

E-Showroom for Luxury Car



KHWAJA FAREED
UEIT
RAHIM YAR KHAN

Institute of
Computer Science

Submitted By

Umer Ghafoor

Hafiza Asma Rafiq

2020-2024

Institute of Computer Science

**Khwaja Fareed University of Engineering &
Information Technology**

Rahim Yar Khan

2023

E-Showroom for Luxury Cars

**Submitted to
Mr. Muhammad Ahsan Aslam**

Institute of Computer Science

**In partial fulfilment of the requirements
For the degree of**

Bachelors in Computer Science

By

Umer Ghafoor

COSC 19111100

Hafiza Asma Rafiq

COSC 201101030

**Khwaja Fareed University of Engineering &
Information Technology
Rahim Yar Khan
2023**

DECLARATION

We hereby declare that this project report is based on our original work except for citations and quotations which have been duly acknowledged. We also declare that it has not been previously and concurrently submitted for any other degree or award at Khwaja Fareed University of engineering & Information Technology or other institutions.

Reg No : COSC191111100

Reg No : COSC201101030

Name : Umer Ghafoor

Name : Hafiza Asma Rafiq

Signature : _____

Signature : _____

Date : _____

Date : _____

APPROVAL FOR SUBMISSION

I certify that this project report entitled *E-Showroom for Luxury Cars* was prepared by **Umer Ghafoor** and **Hafiza Asma Rafiq** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor/Masters of Science in Computer Science at Khwaja Fareed University of Engineering & Information Technology.

Approved by:

Signature : _____

Supervisor : Mr. Muhammad Ahsan Aslam

Date : _____

MEETING LOG

Project Name: E-Showroom and Garage

Supervisor: Mr. Muhammad Ahsan Aslam

Meeting No.	Date	Outcome	Signature of Supervisor
1	15/09/2023	Finalize the Project Idea	
2	19/09/2023	Prepared the Title Defense Presentation	
3	26/09/2023	We confirmed the project and discussed the functional requirements in our project.	
4	28/09/2023	Discussed the documentation and further project proceedings.	
5	29/09/2023	In this meeting, chapter 1 complete documentation is presented.	
6	02/10/2023	In this meeting, chapter 2 documentation starts.	
7	03/10/2023	In this meeting, Chapter 2 is reviewed.	
8	09/10/2023	Finalize the chapter 2	
9	12/10/2023	Review the chapter 1,2	
10	15/11/2023	Draft the chapter 3	
11	21/11/2023	Review the chapter 3	
12	04/12/2023	Draft the chapter 4	
13	05/12/2023	Recheck mistakes in chapter 4	
14	07/12/2023	Finalize the chapter 4	
15	08/12/2023	Finalize FYP-I Presentation	

Meeting No.	Date	Outcome	Signature of Supervisor
16	20/02/2024	Draft the chapter 5	
17	29/02/2024	Review the chapter 5	
18	07/03/2024	We discuss Analysis and deployment of the project.	
19	11/03/2024	Discuss the feature requirement of project in development.	
20	14/03/2024	Finalize the chapter 5	
21	15/03/2024	Draft the chapter 6	
22	25/03/2024	Review the chapter 6	
23	27/03/2024	Finalize the chapter 6	
24	29/03/2024	Draft the chapter 7	
25	15/04/2024	Review the chapter 7	
26	16/04/2024	Finalize the chapter 7	
27	18/04/2024	Finalize the FYP project and documentation	

ACKNOWLEDGEMENT

We would like to thank everyone who had contributed to this project. We would like to express my/our gratitude to our Project supervisor, *Mr. Muhammad Ahsan Aslam* for his invaluable advice, guidance, and his/her enormous patience throughout the development of the project.

In addition, we would also like to express our gratitude to my/our loving parents and friends who had helped and given me/us encouragement.

ABSTRACT

Envisioned system is focused on developing an E-showroom for luxury cars; (BMW, Bugatti, Audi, Lamborghini, Porsche, Tesla, etc.). The project will have a powerful focus on garage services like repairing cars, engine works, ex-factory modifications, etc. Generic services for the engine and other parts of a vehicle are also to be included. If a customer has any problem with his vehicle based on a guarantee card, we are providing services for that as well. E-Showroom for Luxury Cars is an online garage service platform for four-wheeler servicing. The company aspires to provide hassle-free, timely, and affordable services through a single platform. Make a garage for cars, give internships to people, sell imported cars, do online marketing of luxury cars, give services of cars, etc. Our car servicing module gets you inquiries and gets you appearing above and amongst the main dealers and chains. general repair, denting, painting, cleaning, and detailing services are also available. It has secured access to admin. It is a smart web UI that could assist garage owners in keeping track of all the events in the garage. The admin shall give access to specific modules for the other users. The system will also allow payment for the repair or service done.

TABLE OF CONTENTS

CHAPTER 1.....	1
INTRODUCTION.....	1
1.1. Background	1
1.2. Project Overview	1
1.3. Introduction.....	2
1.4. Problem Statement	3
1.5. Objective	3
1.6. Project Scope	3
1.7. Advantages of the System.....	4
1.7.1. Personalized Garage Management	4
1.7.2. Efficient Car Search and Comparison	4
1.7.3. Brand exposure for luxury cars	4
1.8. Relevance to the study	4
1.9. Chapter summary	4
CHAPTER 2.....	6
EXISTING SYSTEMS	6
2.1. Existing Systems	6
2.2. Drawbacks in Existing Systems	6
2.3. Examples of Existing Systems	7
2.4. Need to Replace Existing Systems.....	8
2.5. Chapter Summary	9
CHAPTER 3.....	10
REQUIREMENT ENGINEERING	10

3.1. Proposed System	10
3.2. Understanding the System	10
3.2.1. User Involvement.....	10
3.2.2. Stakeholders.....	10
3.2.3. Domain.....	11
3.2.4. Needs of System	11
Table 3.1 User Needs of System	11
3.3 Requirement Engineering	11
3.3.1. Functional Requirements	11
3.3.2. Non-Functional Requirements	17
3.3.2.2. Security.....	17
3.3.2.3. Privacy	17
3.3.2.4. Performance.....	17
3.4. Gantt Chart.....	18
3.5. Hurdles in Optimizing the Current System.....	18
3.6. Chapter Summary	19
CHAPTER 4.....	20
DESIGN	20
4.1. Software Process Model	20
4.1.1. Benefits of Selected Model.....	21
4.1.2. Limitations of Selected Model.....	21
4.1.3. Benefits of Selected Model.....	21
4.1.4. Limitations of Selected Model.....	22
4.2. Design	22
4.2.1. Methodology of the Proposed System	22

4.2.2.	Entity Relationship Diagram.....	23
4.2.3.	UML Diagrams	23
4.2.3.1.	Use Case Diagram of the System	25
4.2.3.2.	Class Diagram of the System.....	26
4.2.3.3.	Activity Diagram of the System	27
4.2.3.4.	Sequence Diagram of the System.....	28
4.2.3.5.	Component Diagram of the System.....	29
4.3.	Chapter Summary	29
CHAPTER 5.....		30
DATABASE.....		30
5.1.	Database Introduction	30
5.2.	Selected Database.....	30
5.2.1.	Reasons for Selection of the Database.....	30
5.2.1.1.	Scalability	30
5.2.1.2.	Flexibility.....	31
5.2.1.3.	High Performance	31
5.2.1.4.	Security of data	31
5.2.1.5.	Data integrity	31
5.2.2.	Benefits of the Selected Database.....	31
5.2.2.1.	Scalability	31
5.2.2.2.	Speed	31
5.2.2.3.	Replication and High Availability	32
5.2.2.4.	Support for a wide range of data types	32
5.2.2.5.	Easy integration	32
5.2.3.	Limitations of the Selected Database.....	32

5.3. Database Queries	32
5.4. Database Tables	33
5.5. Database Schema Diagram	35
5.6. Chapter Summary	35
CHAPTER 6.....	36
DEVELOPMENT AND IMPLEMENTATION.....	36
6.1. Development of the Computer Program	36
6.2. Implementation Strategy	37
6.2.1. Control your scope, or it will control you	37
6.2.2. Assign realistic teams to drive software implementation plans	37
6.2.3. Focus on continuous improvements.....	37
6.3. Tools Selection.....	38
6.3.1. Hardware.....	38
6.3.2. Software	38
6.4. Coding.....	38
6.5. User Interface.....	43
6.5.1. Description	44
6.5.2. Interface Screenshots	45
6.6. Program Deployment	46
6.7. Chapter Summary	46
CHAPTER 7.....	47
TESTING.....	47
7.1. Introduction	47
7.2. Testing Methods.....	48
7.3. Comparison	48

7.4. Software Evaluation	49
7.4.1. Testing Strategy	50
7.4.2. Test Plans	51
7.4.1. Test Cases	52
7.4.1. Test Report.....	58
7.5. Chapter Summary	59
REFERENCES.....	60
APPENDIX (OPTIONAL).....	62

LIST OF FIGURES

FIGURE 3.1 GANTT CHART.....	18
FIGURE 4.1 INCREMENTAL MODEL DIAGRAM.....	20
FIGURE 4.2 METHODOLOGY.....	22
FIGURE 4.3 ER-DIAGRAM.....	23
FIGURE 4.4 USE CASE DIAGRAM	25
FIGURE 4.5 CLASS DIAGRAM	ERROR! BOOKMARK NOT DEFINED.
FIGURE 4.6 ACTIVITY DIAGRAM.....	27
FIGURE 4.8 COMPONENT DIAGRAM	29
FIGURE 5.1 SCHEMA DIAGRAM.....	35
FIGURE 6.1 PDLC.....	37
FIGURE 6.2 APP.JS	38
FIGURE 6.3 APP JS.....	39
FIGURE 6.4 APP.JS	39
FIGURE 6.5 LOGIN.....	40
FIGURE 6.6 SIGN UP.....	40
FIGURE 6.7 HOME.JS.....	41
FIGURE 6.8 HOME.JS.....	41
FIGURE 6.9 INDEX.JS.....	42
FIGURE 6.10 EXPRESS.JS.....	42
FIGURE 6.12 USER INTERFACE.....	43
FIGURE 6.13 LOGIN.....	43
FIGURE 6.14 SIGN UP.....	44
FIGURE 6.15 ABOUT US	45

FIGURE 6.16 CONTACT US	45
FIGURE 6.17 FEEDBACK	46
FIGURE 7.1 CASE NO: 01.....	52
FIGURE 7.2 CASE NO: 02.....	53
FIGURE 7.3 CASE NO: 03.....	53
FIGURE 7.4 CASE NO: 04.....	54
FIGURE 7.5 CASE NO: 05.....	54
FIGURE 7.6 CASE NO: 06.....	55
FIGURE 7.7 CASE NO: 07.....	55
FIGURE 7.8 CASE NO: 08.....	56
FIGURE 7.9 CASE NO: 09.....	56
FIGURE 7.10 CASE NO: 10.....	57
FIGURE 7.11 CASE NO: 11.....	58

LIST OF TABLES

TABLE 3.1 USER NEEDS OF SYSTEM	11
TABLE 3.2 FUNCTIONAL REQUIREMENT 01	12
TABLE 3.3. FUNCTIONAL REQUIREMENT 02.....	12
TABLE 3.4 FUNCTIONAL REQUIREMENT 03.....	13
TABLE 3.5 FUNCTIONAL REQUIREMENT 04.....	13
TABLE 3.6 FUNCTIONAL REQUIREMENT 05.....	14
TABLE 3.7 FUNCTIONAL REQUIREMENT 06.....	14
TABLE 3.8 FUNCTIONAL REQUIREMENT 07.....	15
TABLE 3.9 FUNCTIONAL REQUIREMENT 08.....	15
TABLE 3.10 FUNCTIONAL REQUIREMENT 09.....	16
TABLE 3.11 FUNCTIONAL REQUIREMENT 10.....	16
TABLE 3.12FUNCTIONAL REQUIREMENT 11	17
TABLE 5.1 ADMIN.....	33
TABLE 5.2 DEALER	33
TABLE 5.3 CUSTOMER	33
TABLE 5.4 SERVICE.....	34
TABLE 5.5 SHOWROOM	34

Chapter 1

INTRODUCTION

1.1. Background

In recent years, the automotive industry has undergone a profound shift in response to the evolving digital landscape. Traditional methods of luxury car shopping, which often involved visiting physical showrooms and relying on sales representatives for information, have given way to a new era of online exploration and purchasing. This transformation has been driven by several key factors:

Today's car buyers are more informed and discerning than ever before. They demand access to detailed specifications, performance data, pricing options, and financing details to make well-informed choices. The ability to compare multiple models' side by side has become a standard expectation.

This approach provides a taste of a vehicle's performance and handling, adding a new dimension to the online shopping process. Personalization features, such as saving preferences and receiving tailored recommendations, have become integral to enhancing the customer experience. This allows customers to find the luxury car that aligns perfectly with their preferences and needs. Given these advancements and shifts in consumer behavior, the "E-Showroom for Luxury Cars" project aims to capitalize on the digital transformation of luxury car shopping.

It seeks to redefine the car-buying experience by offering a state-of-the-art, immersive, and user-friendly virtual showroom where customers can explore, interact with, and purchase high-end luxury cars.

This project is designed to meet the evolving needs and expectations of customers in the luxury car market, making it an innovative and timely solution for the automotive industry.

1.2. Project Overview

E-Showroom for Luxury Cars is a comprehensive and user-friendly web application that consolidates the entire luxury car buying process into a seamless, one-stop platform. From the initial step of browsing to the final stage of making a purchase, this application streamlines the luxury car shopping experience.

The system allows users to manage their booking online. Customers are allowed to book vehicles and spare parts online. The system manages the upcoming events data for users of various date and times. Each time a user visits Showroom his/her entry is stored in the database. The main objective of the system is to manage all the details of buyers, vehicles, spare parts, services. The project is totally built at administrative end and thus only the administrator is guaranteed the access. Our system will reduce the paperwork and storage area and it will also save a lot of time of Customers.

1.3. Introduction

Welcome to the E Showroom and Garage for Luxury Cars, where the world of luxury automobiles meets the convenience of the digital age. Our project is designed to provide enthusiasts with a sophisticated and user-friendly platform to explore, purchase, and showcase their most coveted luxury cars. In a society driven by technology and elegance, our mission is to create a seamless online space that caters to the desires of luxury car enthusiasts. With a curated selection of the finest automobiles, coupled with a unique digital garage feature, users can immerse themselves in the world of opulence from the comfort of their screens. Elevate your car-buying experience as we bring together a collection of the world's most prestigious vehicles, ensuring that every detail is at your fingertips. Whether you're an avid collector, a connoisseur of automotive design, or someone aspiring to own a piece of automotive excellence, E Showroom and Garage for Luxury Cars is your destination.

