ASP.NET MVC – Part 1

Agenda

- Creating & Running a Simple Project in ASP.NET MVC

Tools

- Visual Studio 2013

Pre-requisite

- Understanding of ASP.NET MVC Basic Concepts

Version	Last updated	Comments	Modified By
V1.0	07-05-2016		Bilal Shahzad

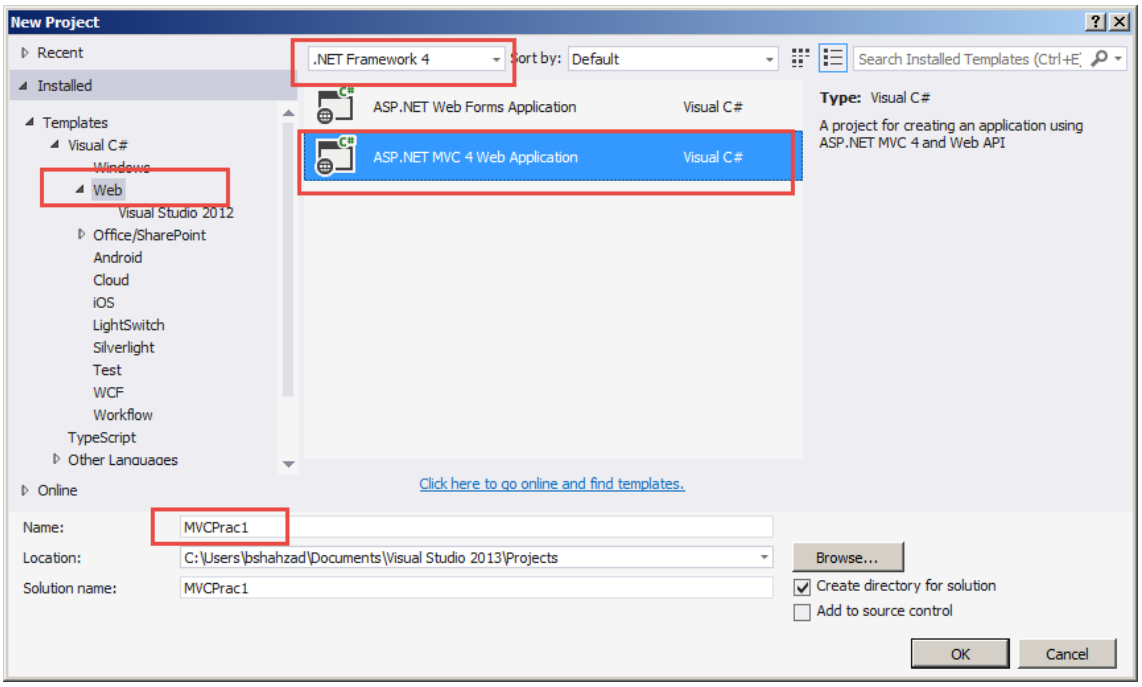
Brief Introduction

ASP.NET MVC follows MVC architectural design pattern. In this design pattern, elements are mainly divided in three main components (Model, View and Controller). To understand, we can consider Model as BAL + DAL. Controller contains the classes which handles the user requests, may talk to Model (if required) and then process appropriate view and returns the response to client.

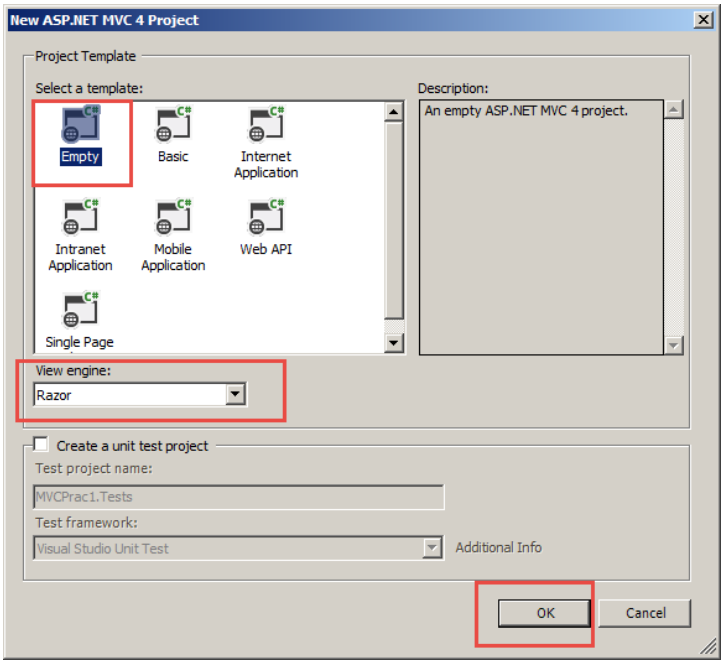
Visual Studio provides different options while working with ASP.NET MVC. We'll start with very basic template. This tutorial will not discuss concepts in detail but will give step by step guide. I'll advise you to explore concepts in depth wherever you feel any confusion.

