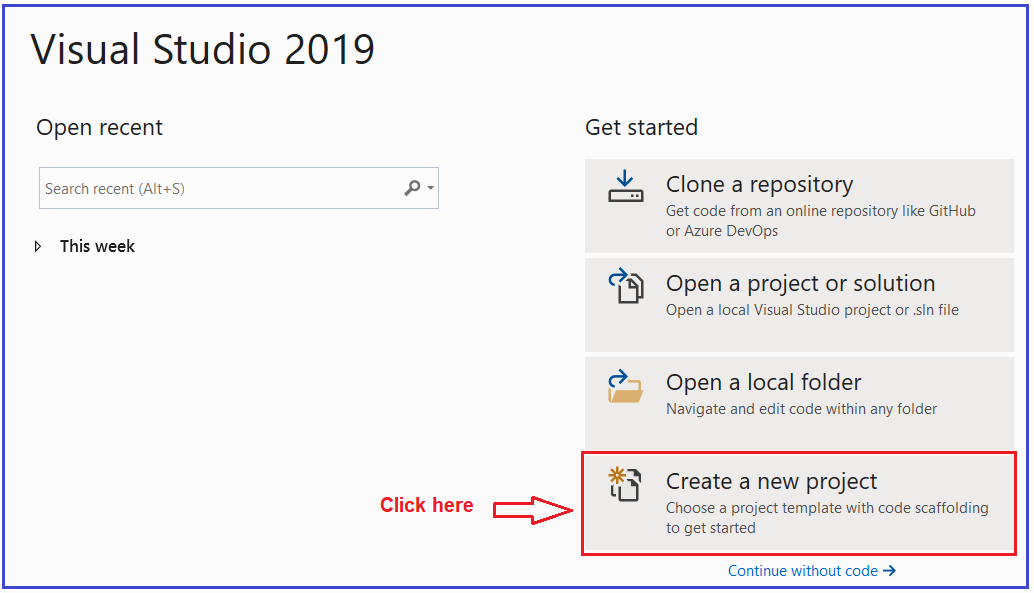
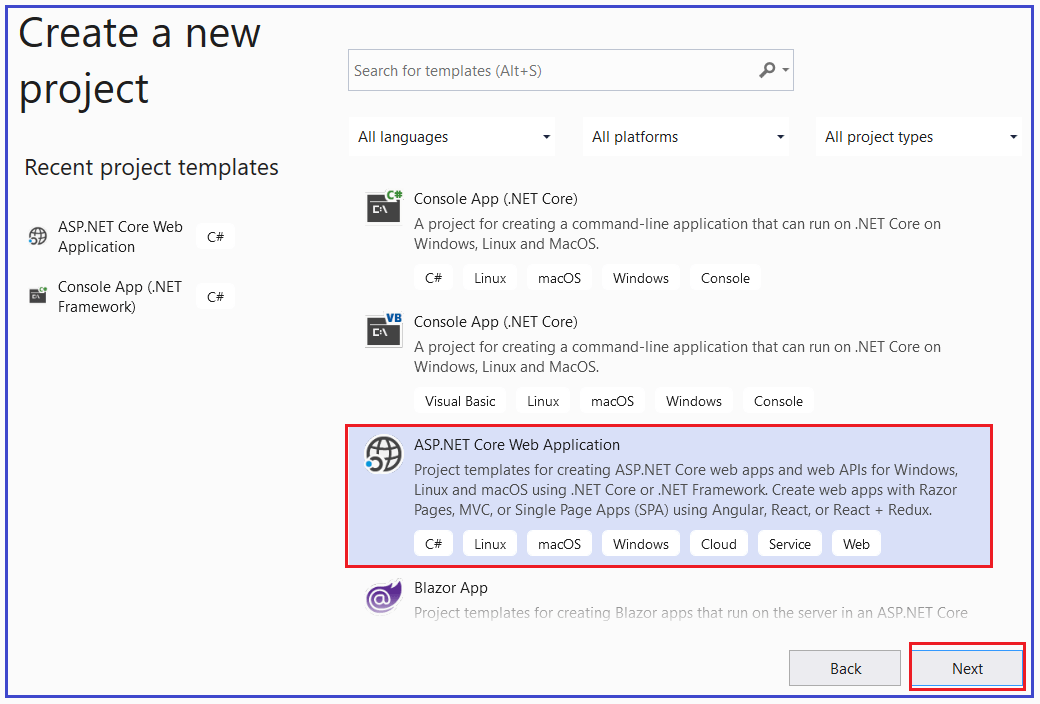
**ASP.NET Core Project File**

