Date ______
Page ______

Active Service Page

ASP. NET MVC

is an open source platform for building desktop, web and mobile application. It includes tools, libraries & languages to high performance software development.

It is a web application framework developed by Microsoft.
It is a development framework for building webpages and websites. MVC stands for Model View Controller.
MVC is open source software design pattern.
This framework is a lightweight, highly testable framework that is integrated with existing ASP.NET features.
First official version 1.0, of ASP.NET MVC is released in 2009.
Latest version of ASP.NET MVC is 6.0 that is released in year 2016.
Its inventor is Scott Guthrie.

Its Features

Seperation of Concern
Loosely Coupling
Parallel Development
Easy to perform unit testing
Test Driven Development (TDD) support
Clean URL's
Improved performance
Task support for Asynchronous Support
Bundling and Minification
More Controlling on HTML
Introduced new Concepts
Exeption
① Filters ② Scaffolding Template
③ WebAPI ④ LINQ etc

ASP-

* An ASP, commonly called an ASP page, is a webpage that may contain scripts as well as standard HTML.
* ASP enables web servers to dynamically (तौर से) generate webpages and create interactive web application by using server side scripting technology.
* It contains HTML, which defines the page layout fonts and graphic elements and embedded programming code that's written in a microsoft scripting language.
* The use of ASP with Microsoft IIS (Internet Information services) are tied to support lifecycle of the host OS.

Razor Engine

* ☆ It is a view engine i.e used to process a view page in .NET MVC.
* ☆ Razor engine identifies the code of C# and HTML to generate webpage.
* ☆ Razor is new view engine. aspx view engine is old view engine i.e, used to asp.NET.
* ☆ To write the code of programming language in view page we have to use razor block as shown below-

```
@{
        //statements....
}
```

☆ Some Important

* ☆ App data folder is used to write backup of databas
* ☆ App-start folder is used to write the setting related of project start.
* ☆ Content folder have the content of project (images, video, js, css, url, etc.)
* ☆ Controllers folder are used to write programmi or logic of project.
* ☆ Fonts folder are used to ~~[illegible]~~ the file of font family
* ☆ Models folder are used to keep the files relat of database.

* Scripts folder is used to keep the files of JS.
* Views folder is used to keep the design page of project.

Web.config File

* It is a special file of .net web application.
* This files contains setting related to our project like as which framework version we are using, maximum upload length of project, database server name and traditionals, assembly version etc.
* This file is written in XML language

Views in MVC-

* In MVC, V stands for view.
* View is used to design a webpage.
* View is GUI part of project
* View page contains HTML, CSS, JS, jQuery, and C#. Mainly it contains HTML & C#. Hence extension of a view page in MVC is .cshtml
* In MVC project, views are created inside "views" folder in related controller folder For example if we create a view for Login page of Home controller then it will be stired inside "Views/Home/" folder
* view pages are processed by action methods.

