailInput.value && validateEmail(emailInput.value)) {
        clearError(emailInput);
    }
});

passwordInput.addEventListener('input', () => {
    if (passwordInput.value) {
        clearError(passwordInput);
    }
});
</script> -->
<script>
    document.getElementById('loginForm').addEventListener('submit', async function
(e) {
    e.preventDefault(); // Prevent form submission to handle validation first
    let isValid = true;
    const emailInput = document.getElementById('email');
    const passwordInput = document.getElementById('password');
    const errorMessage = document.querySelector('.error-message');

    // Clear previous errors
    emailInput.classList.remove('error');
    passwordInput.classList.remove('error');
    errorMessage.style.display = 'none';

    // Validate Email
    if (!emailInput.value || !validateEmail(emailInput.value)) {
        emailInput.classList.add('error');
        isValid = false;
```

```
            errorMessage.style.display = 'block';  // Show the error message
        }

        // Validate Password
        if (!passwordInput.value) {
            passwordInput.classList.add('error');
            isValid = false;
        }

        if (isValid) {
            const submitButton = document.querySelector('.btn-primary');
            const buttonText = submitButton.querySelector('span');

            submitButton.classList.add('loading');
            buttonText.textContent = 'Logging in...';

            try {
                const formData = new FormData(document.getElementById('loginForm'));
                const response = await fetch('login_process.php', {
                    method: 'POST',
                    body: formData
                });

