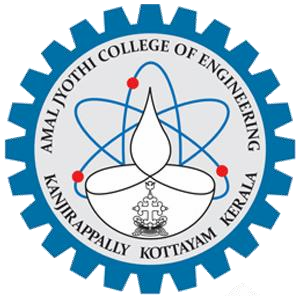
**AUTOSHOP**

*Project Report Submitted by*

**JISHNUMON PB**

**Reg. No: LAJC16MCA042**

*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]

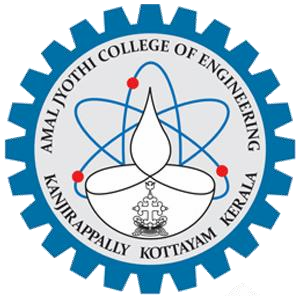
## 2017-2019

**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 **“AUTOSHOP”** is a bonafide record of the work done by **JISHNUMON PB LAJC16MCA042,** during the academic year **2017-2019** 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** | **Mr. Binumon Joseph** | **Mr. Jinson Devis** |
| Head of the Department | Project Coordinator | Project Supervisor |

**Expert from dept. of Computer Science and Engineering**

Amal Jyothi College of Engineering

**External Expert (Academic) External Expert (Industry)**

**External Expert appointed by the university**

**DECLARATION**

I hereby declare that the project report **“AUTOSHOP”** 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 2017-2019.

**Date................ JISHNUMON PB**

**KANJIRAPPALLY Reg. No: LAJC16MCA042**

**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 **Mr. Jinson Devis** 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.

**JISHNUMON PB**

**ABSTRACT**

The project entitled ‘**Autoshop’** is an online Automobile shopping system that allows web users to purchase used vehicles online without visiting any physical location.

**‘Autoshop’** allows the user to purchase Automobile online and Bid vehicles in Auction. The Site Administrator updates the information about new Automobiles concurrently. Only registered customers can purchase Automobiles bid from Autoshop. The user must register in the site to access their accounts, after login they can buy used vehicles by bidding Automobiles which are available for auction. The user can bid a vehicle with a high rate than the latest Bid Rate, before the date expires. After date expires, the site sends the Confirmation letter to the user who bid the car with highest rate through email-id which they specified.

This system allows the users to search items category wise, then Brand wise and Model wise. Thus this system provides all the basic functionalities to a user who would like to purchase used cars in online.

|  |  |  |
| --- | --- | --- |
| **Sl. No** | **Topic** | **Page No** |
| **Part 1** | **Technology Frameworks** | 1 |
| **P1.1** | **ASP.NET MVC** | 2-4 |
| **P1.2** | **Laravel for PHP** | 5-7 |
| **P1.3** | **Angular JS** | 08-10 |
| **P1.4** | **Android** | 11-17 |
| **P1.5** | **Server Hardening** | 18-19 |
| **Part 2** | **Project Documentation** | 20 |
| **P2.1** | **Introduction** | 21 |
| **P2.1.1** | **Project Overview** | 21 |
| **P2.1.2** | **Project Specification** | 25 |
| **P2.2** | **System Study** | 26 |
| **P2.2.1** | **Introduction** | 26 |
| **P2.2.2** | **Proposed system** | 27 |
| **P2.3** | **Requirement Analysis** | 28 |
| **P2.3.1** | **Feasibility Study** | 28 |
| **P2.3.1.1** | **Economical Feasibility** | 28 |
| **P2.3.1.2** | **Technical Feasibility** | 29 |
| **P2.3.1.3** | **Operational Feasibility** | 29 |
| **P2.4** | **Requirement Modelling** | 30 |
| **P2.4.1** | **Class Diagram** | 32 |
| **P2.4.2** | **Object Diagram** | 33 |
| **P2.4.3** | **Deployment Diagram** | 34 |
| **P2.4.4** | **Use case Diagram** | 35 |
| **P2.4.5** | **Sequence Diagram** | 38 |
| **P2.4.6** | **Collaboration Diagram** | 37 |
| **P2.4.7** | **Statechart Diagram** | 37 |
| **P2.4.8** | **Activity Diagram** | 39 |
| **P2.5** | **System Specification** | 40 |
| **P2.5.1** | **Hardware Specification** | 40 |
| **P2.5.2** | **Software Specification** | 41 |
| **P2.6** | **Software Description** | 41 |
| **P2.6.1** | **PHP** | 41 |
| **P2.6.2** | **MySQL** | 41 |
| **P2.7** | **System Design** | 43 |
| **P2.7.1** | **Architectural Design** | 44 |
| **P2.7.2** | **Module Design** | 45 |
| **P2.7.3** | **Data Base Design** | 46 |
| **P2.8** | **System Testing** | 54 |
| **P2.8.1** | **Introduction** | 54 |
| **P2.8.2** | **Test Plan** | 55 |
| **P2.8.2.1** | **Unit Testing** | 56 |
| **P2.8.2.2** | **Integration Testing** | 56 |
| **P2.8.2.3** | **Validation Testing** | 57 |
| **P2.8.2.4** | **User Acceptance Testing** | 57 |
| **P2.9** | **Implementation** | 60 |
| **P2.9.1** | **Implementation Procedure** | 61 |
| **P2.9.2** | **User Training** | 61 |
| **P2.9.3** | **Operational Document** | 62 |
| **P2.9.4** | **System Maintenance** | 62 |
| **P2.10** | **Conclusion & Future Enhancements** | 62 |
| **P2.10.1** | **Future Enhancement** | 62 |
| **P2.10.2** | **Conclusion** | 63 |
| **P2.11** | **Bibliography** | 63 |
| **P2.12** | **Appendix** | 64 |
| **P2.12.1** | **Sample Code** | 64 |
| **P2.12.2** | **Screen Shots** | 67 |
|  |  |  |

**LIST OF ABBREVIATIONS**

IDE - Integrated Development Environment HTML - Hyper Text Markup Language.

CSS - Cascading Style Sheet

SQL - Structured Query Language DFD - Data Flow Diagram

GCP - Google Cloud Platform AWS - Amazon Web Services. C2 - Elastic Compute Cloud S3 - Simple Storage Systems

IAM - Identity Access Management

# Part 1 Technology Frameworks

## P1.1 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.

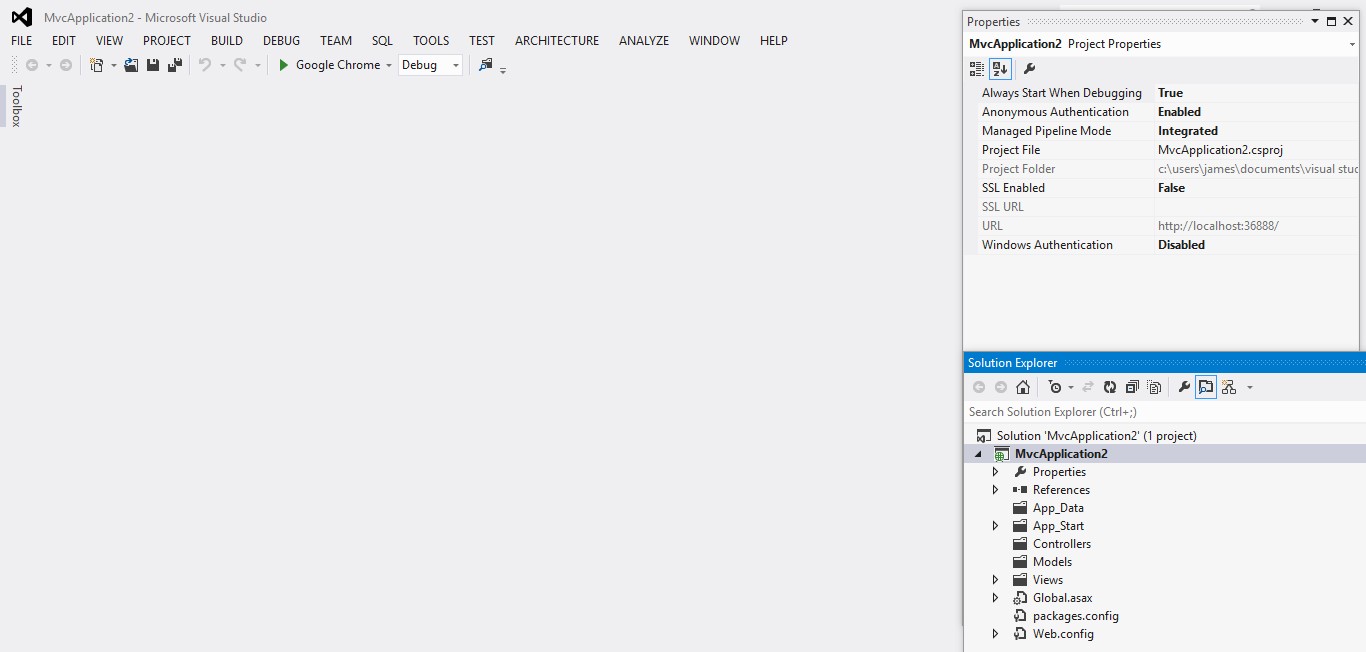
ASP.NET MVC is not something, which is built from ground zero. It is a complete alternative to traditional ASP.NET Web Forms. It is built on the top of ASP.NET, so developers enjoy almost all the ASP.NET features while building the MVC application.