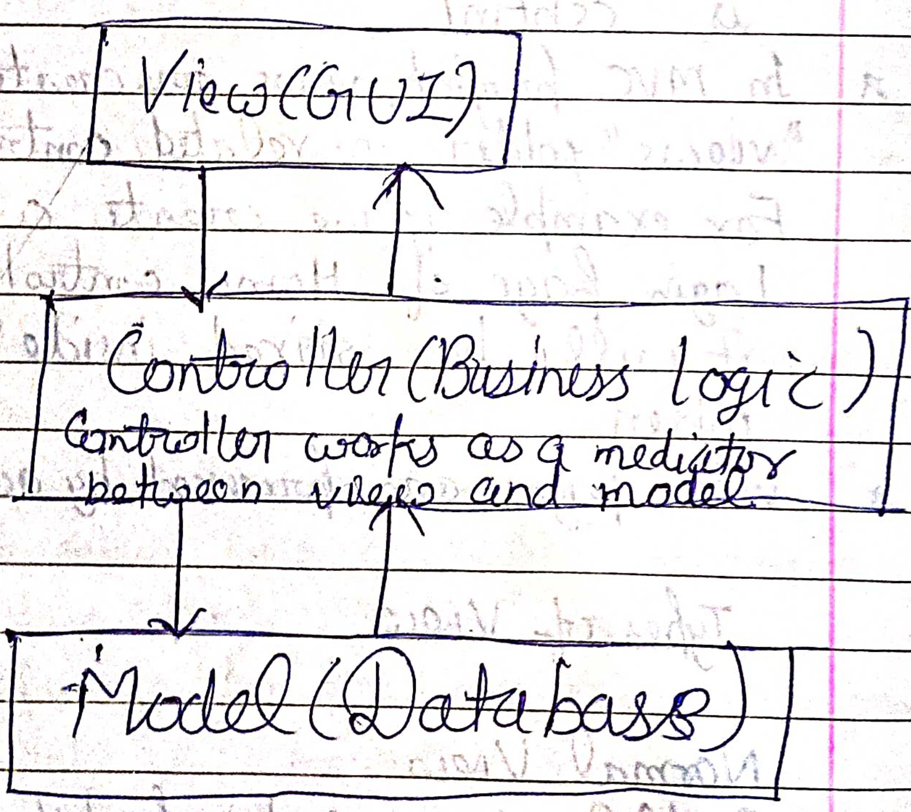
Types of View

Normal View
Partial View: contains part of another view page.
Shared View: It works like master page. It can contains common design of webpage.

This type of views are created inside "Shared" folder in "Views folder.

Controllers in MVC

- ☆ Controller is the request handler
- ☆ In MVC, C stands for controller
- ☆ Controller is responsible for business logic of project.
- ☆ Controllers contains programming code (logic of project
- ☆ Controllers are the most important part of a MVC application.



View (GUI)
Controller (Business logic)
Controller works as a mediator between views and model.
Model (Database)

-{Architecture of MVC}-

* Controllers are created as a class in MVC application. To create a class as a controller we have to inherit "Controller" class in our class.
* All controllers must inherit a base class "Controller".
* Controller name must have a postfix "Controller". name of controller must be end from "Controller".
* Controllers are the collection of various action and nonaction methods.
* We can create multiple controllers in our project as per our need.

Action Methods

* Action Methods are special methods of a controller that are used to process/fulfill a view page.
* Action methods are directly callable from a view page.
* Return type of action method is object of "Action Result" class. It can also return object of another class that is present in hierarchy (family) of "Action Result" class. Ex - "Json Result".
* Each view page has at-least one action method.

Types of Action Methods

Get Action Method
Post Action Method

Note- For a view page, name of Get and and Post action method must be same.
Ex- For index page, name of Get and Post action method will be "Index".
Hence we have to overload get and post action methods.

① Get Action Methods-

☆ Get Action method is used to process a page when page (view page) loads first time.
☆ When a view page directly comes from server then it will be processed by get action method.
☆ We can also process request of from tags in case of get method by using get action method.
☆ ~~[HTTP~~ [HttpGet] attribute is used ~~to~~ before a get action method to define it as a get action method.

Post Action Method

* It is used to process a webpage (view page) when user submits data of form by using "post" method of form tag.
* When user sends data of page to server then post action method is used to process that data.
* [HttpPost] attribute is used before post action method name.

Data Transfer from view to Controller

* We can send data of a view page to controller page so that we process that data inside controller.
* To send data from view page in controller MVC provides below option-

① Name of form-controls as action method parameter

② Object of FormCollection class as action method parameter.

③ Object of Model class as action method parameter.

Date
Page

(1) Name of form-controls as action method parameters -

* In this method we can read data of a view page by passing name of form controls as parameters of action method
* In this method we don't have to perform ~~ty~~ type conversion because it is performed automatically
* In this method name of form controls and name of parameters must be same
* Form big web forms we have to pass a very long list of parameters. It is complexity of this method

Object of FormCollection class as Action method parameter -

* We can pass object of 'Form Collection' class as parameter inside action method. It will read all value of Form controls
* 'FormCollection' is a built-in class.
* In this method, we will get values in string format. So we have to perform type conversion as per our need.
* We don't have to pass a long list of parameters inside action method

Object of Model class as Action method parameter –

* We can pass the object of model class inside action method as parameter to read values of form controls (view) in controller.
* In this method, we don't have to pass a long list of parameters inside action method
* In this method, we don't have to perform type conversion.
* It is necessary to have same name of properties of model class and name of form controls

Model Class

* Model represents the data.
* It is a special class in MVC application i.e, used to perform database operations.
* Model class is also used to read values from view page.
* Model classes are also created inside 'Models' folder.
* To use Model classes we have to include a namespace in controller/view.

