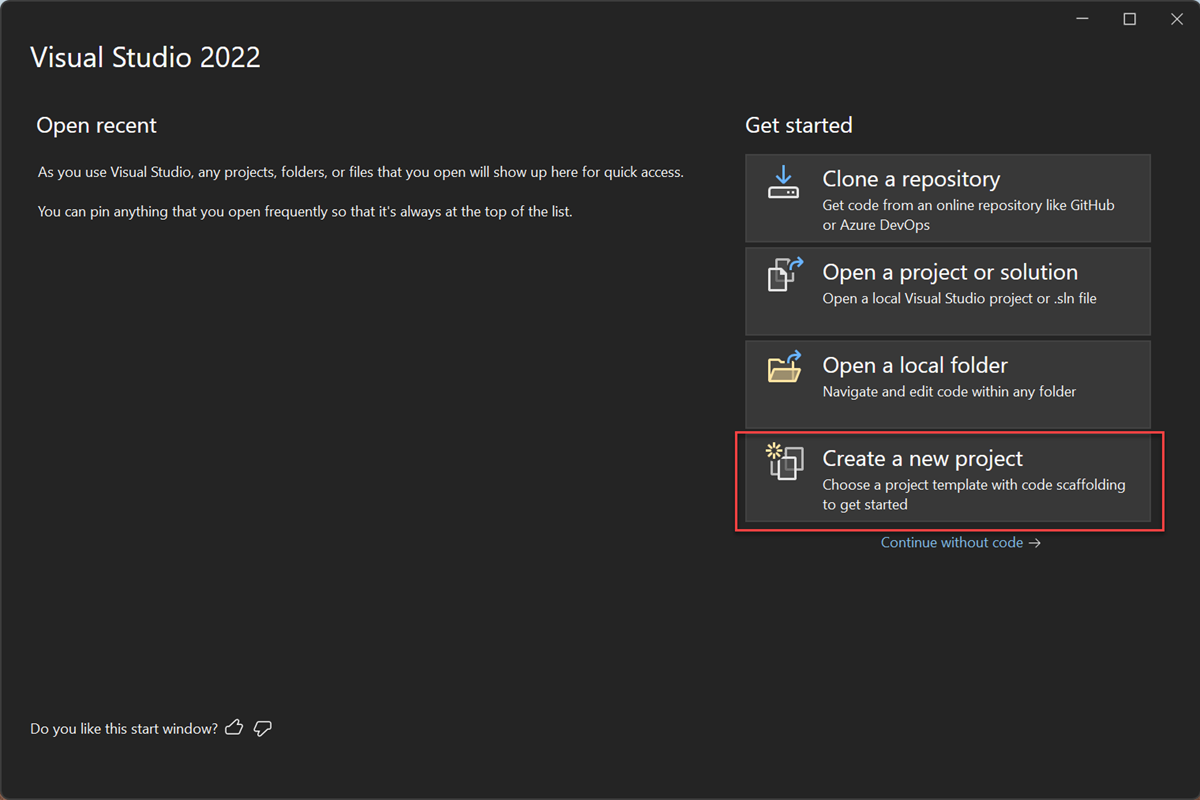
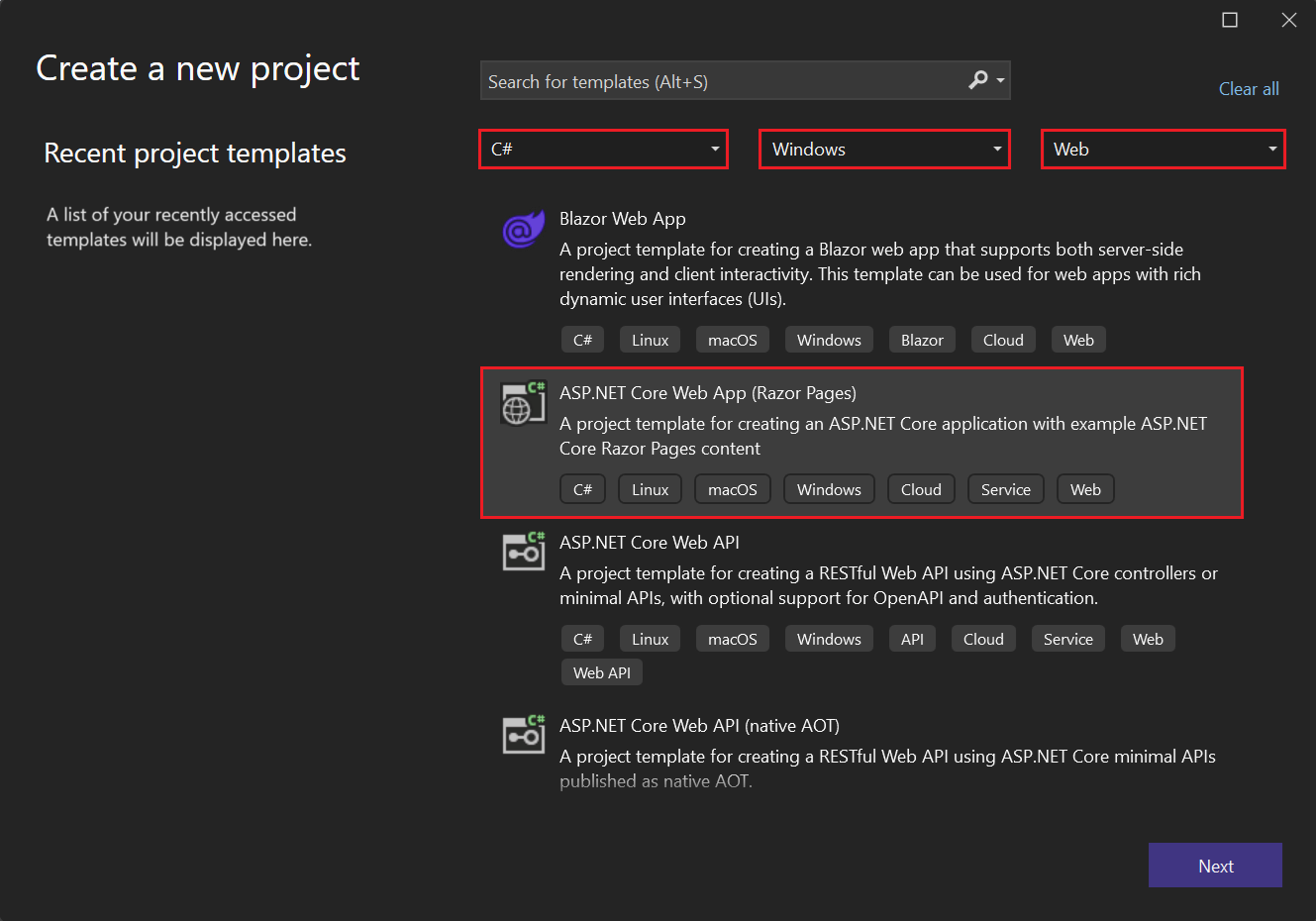