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

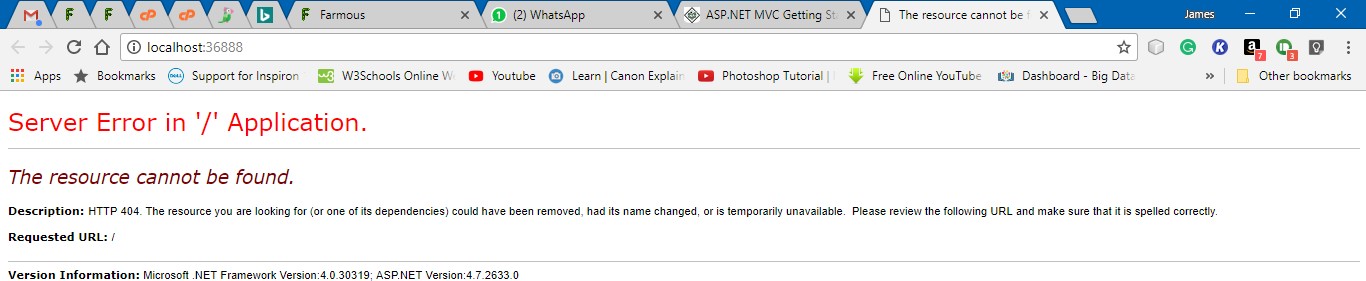
* **The Model** − A set of classes that describes the data you are working with as well as the business logic.
* **The View** − Defines how the application’s UI will be displayed. It is a pure HTML, which decides how the UI is going to look like.
* **The Controller** − A set of classes that handles communication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

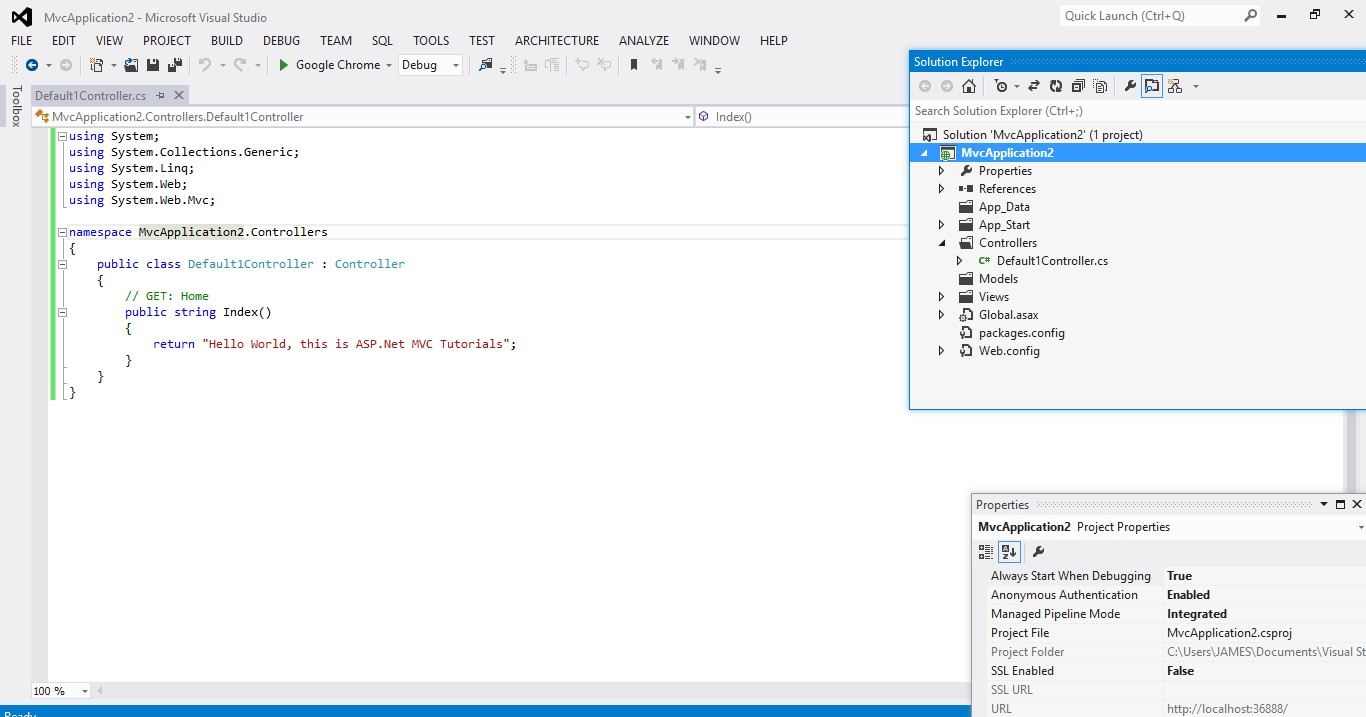


* Run this application from Debug > Start Debugging menu option and you will see a **404 Not Found** Error.

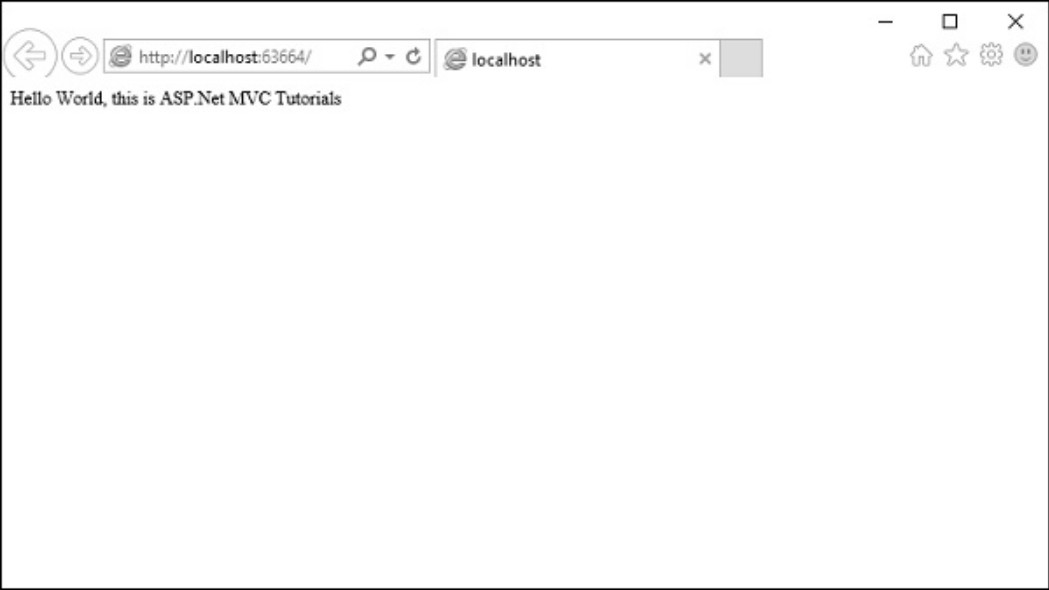


##### Add Controller

* 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



**P1.2 Laravel for PHP**

**Laravel** is a free, open-sourcePHP 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

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

D:\laravel\hello-world

1. Create a controller:

php artisan make:controller HelloController

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

Route::get(‘hello’,HelloController@index’);

1. Create a Blade template in the views directory:

**resources/views/hello.blade.php:**

<html>

<body>

<h1>HelloWorld</h1>

</body>

</html>

1. 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 ()

{

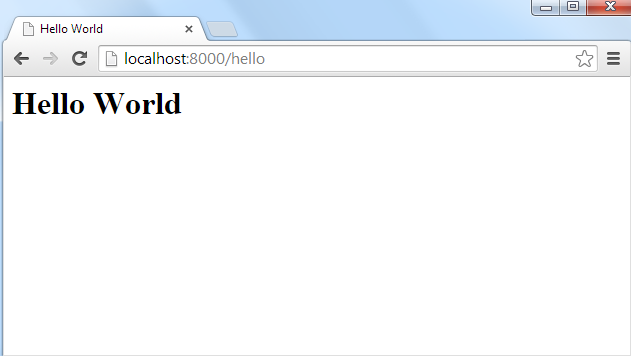
return view('hello');

}

// ... other resources are listed below the index one above

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

php artisan serve;



**P1.3 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:



**Step2: Create a workspace and initial application** 

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.

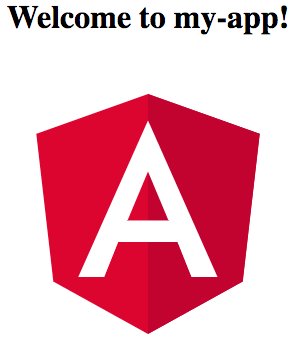


The ng serve command launches the server, watches your files, and rebuilds the app as you make changes

to those files.

