Static Files Middleware in ASP.NET Core ||UseStaticFiles, UseDefaultFiles and UseFileServer .

Static Files Middleware

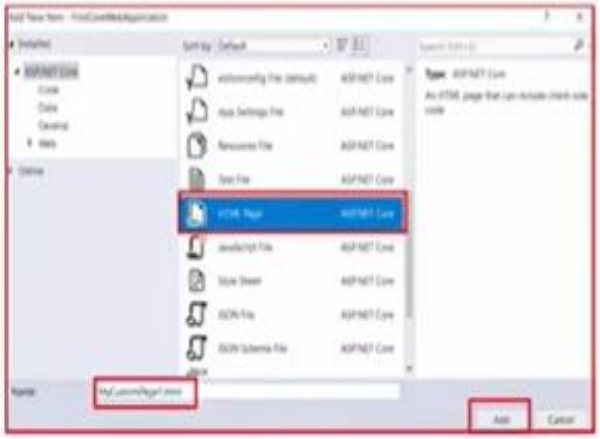
- One of the most important features almost all web applications should have the ability to serve the static files directly from the file system.
- The static files such as HTML, Images, CSS, and JavaScript are the important assets of an application and ASP.NET Core can serve these files directly to the clients. But the important point that you need to keep in mind by default the ASP.NET Core cannot serve these static files.
- Some configuration is required in order to enable the ASP.NET Core to serve these static files direct

Where do we need to store the static files in ASP.NET Core?

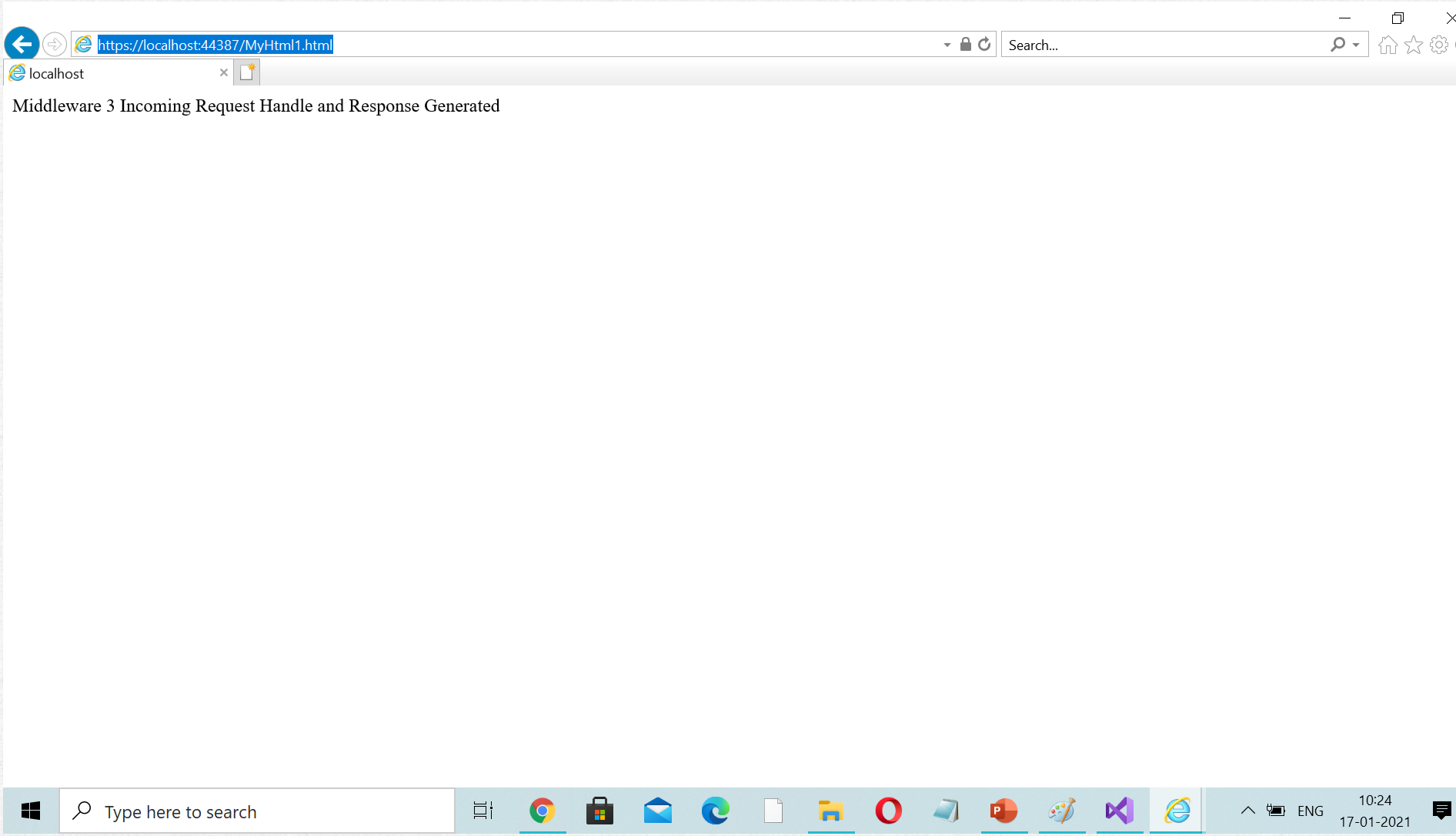
- In ASP.NET Core, the default directory or location for the static files is **wwwroot** (webroot) folder and moreover, this folder or directory should be present in the root project folder.
- By default, this is the only place where the ASP.NET Core application can serve the static files directly. But we can change this default behavior by using the **UseWebRoot** method.