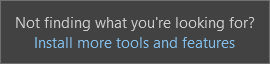
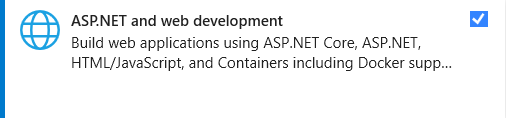
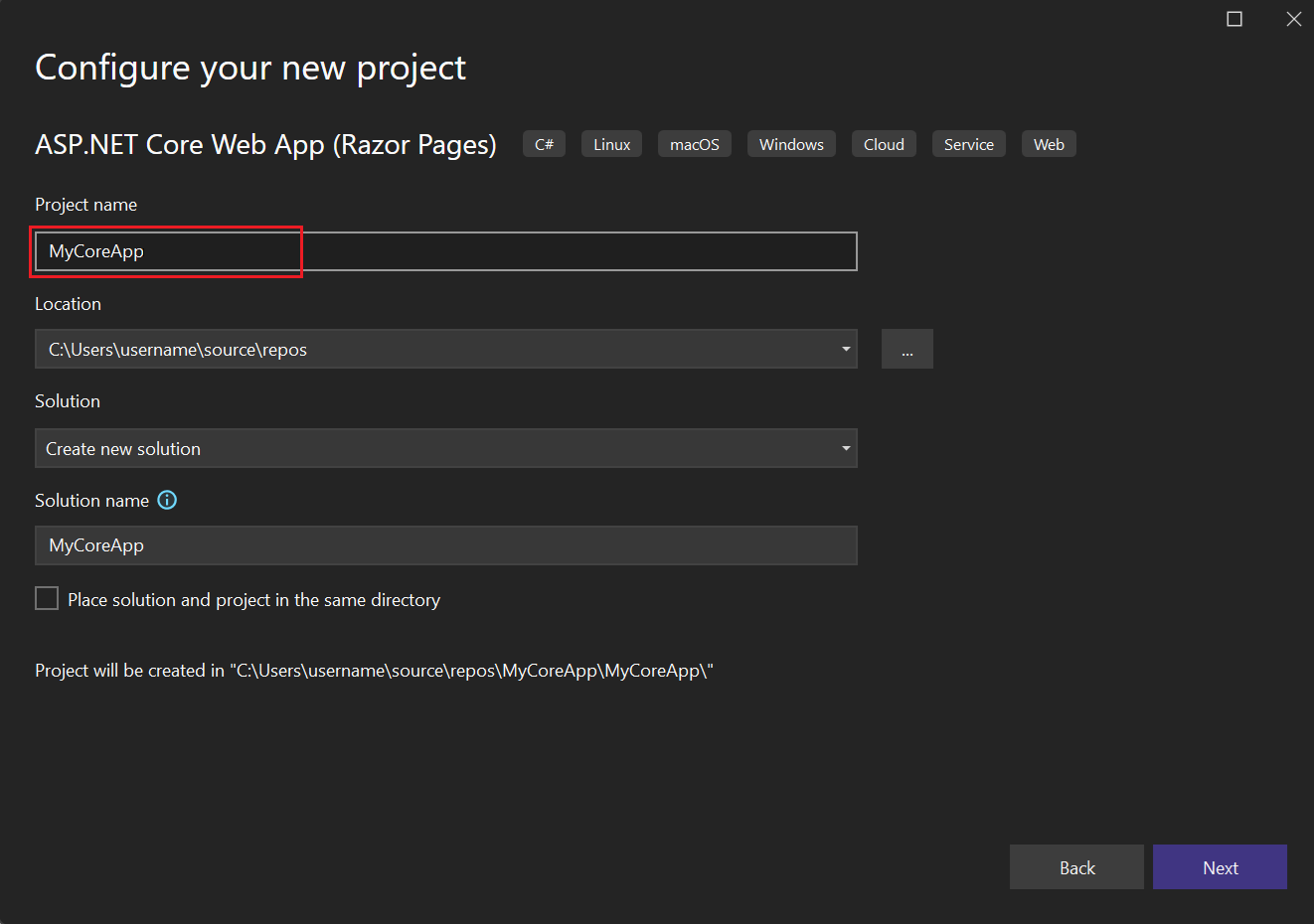
