



Vehicle Services and Road-Side Assistance Portal

A dissertation submitted for the Degree of Master of Information Technology

W.K.S.B. Walallawita

University of Colombo School of Computing

2020



Abstract

Vehicles have become a vital component of the life of each and everyone in the society. Most of the people tend to use a personal vehicle due to difficulties they face in public transport in Sri Lanka. However, with the use of vehicle, the next thing comes in is vehicle maintenance and repair. In most scenarios, vehicle owners face the difficulty in finding a proper mechanic / service station to repair or perform routine maintenance of their vehicles. On the other hand, there are plenty of mechanics / service stations in the country with plenty of expertise to perform above mentioned repairs or routine maintenance, who are struggling to find proper customer base. As a solution for this problem, my client Rapid Taxies (Pvt) Ltd. requested to come up with a suitable system to interconnect vehicle owners and mechanics.

To address the above-mentioned problem, a web-based portal and a mobile application was proposed as a platform to interconnect vehicle owners and mechanics, based on their location, required expertise and preferences. This system was developed with the combination of Agile Methodology and Throw-away prototyping. Throw-away prototyping was mainly used to design the User Interface of the portal for client's approval and Agile Programming was mainly used for the functional development of the system.

With the use of this system, vehicle owners would be able to find nearby mechanics based on their vehicle type and location. In the same way, mechanics would also be able to find nearby vehicle owners who are seeking their support. The developed system, web site and the mobile application will be evaluated against the detailed requirements specified in this dissertation under Chapter 2.

This dissertation compiles the areas addressed through the developed system, analysis of the problem, system design and implementation areas. Finally, it is argued that the challenges that the developer had to face while implementing each stage of the developed system as well as the future implementations and expansions of the project.

Declaration

The dissertation is my original work and has not been submitted previously for a degree at this or any other university/institute. To the best of my knowledge it does not contain any material published or written by another person, except as acknowledged in the text.

Student's Name: W.K.S.B. Walallawita

Signature:

Date: 15.05.2020

Acknowledgements

This work would not have been possible without the tremendous support from my project supervisor Dr. H.A. Caldera and the MIT3201 Course Coordinator Senior Lecturer, Mr. Viraj Welgama. I am especially indebted to the colleagues of Rapid Taxies (Pvt.) Ltd. who provided me with a great support throughout the completion of this project.

I would like to thank and extend the gratitude towards my parents who were always behind me by motivating me and providing all the necessary support for my education and career.

Table of Contents

1	Introduction	6
2	Background	8
2.1	Requirement Analysis	8
2.2	Review of Similar Systems Available	11
2.3	A comparison of alternative design strategies	12
3	Problem Analysis	17
4	System Design	19
4.1	System Architecture and Architecture Design	19
4.2	Database Design	24
4.3	Use Case Diagram	27
4.4	User Interface Design	28
5	Implementation	33
5.1	Implementation of Admin Panel	33
5.2	Implementation of Web Portal	36
5.3	Implementation of Mobile App	39
6	Testing and Evaluation	42
6.1	Testing	42
6.2	Evaluation	42
7	Conclusion	44
8	References	45
9	Appendices	46
9.1	User Manual	46
9.2	Test Cases	50

List of Figures

Figure 2-1 Uber System Architecture	11
Figure 2-2 - Waterfall Model	12
Figure 2-3 - Spiral Model	13
Figure 2-4 - Rapid Application Development (RAD)	14
Figure 2-5 - Extreme Programming (XP)	15
Figure 4-1 External Systems Architecture	19
Figure 4-2 AWS Security Architecture	22
Figure 4-3 - 3 Tier Architecture	23
Figure 4-4 Entity Relationship Diagram	25
Figure 4-5 Use Case Diagram	27
Figure 4-6 - User Interface of Mobile App	32
Figure 5-1 - Admin Panel Models	33
Figure 5-2 - Controllers of Admin Panel	34
Figure 5-3 - CRUD Operations in Car Type Model	34
Figure 5-4 - Controller for Car Type	35
Figure 5-5 - Views for CRUD Operations in Admin Panel Car Types	35
Figure 5-6 - View Car Types view	35
Figure 5-7 - Folder structure of Web Portal pages	36
Figure 5-8 - Home Page of Web Portal	36
Figure 5-9 - Models for Web Portal	37
Figure 5-10 - JQuery AJAX requests	38
Figure 5-11 - User Interface files	39
Figure 5-12 - Frontend JS files	39
Figure 5-13 - Controller.js	40
Figure 5-14 - List Vehicles	40

1 Introduction

Vehicles have become a vital component of the life of each and everyone in the society. Most of the people tend to use a personal vehicle due to difficulties they face in public transport in Sri Lanka. However, when it comes to the maintenance and break-down of vehicles, with the busy schedules, people find it difficult to allocate time for the maintenance or repairs of their vehicles. Also, people may find that some Vehicle Repair Stations or Service Stations are very expensive (prices are not standardized). On the other hand, it is very helpful for everyone, if they can locate a nearby service station, repair center or a person who is willing to visit your place and do the needful.

On the other hand, there might be skilled mechanics those who are looking for work to progress in their career. Also, there might be corporate level organizations who wish to provide their services in mobile basis.

The main objective of this proposed system is to minimize the hassle that a vehicle owner has to face when it comes to repairs and maintenance of their vehicles. This portal will be developed as an inter-connection platform for Customers and mechanics which is inspired from the same Service Oriented Architecture, which was brought into action by Uber, Inc [2]. One of the main aims are to implement this project with location-based services of Mobile Devices and features of Google Maps API.

Also, this system is a complete development from scratch, even the business model of the system. Unlike other systems that is being developed, which is to replace the current manual or obsolete systems, the complete project from the idea has to be developed from the scratch.

The proposed system will consist of a backend for administrators, with the capability of performing CRUD operations and reporting related to the system, it's users and services. There will be a front-end website and a mobile application for system's users, which are mechanics and vehicle owners, which will facilitate them to get registered with the portal and make use of its services. Scope of these services would be, maintaining profiles for each type of user with their personal details, contact details, vehicles and service expertise etc. and working in the manner of SaaS to provide necessary services to each user type. As an example, a vehicle owner would be able to add his details, vehicle details to the profile and request services from nearby mechanics when required. In the same way, mechanics also would be able to maintain his own profile in the portal with his details and expertise and then provide services when a nearby vehicle owner requests.

At the end of this project, a web portal and a mobile application will be developed to address the above-mentioned problems. In this dissertation, initially the Background of the proposed project is discussed. Then, an elaborated analysis has been carried out under Chapter 3 about the scope of this project and the requirements of the proposed system. Next, an overview on System Designing, Architecture, Database Design and User Interface Design have been provided through Chapter 4. Under Chapter 5, Implementation stage of the project is discussed and then, testing and evaluation stage of the whole project is discussed. At this stage, what has been achieved and what has not been achieved is also elaborated with proper reasoning.

2 Background

2.1 Requirement Analysis

The main aim of the proposed system is to provide a common portal to its' main stakeholders; Customers and Mechanics. The platform should be an online one and it should provide a web-based interface as well as a mobile application for the users. There will be mainly three user groups in the system and the identified requirements are as follows.

2.1.1 Functional Requirements

Since, there is no current system to study, the requirements of the proposed system have been briefed by the client with similar SaaS systems available. According to the requirements briefed by the client, following functional requirements were identified.

- Users and User Profiles

There will be three main user groups in the system and the roles of each user can be customized as needed. Three main user groups are Administrators, Customers and Mechanics.

Administrators will only have access to the backend of the system. In the backend, new user creation, role assignment, user profile management, monitoring and managing ongoing tasks and report generation functions will be facilitated. Apart from that, admins should have the access to create, edit and delete new customers, mechanics, services and service areas.

When it comes to Customers, anyone should be able to get registered with the (Sign Up) Portal by filling the information required. At the registration process, as the first step, vehicle owner will be prompted to enter his/her phone number and that should be verified with an OTP code sent as a text message. Once the phone number is confirmed, user can proceed with completing his/her profile on the portal. At this point user's Full Name, Email Address, Home Address, NIC Number and secondary phone numbers should be captured. Also, user will have the option to store his/her preferred payment method in the system. All this information must be encrypted and stored in the database. After completing the user profile, user should have the option to add his/her vehicles to the system. Vehicle number, VIN number, engine number, vehicle make, model and manufactured year should be captured at this point. Whole process mentioned above must be able to complete through the Mobile application as well as the web portal.

When it comes to Mechanics, they should make the registration through web portal. At the registration process, Username, Password, Owner's name, workshop's name, owner's NIC number, location (geocode) of the workshop, postal address, email address and contact numbers should be captured. After completion of above process, workshop owners can proceed to define their expertise. At this point, types of repairs that they undertake, vehicle makes and models that they are specialized in etc. should be captured. After completion of the registration process, an Administrator will have to confirm the registration as a valid one.

- Service requests by Customers

When a registered vehicle owner logs in to the portal (either with mobile application or web portal), the user should be able to see a map with his current location pinned down. At this point, user should be able to select the previously added vehicle from the list and call for a mechanic. When calling for a mechanic, user must select the type of problem and a brief description about the problem from a predefined list. Also, some photographs should also be attached if applicable. Once the user requests a mechanic, the request must be sent to a set of Automotive workshops / workshop mechanics within a predefined distance from vehicle owner's location.

There should be a list of past requests and the amount spent on each service as history in each vehicle owner's profile.

- Request acceptance by Mechanics / Automotive workshops

The Mechanic should be able to log in to the portal using the Web Application. When a vehicle owner makes a request, the request must be dispatched to mechanics within a given radius, based on their expertise. When the mechanic receives the request from a vehicle owner, the mechanic has the choice to Accept or Decline the request. Once a mechanic approves the request, he should be given with the directions on the Map of the mobile phone to the location of the vehicle owner, and choices to make call or send a text message to the vehicle owner. If a mechanic is rejecting an order from a customer, it should be saved as Rejected Orders in the system.

- Completing a service

For each request, once the request is accepted by a mechanic, the mechanic will do a basic inspection after visiting the vehicle owner's location. There should be a fixed charge for inspection. Once the inspection is done, Mechanic should define the type of repair and parts to be replaced in the portal using Mobile Application. Vehicle owner will get a notification with the approximate quotation for the service to be

performed. Once the customer agreed with the quotation, Mechanic can proceed with the job.

Once the job is completed mechanic must mark the job as completed using the Mobile Application and vehicle owner should accept the completion of job through the portal. Then the vehicle owner should be able to make the payment with cash or credit / debit card using the portal itself.

- **Rating System**

A rating system should be there for Customers, mechanics and workshop owners. Once the job is completed Customers must be able to rate the mechanic within a given criteria as well as the workshop.

In the same way, mechanics must also be able to rate the Customers in the same manner.

Star rating system is preferred where the 5 Stars are the highest rating and 1 star is the lowest. With the rating system, all users must be able to write a review upon the completion of each job. Based on the star ratings given for each job, a score should be calculated and displayed in each user's profile.

- **Activity Log**

Apart from above mentioned requirements, a system wide User Activity Log should be maintained. Each time when a user performs an action, a log should be recorded in the database with a timestamp.

2.1.2 Non-Functional Requirements

- Minimal response time – since the system is working with web servers and database connections, it should be optimized to work under minimum response time.
- Reliability and consistency
- Security – A SSL certificate should be enabled at the web interface to ensure security in transactions happening through the system.
- User friendliness and Mobile friendliness – The user interfaces should be much user friendly and at the same time the interfaces should be responsive to all standard screen sizes and devices types.

2.2 Review of Similar Systems Available

The proposed system uses the same service-oriented architecture as Uber and Pick Me. Location based services, radius search based on current location of the users (Latitude and Longitude) etc. follow the same strategy as above-mentioned systems.

As in Uber and Pick Me, estimated time of arrival, directions, traffic etc. will be taken with the inputs of Google Maps API and the location of mobile devices [2,5]. However, Uber uses machine learning and AI to recognize the practices of each user to provide better service. Due to the time constraints and the scope of the proposed system, such advanced factors will not be taken into consideration at the initial phase of development.

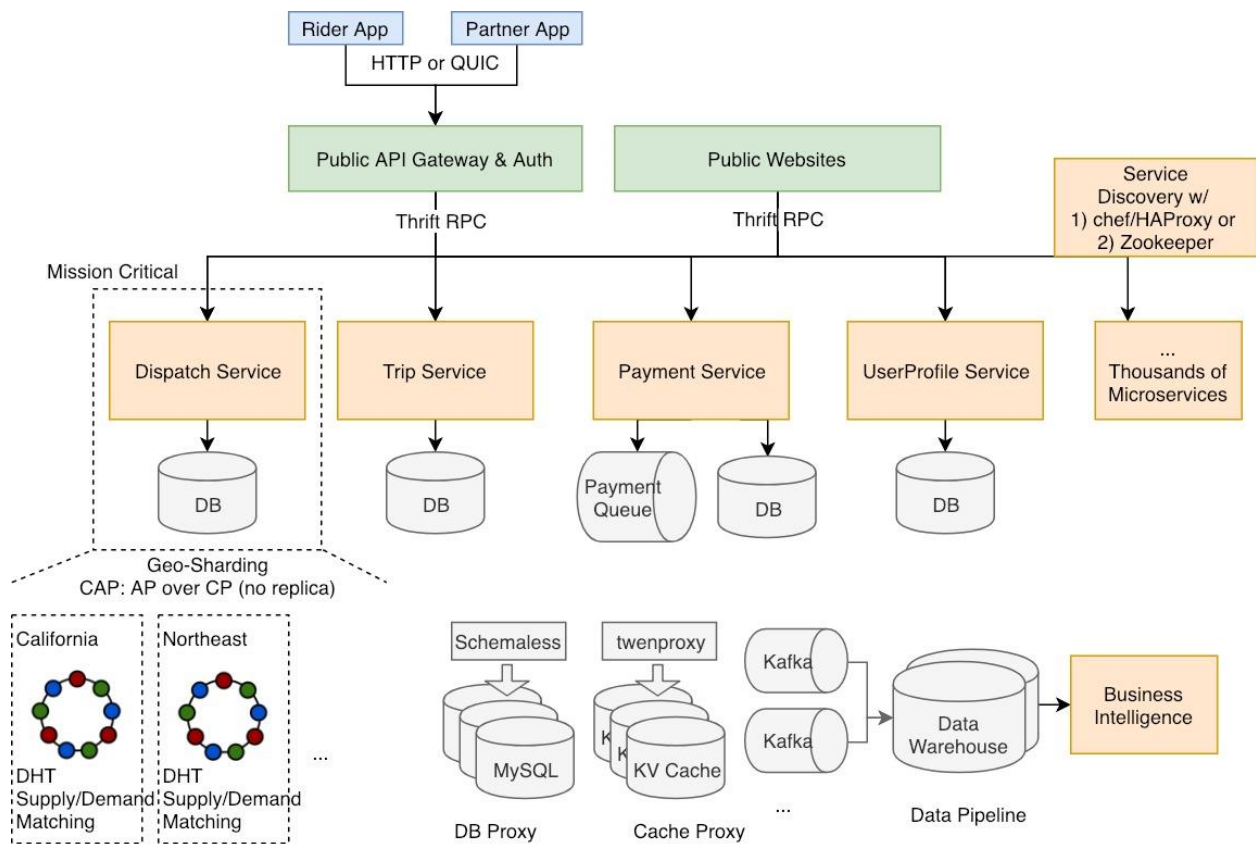


Figure 2-1 Uber System Architecture

In Uber and Pick Me, when a request is made by a user for a Taxi, the request is rotated among the nearest available drivers. Once the request is accepted by a driver, it will be notified to the rider [2]. The same approach will be implemented in the proposed system as well. As in Uber and Pick me, the contact details of each other will be notified to them only after the request is accepted.