##### **Creating a new ASP.NET Core Project with Empty template:**

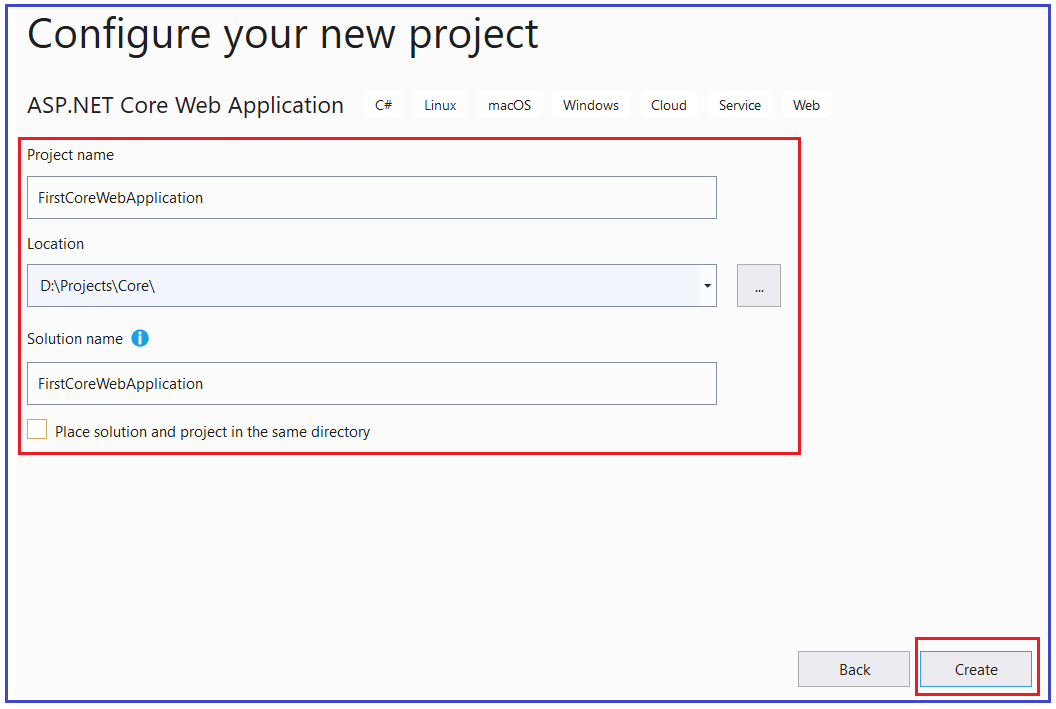
Open Visual Studio 2019. And then click on the Create a new project box as shown in the below image.