Step by Step Walkthrough

1- Create a new Project (File -> New -> Project) and choose the option highlighted below. Give it a name.

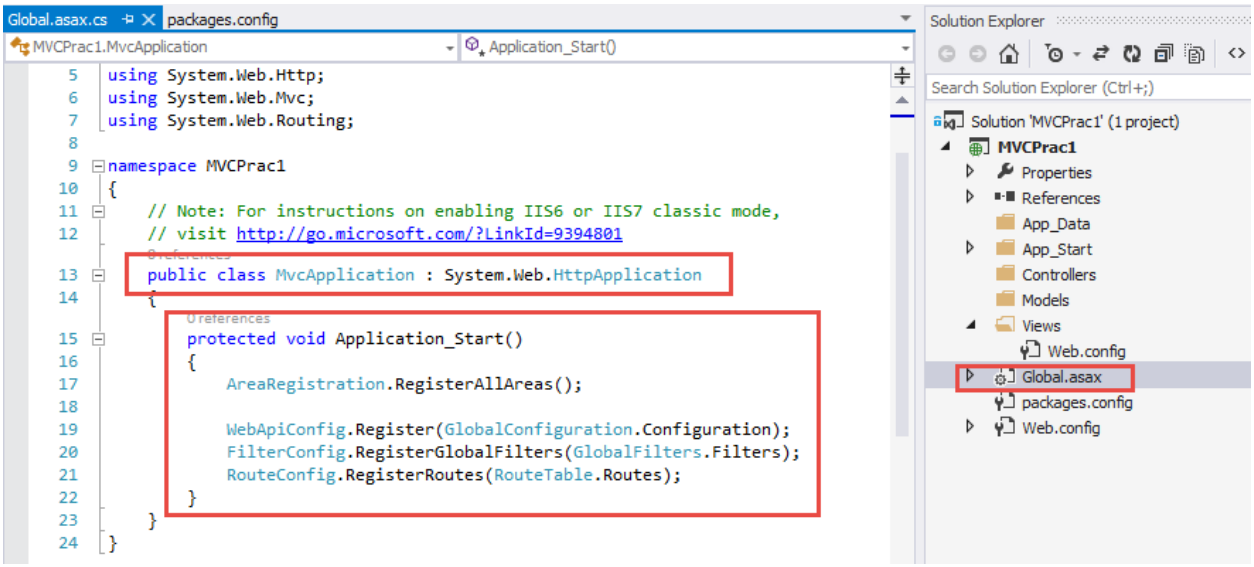


2- In next window, you will see different options to start with. For now, we'll choose "Empty" template and "Razor" as "View Engine".



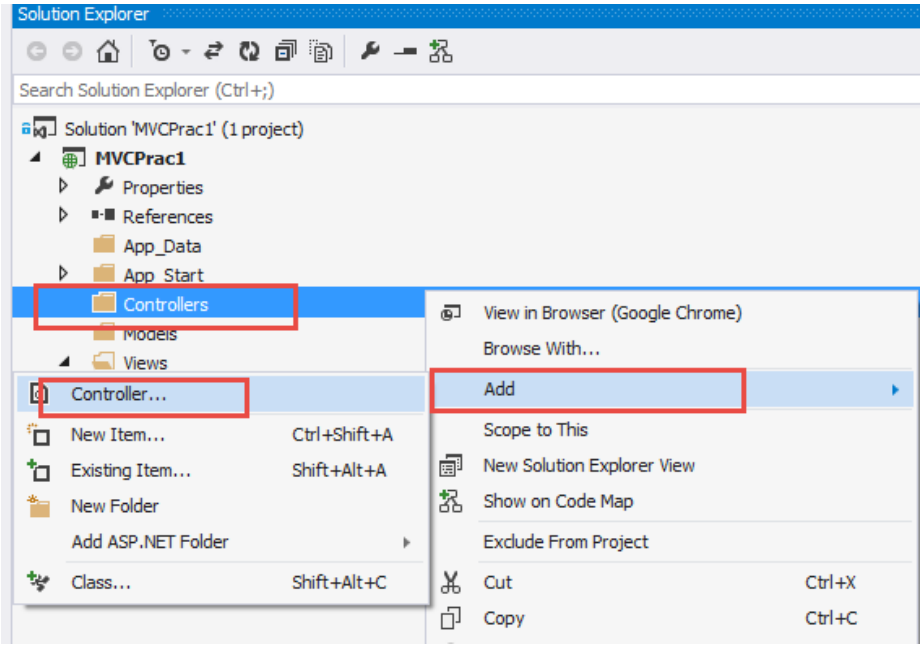
3- You will see different folders and files in the solution (created by default). Names of folders are self-explanatory (e.g. Controllers, Models and Views). You will see that some packages are by default added. You can check detail of those packages by opening "packages.config" file. **(Note: Nothing to do for you in this step)**