Similarly, there will be records on previous jobs that each user has completed and requested. Also, the amount spent on each vehicle and the type of payments etc. will be available. All these features are inspired from above mentioned systems like Uber and Pick Me. These features and approaches will be adopted to cater the requirements of proposed system.

In Uber and Pick Me, the system is mainly targeted for the mobile devices. Due to technical expertise constraints and time constraints, during the initial phase of this project, the proposed system will mainly be targeted for Web Browsers even though a mobile app is also developed for the same purpose. (Only selected functions will be available at the mobile app)

2.3 A comparison of alternative design strategies

After carrying out a comprehensive research, 4 methodologies that are suitable for web / mobile application projects were selected and evaluated.

Waterfall Model

Waterfall model is a sequential development approach, in which development is seen as flowing steadily downwards through several phases.

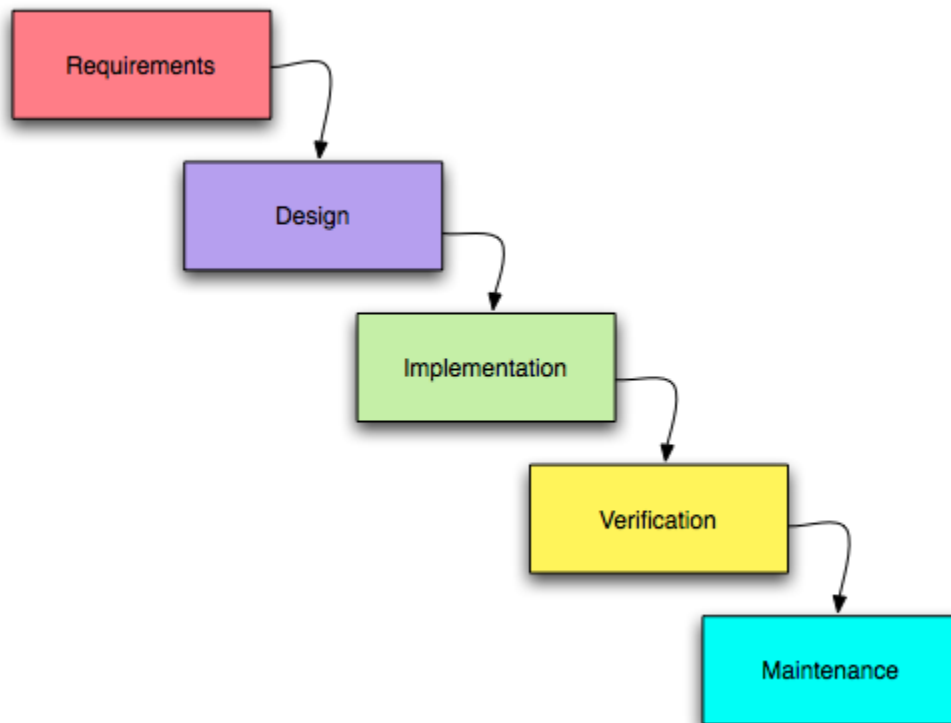


Figure 2-2 - Waterfall Model

Pros.

- Easy to understand and implement
- Saves significant amount of time as the methodology is straight forward.
- Allows easy testing and analysis

Cons.

- Requirements should be predefined and stable.
- Not applicable for maintenance projects
- Not suitable for long and ongoing projects with changing requirements
- There's no option to go back to the previous phase if required.

Since, the requirements of the proposed system are not stable and the project is long and ongoing one, Waterfall model was not selected.

Spiral Model

Spiral Model extends the waterfall model by adding rapid prototyping to combine advantages of top-down and bottom-up concepts.

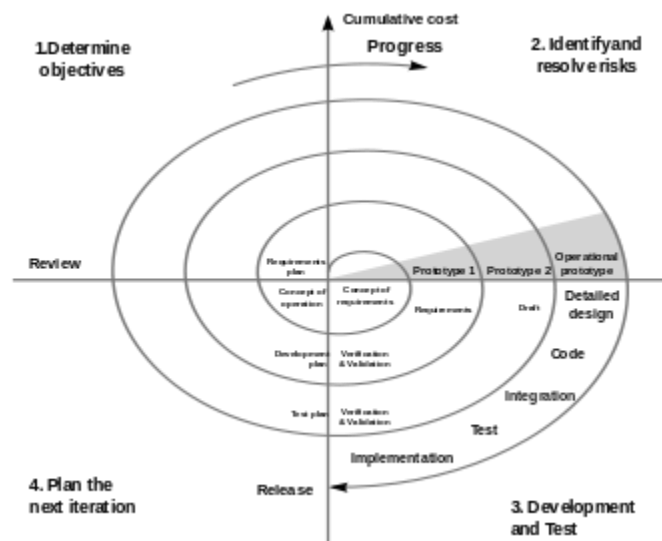


Figure 2-3 - Spiral Model

Pros.

- Excellent for large and complex projects
- Facilitate changing business requirements to be implemented throughout the project.
- Suitable for projects with a high risk

Cons.

- Higher costs
- Failures in risk analysis phase may damage the integrity of whole project
- The project may get continued and never completed

Even though this model supports varying requirements, due to the additional phases and costs involved in this methodology, Spiral Model was not selected for the project.

Rapid Application Development Methodology (RAD)

Rapid Application Development (RAD) is a methodology that prioritizes rapid prototype releases and iterations. Unlike the Waterfall method, Rapid Application Development emphasizes the use of software and user feedback over strict requirement analysis.

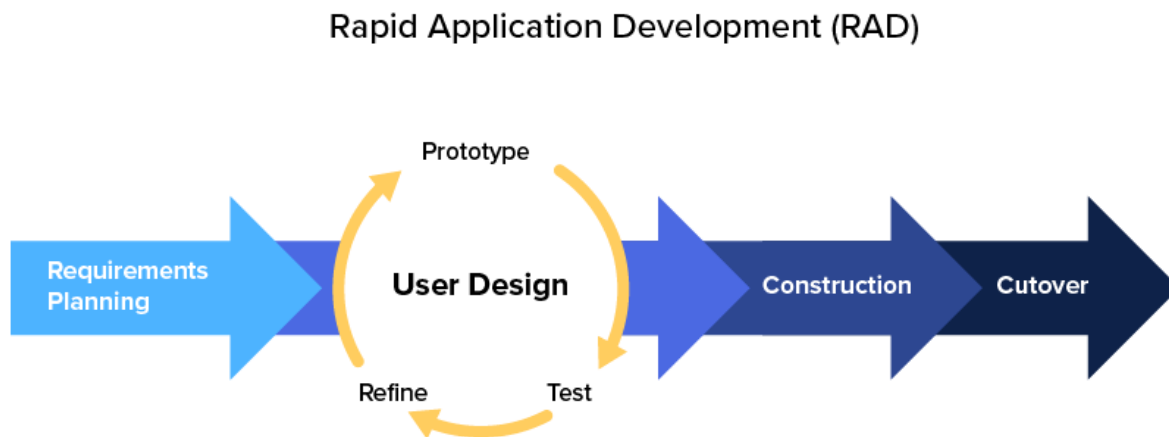


Figure 2-4 - Rapid Application Development (RAD)

Pros.

- Depends on the skills of the team
- Works better on modularized systems
- Facilitate changing business requirements to be implemented throughout the project.

Cons.

- Requires extremely skilled persons
- Not applicable to small budgeted projects

Even though this supports and suitable for the proposed system, due to the higher skills and cost involved, this methodology was not selected.

Extreme Programming

Extreme Programming (XP) is an agile software development framework that aims to produce higher quality software, and higher quality of life for the development team. XP is the most specific of the agile frameworks regarding appropriate engineering practices for software development.

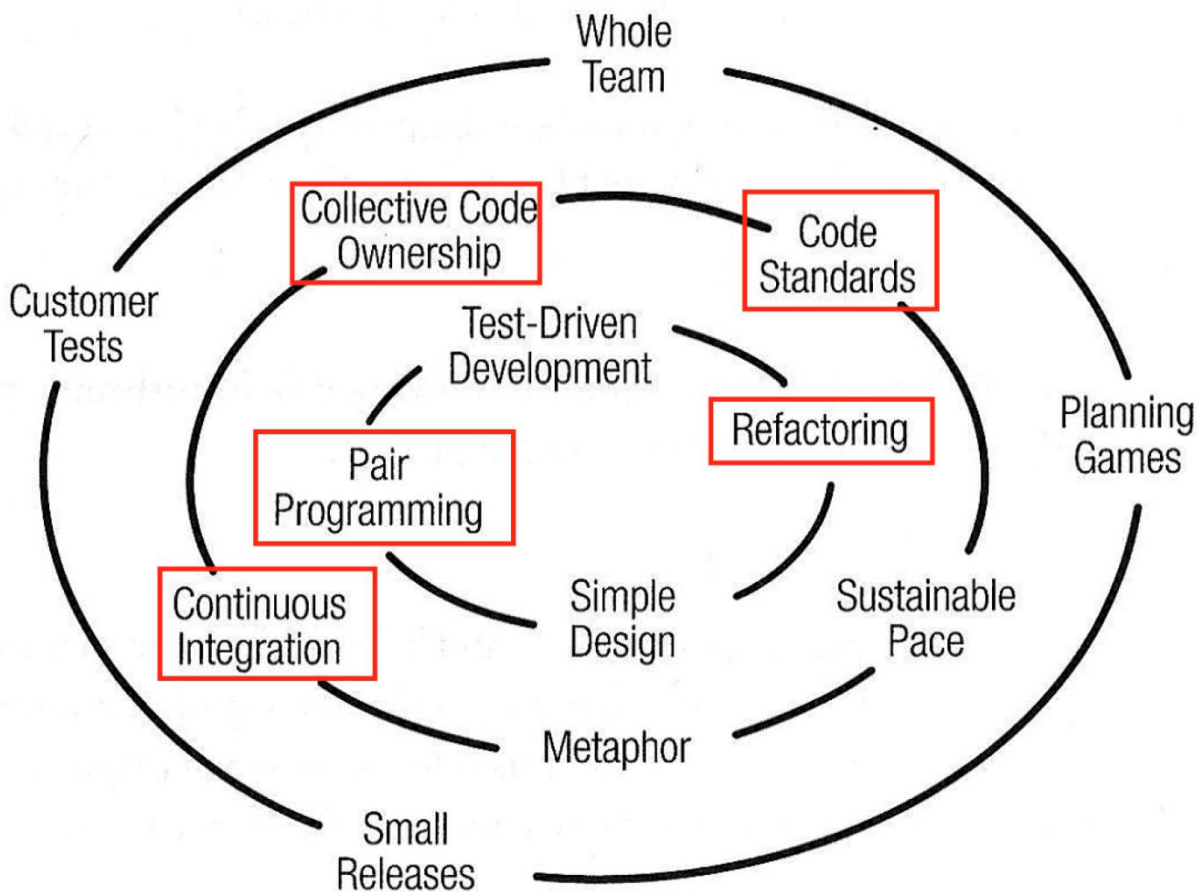


Figure 2-5 - Extreme Programming (XP)

Pros.

- Higher client involvement
- Developers are exceptionally committed to the project
- Equipped with modernistic methods for quality software

Cons.

- Effectiveness depends on the people involved
- Requires frequent meeting for development raising total costs
- Necessitates for excessive development changes

Due to higher client involvement, simplicity and support for frequent changes, this methodology was selected for the development of the proposed system.

The system will be developed from scratch using frameworks such as Bootstrap, Codeigniter, React JS and React Native [3,4]. For the development of web interface, Bootstrap framework and Codeigniter framework will be used [1]. Initially, it was decided to use MongoDB as the database as it will be easier when developing as the developer doesn't have to spend much time developing the Server Side. However, due to some technical constraints and program architectural issues, it was decided to carry out the development associated with MySQL database.

For the development of Mobile application, React Native framework will be used and also some open source libraries will also be used to handle location-based services, Google Maps and mobile device features like Camera, Sensors etc. [4]

When it comes to servers, a Linux server will be used which is running on Apache Web Server, PHP 7.2 and MariaDB as the database. PHPMyAdmin will also be installed on this server to make the database management process easier. Web Application / the portal will be hosted on this server and it will be accessible through web browsers from any location. When it comes to the mobile application, mobile application will send requests through XML or JSON to the web services developed on this server.

However, instead of above-mentioned technologies, for server-side development, Microsoft ASP.NET and MS SQL Server could be used. However, since these technologies are non-open source ones and the server cost of these technologies are higher, it has been decided to move forward with above-mentioned open source technologies. Also, flexibility, availability of support and tools for open source technologies are much higher when compared to commercial tools like ASP.NET and MS SQL server.

3 Problem Analysis

- User and User Profiles

According to the Requirement Specification, there are 3 user types in the system. Those are namely, Back end users, Customers and Mechanics. Users from one user type is not going to move into another type. Therefore, separate tables for each user type can be created in the database.

When it comes to user roles, only the back-end users will have user roles. System will be developed in a way that the super user will be able to define user roles, assign functions to each user role and assign users to each user role.

Customers and Mechanics will have separate user profiles. In mechanic's profiles, he will be able to define his expertise (vehicle make, model and types of repairs), qualifications and the working hours. In vehicle owner's profile, he will be able to define his payment information, billing information and the details (VIN, Registration Number, Make, Model and Year of Manufacture) vehicles that he currently own.

Since, each user type has different functions of their own, it was decided that it is the best to have separate tables for each user type in the database. Full Name, Email address, mobile number, NIC number, present address will be considered as mandatory information to be collected from each user.

- Service Requests

When a vehicle owner is making a request, the current location of the user will be displayed on a map. This map will be added using Google Maps API and user has the ability to change the location by moving the pin. At this point, the application should call Google Maps API and fetch the Latitude and Longitude of the selected location. Also, user inputs will be taken regarding the type of service that user requests, and details of the vehicle.

Then, the application should query the Mechanics from the database within the radius of 1 km from the given Latitude and Longitude, the service type and the vehicle type. Then the request should be sent to the Mechanic list resulted out by the query.

- Request Acceptance

Requests will be sent to mechanics as push notifications or a notification to his web application. Once a mechanic accepts a request for a service, he will be given with the vehicle model, contact number of vehicle owner, service type and the

location of the user. At this point, user will also be notified with a map of the mechanic's current location and estimated time of arrival (ETA). This location displayed for the vehicle owner should be a live display. Thus, the location should be updated with the movement of mechanic's current location. Google Maps API and the location-based services of mechanic's and vehicle owner's mobile devices should be used for this.

- Completion of a service

Once the mechanic visits the location for the inspection, an inspection charge should be added to the Shopping Cart of vehicle owner as a pending payment. After the inspection, the mechanic can make a quotation using the Mobile Application itself. Once the mechanic confirms this quotation, it should be available at the vehicle owner's profile as pending quotations. If the vehicle owner confirms this quotation, then again it will be added to vehicle owner's shopping cart.

After the completion of a service, vehicle owner can make the payment either using a credit card or cash. The job will only be completed after mechanic confirms the successful retrieval of payment.

A payment gateway should be integrated with the portal. After analyzing the requirements, it was identified that PayHere is the ideal payment gateway to cater the requirements of the proposed system.

- Rating System

Upon the completion of each job, both vehicle owner and the mechanic can rate each other. This should be recorded with the unique identification of the Job and the username of the user. Maximum rating will be 5 stars while the minimum will be 1 star. Apart from that it is needed to have a review writing option upon the completion of each job.

The average value of this ratings should be calculated and displayed on the profile of each user.

- Activity Log

Whenever a user committing an action, that needs to be recorded in a separate table with the username, description of the action and the timestamp.

Administrators should be able to query these user logs when required.

4 System Design

4.1 System Architecture and Architecture Design

This section mainly outlines the system and hardware architecture design of the system. Since, a website, a web application and web services for mobile application is also involved, a considerable attention has to be paid on system architecture design, hardware design, performance and security measures.