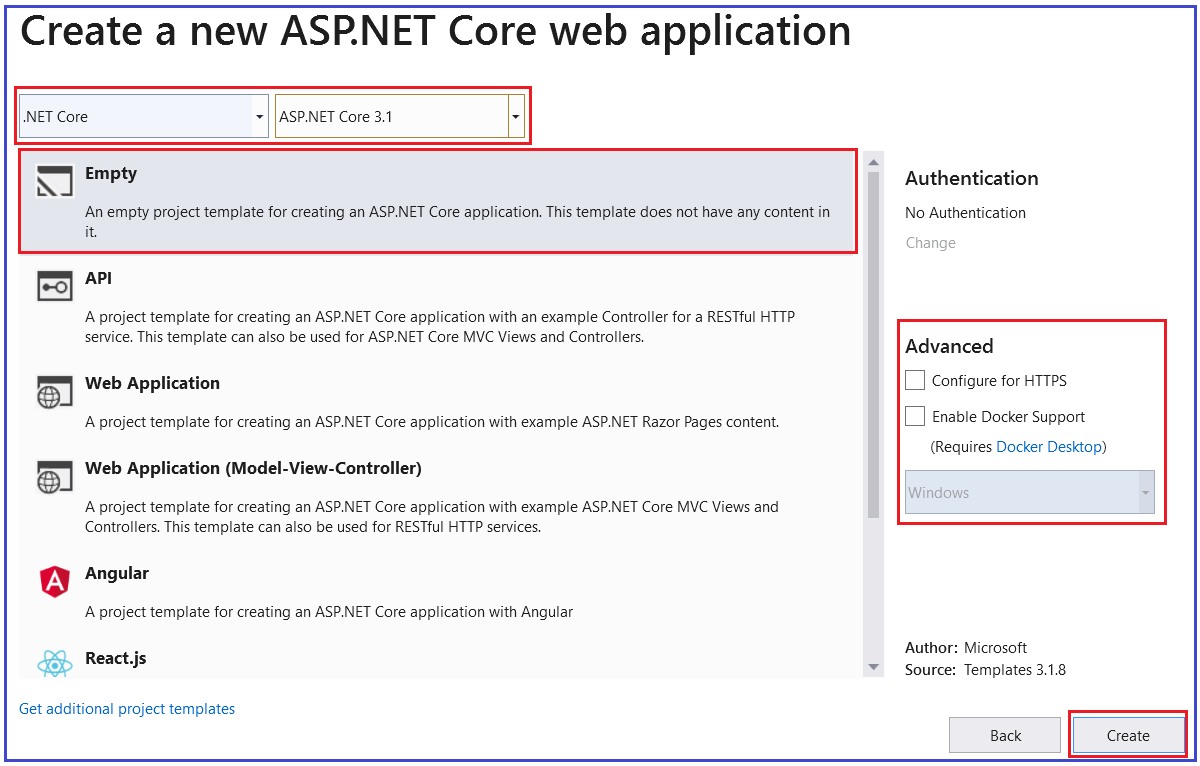
Once you click on the **Create a new project** box, it will open the “**Create a new project**” window. Here, select the **ASP.NET Core Web Application** template and click on the Next button as shown in the below image.



Once you click on the Next button, it will open the **Configure Your New Project** window. Here, you need to give the project name, the location where you want to create this project, and the solution name. In this example, let us give the name as “**FirstCoreWebApplication**” (you can give any name as per your choice) and click on the Create button as shown in the image below.



Once you click on the create button, it will open the “**Create a new ASP.NET Core Web Application**” window. Here, you need to select the Empty Template. Also, make sure that you have selected the.NET Core and ASP.NET Core versions (latest 3.1). Make sure to uncheck all the checkboxes from the Advanced section and finally click on the **Create** button as shown in the below image.



Once you click on the Create button, it will create a new ASP.NET Core Web Project in Visual Studio 2019.

##### **Changes in the Project file in ASP.NET Core Application:**

The ASP.NET Core Project File does not contain any references for folder or file. In previous versions of ASP.NET Framework, when we add a folder or a file to the project using Solution Explorer, then a reference to that folder or file is added in the project file. But with ASP.NET Core, the Project File does not include any reference for the Files or Folders added the project using Solution Explorer.

Another significant change in ASP.NET Core is, the File System determines what folders and files belong to the project. Generally, the files and folders which are present at the project root folder are part of the project. Those files and folders which are present at the project root folder are only going to display in the Solution Explorer.