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

Your app greets you with a message:



**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



**P1.4 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.

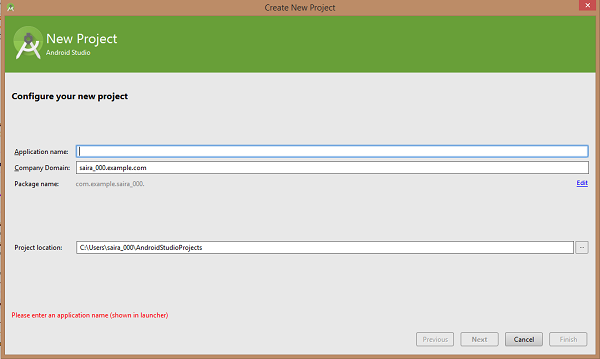
**Creating Android Application**

The first step is to create a simple Android Application using Android studio. When you click

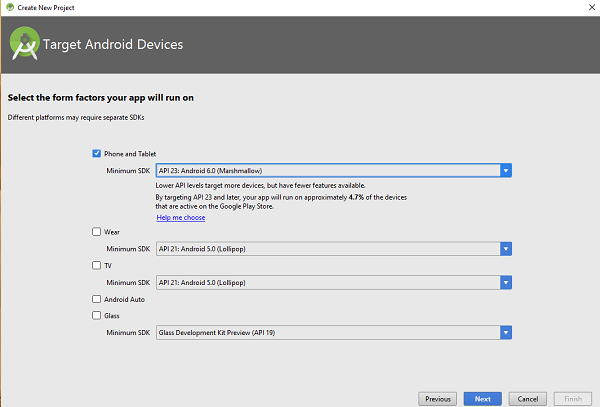
on Android studio icon, it will show screen as shown below



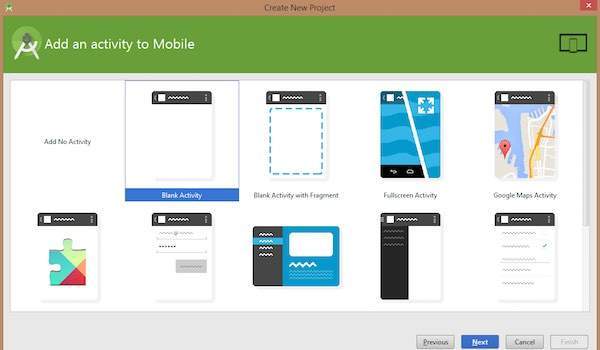
You can start your application development by calling start a new android studio project. in a new installation frame should ask Application name, package information and location of the project.



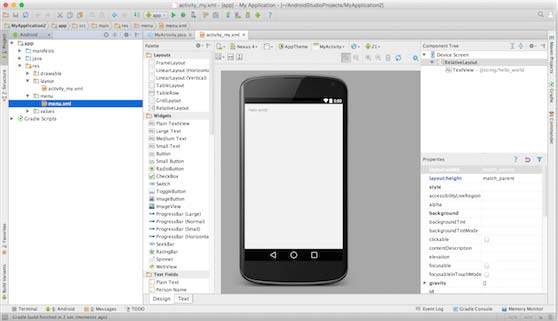
After entered application name, it going to be called select the form factors your application runs on, here need to specify Minimum SDK, in our tutorial, I have declared as API23: Android 6.0(Mashmallow)



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

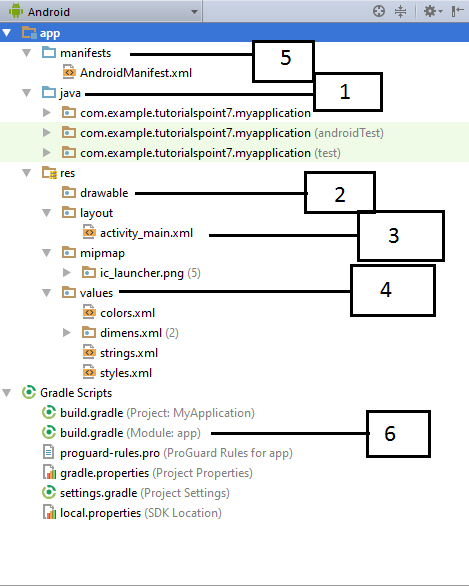


At the final stage it going to be open development tool to write the application code.



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 Eclipse Run Icon 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

**P7.4 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 is 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

# Part 2

**Project Documentation**

## P2.1 INTRODUCTION

**P2.1.1 Project Overview**

The project entitled ‘**Autoshop’** is an online Automobile shopping system that allows web users to purchase used vehicles online without visiting any physical location.  **‘Autoshop’** allows the user to purchase Automobile online and Bid vehicles in Auction. The Site Administrator updates the information about new Automobiles concurrently. Only registered customers can purchase Automobiles bid from Autoshop. The user must register in the site to access their accounts, after login they can buy used vehicles by bidding Automobiles which are available for auction. The user can bid a car with a high rate than the latest Bid Rate, before the date expires. After date expires, the site sends the Confirmation letter to the user who bid the vehicle with highest rate through email-id which they specified.

This system allows the users to search items category wise, then Brand wise and Model wise. Thus this system provides all the basic functionalities to a user who would like to purchase used vehicles in online.

1. To what extend the system is proposed for?

The Autoshop system was developed by easy to sale by vehicles in online format.

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

The system is developed by public use. It is easy to sale vehicles in online format.

1. List the Modules included in your System?

**1. User Module**

The main processes in this module are given below:

* + - Edit Details
    - Purchase
    - Bidding
    - Product Review
    - View Updations about new  product

**2. Auction**

* Select Vehicles
* Add to Auction
* Updating Details
* Setting Initial Rate
* Sending Confirmations

**3.Admin Module**

* + - Edit Profile
    - Confirm Employee account
    - Manage Customers
    - Checking Products
    - Add Vehicle to Auction
    - Add delivery details

**4. Workshop**

* + - Customer register
    - Viewing live status
    - Delivery details
    - Adding extra fitting

1. Identify the users in your project?

* User Module
* Auction
* Administrator Module
* Workshop

1. Who owns the system?

System Admin

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

Public use

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

a) Search for OLX site.

b) Communication with an automobile workshop.

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

1.What are the different types of users in your system?

**Ans**:- Admin Module, User Module , Auction , Purchasing , Workshop

2.What are the different types of vehicles in your system?

Ans :-Any types of vehicles are sale with auction.

3.How the availability of vehicles in the showroom are informed to the customers?

Ans:-Admin can be updated new used vehicles in Atoshop site. The user can search and buy the vehicles.

4.How to add advertisement in your system?

Ans:-Either by advertising or by direct contacting the pre-registered. Customers.

5.How to provide warranty in your system?

Ans :-The warranty is important factor. The warranty is provided by 6 months form any complaint .

6.what are the different service providers in your system?

Ans:-My system can provide any types of vehicle company are provided the services.

7.Which types of payment systems are allocated form your system?

Ans:-My system can provide online payment is used(debit card, phonePe, google pay etc.

8.How to manage extra fitting add different vehicles in your system?

Ans :-The adding extra fitting facility are provided my system. The system workshop module can manage extra fittings fron the system.

9.Spot assistant providence is possible for your system?

Ans :-yes. Spot assistant providence is provided from the system.

10.How to communicate with user and site?

Ans :-The user and site can be communicate with online communication (fb, whatsapp) and mobile contact, e-mail etc.

## 

## P2.1.2 Project Specification

This is a website in which we will get the services of several service providers. It will be a simple platform for users to access services for their huge needs. It provides the services are the vehicle service appointment, buy spare, buy used vehice….etc

The system includes 2 modules. They are:

* **Administrator**: The administrator of the company is allowed to access all the services in the system. And approve vehicle services appointment, add all products
* **Registered User Module :**customers can book bid for vehicles online and also customer can Buy second hand vehicles . And chat to the admin

The main features of the project

* Booking auction vehicle Facility
* Booking approval via message notification
* Searching and buy vehicles
* Online demo payment facility
* Buy Used Vehicle Facility
* Compare Vehicles
* Chatting to the admin

## P2.2 SYSTEM STUDY

### P8.2.1 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

### P8.2.2 PROPOSED SYSTEM

To overcome limitations of existing system, we can introduce a site for Autoshop. In the proposed system. It provides the services to the users who are searching for service like auction for vehicle service, buy used vehicles and also provided vehicles compare facility and chatting facility . This is a website in which we will get the several services.

The existing system has several limitations and more difficulties to work well. The proposed system provide 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 main features include:

* Registered users can book the service of system
* Provide facility of giving feedback admin
* Easy access information

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.

## P2.3 REQUIREMENT ANALYSIS

### P8.3.1 Feasibility Study

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spend on it. Feasibility study lets the developer foresee the future of the project and the usefulness. A feasibility study of a system proposal is according to its workability, which is the impact on the organization, ability to meet their user needs and effective use of resources. Thus, when a new application is proposed it normally goes through a feasibility study before it is approved for development.

The document provides the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as Technical, Economic and Operational feasibilities. The following are its features:

### P8.3.1.1 Economical Feasibility

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.