Adding the wwwroot (webroot) folder:

- Right-click on the project and then select **add => new folder** option and then provide the folder name as **wwwroot**.
- Once you created the **wwwroot** folder, let's add an HTML file within that folder. To do so, right-click on the **wwwroot** folder and then select **add => new item** which will open add new item window. From that window select the **HTML template**, provide a name such as "**MyCustomPage1**" and then click on the **Add** button as shown in the below image.

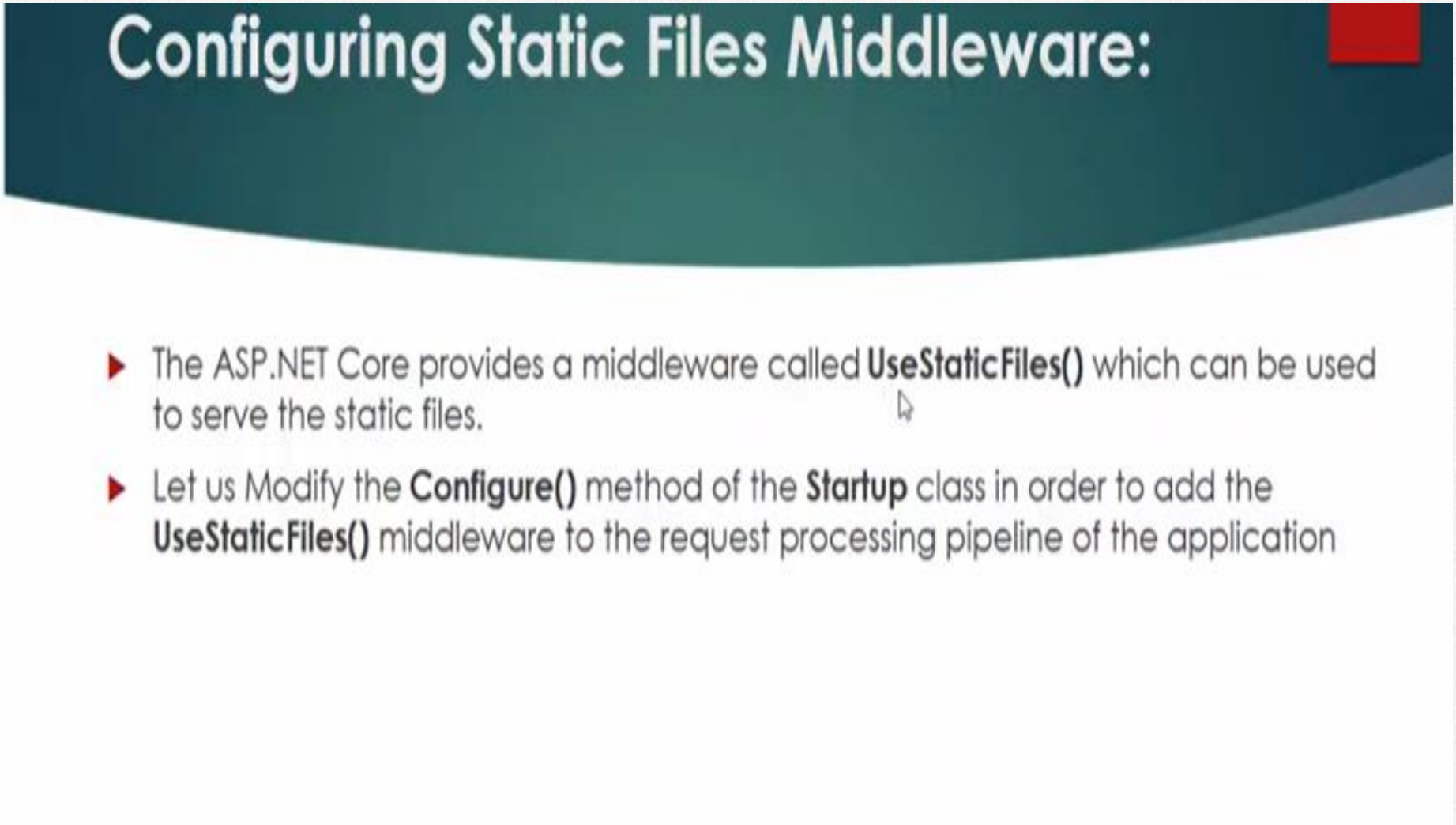


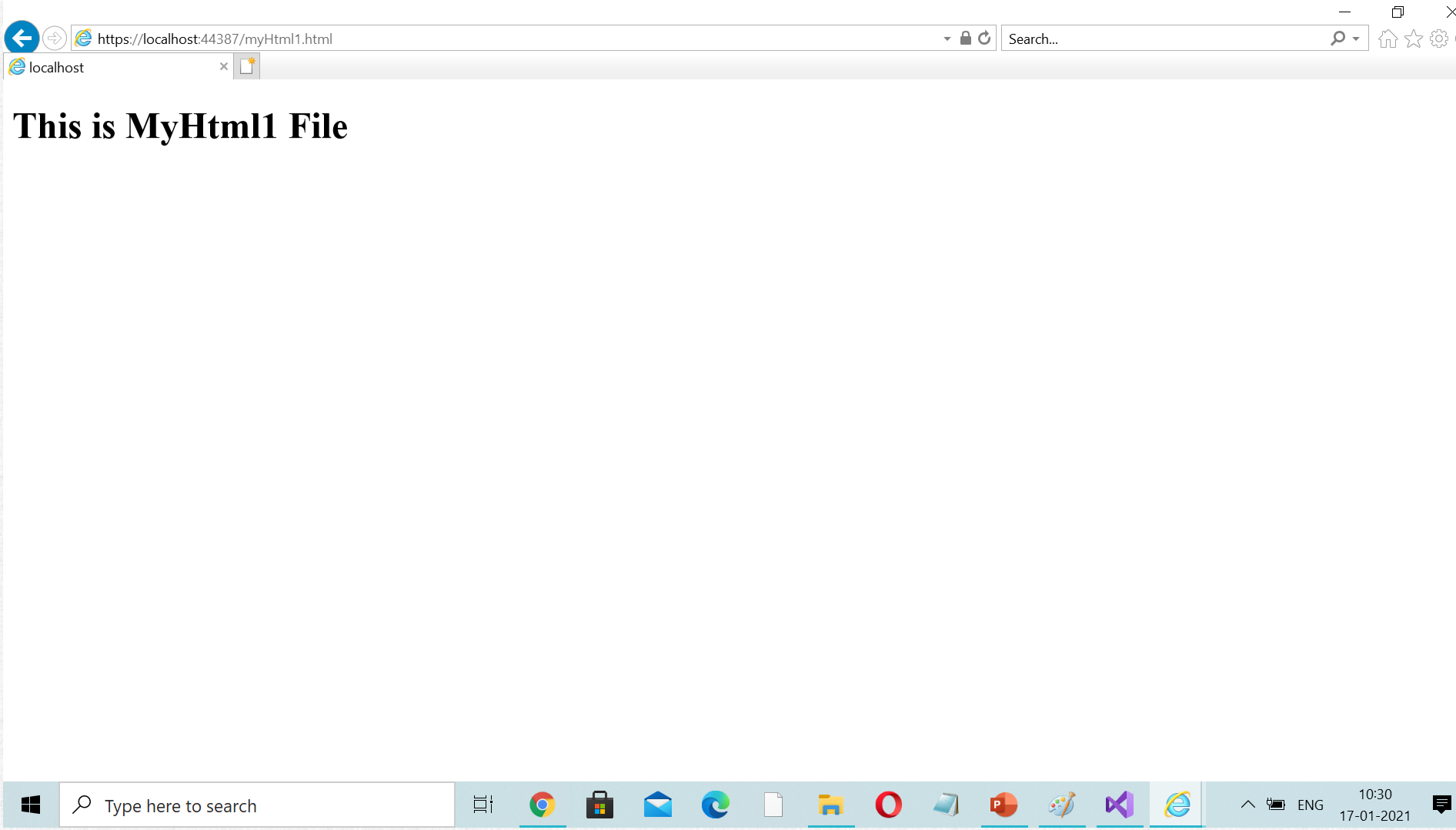
If we run html file but run method content will be display



