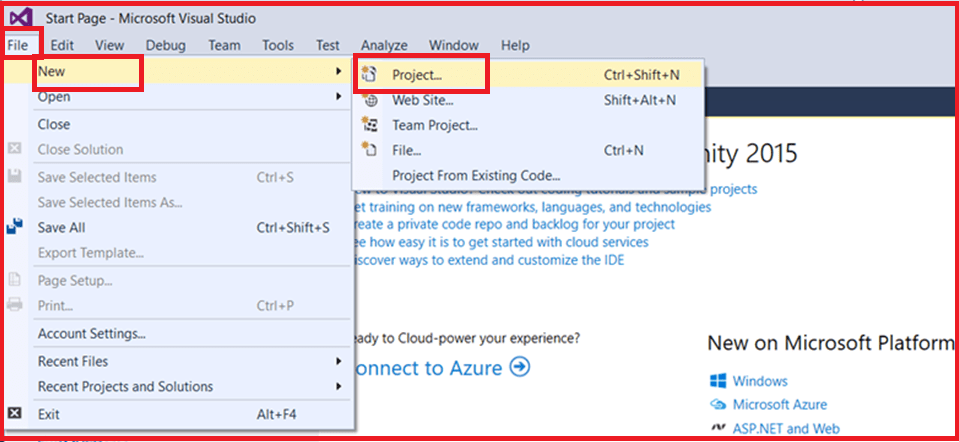
**Why we need Authorization Filter in MVC?**

By default, In ASP.NET MVC application, all the action methods of all controllers can be accessed by both authenticated and anonymous users. But If you want the action methods to be available only for the authenticated and authorized users, then you need to use the **Authorization Filter in MVC**.

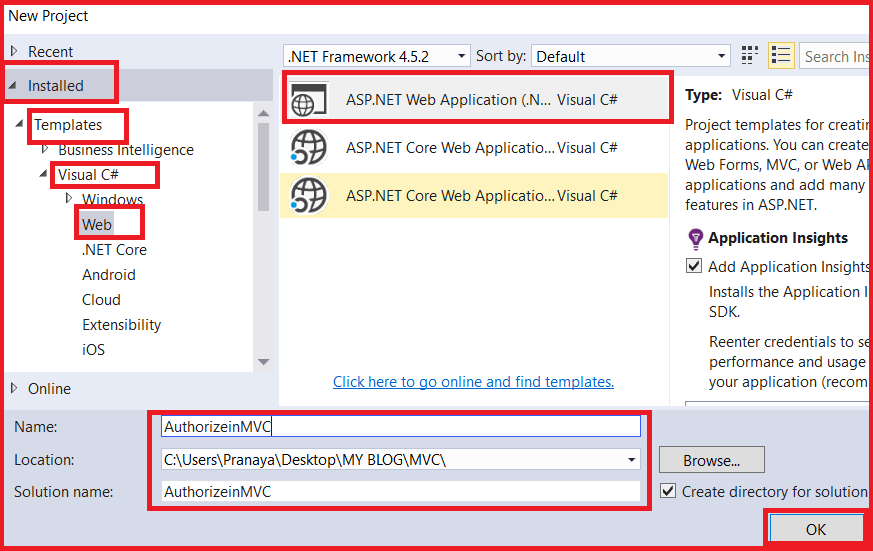
The Authorization Filter provides two built-in attributes such as Authorize and AllowAnonymous which we can use as per our business requirement.  Let us understand the **“Authorize”** and **“AllowAnonymous”** filters with an example.

**Understanding Authorization Filters in ASP.NET MVC Application:**

In order to understand the Authorization Filters, let’s create a new ASP.NET MVC Application. Open the Visual Studio in Administrator mode and then select **File => New Project** as shown in the below image

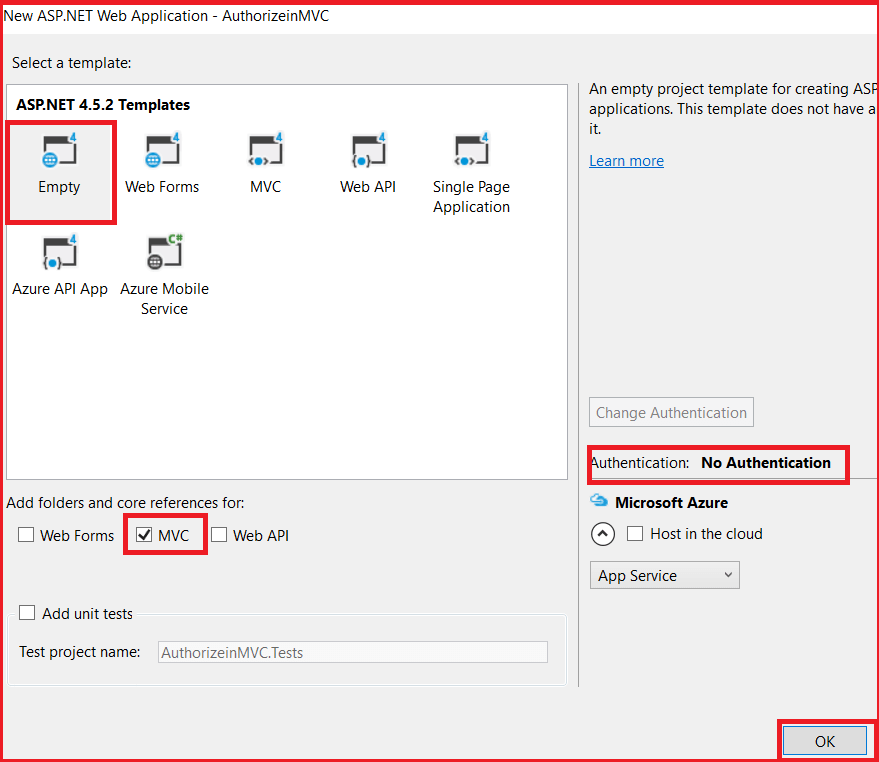


Once you click on the **Project** link the **New Project** dialogue window will open. From the **New Project** window select **Web** tab which is under the **Visual C#** tab which is again under the “**Installed – Templates**” section. From the middle pane select the **ASP.NET Web Application** and name the project as “**AuthorizeinMVC**” and then click on the “**OK**” button as shown in the below image