Join us on this journey as we redefine luxury car shopping and collection management in the digital era. This is more than a showroom; it's a virtual sanctuary for those who appreciate the artistry, craftsmanship, and thrill of luxury automobiles.

Our project owns a virtual space for your automotive treasures. The digital garage feature allows users to organize, manage, and own a virtual space for your automotive treasures. The digital garage feature allows users to organize, manage, and display their luxury car collections. Whether it's a classic masterpiece or the latest in cutting-edge design, your garage is a reflection of your unique taste and style. d display their luxury car collections. Whether it's a classic masterpiece or the latest in cutting-edge design, your garage is a reflection of your unique taste and style.

Welcome to the future of luxury car exploration – Welcome to E Showroom and Garage for Luxury Cars.

1.4. Problem Statement

Other showrooms can only attract people who visit physically. Buying a luxury car is complicated, and customers want to customize them. We don't have a good way to schedule car maintenance or track service history for luxury cars.

Luxury car purchases often involve complicated financing options. The luxury car showroom industry is highly competitive and relies heavily on a strong physical presence to attract and engage potential customers. It is challenging for users to navigate, explore, and inquire about the vehicles or services offered by the dealership.

1.5. Objective

The main idea of this work is to provide ease and comfort to user while choosing vehicles, and it also resolves the problems that the user has to face while choosing his/her dream car. The main objective of the system is to manage all the details of the showroom, user, vehicles, garage, spare parts, etc. The system is totally built at the administrative end and thus only the administrator is guaranteed access. The purpose of the system is to build an application program to reduce the manual work for showroom, users. Users can book orders, do online marketing of luxury cars, sell imported cars, list vehicles give services of cars, etc.

1.6. Project Scope

It is designed in a simple interface to work in and facilitate any showroom. It may help collect perfect management in detail. In a very short time, the collection will be obvious, simple, and sensible. It will also reduce the cost of collecting the management and the collection procedure will go on smoothly.

Our system aims at business process automation i.e. we have tried to computerize various processes of the showroom.

- In a computer system it is not necessary to create the manifest but we can directly print it, which saves you time.
- The system generates types of information that can used for various purposes.
- It satisfies the user requirement.
- Be easy to understand by the user and operator.
- Be easy to operate
- It has a good user interface

Overall, the scope of this project is local (national) in its field. And the system design is quite feasible to use and maintain.

1.7. Advantages of the System

Users can explore and experience luxury cars from the comfort of their homes, providing a unique and convenient virtual showroom experience. Access to detailed information, high-quality images, and videos enhances the decision-making process for potential buyers.

1.7.1. Personalized Garage Management

Users can create and manage a virtual garage to keep track of their luxury car collections. Features such as maintenance reminders and detailed car profiles contribute to a personalized and organized car ownership experience

1.7.2. Efficient Car Search and Comparison

Advanced search and filtering options enable users to find specific luxury cars based on their preferences. Users can compare different cars side by side, aiding in the decision-making process.

1.7.3. Brand exposure for luxury cars

Luxury car dealers and sellers benefit from increased online visibility and exposure to a global audience. The platform serves as an additional channel for marketing and showcasing their luxury car inventory.

1.8. Relevance to the study

Relevance to the study includes consideration of all the possible ways to provide a solution to the given problem. The proposed solution should satisfy all the user requirements and should be flexible enough so that further changes can be easily done based on future upcoming requirements.

1.9. Chapter summary

E Showroom and Garage for Luxury Cars, against the backdrop of a transforming automotive industry influenced by digital advancements. The shift from traditional luxury car shopping to an online, information-driven approach is outlined, emphasizing the changing expectations of informed and discerning car buyers.

The project, born out of this digital evolution, is named the "E-Showroom for Luxury Cars." Its goal is to revolutionize the luxury car-buying experience by offering a sophisticated, immersive, and user-friendly virtual showroom. This comprehensive web application aims to streamline the entire luxury car purchasing process, from browsing to finalizing a purchase, catering to the needs of tech-savvy and detail-oriented customers.

The documentation outlines the background, objectives, and advantages of the project. It stresses the importance of addressing current challenges in the luxury car market, such as the complexity of purchasing, customization demands, and the lack of effective maintenance tracking systems. The chapter concludes with a clear statement of objectives, emphasizing the project's timeliness and innovation in meeting the evolving needs of luxury car consumers.

Chapter 2

EXISTING SYSTEMS

2.1. Existing Systems

In a competitive business organization, the ability to efficiently align resources and business activities with strategic objectives can mean the difference between succeeding and just surviving. To achieve strategic alignment, the organization is increasingly managing its activities and processes as projects, in essence, projecting its business to monitor performance more closely and make better business decisions above its overall work portfolio.

By planning and tracking projects with clarity and precision, organizations can respond with greater ability to the demands of a fast-changing business environment. Making strategic goals or reality requires technology that is robust enough to support your core business and yet flexible enough to accommodate your existing processes. The car showroom manager software is one of the existing tools that are helpful for car showroom management. The primary purposes of any such tools are to improve productivity, reduce cycle time, and decrease costs by increasing quality.

2.2. Drawbacks in Existing Systems

The old manual system was suffering from a series of drawbacks. Since whole of the system was to be maintained with hands the process of keeping, maintaining and retrieving the information was very tedious and lengthy. The records were never used to be in a systematic order. There used to be lots of difficulties in associating any particular transaction with a particular context. If any information was to be found it was required to go through the different registers, documents there would never exist anything like report generation. There would always be unnecessary consumption of time while entering records and retrieving records. One more problem was that it was very difficult to find errors while entering the records. Once the records were entered it was very difficult to update these records.

- Only focus on a specific local based vehicle.
- Lack features to garage management.
- Critically required portions on the focused part are left out.

2.3. Examples of Existing Systems

The existing tools do not consider all the factors that affect a project directly. The entire process is built upon using just one or two parameters which do not give us complete and desired results. So, we need a system where we can deliver the correct results that would ultimately lead to a position where the overall cost and time will be consolidated. No specific training is required for the distributors to use this application. They can easily use the tool that decreases manual hours spent on normal things and hence increases performance. It is very easy to record the information of online sales and purchases in the databases.

There are some of the systems that exist in the world of internet are same effective as our system. On showroom level, there has never been a dedicated software system employed for dealing with luxury cars and garage. There used to be lots of difficulties in associating any particular transaction with a particular context. Some of the existing system examples are giving below:

- <http://www.pakistancardealers.com/>
- <https://autodeals.pk/new-cars-dealers>
- <https://unitedcars.com.pk/united-bravo-car/>

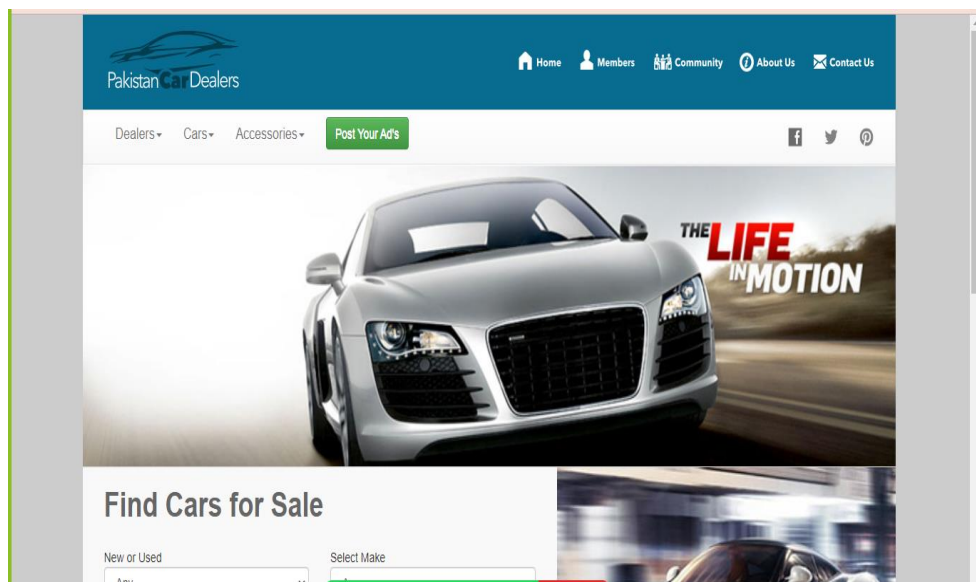


Figure 2.1 Pak dealers

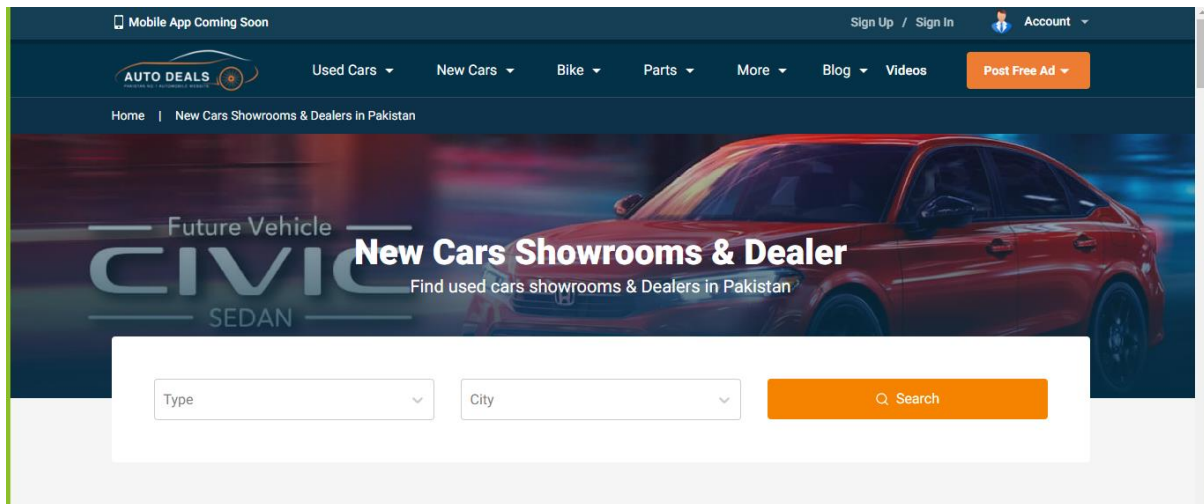


Figure 2.2 Auto deals

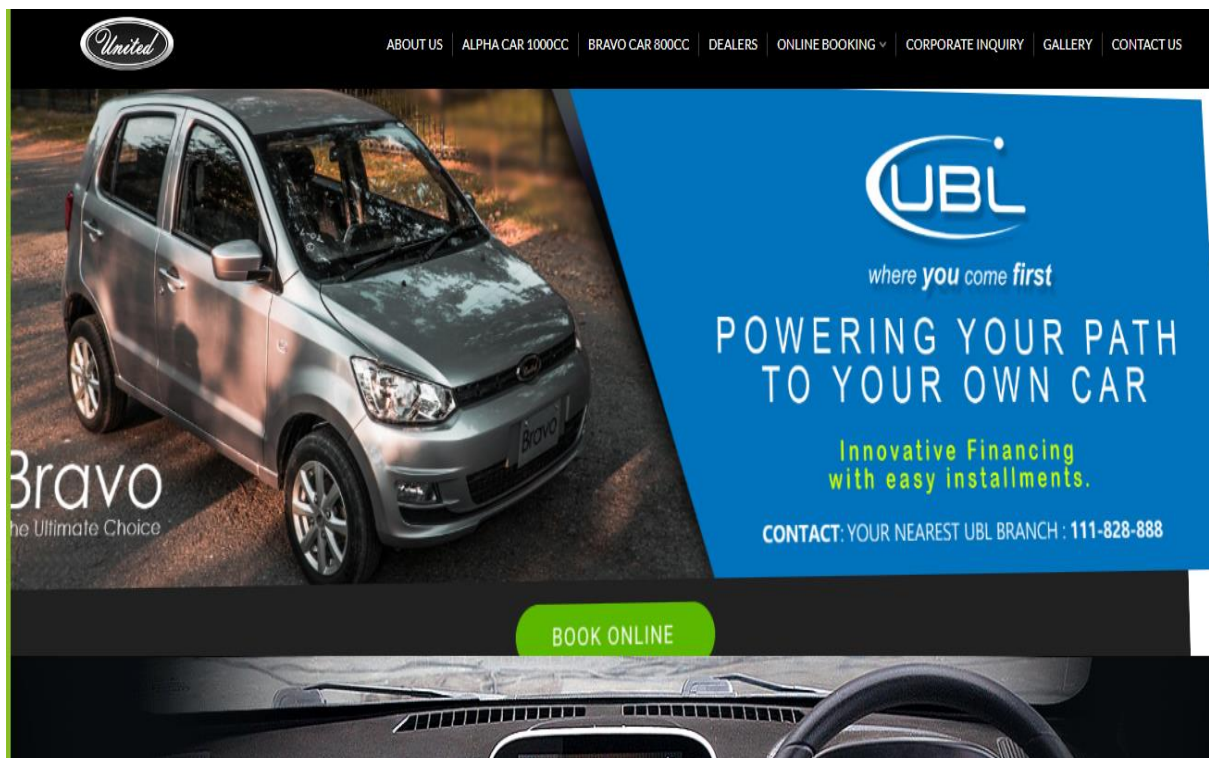


Figure 2.3 United Motors.pk

2.4. Need to Replace Existing Systems

Old systems do not provide a list of luxury cars which is our first priority, those systems do not provide a garage and management system (like spare parts and maintainers), and the

existing system doesn't provide a marketing strategy for brands. The existing systems do not have the feature of comparing vehicles.

We are going to develop a full fledged system to cover most requirements of the showroom and to facilitate the dealers, customers, garage and auto stores as much as possible.

2.5. Chapter Summary

In this chapter, we discuss about the existing system and drawbacks of the existing system. During the system analysis examination of end-user data is carried out in order to generate functional requirements of purposed system. The manual system was suffering from many problems. E-Showroom will provide users to get relief and easy to access by the users. Adding some extra feature, so the user can save his/her time.

