**Tutorial: Get started with C# and ASP.NET Core in Visual Studio**

* Article
* 11/17/2023
* 15 contributors

Feedback

**In this article**

1. [Prerequisites](https://learn.microsoft.com/en-us/visualstudio/get-started/csharp/tutorial-aspnet-core?view=vs-2022#prerequisites)
2. [Create a project](https://learn.microsoft.com/en-us/visualstudio/get-started/csharp/tutorial-aspnet-core?view=vs-2022#create-a-project)
3. [Tour your solution](https://learn.microsoft.com/en-us/visualstudio/get-started/csharp/tutorial-aspnet-core?view=vs-2022#tour-your-solution-1)
4. [Run, debug, and make changes](https://learn.microsoft.com/en-us/visualstudio/get-started/csharp/tutorial-aspnet-core?view=vs-2022#run-debug-and-make-changes-1)

Show 3 more

In this tutorial for C# development with ASP.NET Core, you create a C# ASP.NET Core web app in Visual Studio.

This tutorial shows you how to:

* Create a Visual Studio project
* Create a C# ASP.NET Core web app
* Make changes to the web app
* Explore IDE features
* Run the web app

**Prerequisites**

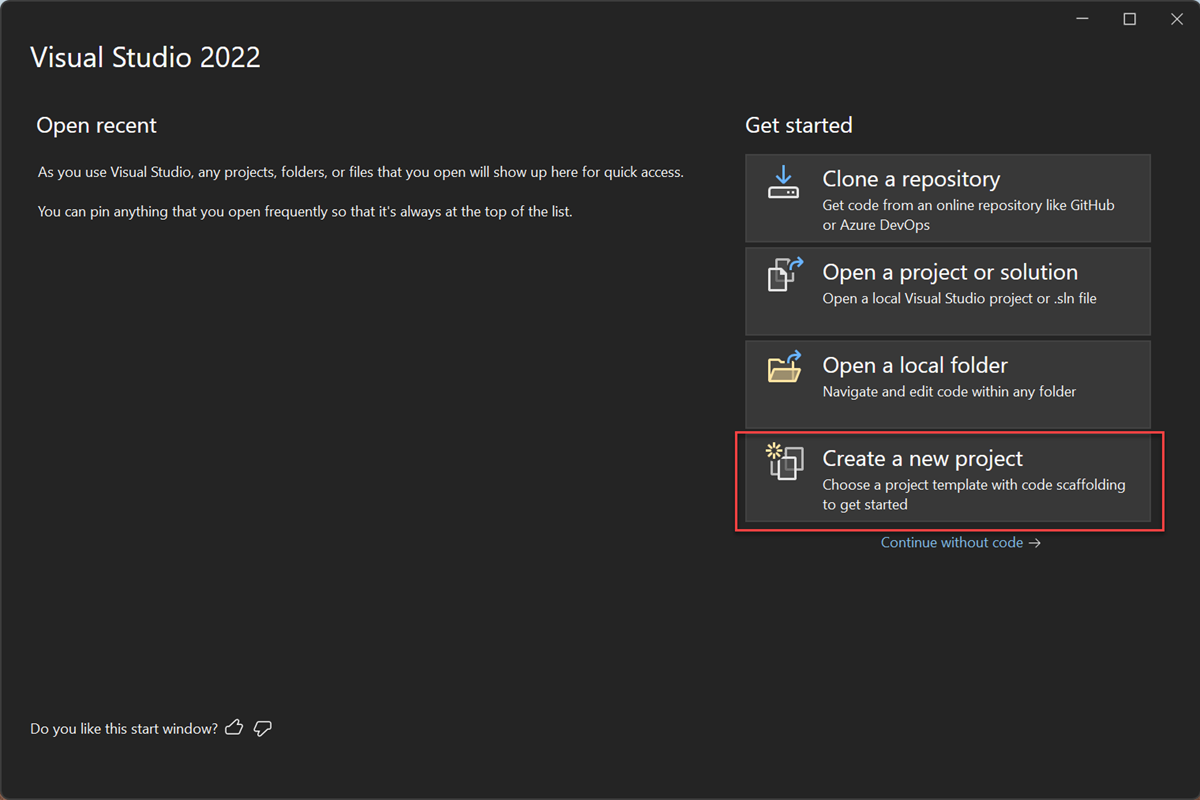
You need Visual Studio to complete this tutorial. Visit the [Visual Studio downloads page](https://visualstudio.microsoft.com/downloads/?cid=learn-onpage-download-tutorial-create-csharp-aspnetcore-web-app-page-cta) for a free version.

