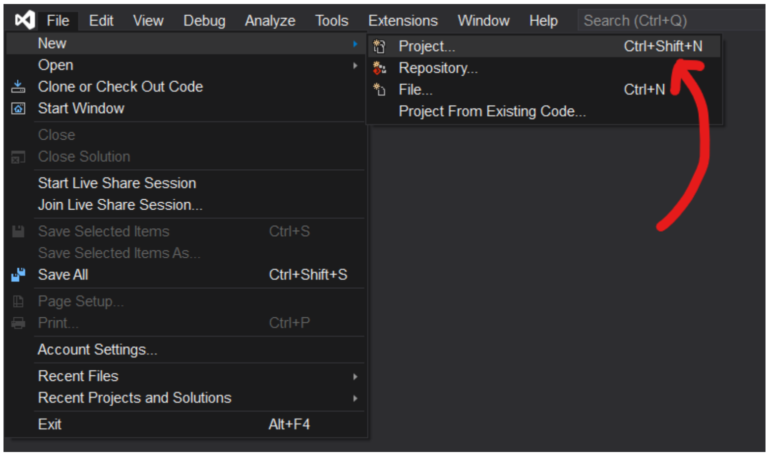
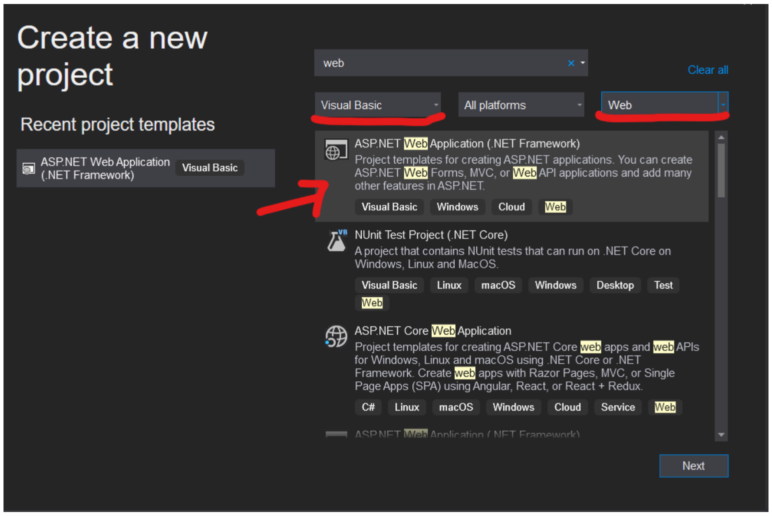
Build My Last MVC vb Project

File > New > Project



Find "web" > Visual Basic & Web > APS.NET Web Application > **Next**



web

Visual Basic | All platforms | Web

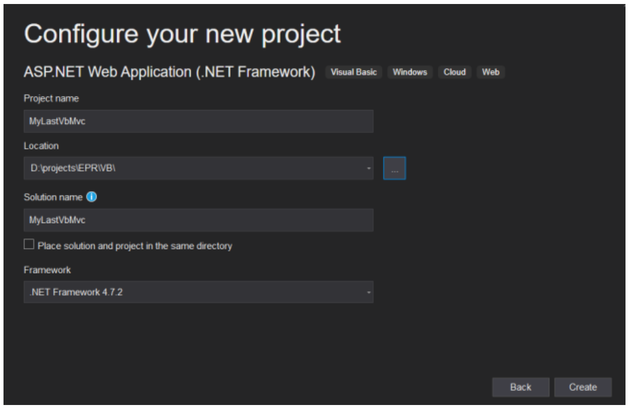
ASP.NET Web Application (.NET Framework)
Project templates for creating ASP.NET applications. You can create ASP.NET Web Forms, MVC, or Web API applications and add many other features in ASP.NET.
Visual Basic Windows Cloud Web

NUnit Test Project (.NET Core)
A project that contains NUnit tests that can run on .NET Core on Windows, Linux and MacOS.
Visual Basic Linux macOS Windows Desktop Test Web

ASP.NET Core Web Application
Project templates for creating ASP.NET Core web apps and web APIs for Windows, Linux and macOS using .NET Core or .NET Framework. Create web apps with Razor Pages, MVC, or Single Page Apps (SPA) using Angular, React, or React + Redux.
C# Linux macOS Windows Cloud Service Web