Once you click on the **OK** button, then a new dialogue window will open with the name **New ASP.NET Project** for selecting the **Project Templates.** From that window select the **Empty**project template as we are going to do everything from scratch. Again From the **add folder and core reference** section select the **MVC** Checkbox as we are going to create an MVC application.

Here we need to change the Authentication type for doing that just click on **Change Authentication** button. A new dialog will pop up with the name **Change Authentication** here we are going to choose **No Authentication** and then click on the **OK** button as shown in the below image.



Once you click on the OK button, then it will take some time to create the project for us.

**Creating the Home Controller:**

Right-click on the **“Controllers”** folder and add a controller with the name **HomeController**. Once you create the Home Controller then Copy and paste the following code**.**

**public** **class** HomeController : Controller

**{**

**public** ActionResult NonSecureMethod**()**

**{**

**return** View**()**;

**}**

**public** ActionResult SecureMethod**()**

**{**

**return** View**()**;

**}**

**}**

As you can see we create the above HomeController with two action methods i.e. NonSecureMethod and SecureMethod. We want the secure method to be accessed by authenticated users while the non-secure method to be accessed by anyone.

**Creating Login Controller**

Again, right-click on the **“Controllers”** folder and add a controller with the name **LoginController**. Once you create the Login controller then copy and paste the following code in it**.**

**public** **class** LoginController : Controller

**{**

**public** ActionResult Login**()**

**{**

**return** View**()**;

**}**

**}**

As you can see we create the above Login Controller with one action method i.e. Login. Whenever an unauthenticated user wants to access the secure method then we need to redirect that user to the Login action method.

**Creating NonSecureMethod View**

Right-click on the **NonSecureMethod()** and then add a view with name **NonSecureMethod**. Once you create the view then copy and paste the following code in it.

@{

ViewBag.Title = "NonSecureMethod";

}

**<h2>**This method can be accessed by everyone as it is non-secure method**</h2>**

**Creating SecureMethod View**

Similarly, right-click on SecureMethod() and add a view with name SecureMethod. Then copy and paste the following code in **SecureMethod.cshtml** view

@{

ViewBag.Title = "SecureMethod";

}

**<h2>**This method needs to be access by authorized users as it SecureMethod**</h2>**

**Creating Login View**

Similarly Right-click on the **Login()** method of **Login Controller** and add the view with name **Login.cvshtml**. Then copy and paste the following code in **Login.cshtml** view

@{

ViewBag.Title = "Login";

}

**<h2>**Login Page**</h2>**

At this point, both authenticated and anonymous users can access both the “SecureMethod” and the “NonSecureMethod” method by using the following two URLs.

**/Home/SecureMethod**

**/Home/NonSecureMethod**

If you want the **“**SecureMethod**“** to be accessed only by the authenticated and authorized users, then you need to decorate this method with the **“**Authorize**“** attribute as shown below.

**[**Authorize**]**

**public** ActionResult SecureMethod**()**

**{**

**return** View**()**;

**}**

**Changing the web.config file**

Made the following changes in the **web.config** file. What we are doing here is, if the user is an unauthorized user, then we are just navigating that user to **Login** Page. So, add the following code under the **system.web** section of the **web.config** file.

**<**authentication mode="Forms"**>**

**<**forms loginUrl="/Home/Login"**><**/forms**>**

**<**/authentication**>**

That’s it. Now run the application and navigate to /**Home/SecureMethod**. Then you will see that it will redirect you to the Login page. On the other hand, you can access the NonSecure method. Now remove the **Authorize** attribute from the **SecureMethod**of **HomeController.**Then apply the **Authorize**attribute at the controller level as shown below.

[Authorize]

public class HomeController : Controller

{

public ActionResult NonSecureMethod()

{

return View();

}

public ActionResult SecureMethod()

{

return View();

}

}

When you apply the Authorize attribute at the controller level then it is applicable to all the action methods that are present within that controller.  Here all the action methods of Home Controller are now protected with the Authorize Attribute, So, now only the authenticated users can access both SecureMethod() and NonSecureMethod().

**How to use the AllowAnonymous Attribute in MVC?**

If you want to allow anonymous access to the NonSecureMethod of Home controller, then you need to decorate the AllowAnonymous attribute to that NonSecureMethod method as shown below. The AllowAnonymous attribute in MVC is used to skip the authorization which is enforced by Authorization Filter in MVC.

**[**Authorize**]**

**public** **class** HomeController : Controller

**{**

**[**AllowAnonymous**]**

**public** ActionResult NonSecureMethod**()**

**{**

**return** View**()**;

**}**

**public** ActionResult SecureMethod**()**

**{**

**return** View**()**;

**}**

**}**

Now, run the application and navigate to **/Home/NonSecureMethod** and you will see that it display the page as expected and when you navigate to **/Home/SecureMethod** then it will redirect you to the Login page.