Chapter 3

REQUIREMENT ENGINEERING

3.1. Proposed System

Our proposed system is to design an integrated E-showroom and luxury car services platform, offering a seamless online experience for exploring and purchasing top-tier brands like BMW, Bugatti, Audi, Lamborghini, Porsche, Tesla, and more. Alongside virtual showroom features, the platform provides comprehensive garage services, including repairs, engine works, and modifications. Free services are offered to guarantee cardholders, and strategic online marketing enhances brand visibility. Internship programs contribute to developing talent. This holistic approach aims to redefine the luxury car experience, providing customers with a one-stop solution for purchases, services, and enthusiast engagement.

Our proposed system, E-showroom and luxury car, is a web-based application. This system will tackle all the shortcomings of existing systems. It is a complete solution set for dealing with the luxury cars of any brand in Pakistan

3.2. Understanding the System

3.2.1. User Involvement

We make sure that the basic need of users should be fulfilled and satisfied with booking. According to the requirements of our users this website could improve their life by dealing easily with luxury car showroom.

3.2.2. Stakeholders

We know this website is for final-year project-related problems so the end-user will be:

- Customers
- Admin
- Car Manufacturers (BMW, Bugatti, Audi, Lamborghini, Porsche, Tesla, etc.)
- Technicians and Engineers
- Online Influencers and Collaborators
- Developer

3.2.3. Domain

- Web application
- Business, Automobiles

3.2.4. Needs of System**Table 3.1 User Needs of System**

SR #	Needs	Need ID
1	There must be admin and user.	N-01
2	Admin help the management of registration, Showroom, garage.	N-02
3	Admin can also change the setting if required	N-03
4	Admin must contain the data of user.	N-04
5	Admin must maintain all the details about car, brand, colour, model.	N-05

3.3 Requirement Engineering**3.3.1. Functional Requirements**

The proposed system provides the functionality to the user or as well as to the management, the functionality of the User, Showroom, Garage. The system is web-based application software. Thus, every module requires Internet connection by default.

Table 3.2 Functional Requirement 01

Functional Requirement ID	FR-01
Name	Admin Login
Description	Admin have to enter login credentials to sign in into the system.
Input	Email Password
Output	Error or successful Login.
Precondition	Admin must have login credentials.
Post condition	The message will receive after login.

Table 3.3. Functional Requirement 02

Functional Requirement ID	FR-02
Name	Add User (Customer)
Description	User can add himself/herself.
Input	User will give his/her required information.
Output	Error or login successfully.
Precondition	User is registered and his/her data saved in system.
Post condition	<ul style="list-style-type: none"> • Logged in. • System display dashboard.

Table 3.4 Functional Requirement 03

Functional Requirement ID	FR-03
Name	Delete User (Customer)
Description	User can delete account.
Input	User access the profile and select the delete account button.
Output	The profile of user will be deleted from database.
Precondition	User must have his/her profile before.
Post condition	Message will show after deletion on display.

Table 3.5 Functional Requirement 04

Functional Requirement ID	FR-04
Name	Update User (Customer)
Description	User can edit/update his/her profile data.
Input	User will give his/her requirements.
Output	New data will save in database.
Precondition	User must login into the system.
Post condition	Message will show after updating.

Table 3.6 Functional Requirement 05

Functional Requirement ID	FR-05
Name	Book Car (Customer)
Description	User (Customer) can book car
Input	User (Customer) have to visit the category / manufacture's profile. Select the category button.
Output	Booking details will save in the system.
Precondition	User must login into the system.
Post condition	User will get the message of booked car.

Table 3.7 Functional Requirement 06

Functional Requirement ID	FR-06
Name	Brands Registration Portal
Description	Brands can register himself in the system.
Input	User will give his/her required information.
Output	After Admin's confirmation, Brand will register into the system.
Precondition	Brand must login into the system.
Post condition	Brand will receive the message of registration.

Table 3.8 Functional Requirement 07

Functional Requirement ID	FR-07
Name	Update User (Brand)
Description	User (Brand) can edit/update his/her profile data.
Input	User will give his/her requirements.
Output	New data will save in database.
Precondition	User must have an account in the system.
Post condition	Message will show after updating.

Table 3.9 Functional Requirement 08

Functional Requirement ID	FR-08
Name	Delete User (Brand)
Description	User (Brand) can delete account.
Input	User has to access the profile and select the delete account button.
Output	The profile of user will be deleted from the database.
Precondition	User must have an account.
Post condition	Message will show after deleting.

Table 3.10 Functional Requirement 09

Functional Requirement ID	FR-9
Name	Garage Stores Registration
Description	Owner will register the store.
Input	Owner will give required information.
Output	Show the required information
Precondition	Garage stores should be register.
Post condition	The message will receive after registration.

Table 3.11 Functional Requirement 10

Functional Requirement ID	FR-10
Name	Garage Stores
Description	Stores holder can add the data of available Items.
Input	Add every new item and remove the items which has end at store.
Output	Show the list of items.
Precondition	Garage stores should be register.
Post condition	The message will receive after registration.

Table 3.12 Functional Requirement 11

Functional Requirement ID	FR-11
Name	Parts Usage
Description	User can search for an item and can see the usage of that items.
Input	User can search for an item.
Output	User will see the list of items and garage stores.
Precondition	The garage store operator has to manage the details of items.
Post condition	Receive information about the items.

3.3.2. Non-Functional Requirements

3.3.2.1. Interaction

The application consists of simple yet elegant user interface (UI). The application is designed to minimize any sorts of confusion. All the components are self-explanatory and the text descriptions provide the user additional helping instructions.

3.3.2.2. Security

Login system is employed to protect the system against breaches, SQL injections, and other vulnerabilities. Only people registered in the educational institute's database can login the system. Thus, no one from outside can enter the system.

3.3.2.3. Privacy

All the data that the application utilizes is stored in a database. By ensuring the security via login, only the admin has the access to most of the data from the database. The admin is designated by the officials of the showroom itself, therefore, the privacy of official records and user's data is safe.

3.3.2.4. Performance

The frameworks being used in the development of this application are maintained by Facebook and a global community of developers ensuring the performance of its code. Furthermore, the code of our application follows the standard practices and thus, it is highly optimized for performance (quick response).

3.4.Gantt Chart

Gantt chart is a commonly used graphical depiction of a project schedule. It's a type of bar chart showing the start and finish dates of a project's Gantt chart is a visualization that helps in scheduling, managing, and monitoring specific tasks and resources in a project.

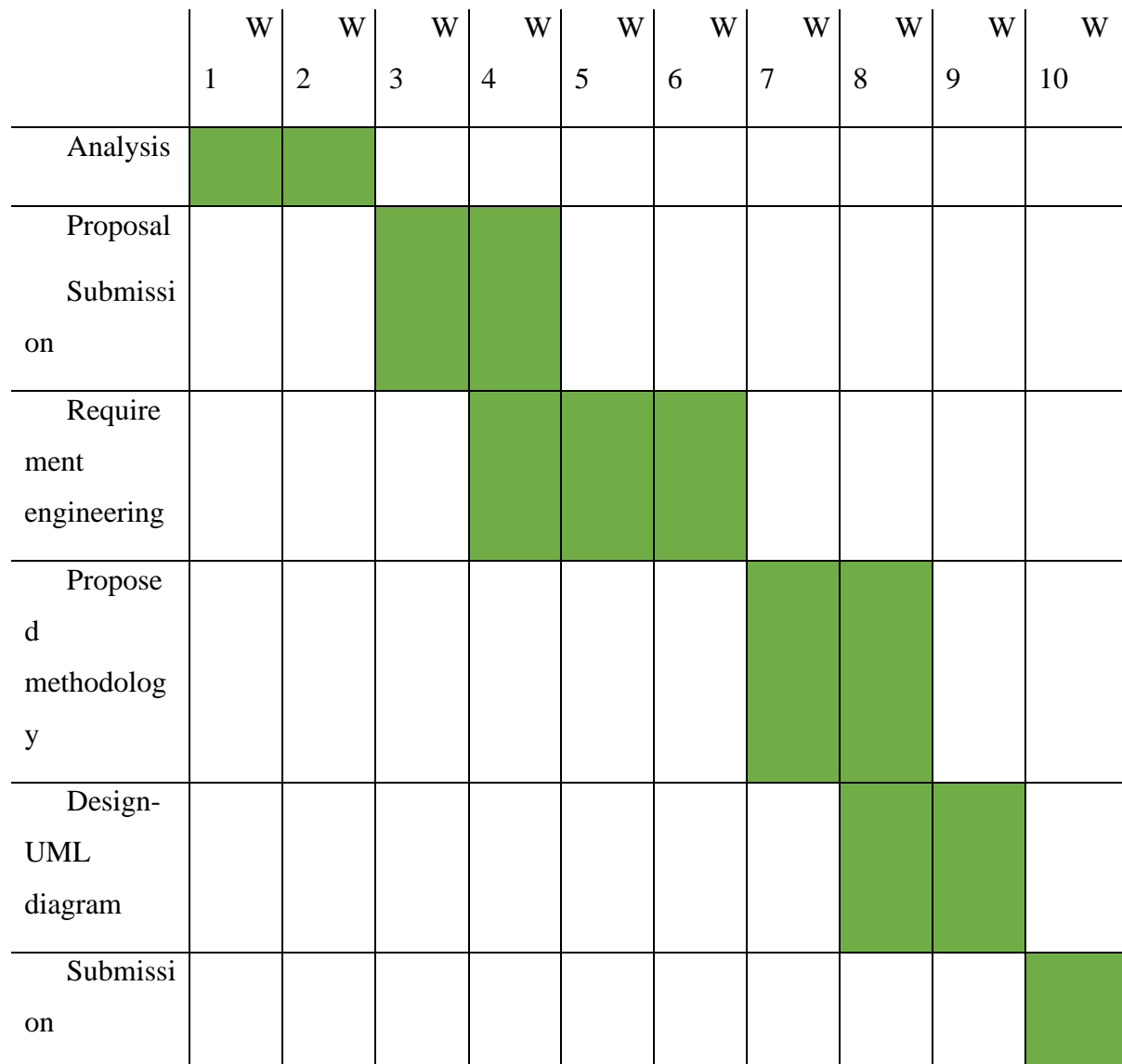


Figure 3.1 Gantt Chart

3.5.Hurdles in Optimizing the Current System

- To make performance of application accurate.
- To maintain the performance of application brilliantly.
- Creating user friendly interface.
- Handle too much records and automatic telling system.

- To make it faster.

3.6.Chapter Summary

The proposed system is a web-based platform aiming to redefine the luxury car experience in Pakistan. It integrates E-showroom features, luxury car services, and comprehensive garage services, providing users with a seamless online solution for exploring and purchasing top-tier brands. The stakeholders include customers, admins, car manufacturers, technicians, engineers, and online influencers. The system's functional requirements cover aspects like admin login, user management, brand registration, and garage store operations. Non-functional requirements focus on interaction, security, privacy, and performance. The Gantt chart outlines the project schedule. However, challenges include optimizing performance, creating a user-friendly interface, and improving system speed. Despite these hurdles, the proposed system holds the potential to revolutionize luxury car experiences through its comprehensive and integrated approach.

Chapter 4

DESIGN

4.1. Software Process Model

The software development models are the different methods or approaches that are adopted for project development based on the project's goals and objectives. There are a variety of development life cycle models that have been created to meet various requirements. So, in this project, we've opted a incremental model to facilitate us in our task. The development process is separated into distinct phases in this technique, and the output of one phase serves as the input for the following step in a timely order. Each model has its own benefits and limitations.

Most commonly practiced models are:

- Waterfall Model
- Iterative Model
- Agile Model
- Spiral Model
- Incremental Model

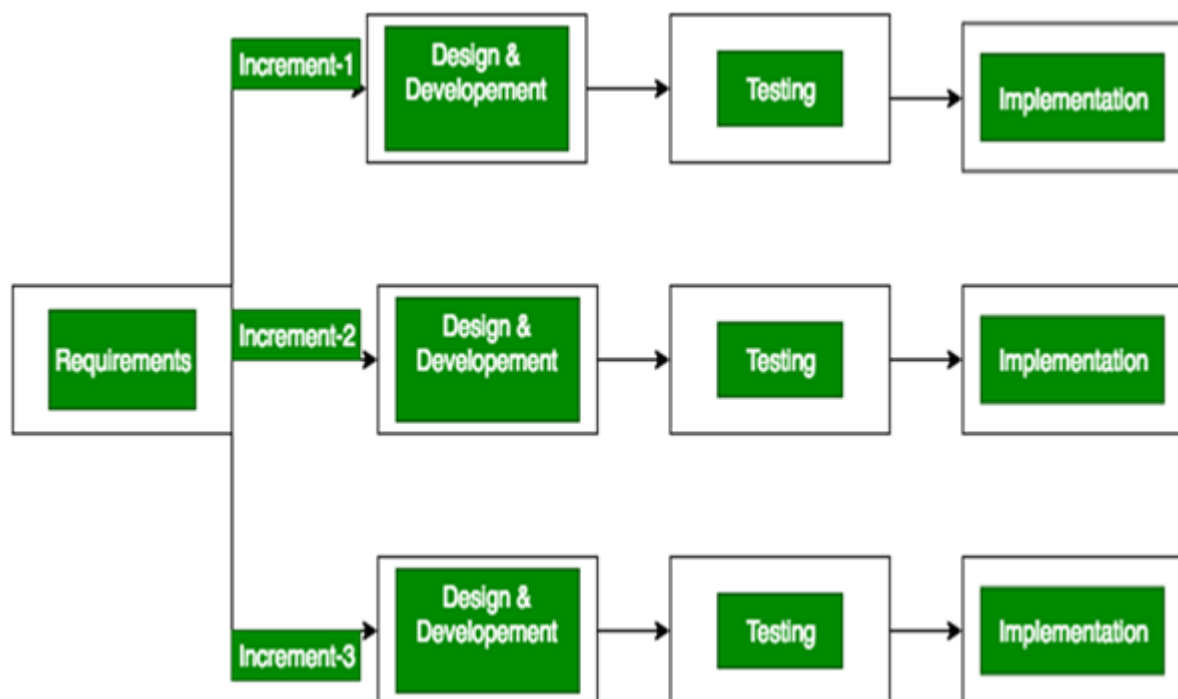


Figure 4.1 Incremental Model Diagram

4.1.1. Benefits of Selected Model

The adoption of the Incremental model is most suited in the following situations:

- Requirements are well-documented, well-defined, and well-define.
- Technology is well-understood and unchanging.
- There are no prerequisites that are unclear.
- The product is supported by a large number of resources with the necessary experience.
- A timetable may be created with deadlines for each step of development, and a product can be guided through the various phases of the development process one by one.