ASP.NET Web Application (.NET Framework)

Project Name, Location, Solution Name > **Create**


Configure your new project
ASP.NET Web Application (.NET Framework)
Visual Basic
Windows
Cloud
Web
...
Back
Create

Project name
MyLastVbMvc

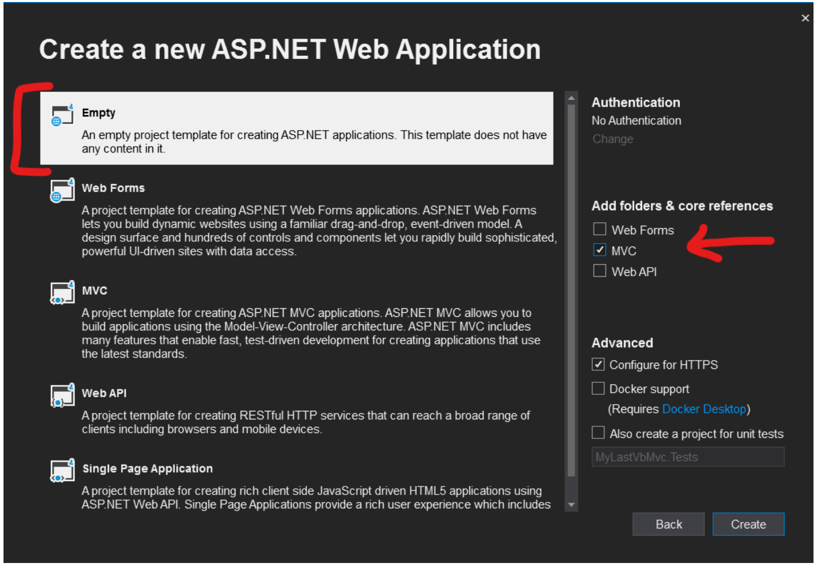
Location
D:\projects\EPR\VB\

Solution name
MyLastVbMvc

☐ Place solution and project in the same directory

Framework
.NET Framework 4.7.2

Mark Empty > check MVC > **Create**



Create a new ASP.NET Web Application

Empty
An empty project template for creating ASP.NET applications. This template does not have any content in it.

Web Forms
A project template for creating ASP.NET Web Forms applications. ASP.NET Web Forms lets you build dynamic websites using a familiar drag-and-drop, event-driven model. A design surface and hundreds of controls and components let you rapidly build sophisticated, powerful UI-driven sites with data access.

MVC
A project template for creating ASP.NET MVC applications. ASP.NET MVC allows you to build applications using the Model-View-Controller architecture. ASP.NET MVC includes many features that enable fast, test-driven development for creating applications that use the latest standards.

Web API
A project template for creating RESTful HTTP services that can reach a broad range of clients including browsers and mobile devices.

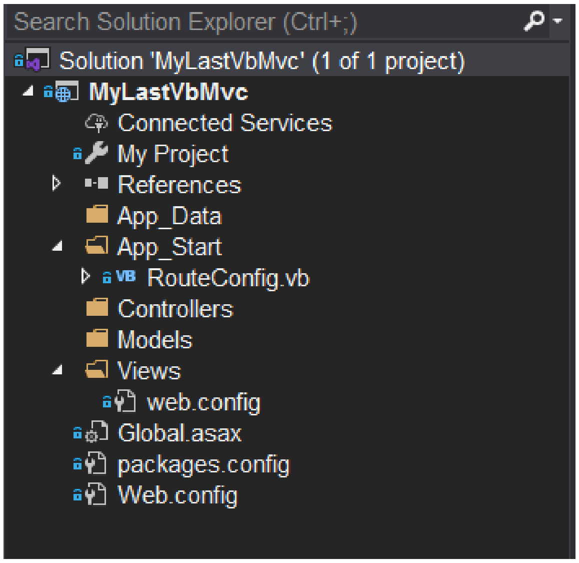
Single Page Application
A project template for creating rich client side JavaScript driven HTML5 applications using ASP.NET Web API. Single Page Applications provide a rich user experience which includes

Authentication
No Authentication
Change

Add folders & core references
□ Web Forms
☑ MVC
□ Web API

Advanced
☑ Configure for HTTPS
□ Docker support
(Requires Docker Desktop)
□ Also create a project for unit tests
MyLastVbMvc.Tests