So, if you add a file or a folder using the File Explorer, then that file or folder is part of the project. As soon as you add a file or folder using the file explorer, it immediately displayed in the solution explorer. In the same way, if you delete a file or folder from the Project Folder or any of its subfolders, then that deleted file or folder is no longer part of the project and that is reflected immediately in the Solution Explorer. And the vice-versa is also true.

##### **How we can Edit the Project File in Previous Versions of ASP.NET:**

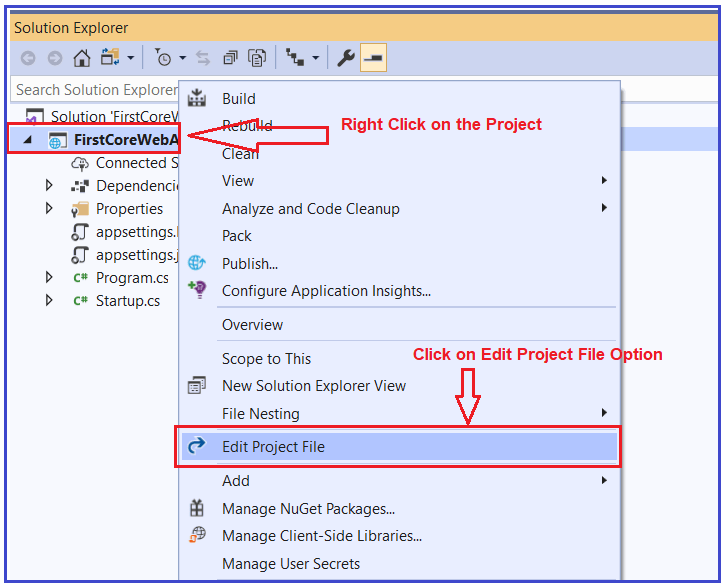
In our previous versions of ASP.NET Framework, in order to edit the project file, we need to follow the below steps

1. First, we need to unload the project
2. Then we need to edit the project file
3. Once we edit the file then we need to save the project file
4. Then reload the project.

But with ASP.NET Core we can edit the project file without unloading the project.

##### **How we can Edit the ASP.NET Core Project File:**

ASP.NET Core 1.0 does not create a .csproj file, instead, it uses .xproj and project.json files to manage the project. This has changed in ASP.NET Core 2.0. Visual Studio now uses the .csproj file to manage projects. To edit the ASP.NET Core project file, right-click on the project name in the Solution Explorer, and then select “**Edit Project File**” from the context menu as shown in the below image.



Once you click on the “Edit Project File” then the Project file will open in the Visual Studio editor as shown in the below image.



##### **Understanding the ASP.NET Core Project File:**

Let us understand the ASP.NET Core Project File elements.

##### **TargetFramework:**

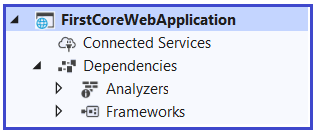
The TargetFramework element in the project file is used to specify the target framework for your application. To specify the target framework in the project file it is using something called Target Framework Moniker (TFM).

In our example, the application targets the framework netcoreapp3.1. The netcoreapp3.1 is the Moniker for .NET Core 3.1. If you remember, we created this project with .NET Core 3.1.