From idea to design, execution, testing, installation, and troubleshooting, development leads to operation and maintenance. Each stage of development must be completed in a specific order.

4.1.2. Limitations of Selected Model

The limitations of the suggested system are listed below:

- Until late in the life cycle, no functioning software is developed.
- There is a lot of danger and ambiguity.
- This isn't an appropriate approach for projects that are both complicated and object-oriented.
- Ineffective model for long-term initiatives.
- Measuring progress within phases is tough.
- Unable to adapt to changing need.

Changing the scope of a project during its life cycle might lead to its termination

4.1.3. Benefits of Selected Model

The adoption of the Incremental model is most suited in the following situations:

- Requirements are well-documented, well-defined, and well-define.
- Technology is well-understood and unchanging.
- There are no prerequisites that are unclear.
- The product is supported by a large number of resources with the necessary experience.
- A timetable may be created with deadlines for each step of development, and a product can be guided through the various phases of the development process one by one.

From idea to design, execution, testing, installation, and troubleshooting, development leads to operation and maintenance. Each stage of development must be completed in a specific order.

4.1.4. Limitations of Selected Model

The limitations of the suggested system are listed below:

- Until late in the life cycle, no functioning software is developed.
- There is a lot of danger and ambiguity.
- This isn't an appropriate approach for projects that are both complicated and object-oriented.
- Ineffective model for long-term initiatives.
- Measuring progress within phases is tough.
- Unable to adapt to changing need.

Changing the scope of a project during its life cycle might lead to its termination

4.2.Design

4.2.1. Methodology of the Proposed System

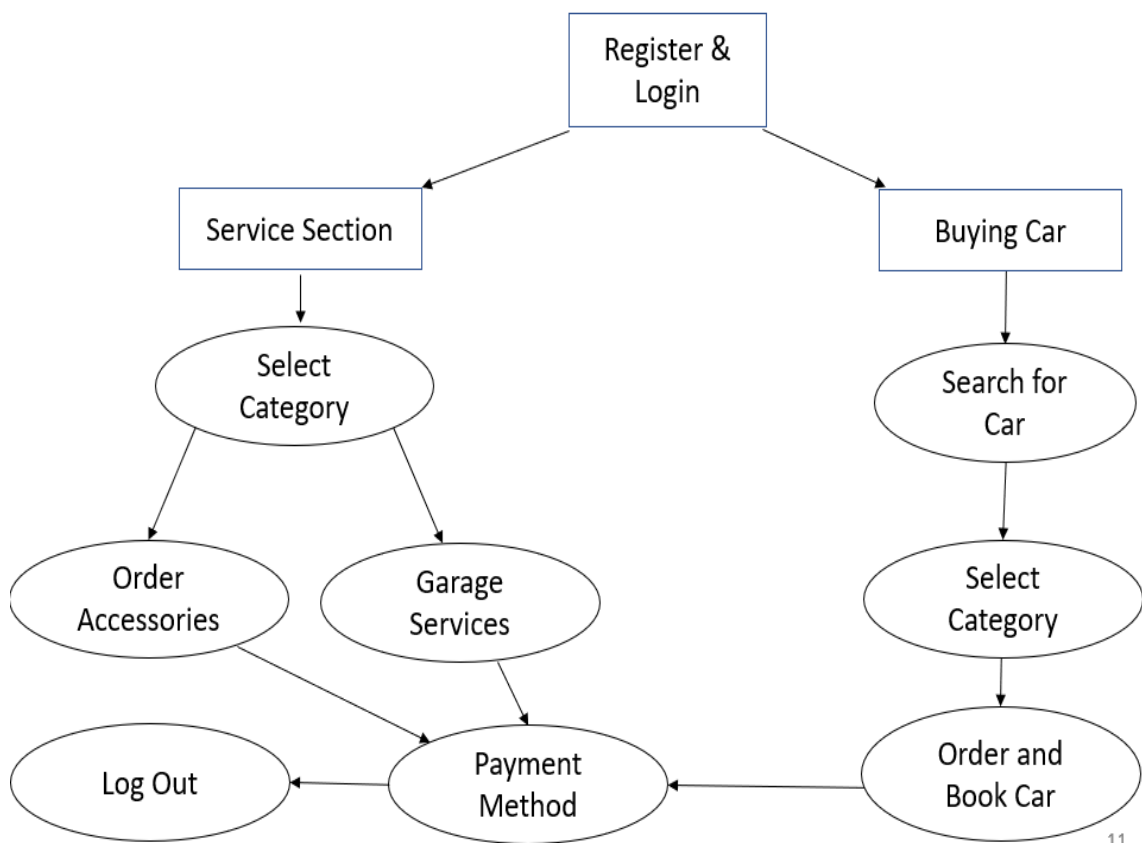


Figure 4.2 Methodology

4.2.2. Entity Relationship Diagram

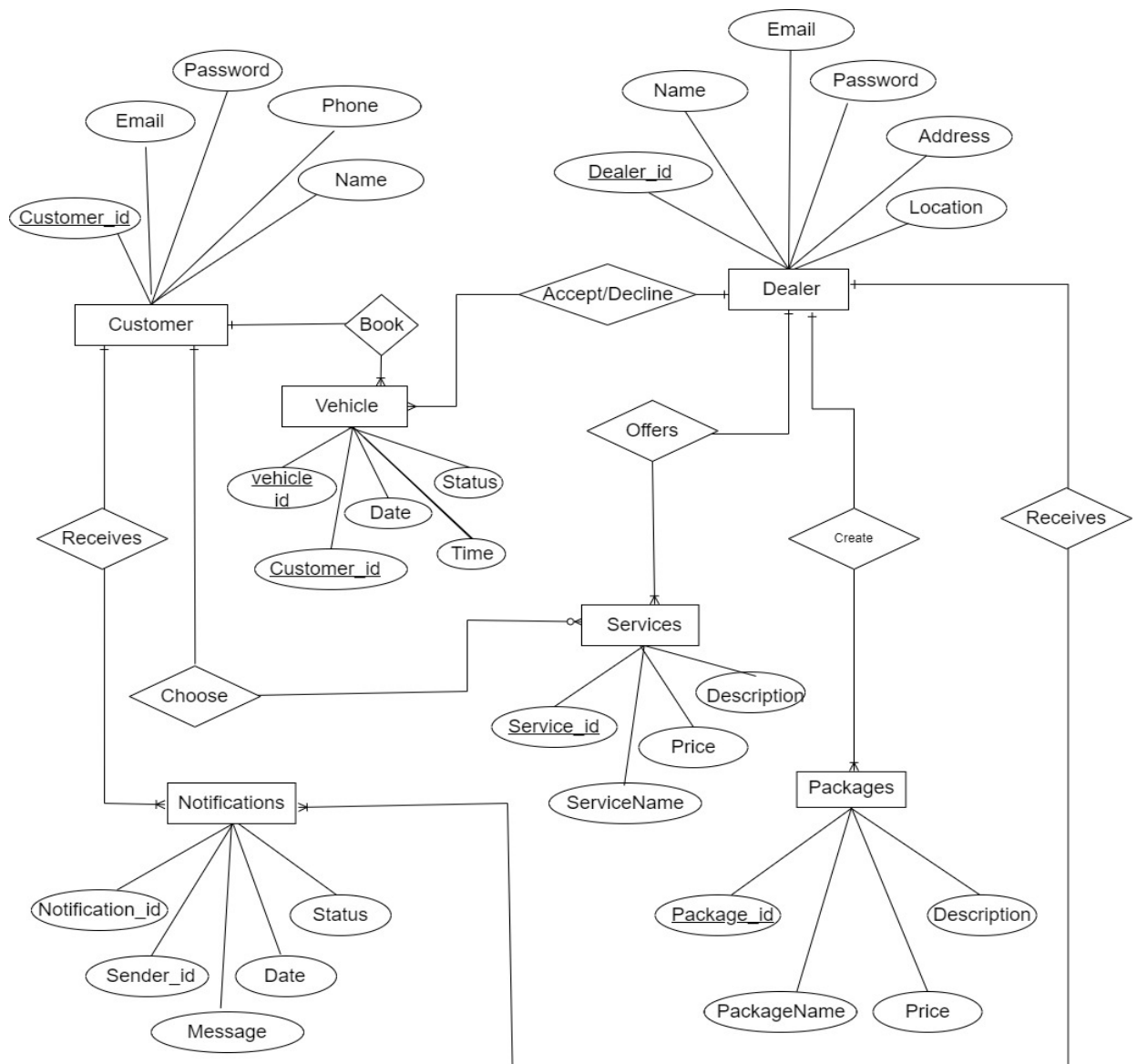


Figure 4.3 ER-Diagram

4.2.3. UML Diagrams

UML stands for Unified Modelling Language. It's a rich language to model software solutions, application structures, system behaviour and business processes. There are 14 UML

diagram to help you model this behaviour. UML is standardized modelling language that can be used across different programming language and development process so, the majority of software developer will be understand it and be able to apply it to their work.

4.2.3.1. Use Case Diagram of the System

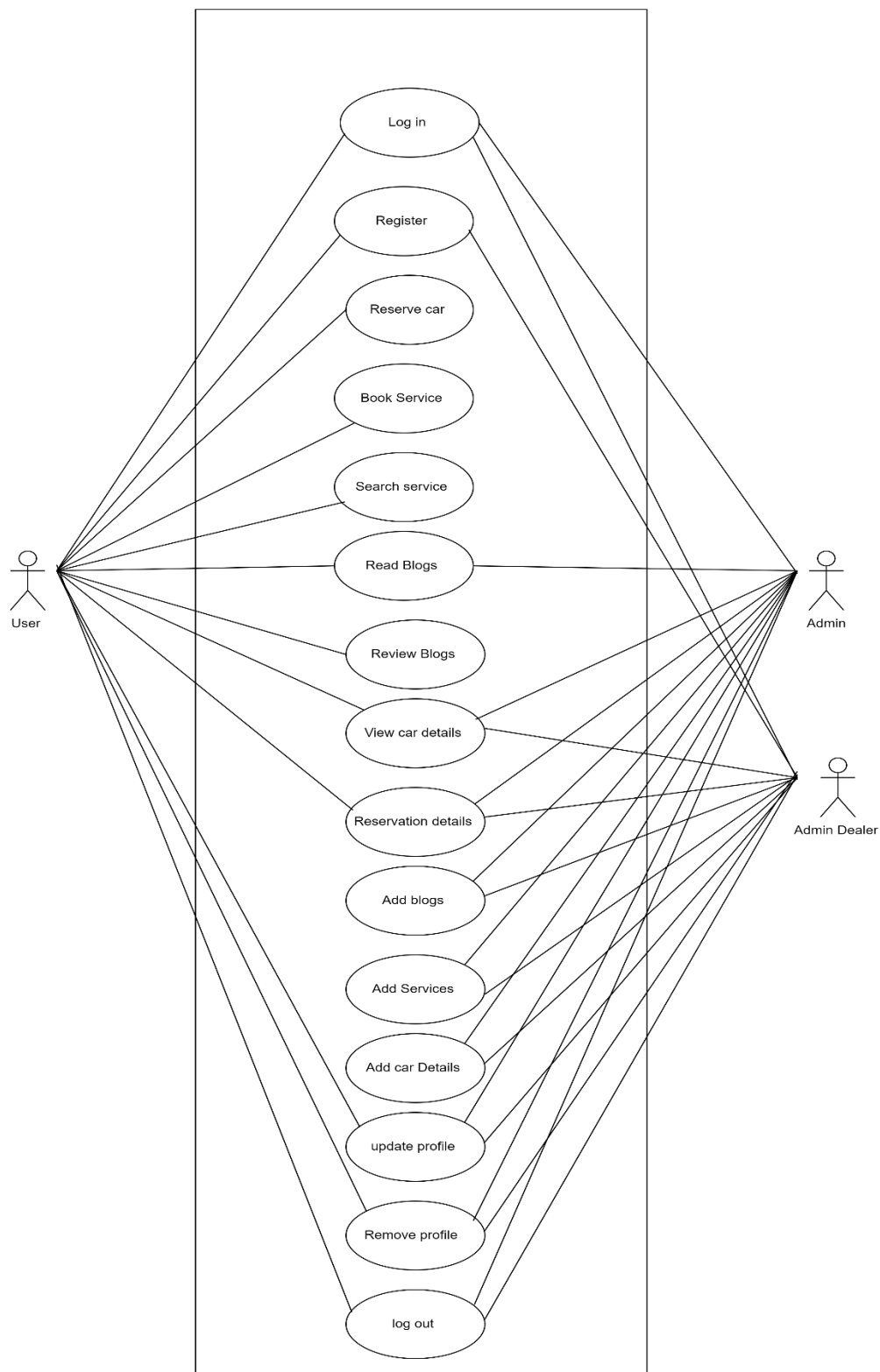


Figure 4.4 Use Case Diagram

4.2.3.2. Class Diagram of the System

A class diagram is a diagram used in designing and modelling software to describe classes and their relationships. The diagram shows the names and attributes of the classes, connections between the classes, and sometimes also the methods of the classes.

4.2.3.3. Activity Diagram of the System

Activity diagram is another important behavioural diagram in UML diagram to describe dynamic aspects of the system. Activity diagram is essentially an advanced version of flow chart that modelling the flow from one activity to another activity.