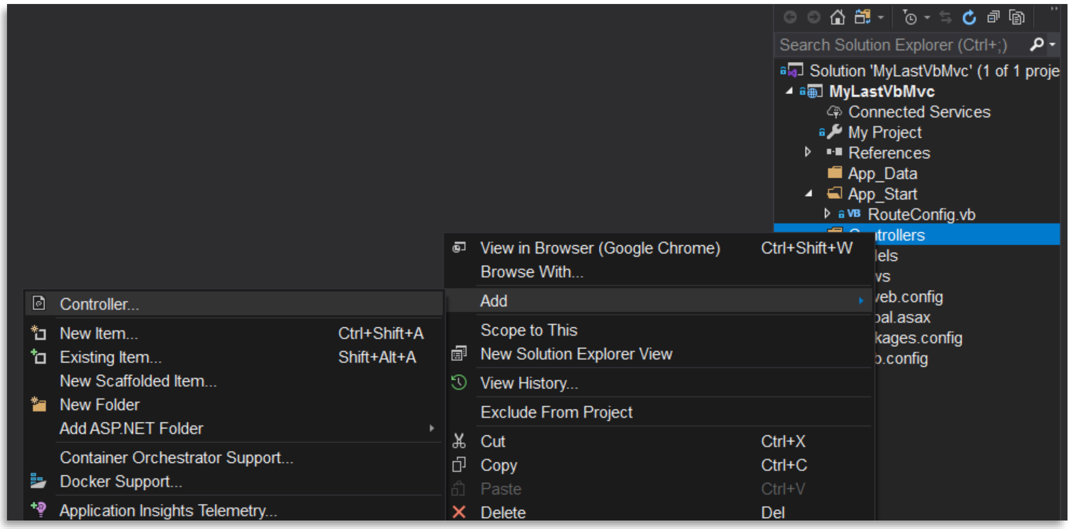
Our Empty Solution:



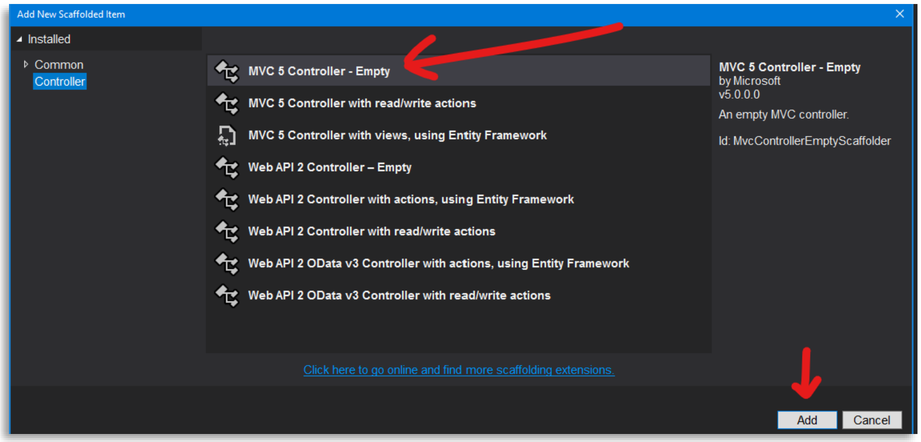
```
Solution 'MyLastVbMvc' (1 of 1 project)
  MyLastVbMvc
      Connected Services
      My Project
      References
      App_Data
      App_Start
          RouteConfig.vb
      Controllers
      Models
      Views
          web.config
      Global.asax
      packages.config
      Web.config
```

Now, Let’s make it a MVC Project!!
First step is Controller:

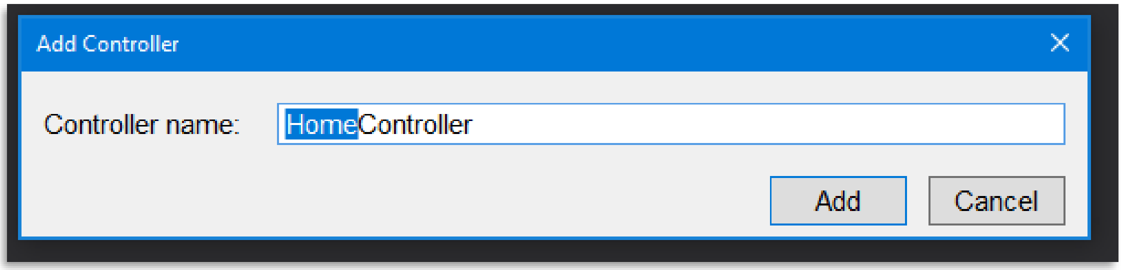
Right click on Controllers > Add > Controller



