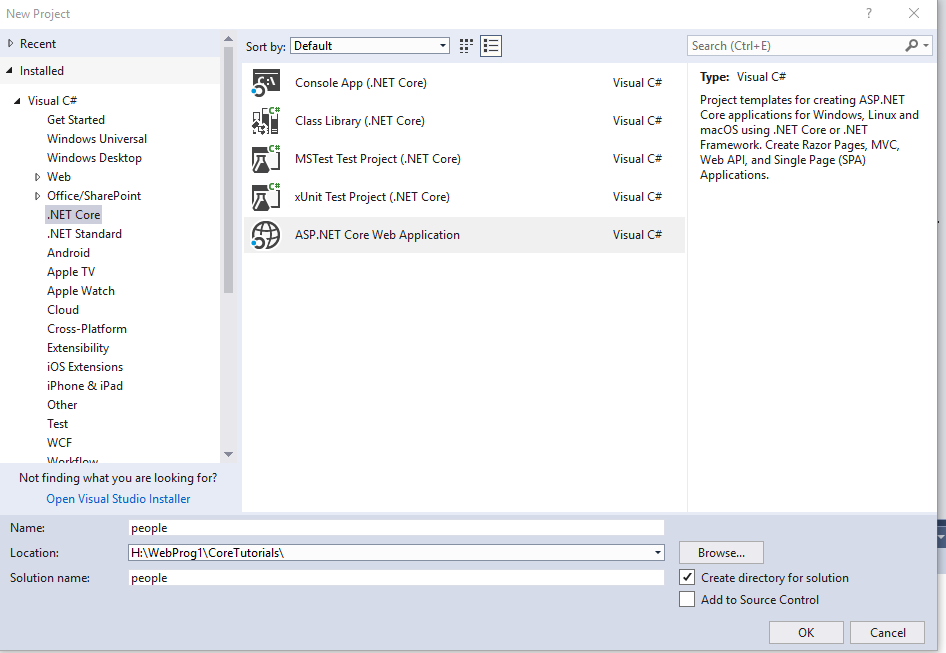
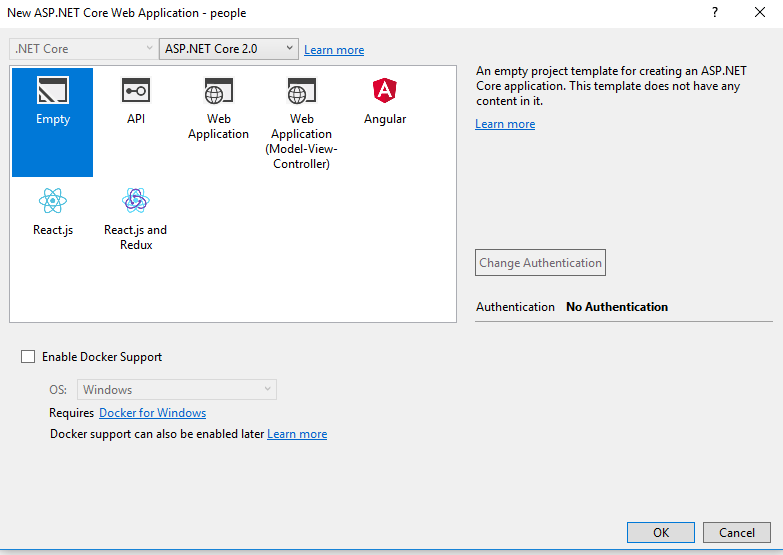
**Task 1:**

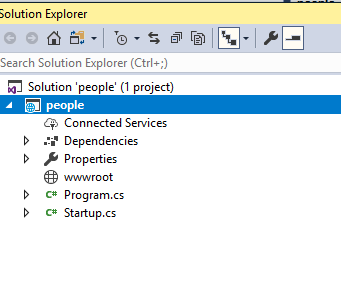
Start Visual Studio, create a new ASP.NET Core Web Application.

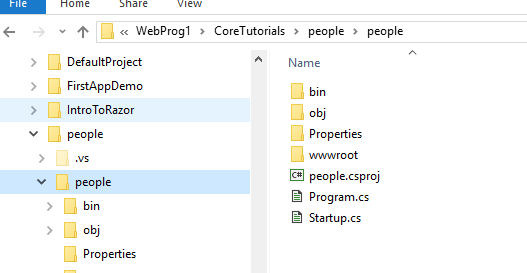
Remember to pick an appropriate folder to store it.





Create the application using the empty project template.