The following are some of the important financial questions asked during preliminary investigation:

* The costs conduct a full system investigation.
* The cost of the hardware and software.
* The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project , there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

E-workshop will be a simple platform for users to access services for their huge needs. It is completely free. Using this system large number people can solve their problems with free of cost.

### P8.3.1.2 Technical Feasibility

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs and procedures. Having identified an outline system, the investigation must go on to suggest the type of equipment, required method developing the system, of running the system once it has been designed.

Technical issues raised during the investigation are:

* Does the existing technology sufficient for the suggested one?
* Can the system expand if developed?

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 in server in back end, the project is technically feasible for development.

### P8.3.1.3 Behavioural Feasibility

This includes the following questions:

* Is there sufficient support for the users?
* Will the proposed system cause harm?

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.

At your service, GUI is simple so that users can easily use it. E-workshop is simple enough so that no training is needed.

## P2.4 Requirement Modeling

### P2.4.1 UML Use Case Diagram

A use case diagram is a graphic depiction of the interactions among the elements of a system. A [use case](http://searchsoftwarequality.techtarget.com/definition/use-case) is a methodology used in system analysis to identify, clarify, and organize system requirements. In this context, the term "system" refers to something being developed or operated, such as a mail-order product sales and service [Web site](http://searchsoa.techtarget.com/definition/Web-site). Use case diagrams are employed in [UML](http://searchsoftwarequality.techtarget.com/definition/Unified-Modeling-Language) (Unified Modeling Language), a standard notation for the modeling of real-world objects and systems.

System objectives can include planning overall requirements, validating a [hardware](http://searchcio-midmarket.techtarget.com/definition/hardware) design, testing and [debugging](http://searchsoftwarequality.techtarget.com/definition/debugging) a [software](http://searchsoa.techtarget.com/definition/software) product under development, creating an online help reference, or performing a consumer-service-oriented task. For example, use cases in a product sales environment would include item ordering, catalog updating, payment processing, and customer relations. A use case diagram contains four components.

* The boundary, which defines the system of interest in relation to the world around it.
* The actors, usually individuals involved with the system defined according to their roles.
* The use cases, which are the specific roles played by the actors within and around the system.
* The relationships between and among the actors and the use cases.

**UML Diagrams**

### P2.4.2 UML Sequence Diagram

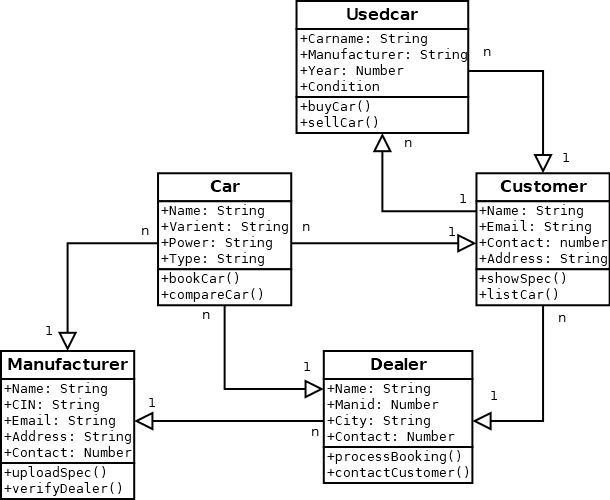
A sequence diagram is a[n interaction diagram](https://en.wikipedia.org/wiki/Interaction_diagram) that shows how objects operate with one another and in what order. It is a construct of a [message sequence chart](https://en.wikipedia.org/wiki/Message_sequence_chart).

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

A sequence diagram shows, as parallel vertical lines (*lifelines*), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner

**Structural Diagrams**

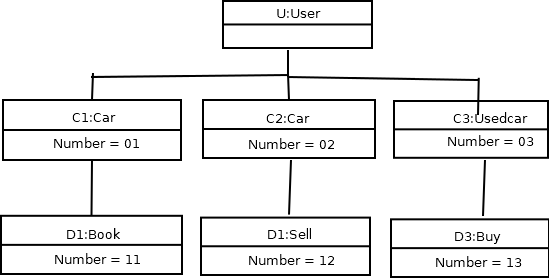
**Class Diagram**



**vehicles**

**Used Vehicle**

**Object Diagram**

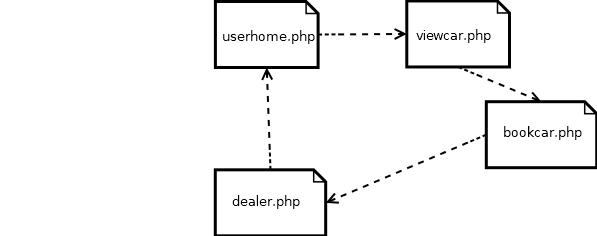
****

V3:Used Vehicle

V2:Vehicle

V1:Vehicle

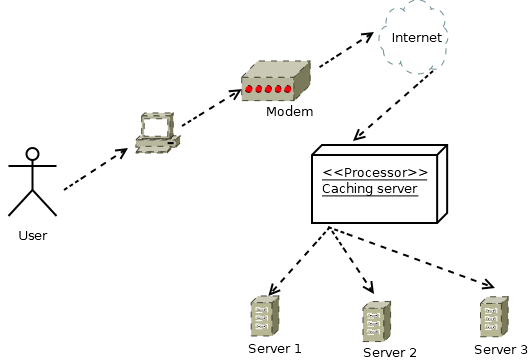
**Component Diagram**



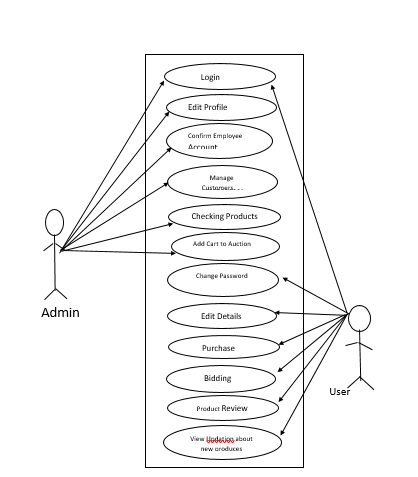
Book vehicle.php

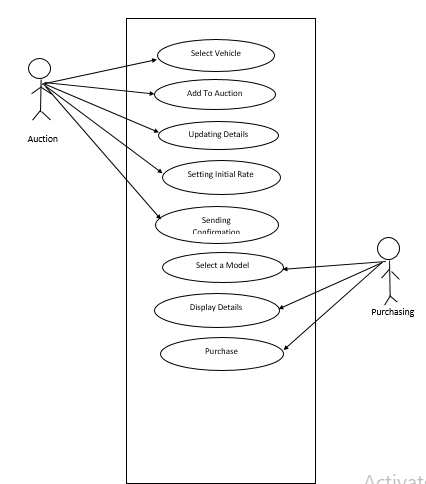
View vehicle.php

**Deployment Diagram**

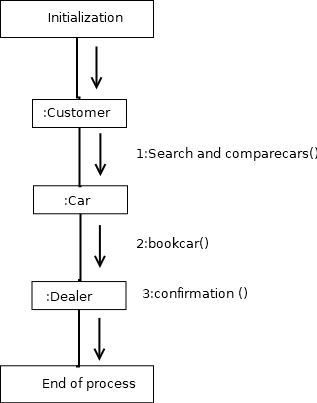


**Behavioral Diagrams**

**Use case diagram**



**Collaboration Diagram**

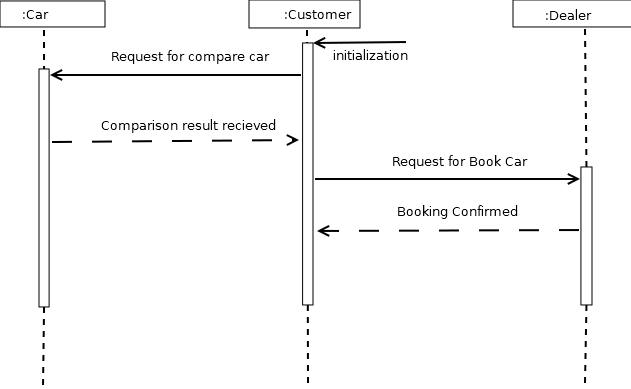
****

Compare vehicles()

:vehiclee

Book vehicle()

**Sequence Diagram**

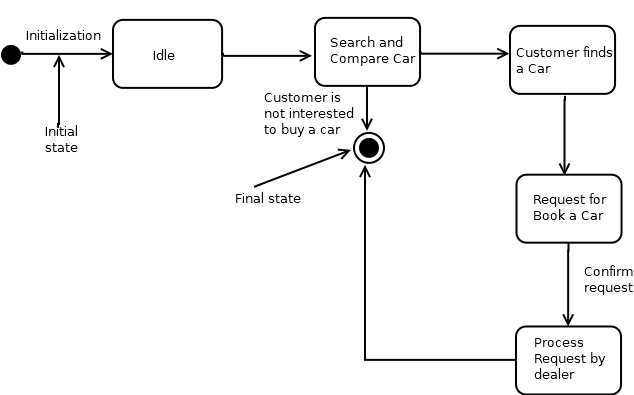
****

vehicles

vehicles

Vehicless()

**Statechart Diagram**

****

Book vehicle

the vehicle

vehicles

## P2.5 System Specification

### P2.5.1 Hardware Specification

Processor - Pentium IV/AMD Dual core

RAM - 1 GB

Hard disk - 500 GB

### P2.5.2 Software Specification

Front End - PHP

Backend - MYSQL

Client on PC - Windows 10

Technologies used - JS, HTML5, AJAX, J Query, PHP, CSS

## P2.6 Software Description

### P2.6.1 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 Ledorf 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 Preprocessor, 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.