Mark Empty

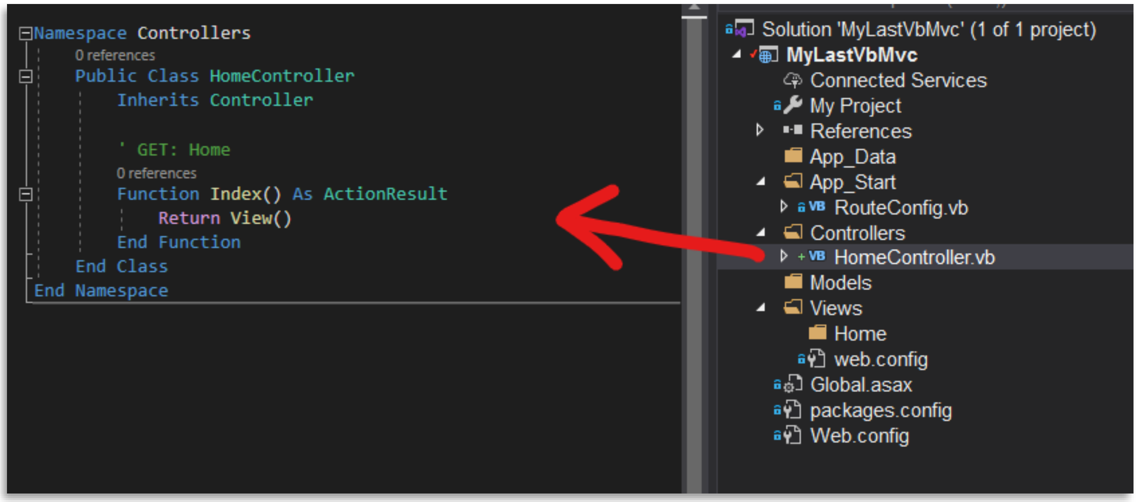

Add New Scaffolded Item
Installed
Common
Controller
MVC 5 Controller - Empty
MVC 5 Controller with read/write actions
MVC 5 Controller with views, using Entity Framework
Web API 2 Controller – Empty
Web API 2 Controller with actions, using Entity Framework
Web API 2 Controller with read/write actions
Web API 2 OData v3 Controller with actions, using Entity Framework
Web API 2 OData v3 Controller with read/write actions
MVC 5 Controller - Empty
by Microsoft
v5.0.0.0
An empty MVC controller.
Id: MvcControllerEmptyScaffolder
Click here to go online and find more scaffolding extensions.
Add
Cancel

Give a Name to the controller but keep the suffix Controller:



Controller name: HomeController

Congratulation! We just create our First Controller!



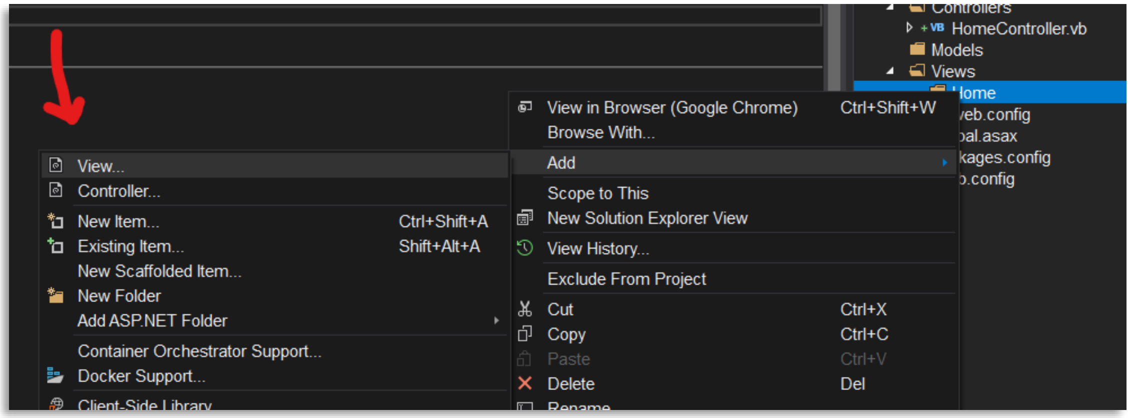
```
Namespace Controllers
    0 references
    Public Class HomeController
        Inherits Controller

        ' GET: Home
        0 references
        Function Index() As ActionResult
            Return View()
        End Function
    End Class
End Namespace
```