                const result = await response.json();  // Parse JSON response

                if (result.success) {
                    window.location.href = 'index.php'; // Redirect on success
                } else {
                    alert(result.message); // Show error message from the server
                }
            } catch (error) {
                alert('An error occurred. Please try again later.');
            } finally {
                submitButton.classList.remove('loading');
                buttonText.textContent = 'Login';
            }
        }
    });
    function validateEmail(email) {
        const re = /^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,4}$/;
        return re.test(email);}
  </script>
</body>
</html>
```

Resigter.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Register - Premium Auto Marketplace</title>
   <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500;600;700&
display=swap" rel="stylesheet">
   <link rel="stylesheet" href="style.css">
   <script src="https://www.google.com/recaptcha/api.js" async defer></script>
</head>
<body>
   <div class="header"></div>
      <nav>
         <ul>
            <li><a href="index.html">Home</a></li>
            <li><a href="http://127.0.0.1:8000/cars">Cars</a></li>
            <li><a href="login.html">Login</a></li>
            <li><a href="register.html">Register</a></li>
         </ul>
      </nav>
   </div>
   <div class="container">
      <div class="form-box">
         <h1>Create Account</h1>
         <form id="myForm" action="data.php"  method="post">
            <div class="input-field">
               <input type="text" name ="name" id="name" placeholder=" ">
               <label for="name">Full Name</label>
            </div>
            <div class="input-field">
               <input type="email" name ="email"id="email" placeholder=" " >
               <label for="email">Email Address</label>
            </div>
            <div class="input-field">
               <input type="tel" name="phone" id="phone" placeholder=" " >

               <label for="phone">Phone Number</label>
            </div>
            <div class="input-field">
               <input type="password" name="password" id="password" placeholder=" "
>
               <label for="password">Password</label>
            </div>
```

```
            <div class="g-recaptcha" data-sitekey="6Lc2CN0qAAAAAK-k7Ipm-
rPThRNiDyBPnciDcEay"></div>
            <p id="recaptchaError" style="color: red; font-size: 15px; font-family: 'Inter',
sans-serif "></p>

            <div class="btn-field">
              <button type="submit" id="submitButton" class="btn btn-primary mt-3">
                <span>Register</span>
              </button>
              <button type="button" onclick="window.location.href='index.html'"
class="btn btn-secondary mt-3">
                <span>Back</span>
              </button>
            </div>
            <p class="login-link">
              Already have an account? <a href="login.html">Login here</a>
            </p>

            <div id="thankYouMessage" class="alert alert-info mt-3"
style="display:none;">
              Thank you for your submission. Your Details Are Submitted!
            </div>
          </form>

        </div>
    </div>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-
validate/1.19.3/jquery.validate.min.js"></script>
    <script>
      $(document).ready(function() {
        $.validator.addMethod("noOnlySpaces", function(value) {
          return $.trim(value).length !== 0;
        }, "This field cannot contain only spaces");

        $.validator.addMethod("customEmail", function(value, element) {
          return this.optional(element) ||
            /^[a-zA-Z0-9._%+-]+@(?:[a-zA-Z0-9-]+\.)+(?:in|com|org)$/.test(value);
        }, "Please enter a valid email address ending with .in, .com, or .org");

        $.validator.addMethod("numericOnly", function(value, element) {
          return this.optional(element) || /^\d+$/.test(value);
        }, "Please enter a valid phone number");
```

```
$('#myForm').validate({
  rules: {
    name: {
      required: true,
      noOnlySpaces: true
    },
    email: {
      required: true,
      email: true,
      customEmail: true,
      noOnlySpaces: true
    },
    phone: {
      required: true,
      minlength: 7,
      maxlength: 15,
      noOnlySpaces: true
    },
    password: {
      required: true,
      noOnlySpaces: true
    }

  },
  messages: {
    name: {
      required: "Please enter your name",
      noOnlySpaces: "This field cannot contain only spaces"
    },
    email: {
      required: "Please enter your email address",
      email: "Please enter a valid email address",
      customEmail: "Please enter a valid email address ending with .in, .com, or
.org",
      noOnlySpaces: "This field cannot contain only spaces"
    },
    phone: {
      required: "Please enter your phone number",
      minlength: "Phone number must be at least 7 digits",
      maxlength: "Phone number must not exceed 15 digits",
      noOnlySpaces: "This field cannot contain only spaces"
    },
    password: {
      required: "Please enter your password",
      noOnlySpaces: "This field cannot contain only spaces"
```

```
    }

},
errorElement: "span",
errorPlacement: function(error, element) {
    error.addClass("invalid-feedback");
    if (element.prop("type") === "checkbox") {
        error.insertAfter(element.closest(".input-checkbox-wrapper"));
    } else {
        error.insertAfter(element);
    }
},
highlight: function(element, errorClass, validClass) {
    $(element).addClass("is-invalid").removeClass("is-valid");
},
unhighlight: function(element, errorClass, validClass) {
    $(element).removeClass("is-invalid");
},
submitHandler: function(form) {
    var recaptchaResponse = grecaptcha.getResponse(0);
    if (recaptchaResponse.length == 0) {
        $('#recaptchaError').text('Please complete the reCAPTCHA.').show();
    } else {
        $('#recaptchaError').hide();
        $("#submitButton").attr("disabled", false);
        var formData = $(form).serialize();
        $.ajax({
            type: "POST",
            url: $(form).attr('action'),
            data: formData,
            success: function(response) {
                $("#thankYouMessage").show();
                form.reset();
                setTimeout(function() {
                    $("#thankYouMessage").hide();
                }, 7000);
                $('#myForm').get(0).reset();
                grecaptcha.reset(0);
            },
            error: function(xhr, status, error) {
                console.error("Error submitting the form: ",
                    error);
            }
        });
    }
```

```
            }
          });
        });
    </script>
  </body>
</html>
```