### P2.6.2 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. We use the phrase “the SQL standard” to mean the current version of the SQL Standard at any time.

##### 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*.*

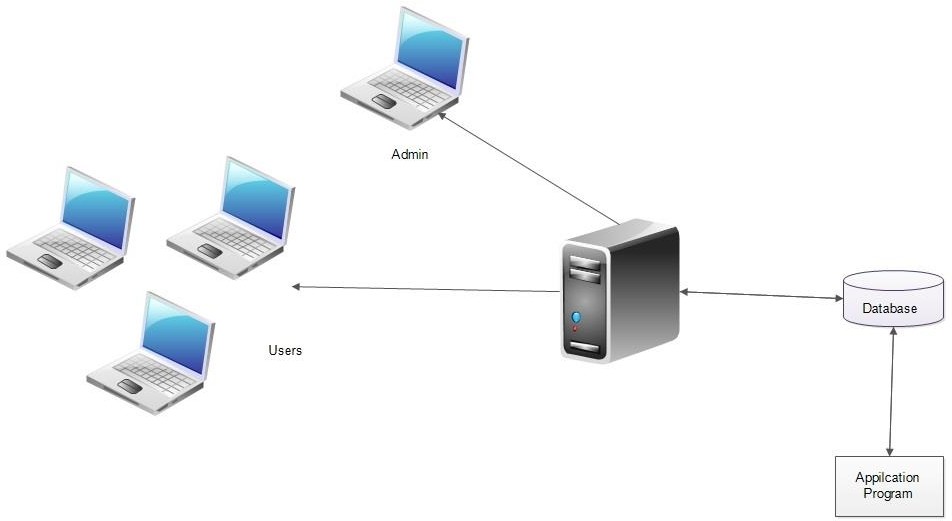
MySQL Server has a practical set of features developed in close cooperation with our users. It is very likely that your favorite application or language supports the MySQL Database Server.

## P2.7 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

### P2.7.1 Architectural Design



The registered user, admin, service provider can accesses the e-workshop 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

### P2.7.2 Module Design

**Admin Module**

The administrator of the company is allowed to access all the services in the system. And approve vehicle services appointment, add all products

|  |  |
| --- | --- |
| Manage user details, Add Used Vehicles | Block/Activate the registered users. |
| Adding Bid Vehicles | View ending auction details |

### Registered User Module

After registration, customers can book appointments for vehicle service online and also customer can Buy second hand vehicles and spare parts. And chat to the admin

|  |  |
| --- | --- |
| User registration, login | Search products/Add to cart |
| Request for bid vehicle | Manage profile |
| Compare Vehicles | Search Used Vehicles |

**P2.7.4 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 tale 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 key 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 donor 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 other non-key attribute.

##### TABLES

### 1.Registration

Table no : 1

Table name : tbl\_registration

Primary key :user\_id

Table Description : To store User details

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELDS** | **DATA TYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| user\_id | Int(10) | Primary Key | Unique User\_identification |
| l\_id | Int(10) | Foreign Key | Login\_identification |
| f\_name | Varchar(50) | Not null | First Name of the user |
| m\_name | Varchar(50) | Not null | Middle Name of the user |
| l\_name | Varchar(50) | Not null | Last Name of the user |
| address | Varchar(50) | Not null | Address of the user |
| dob | Varchar(50) | Not null | Dob of the user |
| phone | Numeric(50) | Not null | Contact\_nunmber |
| e-mail | Varchar(50) | Not null | Email\_id of the user |
| password | Varchar(50) | Not Null | Enter the valid password |

**2.Login**

Table no : 2

Table Name:-tbl\_login\_autoshop1

Primary key :l\_id

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELDS** | **DATA TYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| l\_id | Int(10) | Primary Key | Unique Login\_identification |
| Username | Varchar(50) | Not null | Username |
| Password | Varchar(50) | Not null | Password |
| Rol | Varchar(20) | Not null | Role |
| Status | Varchar(20) | Not null | Status |

**3.AUCTION**

Table No:-3

Table Name:-tbl\_auction

Primary Key:-auction\_id

|  |  |  |  |
| --- | --- | --- | --- |
| **FILDS** | **DATA TYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| auction\_id | Int(10) | Primary Key | Unique Auction\_identification |
| u\_id | Int(10) | Foreign\_Key | User\_identfiction |
| Amount | Numeric(50) | Not null | Auction Bid\_amount |
| Time | Timestamp | Not Null | Auction Bid\_Time |
| v\_id | Int(10) | Foreign\_Key | Car identification |

**4.ADD VEHICLE**

**Table No:-4**

**Table Name:-tbl\_addvehicle**

**Primary Key:-vehicleid**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELDS** | **DATA TYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| v\_id | Int(10) | Primary Key | Unique Vehicle identification |
| vehicle | Varchar(50) | Not null | Name of the Vehicles |
| basic\_price | Numeric(50) | Not null | Basic price of the vehicles |
| image | Varchar(50) | Not Null | Adding Vehicle images |
| description | Varchar(200) | Not null | Description of the vehicles |
| model | Varchar(50) | Not null | Model |
| color | Varchar(50) | Not null | Different Colors of vehicles |
| Fuel | Varchar(50) | Not null | Fueltype |
| Year | Numeric(50) | Not null | Year |
| mileage | Numeric(50) | Not null | Mileage of the vehicles |
| transmission | Varchar(50) | Not null | Transmission of the vehicles |
| registered | Varchar(50) | Not null | Check for Registered vehicles |
| add\_date | Date | Not Null | Adding The Auction starting Date |
| l\_date | Date | Not Null | Adding The Auction Ending Date |
| brate | Numeric(10) | Not Null | Adding Starting Bid amount |
| status | Varchar(50) | Not null | Status |

**5.NOTIFICATION MESSAGE**

**Table No:-5**

**Table Name:-tbl\_send\_sms**

**Primary Key:-msg\_id**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELDS** | **DATA TYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| msg\_id | Int(20) | Primary Key | Message\_identification |
| Uid | Int(20) | Foregin Key | Unique user\_identification |
| message | Varchar(100) | Not null | Admin send the message |
| con\_message | Varchar(100) | Not Null | User send the Confirmation Message |
| status | Varchar(20) | Not Null | Status of the message |

**6.PAYMENT**

**Table No:-6**

**Table Name:-tbl\_bank\_details**

**Primary Key:-bank\_id**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD** | **DATA TYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| bank\_id | Int(10) | Primary Key | Bank\_identification |
| Bank name | Varchar(50) | Not Null | Current Bank Details |
| cardtype | Varchar(50) | Not null | ATM Card Type |
| accountno | Numeric(16) | Not null | Enter The ATM Card Number |
| month | Varchar(20) | Not null | Adding the Month of the Card |
| year | Varchar(20) | Not null | Adding the year Expery Date of the ATM Card |
| cvv | Numeric(10) | Not null | Adding The cvv Number of the card |
| amount | Numeric(20) | Not null | Adding the balance amount of the card |
| status | Varchar(20) | Not null | Adding bank details status |

## P2.8 System Testing

### P2.8.1 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 analyzing 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 behavior 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 appear to be working according to the specification, that performance requirement appear 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.

### P2.8.2 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

### P2.8.2.1 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 where removed and ensured that all modules are working, and gives the expected result.

### P2.8.2.2 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.

### P2.8.2.3 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.

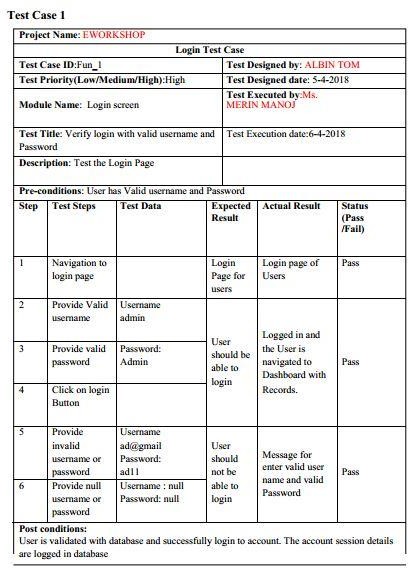
### P2.8.2.4 User Acceptance Testing

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.