* For more information about upgrading to the latest Visual Studio release, see [Visual Studio updates](https://learn.microsoft.com/en-us/visualstudio/install/update-visual-studio?view=vs-2022).
* To customize your Visual Studio experience, see [personalize the Visual Studio IDE and Editor](https://learn.microsoft.com/en-us/visualstudio/ide/personalizing-the-visual-studio-ide?view=vs-2022).

**Create a project**

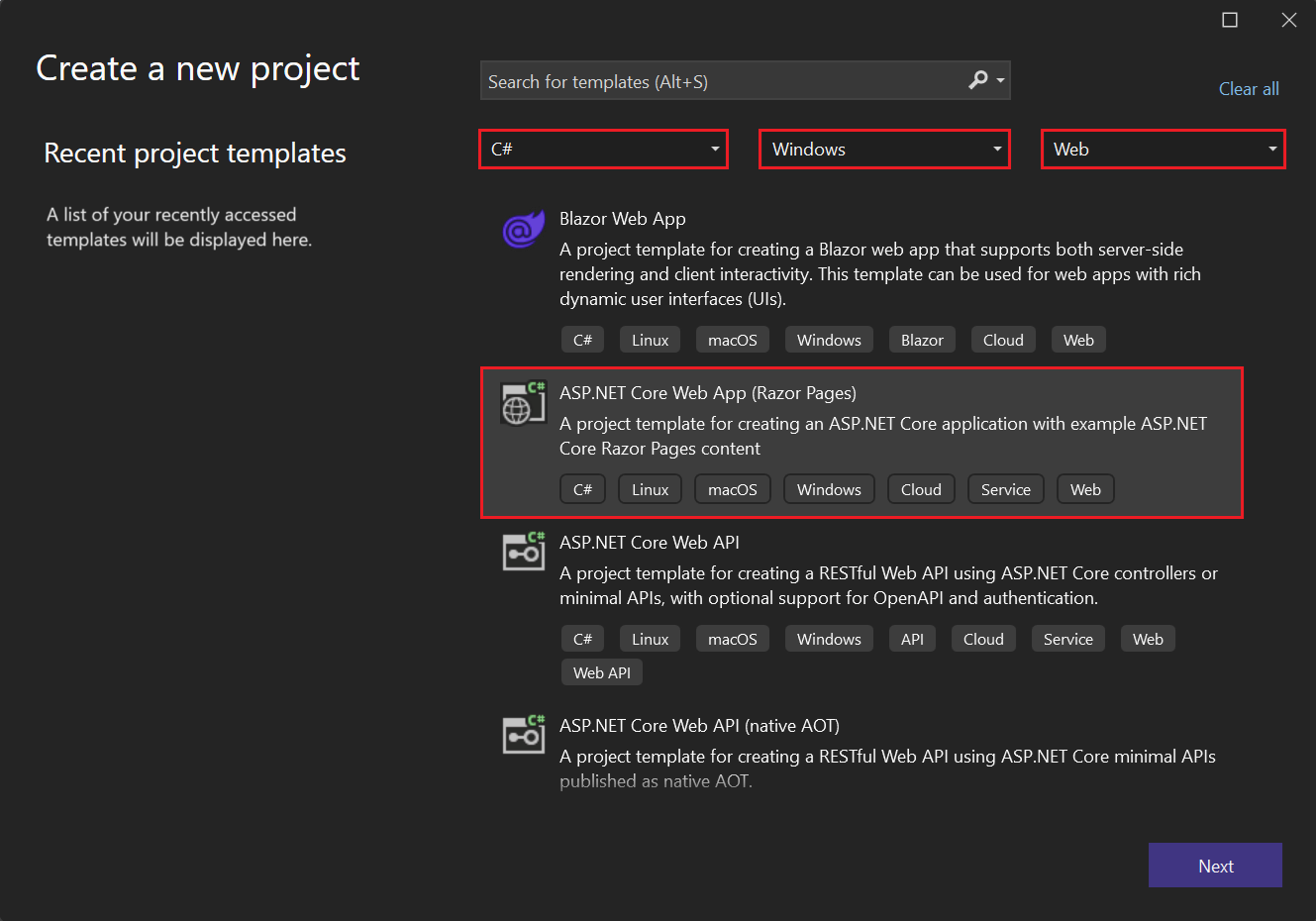
First, you create an ASP.NET Core project. The project type comes with all the template files you need to build a fully functional website.