4.1.1 External Systems Diagram

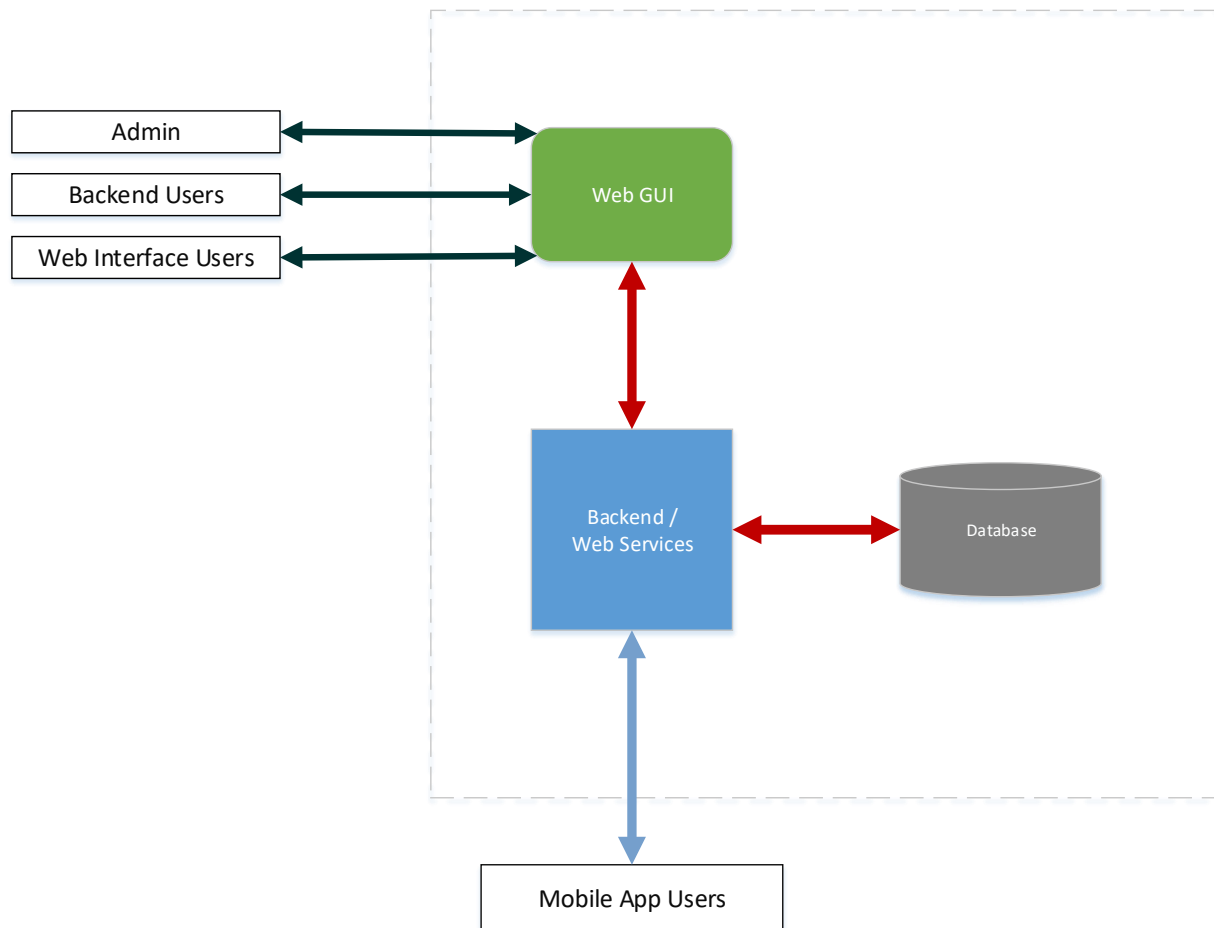


Figure 4-1 External Systems Architecture

4.1.2 Hardware Architecture

The proposed system will be deployed on a web server running Apache. However, ideal way to host the system would be using a cloud platform such as Amazon Web Services. After the research carried out on this matter and the discussions had with the client, it was decided to use Amazon EC2 cloud server instance as the hosting environment for the proposed system.

Technologies used for the proposed system are as below.

Table 1 - Technologies used for Development

Server Operating System	Ubuntu Server 18.04	A Canonical and open source community developed operating system, that works with almost all hardware platforms and which has the capability to serve up web applications, websites, file servers as well as your presence on cloud.
Web Server	Apache	An Open Source web server which is being used by the majority of web services and web applications on internet. Also, it has the ability and flexibility of customizing to cater different environments with plugins and addons.
Programming Languages	PHP	PHP will be used as the main server-side development language throughout the development of proposed system. PHP 7.0 will be used for the development with Codeigniter framework.
	HTML, CSS, JS	For frontend development, HTML, CSS and JS will be used along with Bootstrap Framework
	ReactJS / ReactNative	For the development of Mobile Application, React JS and ReactNative frameworks will be used. ReactNative is a framework developed by facebook based on ReactJS and currently being used by giants in the industry for their mobile apps such as Uber, Facebook, Instagram etc. [3,4]

Database Management System	MariaDB	The community developed and open source fork of MySQL which is intended to be remain free and open source under the GNU General Public License. MariaDB has some enhanced features to ensure security, performance and reliability than MySQL.
Version Controlling	GIT	A version control system that is being used for software versioning and other version controlling requirements. It is a free software that is being distributed under GNU General Public License v2.
GIT Hosting	Github	<p>A web-based Git repository hosting service, which has graphical tools to perform GIT version controlling tasks. Github also provides access controlling features as well as advanced collaboration tools that are important for collaborative software development methodologies.</p> <p>Github has a free plan which offers both private and public repositories. For this project a Github private repository will be used.</p>

4.1.3 Security Hardware Architecture

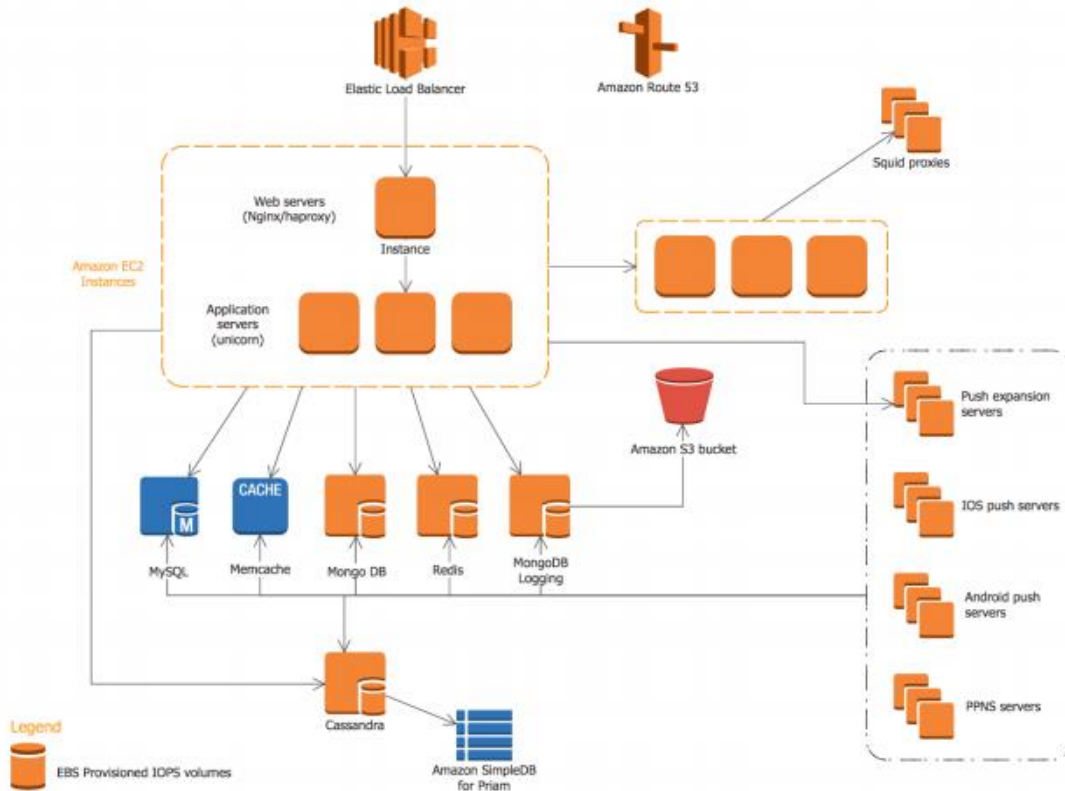


Figure 4-2 AWS Security Architecture [6]

4.1.4 Performance Hardware Architecture

The current and proposed solution utilizes AWS EC2 for hardware performance and reliability. Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers [6].

Since the proposed system starts with minimum number of users and requests for its web services, client wish to keep the investment for servers at a minimum level, yet expandable when required. Also, due to the usage of Web Services by the mobile application, uptime of the server is a vital factor. Due to all these facts, it was decided to select AWS EC2 as the cloud hosting environment [6].

4.1.5 Software Architecture

For user-facing applications like this, the three-tier architecture is a popular pattern. The tiers that comprise this architecture include the presentation tier, the logic tier, and the data tier. The data tier consists of storage media (databases, object stores, caches, file systems, etc.) that hold the data relevant to the application. The logic tier contains the code required to translate user actions at the presentation tier to the functionality that drives the application's behavior. . The presentation tier represents the component that users directly interact with (such as a web page, mobile app UI, etc.).

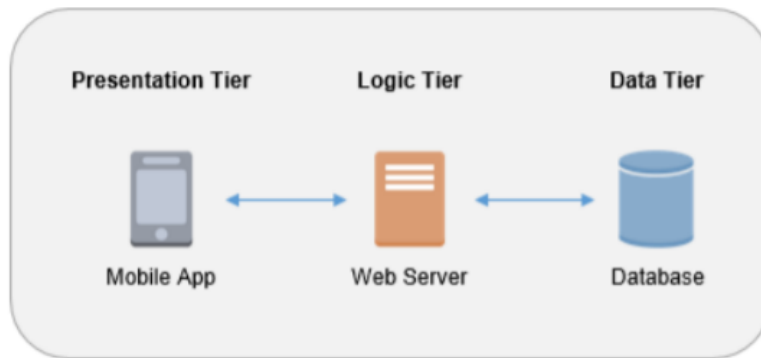


Figure 4-3 - 3 Tier Architecture

4.2 Database Design

As mentioned in earlier chapters, initially it was decided to use NoSQL Mongo DB as the database management system of the proposed system. However, due to some technical and software architectural constraints, finally it was decided to go forward with a relational database system. Since, the system is mainly developed with the use of Open Source technologies and due to the flexibility, MySQL (MariaDB) was selected as the database management system.

4.2.1 Entity Relationship Diagram

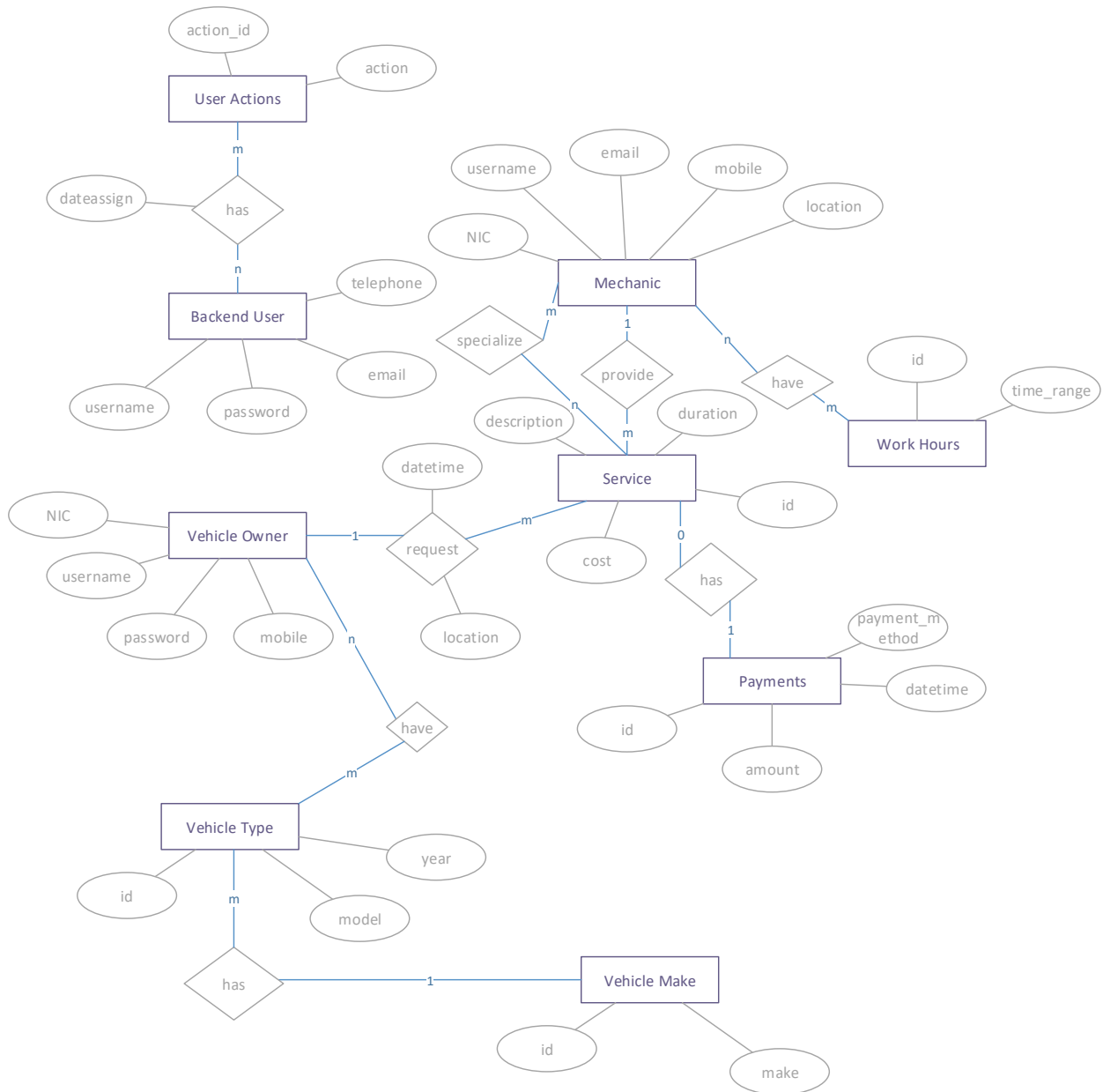


Figure 4-4 Entity Relationship Diagram

4.2.2 Database Table Structure

After designing above Entity Relationship Diagram, it was mapped to a Relational Database structure as below.