4- If you open "Global.axax" file, you will see a class which is being inherited from "HttpApplication" class. This class contains application level methods. For example, "Application_Start" method is called when application starts the first time. You may write your one time initialization stuff in this method. **(Note: Nothing to do for you in this step)**

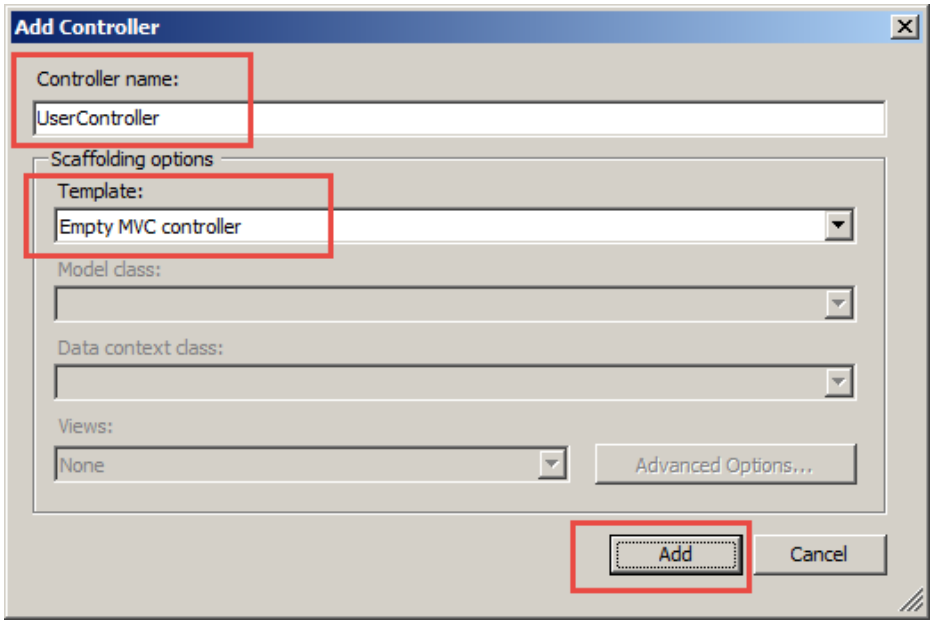


5- If you open main "web.config" file, you will not that there are some default configuration available in this. Discussion of whole web.config is out the scope of this tutorial but you can see "namespaces" section here. Here you can mention all the namespaces which you don't want to add on every page. So for example, you can access items of these namespaces in your controllers, views etc. without adding reference of namespace at top.