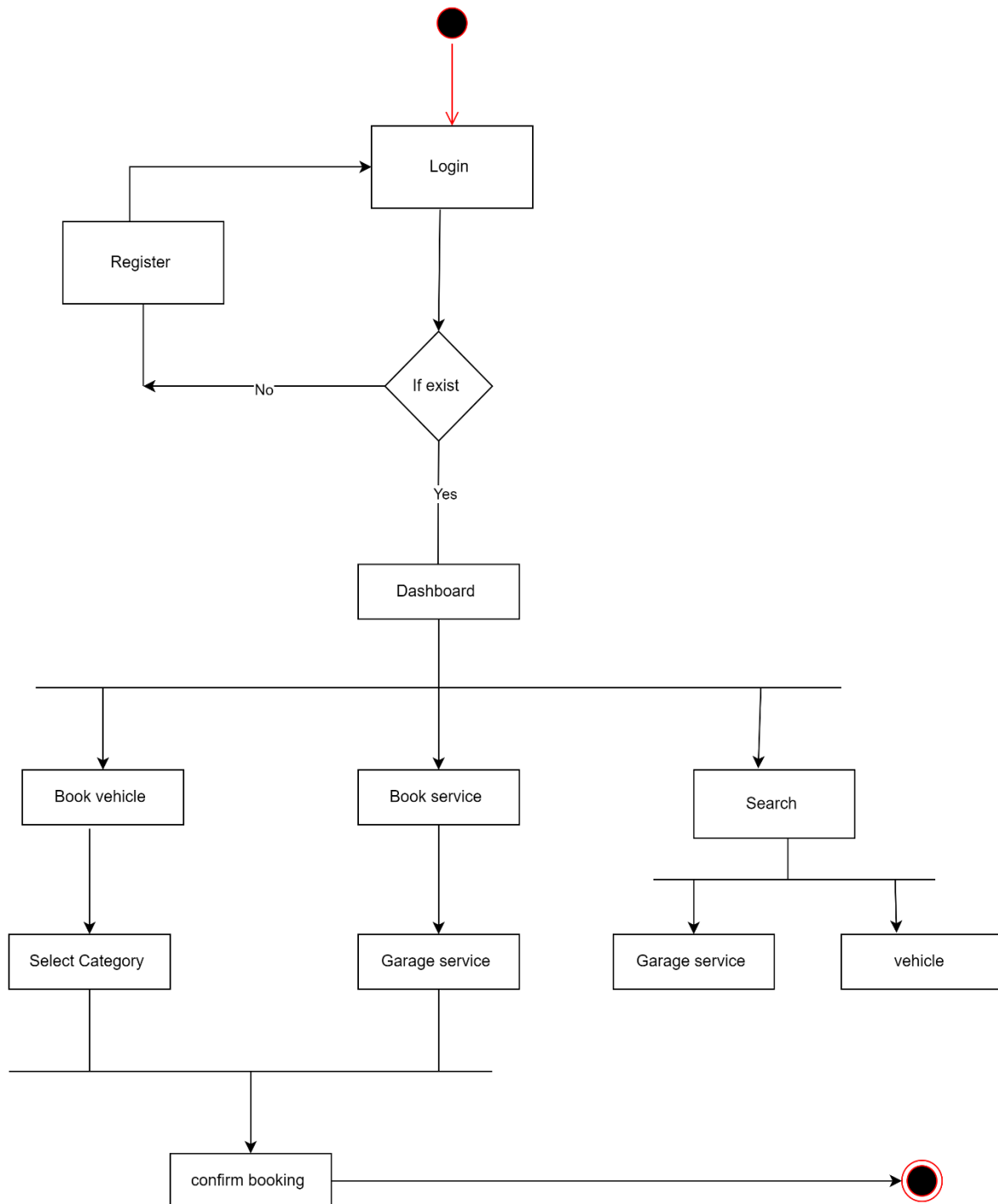


Figure 4.5 Activity Diagram

4.2.3.4. Sequence Diagram of the System

UML Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent.

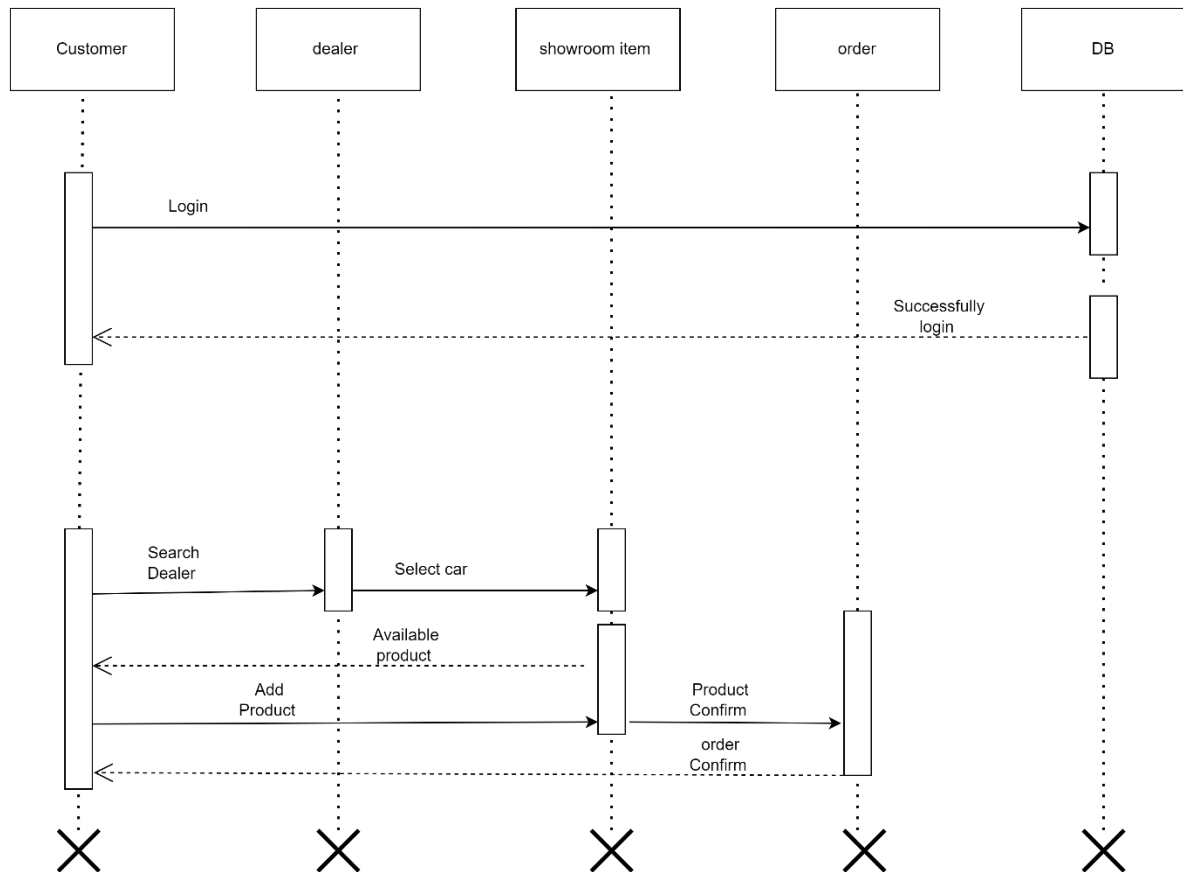


Figure 4.7 Activity Diagram

4.2.3.5. Component Diagram of the System

UML Component diagrams are used in modelling the physical aspects of object-oriented systems that are used for visualizing, specifying, and documenting component-based systems and also for constructing executable systems through forward and reverse engineering. Component diagrams are essentially class diagrams that focus on a system's components that often used to model the static implementation view of a system.

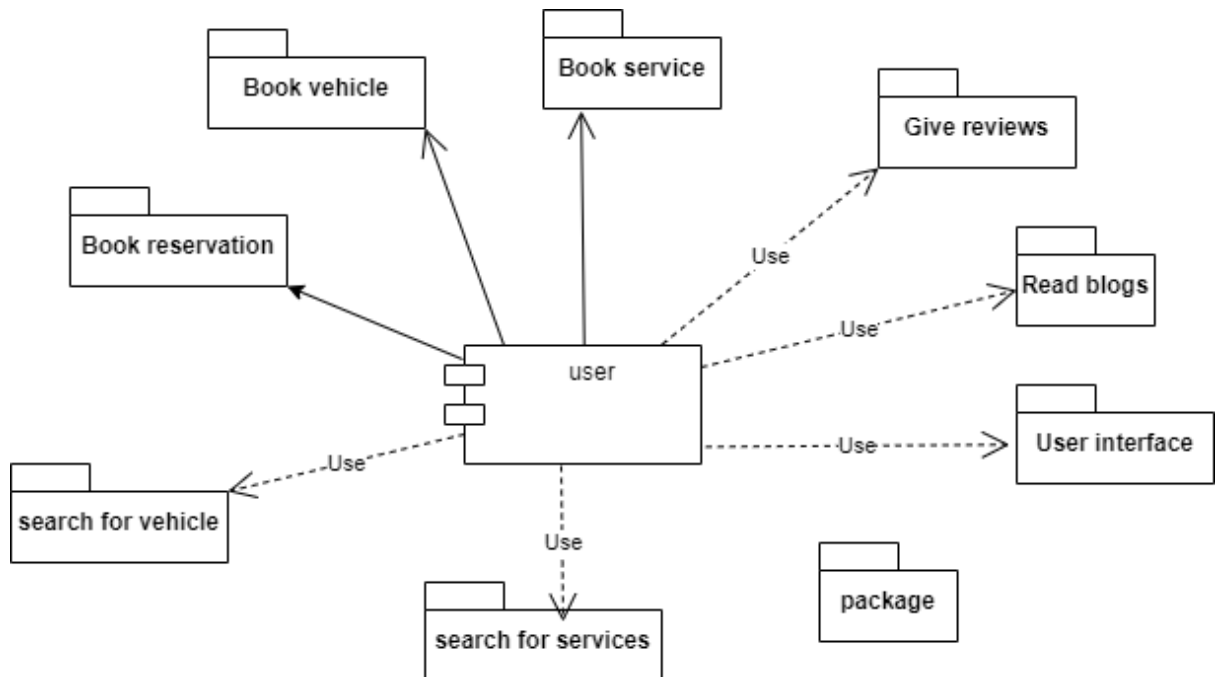


Figure 4.6 Component Diagram

4.3.Chapter Summary

In this Chapter we discuss about the Software process model which tell the complete detail of model in such a way the book reviewer. The methodology model shows what the specification of system is. Incremental model has designed. Each iteration passes through the requirements, design, coding and testing phases. And each subsequent release of the system adds function to the previous release until all designed functionalities has been implemented. ER-diagram has design to show Entity relation of Whole system and its user.

Chapter 5

DATABASE

5.1. Database Introduction

Database is a physical container for collections. Each database gets its own set of files on the file system. MySQL is a popular open-source relational database management system (RDBMS) that is widely used for managing and manipulating structured data.

MySQL is one of the most widely used databases in web applications and is known for its scalability, performance, and ease of use.

Following are the benefits of MY SQL database:

- **Relational Database:** MySQL organizes data into tables with rows and columns, allowing for the establishment of relationships between tables using primary and foreign keys.
- **SQL Language:** MySQL uses SQL for managing and querying data, providing a standardized way to interact with the database.
- **Efficient Data Retrieval and Manipulation:** With its relational model and SQL capabilities, MySQL enables efficient retrieval and manipulation of data.
- **Open Source:** MySQL is an open-source RDBMS, which means it is freely available and can be modified and distributed.

5.2. Selected Database

MySQL is an open-source relational database management system (RDBMS) that is widely used for managing structured data. It is one of the most popular database systems in the world, known for its reliability, scalability, and performance. MySQL is developed, distributed, and supported by Oracle Corporation.

5.2.1. Reasons for Selection of the Database

5.2.1.1. Scalability

MySQL DB is horizontally scalable, meaning you can add more servers to increase capacity, making it a good choice for applications that need to handle large amounts of data.

5.2.1.2. Flexibility

MySQL DB's document-oriented data model allows for flexible and dynamic schema design, making it easier to accommodate changes to data requirements over time.

5.2.1.3. High Performance

MySQL DB is optimized for high-performance queries and supports rich indexing capabilities, allowing for fast data retrieval.

5.2.1.4. Security of data

MySQL DB provides various features, such as authentication, access control, and encryption, to secure your MySQL DB deployments. Some key security features include.

- Authentication
- Authorization
- Encryption

5.2.1.5. Data integrity

Multi-document transactions allow an application developer to write code that will start a transaction, modify multiple documents, and then commit that transaction. The commit will try and do all those changes unless it finds a problem. If an error occurs, the data in the database will remain unchanged.

5.2.2. Benefits of the Selected Database

MySQL is widely used in various applications and industries, especially in web development, due to its scalability, performance, and ease of use. MySQL offers security features such as user authentication, access control, and encryption to protect data. MySQL supports features like replication, clustering, and partitioning to scale horizontally and handle large volumes of data and high traffic loads. MySQL has a large and active community of users and developers, providing support, resources, and updates.

5.2.2.1. Scalability

MySQL DB is designed to be highly scalable and can easily handle large amounts of data, making it ideal for applications with high data volume and high traffic.

5.2.2.2. Speed

MySQL DB can provide faster query responses compared to traditional relational databases. It uses a document-oriented model that allows for fast data access and retrieval.

5.2.2.3. Replication and High Availability

MySQL DB offers built-in replication and failover features, which means that if one node fails, another can quickly take over, ensuring high availability and data redundancy

5.2.2.4. Support for a wide range of data types

MySQL DB supports a wide range of data types, including text, numeric, and geospatial data, making it ideal for use in applications that require complex data processing and analysis.

5.2.2.5. Easy integration

MySQL DB is easy to integrate with other technologies, making it a popular choice for modern web applications and big data solutions.

5.2.3. Limitations of the Selected Database

Maximum Table Size: Limited by storage engine, e.g., MyISAM: 256TB, InnoDB: 64TB.

Maximum Column Length: Varies by data type, e.g., VARCHAR: 65,535 characters.

Maximum Index Length: InnoDB: 767 bytes for single-column index, 3072 bytes for multi-column index.

Maximum Connections: Determined by `max_connections` configuration parameter.

Maximum Database Size: No fixed limit, but constrained by available disk space and OS/file system limitations.

5.3.Database Queries

A database query is a request for a database's data so we can retrieve or manipulate it. It is a similar action that is most closely associated with some sort of CRUD (create, read, update, delete) function. A database query is a request to access data from a database to manipulate it or retrieve it. This allows us to perform logic with the information we get in response to the query.

5.4.Database Tables

Table 5.1 Admin

ID	This is unique ID.
Name	This contains user name.
Email	This contains email of user.
Password	This contains Message of user.

Table 5.2 Dealer

ID	This is unique ID.
Name	This contains user name.
Email	This contains email of user.
Password	This contains Message of user.
Phone	This contains Dealer Phone Number.
Address	This Contain Dealer Address.

Table 5.3 Customer

ID	This is unique ID.
Name	This contains user name.
Email	This contains email of user.
Password	This contains Message of user.
Details	This contains Customer's Details.
Phone	This contains Customer's Phone Number.
Address	This Contains Customer's Address.

