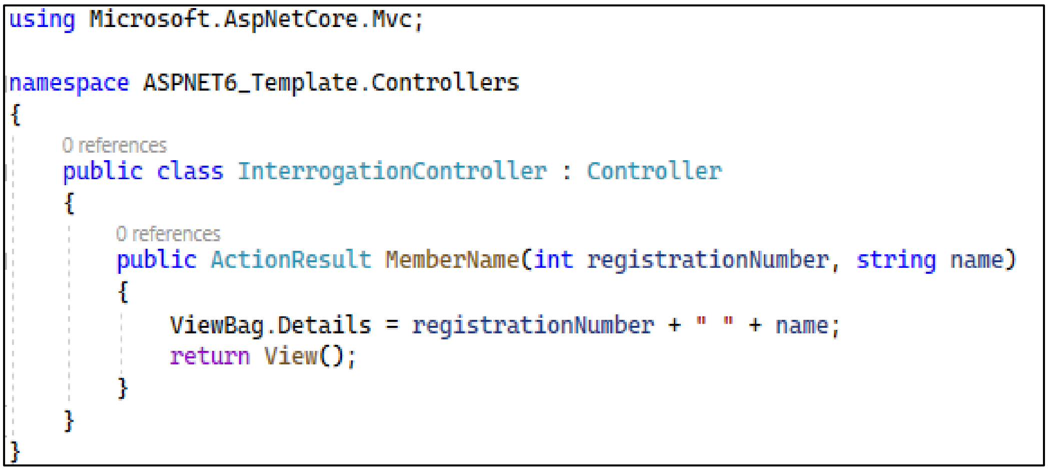
Scenario:

The requirement is to create an online interrogation panel application using the Model-View-Controller (MVC) architecture that allows to display the user name, get the problem and display the solution for that problems. Use ViewBag, ViewData and TempData concepts to achieve it.

Exercise Steps:

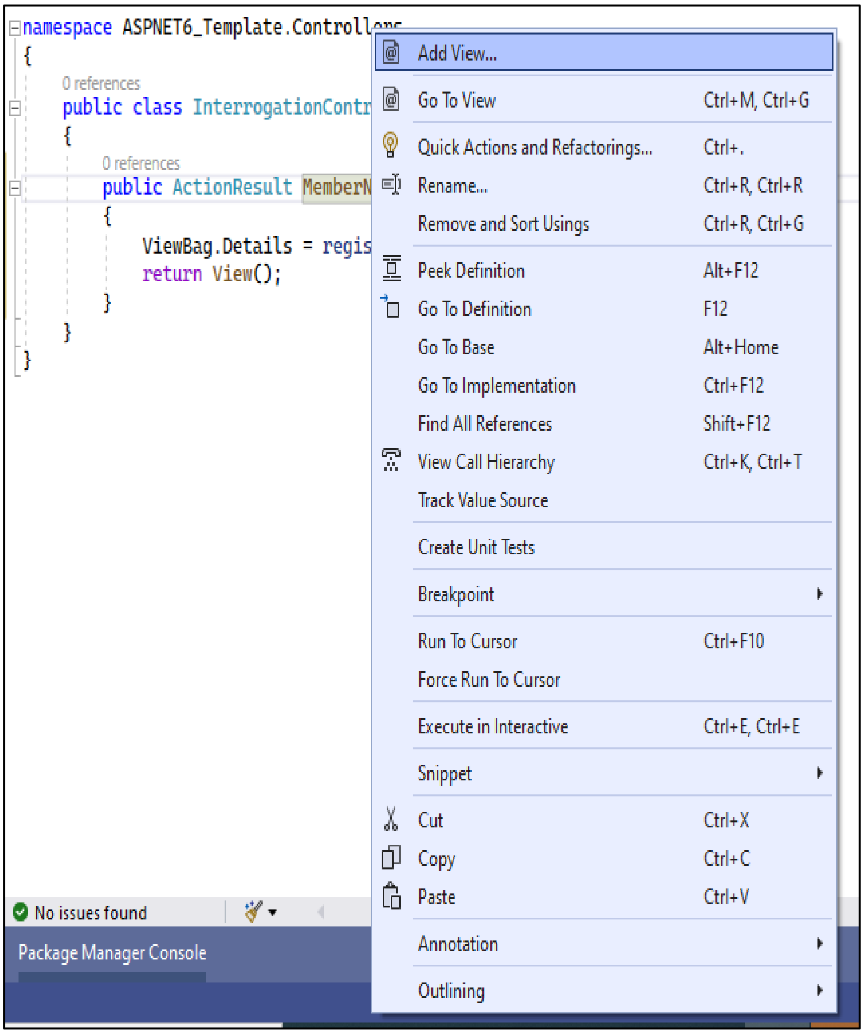
1. Create a new ASP.NET Core MVC project in Visual Studio.
2. Add a new controller called **"InterrogationController"** and create an **"MemberName"** action method with arguments that stores the registration number and name of the member in a **ViewBag** returns a view. Add one space in-between the registration numberand name while storing the values in the ViewBag.