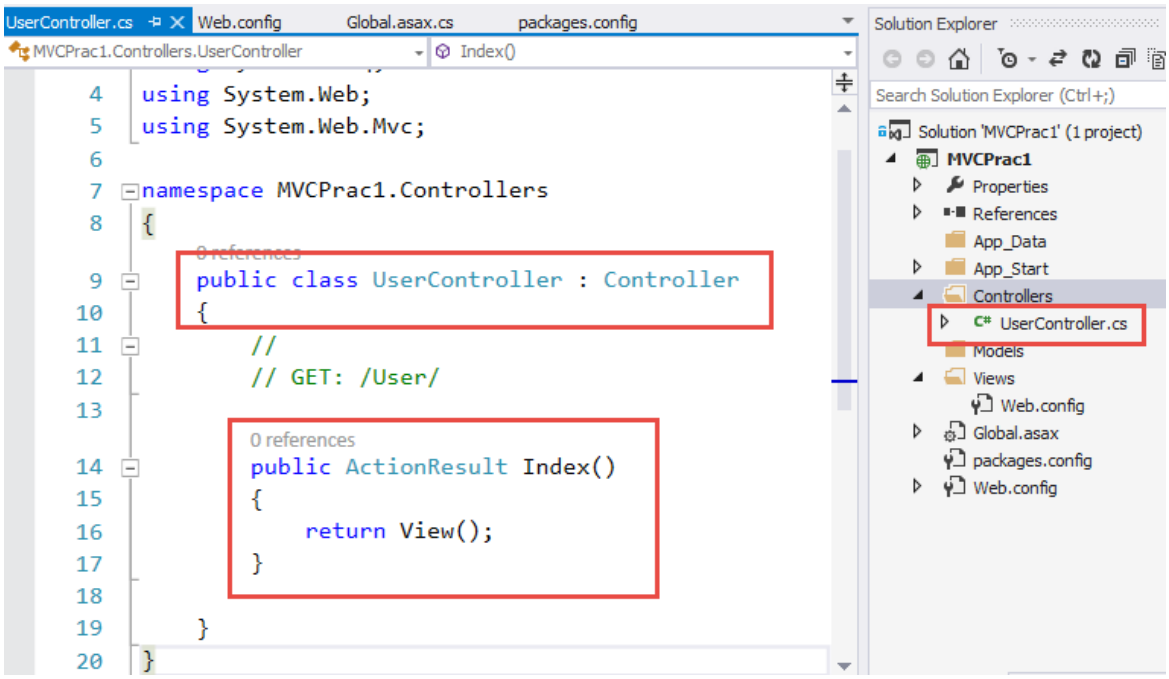
6- Now let's add a controller in our "Controllers" folder. To do so, right click on Controllers folder => Add => Controller.



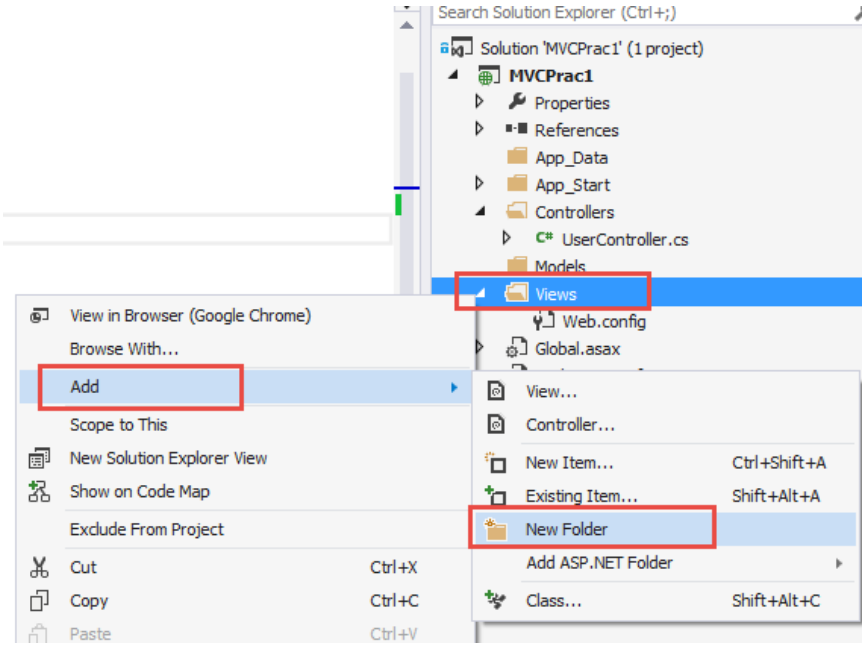
7- You will see following dialog box. Provide the name of controller here. Note that "Controller" post fix should be there in your controller name. For now, we'll use "Empty MVC Controller" template. Click Add.