1. On the start window, select **Create a new project**.



1. In the **Create a new project** window, select **C#** from the Language list. Next, select **Windows** from the platform list, and **Web** from the project types list.

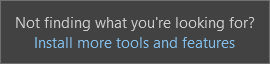
After you apply the language, platform, and project type filters, select the **ASP.NET Core Web App** template, and then select **Next**.

[](https://learn.microsoft.com/en-us/visualstudio/get-started/csharp/media/vs-2022/csharp-create-new-project-aspnet-core.png?view=vs-2022#lightbox)

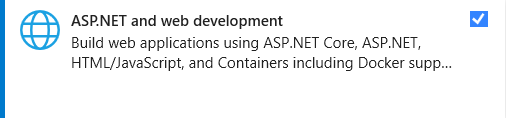
**Note**

If you don't see the **ASP.NET Core Web App** template, you can install it from the **Create a new project** window.

In the **Not finding what you're looking for?** message at the bottom of the list of templates, select the **Install more tools and features** link.



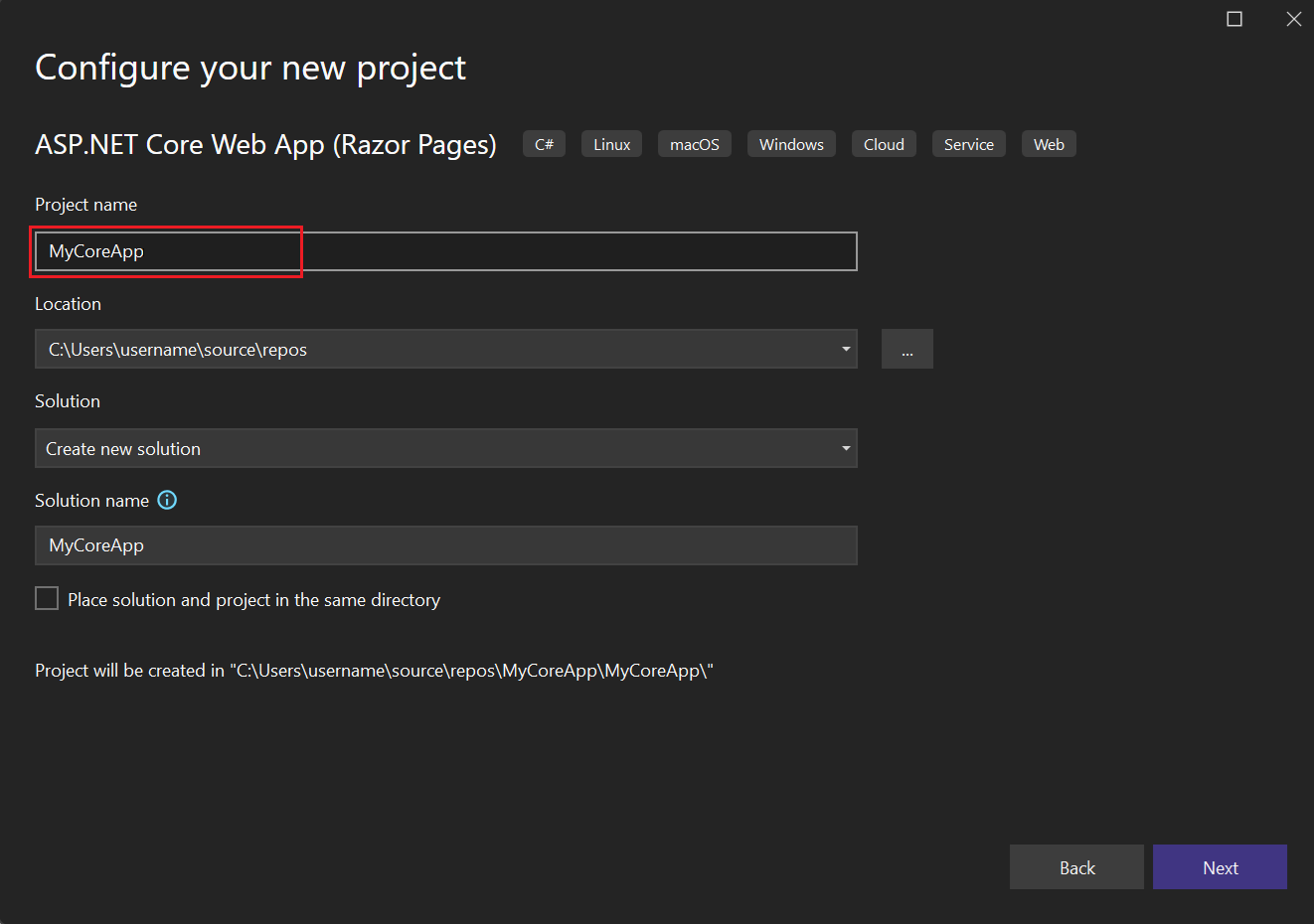
In the Visual Studio Installer, select the **ASP.NET and web development** workload.