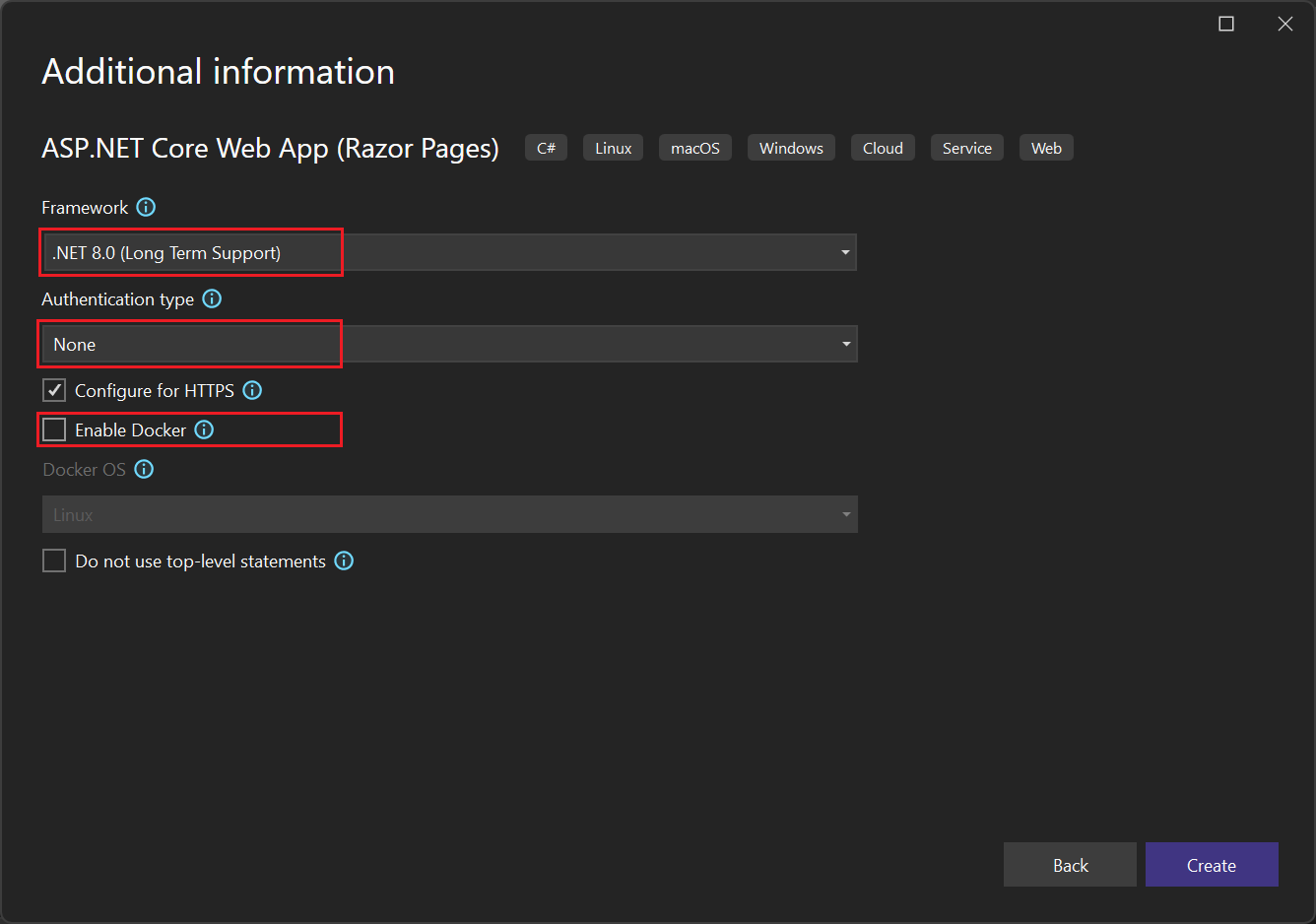
* Create a Visual Studio project
* Create a C# ASP.NET Core web app
* Make changes to the web app
* Explore IDE features
* Run the web app