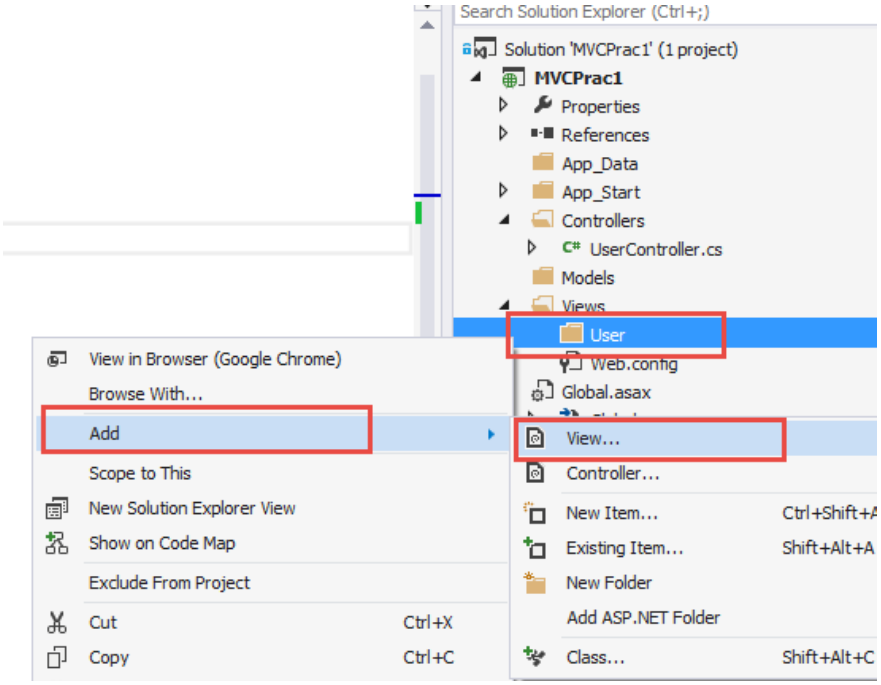
8- This will add a class file. Here we can see that a controller is a class which is being inherited from "Controller" class. A function named "Index()" is also available by default. This function (in the controller) will be called "Action". Action methods return result type of "ActionResult" class. So here "View()" function is being called and the return value of this function is being returned from our action method.
Also, we know that request lands on a controller and then on a specific action. As here "View()" method is empty, it will look for a folder named "User" in Views folder and then a file "Index.cshtml" in that folder.



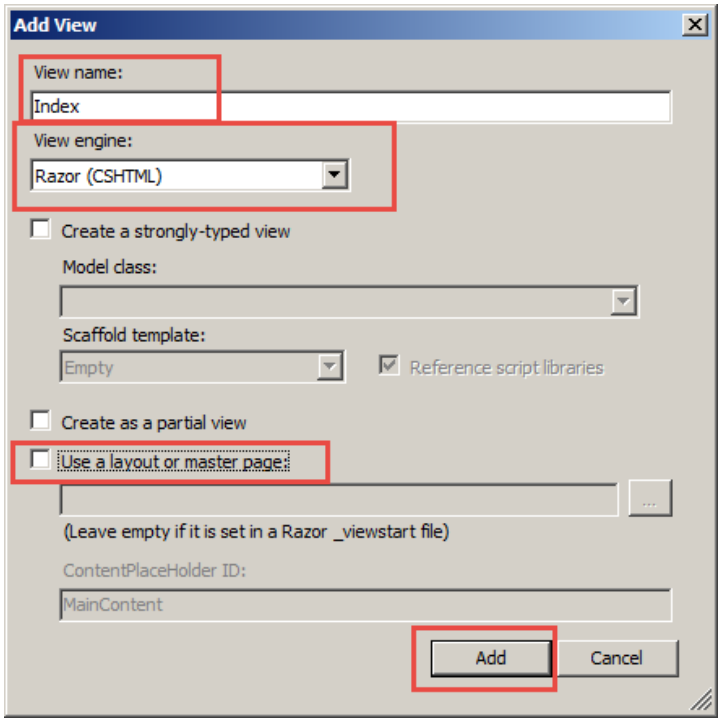
9- Now let's add a view file (which will contain our html). By following basic conventions, we'll have to add a folder first in our "Views". Name of this folder will be same as our controller name. Let's create a folder named "User" by right clicking on views -> Add -> New Folder.



10- Now let's create a view (with same name of our action name) in "User" folder. To do so, right click on "User" folder -> Add -> View.