## P8.8.3 Test Case



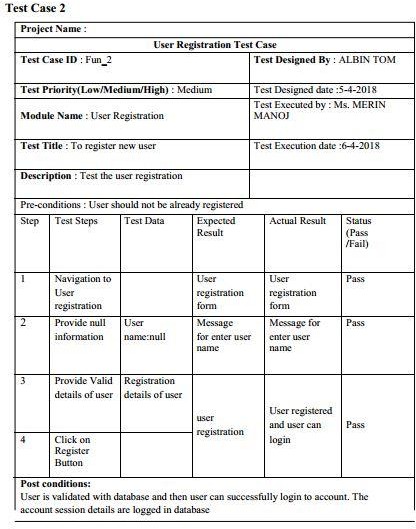
02-05-2019

13-05-2019

JISHNUMON PB

MR.JINSON DEVIS

AUTOSHOP



12-05-2019

02-05-2019

Mr. JINSON DEVIS

JISHNUMON PB

**P2.9 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.

### P2.9.1 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

### 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.

### 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.

### P8.9.4 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".

## P8.10 Conclusion &Future Enhancements

### P8.10.1 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

### P2.10.2 CONCLUSION

The software reduces the time consumption and the manual efforts of searching a products. 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.

### P2.11 Bibliography BOOKS/REFERENCES:

* + Gary B. Shelly, Harry J. Rosenblatt, “*System Analysis and Design*”, 2009.
  + Roger S Pressman, “*Software Engineering*”, 1994.
  + PankajJalote, “So*ftware engineering*: a precise approach”, 2006.
  + James lee and Brent ware Addison, “Open source web development with LAMP”, 2003
  + IEEE Std 1016 Recommended Practice for Software Design Descriptions.

### WEBSITES:

* + [www.w3schools.com](http://www.w3schools.com/)
  + [www.jquery.com](http://www.jquery.com/)
  + [http://homepages.dcc.ufmg.br/~rodolfo/es-1-03/IEEE-Std-830-1998.pdf](http://homepages.dcc.ufmg.br/%7Erodolfo/es-1-03/IEEE-Std-830-1998.pdf)
  + [www.agilemodeling.com/artifacts/useCaseDiagram.html](http://www.agilemodeling.com/artifacts/useCaseDiagram.html)

## P2.12 APPENDIX

### P2.12.1 SAMPLE CODE

##### Connection code

<?php

$conn = mysqli\_connect("localhost","root","","autoshop");

// Check connection

if (mysqli\_connect\_errno())

{

echo "Failed to connect to MySQL: " . mysqli\_connect\_error();

}

?>

##### Admin Add Vehicles

<?php

require "connect.php";

if(isset($\_POST["submit"]))

{

$vehicle=$\_POST["vehicle"];

//$basic\_price=$\_POST["basic\_price"];

$description=$\_POST["description"];

$model=$\_POST["model"];

$color=$\_POST["color"];

$fuel=$\_POST["fuel"];

$registred=$\_POST["registred"];

$year=$\_POST["year"];

$milage=$\_POST["milage"];

$transmission=$\_POST["transmission"];

$target\_dir = "uploads/";

$target\_file = $target\_dir . basename($\_FILES["image"]["name"]);

move\_uploaded\_file($\_FILES["image"]["tmp\_name"], $target\_file);

$image=$target\_file;

echo $suc =mysqli\_query($conn, "insert into addvehicle(vehicle,image,description,model,color,fuel,registred,year,milage,transmission) values('$vehicle','$target\_file','$description','$model','$color','$fuel','$registred','$year','$milage','$transmission')") or die (mysqli\_error());

}

?>

<html>

<head>

<link href="css\style1.css" rel="stylesheet">

</head>

<body background="images/123.PNG">

<div id="main">

<center>

<div id="top"><b>ADD VEHICLES</b></div>

<div id="menu"><center><a href="adminhome1.php">HOME &nbsp&nbsp </a>

<!-- </a><a href="addvehicles.php" >ADD CARS &nbsp&nbsp</a>

</a><a href="confirmation\_mail.php">CONFIRM &nbsp&nbsp</a>

</a><a href="login\_autosh.php">CUSTOMERS &nbsp&nbsp</a>

</a><a href="login\_autosh.php">SALES &nbsp&nbsp</a>

</a><a href="login\_autosh.php">PROFILE &nbsp&nbsp</a>

</a><a href="logout.php">LOGOUT &nbsp&nbsp</a> -->

<br>

<br>

<br>

<form action="#" method="post" onsubmit="Alert()" enctype="multipart/form-data" >

<div id="signup">

<h2 class="smp">UPLOAD DETAILS<h2>

<center>

<font size="4px">

<label>Vehicle Name:</label>

<input id="vehicle" name="vehicle" placeholder="Please Enter the Vehicle Name" type="text" required pattern="^[a-zA-Z0-9]+$"><br>

<!--<label>Basic Price:</label>

<input id="basic\_price" name="basic\_price" placeholder="Basic Price" type="number" pattern="\d+(\.\d{2})?" required><br>

!--><br>

<label>Upload Image:<br></label>

<input type="file" id ="image" name="image" accept=".png,.jpg,.jpeg,.JPG" placeholder="Upload Vehicles Image"><br><bR>

<label>Description:</label><br>

<textarea placeholder="description" name="description" required></textarea><br>

<h2 class="smp">FEATURES<h2>

<label>Models:</label>

<input id="model" name="model" placeholder="Model Name" type="text" required><br>

<label>Color:</label>

<input id="color" name="color" pattern="^[a-zA-Z0-9]+$" placeholder="Color" type="text" required><br>

<label>Fuel Type:</label>

<select id="fuel" name="fuel" pattern="^[a-zA-Z0-9]+$" placeholder="Fuel" type="text" required>

<option value="">Select</option>

<option value="Petrol">Petrol</option>

<option value="Diesel">Diesel</option>

<option value="Diesel">Gas</option>

</select><br>

<label>Registerd:</label><br>

Yes<input id="registred" name="registred" placeholder="Registred" type="radio" value="yes" required><br>

No<input id="" name="registred" placeholder="Registred" type="radio" value="no" required><br>

<label>Year:</label>

<input id="year" name="year" pattern="[0-9]{4}" placeholder="Year" type="date" required><br>

<label>Milage:</label>

<input id="milage" name="milage" placeholder="Milage" type="text" required><br>

<label>Transmission:</label>

<select id="transmission" name="transmission" pattern="^[a-zA-Z0-9]+$" placeholder="transmission" type="text" required>

<option value="">Select</option>

<option value="Automated">Automated</option>

<option value="Manuel">Manuel</option>

<option value="Manuel">Automated & Manuel</option>

</select><br>

<!--<label>Transmission:</label>

<input id="transmission" name="transmission" placeholder="Transmission" type="text" required><br>

<br>!--><br>

<input name="submit" type="submit" value=" SUBMIT ">

<br>

<script>

function Alert()

{

alert("Successfully Updated!!!!!");