Stlye.css

```
:root {
    --primary-color: #fe3265;
    --primary-dark: #e41f4f;
    --text-light: #ffffff;
    --text-dark: #333333;
    --background-light: #ffffff;
    --error-color: #ff3b3b;
    --success-color: #00c853;
}

body, html {
    margin: 0;
    padding: 0;
    font-family: 'Poppins', sans-serif;
    height: 100%;
}

nav {
    display: flex;
    align-items: center;
    justify-content: flex-end;
    padding: 20px 0;
    position: fixed;
    top: 0;
    left: 0;
    right: 0;
    z-index: 1000;
    padding: 20px 8%;
    /* background: rgba(0,0,0,0.8); */
    /* backdrop-filter: blur(10px); */
}
nav ul {
    list-style: none;
    display: flex;
    gap: 30px;
}
nav ul li a {
    color: #fff;
```

```
    text-decoration: none;
    font-weight: 500;
    transition: all 0.3s ease;
    position: relative;
    font-size: 18px;
}
nav ul li a::after {
    content: '';
    position: absolute;
    bottom: -5px;
    left: 0;
    width: 0;
    height: 2px;
    background: var(--primary-color);
    transition: width 0.3s ease;
}
nav ul li a:hover {
    color: var(--primary-color);
}
nav ul li a:hover::after {
    width: 100%;
}
.content {
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
    text-align: center;
    color: #ffff;
    width: 90%;
    max-width: 800px;
}

/* .container {
    width: 100%;
    min-height: 100vh;
    background-image: linear-gradient(rgba(0,0,0,0.7), rgba(0,0,0,0.7)), url('car.jpg');
    background-position: center;
    background-size: cover; `
    display: flex;
    align-items: center;
    justify-content: center;
    padding: 20px;
} */
```

```
.container {
    width: 100%;
    min-height: 100vh;
    background-image: linear-gradient(rgba(0,0,0,0.7), rgba(0,0,0,0.7)), url('img3.jpg');
    background-position: center;
    background-size: cover;
    display: flex;
    align-items: center;
    justify-content: center;
    padding: 20px;
    animation: change 20s infinite linear;
}

@keyframes change {
    0% {
        background-image: url('car.jpg');
    }
    25% {
        background-image: url('img2.jpg');
    }
    50% {
        background-image: url('img3.jpg');
    }
    75% {
        background-image: url('img4.jpg');
    }
    100% {
        background-image: url('car.jpg');
    }
}

.error {
    color:red;
}

.alert {
    padding: 15px;
    border-radius: 5px;
    font-size: 16px;
    font-weight: 500;
    display: block;
    width: 100%;
    max-width: 600px;
    margin: 0 auto;
}
```

```
.alert-info {
    background-color: #d1ecf1;
    border: 1px solid #bee5eb;
    color: #0c5460;
}

.mt-3 {
    margin-top: 1rem;
}

.form-box {
    width: 100%;
    max-width: 450px;
    background: var(--background-light);
    padding: 40px;
    border-radius: 20px;
    box-shadow: 0 15px 30px rgba(0,0,0,0.2);
    animation: slideUp 0.5s ease;
}

@keyframes slideUp {
    from { opacity: 0; transform: translateY(20px); }
    to { opacity: 1; transform: translateY(0); }
}

.form-box h1 {
    font-size: 32px;
    margin-bottom: 30px;
    color: var(--text-dark);
    text-align: center;
    position: relative;
}

.form-box h1::after {
    content: '';
    position: absolute;
    bottom: -10px;
