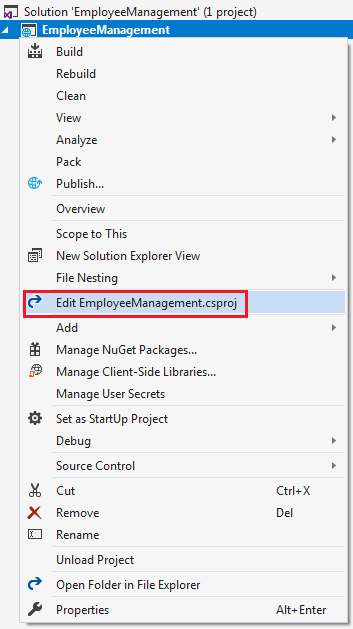
**ASP.NET Core Project file**

we will **explore and understand the asp.net core project file.**  
  
  
We are using **C#** as the programming language, so the project file has **.csproj**extension. If you use **Visual Basic**as the programming language, then the project file extension is **.vbproj**. If you have worked with previous versions of ASP.NET, then this file might be very familiar to you, but the format and content that is included in this file has changed significantly in asp.net core.    
  
  
One significant change is, **the project file does not contain any folder or file reference.** Let me explain what I mean. In previous versions of ASP.NET, when we add a file or folder to the project using solution explorer, **a reference to that file or folder is included in the project file**. In ASP.NET core, the project file does not contain any folder or file reference.   
  
**The File System determines what files and folders belong to the project.** All files and folders that are present in the project root folder are part of the project and will be displayed in the solution explorer. When you add a file or folder using the File Explorer, that file or folder is part of the project and will be immediately displayed in the solution explorer. Similarly when you delete a file or folder from the Project folder or any of it's sub folders, that deleted file or folder is no longer part of the project and that fact is immediately reflected in the Solution Explorer.  
  
Also **the way we work with this file has changed**. In previous versions of asp.net to be able to edit the project file we first have to unload the project, edit and save the project file and then reload the project. Where as in asp.net core, we can edit the project file without unloading the project.  
  
In the Solution Explorer, right click on the project Name and select **"Edit EmployeeManagement.csproj"**   
  
   
  
This opens the **.csproj**file in the editor.

<Project Sdk="Microsoft.NET.Sdk.Web">

  <PropertyGroup>

    <TargetFramework>netcoreapp2.2</TargetFramework>

    <AspNetCoreHostingModel>InProcess</AspNetCoreHostingModel>

  </PropertyGroup>

  <ItemGroup>

    <PackageReference Include="Microsoft.AspNetCore.App" />

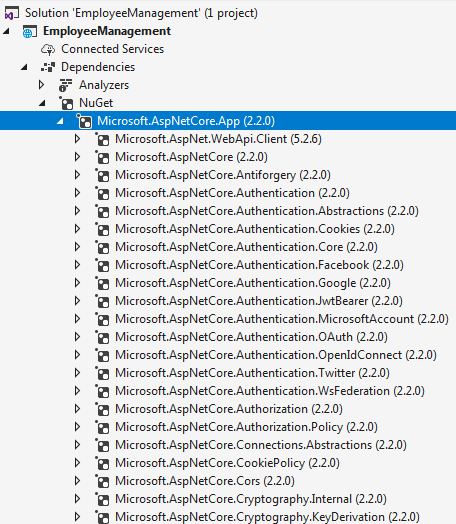
    <PackageReference Include="Microsoft.AspNetCore.Razor.Design" Version="2.2.0"PrivateAssets="All" />

  </ItemGroup>

</Project>

**TargetFramework :**As the name implies this element is used to specify the target framework for your application i.e the set of APIs that you'd like to make available to the application. To specify a target framework we use something called **Target Framework Moniker (TFM)**. As you can see in the above example, our application targets one frameowrk **netcoreapp2.2**. netcoreapp2.2 is the Moniker for .NET Core 2.2. When we created this application, we selected .NET Core 2.2 as the target framework from the New Project Dialog dropdown list.   
  
**AspNetCoreHostingModel :**This element specifies how the asp.net core application should be hosted. Should it be hosted **InProcess**or **OutOfProcess**. The value of InProcess specifies that we want to use in-process hosting model i.e host our asp.net core app inside of the IIS worker process (**w3wp.exe**). The value of OutOfProcess specifies that we want to use  out-of-process hosting model i.e forward web requests to a back-end ASP.NET Core app running the **Kestrel**server. We will discuss InProcess and OutOfProcess hosting in detail in our upcoming sessions.  
  
**PackageReference :** As the name implies, this element is used to include a reference to all the NuGet packages that are installed for your application. At the moment in the project file we have the following 2 NuGet packages.

* Microsoft.AspNetCore.App
* Microsoft.AspNetCore.Razor.Design

**Microsoft.AspNetCore.App :** This NuGet package is called **metapackage**. A metapackage has no content of its own but is a list of dependencies (other packages). You can find this metapackage, in the Solution Explorer, under **NuGet**which in turn is under **Dependencies**. When you expand the metapackage, you can find all the dependencies.   
  
   
  
All the features of ASP.NET Core 2.1 and later and Entity Framework Core 2.1 and later are included in this **Microsoft.AspNetCore.App**package. The default project templates targeting ASP.NET Core 2.1 and later use this package.  
  
**Notice there is no version number on the metapackage.** When the version is not specified, an**implicit version**is specified by the SDK. The .NET Core team recommends relying on the implicit version specified by the SDK and not explicitly setting the version number on the package reference. Don't worry if this is not entirely clear at the moment. We will discuss metapackage and implicit version in detail in our upcoming sessions.  
  
**Microsoft.AspNetCore.Razor.Design :**This package contains MSBuild support for Razor and is referenced by Microsoft.AspNetCore.App meta package.