Table 5.4 Service

ID	This is unique ID.
Name	This contains User name.
Details	This contains Service Details.
Status	This contains Service status.
Category	This Contain Service category. .

Table 5.5 showroom

ID	This is unique ID.
Name	This contains user name.
Email	This contains email of user.
Password	This contains Message of user.
Details	This contains Showroom Details.
Phone	This contains Showroom Phone Number.
Address	This Contains Showroom Address.

5.5.Database Schema Diagram

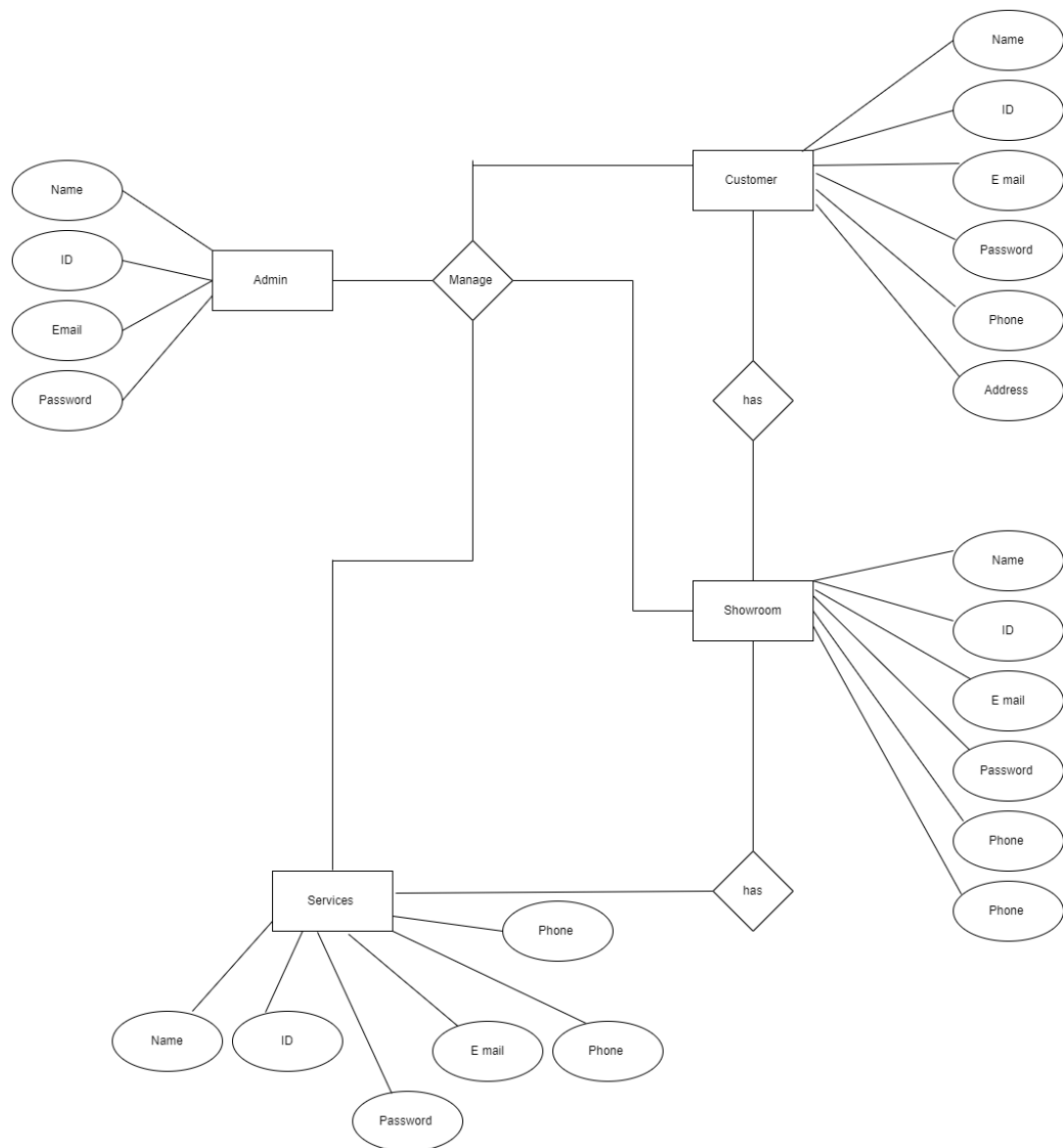


Figure 5.1 Schema Diagram

5.6.Chapter Summary

In this chapter we have discussed about the database use in this project. MySQL DB is updated database of this time use in Laravel. It has many benefits. It is more Flexible. Authentication, Authorization and Encryption are some good features of MySQL DB security. Tables of Database show the details saved in database of the website.

Chapter 6

DEVELOPMENT AND IMPLEMENTATION

6.1. Development of the Computer Program

Program development is the process of creating application programs. Program development life cycle (PDLC) The process containing the five phases of program development: analysing, designing, coding, debugging and testing, and implementing and maintaining application software.

The following are six steps in the Program Development Life Cycle:

- Analyse the problem. The computer user must figure out the problem, then decide how to resolve the problem - choose a program.
- Design the program. A flow chart is important to use during this step of the PDLC. This is a visual diagram of the flow containing the program. This step will help you break down the problem.
- Code the program. This is using the language of programming to write the lines of code. The code is called the listing or the source code. The computer user will run an object code for this step.
- Debug the program. The computer user must debug. This is the process of finding the "bugs" on the computer. The bugs are important to find because this is known as errors in a program.
- Formalize the solution. One must run the program to make sure there are no syntax and logic errors. Syntax are grammatical errors and logic errors are incorrect results.

Document and maintain the program. This step is the final step of gathering everything together. Internal documentation is involved in this step because it explains the reasoning one might have made a change in the program or how to write a program.

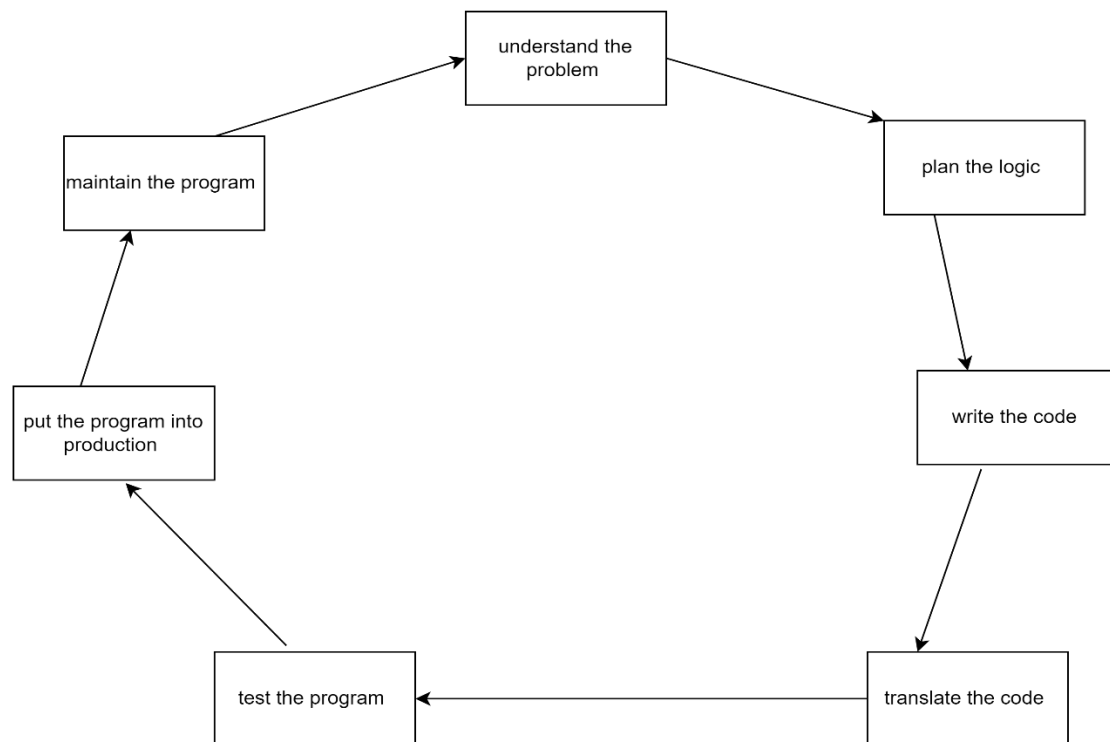


Figure 6.1 PDLC

6.2. Implementation Strategy

Your vendor is a key stakeholder in your software implementation plan. The extent to (and cost at) which your vendor will support your implementation varies—but that doesn't mean you shouldn't squeeze as much assistance as possible from them.

6.2.1. Control your scope, or it will control you

Your needs document defines the capabilities your new system needs so that you can focus on what's most important during your selection stage.

6.2.2. Assign realistic teams to drive software implementation plans

The next critical step in your implementation journey is assembling the team(s) necessary for success. The makeup of an implementation team will vary for every business, depending on the unique needs of your business and the scale of implementation.

6.2.3. Focus on continuous improvements

Though you're probably elated to get running with your new software, you need to master walking with it first. It might take some time to get up to full speed, and that's fine—just continue working better with the software each day. Training is a central pillar in continuous

improvement. The key is to avoid one-size-fits-all training and, similar to the vendor demos, tailor each session to the needs of your stakeholders.

6.3.Tools Selection

6.3.1. Hardware

- Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz 2.40 GHz
- Ram 8GB DDR3

6.3.2. Software

- Web Browser
- Window 10 OS
- VS Code
- MySQL DB xampp
- Live wire

6.4.Coding

```
1  use Illuminate\Support\Facades\Route;  
2  
3  use App\Http\Controllers\CartController;  
4  use App\Http\Controllers\VisitorController;  
5  use App\Http\Controllers\admin\CarsController;  
6  use App\Http\Controllers\admin\AdminController;  
7  use App\Http\Controllers\admin\ServiceController;  
8  use App\Http\Controllers\auth\AuthenticationController;  
9  use App\Http\Controllers\CheckoutController;  
10 use App\Http\Controllers\OrderController;  
11
```

Figure 6.1 App Js

```
1  <?php
2
3  namespace App\Http\Controllers\admin;
4
5  use App\Models\User;
6  use App\Models\Order;
7  use App\Models\Query;
8  use Illuminate\Http\Request;
9  use App\Http\Controllers\Controller;
10 use App\Models\CarShowroom;
11
```

Figure 6.2 App Js

```
2
3  namespace App\Http\Controllers;
4
5  use Stripe\Stripe;
6  use App\Models\Cart;
7  use App\Models\CarShowroom;
8  use Illuminate\Http\Request;
9  use Stripe\Checkout\Session;
10 use Illuminate\Support\Facades\Auth;
11
```

Figure 6.3 App.Js

```

45
46     public function index()
47     {
48         return view('admin.pages.auth.login');
49     }
50
51     public function authentication(Request $request)
52     {
53
54         $request->validate([
55             'email' => 'required',
56             'password' => 'required',
57         ]);
58
59         $userCredentials = $request->only('email', 'password');
60         $adminCredentials = $request->only('email', 'password');
61         // Attempt to authenticate the user
62         if (Auth::attempt($userCredentials)) {
63             return redirect()->route('index');
64         }
65         if (Auth::guard('admin')->attempt($adminCredentials)) {
66             return redirect()->route('dashboard');
67         }
68         return redirect()->back()->with('message', 'Invalid Credentials');
69     }
70     public function logout()
71     {
72         if (Auth::guard('web')->check()) {
73             Auth::guard('web')->logout();
74         } elseif (Auth::guard('admin')->check()) {
75             Auth::guard('admin')->logout();
76         }
77     }

```

Figure 6.4 Login

```

12
13     class AuthenticationController extends Controller
14     {
15
16         public function registerpage()
17         {
18             return view('admin.pages.auth.register');
19         }
20         protected function createuser(Request $request)
21         {
22             // Validate the user input
23             $validatedData = $request->validate([
24                 'name' => 'required|string|max:255',
25                 'email' => 'required|string|email|max:255|unique:users',
26                 'password' => 'required|string|min:8',
27             ]);
28
29             // Create the user if validation passes
30             $user = User::create([
31                 'name' => $validatedData['name'],
32                 'email' => $validatedData['email'],
33                 'password' => Hash::make($validatedData['password']),
34             ]);
35
36             // Check if user creation was successful
37             if ($user) {
38                 return redirect()->route('login')->with('message', 'User Created Successfully');
39             } else {
40                 // Handle the case where user creation failed (e.g., database error)
41                 return redirect()->route('register')->with('error', 'User Creation Failed');
42             }
43         }

```

Figure 6.5 Sign Up


```

11
12 class VisitorController extends Controller
13 {
14     public function index()
15     {
16         return view('visitor.pages.index');
17     }
18     public function services()
19     {
20         $services = Service::all();
21         return view('visitor.pages.service', compact('services'));
22     }
23     public function showroom()
24     {
25         $record = CarShowroom::all();
26         return view('visitor.pages.car', compact('record'));
27     }
28     public function contact()
29     {
30         return view('visitor.pages.contact');
31     }
32     public function query(Request $request)
33     {
34         $data = $request->validate([
35             'name' => 'required|alpha',
36             'email' => 'required',
37             'phonenumber' => 'required|numeric|digits:11',
38             'query' => 'required',
39             'address' => 'required',
40         ]);
41         $data = $request->all();
42         Query::create($data);
43         return redirect()->back()->with('message', 'Your message has been send');
44     }

```

Figure 6.6 Home.Js

```

45
46     public function cart()
47     {
48         return view('visitor.includes.navbar');
49     }
50
51     public function visitororder()
52     {
53         // Check if the user is authenticated
54         if (Auth::check()) {
55             // Retrieve the currently authenticated user's ID
56             $user_id = auth()->user()->id;
57
58             // Retrieve orders for the authenticated user
59             $orders = Order::where('user_id', $user_id)->get();
60
61             // Pass the orders to the view
62             return view('visitor.pages.order', compact('orders'));
63         }
64     }
65     public function cancelorder($id)
66     {
67         $order = Order::find($id);
68         $order->delivery_status = 'you canceled the Order';
69         $order->save();
70         return redirect()->back()->with('message', 'Order canceled');
71     }
72 }
73