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.

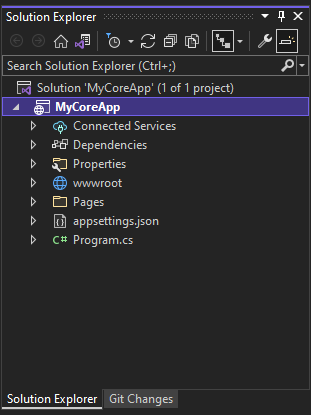
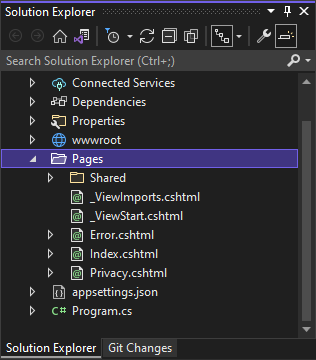
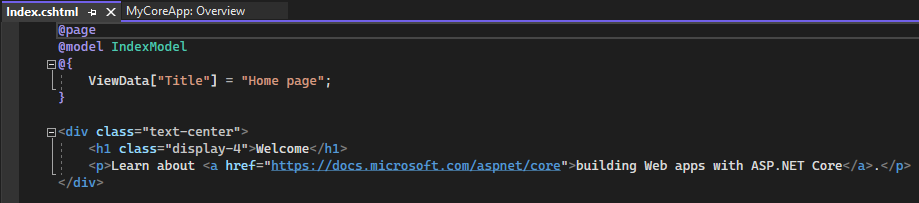
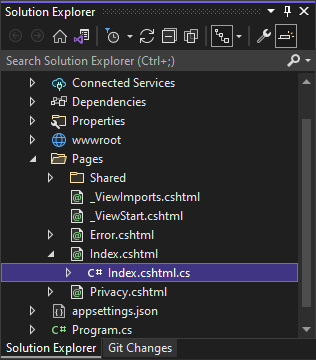
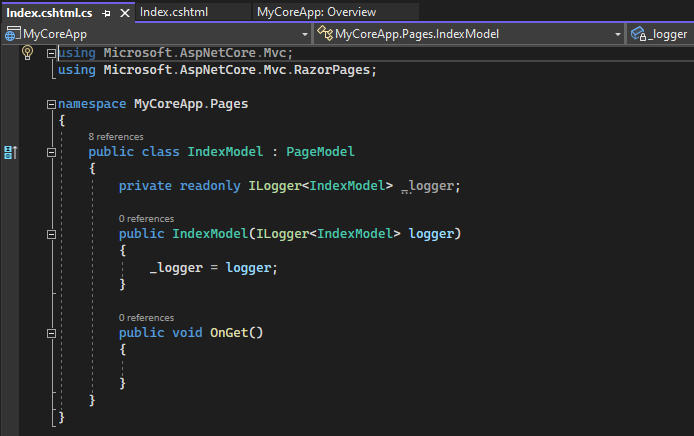
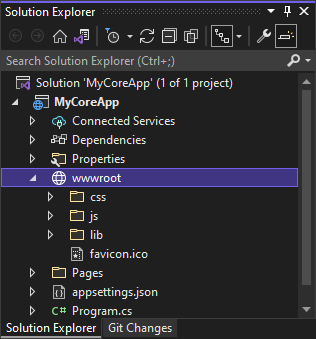
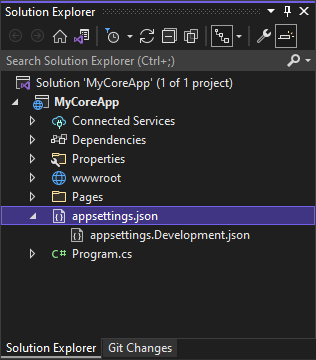
1. On the start window, select Create a new project.  
   