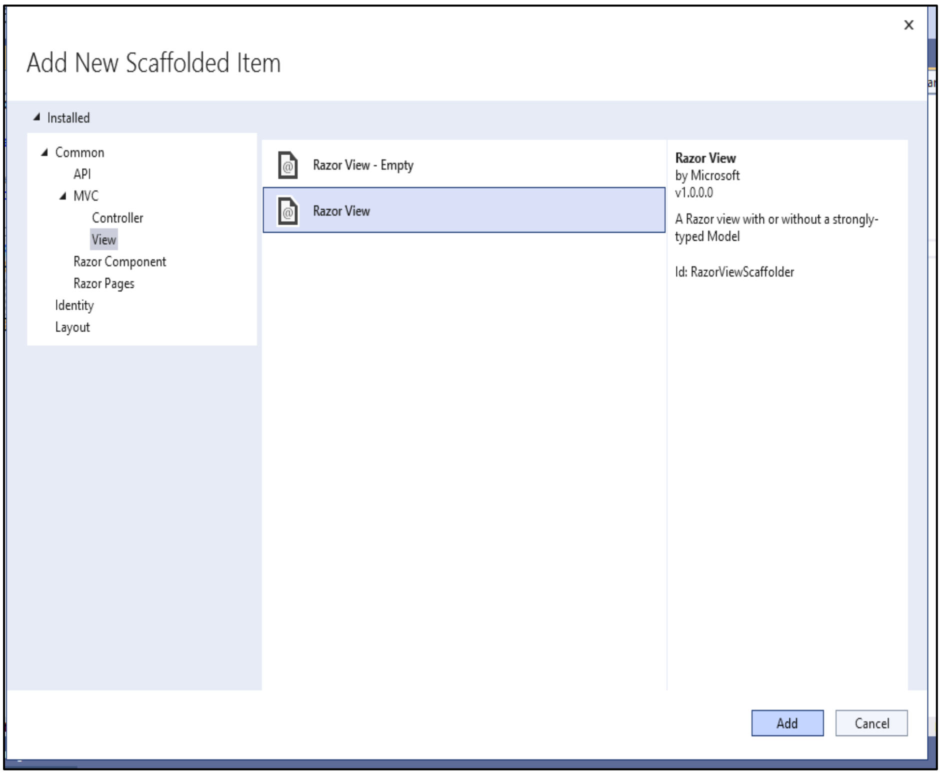
```
using Microsoft.AspNetCore.Mvc;

namespace ASPNET6_Template.Controllers
{
    0 references
    public class InterrogationController : Controller
    {
        0 references
        public ActionResult MemberName(int registrationNumber, string name)
        {
            ViewBag.Details = registrationNumber + " " + name;
            return View();
        }
    }
}
```

3. In **"InterrogationController"**, hover the cursor over the MemberName action method and click the right button and choose **Add View** , Click the **Razor View**, then name it as **MemberName.cshtml**. Please refer the below image for creating views.



```
namespace ASPNET6_Template.Controllers
{
    0 references
    public class InterrogationContr
    {
        0 references
        public ActionResult MemberN
        {
            ViewBag.Details = regis
            return View();
        }
    }
}
```

