

REAL DEAL CARS

Project Report Submitted by

Timin Kurian

Reg. No: AJC16MCA24

In Partial fulfilment for the award of the degree

Of

**MASTER OF COMPUTER APPLICATIONS (MCA)
APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY**



**AMAL JYOTHI COLLEGE OF ENGINEERING
KANJIRAPPALLY**

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE, Accredited by NAAC with 'A' grade. Koovappally, Kanjirappally, Kottayam, Kerala - 686518]

2016-2019

AMAL JYOTHI COLLEGE OF ENGINEERING

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE, Accredited by NAAC with 'A' grade. Koovappally, Kanjirappally, Kottayam, Kerala - 686518]

DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS



CERTIFICATE

This is to certify that the project entitled **“Real deal cars”** is a bonafide record of the work done by **Timin Kurian AJC16MCA24**, during the academic year **2016-2019** carried out under our supervision. It is certified that all corrections/suggestions indicated for assessment have been incorporated in the report. The work report has been approved as it satisfies the academic requirements in respect of the project work prescribed by the university for the Master of Computer Applications Degree. Certified further, that to the best of our knowledge the exact work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this to any other candidate.

Fr. Rubin Thottupuram

Head of the Department

Mr. Binumon Joseph

Project Coordinator

Sr. Elsin Chakkalackal S H

Project Supervisor

Expert from dept. of Computer Science and Engineering
Amal Jyothi College of Engineering

External Expert appointed by the university

DECLARATION

I hereby declare that the project report “**Real Deal Cars**” is a bonafide work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the Degree of Master of Computer Applications (MCA) from APJ Abdul Kalam Technological University, during the academic year 2016-2019.

Date.....

KANJIRAPPALLY

Timin Kurian

Reg. No: AJC16MCA24

ACKNOWLEDGEMENT

First and foremost, I thank Almighty God for his gracious guidance through the project. I take this opportunity to express my gratitude to all those who have helped me in completing the project successfully

It has been said that gratitude is the memory of the heart. I acknowledge my deep sense of gratitude to our manager **Rev. Fr. Dr. Mathew Paikatt** for providing all the infrastructural facilities for us, our Principal **Dr. Z V Lakaparampil** for providing good faculty for guidance.

I take the immense pleasure in expressing my thanks to Head of the Department of Master of Computer Applications, **Fr. Rubin Thottupuram**, for his kind patronages in making this project a successful one. I would like to extend my sincere thanks to our coordinator **Mr. Binumon Joseph** and my project guide **Sr. Elsin Chakkalackal S H** for their guidance and cooperation, without which this would not have been a success.

I am indebted to my beloved teachers whose cooperation and suggestions throughout the project which helped me a lot. I also thank all my friends and classmates for their interest, dedication and encouragement shown towards the project. I convey hearty thanks to parents for the moral support, suggestion and encouragement to make this venture a success.

Timin Kurian

ABSTRACT

This project is intended to design and develop a website for car service and sales. The project provides the facility for sales of used cars with cent percent quality assurance. The registered customers can find the nearby service centres and book services. The registered customers can sell their used cars through the site. The site includes the facility of selling only the cars that have serviced in the registered service centres. The main advantage of the proposed system is that it avails the full-service history of the cars for the interested buyers, so that they can chose the best of them. This facility ensures the descent deal for the sellers too. The registered service centres can provide the complete history of their servicing that has done in their centre. The ultimate advantage of the system is that it ensures 100 % genuinity in used car selling and purchasing.

Sl. No	Topic	Page No
Part 1	Project Documentation	
P1.1	Introduction	
P1.1.1	Project Overview	
P1.1.2	Project Specification	
P1.2	System Study	
P1.2.1	Introduction	
P1.2.2	Proposed system	
P1.3	Requirement Analysis	
P1.3.1	Feasibility Study	
P1.3.1.1	Economical Feasibility	
P1.3.1.2	Technical Feasibility	
P1.3.1.3	Operational Feasibility	
P1.4	Requirement Modelling	
P1.4.1	Class Diagram	
P1.4.2	Object Diagram	
P1.4.3	Deployment Diagram	
P1.4.4	Use case Diagram	
P1.4.5	Sequence Diagram	
P1.4.6	Collaboration Diagram	
P1.4.7	Statechart Diagram	
P1.4.8	Activity Diagram	
P1.5	System Specification	
P1.5.1	Hardware Specification	
P1.5.2	Software Specification	
P1.6	Software Description	
P1.6.1	PHP	
P1.6.2	MySQL	
P1.7	System Design	
P1.7.1	Architectural Design	
P1.7.2	Module Design	
P1.7.3	Data Base Design	

P1.8	System Testing	
P1.8.1	Introduction	
P1.8.2	Test Plan	
P1.8.2.1	Unit Testing	
P1.8.2.2	Integration Testing	
P1.8.2.3	Validation Testing	
P1.8.2.4	User Acceptance Testing	
P1.9	Implementation	
P1.9.1	Implementation Procedure	
P1.9.2	User Training	
P1.9.3	Operational Document	
P1.9.4	System Maintenance	
P1.10	Conclusion & Future Enhancements	
P1.10.1	Future Enhancement	
P1.10.2	Conclusion	
P1.11	Bibliography	
P1.12	Appendix	
P1.12.1	Sample Code	
P1.12.2	Screen Shots	
Part 2	Technology Frameworks	
P2.1	ASP.NET MVC	
P2.2	Laravel for PHP	
P2.3	Angular JS	
P2.4	Android	
P2.5	Server Hardening	

LIST OF ABBREVIATIONS

IDE	-	Integrated Development Environment
HTML	-	Hyper Text Markup Language.
CSS	-	Cascading Style Sheet
SQL	-	Structured Query Language

Part 1

Project Documentation

P1.1 INTRODUCTION

P1.1.1 Project Overview

This project is intended to design and develop a website for car service and sales. The project provides the facility for sales of used cars with cent percent quality assurance. The registered customers can find the nearby service centres and book services. The registered customers can sell their used cars through the site. The site includes the facility of selling only the cars that have serviced in the registered service centres. The main advantage of the proposed system is that it avails the full-service history of the cars for the interested buyers, so that they can chose the best of them. This facility ensures the descent deal for the sellers too. The registered service centres can provide the complete history of their servicing that has done in their centre.

The ultimate advantage of the system is that it ensures 100 % genuinity in used car selling and purchasing.

P1.1.2 Project Specification

A website designed for the sales and services of used cars. Enables registered customers to locate service centres, make appointments and receive various services. Users can also sell their vehicles through this site

The system includes 4 modules. They are:

1) Users

- Register cars
- Buy used cars
- View service schemes
- Book an appointment
- View service history
- Sell vehicle
- View the cars for sales
- View the service history of the interested cars

2) Service centres

- Register Service centre
- Add service scheme
- Add service history
- Manage leave of employees
- Manage employees

3) Admin

- Approve service centres
- Approve the registration of cars
- Overall management of the website

4) Employee

- Update profile
- Apply leave
- Manage the appointments

P1.2 SYSTEM STUDY

P1.2.1 Introduction

System analysis is a process of gathering and interpreting facts, diagnosing problems and the information to recommend improvements on the system. It is a problem-solving activity that requires intensive communication between the system users and system developers. System analysis or study is an important phase of any system development process. The system is studied to the minute's detail and analysed. The system analyst plays the role of the interrogator and dwells deep into the working of the present system. The system is viewed as a whole and the input to the system are identified. The outputs from the organizations are traced to the various processes. System analysis is concerned with becoming aware of the problem, identifying the relevant and decisional variables, analysing and synthesizing the various factors and determining an optimal or at least a satisfactory solution or program of action.

A detailed study of the process must be made by various techniques like interviews, questionnaires etc. The data collected by these sources must be scrutinized to arrive to a conclusion. The conclusion is an understanding of how the system functions. This system is called the existing system. Now the existing system is subjected to close study and problem areas are identified. The designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user is satisfied with proposal.

Preliminary study is the process of gathering and interpreting facts, using the information for further studies on the system. Preliminary study is problem solving activity that requires intensive communication between the system users and system developers. It does various feasibility studies. In these studies, a rough figure of the system activities can be obtained, from which the decision about the strategies to be followed for effective system study and analysis can be taken

P1.2.2 PROPOSED SYSTEM

To overcome limitations of existing system, we can introduce this site for service booking and used car sales. In the proposed system, it provides the services to the users who are searching for service like appointment for vehicle service, buy used cars and also provide car service history to the interested buyers. This is a website in which we will get the several services.

The market for old cars is 1.2 times the market of new cars. Like Quikr and Olx work on a customer to customer (C2C) model where a customer looking to sell an old product uploads photo of the car, sets a selling price and furnishes some basic details (not compulsory) related to the car like Kms driven, features etc. For somebody looking to buy a second hand car, this information is of little use. Moreover, approx. 7 out of 10 cars even on a C2C platform are uploaded by the dealer only. This you would get to know after talking to the person on the other side.

The existing system has several limitations and more difficulties to work well. The proposed system provides proper security and reduces the manual work, and it helps the user to work user friendly and he can easily do this job without time delay.

The project provides the facility for sales of used cars with cent percent quality assurance. The registered customers can find the nearby service centres and book services. The registered customers can sell their used cars through the site. The main advantage of the proposed system is that it avails the full-service history of the cars for the interested buyers, so that they can chose the best of them. This facility ensures the descent deal for the sellers too. The registered service centres can provide the complete history of their servicing that has done in their centre.

The main features include:

- Car owners can easily find the service centres near to them
- The system will ensure a good price to sellers
- The buyers can identify the condition of the vehicle
- Buyers can view the service history of the cars

ADVANTAGES OF PROPOSED SYSTEM

The system is very simple in design and to implement. The system requires very low system resources and the system will work in almost all configurations. It has got following features:

- *Better security:* -

For data to remain secure measures must be taken to prevent unauthorized access. Security means that data are protected from various forms of destruction. The system security problem can be divided into four related issues: security, integrity, privacy and confidentiality. Username and password requirement to sign in ensures security. It will also provide data security as we are using the secured databases for maintaining the documents.

- *Ensure data accuracy:* -

The proposed system eliminates the manual errors while entering the details of the users during the registration.

- *Better service:* -

The product will avoid the burden of hard copy storage. We can also conserve the time and human resources for doing the same task. The data can be maintained for longer period with no loss of data.

- *User friendliness and interactive:* -

The proposed system will help the user to reduce the workload and provides user friendly environment so that they can easily do their jobs. The system alerts the users for each activity to be carried out, through notification.

- *Minimum time required:* -

The data are management is in such a way that a particular registered user can search service provider very easily.

P1.3 REQUIREMENT ANALYSIS

1. Project Overview?

This project is intended to design and develop a website for car service and sales. The project provides the facility for sales of used cars with cent percent quality assurance.

2. To what extend the system is proposed for?

The registered customers can find the nearby service centres and book services. The registered customers can sell their used cars through the site. The site includes the facility of selling only the cars that have serviced in the registered service centres. The main advantage of the proposed system is that it avails the full-service history of the cars for the interested buyers, so that they can chose the best of them. This facility ensures the descent deal for the sellers too. The registered service centres can provide the complete history of their servicing that has done in their center.

3. Specify the Viewers/Public which is to be involved in the System?

Car owners can book services and sell their cars, service center staffs who manage daily activities of the service centers.

4. List the Modules included in your System?

Service booking, Service center, Search, Chat, Admin and Payment

5. Identify the users in your project?

Admin: He has the overall control of the system. Admin can approve new users and service centers.

Users: The registered users can book services and sell their cars and they can also view the service history of the cars.

Service Center Manager: They manage all the appointments with their service centers. They can add new service schemes of their service center.

Service center employees: They manage all the appointments with their service centers. The can applies for leaves.

6. Who owns the system?

Admin owns the system. He has the control over other users. Admin approves the service centers and users of the system.

7. System is related to which firm/industry/organization?

The system is related to the automobile industry. It includes the service booking and sales of used cars.

8. Details of person that you have contacted for data collection?

Rintu Mathew

The manager

Team Professionals

Near Municipal Stadium

Puzhakkara Nagar

Pala

9. Questionnaire to collect details about the project? (min 10 questions, include descriptive answers, attach additional docs (e.g. Bill receipts, certificate models), if any?)

a) Is the users service their vehicles in service center of their own district?

No, can be done in anywhere.

b) Different service dealers pay same rate for same service?

Authorized service centers bill the same rates

c) The employees in service centers are working on the basis of department wise?

Yes, based on the service centers status the employees count can be different.

d) Is it possible to allot a particular time limit for each service?

No, based on vehicle conditions the service time will be changed, if it is a same type of vehicle too.

e) The servicing of each vehicle is based on any time period or kilometer period?

Yes, almost all vehicles are serviced at 10000 km period but sometimes it may be 6 months.

f) There is any drawback in used cars selling sites like Olx?

Yes, now in olx the number of used car dealers is more than genuine customers, which may lead to some problems.

g) What was the big problem faced by used car sales?

Not all sellers get appropriate price based on their vehicle condition or service history.

h) Is the service center get profit by doing this project?

Yes, definitely increase the count of customers by providing a platform for selling of used cars.

i) Is this project idea getting success?

Yes, but there is a limited chance for get the support of an authorized centers when we add unauthorized centers.

j) Any advice for me?

It is better to avoid adding of unauthorized service centers and also better to provide service history to interested peoples not all.

P1.3.1 Feasibility Study

The feasibility study is concerned with the consideration made to verify whether the system fit to be developed in all terms. Once the idea to develop software is put forward, the question that rises first will pertain to be the feasibility aspects. A feasibility study is conducted to select the best system that meets the system performance requirements. This entitles an identification description, an evaluation of candidate systems and the selection of the best system for the job. It also helps in identifying the risk factors involved in developing and deploying the system. So, a feasibility study is a report which could be used by the senior or top persons in the organization. This is because based on the report the organization decides about cost estimation, funding and other important decisions which is very essential for an organization to run profitably and for the system to run stable.

P1.3.1.1 Economical Feasibility

Economic feasibility analysis is the most commonly used method for determining the efficiency of a new project. It is also known as cost analysis. It helps in identifying profit against investment expected from a project. Cost and time are the most essential factors involved in this field of study. The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require. Existing systems use offline service, so compared to these proposed system software development needs some costs.

The Real Deal Cars software production needs various costs like system costs. The system cost may be classified into different categories including developing cost, on-going operational cost, fixed cost and variable cost. Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also, all the resources are already available, it gives an indication of the system is economically possible for development.

By avoiding these costs, we have to improve the quality of work or to permit new activities to be undertaken and common performance benefits might be error reduction, increased speed of activity, and access to information that was not previously available.

P1.3.1.2 Technical Feasibility

It helps organizations determine whether the technical resources meet capacity and whether the technical team is capable of converting the ideas into working systems. Technical feasibility also involves evaluation of the hardware, software, and other technology requirements of the proposed system.

The users have the capability and resources to undertake the Real Deal Cars. The organization doesn't need any other resources they have the capability to access our system with their own existing computer and proper internet connection.

The project should be developed such that the necessary functions and performance are achieved within the constraints. Through the technology may become obsolete after some period of time, due to the fact that newer version of same software supports older versions, the system may still be used. So, there are minimal constraints involved with this project. The system has been developed using php in front end and MySQL Server in back end, the project is technically feasible for development.

P1.3.1.3 Operational Feasibility

Economic feasibility determines whether the required software is capable of generating financial gains for an organization. It involves the cost incurred on the software development team, estimated cost of hardware and software, cost of performing feasibility study, and so on.

The organization is satisfied by the alternative solution proposed by the software development team. Our proposed system works to minimize the human errors, take less time, easy interaction with user, bug free

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioural aspects are considered carefully and conclude that the project is behaviourally feasible. Real Deal Cars, GUI is simple so that users can easily use it. Real Deal Cars is simple enough so that no training is needed.

P1.4 Requirement Modelling

Structural Diagrams

The structural diagrams represent the static aspect of the system. These static aspects represent those parts of a diagram, which forms the main structure and are therefore stable.