2. In the Create a new project window, select C# from the Language list. Next, select Windows from the platform list, and Web from the project types list.  
   After you apply the language, platform, and project type filters, select the ASP.NET Core Web App template, and then select Next.  
     
    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.
3. In the Configure your new project window, enter MyCoreApp in the Project name field. Then, select Next.  
   
4. 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.
5. Leave the Enable Docker box unchecked, and select None for Authentication Type.  
   
6. Select Create.

Visual Studio opens your new project.

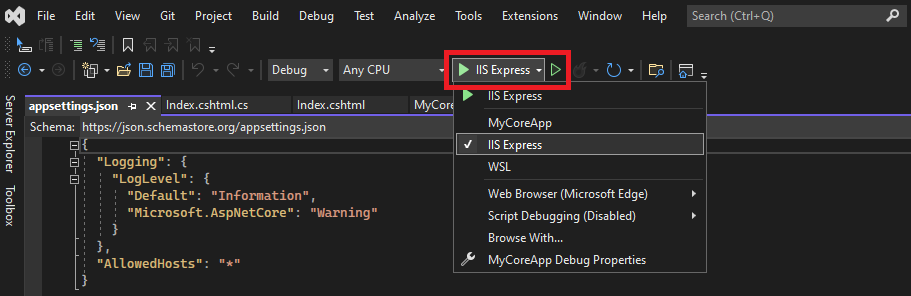
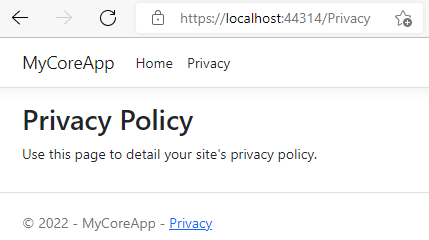
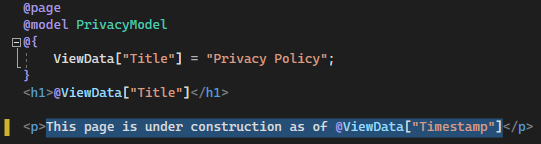
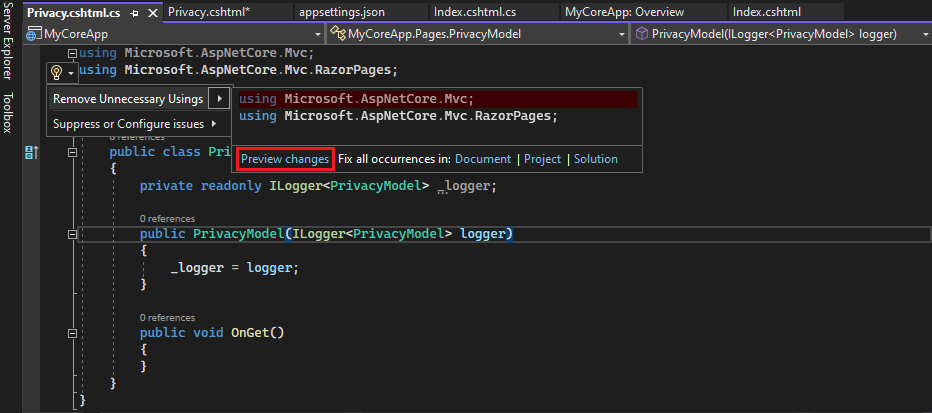
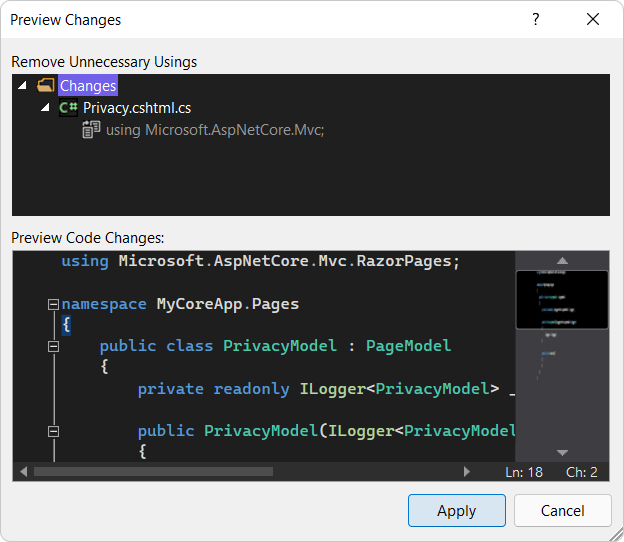
### **About your solution**

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.  
   
2. Expand the Pages folder.  
   
3. Select the Index.cshtml file, and view in the code editor.  
   
4. 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.  
   
5. View the Index.cshtml.cs file in the code editor.  
   
6. 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.
7. 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.
2. Visual Studio launches a browser window. You should then see Home and Privacy pages in the menu bar.
3. Select Privacy from the menu bar. The Privacy page in the browser renders the text that's set in the Privacy.cshtml file.  
   
4. Return to Visual Studio, and then press Shift+F5 to stop debugging. This action closes the project in the browser window.
5. 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"]*.  
   