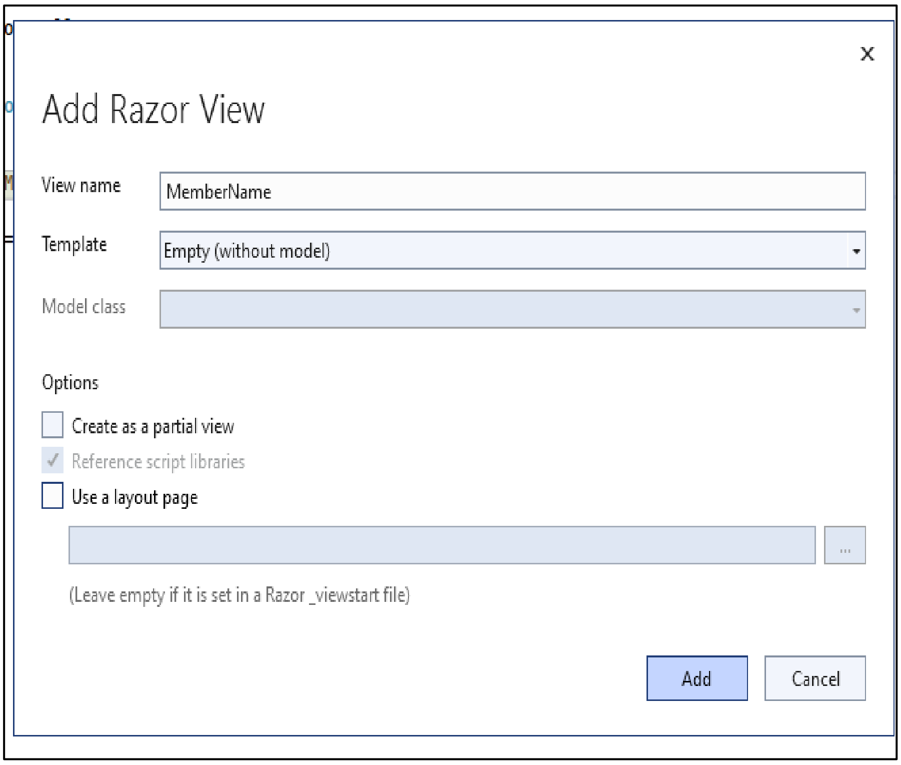


Add New Scaffolded Item

Razor View
by Microsoft
v1.0.0.0

A Razor view with or without a strongly-typed Model

Id: RazorViewScaffolder



Add Razor View

View name: MemberName

Template: Empty (without model)

Model class:

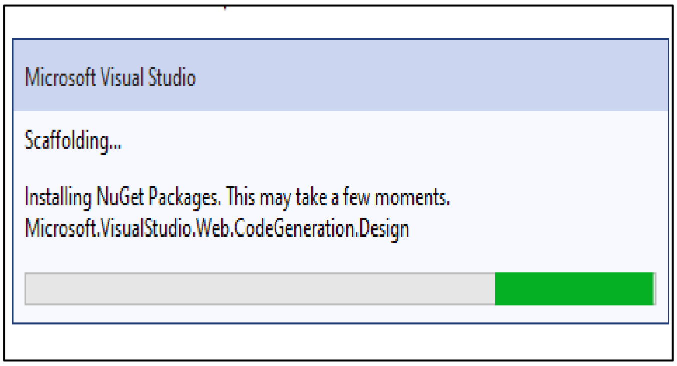
Options

☐ Create as a partial view

☑ Reference script libraries

☐ Use a layout page

(Leave empty if it is set in a Razor _viewstart file)

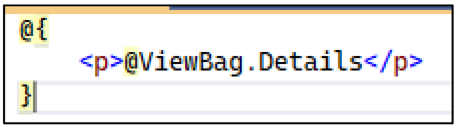


Microsoft Visual Studio

Scaffolding...