Select **Modify** in the Visual Studio Installer. You might be prompted to save your work. Select **Continue** to install the workload.

Return to step 2 in this "[**Create a project**](https://learn.microsoft.com/en-us/visualstudio/get-started/csharp/tutorial-aspnet-core?view=vs-2022#create-a-project)" procedure.

1. In the **Configure your new project** window, enter **MyCoreApp** in the **Project name** field. Then, select **Next**.

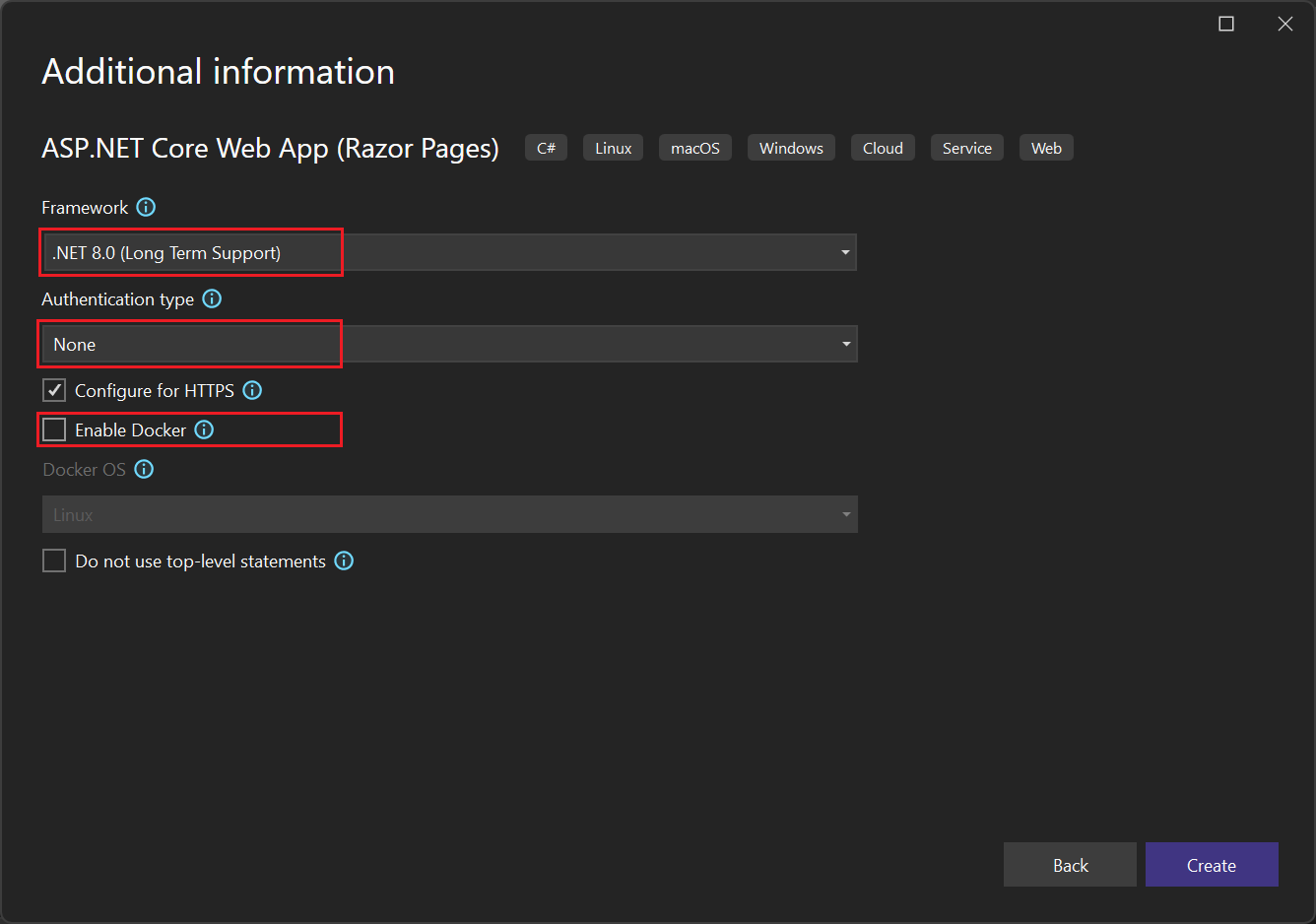
[](https://learn.microsoft.com/en-us/visualstudio/get-started/csharp/media/vs-2022/csharp-name-your-aspnet-app.png?view=vs-2022#lightbox)