}

</script>

</font>

</center>

</div>

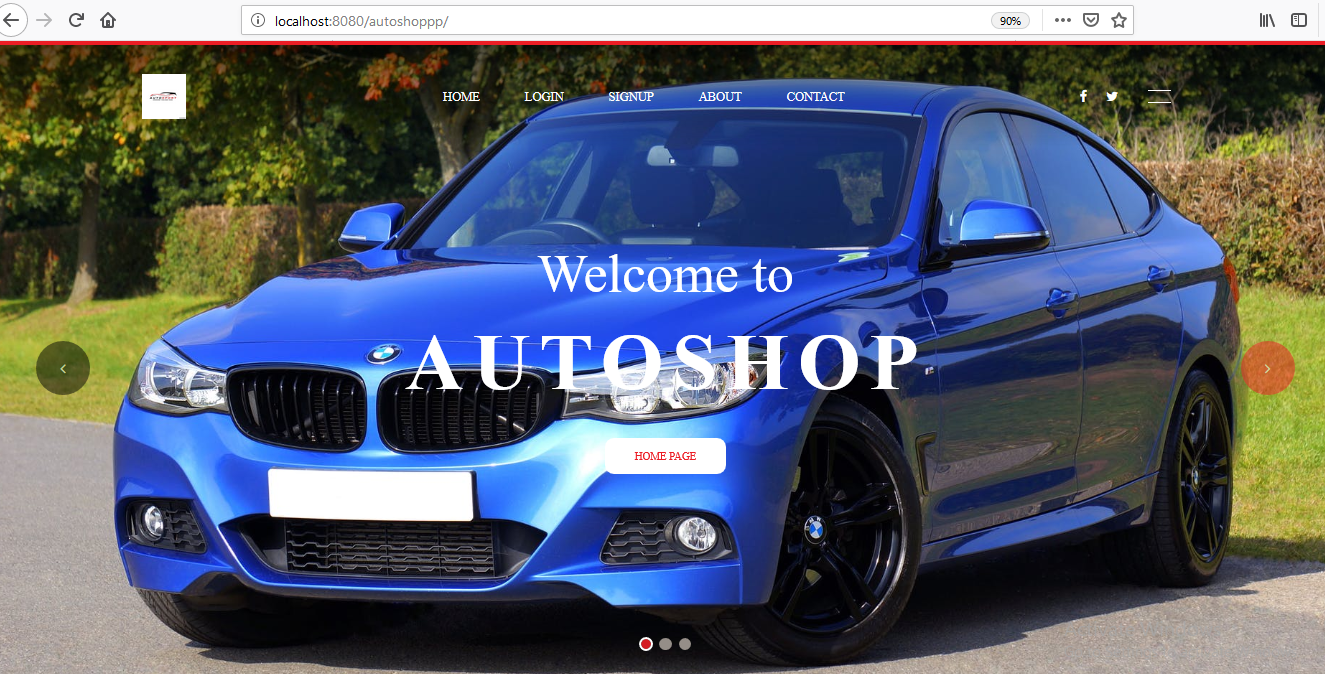
</form>

<body>

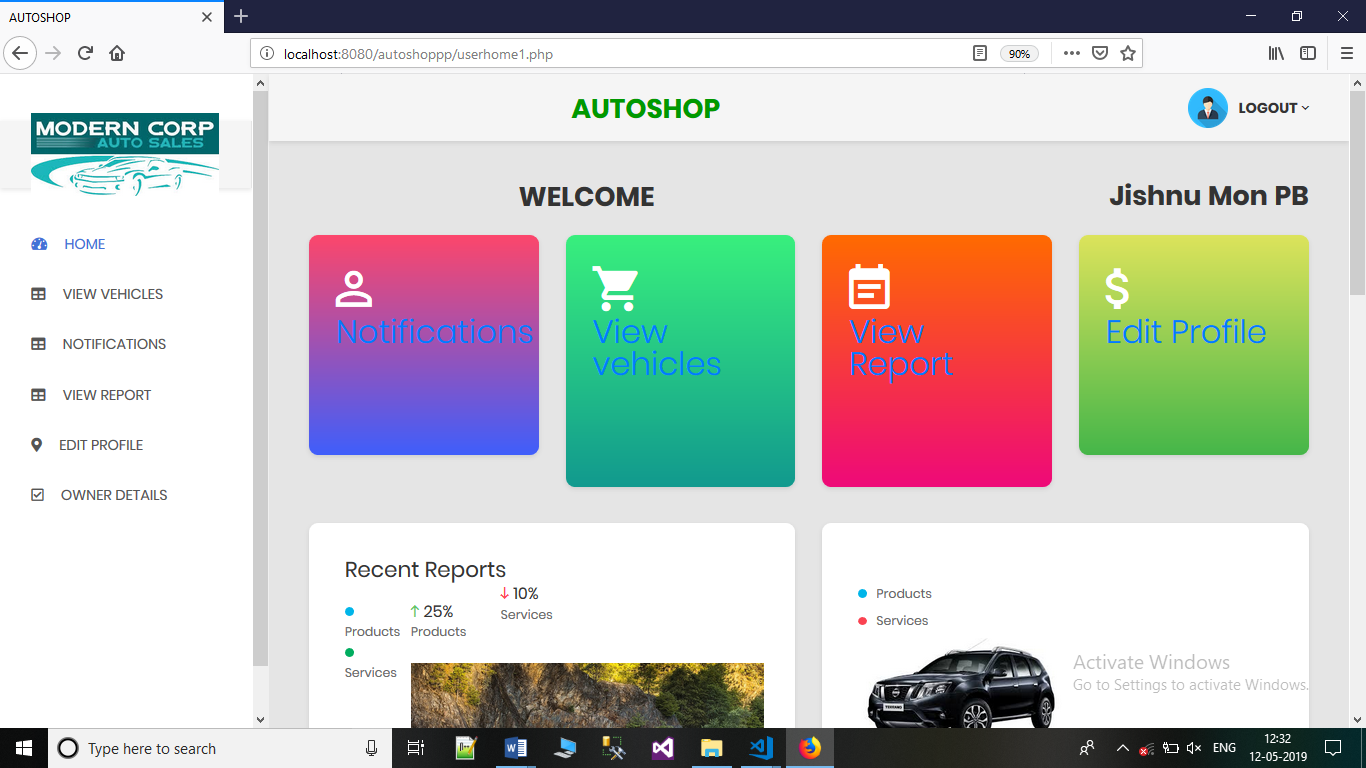
</html>

### P2.12.2 SCREENSHOTS

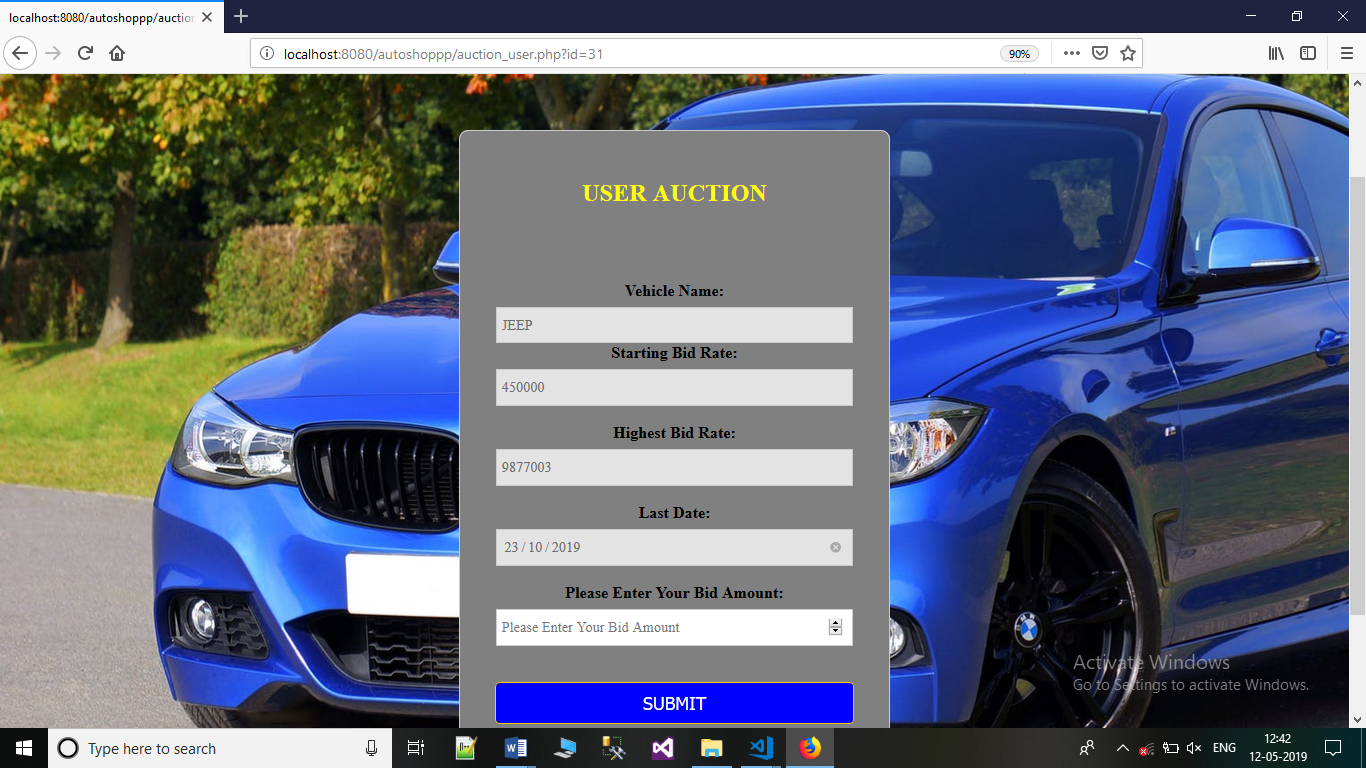
Main Home (Index) page