```

Figure 6.7 Home.Js

```

24 // Login Routes
25 Route::get('/register/page', [AuthenticationController::class, 'registerpage'])->name('register.page');
26 Route::post('/create/user', [AuthenticationController::class, 'createuser'])->name('create.user');
27 Route::get('/login', [AuthenticationController::class, 'index'])->name('login');
28 Route::post('/authentication', [AuthenticationController::class, 'authentication'])->name('authentication');
29 Route::get('/logout', [AuthenticationController::class, 'logout'])->name('logout');
30 // Admin Routes
31 Route::middleware(['auth:admin'])->group(function () {
32     Route::get('/dashboard', [AdminController::class, 'dashboard'])->name('dashboard');
33     // Car module
34     Route::get('/view/cars', [CarsController::class, 'index'])->name('view.cars');
35     Route::any('/create/cars/info/{id?}', [CarsController::class, 'create'])->name('create.carsinfo');
36     Route::get('/delete/cars/info/{id?}', [CarsController::class, 'destroy'])->name('delete.carsinfo');
37     // Service module routes
38     Route::get('/view/services', [ServiceController::class, 'index'])->name('view.services');
39     Route::any('/create/services/info/{id?}', [ServiceController::class, 'create'])->name('create.services');
40     Route::get('/delete/services/info/{id?}', [ServiceController::class, 'destroy'])->name('delete.services');
41     // Order manage
42     Route::get('/manage/order', [AdminController::class, 'order'])->name('order.manage');
43     Route::get('/query/list', [AdminController::class, 'query_list'])->name('query.list');
44     Route::get('/order/delivered/{id}', [AdminController::class, 'orderDelivered'])->name('order.delivered');
45     Route::get('/order/delete/{id}', [OrderController::class, 'destroy'])->name('delete.order');
46 });
47 Route::middleware(['auth:web'])->group(function () {
48     Route::post('/query', [VisitorController::class, 'query'])->name('query');
49     Route::post('/add-to-cart', [CartController::class, 'index'])->name('cart.add');
50     Route::get('/checkout', [CheckoutController::class, 'index'])->name('check.out');
51     Route::get('/checkout', [CheckoutController::class, 'viewCheckout'])->name('view.checkout');
52     Route::post('/confirm-payment', [CheckoutController::class, 'checkout'])->name('payment.using.stripe');
53     Route::get('/order/user', [VisitorController::class, 'visitororder'])->name('visitor.order');
54     Route::get('/cancel/order/{id}', [VisitorController::class, 'cancelorder'])->name('visitor.cancelorder');
55 });
56 // User Routes

```

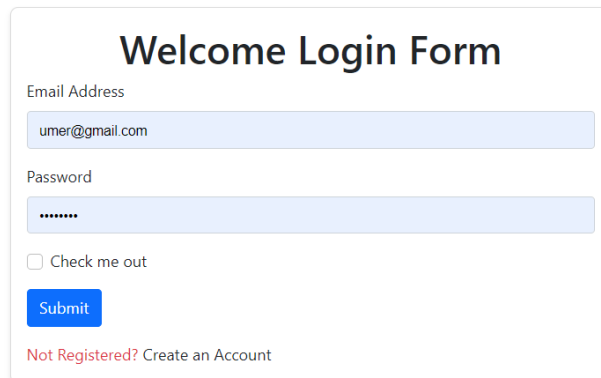
Figure 6.8 Index.Js

6



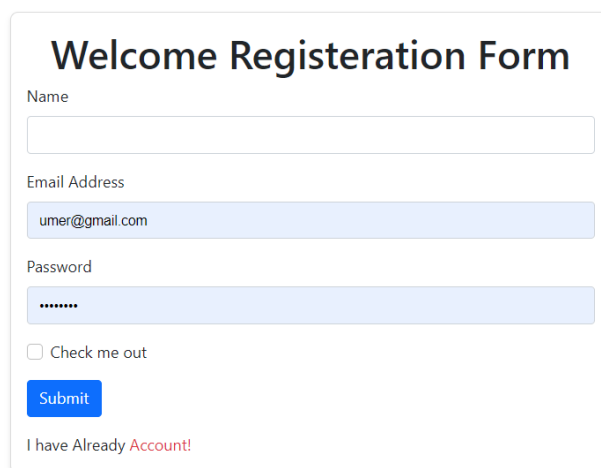
Figure 6.9 Express.Js

6.5. User Interface



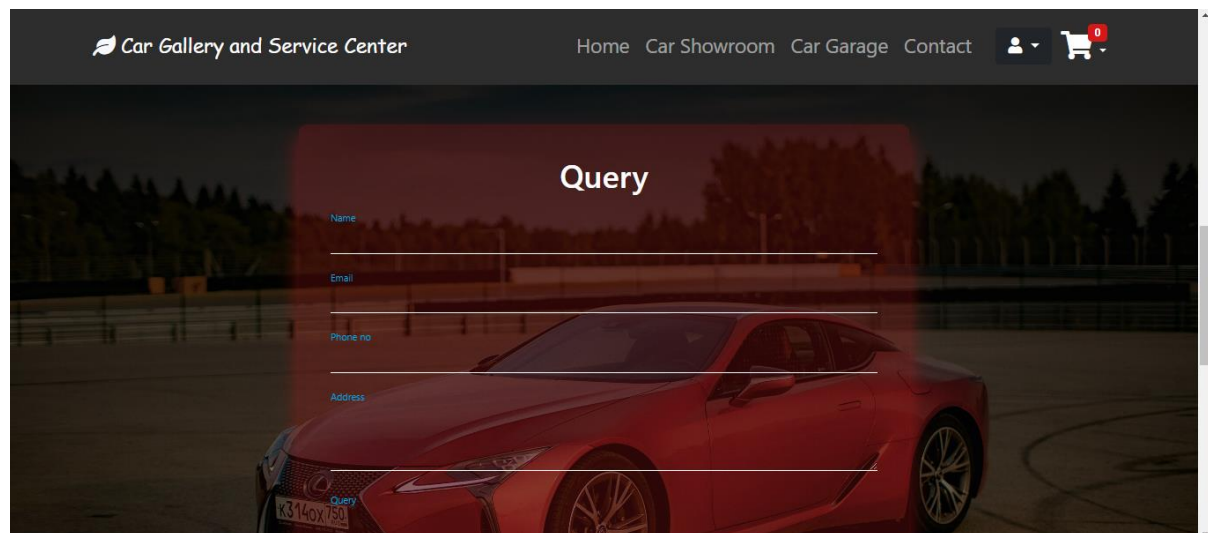
A login form titled "Welcome Login Form". It contains two input fields: "Email Address" with the value "umer@gmail.com" and "Password" with masked characters "*****". Below the password field is a checkbox labeled "Check me out". A blue "Submit" button is positioned below the checkbox. At the bottom, there is a link "Not Registered? Create an Account" where "Not Registered?" is in red.

Figure 6.10 User Interface



A registration form titled "Welcome Registration Form". It contains three input fields: "Name" (empty), "Email Address" with the value "umer@gmail.com", and "Password" with masked characters "*****". Below the password field is a checkbox labeled "Check me out". A blue "Submit" button is positioned below the checkbox. At the bottom, there is a link "I have Already Account!" where "Account!" is in red.

Figure 6.11 Login



The screenshot displays a web application interface for a 'Car Gallery and Service Center'. The top navigation bar includes links for 'Home', 'Car Showroom', 'Car Garage', and 'Contact', along with user and shopping cart icons. A semi-transparent red overlay titled 'Query' is centered on the page, featuring a background image of a red sports car. The form contains five input fields labeled 'Name', 'Email', 'Phone no', and 'Address', each with a corresponding text input line. The 'Query' title is prominently displayed at the top of the overlay.

Figure 6.12 Sign Up

6.5.1. Description

A dealer System is a digital platform that allows users to schedule booking with garage and service provider online. It simplifies the process of booking services, managing services records.

The system typically includes a user-friendly interface for users to select a preferred date and time for their booking, view the availability of techniques. It may also provide options for selecting the type of services, such as in-person or consultation.

Overall, a auto garage System is a useful tool for both user and Dealers as it streamlines the services booking process, enhances communication, and improves the overall user experience.

6.5.2. Interface Screenshots

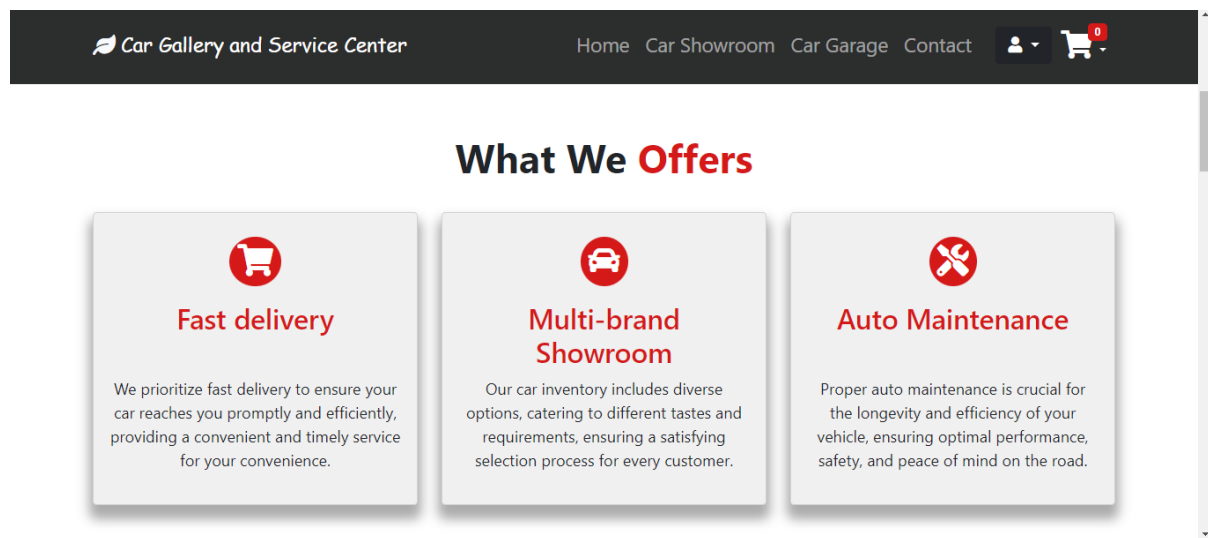


Figure 6.13 About Us

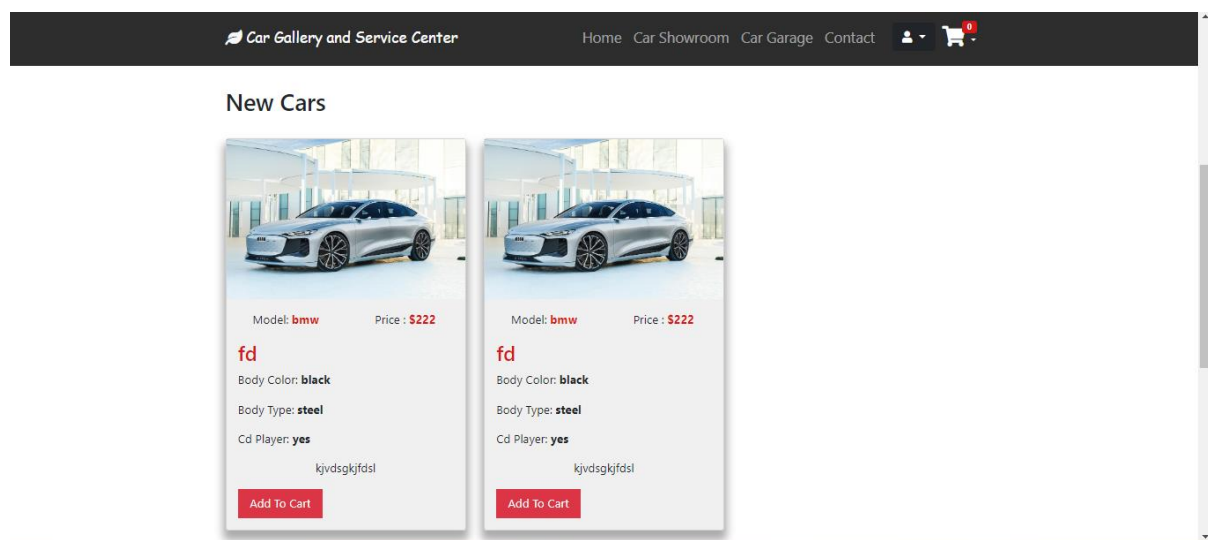


Figure 6.14 Contact Us

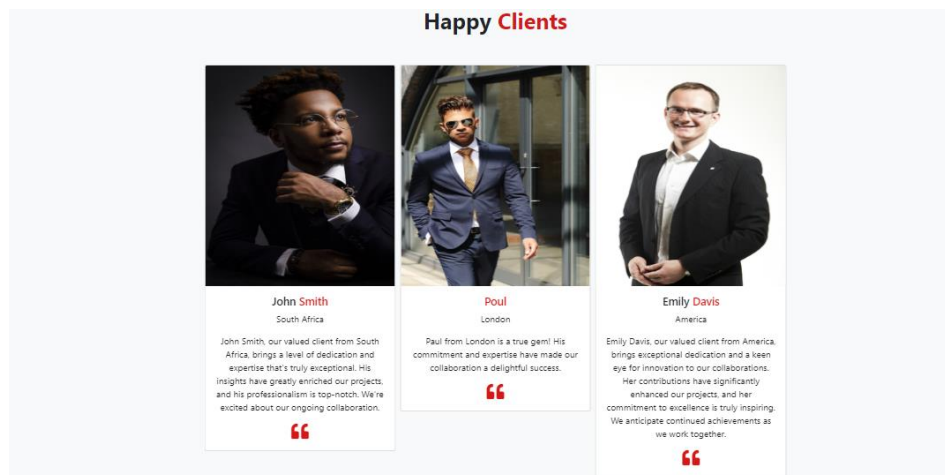


Figure 6.15 Feedback

6.6. Program Deployment

System process mainly consists of development, testing and monitoring. The methods used to build, test and deploy new code will impact how fast a system can respond to changes in requirements and the quality of each change. System performance guide the user about the programs that used in beneficial way for the better performance of the system. Program deployment is the phase where visions and plans become reality. In this phase, we deliver our project in working form. Deployment is the mechanism through which applications, modules, updates, and patches are delivered from developers to users. The methods used by developers to build, test and deploy new code will impact how fast a product can respond to changes in customer preferences or requirements and the quality of each change

6.7. Chapter Summary

In this chapter we discuss development and implementation of my software. Program development is the process of creating application programs. Program development life cycle (PDLC) The process containing the five phases of program development: analysing, designing, coding, debugging and testing, and implementing and maintaining application software. Tools selection is described in this chapter. Coding and interfaces screenshots add in this chapter.