1. In the **Additional information** window, verify that **.NET 8.0** appears in the **Target Framework** field.

From this window, you can enable Docker support and add authentication support. The drop-down menu for **Authentication Type** has the following four options:

* + **None**: No authentication.
  + **Individual accounts**: These authentications are stored in a local or Azure-based database.
  + **Microsoft identity platform**: This option uses Microsoft Entra ID or Microsoft 365 for authentication.
  + **Windows**: Suitable for intranet applications.

Leave the **Enable Docker** box unchecked, and select **None** for Authentication Type.

[](https://learn.microsoft.com/en-us/visualstudio/get-started/csharp/media/vs-2022/aspnet-core-additional-information.png?view=vs-2022#lightbox)

1. Select **Create**.

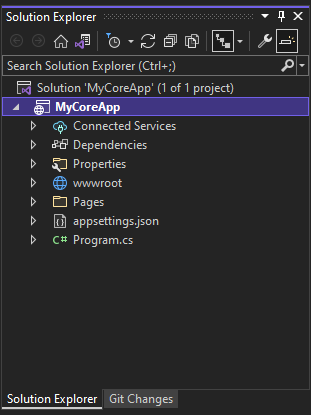
Visual Studio opens your new project.

**About your solution**

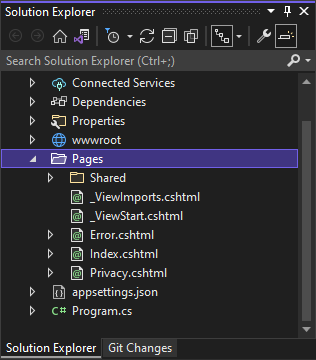
This solution follows the **Razor Page** design pattern. It's different than the [Model-View-Controller (MVC)](https://learn.microsoft.com/en-us/aspnet/core/tutorials/first-mvc-app/start-mvc?view=aspnetcore-2.1&tabs=aspnetcore2x&preserve-view=true) design pattern in that it's streamlined to include the model and controller code within the Razor Page itself.

**Tour your solution**

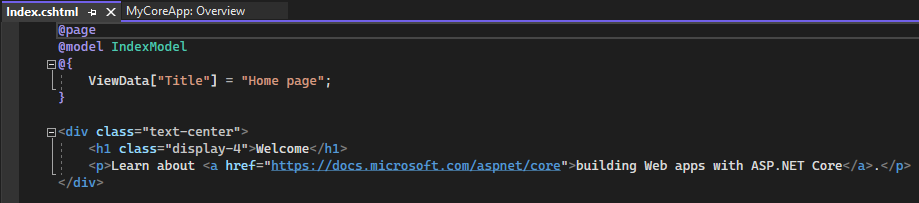
1. The project template creates a solution with a single ASP.NET Core project named **MyCoreApp**. Select the **Solution Explorer** tab to view its contents.



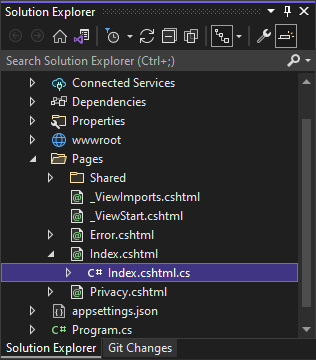
1. Expand the **Pages** folder.