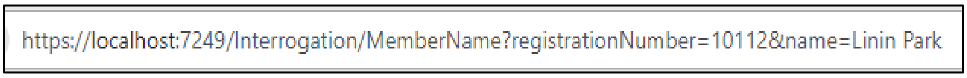
Installing NuGet Packages. This may take a few moments.
Microsoft.VisualStudio.Web.CodeGeneration.Design

4. In the **"MemberName.cshtml"** view, if any code appears first, then remove that and add the needed HTML code to display the stored ViewBag's Value. You can use an HTML anchor tag. When the user navigate the link, it will display the value which is stored in the ViewBag



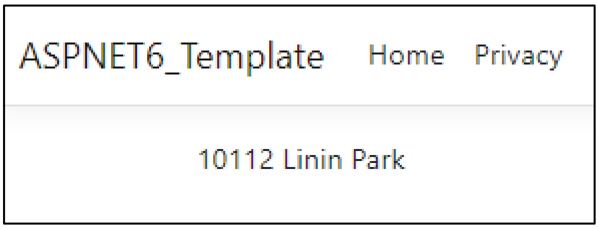
```
@{
    <p>@ViewBag.Details</p>
}
```

5. Run the application and navigate to the **"MemberName"** page, After the localhost and port number, use the below-given url format to enter the input for MemberName page.



https://localhost:7249/Interrogation/MemberName?registrationNumber=10112&name=Linin Park

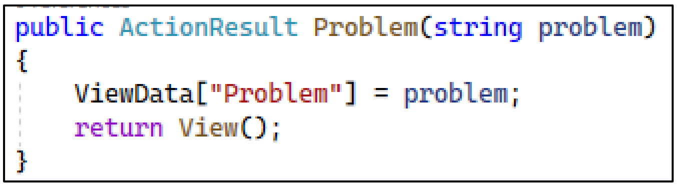
6. Let’s see the outcome of the above instructions



10112 Linin Park

7. We have successfully created a view named as **“MemberName.cshtml”.**

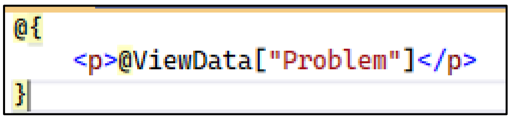
8. In **"InterrogationController"**, Create a **"Problem"** action method with argument that store the problem of the online introgator in a **ViewData** returns a view.



```
public ActionResult Problem(string problem)
{
    ViewData["Problem"] = problem;
    return View();
}
```

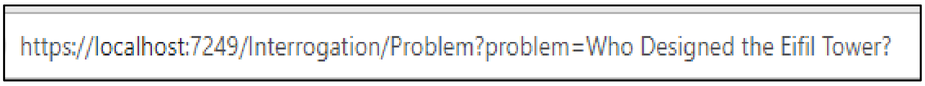
9. In **"InterrogationController"**, hover the cursor over the **Problem** action method and click the right button and choose **Add View** and name it as **Problem.cshtml. (Refer step 3 for creating a view)**

10. In the **"Problem.cshtml”** view, add HTML code to display the stored ViewData’s Value. You can use an HTML anchor tag. When the user navigate the link, it will display the question which is stored in the ViewData



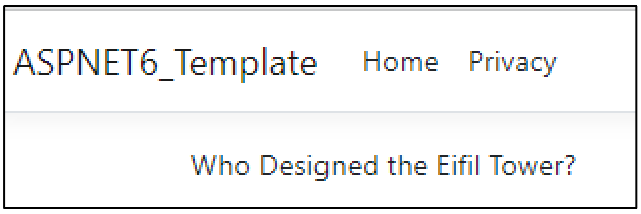
```
@{
    <p>@ViewData["Problem"]</p>
}
```

11. Run the application and navigate to the **“Problem"** page, After the localhost and port number, use the below-given url format to enter the input for **Problem** page.



```
https://localhost:7249/Interrogation/Problem?problem=Who Designed the Eifil Tower?
```