These static parts are represented by classes, interfaces, objects, components, and nodes. The following are the structural diagrams

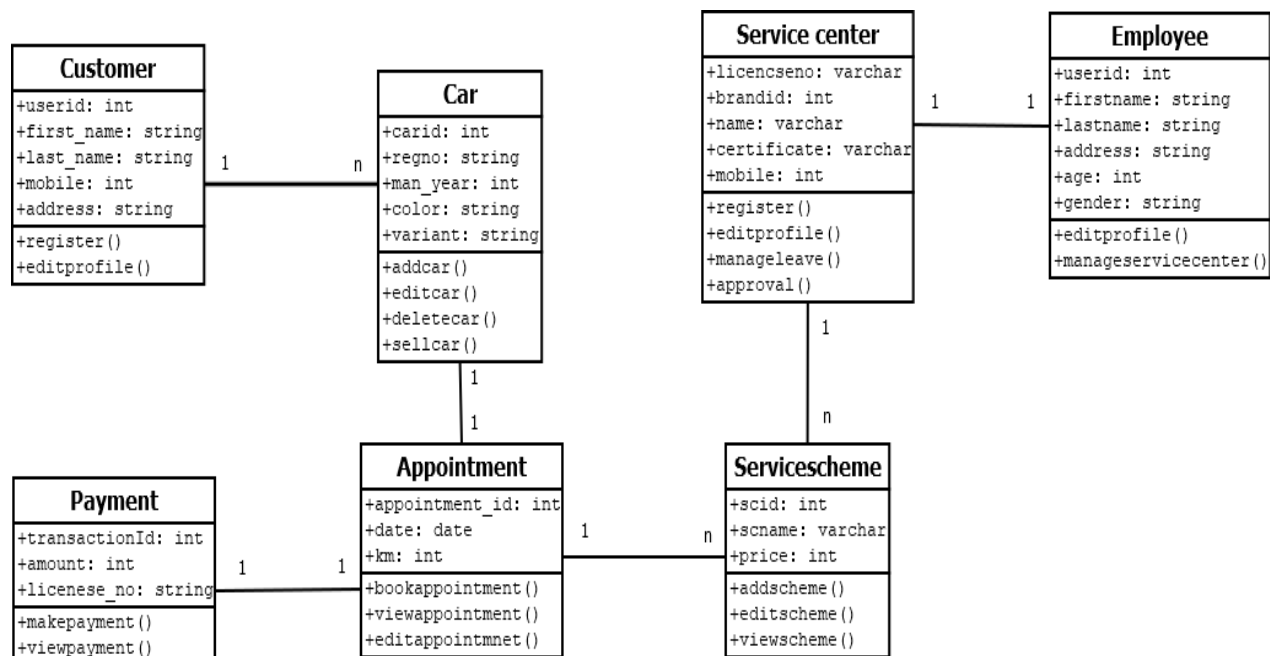
- Class diagram
- Object diagram
- Deployment diagram

P1.4.1 Class diagram

Class diagrams are the most common diagrams used in UML. Class diagram consists of classes, interfaces, associations, and collaboration. Class diagrams basically represent the object-oriented view of a system, which is static in nature.

Active class is used in a class diagram to represent the concurrency of the system.

Class diagram represents the object orientation of a system. Hence, it is generally used for development purpose. This is the most widely used diagram at the time of system construction.

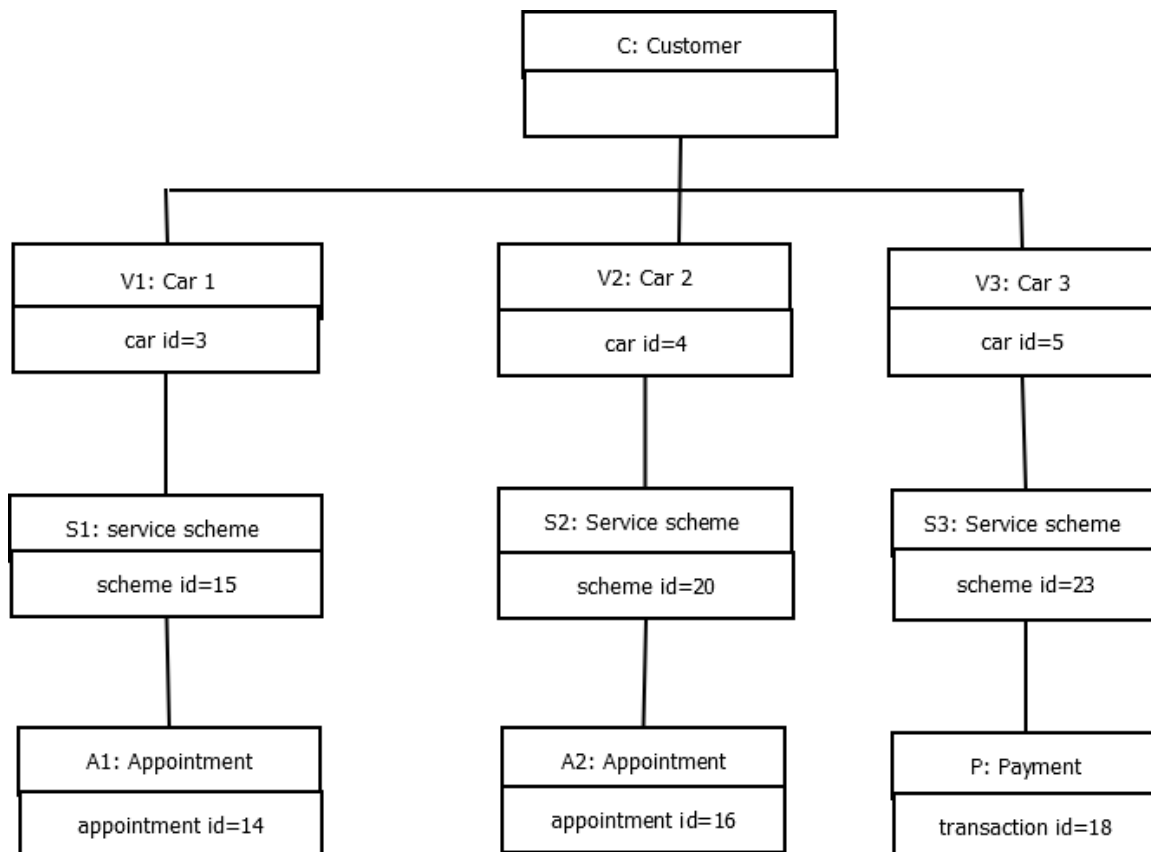


P1.4.2 Object diagram

Object diagrams can be described as an instance of class diagram. Thus, these diagrams are closer to real-life scenarios where we implement a system.

Object diagrams are a set of objects and their relationship is just like class diagrams. They also represent the static view of the system.

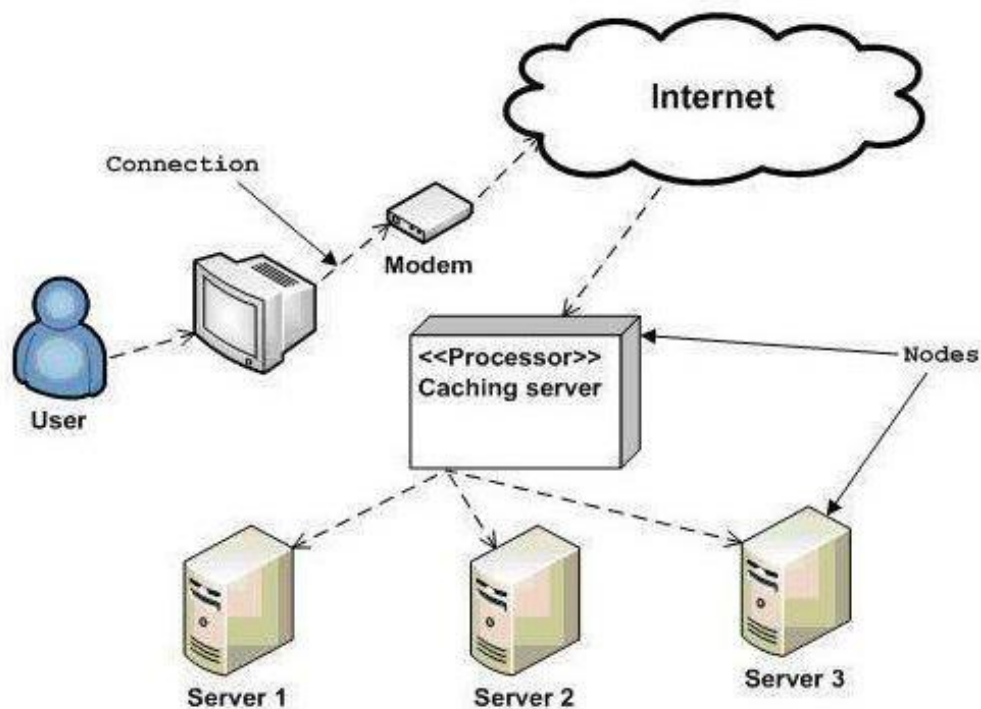
The usage of object diagrams is similar to class diagrams but they are used to build prototype of a system from a practical perspective.



P1.4.3 Deployment diagram

Deployment diagrams are a set of nodes and their relationships. These nodes are physical entities where the components are deployed.

Deployment diagrams are used for visualizing the deployment view of a system. This is generally used by the deployment team.



Behavioural Diagrams

Any system can have two aspects, static and dynamic. So, a model is considered as complete when both the aspects are fully covered.

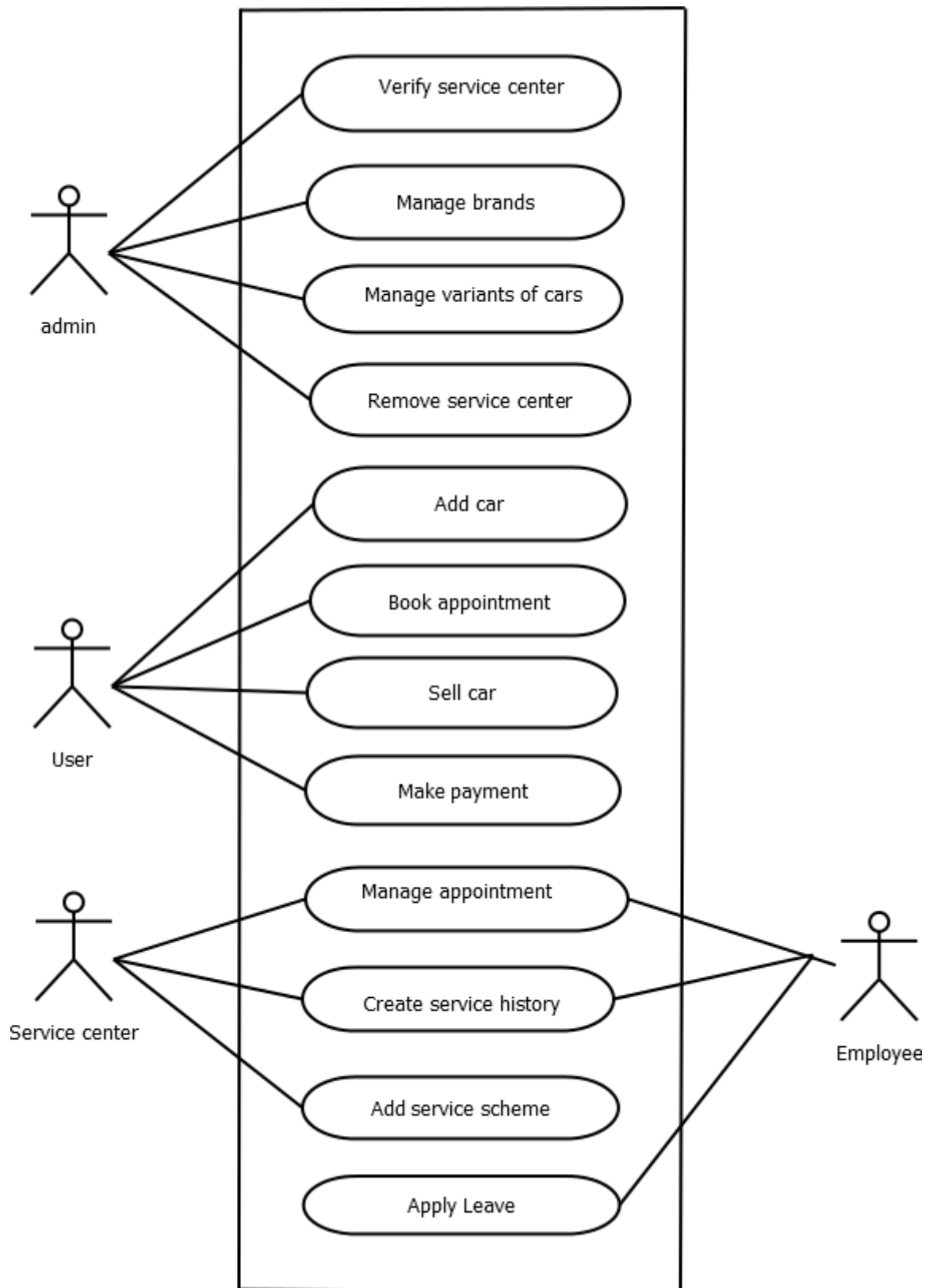
Behavioural diagrams basically capture the dynamic aspect of a system. Dynamic aspect can be further described as the changing/moving parts of a system.

UML has the following five types of behavioural diagrams

- Use case diagram
- Sequence diagram
- Collaboration diagram
- Statechart diagram
- Activity diagram

P1.4.4 Use Case Diagram

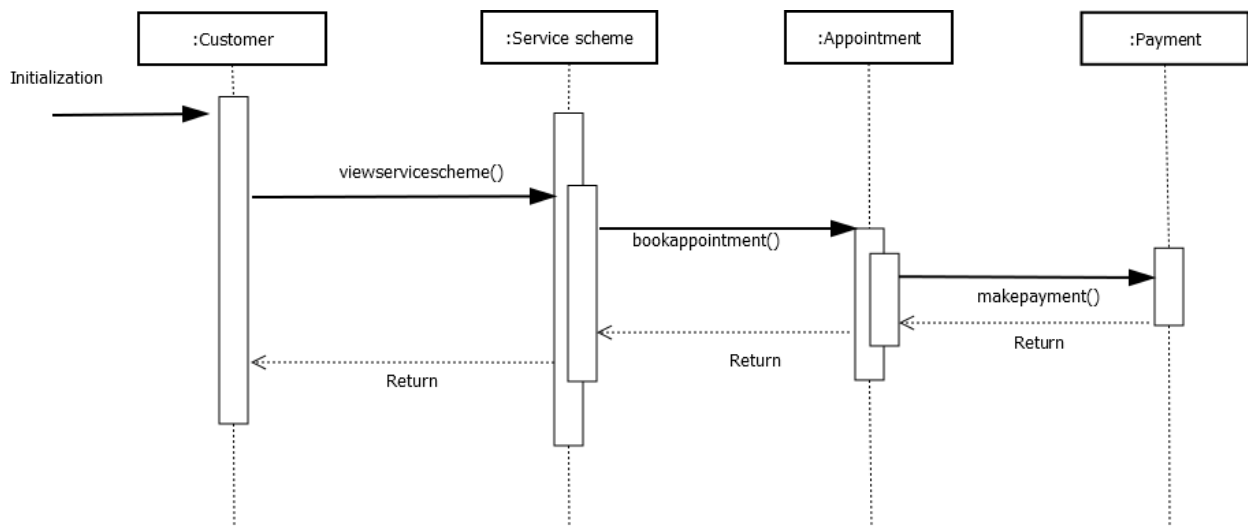
Use case diagrams are a set of use cases, actors, and their relationships. They represent the use case view of a system. A use case represents a particular functionality of a system. Hence, use case diagram is used to describe the relationships among the functionalities and their internal/external controllers. These controllers are known as actors.



P1.4.5 Sequence Diagram

A sequence diagram is an interaction diagram. From the name, it is clear that the diagram deals with some sequences, which are the sequence of messages flowing from one object to another.

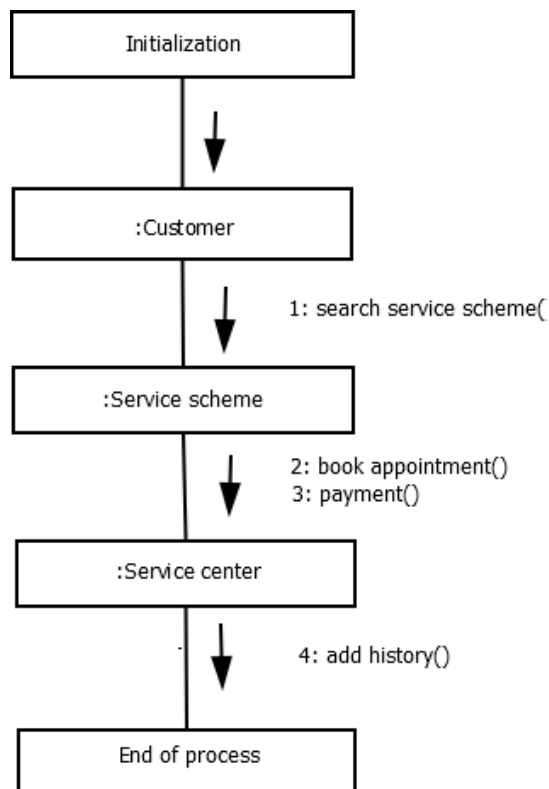
Interaction among the components of a system is very important from implementation and execution perspective. Sequence diagram is used to visualize the sequence of calls in a system to perform a specific functionality.