We can see that our controller is a class with special name `HomeController` and `Inherits Controller`. By default we get **Index Action.**

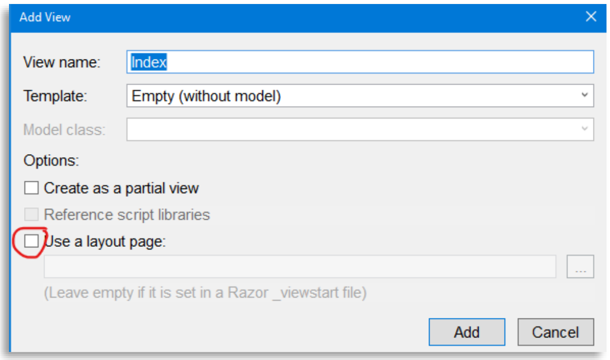
```
Public Class HomeController
      Inherits Controller

      ' GET: Home
      Function Index() As ActionResult
          Return View()
      End Function
  End Class
```

Action – The Controller's method that we want to expose to the client.
`Return View()` have 7+ overloads. If its empty, it will try to find a View with the same name as our action under the `Views\<controller_name>` in our case is `"Home"`.
`Return View("Home")` will return the view with the specific name we choose.

Right click on `Views\Home` > Add > View



View Name : Index > unchecked User a layout page > Click **Add**



View name: Index

Template: Empty (without model)

Model class:

Options:

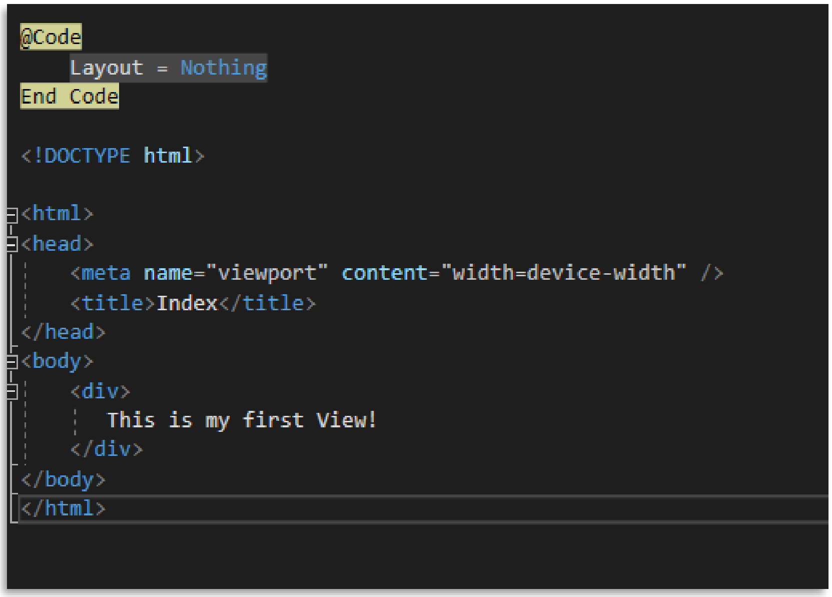
☐ Create as a partial view

☐ Reference script libraries

☐ Use a layout page:

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

The new file is Index.vbhtml witch is combination of VB & Html:



```
@Code
    Layout = Nothing
End Code

<!DOCTYPE html>

<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>Index</title>
</head>
<body>
    <div>
        This is my first View!
    </div>
</body>
</html>
```

Now Press `Ctrl + F5` for build & host our app to an iis express server. Change the Html and see the changes in the browser:

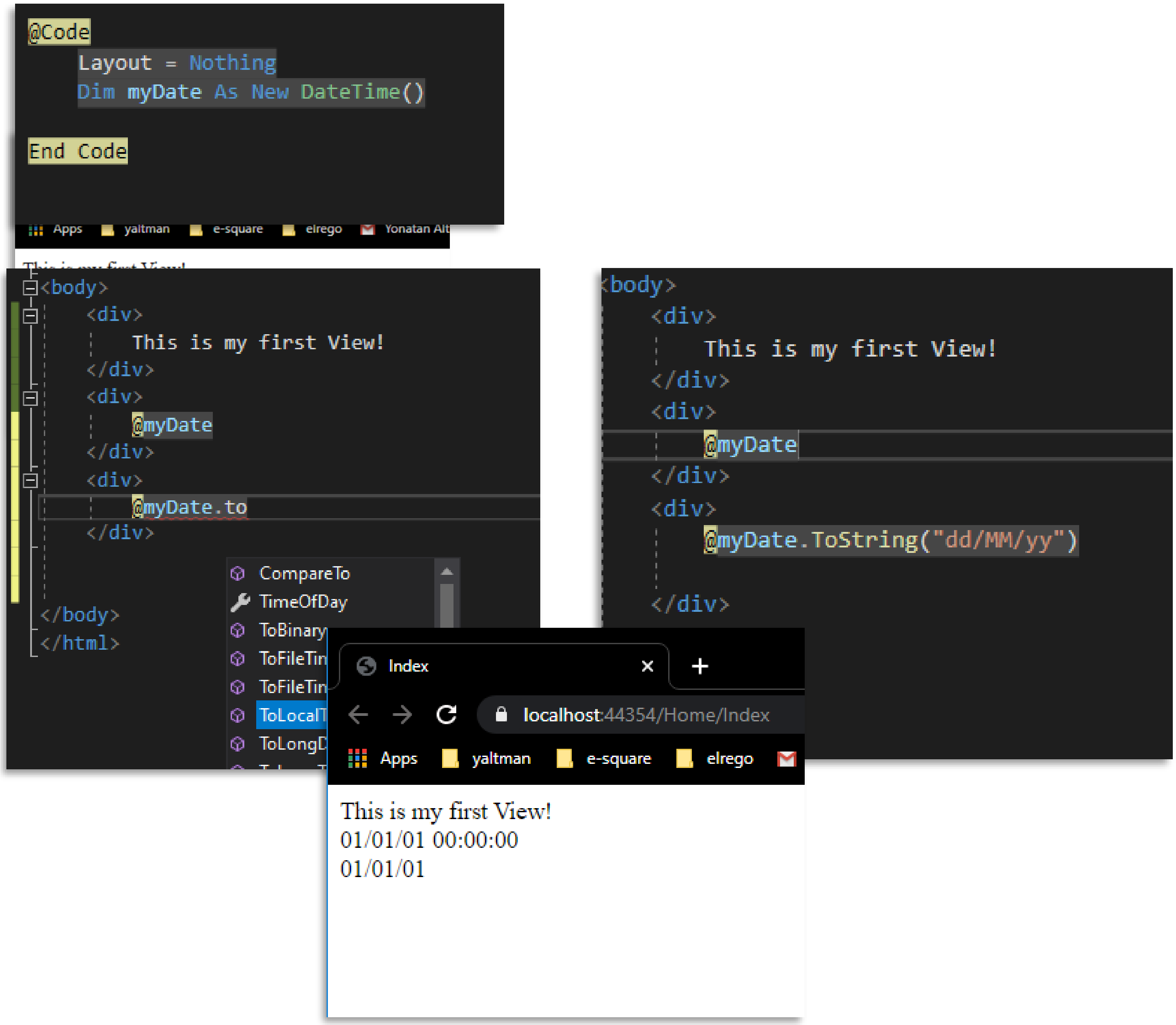
Passing Data From Controller To View:

1. Razor
2. ViewBag
3. ViewData
4. TempData

1. Razor

The razor used by the @ sign:

Inside the @Code – End Code its like writing code in a .vb file, and the we can access the data with @ and the name of the data:



```
@Code
    Layout = Nothing
    Dim myDate As New DateTime()

End Code
```

```
<body>
    <div>
        This is my first View!
    </div>
    <div>
        @myDate
    </div>
    <div>
        @myDate.to
    </div>

</body>
</html>
```

```
<body>
    <div>
        This is my first View!
    </div>
    <div>
        @myDate
    </div>
    <div>
        @myDate.ToString("dd/MM/yy")

    </div>
```

```
This is my first View!
01/01/01 00:00:00
01/01/01
```