12. Let’s see the outcome of the above instructions



ASPNET6_Template

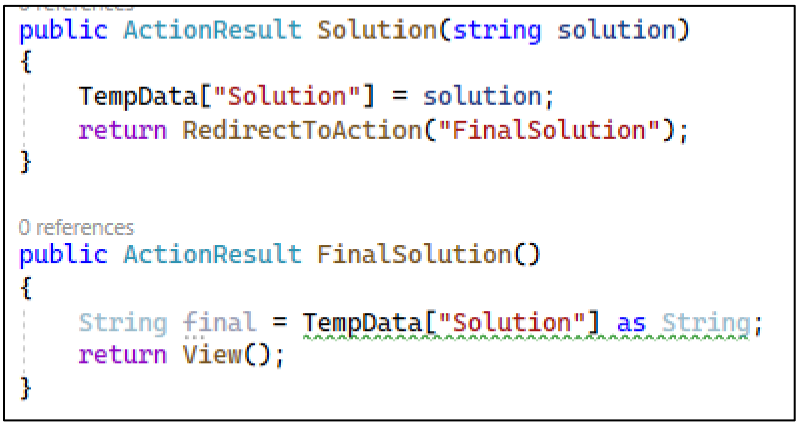
Who Designed the Eifil Tower?

13. We have successfully created a view named as **“Problem.cshtml”**

14. In **"InterrogationController"** Create a **"Solution"** action method with argument that store the solution for the problem in a **TempData**. Create another action method **“FinalSolution”** without any arguments.

15. In **Solution** method after storing the solution in TempData, then redirect to FinalSolution action method.

16. In FinalSolution method store the same TempData value as string and return that view.



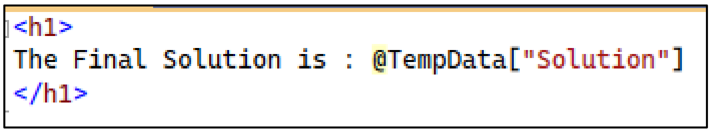
```
public ActionResult Solution(string solution)
{
    TempData["Solution"] = solution;
    return RedirectToAction("FinalSolution");
}

0 references
public ActionResult FinalSolution()
{
    String final = TempData["Solution"] as String;
    return View();
}
```

17. In **"InterrogationController"**, hover the cursor over the **FinalSolution** action method and click the right button and choose **Add**

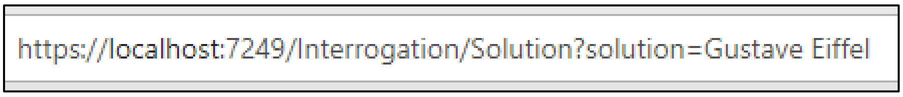
View and name it as **FinalSolution.cshtml. (Refer step 3 for creating a view)**

18. In the **"FinalSolution.cshtml"** view, add HTML code to display the stored TempData's Value. You can use an h1 tag. When the user navigate the link, it will display the question which is stored in the **TempData**



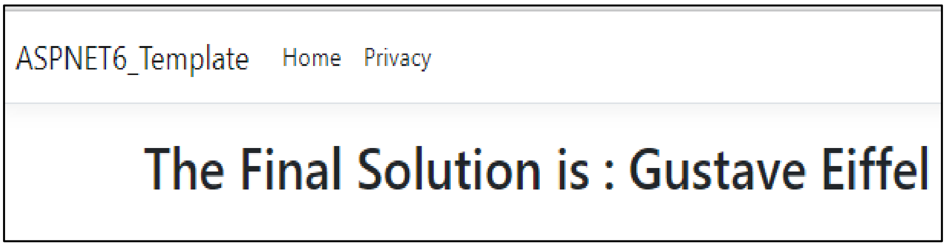
```
<h1>
The Final Solution is : @TempData["Solution"]
</h1>
```

19. Run the application and navigate to the **"Solution"** page, use the below-given url format to enter the input for **Solution** page. It should navigate and display the output in **FinalSolution** page



https://localhost:7249/Interrogation/Solution?solution=Gustave Eiffel

20. Let's see the outcome of the above instructions



The Final Solution is : Gustave Eiffel

21. We have successfully created a view named as **"FinalSolution.cshtml".**

To summarize,

This exercise demonstrates how ViewBag, ViewData and TempData can be used to reuse code across multiple views in an ASP.NET Core application. By creating these we can use this from a conroller to view.