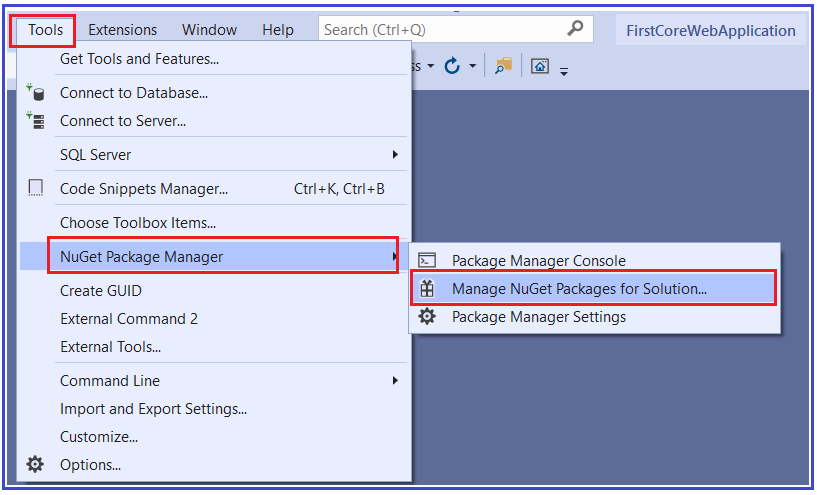
##### **Adding Packages in ASP.NET Core:**

As we already discussed ASP.NET Core Framework follows modular approaches. It means by default it will not include anything i.e. packages references to the project. Only the necessary packages which are required are added.

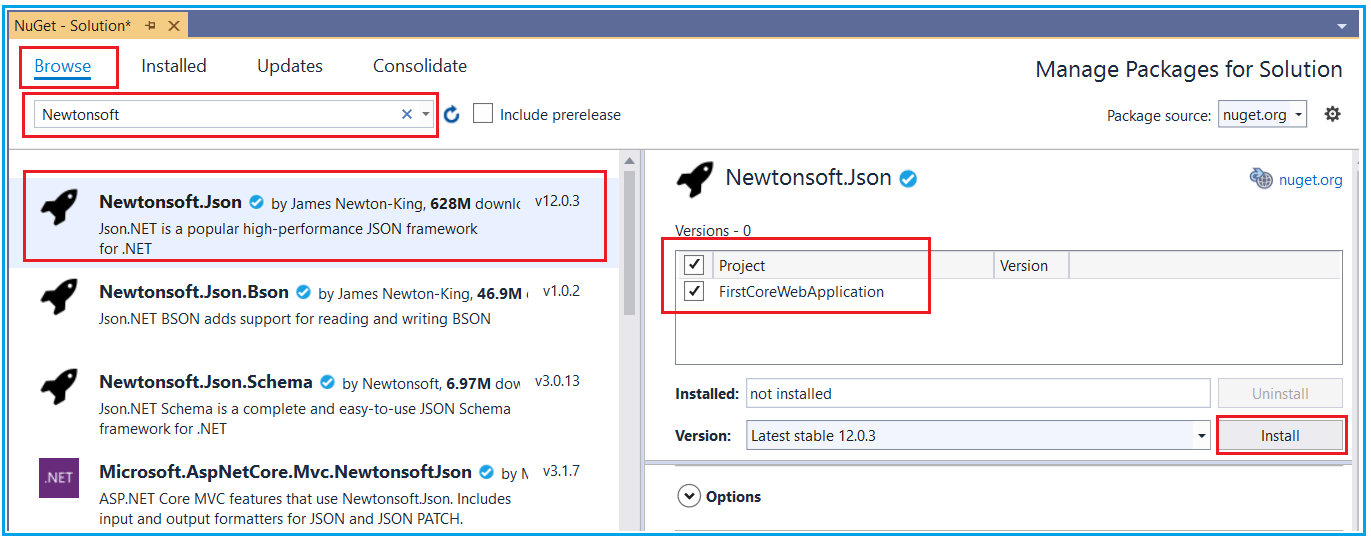
Whenever we add any new packages into our application, then that packages reference is also going to be added into the application project file. Please have a look at the following Dependencies section. Whenever we add any packages, then that package and its dependency packages are going to be stored here.



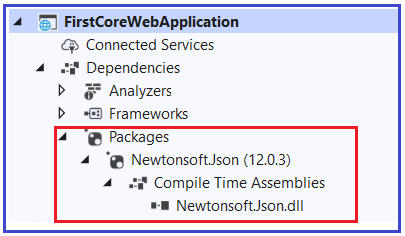
Let us first add a package from the NuGet Package Manager and see what happens. Go to tools => NuGet Package Manager => Manage NuGet Packages for Solution option as shown in the below image.