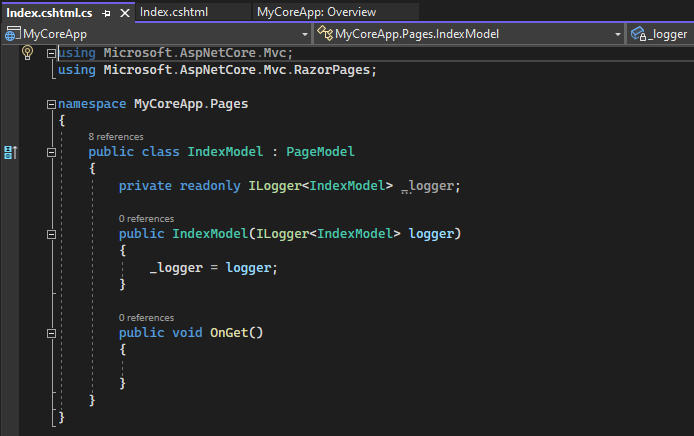
1. Select the **Index.cshtml** file, and view in the code editor.



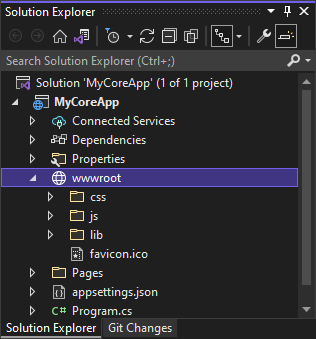
1. Each .cshtml file has an associated code file. To open the code file in the editor, expand the **Index.cshtml** node in Solution Explorer, and select the **Index.cshtml.cs** file.



1. View the **Index.cshtml.cs** file in the code editor.

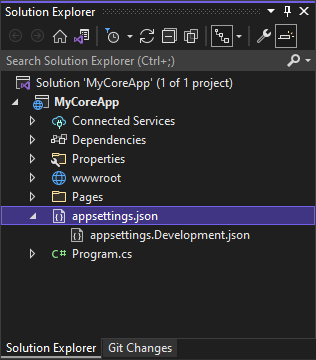


1. The project contains a **wwwroot** folder, which is the root for your website. Expand the folder to view its contents.



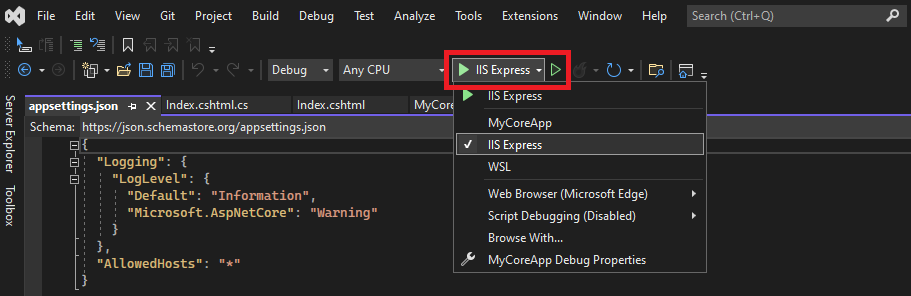
You can put static site content such as CSS, images, and JavaScript libraries directly in the paths where you want them.

1. The project also contains configuration files that manage the web app at run time. The default application [configuration](https://learn.microsoft.com/en-us/aspnet/core/fundamentals/configuration) is stored in **appsettings.json**. However, you can override these settings by using **appsettings.Development.json**. Expand the **appsettings.json** file to view the **appsettings.Development.json** file.



**Run, debug, and make changes**

1. In the toolbar, select the **IIS Express** button to build and run the app in debug mode. Alternatively, press **F5**, or go to **Debug** > **Start Debugging** from the menu bar.

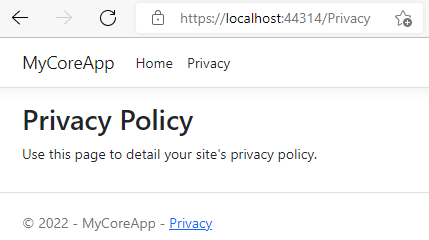


**Note**

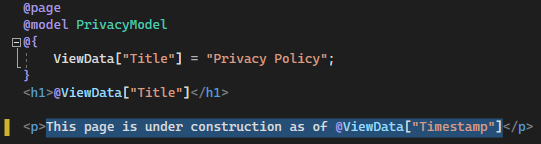
If you get an error message that says **Unable to connect to web server 'IIS Express'**, close Visual Studio and then relaunch the program as an administrator. You can do this task by right-clicking the Visual Studio icon from the Start Menu, and then selecting the **Run as administrator** option from the context menu.

You might also get a message that asks if you want to accept an IIS SSL Express certificate. To view the code in a web browser, select **Yes**, and then select **Yes** if you receive a follow-up security warning message.