P1.4.6 Collaboration Diagram

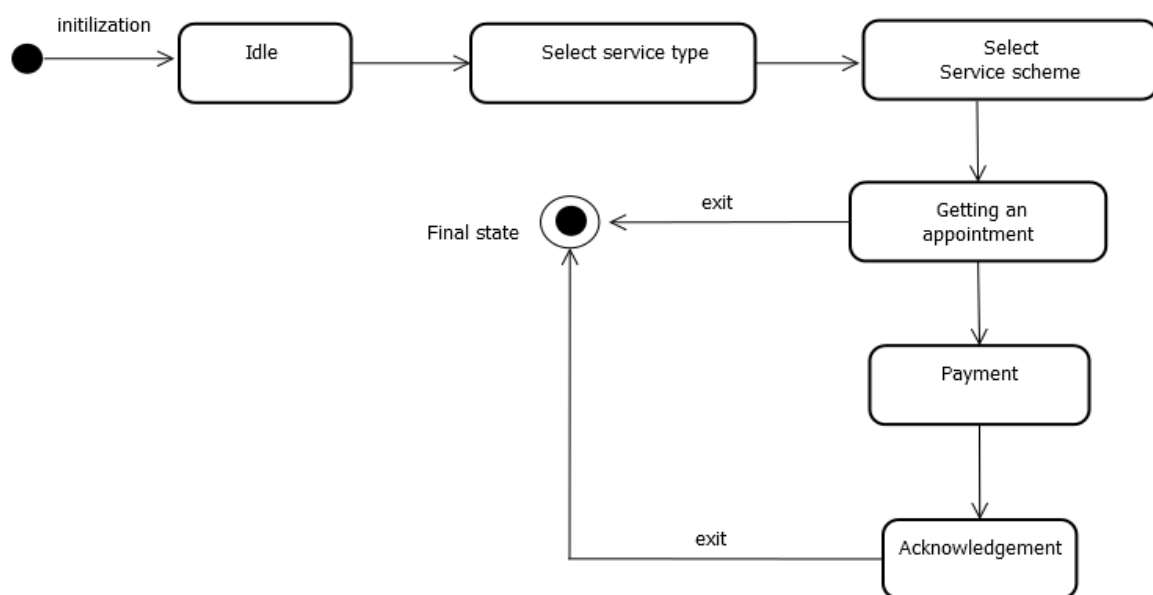
Collaboration diagram is another form of interaction diagram. It represents the structural organization of a system and the messages sent/received. Structural organization consists of objects and links.

The purpose of collaboration diagram is similar to sequence diagram. However, the specific purpose of collaboration diagram is to visualize the organization of objects and their interaction.



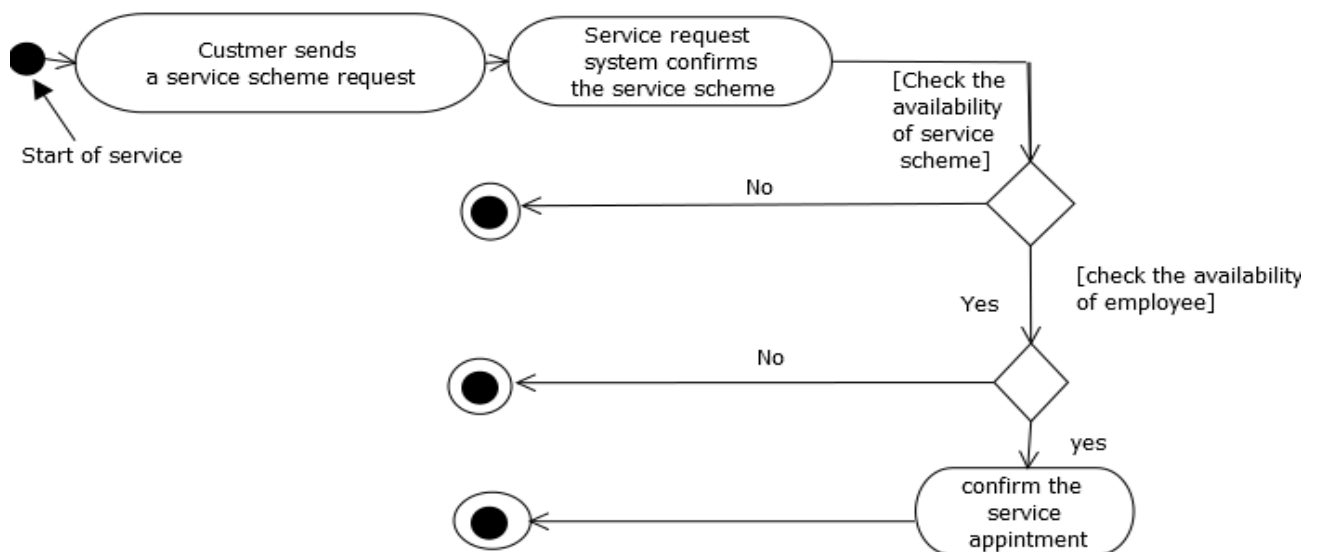
P1.4.7 Statechart Diagram

Any real-time system is expected to be reacted by some kind of internal/external events. These events are responsible for state change of the system. Statechart diagram is used to represent the event driven state change of a system. It basically describes the state change of a class, interface, etc. State chart diagram is used to visualize the reaction of a system by internal/external factors.



P1.4.8 Activity Diagram

Activity diagram describes the flow of control in a system. It consists of activities and links. The flow can be sequential, concurrent, or branched. Activities are nothing but the functions of a system. Numbers of activity diagrams are prepared to capture the entire flow in a system. Activity diagrams are used to visualize the flow of controls in a system. This is prepared to have an idea of how the system will work when executed.



P1.5 System Specification

P1.5.1 Hardware Specification

Processor: Intel core i3 | RAM:4 GB | Hard Disk:500 GB

P1.5.2 Software Specification

Front End: HTML5, Bootstrap | Back End: PHP

DB Connectivity: MySQL

Technologies Used: HTML5, CSS, Bootstrap, AJAX, PHP

P1.6 Software Description

P1.6.1 PHP

PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language. PHP is now installed on more than 244 million websites and 2.1 million web servers. Originally created by Rasmus Lerdorf in 1995, the reference implementation of PHP is now produced by the PHP group. While PHP originally stood for personal Home page, it now stands for PHP: Hypertext Pre-processor, a recursive acronym. PHP code is interpreted by a web server with a PHP processor module which generates the resulting web page. PHP commands can be embedded directly into a HTML source document rather than calling an external file to process data. It has also evolved to include a command-line interface capability and can be used in standalone incompatible with the GNU General Public License (GPL) due to restrictions on the usage of the term PHP. PHP can be deployed on most web servers and also as a standalone shell on almost every operating system and platform, free of charge.

P1.6.2 MySQL

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation.

The MySQL Web site provides the latest information about MySQL software.

- **MySQL is a database management system.**

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

- **MySQL databases are relational.**

A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and “pointers” between

different tables. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data. The SQL part of “MySQL” stands for “Structured Query Language”. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax.

SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, “SQL92” refers to the standard released in 1992, “SQL:1999” refers to the standard released in 1999, and “SQL:2003” refers to the current version of the standard.

- **MySQL software is Open Source.**

Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License), to define what you may and may not do with the software in different situations. If you feel uncomfortable with the GPL or need to embed MySQL code into a commercial application, you can buy a commercially licensed version from us. See the MySQL Licensing Overview for more information.

- **The MySQL Database Server is very fast, reliable, scalable, and easy to use.**

If that is what you are looking for, you should give it a try. MySQL Server can run comfortably on a desktop or laptop, alongside your other applications, web servers, and so on, requiring little or no attention. If you dedicate an entire machine to MySQL, you can adjust the settings to take advantage of all the memory, CPU power, and I/O capacity available. MySQL can also scale up to clusters of machines, networked together.

MySQL Server was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Although under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet.

- **MySQL Server works in client/server or embedded systems.**

The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different backends, several different client programs and libraries,

administrative tools, and a wide range of application programming interfaces (APIs).

We also provide MySQL Server as an embedded multi-threaded library that you can link into your application to get a smaller, faster, easier-to-manage standalone product.

- **A large amount of contributed MySQL software is available.**

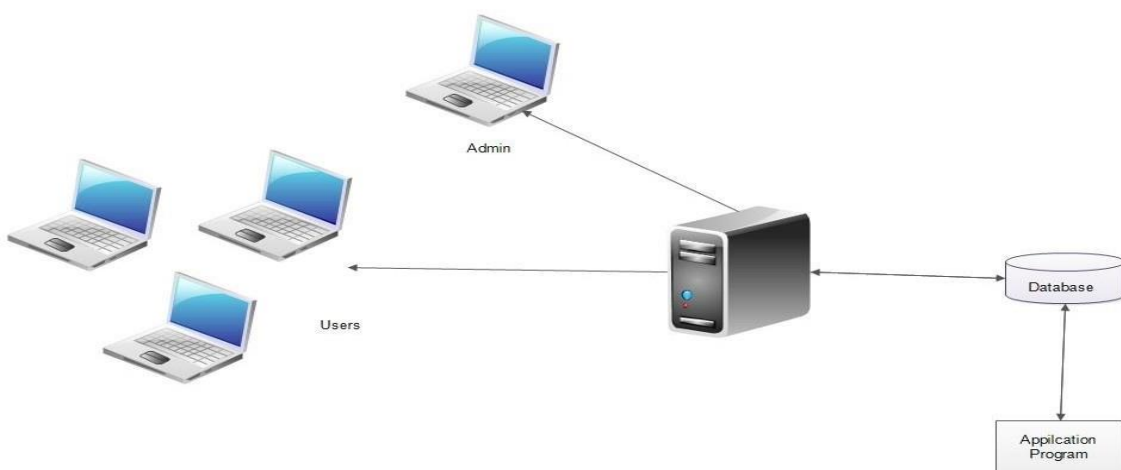
MySQL Server has a practical set of features developed in close cooperation with our users. It is very likely that your favourite application or language supports the MySQL Database Server.

P1.7 System Design

Design is the first step into the development phase for any engineered product or system. Design is a creative process. A good design is the key to effective system. The term “design” is defined as “the process of applying various techniques and principles for the purpose of defining a process or a system in sufficient detail to permit its physical

realization”. It may be defined as a process of applying various techniques and principles for the purpose of defining a device, a process or a system in sufficient detail to permit its physical realization. Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm that is used. The system design develops the architectural detail required to build a system or product. As in the case of any systematic approach, this software too has undergone the best possible design phase fine tuning all efficiency, performance and accuracy levels. The design phase is a transition from a user-oriented document to a document to the programmers or database personnel. System design goes through two phases of development: Logical and Physical Design

P1.7.1 Architectural Design



The registered user, admin, service centres and employees can access the real deal cars through internet using their Laptop, Smart Phone, Tablet or Desktop Computer. The System's application program processes the user's request and provides the required services by taking data from the system database

P1.7.2 Module Design

Admin Module

The administrator is allowed to access all the services in the system. And approve vehicle services centres, and overall management of the system.

Approve Service centres	Overall management of the system
-------------------------	----------------------------------

User Module

After registration, customers can book appointments for vehicle service online and also customer can Buy second hand vehicles.

User registration, login	Register cars, Delete cars
Book appointment, View service history	Manage profile
Sell cars	Search Used Vehicles, Buy cars

Service centre Module

The service centres can register their centres, add new service schemes, and manage the employee leave and appointments.

Register Service centre, login	Add service scheme
Add service history	Manage employee leave

Employee Module

Update profile, login	Apply leave
Add service history	Manage appointments

P1.7.3 Database Design

A database is an organized mechanism that has the capability of storing information through which a user can retrieve stored information in an effective and efficient manner. The data is the purpose of any database and must be protected.

The database design is a two-level process. In the first step, user requirements are gathered together and a database is designed which will meet these requirements as clearly as possible. This step is called Information Level Design and it is taken independent of any individual DBMS.

In the second step, this Information level design is transferred into a design for the specific DBMS that will be used to implement the system in question. This step is called Physical Level Design, concerned with the characteristics of the specific DBMS that will be used. A database design runs parallel with the system design. The organization of the data in the database is aimed to achieve the following two major objectives.

- Data Integrity
- Data independence

Relational Database Management System (RDBMS)

A relational model represents the database as a collection of relations. Each relation resembles a table of values or file of records. In formal relational model terminology, a row is called a tuple, a column header is called an attribute and the table is called a relation. A relational database consists of a collection of tables, each of which is assigned a unique name. A row in a table represents a set of related values.

Relations, Domains & Attributes

A table is a relation. The rows in a table are called tuples. A tuple is an ordered set of n elements.

Columns are referred to as attributes. Relationships have been set between every table in the database. This ensures both Referential and Entity Relationship Integrity. A domain D is a set of atomic values. A common method of specifying a domain is to specify a data type from which the data values forming the domain are drawn. It is also useful to specify a name for the domain to help in interpreting its values.

Every value in a relation is atomic, that is not decomposable.

Relationships

- Table relationships are established using Key. The two main keys of prime importance are Primary Key & Foreign Key. Entity Integrity and Referential Integrity Relationships can be established with these keys.
- Entity Integrity enforces that no Primary Key can have null values.
- Referential Integrity enforces that no Primary Key can have null values.
- Referential Integrity for each distinct Foreign Key value, there must exist a matching Primary Key value in the same domain. Other keys are Super Key and Candidate Keys.

Normalization

Data are grouped together in the simplest way so that later changes can be made with minimum impact on data structures. Normalization is formal process of data structures in manners that eliminates redundancy and promotes integrity. Normalization is a technique of separating redundant fields and breaking up a large table into a smaller one. It is also used to avoid insertion, deletion, and updating anomalies. Normal form in data modelling use two concepts, keys and relationships. A key uniquely identifies a row in a table. There are two types of keys, primary key and foreign key. A primary key is an element or a combination of elements in a table whose purpose is to identify records from the same table. A foreign key is a column in a table that uniquely identifies record from a different table. All the tables have been normalized up to the third normal form.

As the name implies, it denotes putting things in the normal form. The application developer via normalization tries to achieve a sensible organization of data into proper tables and columns and where names can be easily correlated to the data by the user. Normalization eliminates repeating groups at data and thereby avoids data redundancy which proves to be a great burden on the computer resources. These include:

- ✓ Normalize the data.
- ✓ Choose proper names for the tables and columns.
- ✓ Choose the proper name for the data.

First Normal Form

The First Normal Form states that the domain of an attribute must include only atomic values and that the value of any attribute in a tuple must be a single value from the domain of that attribute. In other words, 1NF disallows “relations within relations” or “relations as attribute values within tuples”. The only attribute values permitted by 1NF are single atomic or indivisible values. The first step is to put the data into First Normal Form. This can be done by moving data into separate tables where the data is of similar type in each table. Each table is given a Primary Key or Foreign Key as per requirement of the project. In this we form new relations for each non-atomic attribute or nested relation. This eliminated repeating groups of data. A relation is said to be in first normal form if only if it satisfies the constraints that contain the primary key only.

Second Normal Form

According to Second Normal Form, for relations where primary key contains multiple attributes, no non-key attribute should be functionally dependent on a part of the primary key. In this we decompose and setup a new relation for each partial key with its dependent attributes. Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it. This step helps in taking out data that is only dependent on a part of the key. A relation is said to be in second normal form if and only if it satisfies all the first normal form conditions for the primary key and every non-primary key attributes of the relation is fully dependent on its primary key alone.

Third Normal Form

According to Third Normal Form, Relation should not have a non-key attribute functionally determined by another non-key attribute or by a set of non-key attributes. That is, there should be no transitive dependency on the primary key. In this we decompose and set up relation that includes the non-key attributes that functionally determines other non-key attributes. This step is taken to get rid of anything that does not depend entirely on the Primary Key. A relation is said to be in third normal form if only if it is in second normal form and more over the non key attributes of the relation should not be depend on another non-key attribute.