BackendUser (username, password, email, telephone)

Actions (action_id, action)

User_Actions (username, action_id, date_assign)

VehicleOwner (username, password, email, telephone, address1, address2, city, country, zipcode, NIC)

Mechanic (username, password, email, telephone, address1, address2, city, country, zipcode, NIC, location)

WorkHours (id, time_range)

Vehicle_Type(id, make_id, Model, Year)

Vehicle_Make(id, make)

VehicleOwner_Vehicle(owner_username, vehicletype_Id, VIN, RegNo, Mileage)

Specialize (mechanic_username, service_id)

Service (id, description, cost, duration)

Service_Request (vehicleowner_username, service_id, datetime, location, rating)

Service_Provide (mechanic_username, service_id, datetime, location, rating)

Payments (payment_id, amount, datetime, service_id)

4.3 Use Case Diagram

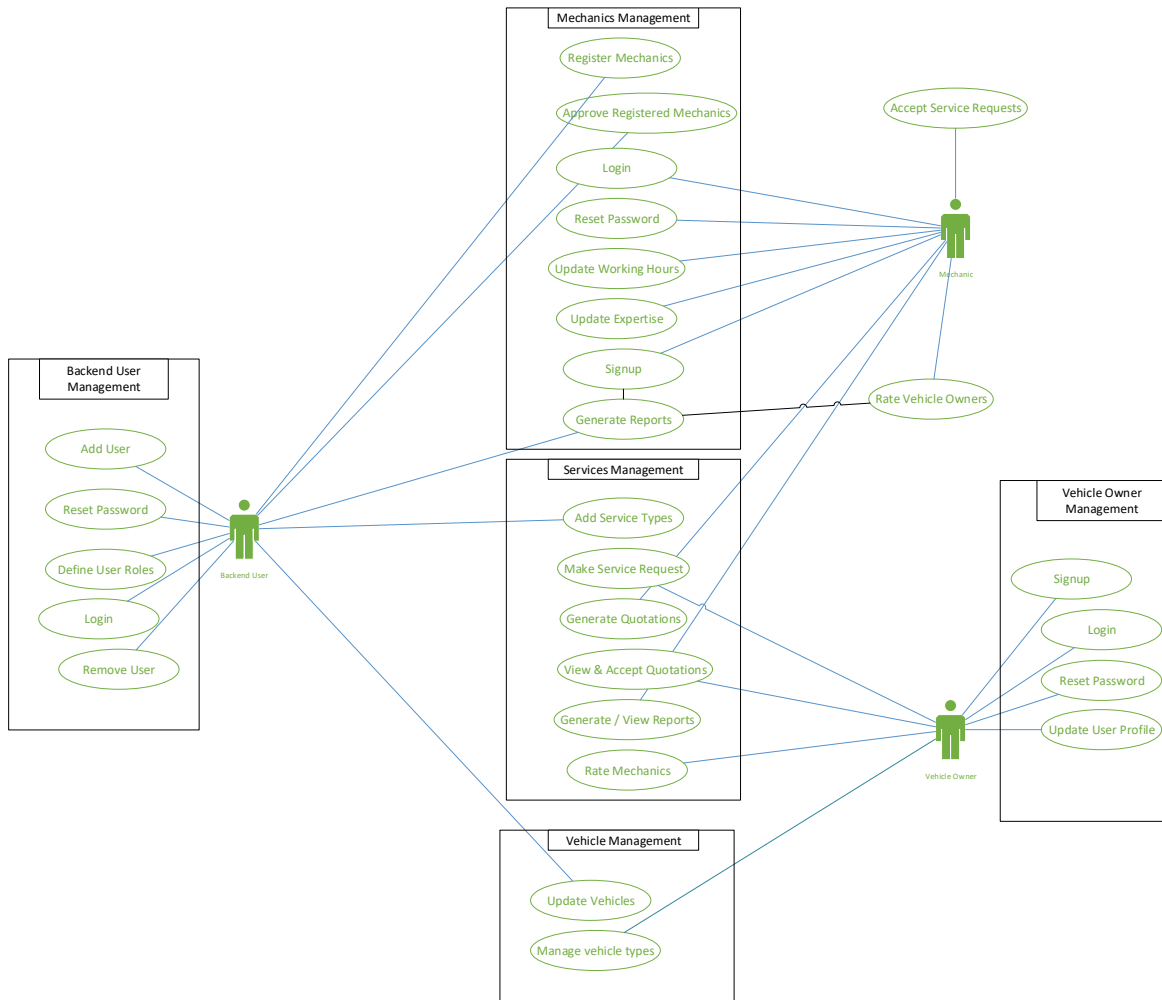


Figure 4-5 Use Case Diagram

4.4 User Interface Design

User interface is the front-end application view to which user interacts in order to use the software. User can manipulate and control the software as well as hardware by means of user interface. Today, user interface is found at almost every place where digital technology exists, right from computers, mobile phones, cars, music players, airplanes, ships etc.

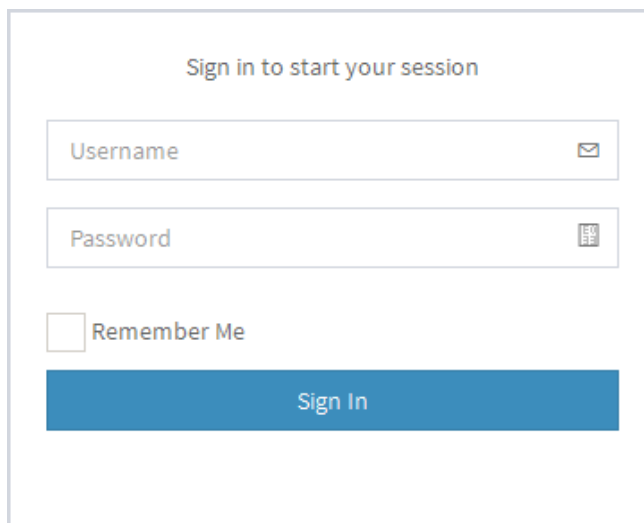
Mainly, there will be two User Interface design instances for the Web Portal and the Mobile Application. In Web Portal also, there will be two User Interfaces. One is for the Customers and Mechanics, while other one is for the Backend Users.

These User Interfaces were designed using Adobe XD and will be implemented with Bootstrap framework in the web portal and React Native framework in the Mobile Application.


4.4.1 User Interface of Backend Users / Admin Panel


The Backend / Admin portal of the web application will mainly be used for administrative purposes. Admin users can add new users, new vehicle models, review orders and jobs, approve new user registrations and generate reports through this admin portal.

Standard elements of Bootstrap framework were used to design the user interface of admin portal with some customizations. For reports and graphs, Chart.js library's elements were used. Also, standard colors and icons were used for alerts, message boxes, error messages etc. to enhance the user experience throughout the admin portal.



Sign in to start your session

Username 

Password 

Remember Me

Sign In

AdminLTE Admin

Add Customer Details Home > Customer > AddNew

Admin
Online

- Customer Details
- View All
- Add New
- Services
- Car Type
- Role Management
- Promocode
- Mechanical Details
- Backend User
- Services Area Details
- Settings

Add Customer Details

First Name

Last Name

Email address

Phone

State

City

Zip

Username

Password

AdminLTE Admin

Services Details Home > Services > ViewAll

Admin
Online

- Customer Details
- Services
- View All
- Add New
- Car Type
- Role Management
- Promocode
- Mechanical Details
- Backend User
- Services Area Details
- Settings

Services Details

Show entries Search:

Name	Hour	Level	Service Charge	Description	Action
AC (Air Conditioner) is not working	1	standard	300		Edit Delete
ABS (Anti-lock Braking System) light is on	1	advanced	1000	ABS (Anti-lock Braking System) light is on	Edit Delete
Air bag light is on	1	standard	0		Edit Delete
Air is not coming out of vents	1	standard	0	fdffdsf	Edit Delete
Battery light is on	1	standard	0		Edit Delete
Brake pedal is hard to push	1	standard	0		Edit Delete
Brake pedal vibrates or shakes	1	standard	0		Edit Delete
Brake warning light is on	1	standard	0		Edit Delete
Brake light is not working	1	standard	0		Edit Delete

Admin LTE Admin

View Car Type Details Home > CarType > ViewAll

Admin
Online

- Customer Details
- Services
- Car Type
- View All
- Add New
- Role Management
- Promocode
- Mechanical Details
- Backend User
- Services Area Details
- Settings
- Rejected Order Details

View Car Type Details

Show entries Search:

Name	Make Year	Car Model	Car Engine	Action
ACURA	2001	ACURA GUD	E11158	Edit Delete
AUDI	2000	dfds	W222	Edit Delete
BENTLEY	2000	dsfd	R333	Edit Delete
BMW	2002	dsf	S333	Edit Delete
BUICK	2003	dfdsf	Y555	Edit Delete
CADILLAC	2004	dfgdf	Y555	Edit Delete
CHECKER	2005	Y555	Y555	Edit Delete
CHEVROLET	1989	Y555	Y555	Edit Delete
CHEVY MD TRK	1989	Y555	Y555	Edit Delete
CHRYSLER	1989	Y555	Y555	Edit Delete

4.4.2 User Interface of the Web Portal

Web portal is the default portal a user would see when he/she visits the website of the proposed system. There will be sign in and sign up screens for Mechanics and Customers. Also, standard screens such as Forget Password, About Page, Services Page etc.

Once a user is signed in, according to his role within the system (either a vehicle owner or Mechanic), the user will be taken into a portal. Customers will be facilitated with below screens.

- View Vehicles
- Add Vehicle
- View Past Orders
- Make a new order
- Reviews

Mechanic / mechanic will be facilitated with below screens.

- Complete profile.
- Set skills
- Upload documents
- Received orders
- Work hours

However, screens such as Password reset, profile settings will be common to both type of users. User interface of web portal was designed from scratch, paying attention to color schemes, fonts, message boxes, alerts and layouts. This user interface will be responsive through the range of target devices.

The image displays two parts of the web portal's user interface. The left part is a login/sign-up form with a red header containing 'Sign in' and 'New account' buttons. Below the header, there are radio buttons for 'Customer' (selected) and 'Mechanic'. The form includes a text input field with 'admin', a password field with a 'Show' button, a 'Remember me' checkbox, and a large red 'Login' button. The right part shows a service selection process with a progress bar at the top containing 'SELECT CAR' (active), 'CHOOSE SERVICE', and 'ESTIMATE'. Below the progress bar, there are four dropdown menus: 'Year' (with 'Select' as the current selection), 'Choose Make' (with 'Select an option'), 'Choose Model' (with 'Select an option'), and 'Choose Engine' (with 'Select an option'). A 'Zip Code' input field with 'zip code' as the placeholder is also present. A red 'NEXT' button with a right-pointing arrow is located at the bottom right of the selection area.

ENTER DETAILS

CHOOSE SERVICE

PACKAGE

Select Type
 Mechanic Service station

Full Name
First name Last name

Email ID

Address
Address
Street name

Select an option City

Zip code Country

Cell Phone **Year of Experience**
Phone Number Experience

Diagnostic lead time **Working radius**
0hr

Cell phone Type **User Name**
 Iphone Android Username

Password **Confirm Password**
Password Confirm Password

NEXT

Sign in

New account

Name

Zip Code

Username

E-mail

Password Show

Confirm Password Show

I agree to the Terms

Create account

4.4.3 User Interface of Mobile App

Initially, the Mobile Application was intended to be developed for both Customers and Mechanics. But, due to the timeline constraints and technical difficulties, it was decided by the client to limit the usage of Mobile Application only for the Customers.

Following screens will be mainly available for the Customers in the mobile app.

- View Vehicles
- Add Vehicle
- View Past Orders
- Make a new order
- Reviews
- Profile settings

User interface of mobile application was designed from scratch, paying attention to color schemes, typography, message boxes, alerts and layouts. This user interface will be responsive through the range of target devices. Since, it might be used on a range of devices from entry level smart phone with smaller screen to a tablet device, spacing among buttons, menu items etc. were thoroughly considered.



Figure 4-6 - User Interface of Mobile App

5 Implementation

Implementation / coding of the system was carried out under several phases. Under the first phase, Admin panel was implemented. Second phase was the implementation of Web portal / web site and finally the implementation of the Mobile Application. Since the system was planned to develop adhering to the 3-tier architecture using Codeigniter framework, Codeigniter 3.3 was installed in a local server and following were configured.

- Database connection
- Base URLs and required library autoloads.
- Folder structure for the application.

5.1 Implementation of Admin Panel

It was decided to have separate Controllers and Models in Codeigniter for the admin panel. Models were created according to the UML class diagram defined under Chapter 4.

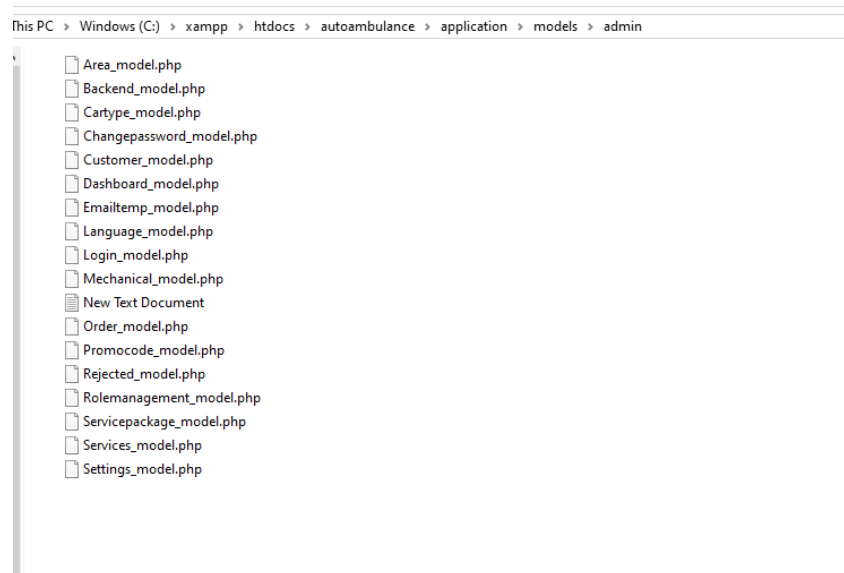


Figure 5-1 - Admin Panel Models

Next, Controllers were created for the admin panel. However, due to some technical difficulties aroused, Controllers of Admin Panel were not differentiated and put into separate folders like it was done for Models.

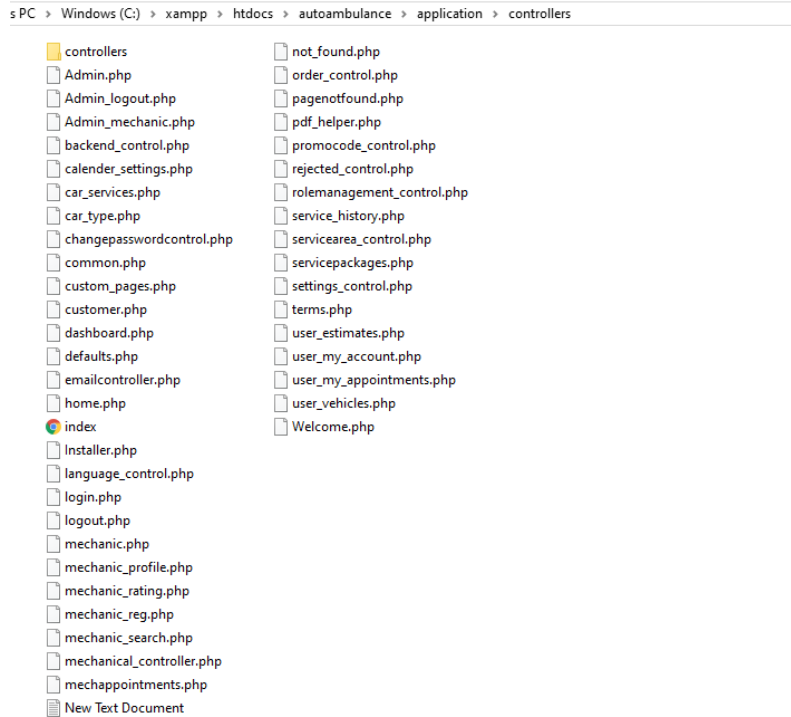


Figure 5-2 - Controllers of Admin Panel

First, the coding of Models and Controllers were done according to the basic CRUD (Create, Read, Update Delete) under those classes. Afterwards, Views for each controller method were developed according to the requirements.

```

<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class cartype_model extends CI_Model {

    public function _construct(){
        parent::_construct();
    }

    public function view_car_type()
    {
        $this->db->select('ot_car_type.id as id, ot_car_engine.car_type_id as id, ot_car_type.name, ot_car_type.mak
        $this->db->from('ot_car_type');
        $this->db->join('ot_car_makes', 'ot_car_type.id = ot_car_makes.car_type_id','left');
        $this->db->join('ot_car_engine', 'ot_car_makes.id = ot_car_engine.car_make_id','left');
        $this->db->group_by("ot_car_type.id");

        $query = $this->db->get();
        $result = $query->result();
        return $result;
    }

    public function edit_user_vehicles($request)
    {
        $table = 'ot_customer_vehicle';
    }
}

```

Figure 5-3 - CRUD Operations in Car Type Model

```

<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class car_type extends CI_Controller {

    public function __construct(){
        parent::__construct();

        date_default_timezone_set("Asia/Colombd");

        $this->load->model('admin/cartype_model');
        if(!$this->session->userdata('logged_ins'))
        {
            redirect(base_url());
        }
    }

    public function cartype_view()
    {
        $template['page'] = 'Admin/cartype/view-cartype';
        $template['page_title'] = "View Cartype";
        $template['data'] = $this->cartype_model->view_car_type();
        $this->load->view('admin-template',$template);
        $this->load->view('Templates/admin/custom-datatable');
    }
}

```

Figure 5-4 - Controller for Car Type

View

his PC > Windows (C:) > xampp > htdocs > autoambulance > application > views > Admin > cartype

Name	Date modified	Type	Size
cartype-add.php	11/22/2018 11:02 PM	PHP File	4 KB
edit-cartype.php	11/22/2018 11:02 PM	PHP File	4 KB
view-cartype.php	11/22/2018 11:02 PM	PHP File	3 KB

Figure 5-5 - Views for CRUD Operations in Admin Panel Car Types

```

<div class="alert alert-<?php echo $message['class']; ?>">
<button class="close" data-dismiss="alert" type="button">x</button>
<?php echo $message['message']; ?>
</div>

<?php
}
?>

<section class="content-header">
    <h1>
        View Car Type Details
    </h1>
    <ol class="breadcrumb">
        <li><a href="#"><i class="fa fa-car"></i> Home</a></li>
        <li><a href="#">CarType</a></li>
        <li class="active">ViewAll</li>
    </ol>
</section>

<!-- Main content -->
<section class="content">
    <div class="row">
        <div class="col-xs-12">

            <div class="box">
                <div class="box-header">

```

Figure 5-6 - View Car Types view

As decided in Chapter 4.5.1 User Interface of Backend Users / Admin Panel, user interface for Admin Panel was implemented with Bootstrap 3 framework. To save time when it comes to development, rather than implementing the UI from scratch, SBAdmin bootstrap template was used. A custom stylesheet was also used to make changes when required. JQuery were also used to facilitate form validations and AJAX. Standard JQuery plugins, such as JQuery date time picker, JQuery Block UI and JQuery Alerts were used for the implementation.