    left: 50%;
    transform: translateX(-50%);
    width: 60px;
    height: 4px;
    background: var(--primary-color);
    border-radius: 2px;
}
```

```
.input-field {
    position: relative;
    margin-bottom: 20px;
}

.input-field label {
    position: absolute;
    top: 50%;
    left: 15px;
    transform: translateY(-50%);
    color: #666;
    pointer-events: none;
    transition: all 0.3s ease;
}

.input-field input {
    width: 100%;
    padding: 15px;
    border: 2px solid #eee;
    border-radius: 10px;
    outline: none;
    font-size: 16px;
    transition: all 0.3s ease;
}

.input-field input:focus {
    border-color: var(--primary-color);
}

.input-field input:focus + label,
.input-field input:not(:placeholder-shown) + label {
    top: 0;
    left: 10px;
    transform: translateY(-50%);
    background: white;
    padding: 0 5px;
    font-size: 14px;
    color: var(--primary-color);
}

.btn-field {
    display: flex;
    gap: 15px;
}
```

```
.btn {
    flex: 1;
    padding: 15px;
    border-radius: 10px;
    font-weight: 600;
    font-size: 16px;
    cursor: pointer;
    transition: all 0.3s ease;
    display: flex;
    align-items: center;
    justify-content: center;
    gap: 10px;
}

.btn-primary {
    background: var(--primary-color);
    color: var(--text-light);
    border: none;
}

.btn-primary:hover {
    background: var(--primary-dark);
    transform: translateY(-2px);
}

.btn-secondary {
    background: transparent;
    color: var(--text-dark);
    border: 2px solid #eee;
}

.btn-secondary:hover {
    border-color: var(--primary-color);
    color: var(--primary-color);
    transform: translateY(-2px);
}

.error-message {
    color: var(--error-color);
    font-size: 14px;
    margin-top: 5px;
    display: none;
}
```

```
.input-field.error input {
    border-color: var(--error-color);
}

.input-field.error .error-message {
    display: block;
}

.loading {
    position: relative;
    pointer-events: none;
}

.loading::after {
    content: '';
    position: absolute;
    width: 20px;
    height: 20px;
    border: 2px solid #fff;
    border-radius: 50%;
    border-top-color: transparent;
    animation: spin 0.8s linear infinite;
}

@keyframes spin {
    to { transform: rotate(360deg); }
}

/* Specific styles for login page */
.forgot-password {
    text-align: right;
    margin-bottom: 20px;
}

.forgot-password a {
    color: var(--primary-color);
    text-decoration: none;
    font-size: 14px;
    transition: color 0.3s ease;
}

.forgot-password a:hover {
    color: var(--primary-dark);
}
```

```
/* Specific styles for register page */
.password-strength {
    margin-top: 5px;
    font-size: 14px;
}

.strength-meter {
    height: 4px;
    background: #eee;
    margin-top: 5px;
    border-radius: 2px;
    overflow: hidden;
}

.strength-meter div {
    height: 100%;
    width: 0;
    transition: width 0.3s ease;
}

.weak { background: var(--error-color); }
.medium { background: #ffa000; }
.strong { background: var(--success-color); }

/* Responsive styles */
@media (max-width: 480px) {
    .form-box {
        padding: 30px 20px;
    }

    .btn-field {
        flex-direction: column;
    }
}
```