How to run static file

- By using middleware method (UserStaticFiles)




https://localhost:44387/myHtml1.html
Search...
localhost
This is MyHtml1 File
Type here to search
ENG
10:30
17-01-2021

Setting the Default Page:

- Most of the web applications have a default page such as **index.htm(l)** or **default.htm(l)** as their startup page as it is easy to remember.
- This is the web page that is going to be displayed when a user visits the root URL of that application.
- For example, if you have a page with the name **index.html** and you want that page to be your default page so that whenever any user visits your root URL, then that page is going to be displayed.

- Run the Default static Page:-
- app.UseDefaultFiles(); app.UseStaticFiles();

UseDefaultFiles()

- You are getting the response from the middleware which is registered using the **Run()** extension method.
- In order to serve the index.html page as the default page of your application, you need to add another middleware i.e. **UseDefaultFiles()** middleware into the request processing pipeline.
- So, modify the **Configure()** method of the **Startup** class to use the **UseDefaultFiles()** middleware which will set the default page for your application.
- You need to add the **UseDefaultFiles()** middleware before the **UseStaticFiles()** middleware in order to serve the default file. The point that you need to remember is the UseDefaultFiles() middleware is just a URL rewriter and it never serves the static files. The job of this middleware is to simply rewrite the incoming URL to the default file which will then be served by the Static Files Middleware.

How to create your own page as default page as startup page

- DefaultFilesOptions df = new DefaultFilesOptions();
- df.DefaultFileNames.Clear();//Clear to the default page options
- df.DefaultFileNames.Add("MyHtml1.html");//set this page as default page to run
- app.UseDefaultFiles();
- app.UseStaticFiles();

UserFileServer Middleware Component:-

UseFileServer()

- The **UseFileServer()** middleware components combines the functionality of **UseStaticFiles**, **UseDefaultFiles** and **UseDirectoryBrowser** middleware. We already discussed the UseStaticFiles and UseDefaultFiles middleware. The **DirectoryBrowser** middleware as the name says enables the directory browsing which allows the users to see the files which are stored in a specific directory. In our example, we can replace the **UseStaticFiles()** and **UseDefaultFiles()** middlewares with the **UseFileServer()** Middleware

- app.UseFileServer();:-- In place of app.UseDefaultFiles(), app.UseStaticFiles(); we have to use only app.UseFileServer() method to run the default page(Index or default page).

- FileServerOptions fs = new FileServerOptions();
- fs.DefaultFilesOptions.DefaultFileNames.Clear();
- fs.DefaultFilesOptions.DefaultFileNames.Add("MyHtml1.html");
- app.UseFileServer(fs);//run the custom page

- No need to put this middleware to run custom page.
- app.UseDefaultFiles();
- app.UseStaticFiles();

Thanks