5.2 Implementation of Web Portal

First, the User Interface mentioned in Chapter 4.5.2 was implemented based on Bootstrap framework and custom stylesheets. This implementation was done inside Views folder of Codeigniter. Separate template files were generated from Header and Footer sections of the web portal to include it on top and bottom of each page of the portal.

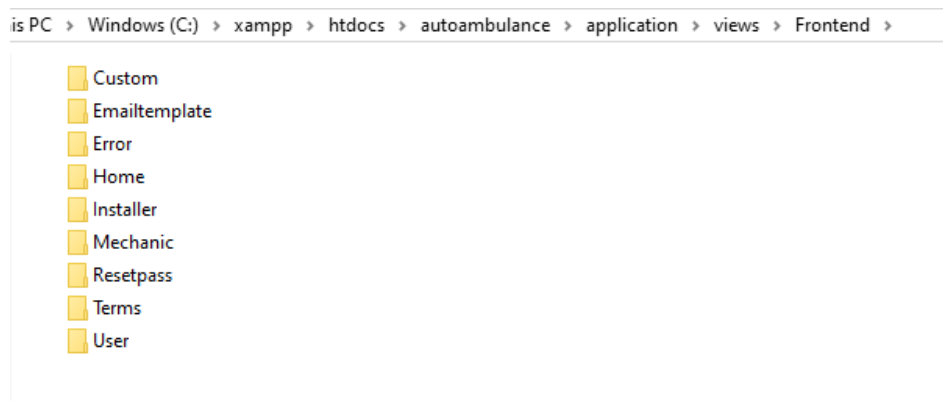


Figure 5-7 - Folder structure of Web Portal pages

```

<div style="overflow:hidden;">
  <div class="auto-title-bar">
    <div class="col-lg-1"></div>
    <div class="col-lg-6">
      <?php
        include 'includes/en_lang.php';
        ?>
      <div class="auto-title-selector">
        <div class="auto-select-inner">
          <ul class="tabs">
            <li class="tab-link current tab-1" data-tab="tab-1"><?php echo $SELECT_CAR; ?></li>
            <li class="tab-link tab-2" style="left:31%;" data-tab="tab-2"><?php echo $CHOOSE_SERVICE; ?>
            <li class="tab-link tab-3" style="left:62%;" data-tab="tab-3"><?php echo $ESTI_MATE; ?></li>
          </ul>
          <div id="tab-1" class="tab-content choosing_car current ">
            <div class="row">
              <div class="col-md-6">
                <div class="ac-select-item">
                  <?php echo $years;?>
                  <br>
                  <select class="select2 margin-t10" id="input" onchange="selectdata()" name="fxs">
                    <option value="0" selected="">Select</option>
                    <option value="fxHurl">1999</option>
                    <option value="fxVacuum">2000</option>
                    <option value="fxShinkansen">2001</option>
                    <option value="fxSnake">2002</option>

```

Figure 5-8 - Home Page of Web Portal

Also, a separate set of Controllers and Models were implemented for the web portal based on the functions that will be performed within the Web Portal.

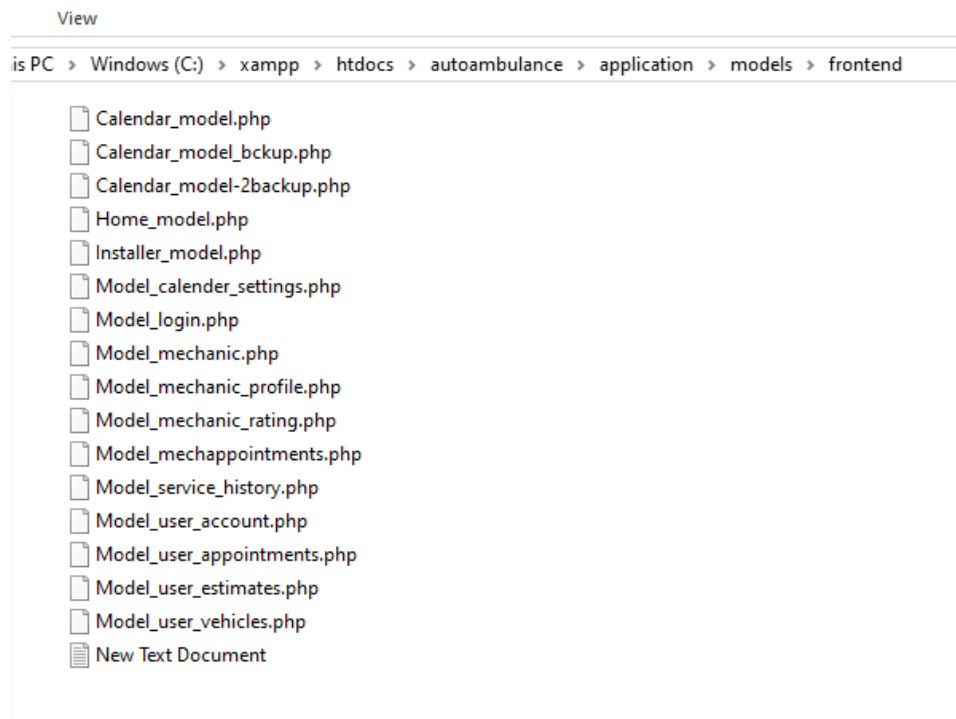


Figure 5-9 - Models for Web Portal

To enhance the User Friendliness, to provide on page validations without reload and to enable on page form submission, AJAX form submits were implemented based on JQuery plugin. Also, wizard type forms were used rather than making users scroll throughout the page for form completion. These wizards were implemented for following sections,

- Customers to make mechanic requests
- Customers to add vehicles for his/her profile
- Mechanic to complete his/her profile.

```

function button() {

    var value = [];

    $.each($("#input[name='marked']:checked"), function(index) {
        value[index]=$ (this).val();
    });
    var newOption = " ";

    $.ajax({
        type: 'POST',
        url: '<?php echo base_url();?>Home/get_estimate',
        data:{"data":value},
        success: function (result) {
            var items = JSON.parse(result);
            var value = JSON.stringify()
            var sum = 0;
            $(items).each(function () {
                sum += parseFloat(this.service_charge);
            });
            // newOption += <div class="service-estimate"><div class="services"></div><div class="est-amnt"> </div> <
            newOption += '<div class="service-estimate"><div class="services">' + this.name + '</div><div class="est-am
        });
    });
}

```

Figure 5-10 - JQuery AJAX requests

Dynamic pages and static pages were separately developed as there were no models involved with static pages.

5.3 Implementation of Mobile App

First the user interface defined on Chapter 4 was implemented with the components of Ionic Framework and UI files were saved under templates folder. CSS was used for the styles of Mobile Application.

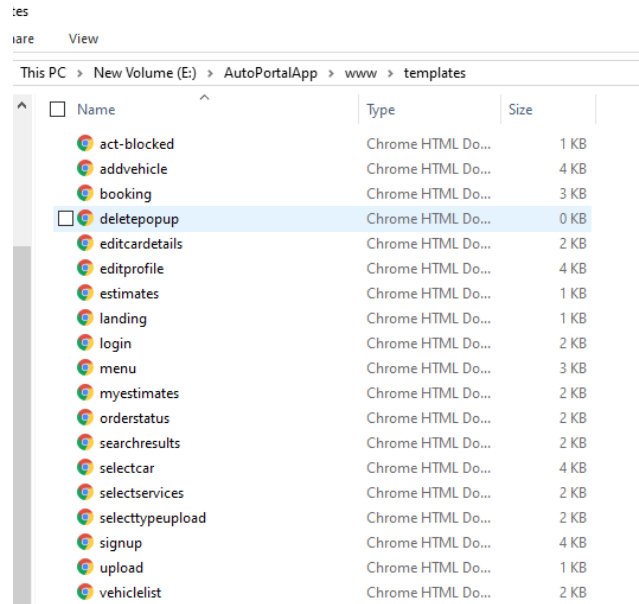


Figure 5-11 - User Interface files

For the frontend development and to enable the functionality of the mobile application with the server-side, Javascript ES6 was used.

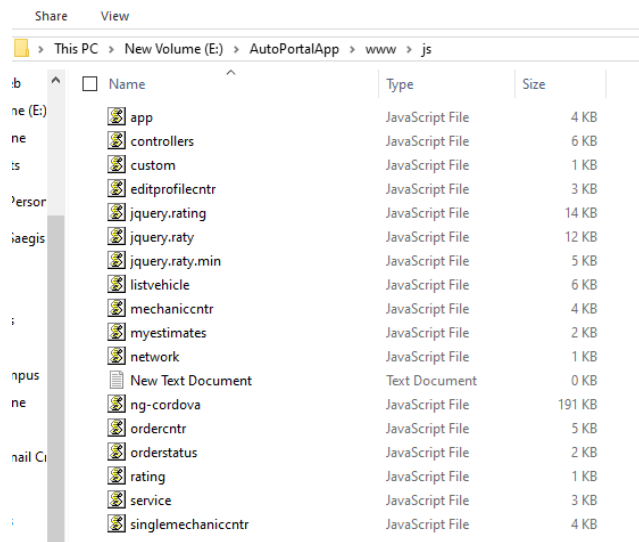


Figure 5-12 - Frontend JS files

Controller.js file was used to implement basic navigation within the application as well as to enable user login and registration processes. Server-side web services are requested by these JS files through AJAX requests.

```
var App = angular.module('starter.controllers', ['ngCordova']);
App.controller('AppCtrl', function($scope,$http, $ionicModal,$rootScope, $location,$timeout,$state,$window,$ionicLc

    post_data ={'my_key': my_key};
    link="webservice/get_app_details";

    WebService.show_loading();
    var promise = WebService.send_data( link,post_data);
    promise.then(function(response) {

        $timeout(function() {
            $ionicLoading.hide();
        },1000);
        var status = response.status;
        console.log(status);
        if(status=='failed'){

            $window.location.href = '#/app/actblocked';
            console.log('bmd-blocked');
        }
    });
// With the new view caching in Ionic, Controllers are only called
// when they are recreated or on app start, instead of every page change.
// To listen for when this page is active (for example, to refresh data),
// listen for the $ionicView enter event:
```

Figure 5-13 - Controller.js

Similarly, separate JS files were created for the functionality of each page defined under templates folder. As an example, to list down the vehicles added to the profile of each user listvehicles.js was coded.

```
});
car_Details();
function car_Details(){
    $rootScope.vehicleDetails='';
    $rootScope.user_data = JSON.parse( localStorage.getItem('user_data') );

    post_data ={"customer_id": $rootScope.user_data.id};
    link="webservice/getUserVehicleDetails";

    WebService.show_loading();
    var promise = WebService.send_data( link,post_data);
    promise.then(function(data){

        // $state.go('app.vehicle');

        if(data!=''){
            $ionicLoading.hide();

            $rootScope.vehicleDetails=data;

        }else{

            $ionicLoading.hide();

        }
    });
}
```

Figure 5-14 - List Vehicles

Also, similar to the web portal, wizard type forms were used rather than making users scroll throughout the page for form completion. These wizards were implemented for following sections,

- Customers to make mechanic requests
- Customers to add vehicles for his/her profile

6 Testing and Evaluation

6.1 Testing

As defined earlier, for the development of this system, Agile Development methodology was adopted with the use of Extreme Programming (XP). At Extreme Programming, acceptance criteria of each iterative cycle are defined earlier. Therefore, unit tests of each iteration / module have been done at the time of the completion of each module.

Tests done at the end of each iteration were mostly into White Box Testing. Rather than testing the system with sample test data without considering the internal structures, the internal structures, classes, methods, objects and user interfaces were tested for errors. However, at this point some test data were also used to verify the functional and non-functional requirements of the system.

After completion of all required modules, a Black Box testing process was carried out throughout the entire system. This can be identified as an integration test of the whole system too. This process was divided into several phases and carried out. These test cases are available in Appendices.

6.2 Evaluation

Main objective of this project was to implement an intermediate platform to interconnect customers and mechanics. The system was also given to few potential customers and mechanics to test with actual data. Through this potential user testing, most of the functions of the system were proven to be working without major problems. However, there were some additional feature requests from potential customers too.

Initially, it was decided to develop the mobile application for both customers and mechanics. Due to timeline constraints and technical constraints, under the first phase of the development, mobile application was developed only for the customers. During the development of mobile application, developer had to go through Web Services development on the server side of the application. Initially, this was accomplished through standard controller methods in Codeigniter. But soon after identifying security problems that might arise, rest controllers had been developed with GET, PUT and POST methods.

Also, as per the initial proposal provided to the client, mobile application was intended to be developed using ReactNative framework. Due to timeline and technical constraints, mobile application was developed using Ionic Framework. This was mainly due to the developer's familiarity with web technologies rather than mobile app

development technologies. Since Ionic Framework is based on HTML5, SCSS and Javascript (Angular JS), developer could easily focus on the development as it was much similar to the development of a web application.

During the planning stage of this project, Google Maps API was free to use till a certain number of API calls per month. However, due to change of business model in Google Maps API, client requested for an alternative solution other than going ahead with Google Maps API. Only other option that was available was Open Maps API. Since that is also not reliable as Google Maps API, it was decided to define service areas with Zip Codes in the system. Therefore, once a client requests a service, it was matched to near by mechanics using their Zip code.

Payment gateway integration was not successful as the client failed to get the support of a suitable payment gateway provider during the development span of the project. However, developer has created web services that are required for payment gateway integration and also, tested with Paypal payment gateway.

Even though the developer had to face challenges while the development is ongoing, the project ended up as a success and that was also endorsed by the feedback from potential users.

7 Conclusion

As the outcome of this year-long project, the web portal for vehicle owners and mechanics were successfully developed. However, the mobile application was not a considerable success as there might be some security problems and usability problems due to the hybrid nature of the application.