TABLES**Table no: 1****Table name: tbl_login****Primary key: user_id**

Sl. No	Field name	Field type	Size	Constraints	Description
1.	user_id	int	10	Primary key	User id
2.	email	varchar	30	Not Null	E-mail
3.	password	varchar	100	Not Null	Password
4.	Designation_id	int	10	Foreign Key	Designation type
5.	status	int	5	Not Null	Status of user

Table no:2**Table name: tbl_users****Foreign key: user_id, place_id**

Sl. No	Field name	Field type	Size	Constraints	Description
1.	user_id	int	10	Foreign Key	User id
2.	place_id	int	15	Foreign key	Place id
3.	first_name	varchar	15	Not Null	First name
4.	last_name	varchar	15	Not Null	Last name
5.	mobile	int	15	Not Null	Mobile Number
6.	photo	varchar	50	Not null	Profile photo

Table No: 3**Table Name: tbl_servicecenter****Primary key: licenceno****Foreign key: user_id, place_id**

Sl. no	Fieldname	Field type	Size	Constraints	Description
1	licenceno	Varchar	20	Primary key	License number
2	user_id	int	10	Foreign Key	Login id
3	place_id	int	10	Foreign key	Place id
4	brand_id	int	10	Foreign key	Brand id
5	center_name	varchar	25	Not Null	Center name
6	certificate	varchar	30	Not Null	License certificate
7	Photo	varchar	50	Not Null	Service center photo
8	mobile	int	15	Not Null	Mobile number

Table No: 4**Table Name: tbl_district****Primary Key: district_id**

Sl. no	Fieldname	Field type	Size	Constraints	Description
1.	district_id	int	10	Primary key	District id
2.	district	Varchar	10	Not Null	District name

Table No: 5**Table Name: tbl_place****Primary Key: place_id****Foreign key: district_id**

Sl. no	Field name	Fieldtype	Size	Constraints	Description
1.	place_id	int	10	Primary key	Place id
2.	district_id	int	10	Foreign key	District id
3.	place	varchar	25	Not Null	Place name

Table No: 6**Table Name: tbl_brand****Primary Key: brand_id**

Sl. no	Field name	Field Type	Size	Constraints	Description
1.	brand_id	int	10	Primary Key	Brand id
2.	brand_name	varchar	10	Primary key	Brand name

Table No:7**Table Name: tbl_model****Primary Key: model_id****Foreign Key: brand_id**

Sl. no	Field name	Field type	Size	Constraints	Description
1.	model_id	int	10	Primary key	Model id
2.	brand_id	int	10	Foreign key	Brand id
3.	model_name	varchar	10	Foreign key	Model name

Table No: 8**Table Name: tbl_variant****Primary Key: variant_id****Foreign Key: model_id, fuel_id**

Sl. no	Field name	Field Type	Size	Constraints	Description
1.	variant_id	int	10	Primary Key	Variant id
2.	model_id	int	10	Foreign key	Model id
3.	fuel_id	int	10	Foreign key	Fuel id
4.	variant_name	varchar	15	Not Null	Variant

Table No: 9**Table Name: tbl_fuel****Primary Key: fuel_id**

Sl. no	Field name	Field type	Size	Constraints	Description
1.	fuel_id	int	10	Primary Key	Fuel id
2.	fuel	varchar	10	Not null	Fuel type

Table No:10**Table Name: tbl_servicescheme****Primary Key: scheme_id****Foreign Key: licenceno, variant_id, department_id, servicetype_id**

Sl. no	Field name	Field type	Size	Constraints	Description
1.	scheme_id	int	10	Primary Key	Service scheme id
2.	licenceno	varchar	25	Foreign key	License number
3.	variant_id	int	10	Foreign key	Variant id
4.	department_id	int	10	Foreign key	Department id
5.	servicetype_id	int	10	Foreign key	Service type id
6.	km	int	10	Not null	Odometer reading
7.	amount	int	10	Not null	Advance

Table No: 11**Table Name: tbl_department****Primary Key: department_id**

Sl. no	Field name	Field type	Size	Constraints	Description
1.	department_id	int	10	Primary Key	Department id
2.	department_name	varchar	25	Not null	Department type

Table No: 12**Table Name: tbl_workcount****Primary Key: count_id****Foreign Key: licenceno**

Sl. no	Field name	Field type	Size	Constraints	Description
1.	Count_id	int	10	Primary key	Count id
2.	department_id	int	10	Foreign key	Department id
3.	date	date		Not null	Date
4.	count	int	10	Not null	Work count
5.	licenceno	varchar	10	Foreign key	License number of service center

Table No: 13**Table Name: tbl_images****Primary Key: image_id****Foreign Key: advertisement_id**

Sl. No	Field name	Field type	Size	Constraints	Description
1.	image_id	int	10	Primary key	Image id
2.	advertisement_id	int	10	Foreign key	Advertisement id
3.	image1	varchar	100	Not null	Car image11
4.	image2	varchar	100	Not null	Car image2
5.	Image3	varchar	100	Not null	Car image2
6.	Image4	varchar	100	Not null	Car image2

Table No: 14**Table Name: tbl_car****Primary Key: car_id****Foreign Key: user_id, variant_id**

Sl. no	Field name	Field type	Size	Constraints	Description
1.	Car_id	Int	10	Primary key	Car id
2.	user_id	int	10	Foreign key	User id
3.	variant_id	int	10	Foreign key	Variant id
4.	manufactured_year	int	10	Not null	Manufacturing year
5.	color	varchar	10	Not null	Car color
6.	regno	varchar	20	Not null	Registration number
7.	engineno	varchar	18	Not null	Engine number
8.	chasisno	varchar	18	Not null	Chasis number
9.	rcbook	varchar	30	Not null	Rc book image
10.	photo	varchar	30	Not null	Car image
11.	status	int	10	Not null	Status

Table No: 15**Table Name: tbl_incomplete****Primary Key: incomplete_id****Foreign key: appointment_id**

Sl. No	Field name	Field type	Size	Constraints	Description
1.	incomplete_id	int	10	Primary key	Action id
2.	appointment_id	int	10	Foreign key	Appointment_id
3.	reason	varchar	300	Not null	Reason of incomplete
4.	delivery_date	varchar	20	Not null	Expected work completion date

Table No: 16**Table Name: tbl_appointment****Primary Key: apid****Foreign Key: scheme_id, licenceno**

Sl. No	Field name	Field type	Size	Constraints	Description
1.	appointment_id	int	10	Primary key	Appointment id
2.	registerno	varchar	20	Not null	Car register no
3.	scheme_id	int	10	Foreign key	Scheme id
4.	licenceno	varchar	20	Foreign key	Service center license number
5.	appointment_date	date		Not null	Date
6.	remarks	varchar	500		Remarks from service center
7.	odometer	int	10	Not null	Odometer reading
8.	book_date	varchar	20	Not null	Booking made date
9.	appointment_status	int	10	Not null	Appointment status

Table No: 17**Table Name: tbl_transaction****Primary Key: transaction_id****Foreign Key: user_id**

Sl. No	Field name	Field type	Size	Constraints	Description
1.	transaction_id	int	10	Primary key	Transaction id
2.	paid_from	int	10	Foreign key	User_id of payer
3.	paid_to	int	15	Foreign key	User_id of receiver
4.	transaction_date	date		Not null	Date
5.	amount	int	10	Not null	Amount paid
6.	transaction_type	varchar	50	Not null	Transaction type

Table No: 18**Table Name: tbl_checking****Primary Key: checking_id****Foreign Key: scheme_id, spare_id**

Sl. No	Field name	Field type	Size	Constraints	Description
1.	checking_id	int	10	Primary key	Checking id
2.	scheme_id	int	10	Foreign key	Service scheme id
3.	spare_id	int	10	Foreign key	Spare id

Table No: 19**Table Name: tbl_replacing****Primary Key: replacing_id****Foreign Key: scheme_id, spare_id**

Sl. No	Field name	Field type	Size	Constraints	Description
1.	replacing_id	int	10	Primary key	Replacing id
2.	scheme_id	int	10	Foreign key	Service scheme id
3.	spare_id	int	10	Foreign key	Spare id

Table No: 20**Table Name: tbl_carcondition****Primary Key: condition_id****Foreign Key: appointment_id, employee_id**

Sl. No	Field name	Field type	Size	Constraints	Description
1.	condition_id	int	10	Primary key	Condition id
2.	appointment_id	int	10	Foreign key	Appointment id
3.	employee_id	int	10	Foreign key	Employee id
4.	started_time	timestamp		Not null	Work started time
5.	odometer	int	10	Not null	Odometer reading
6.	fuel	varchar	20	Not null	Fuel condition
7.	damage	varchar	500		Damages of car

Table No: 21**Table Name: tbl_advertisement****Primary Key: advertisement_id****Foreign Key: car_id**

Sl. No	Field name	Field type	Size	Constraints	Description
1.	advertisement_id	int	10	Primary key	Advertisement id
2.	car_id	int	10	Foreign key	Car id
3.	price	int	10	Not null	Expecting price
4.	odometer	int	10	Not null	KM driven
5.	description	varchar	50	Not null	Description about car
6.	latitude	double		Not null	Latitude of location
7.	longitude	double		Not null	Longitude of location
8.	advertisement_date	varchar	20	Not null	Advertisement posted date
9.	status	int	10	Not null	Status of advertisement

Table No: 22**Table Name: tbl_leave****Primary Key: leave_id****Foreign Key: employee_id**

Sl. No	Field name	Field type	Size	Constraints	Description
1.	leave_id	int	10	Primary key	Leave id
2.	employee_id	int	10	Foreign key	Employee id
3.	date	date			Date of leave
4.	reason	varchar	50	Not null	Leave reason
5.	status	int	10	Not null	Leave status

Table No: 23**Table Name: tbl_employee****Primary Key: employee_id****Foreign Key: licenceno, department_id,user_id,place_id**

Sl. No	Field name	Field type	Size	Constraints	Description
1.	employee_id	int	10	Primary key	Employee id
2.	licenceno	varchar	15	Foreign key	Licence number
3.	department_id	int	10	Foreign key	Department id
4.	first_name	varchar	25	Not null	Employee first name
5.	last_name	varchar	25	Not null	Employee last name
6.	user_id	int	10	Foreign key	User id of employee
7.	Mobileno	int	10	Not null	Mobile number
8.	place_id	int	10	Foreign key	Place id
9.	photo	varchar	200	Not null	Image of employee
10.	status	int	10	Not null	Status of employee

Table No: 24**Table Name: tbl_employeecount****Primary Key: count_id****Foreign Key: licenceno, department_id**

Sl. No	Field name	Field type	Size	Constraints	Description
1.	count_id	int	10	Primary key	Employee count id
2.	licenceno	varchar	20	Foreign key	License Number
3.	department_id	int	10	Foreign key	Department id
4.	maximum	int	10	Not null	Maximum no of employees

Table No: 25**Table Name: tbl_designation****Primary Key: designation_id**

Sl. No	Field name	Field type	Size	Constraints	Description
1.	designation_id	int	10	Primary key	Designation id
2.	designation	varchar	20	Not null	Designation

Table No: 26**Table Name: tbl_offeredprice****Primary Key: offer_id****Foreign Key: advertisement_id, user_id**

Sl. No	Field name	Field type	Size	Constraints	Description
1.	offer_id	int	10	Primary key	Offer id
2.	advertisement_id	int	10	Foreign key	Advertisement id
3.	user_id	int	10	Foreign key	User id
4.	offer_amount	int	10	Not null	Offered price
5.	offer_date	varchar	20	Not null	Offered date
6.	offer_status	int	10	Not null	Offer status

Table No: 27**Table Name: tbl_servicetype****Primary Key: servicetype_id**

Sl. No	Field name	Field type	Size	Constraints	Description
1.	servicetype_id	int	10	Primary key	Service type id
2.	servicetype	varchar	20	Not null	Service type

Table No: 28**Table Name: tbl_spare****Primary Key: spare_id**

Sl. No	Field name	Field type	Size	Constraints	Description
1.	spare_id	int	10	Primary key	Spare id
2.	spare	varchar	20	Not null	Spare name

P1.8 System Testing

P1.8.1 Introduction

Software Testing is the process of executing software in a controlled manner, in order to answer the question - Does the software behave as specified? Software testing is often used in association with the terms verification and validation. Validation is the checking or testing of items, includes software, for conformance and consistency with an associated specification. Software testing is just one kind of verification, which also uses techniques such as reviews, analysis, inspections, and walkthroughs. Validation is the process of checking that what has been specified is what the user actually wanted.

Validation: Are we doing the right job?

Verification: Are we doing the job right?

Software testing should not be confused with debugging. Debugging is the process of analysing and localizing bugs when software does not behave as expected. Although the identification of some bugs will be obvious from playing with the software, a methodical approach to software testing is a much more thorough means for identifying bugs. Debugging is therefore an activity which supports testing, but cannot replace testing.

Other activities which are often associated with software testing are static analysis and dynamic analysis. Static analysis investigates the source code of software, looking for problems and gathering metrics without actually executing the code. Dynamic analysis looks at the behaviour of software while it is executing, to provide information such as execution traces, timing profiles, and test coverage information.

Testing is a set of activity that can be planned in advanced and conducted systematically. Testing begins at the module level and work towards the integration of entire computers-based system. Nothing is complete without testing, as it vital success of the system testing objectives, there are several rules that can serve as testing objectives. They are

Testing is a process of executing a program with the intent of finding an error.

- A good test case is one that has high possibility of finding an undiscovered error.
- A successful test is one that uncovers an undiscovered error.

If a testing is conducted successfully according to the objectives as stated above, it would uncover errors in the software. Also testing demonstrate that the software function appears to be working according to the specification, that performance requirement appears to have been met.

There are three ways to test program.

- For correctness
- For implementation efficiency
- For computational complexity

Test for correctness are supposed to verify that a program does exactly what it was designed to do. This is much more difficult than it may at first appear, especially for large programs.

P1.8.2 Test Plan

A test plan implies a series of desired course of action to be followed in accomplishing various testing methods. The Test Plan acts as a blue print for the action that is to be followed. The software engineers create a computer program, its documentation and related data structures. The software developers is always responsible for testing the individual units of the programs, ensuring that each performs the function for which it was designed. There is an independent test group (ITG) which is to remove the inherent problems associated with letting the builder to test the thing that has been built. The specific objectives of testing should be stated in measurable terms. So that the mean time to failure, the cost to find and fix the defects, remaining defect density or frequency of occurrence and test work-hours per regression test all should be stated within the test plan.

The levels of testing include:

- Unit testing
- Integration Testing
- Data validation Testing
- Output Testing

P1.8.2.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of software design – the software component or module. Using the component level design description as a guide, important control paths are tested to uncover errors within the boundary of the module.

The relative complexity of tests and uncovered scope established for unit testing. The unit testing is white-box oriented, and step can be conducted in parallel for multiple components. The modular interface is tested to ensure that information properly flows into and out of the program unit under test. The local data structure is examined to ensure that data stored temporarily maintains its integrity during all steps in an algorithm's execution. Boundary conditions are tested to ensure that all statements in a module have been executed at least once. Finally, all error handling paths are tested.

Tests of data flow across a module interface are required before any other test is initiated. If data do not enter and exit properly, all other tests are moot. Selective testing of execution paths is an essential task during the unit test. Good design dictates that error conditions be anticipated and error handling paths set up to reroute or cleanly terminate processing when an error does occur. Boundary testing is the last task of unit testing step. Software often fails at its boundaries.

Unit testing was done by treating each module as separate entity and testing each one of them with a wide spectrum of test inputs. Some flaws in the internal logic of the modules were found and were rectified. After coding each module is tested and run individually. All unnecessary code were removed and ensured that all modules are working, and gives the expected result.

P1.8.2.2 Integration Testing

Integration testing is systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. The objective is to take unit tested components and build a program structure that has been dictated by design. The entire program is tested as whole. Correction is difficult because isolation of causes is complicated by vast expanse of entire program. Once these errors are corrected, new ones appear and the process continues in a seemingly endless loop.

After performing unit testing in the System all the modules were integrated to test for any inconsistencies in the interfaces. Moreover, differences in program structures were removed and a unique program structure was evolved.

P1.8.2.3 Validation Testing

This is the final step in testing. In this the entire system was tested as a whole with all forms, code, modules and class modules. This form of testing is popularly known as Black Box testing or System tests.

Black Box testing method focuses on the functional requirements of the software. That is, Black Box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program.

Black Box testing attempts to find errors in the following categories; incorrect or missing functions, interface errors, errors in data structures or external data access, performance errors and initialization errors and termination errors.