2. ViewBag

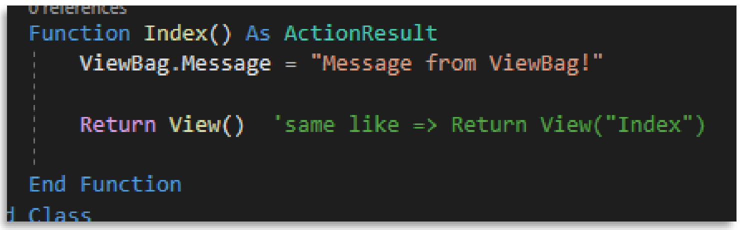
ViewBag can be useful when you want to transfer temporary data from the controller to the view. The viewBag is a dynamic type property of ControllerBase class which is the base class of all the controllers.

In the Action Index:

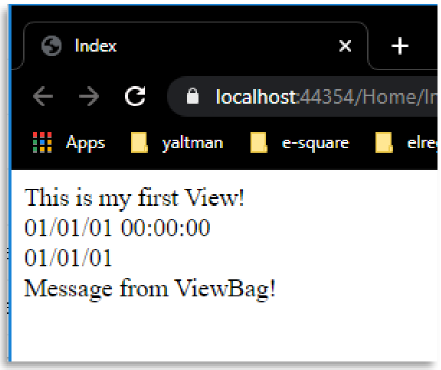
```
ViewBag.Message = "Message from ViewBag!"
```

In the View:

```
<div>
   @ViewBag.Message
</div>
```



```
0 references
Function Index() As ActionResult
    ViewBag.Message = "Message from ViewBag!"

    Return View()  'same like => Return View("Index")

End Function
d Class
```



This is my first View!
01/01/01 00:00:00
01/01/01
Message from ViewBag!

3. ViewData

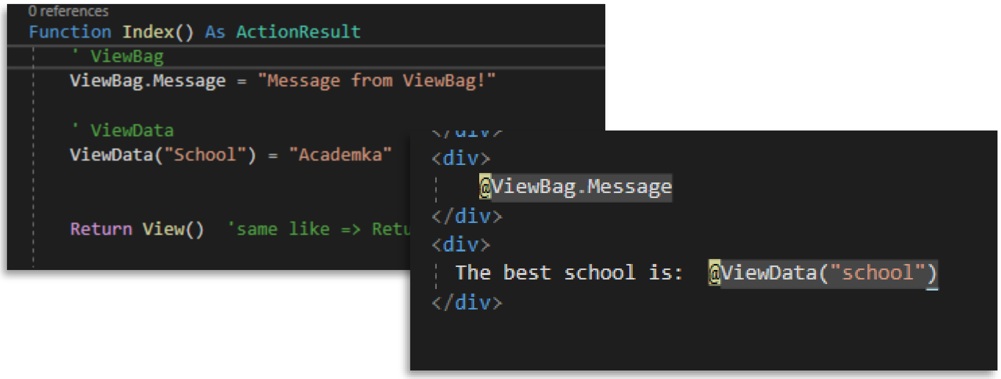
ViewData is a dictionary which can contain key-value pairs where each key must be string.

In the Action Index:

```
ViewData("School") = "Academka"
```

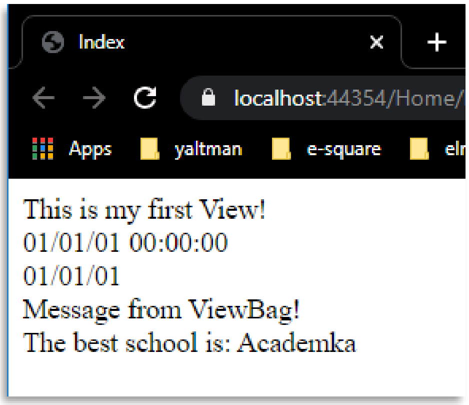
In the View:

```
<div>
   The best school is:  @ViewData("school")
</div>
```



```
0 references
Function Index() As ActionResult
    ' ViewBag
    ViewBag.Message = "Message from ViewBag!"

    ' ViewData
    ViewData("School") = "Academka"

    Return View()  'same like => Retu
```

```
</div>
<div>
    @ViewBag.Message
</div>
<div>
  The best school is:  @ViewData("school")
</div>
```



This is my first View!
01/01/01 00:00:00
01/01/01
Message from ViewBag!
The best school is: Academka