More attention could be given to the User Interface and User Experience of the developed system to enhance the user friendliness. However, the user interface is responsive throughout all standard screen sizes.

In the future, the mobile application can be extended to be used by the mechanics also. At the same time, Google Maps API integration would be a good step ahead as it might make the system more user friendly for potential users.

Also, using the data that is being collected while the system is being used by potential users, if the developer can implement data mining techniques throughout the usage data of users, it will support the client for decision making and future expansions. Also, this could be used to provide more personalized services for each user based on their own behavior.

8 References

[1]"CodeIgniter User Guide – CodeIgniter 3.1.11 documentation", Codeigniter.com, 2019. [Online]. Available: https://codeigniter.com/user_guide/index.html. [Accessed: 25- Feb- 2020].

[2]"Developers | Uber", Developer.uber.com, 2019. [Online]. Available: <https://developer.uber.com/docs/riders/introduction>. [Accessed: 24- Feb- 2020].

[3]"Getting Started · React Native", Reactnative.dev, 2020. [Online]. Available: <https://reactnative.dev/docs/getting-started>. [Accessed: 28- Feb- 2020].

[4]"An advanced guide on setting up a React and PHP web app", Medium, 2019. [Online]. Available: <https://medium.com/@davisonpro/an-advanced-guide-on-setting-up-a-react-and-php-web-app-acaedb21ab3a>. [Accessed: 23- Feb- 2020].

[5]"Creating a Store Locator on Google Maps | Store Locator Solution", Google Developers, 2017. [Online]. Available: <https://developers.google.com/maps/solutions/store-locator/clothing-store-locator>. [Accessed: 28- Feb- 2020].

[6]"AWS Documentation", Amazon Web Services Inc, 2019. [Online]. Available: <https://docs.aws.amazon.com/>. [Accessed: 28- Feb- 2020].

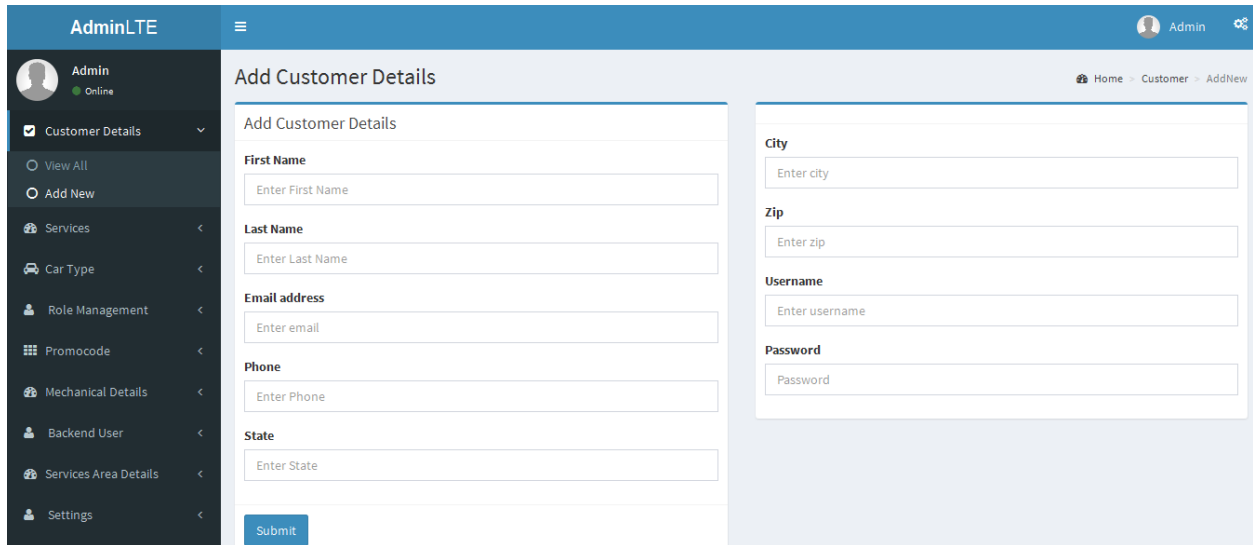
9 Appendices

9.1 User Manual

Admin Panel of the system can be accessed from `<url>/admin`. Enter your admin username and password can click on Log In. You will be redirected to the Dashboard of the Admin Panel.

Customer Details

Customer details can be added, modified and viewed by visiting Customer Details section from the Navigation menu. Customers can also register through the web portal (front end) of the system.



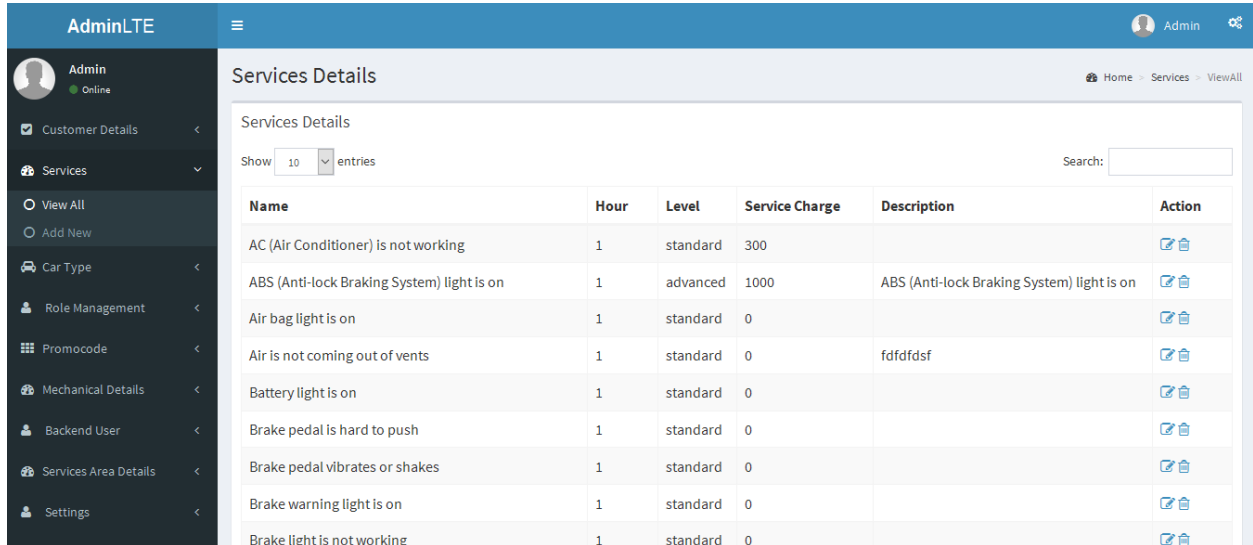
The screenshot displays the 'Add Customer Details' form within the AdminLTE interface. The form is titled 'Add Customer Details' and is located in the main content area. The left sidebar contains a navigation menu with the following items: Customer Details (checked), View All, Add New, Services, Car Type, Role Management, Promocode, Mechanical Details, Backend User, Services Area Details, and Settings. The form fields are as follows:

- First Name:** Enter First Name
- Last Name:** Enter Last Name
- Email address:** Enter email
- Phone:** Enter Phone
- State:** Enter State
- City:** Enter city
- Zip:** Enter zip
- Username:** Enter username
- Password:** Password

A 'Submit' button is located at the bottom left of the form. The top navigation bar shows 'AdminLTE' and 'Admin' with a profile icon. The breadcrumb trail at the top right reads 'Home > Customer > AddNew'.

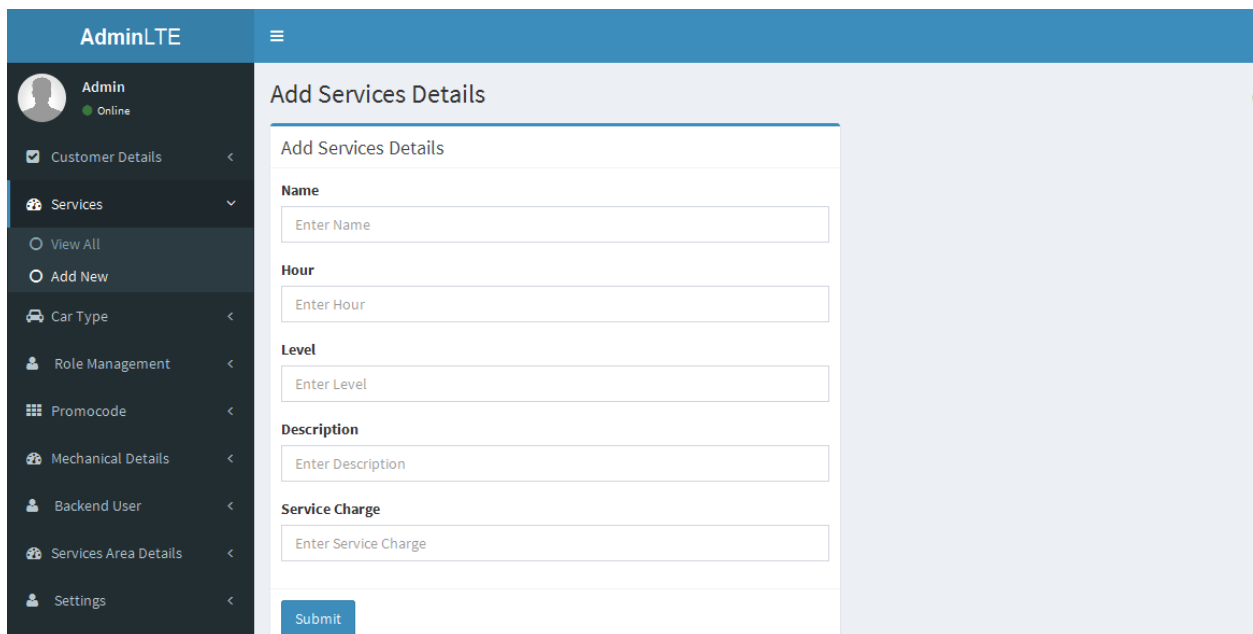
Services

Services can be added, modified, viewed or deleted from the system by an admin using the Services section.



The screenshot shows the 'Services Details' page in AdminLTE. The page features a sidebar with navigation options and a main content area with a table of service records. The table has columns for Name, Hour, Level, Service Charge, Description, and Action. The data rows include various car issues and their associated charges and levels.

Name	Hour	Level	Service Charge	Description	Action
AC (Air Conditioner) is not working	1	standard	300		Edit Delete
ABS (Anti-lock Braking System) light is on	1	advanced	1000	ABS (Anti-lock Braking System) light is on	Edit Delete
Air bag light is on	1	standard	0		Edit Delete
Air is not coming out of vents	1	standard	0	fdffdfs	Edit Delete
Battery light is on	1	standard	0		Edit Delete
Brake pedal is hard to push	1	standard	0		Edit Delete
Brake pedal vibrates or shakes	1	standard	0		Edit Delete
Brake warning light is on	1	standard	0		Edit Delete
Brake light is not working	1	standard	0		Edit Delete



The screenshot shows the 'Add Services Details' form in AdminLTE. The form is a vertical stack of input fields for Name, Hour, Level, Description, and Service Charge, followed by a Submit button.

Add Services Details

Name

Hour

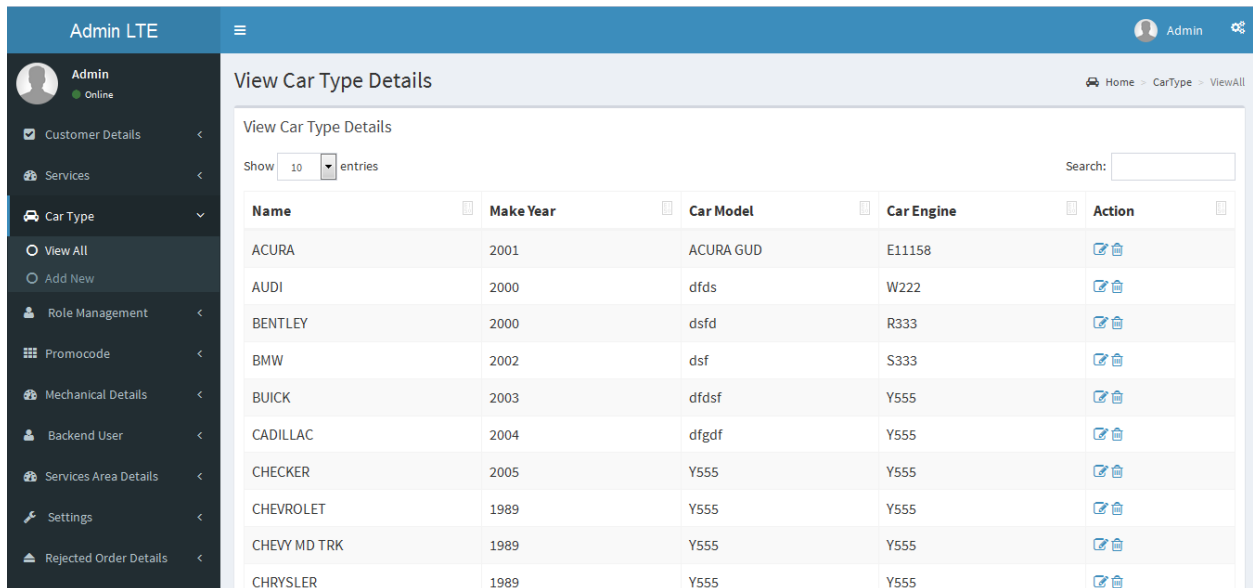
Level

Description

Service Charge

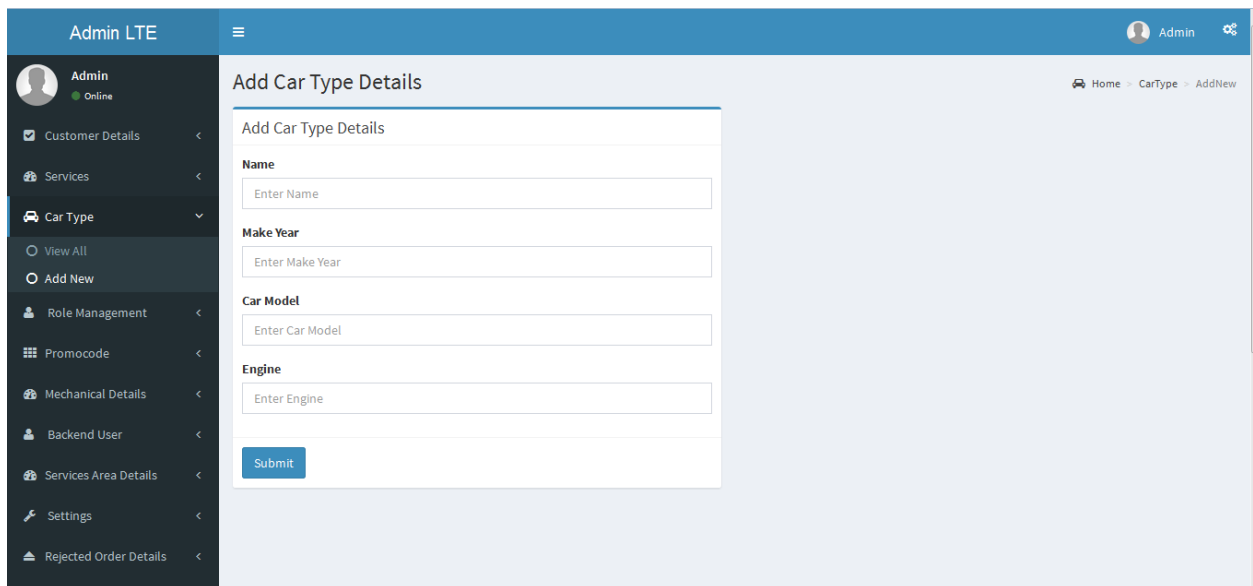
Vehicle Types /Car types

Vehicle types, engines and specifications of those vehicles can be managed in the system by visiting the Car Type section in the navigation menu.