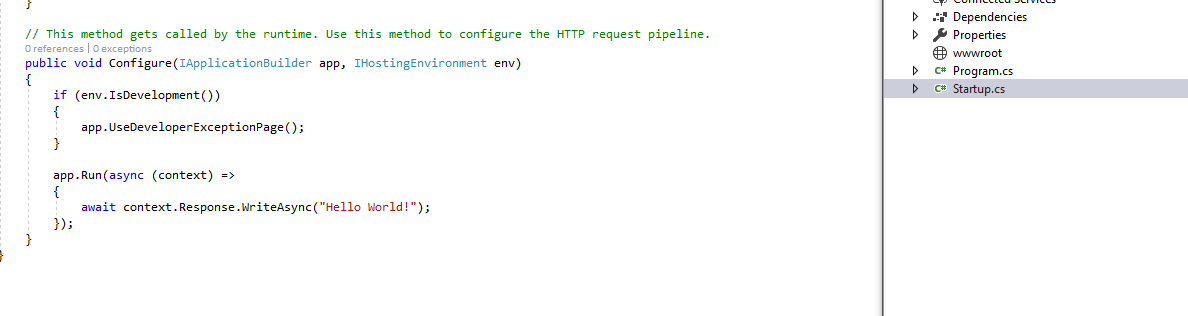


Look at the files and folders which have been created. You can see these in the solution explorer using Visual Studio and out of VS using the file explorer. Remember not to move or rename things in the file explorer.

**Task 2: The Startup.cs file.**

Run the application. You should see “Hello World” in the browser. This has come from the Configure method in the Startup.cs file.

Since we haven’t told it anything else the server gives us a Response.



Change the text of “Hello World” and see what happens.

**Task 3: Static Files**

Add a HTML file called hello.html to the wwwroot folder. This is the folder where all static files should be stored. (images, CSS, Javascipts files, etc). Put a <h1> Hello from a static html file</h1> into the file.

Run the application. Add the path /hello.html to the URL to see the static file – what do you see?

In order to see the static file you need to change the configuration to allow the server to use static files.

Add the following code to the Configure method in the Startup.cs file. This should be added before the app.Run code.

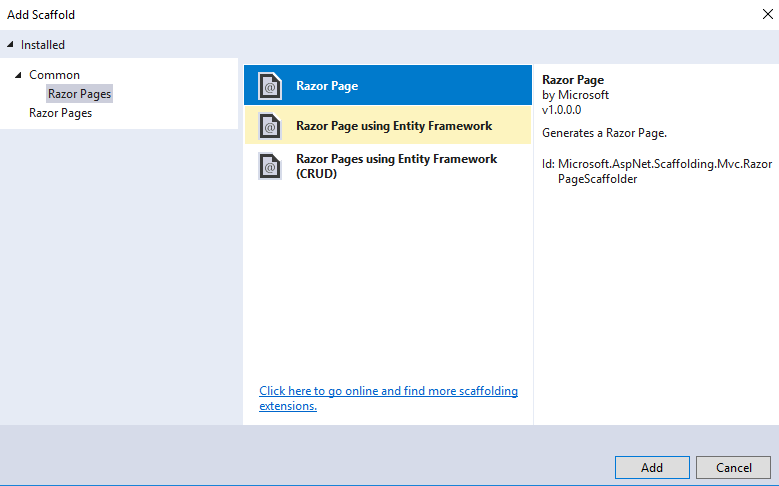


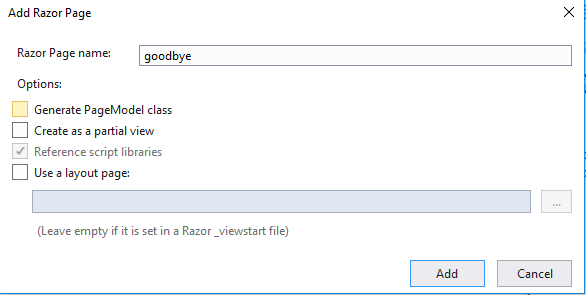
Run the application again and see if you can see the hello.html page this time.

You should be able to make changes to the html page and see the changes by refreshing your browser.

**Task 4: Add a razor page without a Model.**

Add a folder called pages. Add a Razor Page to this folder called **goodbye.cshtml** (Right click on the folder is the quickest way to do this). Be careful what options you choose, for the first page we don’t want a PageModelClass or a layout.





Add some markup to the file **<h1> Goodbye from the goodbye.cshml file</h1>**

In order to be served this page we need to edit the configure method of the Startup.cs file again.

This time add **app.UseMvc()** underneath the line **app.UseStaticFiles().**

We also need to add something to the CongureServices method.

Add the line **services.AddMvc();** to this method

To see the page add the path /goodbye to the URL (note if use goodbye.cshtml you will get not get the correct file).

Add some code to this page to display the current time and whether it is afternoon or morning.

**Task 4: Add a razor page with a Model.**

Add a new razor page called **greeting** this time Generate a PageModel class.

Look at the solution explorer and the file explorer to see what files have been created and where they are stored.

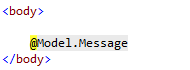
In the file manager you should notice that there are two files in the Pages folder called **greeting.cshtml** and **greeting.cshtml.cs.** In the solution explorer the .cs folder is under the other one.

These files are linked through the @Model directive at the top of the .cshtml file.

Add a property to the .cs file:



This property can be accessed in the .cshtml file by using @Model



Refresh the greeting page and make sure you can see the message.

**Task 5: Add code to the OnGet() method.**

The PageModel file you created has a method called OnGet(), this is called when the page is requested with a GET request (the default when you visit a URL).

Edit the method so that it prints out Good Morning or Good Afternoon depending on the time.