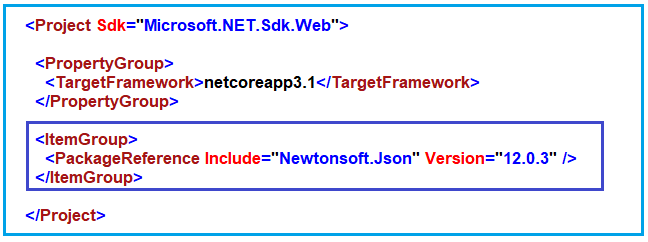
Now let us add the Newtonsoft.json package. So, select the browse tab and then search Newtonsoft and install it as shown in the below image.



Once the package is installed successfully, then you will see, it will add the reference inside the dependencies section as shown in the below image.



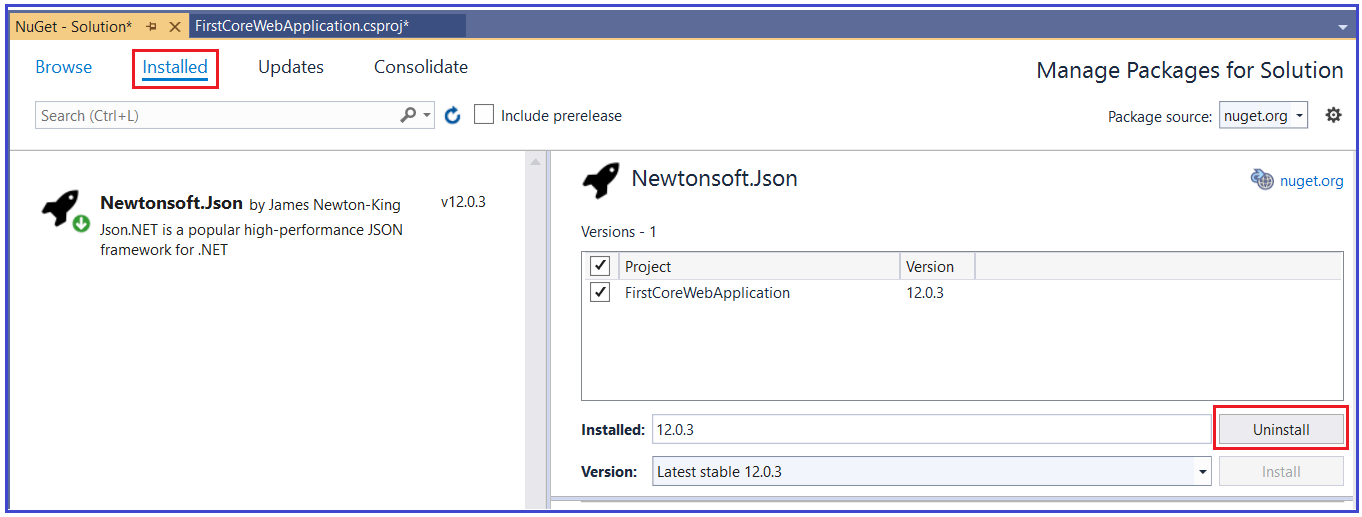
As we have only added one package i.e. Newtonsoft.json, so it added that package references here. Along with the same, it will also add that package reference in the application project file as shown in the below image.



**Note:** Once you delete that package, then it will delete the reference from both the dependencies as well as from the project file.

##### **Deleting Package in ASP.NET Core:**

Again, go to the Nuget Package Manager and select the installed tab and select the package and uninstall it as shown in the below image.



Once you delete the package, then it should remove the reference from both the dependencies as well as from the project file. The Package Reference element of the project file is used to add the references to all the NuGet packages which are installed for your application.

**Note:** The csproj file includes settings related to targeted .NET Core Framework, project folders, NuGet package references, etc…