The screenshot displays the 'View Car Type Details' page. The left sidebar contains a navigation menu with 'Car Type' selected. The main content area features a table with the following data:

Name	Make Year	Car Model	Car Engine	Action
ACURA	2001	ACURA GUD	E11158	Edit Delete
AUDI	2000	dfds	W222	Edit Delete
BENTLEY	2000	dsfd	R333	Edit Delete
BMW	2002	dsf	S333	Edit Delete
BUICK	2003	dfdsf	Y555	Edit Delete
CADILLAC	2004	dfgdf	Y555	Edit Delete
CHECKER	2005	Y555	Y555	Edit Delete
CHEVROLET	1989	Y555	Y555	Edit Delete
CHEVY MD TRK	1989	Y555	Y555	Edit Delete
CHRYSLER	1989	Y555	Y555	Edit Delete



The screenshot displays the 'Add Car Type Details' page. The left sidebar contains a navigation menu with 'Car Type' selected. The main content area features a form with the following fields:

- Name:** Enter Name
- Make Year:** Enter Make Year
- Car Model:** Enter Car Model
- Engine:** Enter Engine

A 'Submit' button is located at the bottom of the form.

Mechanical Details

Mechanical Details is for admin to view, add and manage the details of the Mechanic in the backend.

The screenshot shows the 'Add Mechanical Details' form in the AdminLTE dashboard. The form is titled 'Add Mechanical Details' and contains several input fields for user information. The left sidebar shows the navigation menu with 'Mechanical Details' selected. The top header shows 'AdminLTE' and the user 'Admin'.

Field	Input
First Name	Enter First Name
Last Name	Enter Last Name
Email address	Enter email
Mobile	Enter Mobile
Cellphone type	Enter LandLine
Lead Time	Enter Lead Time
State	Enter State
City	Enter city
Zip	Enter zip
Experience	Enter Experience
Criminal Case	Enter Criminal Case
Review Status	Enter Review Status

Backend Users

Backend users are those who have access to edit and update the dashboard. Capabilities of these backend users can be defined through User Roles section in the Navigation menu.

The screenshot shows the 'View Backend User Details' page in the AdminLTE dashboard. The page displays a table of backend users with columns for Username, Email, mobile, Role, and Action. The left sidebar shows the navigation menu with 'Backend User' selected. The top header shows 'AdminLTE' and the user 'Admin'.

Username	Email	mobile	Role	Action
admin	admin@gmail.com	8976456377	admin	Edit Delete
qw	immanu34@gmail.com	8976456372	user	Edit Delete
gjinu	gjinu@gmail.com	8976456371	user	Edit Delete
jjinu	jjinu@gmail.com	8976456379	user	Edit Delete
nittu	nittu@gmail.com	52637489	user	Edit Delete

9.2 Test Cases

Test Scenario	Login-1			Test Case ID	Login-1A	
Test Case Description	Login - Positive test case			Test Priority	High	
Pre-Requisite	A valid admin user account			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Launch admin panel	http://localhost/autportal/admin	Admin portal login page	Admin portal login page	Chrome	Pass
2	Enter correct Username & Password	Username: admin Password: Admin@123	Login success. View Dashboard	Login success. View Dashboard	Chrome	Pass

Test Scenario	Login-1			Test Case ID	Login-1B	
Test Case Description	Login - Negative test case			Test Priority	High	
Pre-Requisite	An invalid user account			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Enter correct Username & Password	Username: admin Password: password	Login error. View error message	Login error. View error message	Chrome	Pass

Test Scenario	Change Password - 1			Test Case ID	ChangePW-1A	
Test Case Description	Change Password - Positive Test Case			Test Priority	High	
Pre-Requisite	User logged in successfully			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result

1	Click on User Menu in Navigation bar	-	User Menu	User Menu	Chrome	Pass
2	Click on Change Password button	-	Change Password page	Change Password page	Chrome	Pass
3	Enter current password and new password twice	Current Password: Admin@123 New Password: NewPassword23 Confirm Password: NewPassword23	Password changed	Password changed	Chrome	Pass

Test Scenario	Change Password - 1	Test Case ID	ChangePW-1B
Test Case Description	Change Password - Negative Test Case	Test Priority	High
Pre-Requisite	User logged in successfully	Post-requisite	NA

Test Execution Steps:

S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click on User Menu in Navigation bar	-	User Menu	User Menu	Chrome	Pass
2	Click on Change Password button	-	Change Password page	Change Password page	Chrome	Pass
3	Enter incorrect current password and current password twice	Current Password: Admin New Password: NewPassword23 Confirm Password: NewPassword23	Error message: Invalid current password	Error message: Invalid current password	Chrome	Pass
4	Enter in current password and unmatched new password twice	Current Password: Admin New Password: NewPassword23 Confirm Password: NewPassword24	Error message: New passwords don't match	Error message: New passwords don't match	Chrome	Pass

Test Scenario	Logout - 1			Test Case ID	Logout-1A	
Test Case Description	Logout - Positive Test Case			Test Priority	High	
Pre-Requisite	User logged in successfully			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click on User Menu in Navigation bar	-	User Menu	User Menu	Chrome	Pass
2	Click on Sign Out button	-	User logout & Web portal home	User logout & Web portal home	Chrome	Pass

Test Scenario	View Customers - 1			Test Case ID	View_customers-1A	
Test Case Description	View Customers - Positive Test Case			Test Priority	High	
Pre-Requisite	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Customer details on navigation menu	-	Expanded customer details menu	Expanded customer details menu	Chrome	Pass
2	Click View all	-	View Customer Details	View Customer Details	Chrome	Pass
3	Filter Customers	Enter a text to search	Filtered results according to text entered	Filtered results according to text entered	Chrome	Pass

Test Scenario	Add Customers - 1			Test Case ID	Add_customers-1A	
----------------------	-------------------	--	--	---------------------	------------------	--

Test Case Description	Add customers - Negative Test Case	Test Priority	High			
Pre-Requisite	User logged in successfully and have relevant user role	Post-requisite	NA			
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Customer details on navigation menu	-	Expanded customer details menu	Expanded customer details menu	Chrome	Pass
2	Click Add New	-	Add Customer Details	Add Customer Details	Chrome	Pass
3	Click submit without completing form	Input - none	Display error messages under each form field	Display error messages under each form field	Chrome	Pass
4	Click submit with incorrect data	Input - incorrect data	Display error messages under incorrect data entered fields	Display error messages under incorrect data entered fields	Chrome	Pass

Test Scenario	Add Customers - 1	Test Case ID	Add_cust omers-1B			
Test Case Description	Add customers - Positive Test Case	Test Priority	High			
Pre-Requisite	User logged in successfully and have relevant user role	Post-requisite	NA			
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Customer details on navigation menu	-	Expanded customer details menu	Expanded customer details menu	Chrome	Pass
2	Click Add New	-	Add Customer Details	Add Customer Details	Chrome	Pass

3	Complete form and click submit	Input: Fill all form fields with relevant information	Customer added successfully	Customer added successfully	Chrome	Pass
---	--------------------------------	---	-----------------------------	-----------------------------	--------	------

Test Scenario	Edit Customers - 1	Test Case ID	Edit_customers-1A
Test Case Description	Edit customers - Positive Test Case	Test Priority	High
Pre-Requisite	User logged in successfully and have relevant user role	Post-requisite	NA

Test Execution Steps:

S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Customer details on navigation menu	-	Expanded customer details menu	Expanded customer details menu	Chrome	Pass
2	Click View All	-	View all customer details	View all customer details	Chrome	Pass
3	Select customer to edit	Click on edit button at the last column of each customer row.	Edit Customer details form	Edit Customer details form		
4	Do the changes and click submit	Input: Change as required and click submit	Customer edited successfully	Customer edited successfully	Chrome	Pass

Test Scenario	Edit Customers - 1	Test Case ID	Edit_customers-1B
Test Case Description	Edit customers - Negative Test Case	Test Priority	High
Pre-Requisite	User logged in successfully and have relevant user role	Post-requisite	NA

Test Execution Steps:

S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Customer details on navigation menu	-	Expanded customer details menu	Expanded customer details menu	Chrome	Pass
2	Click View All	-	View all customer details	View all customer details	Chrome	Pass

3	Select customer to edit	Click on edit button at the last column of each customer row.	Edit Customer details form	Edit Customer details form		
4	Do the changes and click submit	Input: Change values with irrelevant or incorrect values	Customer cannot be edited	Customer cannot be edited	Chrome	Pass

Test Scenario	Delete Customers - 1			Test Case ID	Delete_customers-1A	
Test Case Description	Delete customers - Delete Test Case			Test Priority	High	
Pre-Requisite	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Customer details on navigation menu	-	Expanded customer details menu	Expanded customer details menu	Chrome	Pass
2	Click View All	-	View all customer details	View all customer details	Chrome	Pass
3	Select customer to Delete	Click on Delete button at the last column of each customer row.	Confirmation for deletion	Confirmation for deletion		
4	Confirm Deletion		Customer deleted successfully	Customer deleted successfully	Chrome	Pass
5	Decline Deletion		Do nothing	Do nothing	Chrome	Pass

Test Scenario	View Services - 1			Test Case ID	View_Services-1A	
Test Case Description	View Services - Positive Test Case			Test Priority	High	
Pre-Requisite	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						

S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Service details on navigation menu	-	Expanded Service details menu	Expanded Service details menu	Chrome	Pass
2	Click View all	-	View Service Details	View Service Details	Chrome	Pass
3	Filter Services	Enter a text to search	Filtered results according to text entered	Filtered results according to text entered	Chrome	Pass

Test Scenario	Add Services - 1	Test Case ID	Add_Services-1A
Test Case Description	Add Services - Negative Test Case	Test Priority	High
Pre-Requisite	User logged in successfully and have relevant user role	Post-requisite	NA

Test Execution Steps:

S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Service details on navigation menu	-	Expanded Service details menu	Expanded Service details menu	Chrome	Pass
2	Click Add New	-	Add Service Details	Add Service Details	Chrome	Pass
3	Click submit without completing form	Input - none	Display error messages under each form field	Display error messages under each form field	Chrome	Pass
4	Click submit with incorrect data	Input - incorrect data	Display error messages under incorrect data entered fields	Display error messages under incorrect data entered fields	Chrome	Pass

Test Scenario	Add Services - 1	Test Case ID	Add_Services-1B			
Test Case Description	Add Services - Positive Test Case	Test Priority	High			
Pre-Requisite	User logged in successfully and have relevant user role	Post-requisite	NA			
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Service details on navigation menu	-	Expanded Service details menu	Expanded Service details menu	Chrome	Pass
2	Click Add New	-	Add Service Details	Add Service Details	Chrome	Pass
3	Complete form and click submit	Input: Fill all form fields with relevant information	Service added successfully	Service added successfully	Chrome	Pass

Test Scenario	Edit Services - 1	Test Case ID	Edit_Services-1A			
Test Case Description	Edit Services - Positive Test Case	Test Priority	High			
Pre-Requisite	User logged in successfully and have relevant user role	Post-requisite	NA			
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Service details on navigation menu	-	Expanded Service details menu	Expanded Service details menu	Chrome	Pass
2	Click View All	-	View all Service details	View all Service details	Chrome	Pass
3	Select Service to edit	Click on edit button at the last column of each Service row.	Edit Service details form	Edit Service details form		
4	Do the changes and click submit	Input: Change as required and click submit	Service edited successfully	Service edited successfully	Chrome	Pass

Test Scenario	Edit Services - 1			Test Case ID	Edit_Services-1B	
Test Case Description	Edit Services - Negative Test Case			Test Priority	High	
Pre-Requirement	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Service details on navigation menu	-	Expanded Service details menu	Expanded Service details menu	Chrome	Pass
2	Click View All	-	View all Service details	View all Service details	Chrome	Pass
3	Select Service to edit	Click on edit button at the last column of each Service row.	Edit Service details form	Edit Service details form		
4	Do the changes and click submit	Input: Change values with irrelevant or incorrect values	Service cannot be edited	Service cannot be edited	Chrome	Pass

Test Scenario	Delete Services - 1			Test Case ID	Delete_Services-1A	
Test Case Description	Delete Services - Delete Test Case			Test Priority	High	
Pre-Requirement	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Service details on navigation menu	-	Expanded Service details menu	Expanded Service details menu	Chrome	Pass
2	Click View All	-	View all Service details	View all Service details	Chrome	Pass
3	Select Service to Delete	Click on Delete button at the last column of each Service row.	Confirmation for deletion	Confirmation for deletion		

4	Confirm Deletion		Service deleted successfully	Service deleted successfully	Chrome	Pass
5	Decline Deletion		Do nothing	Do nothing	Chrome	Pass

Test Scenario	View CarTypes - 1		Test Case ID	View_CarTypes-1A		
Test Case Description	View CarTypes - Positive Test Case		Test Priority	High		
Pre-Requisite	User logged in successfully and have relevant user role		Post-requisite	NA		
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click CarType details on navigation menu	-	Expanded CarType details menu	Expanded CarType details menu	Chrome	Pass
2	Click View all	-	View CarType Details	View CarType Details	Chrome	Pass
3	Filter CarTypes	Enter a text to search	Filtered results according to text entered	Filtered results according to text entered	Chrome	Pass

Test Scenario	Add CarTypes - 1		Test Case ID	Add_CarTypes-1A		
Test Case Description	Add CarTypes - Negative Test Case		Test Priority	High		
Pre-Requisite	User logged in successfully and have relevant user role		Post-requisite	NA		
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result

1	Click CarType details on navigation menu	-	Expanded CarType details menu	Expanded CarType details menu	Chrome	Pass
2	Click Add New	-	Add CarType Details	Add CarType Details	Chrome	Pass
3	Click submit without completing form	Input - none	Display error messages under each form field	Display error messages under each form field	Chrome	Pass
4	Click submit with incorrect data	Input - incorrect data	Display error messages under incorrect data entered fields	Display error messages under incorrect data entered fields	Chrome	Pass

Test Scenario	Add CarTypes - 1		Test Case ID	Add_CarTypes-1B		
Test Case Description	Add CarTypes - Positive Test Case		Test Priority	High		
Pre-Requisite	User logged in successfully and have relevant user role		Post-requisite	NA		
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click CarType details on navigation menu	-	Expanded CarType details menu	Expanded CarType details menu	Chrome	Pass
2	Click Add New	-	Add CarType Details	Add CarType Details	Chrome	Pass
3	Complete form and click submit	Input: Fill all form fields with relevant information	CarType added successfully	CarType added successfully	Chrome	Pass

Test Scenario	Edit CarTypes - 1	Test Case ID	Edit_CarTypes-1A
Test Case Description	Edit CarTypes - Positive Test Case	Test Priority	High

Pre-Requisite	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click CarType details on navigation menu	-	Expanded CarType details menu	Expanded CarType details menu	Chrome	Pass
2	Click View All	-	View all CarType details	View all CarType details	Chrome	Pass
3	Select CarType to edit	Click on edit button at the last column of each CarType row.	Edit CarType details form	Edit CarType details form		
4	Do the changes and click submit	Input: Change as required and click submit	CarType edited successfully	CarType edited successfully	Chrome	Pass
Test Scenario		Edit CarTypes - 1		Test Case ID		Edit_CarTypes-1B
Test Case Description		Edit CarTypes - Negative Test Case		Test Priority		High
Pre-Requisite	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click CarType details on navigation menu	-	Expanded CarType details menu	Expanded CarType details menu	Chrome	Pass
2	Click View All	-	View all CarType details	View all CarType details	Chrome	Pass
3	Select CarType to edit	Click on edit button at the last column of each CarType row.	Edit CarType details form	Edit CarType details form		
4	Do the changes and click submit	Input: Change values with irrelevant or incorrect values	CarType cannot be edited	CarType cannot be edited	Chrome	Pass