P1.8.2.4 User Acceptance Testing

The system considered is tested for user acceptance; here it should satisfy the firm's need. The software should keep in touch with perspective system; user at the time of developing and making changes whenever required. This done with respect to the following points:

- Input Screen Designs,
- Output Screen Designs,

The above testing is done taking various kinds of test data. Preparation of test data plays a vital role in the system testing. After preparing the test data, the system under study is tested using that test data. While testing the system by which test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

P1.9 Implementation

Implementation is the stage of the project where the theoretical design is turned into a working system. It can be considered to be the most crucial stage in achieving a successful new system gaining the users confidence that the new system will work and will be effective and accurate. It is primarily concerned with user training and documentation. Conversion usually takes place about the same time the user is being trained or later. Implementation simply means convening a new system design into operation, which is the process of converting a new revised system design into an operational one.

At this stage the main work load, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned or controlled, it can create chaos and confusion.

Implementation includes all those activities that take place to convert from the existing system to the new system. The new system may be a totally new, replacing an existing manual or automated system or it may be a modification to an existing system. Proper implementation is essential to provide a reliable system to meet organization requirements. The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert from the old system to the new system. The system can be implemented only after through testing is done and if it is found to be working according to the specifications. The system personnel check the feasibility of the system. The more complex the system being implemented, the more involved will be the system analysis and design effort required to implement the three main aspects: education and training, system testing and changeover.

The implementation state involves the following tasks:

- Careful planning.
- Investigation of system and constraints.
- Design of methods to achieve the changeover. Training of the staff in the changeover phase.

P1.9.1 Implementation Procedure

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended uses and the operation of the system. In many organizations someone who will not be operating it, will commission the software development project. In the initial stage people doubt about the software but we have to ensure that the resistance does not build up, as one has to make sure that:

- ☐ The active user must be aware of the benefits of using the new system.
- ☐ Their confidence in the software is built up.
- ☐ Proper guidance is imparted to the user so that he is comfortable in using the application.

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not up running on the server, the actual process won't take place

P1.9.2 User Training

User training is designed to prepare the user for testing and converting the system. To achieve the objective and benefits expected from computer-based system, it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for training is more important. By user training the user comes to know how to enter data, respond to error messages, interrogate the database and call up routine that will produce reports and perform other necessary functions.

Training on the Application Software

After providing the necessary basic training on computer awareness the user will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the date entered. It should then cover information needed by the specific user/ group to use the system or part of the system while imparting the training of the program on the application. This training may be different across different user groups and across different levels of hierarchy.

P1.9.3 Operational Document

After providing the necessary basic training on computer awareness the user will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the date entered. It should then cover information needed by the specific user/ group to use the system or part of the system while imparting the training of the program on the application. This training may be different across different user groups and across different levels of hierarchy.

P1.9.4 System Maintenance

Maintenance is the enigma of system development. The maintenance phase of the software cycle is the time in which a software product performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle. The need for system maintenance is for it to make adaptable to the changes in the system environment. Software maintenance is of course, far more than "Finding Mistakes".

P1.10 Conclusion &Future Enhancements

P1.10.1 Future Enhancement

- The system is designed in such a way that the payment of service provider should be done in completely online mode.
- Provide more security

P1.10.2 CONCLUSION

The software reduces the time consumption and the manual efforts of searching a product. It will be a simple platform for users to access services for their huge needs.

The benefits, we can obtain from the new system are:

- Timely and accurate information will be available
- Reduced data loss
- The access time and process time is highly reduced
- Quick data view
- Error free output

The proposed system is expected to replace manual system and provide more efficient performance and services.

P1.11 BIBLIOGRAPHY REFERENCES:**WEBSITES:**

- www.w3schools.com
- www.jquery.com
- <http://homepages.dcc.ufmg.br/~rodolfo/es-1-03/IEEE-Std-830-1998.pdf>
- www.agilemodeling.com/artifacts/useCaseDiagram.html

P1.12 APPENDIX

P1.12.1 SAMPLE CODE

Appointment Booking Form

```
<?php
require "data/connect.php";
require "data/session.php";
require('layouts/app_top');
$regno=getSession('regno');
$brandid=getSession('brandid');
$modelid=getSession('modelid');
$variantid=getSession('variantid');
?>
<html>
<head>
<style>
.image1 {
    max-width: 100%;
}
.button1 {
    background-color: #aeaeaeed;
    max-height: 20%;
}
</style>
</head>
<body>
<!-- <?php
// print_r($modelid);
// return;
?> -->
<div class="view full-page-intro" >
<!-- Navbar -->
<nav class="navbar fixed-top navbar-expand-lg navbar-dark scrolling-navbar">
<div class="container">
<!-- Brand -->
<a class="navbar-brand" href="user.php">
<strong>Home</strong>
</a>
<button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent" aria-controls="navbarSupportedContent"
aria-expanded="false" aria-label="Toggle navigation">
<span class="navbar-toggler-icon"></span>
</button>
<!-- Links -->
<div class="collapse navbar-collapse" id="navbarSupportedContent">
<ul>
</div>
</div>
```

```

</div>
</nav>
<div class="main">
  <!-- Content -->
  <div class="container">
    <!--Grid row-->
    <div class="row wow fadeIn">
      <!--Grid column-->
      <div class="offset-4 col-md-4 mb-4">
<!--Card-->
<div class="card">
  <!--Card content-->
  <div class="card-body">
    <!-- Form -->
    <form name="" id="login" method="post" action="data/userdata.php"
enctype="multipart/form-data" class="mt-5">
      <!-- Heading -->
      <input type="text" hidden value="appointment" name="type">
      <input hidden name="date" value="<?php echo date("m/d/Y"); ?>>
      <input type="text" hidden value="<?php echo $_POST['licenceno']?>" name="licenceno">
      <input type="text" hidden value="<?php echo $regno?>" name="regno">
      <h3 class="dark-grey-text text-center">
        <strong>Make An Appointment</strong>
      </h3>
      <hr>
      <table>
        <tr>
          <td>Pick a Date</label></td>
          <td>
            <div class="md-form">
              <input type="text" readonly id="update" class="form-control" name="datepicker" data-
type="dat" >
              <!-- datepicker -->
            </div>
          </td>
        </tr>
        <!-- <tr>
          <td>Vehicle number</label></td>
          <td>
            <div class="md-form">
              <?php
                // $regno=getSession('regno');
                // print_r($regno);
                // return;
              ?>
              <label ></label>
            </div>
          </td>
        </tr> -->
        <tr>
          <td><label>Choose Service Type</label></td>

```

```

        <td>
        <div class="md-form">
        <!--<input type="" id="form3" class="form-control" name="fanme"> -->
        <select class="form-control" name="stype" id="stype" required>
        <?php
        include('data/service.php');
        ?>
        </select >
        </div>
        </td>
    </tr>
    <tr>
    <td><label>Advance Payment Amount</label></td>
    <td>
    <div class="md-form">
    <input type="text" readonly class="form-control" name="price" id="price">
    </div>
    </td>
    </tr>
    <tr>
    <td><label>Odometer Reading</label></td>
    <td>
    <div class="md-form">
    <input type="text" class="form-control validate" name="odometer" id="odometer" data-
type="digits" required>
    </div>
    </td>
    </tr>
    <tr>
    <td><label>Remarks</label></td>
    <td>
    <div class="md-form">
    <textarea rows="3" class="form-control" name="remarks" id="remarks"></textarea>
    </div>
    </td>
    </tr>
    </table>
    <div class="text-center">
    <input type="submit" class="btn btn-indigo" value="Book">
    <hr>
    </fieldset>
    </div>
</form>
<!-- Form -->
</div>
</div>
<!--/.Card-->
</div>
<!--Grid column-->

```

```

</div>
<!--Grid row-->
</div>
<!-- Content -->
</div>
<?php
require('layouts/app_end');
?>
</div>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>jQuery UI Datepicker - Default functionality</title>
<link rel="stylesheet" href="assets/css/dtpicker.css">
<!-- //code.jquery.com/ui/1.12.1/themes/base/jquery-ui.css -->
<link rel="stylesheet" href="/resources/demos/style.css">
<!-- <script src="https://code.jquery.com/jquery-1.12.4.js"></script>
<script src="https://code.jquery.com/ui/1.12.1/jquery-ui.js"></script> -->
<script>
</script>
<script>
    var d = new Date();
    var year = d.getFullYear();
    d.setFullYear(year);
    $('#update').datepicker({ changeYear: true, changeMonth: true, maxDate:'7d',minDate:'2d',
defaultDate: d});
</script>
</body>
</html>

```

PHP Code

```

function makeAppointment($conn)
{
    $date = $_POST['datepicker'];
    $bdate = $_POST['date'];
    $vehno = $_POST['regno'];
    $stype = $_POST['stype'];
    $meter = $_POST['odometer'];
    $remarks = $_POST['remarks'];
    $licenceno = $_POST['licenceno'];
    $price = $_POST['price'];
    setSession('date', $date);
    setSession('bdate', $bdate);
    setSession('vehno', $vehno);
    setSession('stype', $stype);
    setSession('meter', $meter);
    setSession('remarks', $remarks);
    setSession('licenceno', $licenceno);
    setSession('price', $price);
    $sql2 = "SELECT `department_id`,`km`,`servicetype_id` FROM `tbl_servicescheme`
WHERE `scheme_id`='".$stype"'";

```



```

$tid = mysqli_query($conn, $sql2);
$data2 = mysqli_fetch_array($tid);
$deptid = $data2['department_id'];
setSession('deptid', $deptid);
$km = $data2['km'];
$type1 = $data2['servicetype_id'];
$sql = "SELECT servicetype FROM tbl_servicetype WHERE servicetype_id='$type1'";
$st = mysqli_query($conn, $sql);
$da = mysqli_fetch_assoc($st);
$name = $da['servicetype'];
if ($name == 'First Service' || 'Second Service' || 'Third Service' || 'Fourth Service' || 'Fifth
Service') {

    if ($km < $meter - 200) {
        echo "<script>alert('You Exceeds the Kilometer Limit of the Service
Type');window.location='../appointment.php';</script>";
        return;
    } else { ?>
        <script>
            window.location = '../payment.php';
        </script>
    <?php
    }
} else {
    if ($km < $meter - 2000) {
        echo "<script>alert('You Exceeds the Kilometer Limit of the Service
Type');window.location='../appointment.php';</script>";
        return;
    } else { ?>
        <script>
            window.location = '../payment.php';
        </script>
    <?php
    }
}
}
}
function payment($conn)
{
    $tdate = $_POST['tdate'];
    $date = getSession('date');
    $bdate = getSession('bdate');
    $vehno = getSession('vehno');
    $stype = getSession('stype');
    $meter = getSession('meter');
    $remarks = getSession('remarks');
    $licenceno = getSession('licenceno');
    $price = getSession('price');
    $deptid = getSession('deptid');
    $userid = getSession('user_id');
    $sql22 = "SELECT user_id FROM tbl_servicecenter WHERE licenceno='$licenceno'";

```

```

$val22 = mysqli_query($conn, $sql22);
$result22 = mysqli_fetch_assoc($val22);
$scid = $result22['user_id'];
//find maximum capacity of srvice center
$sql3 = "SELECT `maximum` FROM `tbl_employeecount` WHERE
`licenceno`=' $licenceno' AND `department_id`=' $deptid'";
$max = mysqli_query($conn, $sql3);
$data3 = mysqli_fetch_assoc($max);
$maxcount = $data3['maximum'];
//finding the booking for service type on a particular day
$sql4 = "SELECT * FROM `tbl_workcount` WHERE `licenceno`=' $licenceno' AND
`date`=' $date' AND `department_id`=' $deptid'";
$count = mysqli_query($conn, $sql4);
if (mysqli_num_rows($count) < 1) {
    //table is empty directly into both tables
    //checking already applied or not
    $sql11 = "SELECT * FROM `tbl_appointment` WHERE `registerno`=' $vehno' AND
`scheme_id`=' $stype' AND `appointment_status`!='-1'";
    $count11 = mysqli_query($conn, $sql11);
    if (mysqli_num_rows($count11) < 1) {
        $sql12 = "SELECT * FROM `tbl_appointment` WHERE `registerno`=' $vehno' AND
`appointment_date`=' $date' AND `appointment_status`!='-1' AND `appointment_status`!='3'";
        $count12 = mysqli_query($conn, $sql12);
        if (mysqli_num_rows($count12) < 1) {

            $sql5 = "INSERT INTO `tbl_workcount`(`date`,`licenceno`,`department_id`,`count`)
VALUES ('$date','$licenceno','$deptid',1)";
            mysqli_query($conn, $sql5);
            $sql6 = "INSERT INTO `tbl_appointment`(`registerno`,`licenceno`,
`scheme_id`,`bookdate`,`appointment_date`,`odometer`,`remarks`,`appointment_status`)
VALUES ('$vehno','$licenceno','$stype','$bdate','$date','$meter','$remarks','0')";
            // status=0 applied
            mysqli_query($conn, $sql6);

            $sql20 = "SELECT appointment_id FROM `tbl_appointment` WHERE
`registerno`=' $vehno' AND `scheme_id`=' $stype' AND `appointment_status`='0'";
            $val20 = mysqli_query($conn, $sql20);
            $result20 = mysqli_fetch_assoc($val20);
            $apid = $result20['appointment_id'];
            // $_SESSION['scid'] = "";

            $sql21="INSERT INTO `tbl_transaction`(`transaction_date`,`appointment_id`,
`paid_from`,`paid_to`,`transaction_type`,`paid_amount`) VALUES
('$tdate','$apid','$userid','$scid','Advance','$price')";
            mysqli_query($conn, $sql21);
            echo "<script>alert('Added
successfully');window.location='../appointmentview.php';</script>";
        } else {
            echo "<script>alert('Sorry!! You already made an appointmenton this
day');window.location='../user.php';</script>";

```

```

    }
    } else {
        echo "<script>alert('Sorry!! You already applied for this
service');window.location='../user.php';</script>";
    }
    } else {
        $data3 = mysqli_fetch_assoc($count);
        $acount = $data3['count'];
        if ($acount < $maxcount) {
            //checking already applied or not
            $sql9 = "SELECT * FROM `tbl_appointment` WHERE `registerno`='$vehno' AND
`scheme_id`='$stype' AND `appointment_status`!='-1'";
            $count1 = mysqli_query($conn, $sql9);
            if (mysqli_num_rows($count1) < 1) {
                $sql13 = "SELECT * FROM `tbl_appointment` WHERE `registerno`='$vehno' AND
`appointment_date`='$date' AND `appointment_status`!='-1' AND `appointment_status`!='3'";
                $count13 = mysqli_query($conn, $sql13);
                if (mysqli_num_rows($count13) < 1) {
                    $acount = $acount + 1;
                    //not already applied and anyone is already applied for that particular service only
upate is performed
                    $sql7 = "UPDATE `tbl_workcount` SET `count`='$acount' WHERE `date`='$date'
AND `licenceno`='$licenceno' AND `department_id`='$deptid'";
                    mysqli_query($conn, $sql7);
                    //inserting to appointment table
                    $sql8 = "INSERT INTO `tbl_appointment`(`registerno`,`licenceno`,
`scheme_id`,`appointment_date`,`appointment_date`,`odometer`,
`remarks`,`appointment_status`)
VALUES
('$vehno','$licenceno','$stype','$bdate','$date','$meter','$remarks','0')";
                    //status=0 means applied
                    //-1 means cancelled
                    //1 started
                    //2 pending
                    //3 completed
                    mysqli_query($conn, $sql8);

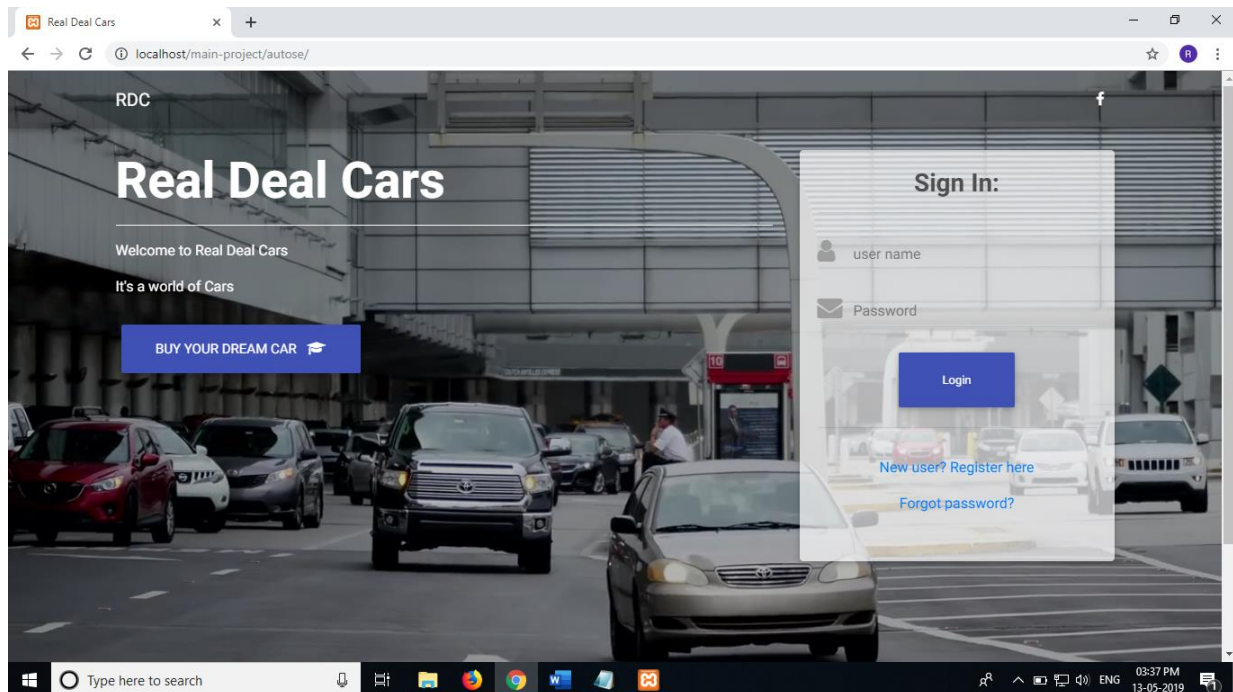
                    $sql20 = "SELECT appointment_id FROM `tbl_appointment` WHERE
`registerno`='$vehno' AND `scheme_id`='$stype' AND `appointment_status`='0'";
                    $val20 = mysqli_query($conn, $sql20);
                    $result20 = mysqli_fetch_assoc($val20);
                    $apid = $result20['appointment_id'];
                    $sql21="INSERT INTO `tbl_transaction`(`transaction_date`,`appointment_id`,
`paid_from`,`paid_to`,`transaction_type`,`paid_amount`) VALUES
('$tdate','$apid','$userid','$scid','Advance','$price')";
                    mysqli_query($conn, $sql21);
                    echo "<script>alert('Added
successfully');window.location='../appointmentview.php';</script>";
                } else {
                    echo "<script>alert('Sorry!! You already made an appointmenton this
day');window.location='../user.php';</script>";