Chapter 7

TESTING

7.1. Introduction

Testing is a critical component of any software development project, including a Service booking. Testing is the process of evaluating the system's performance, functionality, and usability to ensure that it meets the specified requirements and functions as intended. There are several types of testing that can be performed during the development and implementation of a showroom system, including:

- **Unit testing:** This type of testing involves testing individual components of the system, such as modules or functions, to ensure that they work correctly.
- **Integration testing:** This type of testing involves testing how different components of the system interact with each other to ensure that they work seamlessly together.
- **System testing:** This type of testing involves testing the system as a whole to ensure that it meets the specified requirements and functions as intended.
- **User acceptance testing (UAT):** This type of testing involves testing the system from the user's perspective to ensure that it is easy to use, meets their needs, and is intuitive.
- **Performance testing:** This type of testing involves testing the system's performance under different conditions, such as high traffic or heavy usage, to ensure that it can handle the load.
- **Security testing:** This type of testing involves testing the system's security features to ensure that it is secure and protected against unauthorized access.

Overall, testing is an essential component of any Showroom and garage system project to ensure that the system functions as intended, is reliable, and meets the needs of both patients and healthcare providers. It helps to identify and fix issues early in the development process, which ultimately leads to a better product.

7.2. Testing Methods

There are several testing methods that can be used during the development and implementation of a Garage system, depending on the type of testing being performed. Here are some common testing methods:

- **Black box testing:** This method involves testing the system's functionality without knowledge of the system's internal workings. Testers simulate user input and examine the system's output to verify that it meets the specified requirements.
- **White box testing:** This method involves testing the system's internal workings, such as code and algorithms, to verify that they are functioning correctly. Testers use specialized tools to examine the code and identify any potential issues.
- **Manual testing:** This method involves human testers manually performing tests on the system, following predefined test cases or scenarios. Manual testing is typically used for user acceptance testing and exploratory testing.
- **Automated testing:** This method involves using specialized software tools to automate the testing process, including the generation of test data, the execution of tests, and the analysis of test results. Automated testing is typically used for regression testing and performance testing.
- **Acceptance testing:** This method involves testing the system to ensure that it meets the user's requirements and expectations. Acceptance testing is typically performed by end-users or stakeholders to ensure that the system meets their needs.

Overall, testing methods are essential to ensure that the Garage and Showroom system functions correctly, is reliable, and meets the needs of both User and Service providers. By using a combination of testing methods, the development team can identify and fix issues early in the development process, leading to a better product.

7.3. Comparison

A comparison in the context of a Garage and Showroom system can refer to comparing different software systems that offer similar functionalities or comparing different versions of the same software system. Here are some aspects that can be compared:

- **Features:** Comparing features involves identifying the functionalities offered by different Garage and showroom Systems and determining which system offers the most comprehensive set of features.
- **Usability:** Comparing usability involves evaluating the user interface and user experience of different Garage and Showroom Systems to determine which system is the most intuitive and user-friendly.
- **Performance:** Comparing performance involves evaluating the speed and responsiveness of different Garage and Showroom Systems to determine which system offers the best performance.
- **Security:** Comparing security involves evaluating the security features of different Garage and Showroom Systems to determine which system offers the most robust security measures.
- **Cost:** Comparing cost involves evaluating the pricing models of different Garage and Showroom Systems to determine which system offers the best value for money.
- **Integration:** Comparing integration involves evaluating the ability of different Garage and Showroom Systems to integrate with other software systems.

Overall, a comparison can help stakeholders, including service providers and user, to make an informed decision about which Garage and Showroom System to use based on their needs, preferences, and budget.

7.4. Software Evaluation

Software evaluation is the process of assessing a software system to determine its suitability for a particular purpose or task. In the context of a Garage and Showroom System, software evaluation involves assessing the system to determine if it meets the specific needs and requirements of service providers and user. Here are some aspects that can be evaluated:

- **Functionality:** The evaluation should assess if the system has all the necessary functionalities needed for a Garage and Showroom System, including service scheduling, vehicle records management, reminder systems, and reporting tools.
- **Usability:** The evaluation should assess if the system is user-friendly, easy to navigate, and intuitive for both service providers and user.

- **Performance:** The evaluation should assess if the system is fast and responsive and can handle large volumes of data without slowing down.
- **Security:** The evaluation should assess if the system has robust security measures in place to protect user data and ensure compliance with privacy regulations.
- **Integration:** The evaluation should assess if the system can easily integrate with other software systems used by service providers.
- **Support and Maintenance:** The evaluation should assess if the software vendor provides reliable technical support and maintenance services to ensure the system is always up-to-date, secure, and functioning correctly.

Overall, software evaluation is a crucial step in selecting a Garage and Showroom System that meets the specific needs of service providers and users. It helps to ensure that the selected system is reliable, efficient, secure, and provides a positive user experience.

7.4.1. Testing Strategy

A strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vortex of the spiral and, concentrates on each unit, component of the software as implemented in source code. Testing progresses moving outward along the spiral to integration testing, where the focus is on designed the construction of the software architecture. Taking another turn outward on spiral, we encounter validation testing, where requirements established as part of software requirements analysis are validated against the software that has been constructed. Finally, we arrive at system testing, where the software and other system elements are tested as a whole. To test computer software, we spiral out along stream lines that broaden the scope of testing with each turn. Considering the process from a procedural point of view, testing within the context of software engineering is actually a series of four steps that are implemented sequentially. The steps are shown in Figure. Initially, tests focus on each component individually, ensuring that it functions properly as unit. Unit testing makes heavy use of white-box testing techniques, exercising specific paths in module's control structure to ensure complete coverage and maximum error detection. Here are some elements that can be included in a testing strategy for a Garage and Showroom System.

- **Identify testing types:** The testing strategy should identify the types of testing needed, including functional testing, integration testing, regression testing, performance testing, and user acceptance testing.
- **Define test objectives:** The testing strategy should define the objectives of each test type and the expected outcomes.
- **Identify testing tools:** The testing strategy should identify the tools and resources needed to perform the tests, including test automation tools and manual testing resources.
- **Define test scenarios:** The testing strategy should define test scenarios for each type of testing, including positive and negative scenarios, edge cases, and error handling scenarios.
- **Define test data:** The testing strategy should define the data needed for each test scenario, including sample user records, service schedules, and other relevant data.
- **Determine testing environment:** The testing strategy should determine the testing environment, including the hardware and software configurations needed for testing.
- **Develop a testing schedule:** The testing strategy should develop a schedule for each type of testing, including the start and end dates, the resources required, and the expected duration of each testing phase.
- **Reporting and tracking:** The testing strategy should define the reporting and tracking mechanisms used to document and report test results, including bug reports, test case results, and overall test progress.

Overall, a well-defined testing strategy is critical to ensure that the Garage and Showroom System is thoroughly tested and meets the specified requirements. It helps to ensure that the system is reliable, efficient, and meets the needs of service providers and users.

7.4.2. Test Plans

The purpose of this test plan is to ensure that the doctor appointment system meets the functional and non-functional requirements as specified in the project documentation. The test plan will cover various test scenarios that will be executed to verify the system's behavior and performance. The scope of this test plan is to test the Garage and Showroom system's main

functionalities, including booking, cancelling a booking, and viewing booking history. The test plan will cover both positive and negative test scenarios. The Garage and Showroom system will be tested on a development environment that mirrors the production environment. The test environment will include a test database that is separate from the production database. Test data will be created for testing purposes.

- **Test Scenarios:**

Booking a service

Cancelling a Booking

Viewing booking history

7.4.1. Test Cases

Test Scenario ID-1	Register-1	Test Case ID	Register-1A		
Test Case Description	Register-Positive test case	Test Priority	High		
Pre-Requisite	A valid user	Post-Requisite	NA		
Test-Execution Steps:					
S.No	Action	Input	Expected Output	Actual Output	Test Results
1	Enter correct email/Ph: No	Correct OTP	Registration complete	Registration complete	Pass

Figure 7.1 Case No: 01

Test Scenario ID-1	Register-1	Test Case ID	Register-1B		
Test Case Description	Register-Negative test case	Test Priority	High		
Pre-Requisite	NA	Post-Requisite	NA		
Test-Execution Steps:					
S.No	Action	Input	Expected Output	Actual Output	Test Results
1	Enter correct email/Ph: No	Wrong OTP	Please check OTP	No next page	Pass

Figure 7.2 Case No: 02

Test Scenario ID-1	Login-1	Test Case ID	Login-1A		
Test Case Description	Login-Negative test case	Test Priority	High		
Pre-Requisite	A valid user account	Post-Requisite	NA		
Test-Execution Steps:					
S.No	Action	Input	Expected Output	Actual Output	Test Results
1	Check password and email	Wrong password correct email	Invalid user name or password	Invalid user name or password	Pass

Figure 7.3 Case No: 03

Test Scenario ID-1		Login-1	Test Case ID		Login-1B
Test Case Description		Login-Negative test case	Test Priority		High
Pre-Requisite		A valid user account	Post-Requisite		NA
Test-Execution Steps:					
S.No	Action	Input	Expected Output	Actual Output	Test Results
1	Check password and email	Wrong email correct password	Invalid user name or password	Invalid user name or password	Pass

Figure 7.4 Case No: 04

Test Scenario ID-1	Login-1	Test Case ID	Login-1C		
Test Case Description	Login-Negative test case	Test Priority	High		
Pre-Requisite	A valid user account	Post-Requisite	NA		
Test-Execution Steps:					
S.No	Action	Input	Expected Output	Actual Output	Test Results
1	Check password and email	Wrong email Wrong password	Invalid user name or password	Invalid user name or password	Pass

Figure 7.5 Case No: 05

Test Scenario ID-1		Login-1	Test Case ID		Login-1D
Test Case Description		Login-Positive test case	Test Priority		High
Pre-Requisite		A valid user account	Post-Requisite		NA
Test-Execution Steps:					
S.No	Action	Input	Expected Output	Actual Output	Test Results
1	Check password and email	Correct email and password	User Dashboard	User Dashboard	Pass

Figure 7.6 Case No: 06

Test Scenario ID-1	Search-1	Test Case ID	Search-1A		
Test Case Description	Searching-Negative test case	Test Priority	High		
Pre-Requisite	A valid Dealer name	Post-Requisite	NA		
Test-Execution Steps:					
S.No	Action	Input	Expected Output	Actual Output	Test Results
1	Enter Wrong name	Search for a Dealer	Not found	Not found	Pass

Figure 7.7 Case No: 07

Figure 7.8 Case No: 08

Test Scenario ID-1	Search-1	Test Case ID	Search-1B		
Test Case Description	Searching-Positive test case	Test Priority	High		
Pre-Requisite	A valid Dealer name	Post-Requisite	NA		
Test-Execution Steps:					
S.No	Action	Input	Expected Output	Actual Output	Test Results
1	Enter correct Dealer name	Correct name of Dealer being registered	Dealer Profile	Show Dealer profile	Pass

Test Scenario ID-1		Search-1	Test Case ID		Search-1C
Test Case Description		Searching-Negative test case	Test Priority		High
Pre-Requisite		A valid registered garage store	Post-Requisite		NA
Test-Execution Steps:					
S.No	Action	Input	Expected Output	Actual Output	Test Results
1	Enter wrong garage store name	Wrong name of garage store	Not Found	Not Found	Pass

Figure 7.9 Case No: 09

Test Scenario ID-1	Search-1	Test Case ID	Search-1C		
Test Case Description	Searching-Positive test case	Test Priority	High		
Pre-Requisite	A valid registered garage store	Post-Requisite	NA		
Test-Execution Steps:					
S.No	Action	Input	Expected Output	Actual Output	Test Results
1	Enter correct garage store name	Correct name of garage store	Display garage Store	Show the entered garage store	Pass

Figure 7.10 Case No: 10

Test Scenario ID-1	Search-1	Test Case ID	Search-1A		
Test Case Description	Searching-Negative test case	Test Priority	High		
Pre-Requisite	A Registered service	Post-Requisite	NA		
Test-Execution Steps:					
S.No	Action	Input	Expected Output	Actual Output	Test Results
1	Enter Wrong service name	Wrong name of Service	Not Found	Not Found	Pass

Figure 7.11 Case No: 11

Test Scenario ID-1		Search-1	Test Case ID		Search-1B
Test Case Description		Searching-Positive test case	Test Priority		High
Pre-Requisite		A Valid Registered Service	Post-Requisite		NA
Test-Execution Steps:					
S.No	Action	Input	Expected Output	Actual Output	Test Results
1	Enter correct Service name	Correct name of Service	Show the entered service detail	Show the entered Service detail	Pass

7.4.1. Test Report

Test cases are built around specifications and requirements i.e. what the application is supposed to do. Test cases are generally derived from external descriptions of the software, including specifications, requirements and design parameters. Although the tests used are primarily functional in nature, non-functional tests may also be used. The test designer selects both valid and invalid inputs and determines the correct output. The test is unbiased because the designer and the tester are independent of each other. The tester does not need knowledge of any specific programming languages. The test is done from the point of view of the user, not the designer. Test cases can be designed as soon as the specifications are complete.

The Garage and Showroom system is a web-based application designed to help user schedule booking services. The goal of this system is to streamline the appointment booking process, reduce wait times for users, and improve overall efficiency for showroom offices. The system allows user to view available booking slots and book a service from garage online. The system also allows garage offices to manage their booking schedules and view user details.

7.5. Chapter Summary

The aim of testing is often to demonstrate that a program works by showing that it has no errors. The basic purpose of testing phase is to detect the errors that may be presenting the program. Hence one should not start testing with the intent of showing that a program works, but the intent should be to show that a program doesn't work. Testing is the process of executing a program with the intent of finding errors

REFERENCES

- [1] <https://www.armoryauto.com/showroom/index.htm>
- [2] <https://assetstore.unity.com/packages/3d/environments/vehicle-garage-showroom-mobile-ready-162119>
- [3] <https://www.scribd.com/document/337853350/car-Showroom-Management-System-doc>
- [4] <https://carvago.com/electric-hybrid-vehicles>
- [5] <https://www.techmangms.com/features/>
- [6] <https://app.openbay.com/>

REFERENCES

APPENDIX (Optional)