Syntax – using ProjectName.Models;

Ex – using class2_2.Models;

* Properties of Model class are created as getter/setter

Data Transfer from Controller Action Methods to View :-

We can send data/results from ~~MVC~~ actionmethods to our view page in MVC framework. For this MVC provide below options.

1. ViewData
2. View Bag
3. TempData
4. Object/Array of Model class

ViewData

- ★ It is an object of 'ViewDataDictiona class.
- ★ It stores data in key/value pair.
- ★ To read data of a specific type we have to perform type conversion in ViewData. However we can store any type of data in ViewData.
- ★ ViewData is used to transfer data from an action method to releated view page. It is not available for other action methods and other views So it is limited to an action method and a view.

Syntax- ViewData ["keyname"] = value;
Ex - ViewData ["result"] = sum;

Code in Controller for above

```
// ViewBag.result = msg;
ViewData ["result"] = msg;
ViewData ["num"] = 25;
View Data ["num2"] = 45;
View Bag.mynum = 12;
```

Code in View for access

```
<p> @ViewData["result"] </p>
@{
      ij (IsPost == true)
      {
      int n = int.Parse(ViewData["num"].To-
                 String());
     <span> @n </span>
     int p = ViewBag.MyNum;
     <p> @p </p>
      }
}
```

ViewBag

- ☆ It is also an object of 'ViewDataDictionary' class.
- ☆ It is used to transfer some data/result from action method to view page.
- ☆ ViewBag usage dynamic expression. So we can use any alphanumeric name for our viewpage
- ☆ No need to perform type conversion in ViewBag.
- ☆ Data of ViewBag is accessible to a specific action method and a specific view.
- ☆ Syntax - ViewBag.DynamicExpression = value;

  Ex - ViewBag.Result = x;

TempData

- ☆ It is an object of "TempDataDictionary" class.
- ☆ It stores data in key/value pair.
- ☆ To read data of a specific type we have to perform type conversion in TempData However we can store any type of Data in TempData
- ☆ TempData is accessible to current view, current action method and for next requested view/action method. So

its scope is from current page to one next page.

* TempData is temporary form of Session
Syntax-
TempData["keyname"] = value;
Ex- TempData["msg"] = "Record deleted from database.";

Strongly Typed View (Object/Array of object of Model Class)

* When a view page is used to display the data of a specific model then it is known as "Strongly Types View"

* We can display object or array of object of a model class in strongly typed view.

* To create a view as strongly types view we have to pass object/array of object of model class in return view() function of action method.

* To display data in strongly types view we have to use namespace of models folder and also we have to define name of model class by using model keyword as shown below-
@using Projectname.Models
@model Name of model_class

Ex- @using Test3.Models
@model Student

⇒ We will get object/ Array of object of model class in "Model" object. So inside body tag we will use "Model" object to display data.

File Uploading in MVC

★ To upload a file first of all, we have to use post method in form tag and specify enctype = "multipart/form-data" in form tag

★ Take a input type="file" control and submit button

★ To ~~read~~ submitted file inside action method we have to use Request.Files. It returns object of "HttpPostedFileBase" class

Syntax - HttpPostedFileBase objName = Request.Files
["name of file tag"];

Ex - HttpPostedFileBase file = Request.Files["my

Some important properties of HttpPostedFile Base class:-

1 FileName - This property is used to get name of uploaded file in string format.
2. ContentLength - used to get the size of uploaded file in bytes.
3 SaveAs() - used to save uploaded file in a folder of server. In this function we have to specify the path of our folder by using Server.MapPath() function

Syntax -
Obj of HttpPostedFileBase. SaveAs(Server.MapPath ("Path of folder") + file_name));

Note - For other validations, we have to write Manual logic.

Layout ~~Shared~~ View in MVC

* In a web based project we have to same create header and footer for multiple webpages.
* Layout view is used to write common design of multiple webpages. Like as we can create header and footer in layout view.
* Layout views are created inside "Views/Shared" folder
* We have to use @RenderBody() function to load a child page between content of Layout view.

```
static void Main (string [] arg)
{
    int float n1, n2, n3;
    Console.Write ("Please Enter three no.:");
    n1 = float.Parse (Console.ReadLine());
    n2 = float.Parse (Console.ReadLine());
    n3 = float.Parse (Console.ReadLine());
     res = n1+n2+n3;
    Console.WriteLine ("The addition of no. is: "+res
    Console.ReadKey ();
}
```

☆ Layout View is also known as Master Page or Master Layout.

☆ We don't have to create any separate action method for a layout view