```

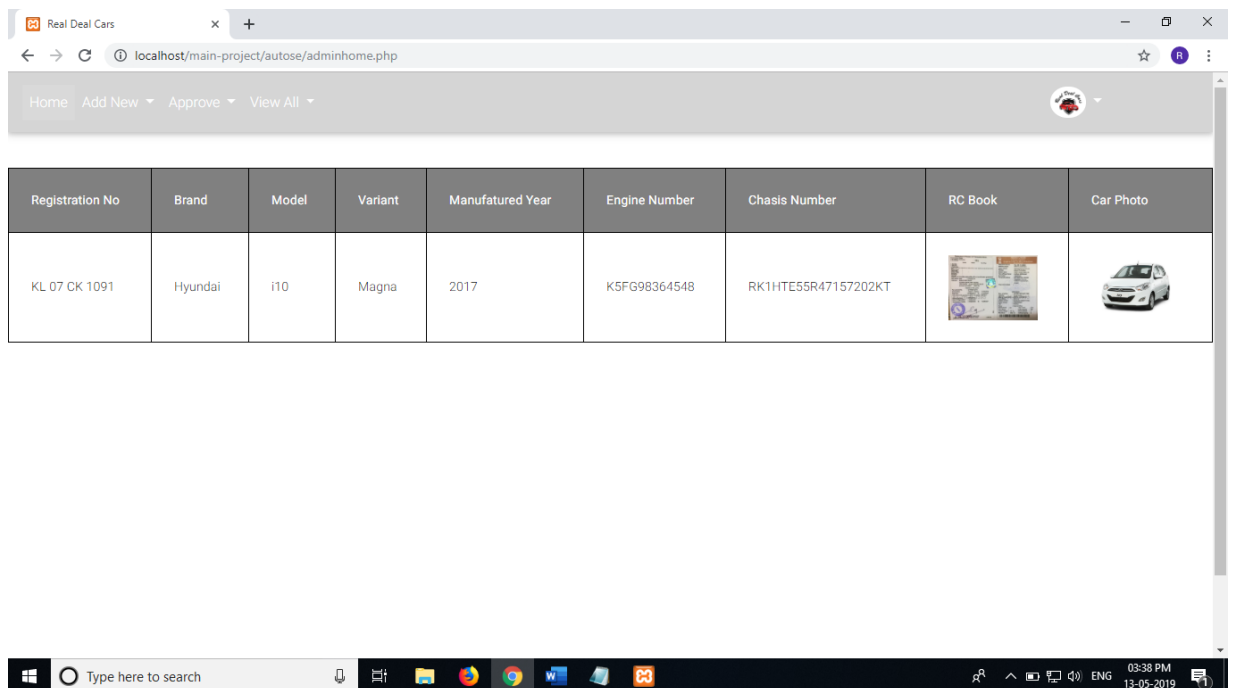
```
    }
  } else {
    echo "<script>alert('Sorry!!You already applied for this
service');window.location='../user.php';</script>";
  }
  } else {
    echo "<script>alert('Please choose another day because of
overloads');window.location='../user.php';</script>";
  }
}
$_SESSION['date'] = "";
$_SESSION['bdate'] = "";
$_SESSION['vehno'] = "";
$_SESSION['stype'] = "";
$_SESSION['meter'] = "";
$_SESSION['remarks'] = "";
$_SESSION['licenceno'] = "";
$_SESSION['price'] = "";
$_SESSION['deptid'] = "";
}
```

P1.12.2 SCREENSHOTS

Main Home (Index) page



Admin car view page





Employee view page

Real Deal Cars

localhost/main-project/autose/servicecenterhome.php

Home Services Appointment Employee Leave Management

Choose Department Search

First Name	Last Name	Department	District	Place	Contact Number	Email	Photo	
Albin	Thomas	Mechanical	Pathanamthitta	Erumeli	9656897423	realdealcarss@gmail.com		REMOVE
Sobia	Dev	Mechanical	Kollam	Kollam	9539784512	sobiad@mca.ajce.in		REMOVE

Type here to search

03:41 PM 13-05-2019


Service centre listing

Real Deal Cars

localhost/main-project/autose/servicecenterbybrand.php


Home

Find service center Select District



Indus
lic3691
9544631245
Kollam
Kollam

BOOK



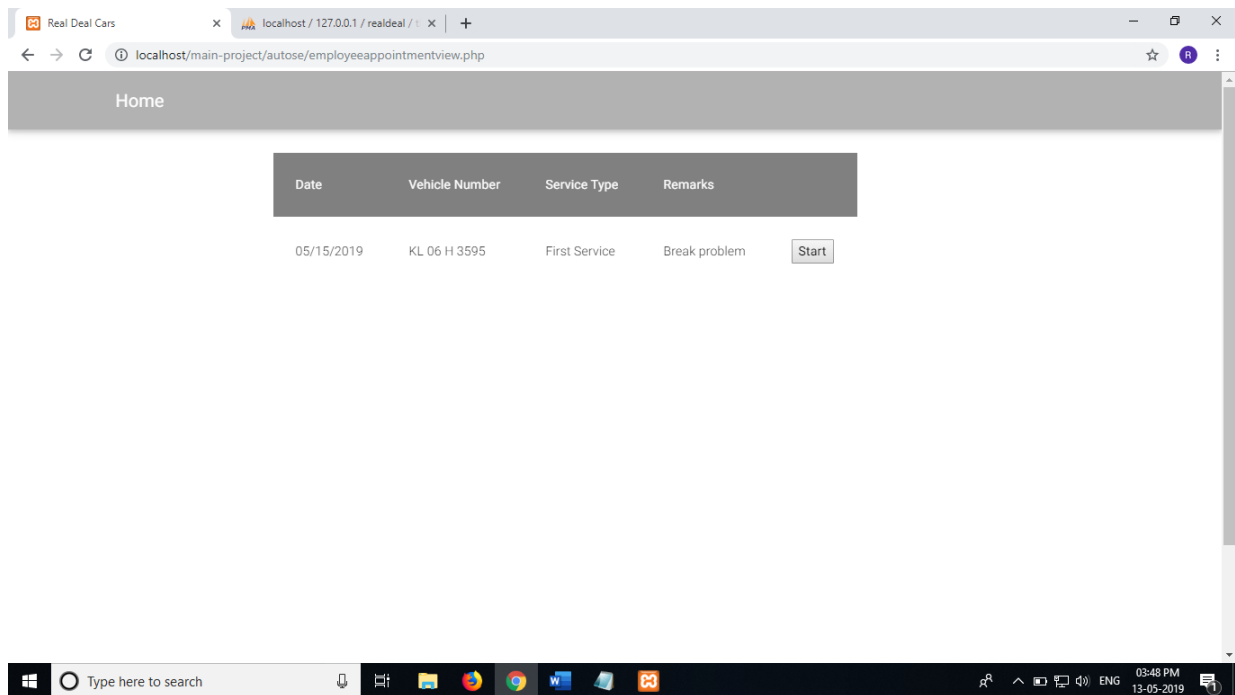
Popular
lic4123
9656332211
Alappuzha
Nedumudy

BOOK

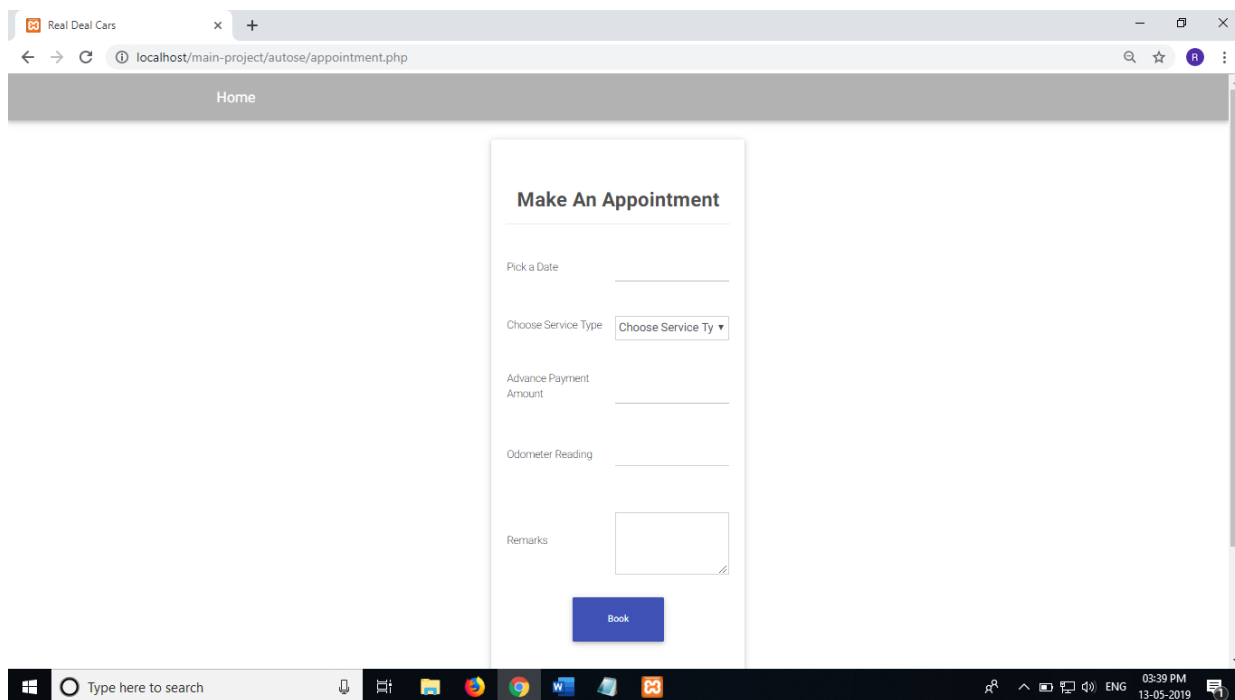
Type here to search

03:39 PM 13-05-2019

Employee appointments view page



User appointment booking page



Admin adding new models page

The screenshot shows a web browser window with the address bar displaying `localhost/main-project/autose/adminmodel.php`. The browser's address bar also shows the page title "Real Deal Cars". The page has a navigation bar at the top with links: "Home", "Add New", "Approve", and "View All". A circular profile picture is visible on the right side of the navigation bar. The main content area features a white box titled "ADD NEW MODEL". Inside this box, there is a "Select Brand" dropdown menu, a "Model Name" text input field, and a blue "Add" button. The Windows taskbar is visible at the bottom of the screen, showing the search bar, task view button, and several application icons (File Explorer, Chrome, Word, and a game). The system tray on the right shows the date and time as "03:38 PM 13-05-2019".

Real Deal Cars

localhost/main-project/autose/adminmodel.php

Home Add New Approve View All

ADD NEW MODEL

Select Brand

Model Name

Add

Type here to search

03:38 PM 13-05-2019

Part 2

Technology Framework

P2.1 ASP.NET MVC

ASP.NET MVC is an open-source software from Microsoft. Its web development framework combines the features of MVC (Model-View-Controller) architecture, the most up-to-date ideas and techniques from Agile development and the best parts of the existing ASP.NET platform. This tutorial provides a complete picture of the MVC framework and teaches you how to build an application using this tool. ASP.NET MVC is basically a web development framework from Microsoft, which combines the features of MVC (Model-View-Controller) architecture, the most up-to-date ideas and techniques from Agile development, and the best parts of the existing ASP.NET platform.

ASP.NET MVC is not something, which is built from ground zero. It is a complete alternative to traditional ASP.NET Web Forms. It is built on the top of ASP.NET, so developers enjoy almost all the ASP.NET features while building the MVC application.

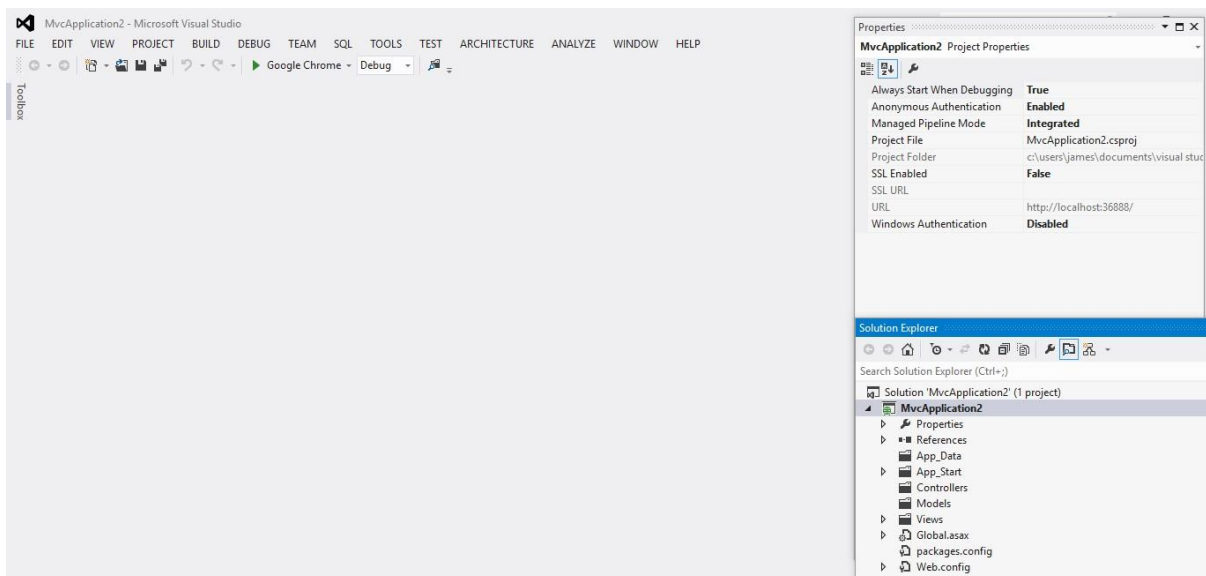
The MVC architectural pattern separates the user interface (UI) of an application into three main parts.

- **The Model** – A set of classes that describes the data you are working with as well as the business logic.
- **The View** – Defines how the application's UI will be displayed. It is a pure HTML, which decides how the UI is going to look like.
- **The Controller** – A set of classes that handles communication from the user, overall application flow, and application-specific logic.