1. Visual Studio launches a browser window. You should then see **Home** and **Privacy** pages in the menu bar.
2. Select **Privacy** from the menu bar. The **Privacy** page in the browser renders the text that's set in the **Privacy.cshtml** file.

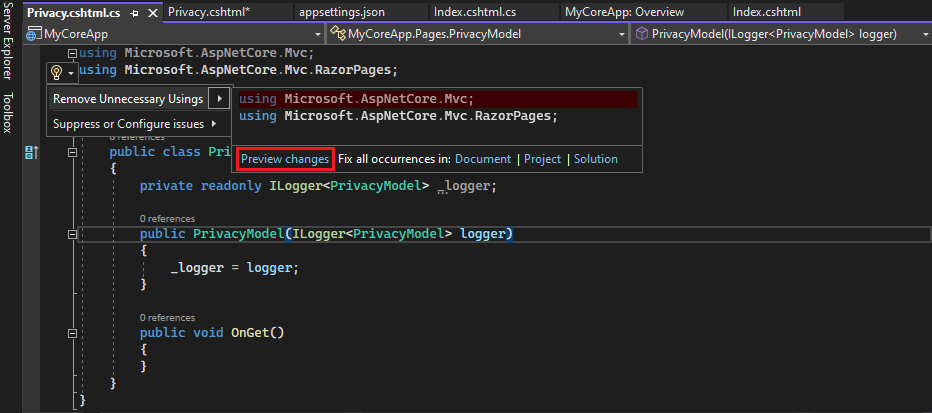


1. Return to Visual Studio, and then press **Shift+F5** to stop debugging. This action closes the project in the browser window.
2. In Visual Studio, open **Privacy.cshtml** for editing. Next, delete the sentence, *Use this page to detail your site's privacy policy* and replace it with *This page is under construction as of @ViewData["TimeStamp"]*.

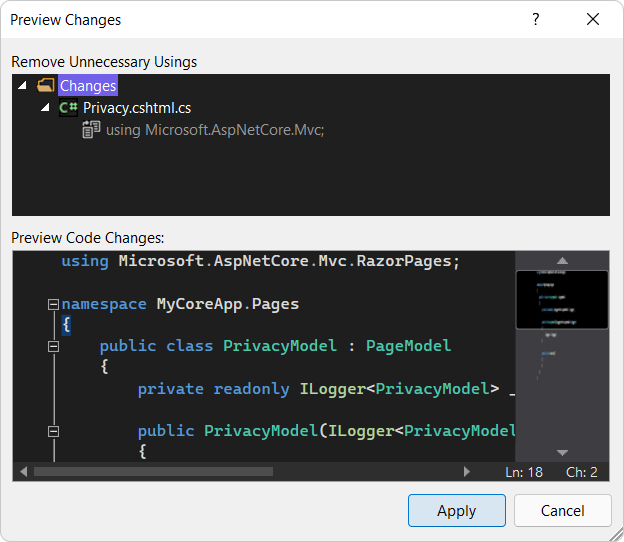


1. Now, let's make a code change. Select **Privacy.cshtml.cs**. Then, clean up the using directives at the top of the file by selecting the following shortcut:

Mouseover or select a greyed out using directive. A [Quick Actions](https://learn.microsoft.com/en-us/visualstudio/ide/quick-actions?view=vs-2022) light bulb appears below the caret or in the left margin. Select the light bulb, and then select the expand arrow next to **Remove unnecessary usings**.



Now select **Preview changes** to see what changes.



Select **Apply**. Visual Studio deletes the unnecessary using directives from the file.

1. Next, create a string for the current date that's formatted for your culture or region by using the [DateTime.ToString](https://learn.microsoft.com/en-us/dotnet/api/system.datetime.tostring) method.
   * The first argument for the method specifies how the date should be displayed. This example uses the format specifier (d) which indicates the short date format.
   * The second argument is the [CultureInfo](https://learn.microsoft.com/en-us/dotnet/api/system.globalization.cultureinfo) object that specifies the culture or region for the date. The second argument determines, among other things, the language of any words in the date, and the type of separators used.