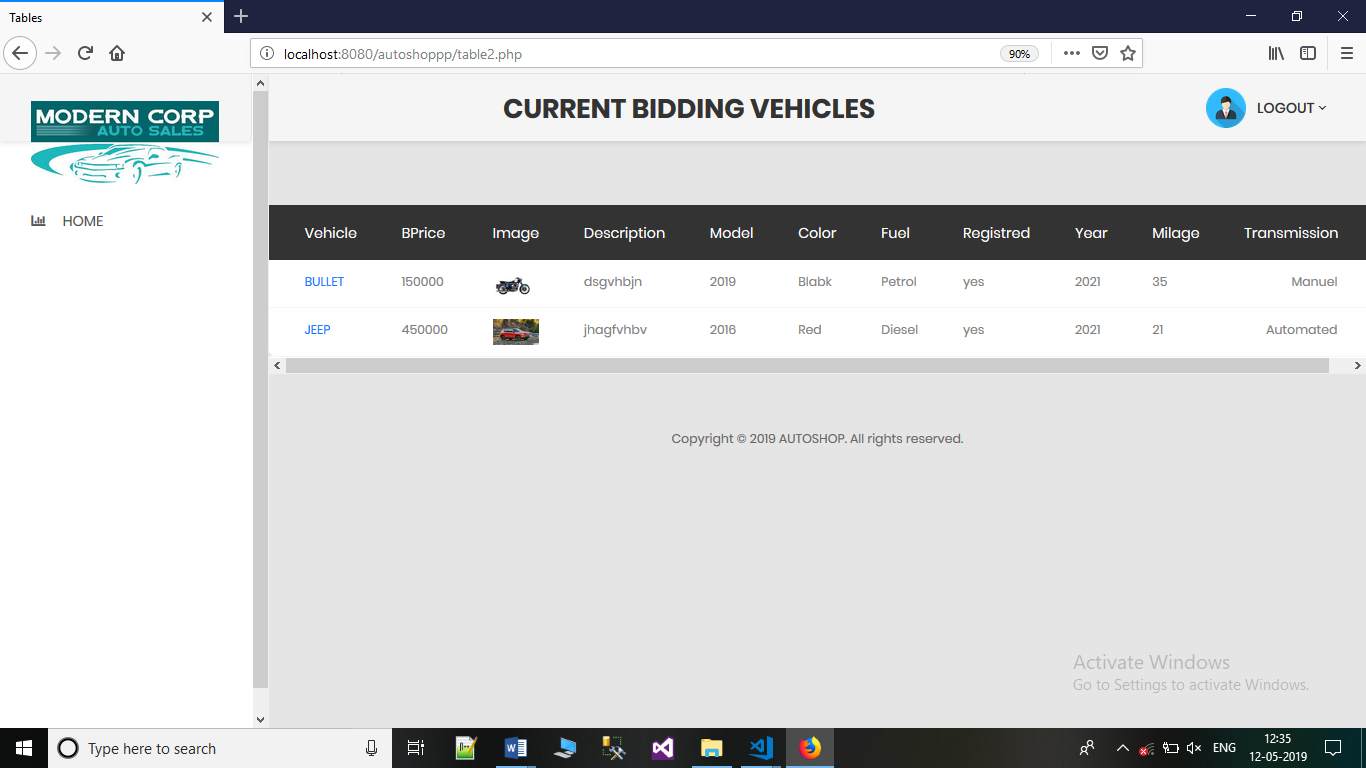


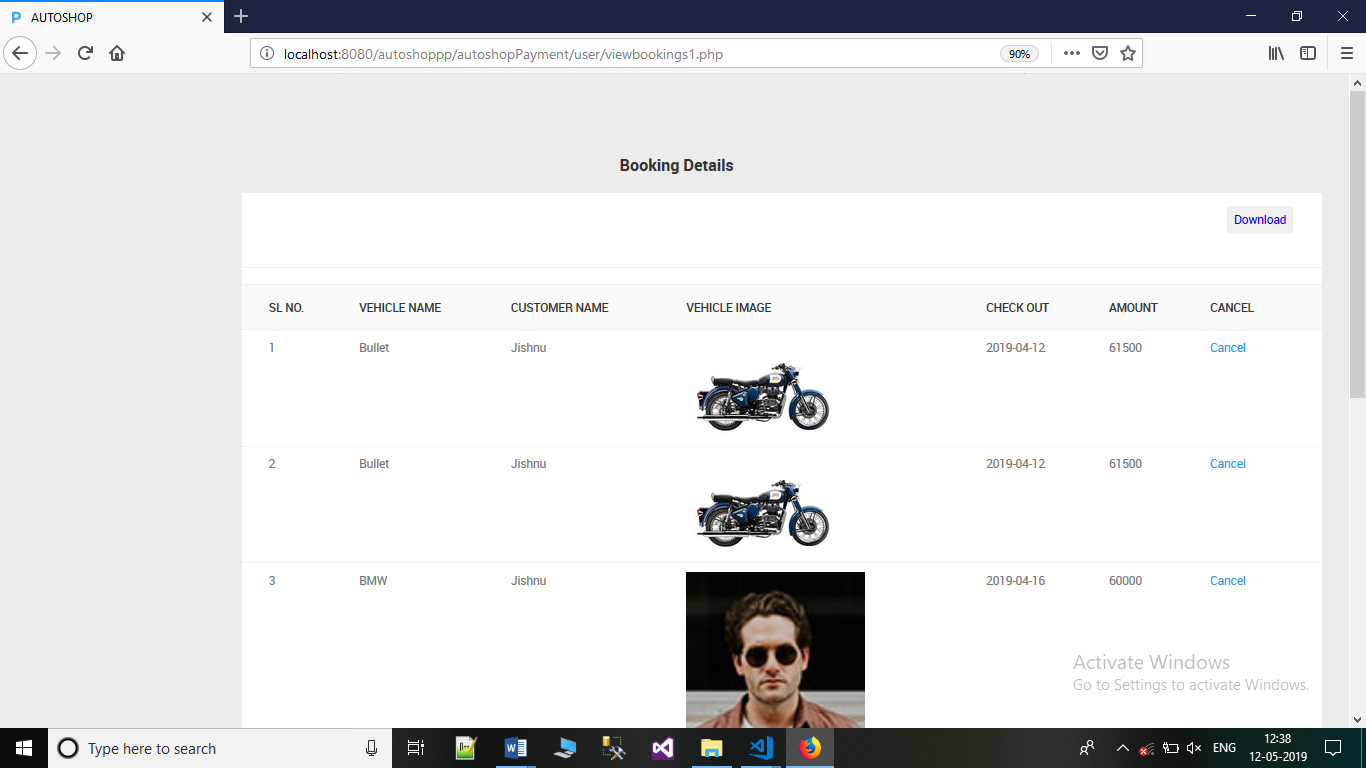
User home

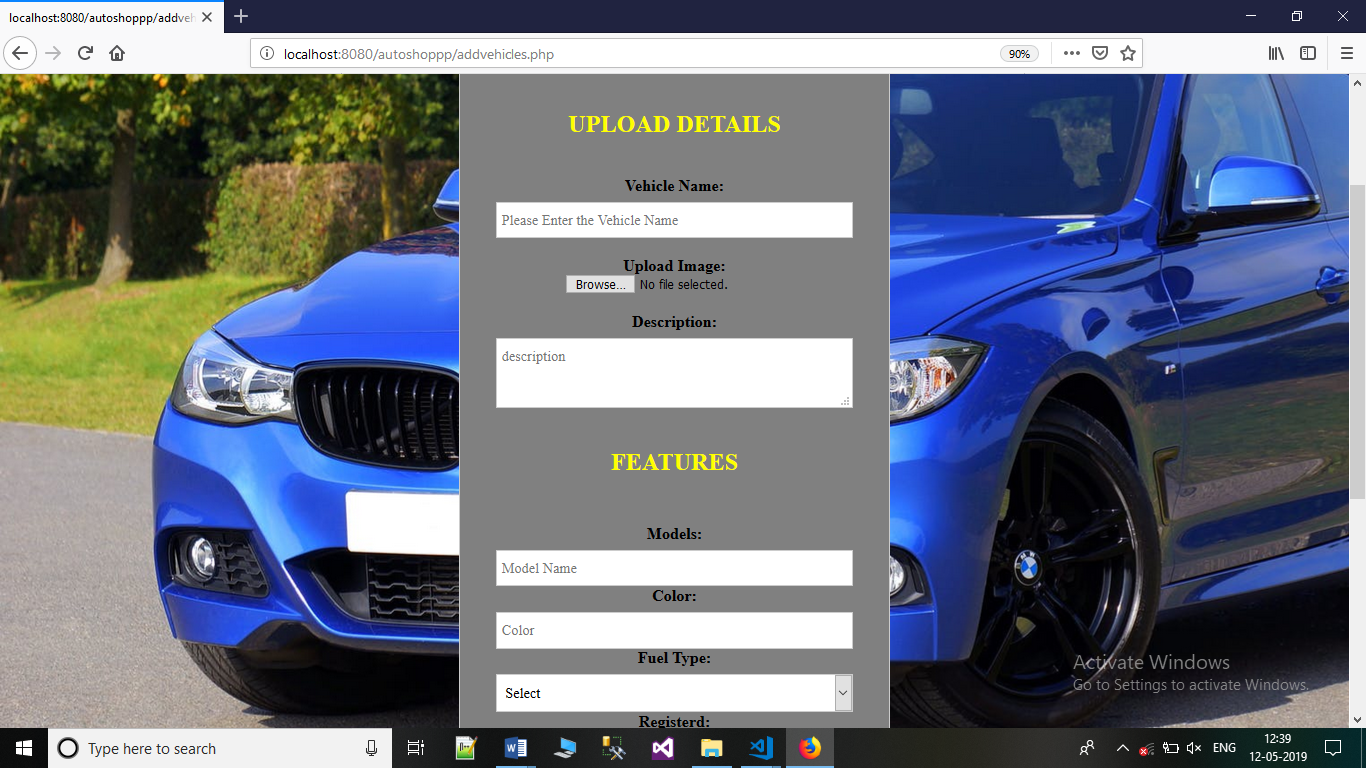


Bidding



Search vehicles page

Admin view appointments

Admin Add vehicle details