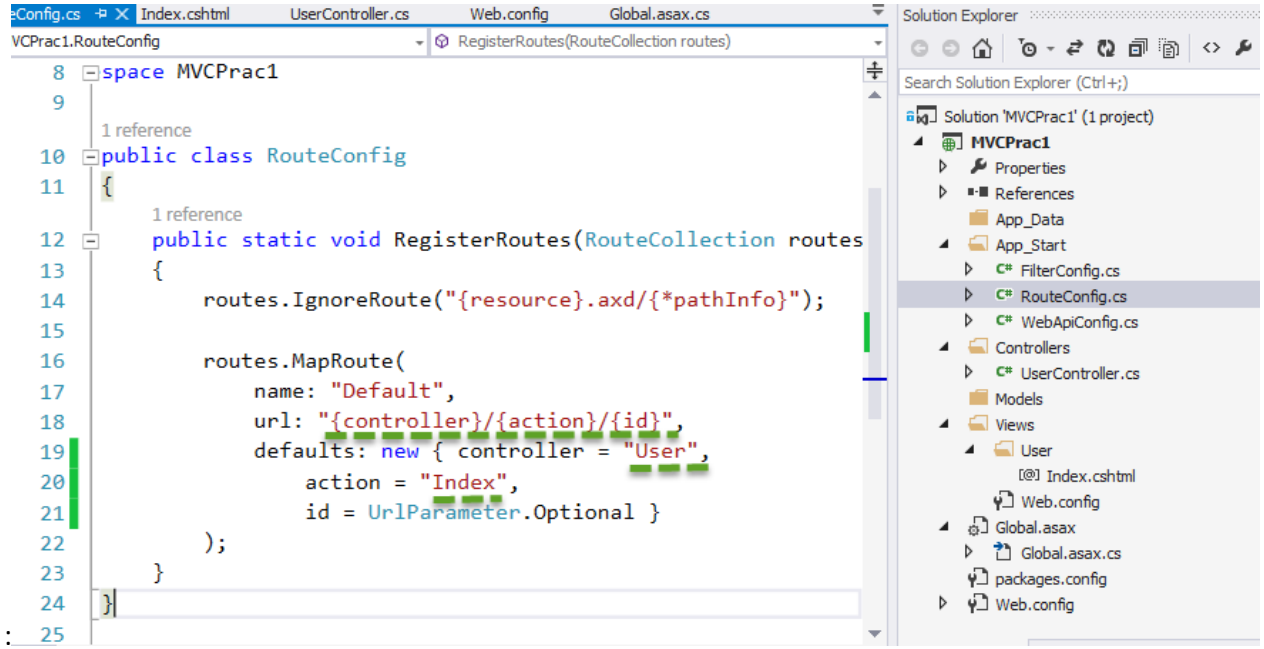
11- Keep the name of View as "Index" (same as our action name). Select "Razor" as View engine. Follow the selection according to following screenshot and click on "Add" button.



12- A file “Index.cshtml” will be added. This is our view file. If you open this file, you will see something like shown in this screenshot. Currently, this file is not using any “Layout” so you can see that “Layout” property is null. You can also see some basic HTML is written in this page.

13- Let’s make a small change in our view file and add a simple text “Pakistan Zindbad” in div element as highlighted below.

14- MVC allows us to set custom routs. Requests land on some controller and then on some action. We can configure the route pattern though. Open "RoutConfig.cs" file in "App_Start" folder. You will see that some sample code is already added here. "MapRoute" function is taking some configuration. Route name is set to "Default". URL pattern is telling that user will have to add controller first then action name first and then some value. Also we are providing default values if some parameter misses in URL (request). Here we've set default controller to "User" and default action to "Index". Now in browser if user types "localhost:someport/User", route engine knows that which action to choose. If user types "localhost:someport", route engine knows that request will be landed on "User" controller and then on "Index" action.



15- Now run the project and see what is URL and what result is. Try changing URL by adding controller name only. Then by adding controller and action name.

Tasks for Practice

Once you are done with above tutorial. Try to work on these tasks.

1- Create one more action “Home” in “User” Controller same like “Index” action.
2- Create view for “Home” action.
3- Create a new Controller “Student”
4- Create a new action “Index” in “Student” Controller.
5- Create a view “Index.cshtml” (in “Student” folder”)
6- Make changes in your RouteConfig that when you run your application, “Student/Index” action should be called by default.
7- Right Click on “Controller” class => “Go to Definition. Check the members of “Controller” class.

Useful Links

http://www.asp.net/mvc