Change the body of the OnGet() method to the following code:

C#Copy

public void OnGet()

{

string dateTime = DateTime.Now.ToString("d", new CultureInfo("en-US"));

ViewData["TimeStamp"] = dateTime;

}

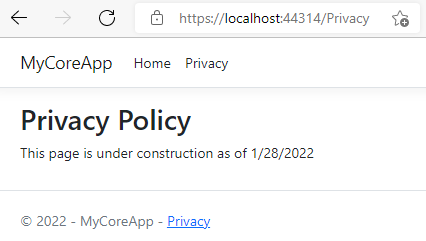
1. Notice that the following using directive automatically gets added to the top of the file:

C#Copy

using System.Globalization;

System.Globalization contains the [CultureInfo](https://learn.microsoft.com/en-us/dotnet/api/system.globalization.cultureinfo) class.

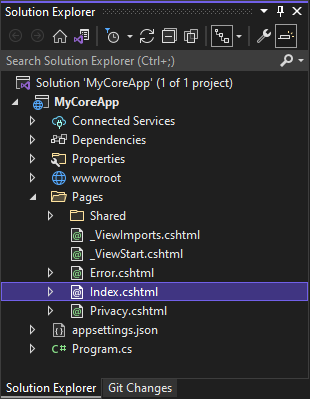
1. Press **F5** to open your project in the web browser.
2. At the top of the web site, select **Privacy** to view your changes.



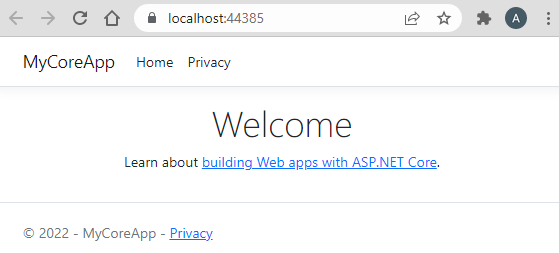
1. Close the web browser, press **Shift**+**F5** to stop debugging.

**Change your Home page**

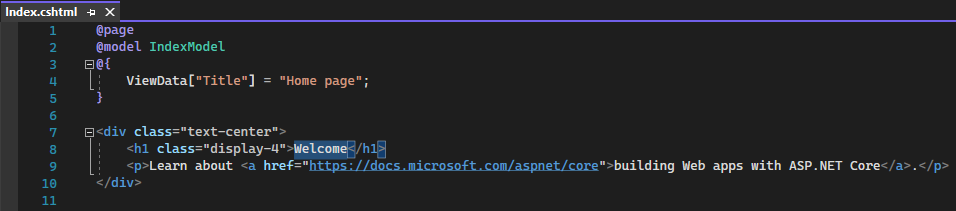
1. In the **Solution Explorer**, expand the **Pages** folder, and then select **Index.cshtml**.



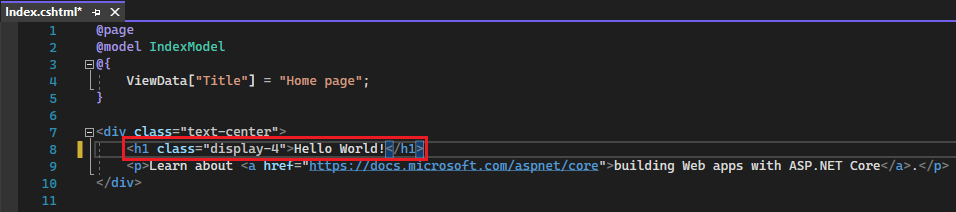
The **Index.cshtml** file corresponds with your **Home** page in the web app, which runs in a web browser.



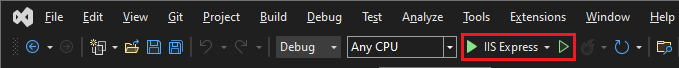
In the code editor, you see HTML code for the text that appears on the **Home** page.



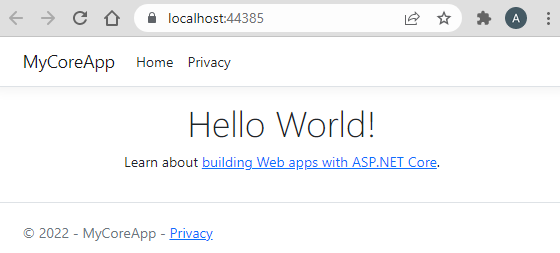
1. Replace the *Welcome* text with *Hello World!*



1. Select **IIS Express** or press **Ctrl**+**F5** to run the app and open it in a web browser.



1. In the web browser, you see your new changes on the **Home** page.



1. Close the web browser, press **Shift**+**F5** to stop debugging, and save your project. You can now close Visual Studio.

**Next steps**