Test Scenario	Delete CarTypes - 1			Test Case ID	Delete_CarTypes-1A	
Test Case Description	Delete CarTypes - Delete Test Case			Test Priority	High	
Pre-Requirement	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click CarType details on navigation menu	-	Expanded CarType details menu	Expanded CarType details menu	Chrome	Pass
2	Click View All	-	View all CarType details	View all CarType details	Chrome	Pass
3	Select CarType to Delete	Click on Delete button at the last column of each CarType row.	Confirmation for deletion	Confirmation for deletion		
4	Confirm Deletion		CarType deleted successfully	CarType deleted successfully	Chrome	Pass
5	Decline Deletion		Do nothing	Do nothing	Chrome	Pass

Test Scenario	View Mechanics - 1			Test Case ID	View_Mechanics-1A	
Test Case Description	View Mechanics - Positive Test Case			Test Priority	High	
Pre-Requirement	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Mechanic details on navigation menu	-	Expanded Mechanic details menu	Expanded Mechanic details menu	Chrome	Pass
2	Click View all	-	View Mechanic Details	View Mechanic Details	Chrome	Pass

3	Filter Mechanics	Enter a text to search	Filtered results according to text entered	Filtered results according to text entered	Chrome	Pass
---	------------------	------------------------	--	--	--------	------

Test Scenario	Add Mechanics - 1			Test Case ID	Add_Mechanics-1A	
Test Case Description	Add Mechanics - Negative Test Case			Test Priority	High	
Pre-Requisite	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Mechanic details on navigation menu	-	Expanded Mechanic details menu	Expanded Mechanic details menu	Chrome	Pass
2	Click Add New	-	Add Mechanic Details	Add Mechanic Details	Chrome	Pass
3	Click submit without completing form	Input - none	Display error messages under each form field	Display error messages under each form field	Chrome	Pass
4	Click submit with incorrect data	Input - incorrect data	Display error messages under incorrect data entered fields	Display error messages under incorrect data entered fields	Chrome	Pass

Test Scenario	Add Mechanics - 1			Test Case ID	Add_Mechanics-1B	
Test Case Description	Add Mechanics - Positive Test Case			Test Priority	High	

Pre-Requirement	User logged in successfully and have relevant user role			Post-requirement	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Mechanic details on navigation menu	-	Expanded Mechanic details menu	Expanded Mechanic details menu	Chrome	Pass
2	Click Add New	-	Add Mechanic Details	Add Mechanic Details	Chrome	Pass
3	Complete form and click submit	Input: Fill all form fields with relevant information	Mechanic added successfully	Mechanic added successfully	Chrome	Pass

Test Scenario	Edit Mechanics - 1			Test Case ID	Edit_Mechanics-1A	
Test Case Description	Edit Mechanics - Positive Test Case			Test Priority	High	
Pre-Requirement	User logged in successfully and have relevant user role			Post-requirement	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Mechanic details on navigation menu	-	Expanded Mechanic details menu	Expanded Mechanic details menu	Chrome	Pass
2	Click View All	-	View all Mechanic details	View all Mechanic details	Chrome	Pass
3	Select Mechanic to edit	Click on edit button at the last column of each Mechanic row.	Edit Mechanic details form	Edit Mechanic details form		
4	Do the changes and click submit	Input: Change as required and click submit	Mechanic edited successfully	Mechanic edited successfully	Chrome	Pass
Test Scenario	Edit Mechanics - 1			Test Case ID	Edit_Mechanics-1B	
Test Case Description	Edit Mechanics - Negative Test Case			Test Priority	High	

Pre-Requisite	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Mechanic details on navigation menu	-	Expanded Mechanic details menu	Expanded Mechanic details menu	Chrome	Pass
2	Click View All	-	View all Mechanic details	View all Mechanic details	Chrome	Pass
3	Select Mechanic to edit	Click on edit button at the last column of each Mechanic row.	Edit Mechanic details form	Edit Mechanic details form		
4	Do the changes and click submit	Input: Change values with irrelevant or incorrect values	Mechanic cannot be edited	Mechanic cannot be edited	Chrome	Pass

Test Scenario	Delete Mechanics - 1			Test Case ID	Delete_M echanics-1A	
Test Case Description	Delete Mechanics - Delete Test Case			Test Priority	High	
Pre-Requisite	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click Mechanic details on navigation menu	-	Expanded Mechanic details menu	Expanded Mechanic details menu	Chrome	Pass
2	Click View All	-	View all Mechanic details	View all Mechanic details	Chrome	Pass
3	Select Mechanic to Delete	Click on Delete button at the last column of each Mechanic row.	Confirmation for deletion	Confirmation for deletion		
4	Confirm Deletion		Mechanic deleted successfully	Mechanic deleted successfully	Chrome	Pass

5	Decline Deletion		Do nothing	Do nothing	Chrome	Pass
---	------------------	--	------------	------------	--------	------

Test Scenario		View ServiceAreas - 1		Test Case ID		View_ServiceAreas-1A
Test Case Description		View ServiceAreas - Positive Test Case		Test Priority		High
Pre-Requirement		User logged in successfully and have relevant user role		Post-requisite		NA
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click ServiceArea details on navigation menu	-	Expanded ServiceArea details menu	Expanded ServiceArea details menu	Chrome	Pass
2	Click View all	-	View ServiceArea Details	View ServiceArea Details	Chrome	Pass
3	Filter ServiceAreas	Enter a text to search	Filtered results according to text entered	Filtered results according to text entered	Chrome	Pass

Test Scenario		Add ServiceAreas - 1		Test Case ID		Add_ServiceAreas-1A
Test Case Description		Add ServiceAreas - Negative Test Case		Test Priority		High
Pre-Requirement		User logged in successfully and have relevant user role		Post-requisite		NA
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result

1	Click ServiceArea details on navigation menu	-	Expanded ServiceArea details menu	Expanded ServiceArea details menu	Chrome	Pass
2	Click Add New	-	Add ServiceArea Details	Add ServiceArea Details	Chrome	Pass
3	Click submit without completing form	Input - none	Display error messages under each form field	Display error messages under each form field	Chrome	Pass
4	Click submit with incorrect data	Input - incorrect data	Display error messages under incorrect data entered fields	Display error messages under incorrect data entered fields	Chrome	Pass

Test Scenario	Add ServiceAreas - 1		Test Case ID	Add_ServiceAreas-1B		
Test Case Description	Add ServiceAreas - Positive Test Case		Test Priority	High		
Pre-Requisite	User logged in successfully and have relevant user role		Post-requisite	NA		
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click ServiceArea details on navigation menu	-	Expanded ServiceArea details menu	Expanded ServiceArea details menu	Chrome	Pass
2	Click Add New	-	Add ServiceArea Details	Add ServiceArea Details	Chrome	Pass
3	Complete form and click submit	Input: Fill all form fields with relevant information	ServiceArea added successfully	ServiceArea added successfully	Chrome	Pass

Test Scenario	Edit ServiceAreas - 1	Test Case ID	Edit_ServiceAreas-1A
----------------------	-----------------------	---------------------	----------------------

Test Case Description	Edit ServiceAreas - Positive Test Case			Test Priority	High	
Pre-Requirement	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click ServiceArea details on navigation menu	-	Expanded ServiceArea details menu	Expanded ServiceArea details menu	Chrome	Pass
2	Click View All	-	View all ServiceArea details	View all ServiceArea details	Chrome	Pass
3	Select ServiceArea to edit	Click on edit button at the last column of each ServiceArea row.	Edit ServiceArea details form	Edit ServiceArea details form		
4	Do the changes and click submit	Input: Change as required and click submit	ServiceArea edited successfully	ServiceArea edited successfully	Chrome	Pass
Test Scenario	Edit ServiceAreas - 1			Test Case ID	Edit_ServiceAreas-1B	
Test Case Description	Edit ServiceAreas - Negative Test Case			Test Priority	High	
Pre-Requirement	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click ServiceArea details on navigation menu	-	Expanded ServiceArea details menu	Expanded ServiceArea details menu	Chrome	Pass
2	Click View All	-	View all ServiceArea details	View all ServiceArea details	Chrome	Pass
3	Select ServiceArea to edit	Click on edit button at the last column of each ServiceArea row.	Edit ServiceArea details form	Edit ServiceArea details form		
4	Do the changes	Input: Change values with irrelevant or incorrect values	ServiceArea cannot be edited	ServiceArea cannot be edited	Chrome	Pass

	and click submit					
--	------------------	--	--	--	--	--

Test Scenario	Delete ServiceAreas - 1			Test Case ID	Delete_ServiceAreas-1A	
Test Case Description	Delete ServiceAreas - Delete Test Case			Test Priority	High	
Pre-Requisite	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click ServiceArea details on navigation menu	-	Expanded ServiceArea details menu	Expanded ServiceArea details menu	Chrome	Pass
2	Click View All	-	View all ServiceArea details	View all ServiceArea details	Chrome	Pass
3	Select ServiceArea to Delete	Click on Delete button at the last column of each ServiceArea row.	Confirmation for deletion	Confirmation for deletion		
4	Confirm Deletion		ServiceArea deleted successfully	ServiceArea deleted successfully	Chrome	Pass
5	Decline Deletion		Do nothing	Do nothing	Chrome	Pass

Test Scenario	View Order Details - 1			Test Case ID	View_OrderDetails-1	
Test Case Description	View Order Details			Test Priority	High	
Pre-Requisite	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click View Order details on	-	Expanded Order	Expanded Order	Chrome	Pass

	navigation menu		Details menu	details menu		
2	Click View All	-	View all order details	View all order details	Chrome	Pass
Test Scenario		Edit Order Details - 1		Test Case ID		Edit_OrderDetails-1
Test Case Description		Edit Order Details		Test Priority		High
Pre-Requisite		User logged in successfully and have relevant user role		Post-requisite		NA
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click View Order details on navigation menu	-	Expanded Order Details menu	Expanded Order details menu	Chrome	Pass
2	Click View All	-	View all order details	View all order details	Chrome	Pass
3	Select order to edit	Click edit icon in front of an order	Edit order details	Edit order details	Chrome	Pass
4	Modify values and click submit	Input: modified values in the existing order	Order edited successfully	Order edited successfully	Chrome	Pass

Test Scenario		View Rejected Order Details - 1		Test Case ID		View_RejectedOrderDetails-1
Test Case Description		View Rejected Order Details		Test Priority		High
Pre-Requisite		User logged in successfully and have relevant user role		Post-requisite		NA
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click View Rejected Order details on	-	Expanded Rejected Order Details menu	Expanded Rejected Order details menu	Chrome	Pass

	navigation menu					
2	Click View All	-	View all Rejected order details	View all Rejected order details	Chrome	Pass

Test Scenario	Edit Rejected Order Details - 1			Test Case ID	Edit_RejectedOrderDetails-1	
Test Case Description	Edit Rejected Order Details			Test Priority	High	
Pre-Requisite	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click View Rejected Order details on navigation menu	-	Expanded Rejected Order Details menu	Expanded Rejected Order details menu	Chrome	Pass
2	Click View All	-	View all order details	View all order details	Chrome	Pass
3	Select order to edit	Click edit icon in front of an order	Edit order details	Edit order details	Chrome	Pass
4	Modify values and click submit	Input: modified values in the existing order	Order edited successfully	Order edited successfully	Chrome	Pass

Test Scenario	View BackendUsers - 1			Test Case ID	View_BackendUsers-1A	
Test Case Description	View BackendUsers - Positive Test Case			Test Priority	High	
Pre-Requisite	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result

1	Click BackendUser details on navigation menu	-	Expanded BackendUser details menu	Expanded BackendUser details menu	Chrome	Pass
2	Click View all	-	View BackendUser Details	View BackendUser Details	Chrome	Pass
3	Filter BackendUsers	Enter a text to search	Filtered results according to text entered	Filtered results according to text entered	Chrome	Pass

Test Scenario	Add BackendUsers - 1			Test Case ID	Add_BackendUsers-1A	
Test Case Description	Add BackendUsers - Negative Test Case			Test Priority	High	
Pre-Requisite	User logged in successfully and have relevant user role			Post-requisite	NA	
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click BackendUser details on navigation menu	-	Expanded BackendUser details menu	Expanded BackendUser details menu	Chrome	Pass
2	Click Add New	-	Add BackendUser Details	Add BackendUser Details	Chrome	Pass
3	Click submit without completing form	Input - none	Display error messages under each form field	Display error messages under each form field	Chrome	Pass
4	Click submit with incorrect data	Input - incorrect data	Display error messages under incorrect data entered fields	Display error messages under incorrect data entered fields	Chrome	Pass

Test Scenario	Add BackendUsers - 1	Test Case ID	Add_BackendUsers-1B			
Test Case Description	Add BackendUsers - Positive Test Case	Test Priority	High			
Pre-Requisite	User logged in successfully and have relevant user role	Post-requisite	NA			
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click BackendUser details on navigation menu	-	Expanded BackendUser details menu	Expanded BackendUser details menu	Chrome	Pass
2	Click Add New	-	Add BackendUser Details	Add BackendUser Details	Chrome	Pass
3	Complete form and click submit	Input: Fill all form fields with relevant information	BackendUser added successfully	BackendUser added successfully	Chrome	Pass

Test Scenario	Edit BackendUsers - 1	Test Case ID	Edit_BackendUsers-1A			
Test Case Description	Edit BackendUsers - Positive Test Case	Test Priority	High			
Pre-Requisite	User logged in successfully and have relevant user role	Post-requisite	NA			
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click BackendUser details on navigation menu	-	Expanded BackendUser details menu	Expanded BackendUser details menu	Chrome	Pass
2	Click View All	-	View all BackendUser details	View all BackendUser details	Chrome	Pass
3	Select BackendUser to edit	Click on edit button at the last column of each BackendUser row.	Edit BackendUser details form	Edit BackendUser details form		

4	Do the changes and click submit	Input: Change as required and click submit	BackendUser edited successfully	BackendUser edited successfully	Chrome	Pass
Test Scenario		Edit BackendUsers - 1		Test Case ID		Edit_BackendUsers-1B
Test Case Description		Edit BackendUsers - Negative Test Case		Test Priority		High
Pre-Requisite		User logged in successfully and have relevant user role		Post-requisite		NA
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click BackendUser details on navigation menu	-	Expanded BackendUser details menu	Expanded BackendUser details menu	Chrome	Pass
2	Click View All	-	View all BackendUser details	View all BackendUser details	Chrome	Pass
3	Select BackendUser to edit	Click on edit button at the last column of each BackendUser row.	Edit BackendUser details form	Edit BackendUser details form		
4	Do the changes and click submit	Input: Change values with irrelevant or incorrect values	BackendUser cannot be edited	BackendUser cannot be edited	Chrome	Pass

Test Scenario		Delete BackendUsers - 1		Test Case ID		Delete_BackendUsers-1A
Test Case Description		Delete BackendUsers - Delete Test Case		Test Priority		High
Pre-Requisite		User logged in successfully and have relevant user role		Post-requisite		NA
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click BackendUser details on	-	Expanded BackendUser details menu	Expanded BackendUser details menu	Chrome	Pass

	navigation menu					
2	Click View All	-	View all BackendUser details	View all BackendUser details	Chrome	Pass
3	Select BackendUser to Delete	Click on Delete button at the last column of each BackendUser row.	Confirmation for deletion	Confirmation for deletion		
4	Confirm Deletion		BackendUser deleted successfully	BackendUser deleted successfully	Chrome	Pass
5	Decline Deletion		Do nothing	Do nothing	Chrome	Pass

Test Scenario	Add User Role - 1	Test Case ID	Add_User Role-1B			
Test Case Description	Add User Role - Positive Test Case	Test Priority	High			
Pre-Requisite	User logged in successfully and have relevant user role	Post-requisite	NA			
Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result
1	Click User Role on navigation menu	-	Expanded User Role menu	Expanded User Role menu	Chrome	Pass
2	Click Add New	-	Add User Role	Add User Role	Chrome	Pass
3	Complete form and click submit	Input: Fill all form fields with relevant information, select relevant user roles	User Role successfully	User Role successfully	Chrome	Pass