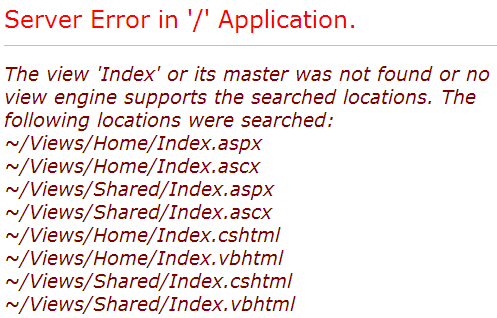
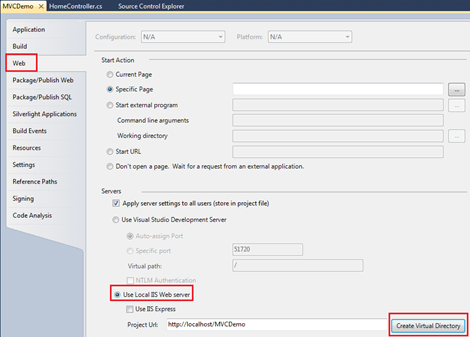
**In this video we will discuss about**  
**1.** Creating an asp.net mvc application  
**2.** Understand how mvc request is processed as apposed to webform request

**Creating an mvc application:**  
**1.** Open visual studio  
**2.** Click **File** > **New Project**  
**3.** Select **"Web"** from **"Installed Templates"** section  
**4.** Select **ASP.NET MVC 4** Web Application  
**5.** Set Name=**"MVCDemo"**  
**6.** Click OK  
**7.** Select **"Empty"** template. Select **"Razor"** as the ViewEngine. There are 2 built in view engines - Razor and ASPX. Razor is preferred by most mvc developers. We will discuss about Razor view engine in detail in a later video session.  
**8.** At this point you should have an mvc application created.

Notice that in the solution explorer, you have several folders - **Models, Views, Controllers** etc. As the names suggest these folders are going to contain Models, Views, and Controllers. We will discuss about Models, Views, and Controllers in a later video session.  
  
**Now let's add a controller to our project**  
**1.** Right Click on **"Controllers"** folder  
**2.** Select **Add** > **Controller**  
**3.** Set Controller Name = **HomeController**  
**4.** Leave rest of the defaults and click **"Add"**  
  
We should have HomeController.cs added to "Controllers" folder.   
  
At this point run the application by pressing **CTRL+F5**. Notice that you get an error as shown below.  
   
  
To fix this error, we need to add a view with name, **"Index"**. We will discuss about views in detail in a later video session. Let's fix it another way. The following is the function that is automatically added to HomeController class  
public ActionResult Index()  
{  
    return View();  
}  
  
Change the return type of Index() function from **"ActionResult"** to **"string"**, and return string **"Hello from MVC Application"** instead of View().  
public string Index()  
{  
    return "Hello from MVC Application";  
}  
  
Run the application and notice that, the string is rendered on the screen. When you run the application, by default it is using **built-in asp.net development** server. Let's use IIS, to run the application instead of the built-in asp.net development server.  
**1.** In the solution explorer, right click on the project and select **"Properties"**  
**2.** Click on **"Web"** tab  
**3.** Select **"Use Local IIS Web Server"** radio button  
**4.** Notice that the Project Url is set to **http://localhost/MVCDemo** by default  
**5.** Finally click on **"Create Virtual Directory"** button  
   
  
Run the application, and notice that the URL is **"http://localhost/MVCDemo/"**  
  
Now change the URL to **"http://localhost/MVCDemo/Home/index"**

In the URL **"Home"** is the **name of the controller** and **"Index"** is the **method** within HomeController class.   
  
So the improtant point to understand is that the **URL** is mapped to a **controller action method**. Where as in web forms application, the **URL** is mapped to a **physical file**. For example, in a web application, if we have to display the same message.  
**1.** We add a webform and in the Page\_load() event include Response.Write("Hello from ASP.NET Web Forms Application");  
**2.** We then access WebForm1.aspx as shown below  
**http://localhost/WebFormsApplication/WebForm1.aspx**  
**3.** The Page load event gets executed and the message string is displayed.