AuthController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\User;
use Hash;
use Session;
use Auth;
```

```
class AuthController extends Controller
{
    public function login(){
        return view('auth.login');
    }

    public function register(){
        return view('auth.register');
    }

    public function home(){
        return view('layouts.home');
    }

    public function registerUser(Request $request){
        $request->validate([
            'name'=>'required',
            'email'=>'required|email|unique:users',
            'password'  =>  'required|min:8|confirmed',

        ]);

        $user = new User();
        $user->name = $request->name;
        $user->email = $request->email;
        $user->password = Hash::make($request->password);
        $user->role = 'admin';
        $result = $user->save();
        if($result){
            return redirect('/')->with('success','you have registered');
        }else{
            return back()->with('fail','something wrong');
        }
    }

    public function loginUser(Request $request){
        $request->validate([
            'email'=>'required|email',
            'password'=>'required',

        ]);
        $user = User::where('email','=',$request->email)->first();
        if($user){
            if(Hash::check($request->password, $user->password)){
                $request->session()->put('loginId', $user->id);
```

```
                return redirect('/view-cars');
            }else{
                return back()->with('fail','incorrect password');
            }
        }else{
            return back()->with('fail','incorrect email');
        }
    }

    public function logout(){
        if(session::has('loginId')){
            session::pull('loginId');
            return redirect('/');
        }
    }
}
```

RESULT AND DISCUSSION

RESULT AND DISCUSSION

Results

The Car Seller Project Website was successfully developed and tested, providing a functional and user-friendly platform for buying and selling vehicles. The system was evaluated based on performance, usability, security, and scalability, yielding the following results:

1.Performance

- The website loads quickly, with optimized queries ensuring fast database interactions
- The search functionality effectively filters vehicles based on user preferences.
- The platform handles multiple users simultaneously without significant delays.

2.Usability

- A clean and intuitive UI ensures easy navigation for users.
- Responsive design allows seamless access on mobile, tablet, and desktop devices.
- Simple vehicle listing and inquiry process enhances user experience.

3.Security

- User authentication prevents unauthorized access
- CSRF and SQL injection protections are implemented in Laravel for secure transactions.
- Data validation and encryption ensure user data safety.

4.Scalability

- The platform is scalable, allowing more users and vehicle listings without performance issues
- The modular backend structure in Laravel enables future expansions like AI recommendations and payment integration.

Discussion

The project met its primary objectives by offering a feature-rich platform for vehicle trading. However, some limitations and areas for improvement were identified:

1.Strengths

- Efficient vehicle search and filtering system.
- Secure user authentication and inquiry handling.
- Scalable database with optimized queries.

2.Challenges Faced

- Integration of advanced features like AI-based recommendations was beyond the current scope.
- Real-time chat functionality was not implemented but could enhance communication.
- Limited payment options, requiring future online payment gateway integration.

3.Future Enhancements

- Implement AI-based recommendations for personalized search results.
- Add a real-time chat system for direct buyer-seller communication.
- Integrate payment gateways for secure transactions.

Develop a mobile application for better accessibility.

CONCLUSION AND FUTURE WORK

CONCLUSION AND FUTURE WORK

Conclusion

The Car Seller Project Website successfully provides a secure, efficient, and user-friendly platform for buying and selling vehicles online. It enables users to list vehicles, search for cars, send inquiries, and manage their listings with ease. The project was developed using Laravel (PHP) for the backend, MySQL for database management, and HTML, CSS, and JavaScript for the frontend, ensuring scalability, security, and responsiveness.

Through comprehensive testing, the system demonstrated high performance, secure user authentication, and a smooth user experience. The implementation of search filters, inquiry management, and an admin monitoring system makes the platform reliable for both buyers and sellers.

Key Achievements

- User authentication and secure access.
- Advanced search and filtering options.
- Easy vehicle listing and inquiry system.
- Optimized database performance with MySQL.

While the project successfully meets its objectives, future improvements such as online payment integration, AI-based car recommendations, real-time chat, and mobile app development can further enhance the platform.

Future Work

To enhance the **Car Seller Project Website**, the following improvements can be implemented in the future:

1. Online Payment Integration

To enhance the transaction process, a secure payment gateway can be integrated, allowing buyers to make direct online payments for vehicles.

2. AI-Based Car Recommendations

Implementing machine learning algorithms will allow the system to analyze user preferences and browsing history to suggest relevant vehicle listings. This will improve user engagement and make the platform more intelligent and personalized.

3. User Reviews & Ratings

A review and rating system will allow buyers to rate sellers and vehicles, helping future users make informed decisions. This feature will increase trust and transparency within the platform

.

4. Real-Time Chat System

Adding a chat feature will enable instant communication between buyers and sellers. This will reduce delays in responses, leading to faster negotiations and improved user experience.

5. Mobile App Development

To reach a wider audience, developing a mobile application for Android and iOS will make the platform more accessible. A mobile app will provide push notifications, location-based vehicle searches, and a better user experience for on-the-go users.

Reference

- https://laravel.com/docs
- https://dev.mysql.com/doc/
- https://developer.mozilla.org/
- https://www.php.net/docs.php
- https://www.w3schools.com/
- https://www.tutorialspoint.com/
- https://plantuml.com/
- https://www.google.com/