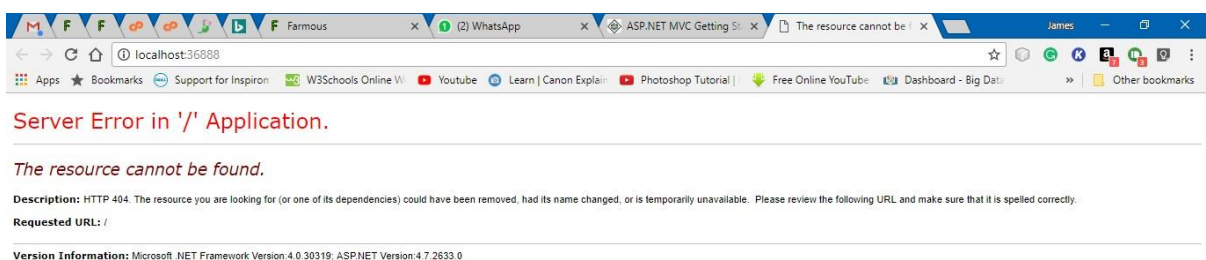
Implementation of ASP.Net MVC

- ☐ Download and install Microsoft Visual Studio 2012 and onwards
- ☐ Create an ASP.Net MVC Application. Open the Visual Studio. Click File>New > Project menu option. A new Project dialog opens.
- From the left pane, select Templates → Visual C# → Web.
- ☐ In the middle pane, select ASP.NET Web Application.

- Enter the project name, MVCApplication2, in the Name field and click ok to continue. You will see the following dialog which asks you to set the initial content for the ASP.NET project.



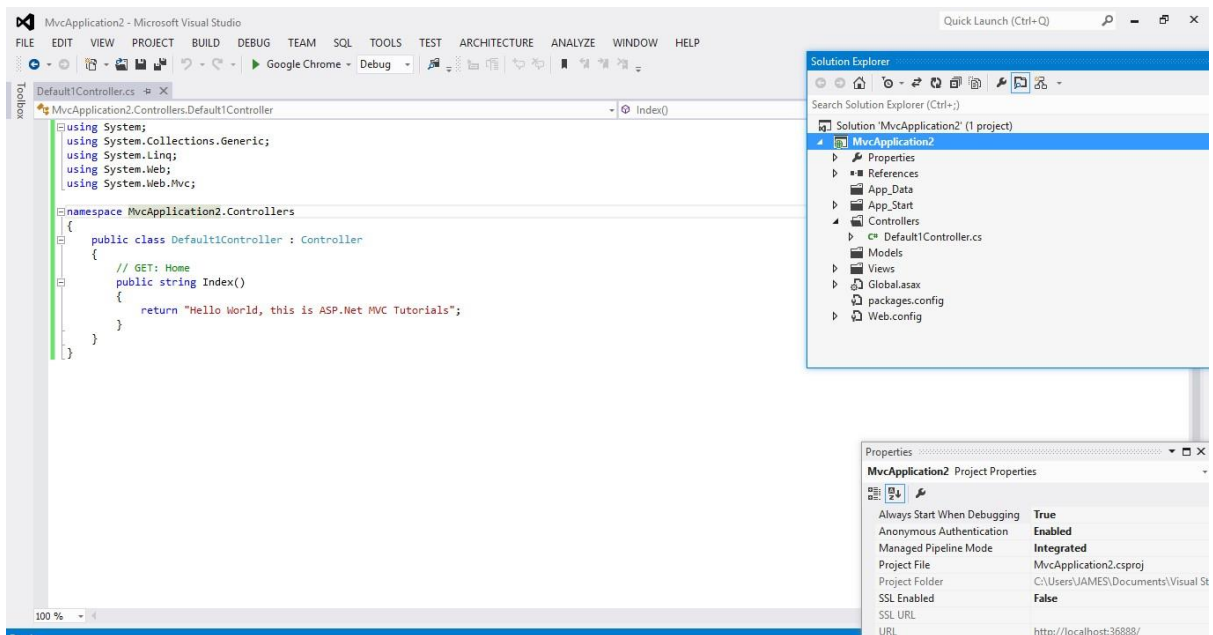
- Run this application from Debug > Start Debugging menu option and you will see a **404 Not Found Error**.



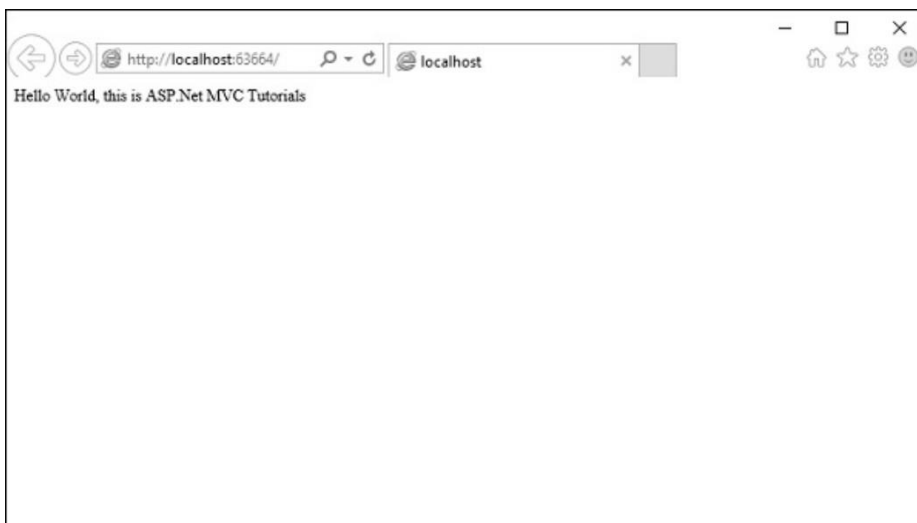
Add Controller

- To remove the 404 Not Found error, we need to add a controller, which handles all the incoming requests.
- To add a controller, right-click on the controller folder in the solution explorer and select Add > Controller.
- Select the MVC 5 Controller – Empty option and click ‘Add’ button. The Add Controller dialog will appear

- ☐ Set a name to Controller and click the Add button.
- To make this a working example, let's modify the controller class by changing the action method called **Index** using the following code.



- ☐ Run this application from Debug



P2.2 Laravel for PHP

Laravel is a free, open-source PHP web framework, created by Taylor Otwell and intended for the development of web applications following the model–view–controller (MVC) architectural pattern. It has a very rich set of functionalities, which will increase the speed of website development work.

If you know PHP well, then Laravel will make your task easier. It has a very rich set of libraries and helpers. By using Laravel, you will save a lot of time, if you are developing a website from scratch. Not only that, a website built in Laravel is secure too, as it has the ability to prevent various attacks that take place through websites.

It is very easy to install Laravel. Just follow the steps given below –

- First, download the Laravel installer using Composer:

Composer global require laravel/installer

- Once installed, the `laravel new` command will create a fresh Laravel installation in the directory you specify

Laravel new helloworld

- Via Composer Create-Project

Composer create-project laravel/laravel hello-world

- Local Development Server

If you have PHP installed locally and you would like to use PHP's built-in development server to serve your application, you may use the `serve Artisan` command. This command will start a development server at <http://localhost:8000>.

php artisan serve

Laravel is based on the **Model-View-Controller (MVC) development pattern**. MVC is a software approach that separates application logic from presentation. In practice, it permits your web pages to contain minimal scripting since the presentation is separate from the PHP scripting.

- The **Model** represents your data structures. Typically, your model classes will contain functions that help you retrieve, insert and update information in your database.
- The **View** is information that is being presented to a user. A View will normally be a web page,

but in Laravel, a view can also be a page fragment like a header or footer. It can also be an RSS page or any other type of “page”.

- The **Controller** serves as an intermediary between the Model, the View, and any other resources needed to process the HTTP request and generate a web page.

Example

1. Create a Laravel application:

```
Composer create-project laravel/laravel hello-world
```

2. Navigate to the project folder, e.g.

```
D:\laravel\hello-world
```

3. Create a controller:

```
php artisan make:controller HelloController
```

4. Register a route to HelloController's index method. Add this line or **routes/web.php**

```
Route::get('hello',HelloController@index');
```

5. Create a Blade template in the views directory:

resources/views/hello.blade.php:

```
<html>

<body>

<h1>HelloWorld</h1>

</body>

</html>
```

6. Now we tell index method to display the **hello.blade.php** template:

app/Http/Controllers/HelloController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

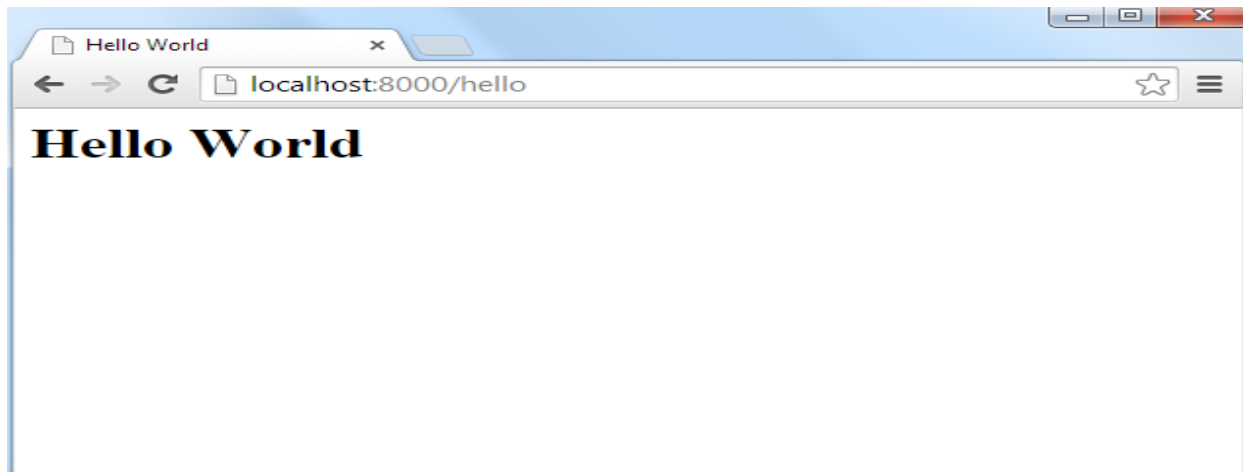
use App\Http\Requests;

class HelloController extends Controller
```

```
|  
-  
  
{  
  
    public function index ()  
  
    {  
  
        return view('hello');  
  
    }  
  
    // ... other resources are listed below the index one above
```

7. You can serve your app using the following PHP Artisan Command:

```
php artisan serve;
```



P2.3 Angular JS

Angular 6 is a JavaScript framework for building web applications and apps in JavaScript, html, and TypeScript, which is a superset of JavaScript. Angular provides built-in features for animation, http service, and materials which in turn has features such as auto-complete, navigation, toolbar, menus, etc. The code is written in TypeScript, which compiles to JavaScript and displays the same in the browser.

Step 1: Install the Angular CLI

Install the Angular CLI globally.

To install the CLI using npm, open a terminal/console window and enter the following command:

```
npm install -g @angular/cli
```

Step1: Create a workspace and initial application

```
ng new my-app
```

You develop apps in the context of an Angular workspace. A workspace contains the files for one or more projects. A project is the set of files that comprise an app, a library, or end-to-end (e2e) tests.

To create a new workspace and initial app project:

1. Run the CLI command `ng new` and provide the name `my-app`, as shown here:

The `ng new` command prompts you for information about features to include in the initial app project. Accept the defaults by pressing the Enter or Return key.

The Angular CLI installs the necessary Angular npm packages and other dependencies. This can take a few minutes.

It also creates the following workspace and starter project files:

- A new workspace, with a root folder named `my-app`
- An initial skeleton app project, also called `my-app` (in the `src` subfolder)
- An end-to-end test project (in the `e2e` subfolder)
- Related configuration files
- The initial app project contains a simple Welcome app, ready to run.

3. Serve the Application

Angular includes a server, so that you can easily build and serve your app locally.

Go to the workspace folder (my-app).

Launch the server by using the CLI command `ng serve`, with the `--open` option.

```
cd my-app  
ng serve --open
```

The `ng serve` command launches the server, watches your files, and rebuilds the app as you make changes

to those files.

The `--open` (or just `-o`) option automatically opens your browser to `http://localhost:4200/`.

Your app greets you with a message:

Welcome to my-app!



Step 4: Edit your first Angular component

Components are the fundamental building blocks of Angular applications. They display data on the screen, listen for user input, and take action based on that input. As part of the initial app, the CLI created the first Angular component for you. It is the root component, and it is named `app-root`.

Open `./src/app/app.component.ts`.

Change the title property from `'my-app'` to `'My First Angular App'`.

`src/app/app.component.ts`

```
@Component({
```

```
    selector: 'app-root',  
    templateUrl: './app.component.html',  
    styleUrls: ['./app.component.css']  
  })  
  export class AppComponent {  
    title = 'My First Angular App!';  
  }  
}
```

The browser reloads automatically with the revised title. That's nice, but it could look better.

Open `./src/app/app.component.css` and give the component some style.

`src/app/app.component.css`