6. 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.
7. 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.
8. Change the body of the OnGet() method to the following code:
9. C#
10. Copy

public void OnGet()

{

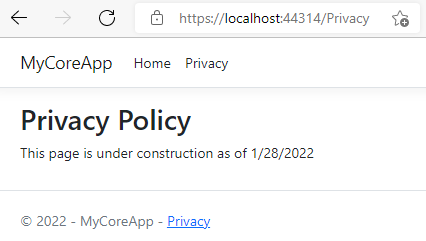
string dateTime = DateTime.Now.ToString("d", new CultureInfo("en-US"));

ViewData["TimeStamp"] = dateTime;

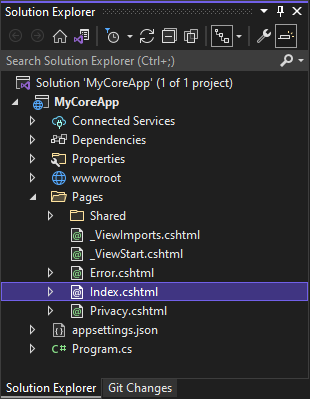
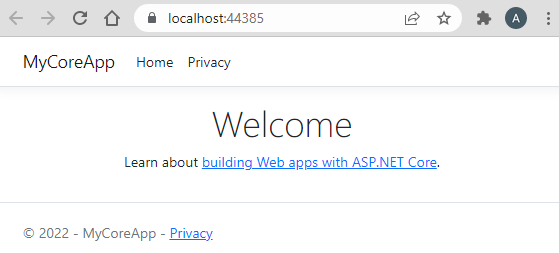
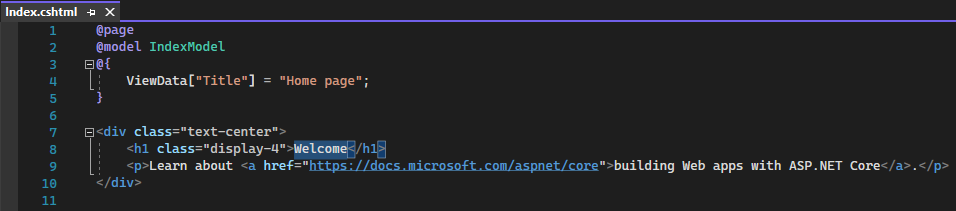
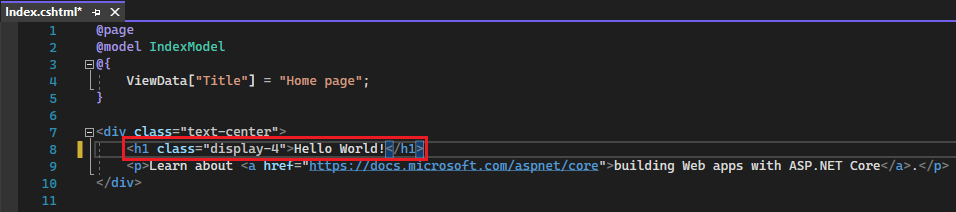
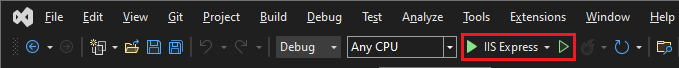
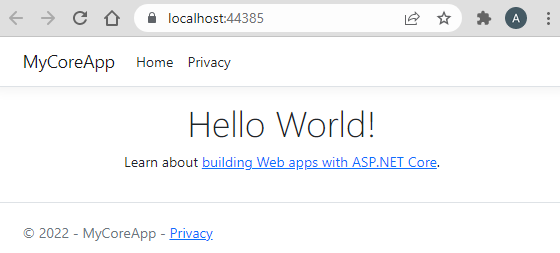
}

1. Notice that the following using directive automatically gets added to the top of the file:
2. C#
3. Copy

using System.Globalization;

1. System.Globalization contains the [CultureInfo](https://learn.microsoft.com/en-us/dotnet/api/system.globalization.cultureinfo) class.
2. Press F5 to open your project in the web browser.
3. At the top of the web site, select Privacy to view your changes.  
   
4. 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.  
   
2. Replace the *Welcome* text with *Hello World!  
   *
3. Select IIS Express or press Ctrl+F5 to run the app and open it in a web browser.  
   
4. In the web browser, you see your new changes on the Home page.  
   
5. Close the web browser, press Shift+F5 to stop debugging, and save your project. You can now close Visual Studio