```
h1 {  
  color: #369;  
  font-family: Arial, Helvetica, sans-serif;  
  font-size: 250%;  
}
```

Output of Getting Started app

Welcome to My First Angular App!



P2.4 Android

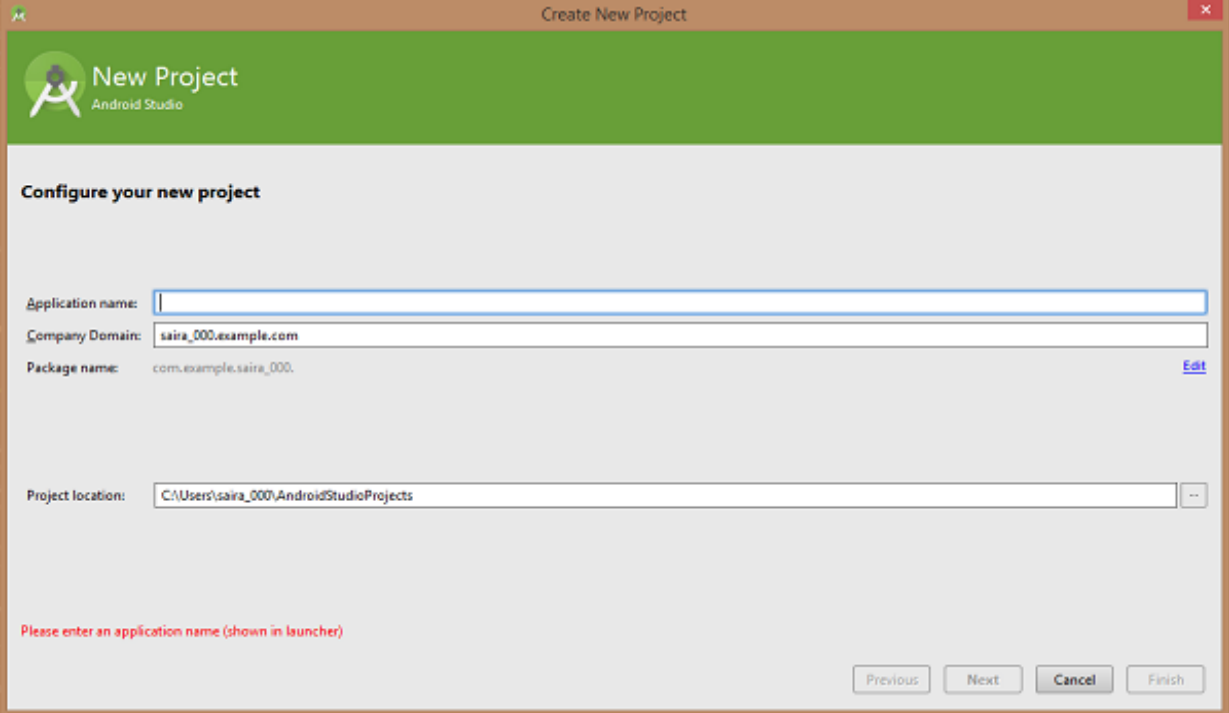
Android is a software package and linux based operating system for mobile devices such as tablet computers and smartphones. It is developed by Google and later the OHA (Open Handset Alliance). Java language is mainly used to write the android code even though other languages can be used. The goal of android project is to create a successful real-world product that improves the mobile experience for end users. There are many code names of android such as Lollipop, KitKat, Jelly Bean, Ice cream Sandwich, Froyo, Ecliar, Donut etc.

Creating Android Application

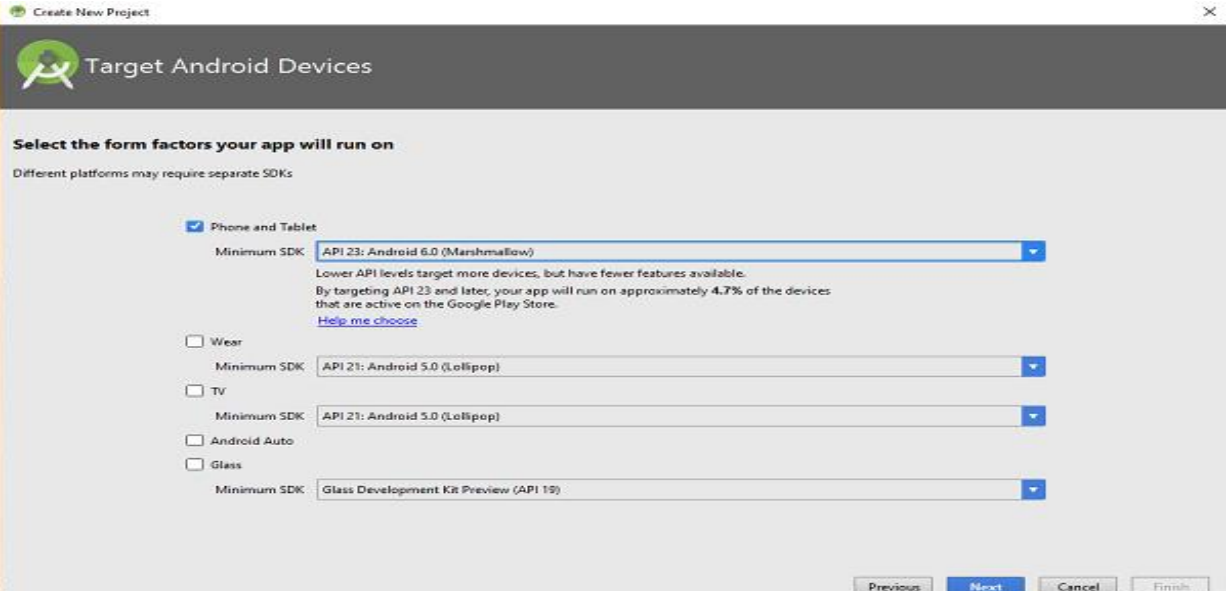
The first step is to create a simple Android Application using Android studio. When you click on Android studio icon, it will show screen as shown below



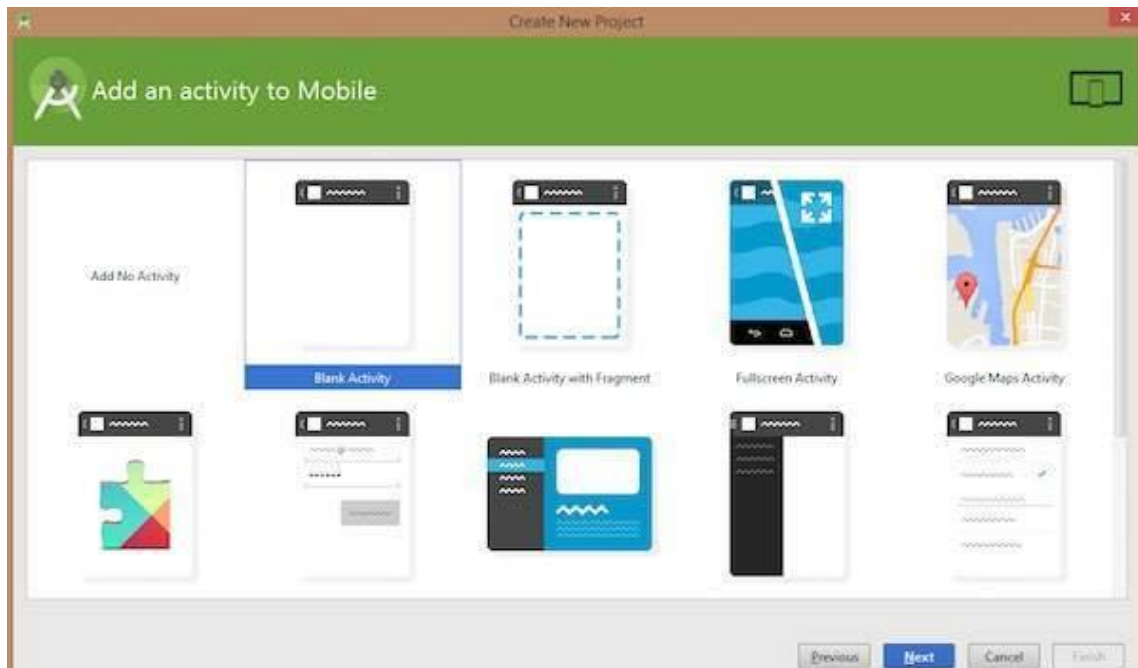
You can start your application development by calling start a new android studio project. in a new installation frame should ask Application name, package information and location of the project.



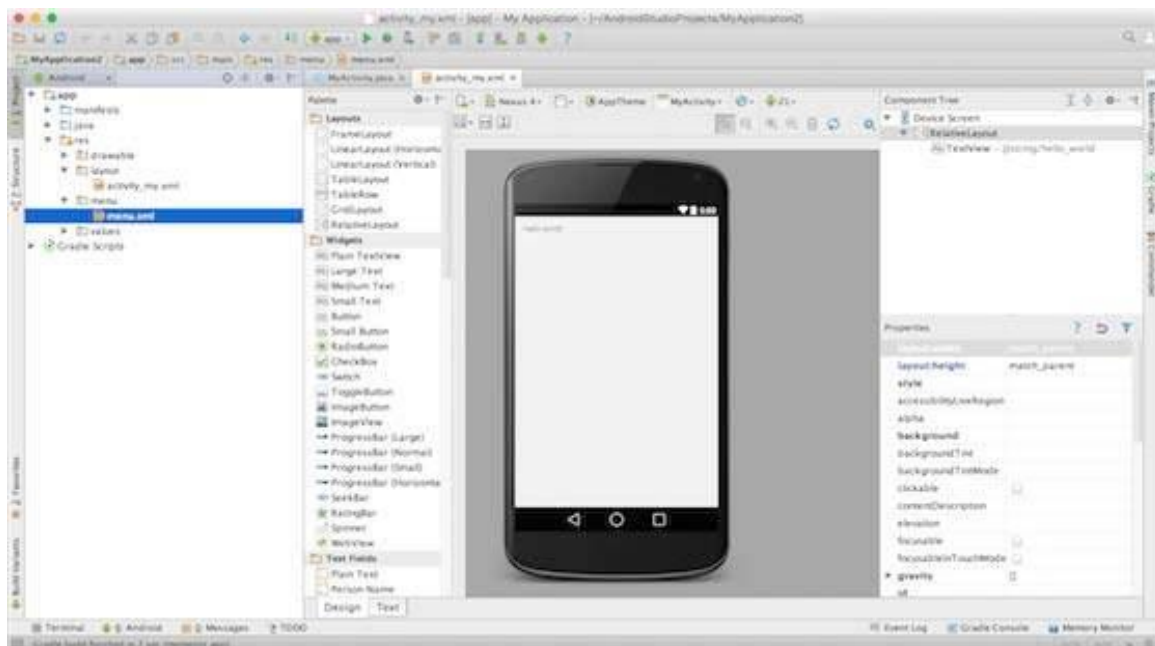
After entered application name, it going to be called select the form factors your application runs on, here need to specify Minimum SDK, in our tutorial, I have declared as API23: Android 6.0(Mashmallow)



The next level of installation should contain selecting the activity to mobile, it specifies the default layout for Applications.

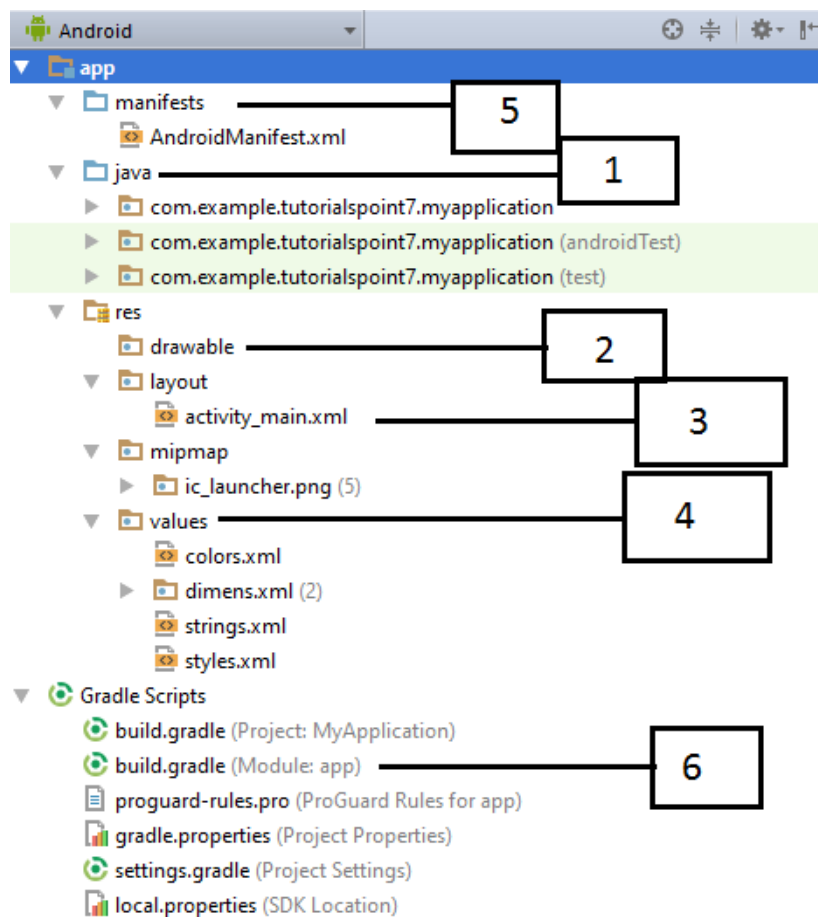


At the final stage it going to be open development tool to write the application code.



Anatomy of Android Application

Before you run your app, you should be aware of a few directories and files in the Android project.



Following section will give a brief overview of the important application files.

The Main Activity File

The main activity code is a Java file **MainActivity.java**. This is the actual application file which ultimately gets converted to a Dalvik executable and runs your application. Following is the default code generated by the application wizard for *Hello World!* application –

```
package com.example. helloworld;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

Here, *R.layout.activity_main* refers to the *activity_main.xml* file located in the *res/layout* folder. The *onCreate()* method is one of many methods that are figured when an activity is loaded.

The Manifest File

Whatever component you develop as a part of your application, you must declare all its components in a *manifest.xml* which resides at the root of the application project directory. This file works as an interface between Android OS and your application, so if you do not declare your component in this file, then it will not be considered by the OS. For example, a default manifest file will look like as following file –

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.tutorialspoint7.myapplication">
<application
android:allowBackup="true"
android:icon="@mipmap/ic_launcher"
android:label="@string/app_name"
android:supportsRtl="true"
android:theme="@style/AppTheme">
<activity android:name=".MainActivity">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>
</manifest>
```

Here `<application>...</application>` tags enclosed the components related to the application. Attribute *android:icon* will point to the application icon available under *res/drawable-hdpi*. The application uses the image named *ic_launcher.png* located in the *drawable* folders

The `<activity>` tag is used to specify an activity and *android:name* attribute specifies the fully qualified class name of the *Activity* subclass and the *android:label* attributes specifies a string to use as the label for the activity. You can specify multiple activities using `<activity>` tags.

The **action** for the intent filter is named *android.intent.action.MAIN* to indicate that this activity

serves as the entry point for the application. The **category** for the intent-filter is named *android.intent.category.LAUNCHER* to indicate that the application can be launched from the device's launcher icon.

The *@string* refers to the *strings.xml* file explained below. Hence, *@string/app_name* refers to the *app_name* string defined in the *strings.xml* file, which is "HelloWorld". Similar way, other strings get populated in the application.

Following is the list of tags which you will use in your manifest file to specify different Android application components

- `<activity>` elements for activities
- `<service>` elements for services
- `<receiver>` elements for broadcast receivers
- `<provider>` elements for content providers

The Strings File

The **strings.xml** file is located in the *res/values* folder and it contains all the text that your application uses. For example, the names of buttons, labels, default text, and similar types of strings go into this file. This file is responsible for their textual content. For example, a default strings file will look like as following file –

```
<resources>
    <string name="app_name">HelloWorld</string>
    <string name="hello_world">Hello world!</string>
    <string name="menu_settings">Settings</string>
    <string name="title_activity_main">MainActivity</string>
</resources>
```

The Layout File

The **activity_main.xml** is a layout file available in *res/layout* directory, that is referenced by your application when building its interface. You will modify this file very frequently to change the layout of your application. For your "Hello World!" application, this file will have following content related to default layout


```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
```



```
android:layout_width="match_parent"
android:layout_height="match_parent" >
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="true"
    android:padding="@dimen/padding_medium"
    android:text="@string/hello_world"
    tools:context=".MainActivity" />
</RelativeLayout>
```

This is an example of simple *RelativeLayout* which we will study in a separate chapter. The *TextView* is an Android control used to build the GUI and it have various attributes like *android:layout_width*, *android:layout_height* etc which are being used to set its width and height etc.. The *@string* refers to the strings.xml file located in the res/values folder. Hence, *@string/hello_world* refers to the hello string defined in the strings.xml file, which is "Hello World!".

Running the Application

Let's try to run our **Hello World!** application we just created. I assume you had created your **AVD** while doing environment set-up. To run the app from Android studio, open one of your project's activity files and click Run  icon from the tool bar. Android studio installs the app on your AVD and starts it and if everything is fine with your set-up and application, it will display following Emulator window

P2.5 Server Hardening

Server Hardening is the process of enhancing server security through a variety of means which results in a much more secure server operating environment. This is due to the advanced security measures that are put in place during the server hardening process.

The term "hardening," in the general sense, implies taking a soft surface or material and making changes to it which result in that surface becoming stronger and more resistant to damage. That is exactly how **server hardening** impacts server security. Hardened servers are more resistant to security issues than non-hardened servers. * In a time when nearly every computing resource is online and susceptible to attack, server hardening is a near absolute must to perform on your servers. * The Internet has vastly altered the complexion of the server hardening industry over the last decade. Much of the applications and system software that is now developed is intended for use on the Internet, and for connections to the Internet. * Many servers online today are attacked thousands of times per hour, tens and sometimes hundreds of thousands of times each and every day. The best defense against such attacks is to ensure that server hardening is a well-established practice within your organization or to outsource this task to an experienced & established server hardening agency.

Server Hardening, probably one of the most important tasks to be handled on your servers, becomes more understandable when you realize all the risks involved. The default config of most operating systems is not designed with security as the primary focus. Instead, default setups focus more on usability, communications and functionality. To protect your servers, you must establish solid and sophisticated server hardening policies for all servers

in your organization. Developing a server hardening checklist would likely be a great first step in increasing your server and network security. Make sure that your checklist includes minimum security practices that you expect of your staff. If you go with a consultant you can provide them with your server hardening checklist to use as a baseline.

Server Hardening Tips & Tricks: Every server security conscious organization will have their own methods for maintaining adequate system and network security. Often you will find that server hardening consultants can bring your security efforts up a notch with their specialized expertise. Some common server hardening tips & tricks include: - Use Data Encryption for your Communications - Avoid using insecure protocols that send your information or passwords in plain text. - Minimize unnecessary software on your servers. Disable Unwanted SUID and SGID Binaries - Keep your operating system up to date, especially security patches. Using security

extensions are a plus. - When using Linux, SELinux should be considered. Linux server hardening is a primary focus for the web hosting industry, however in web hosting SELinux is probably not a good option as it often causes issues when the server is used for web hosting purposes. - User Accounts should have very strong passwords - Change passwords on a regular basis and do not reuse them - Lock accounts after too many login failures. Often these login failures are illegitimate attempts to gain access to your system. - Do not permit empty passwords. - SSH Hardening --- Change the port from default to a non-standard one --- Disable direct root logins. Switch to root from a lower level account only when necessary. - Unnecessary services should be disabled. Disable all instances of IRC - BitchX, bnc, eggdrop, generic-sniffers, guardservices, ircd, psyBNC, ptlink. - Securing /tmp /var/tmp /dev/shm