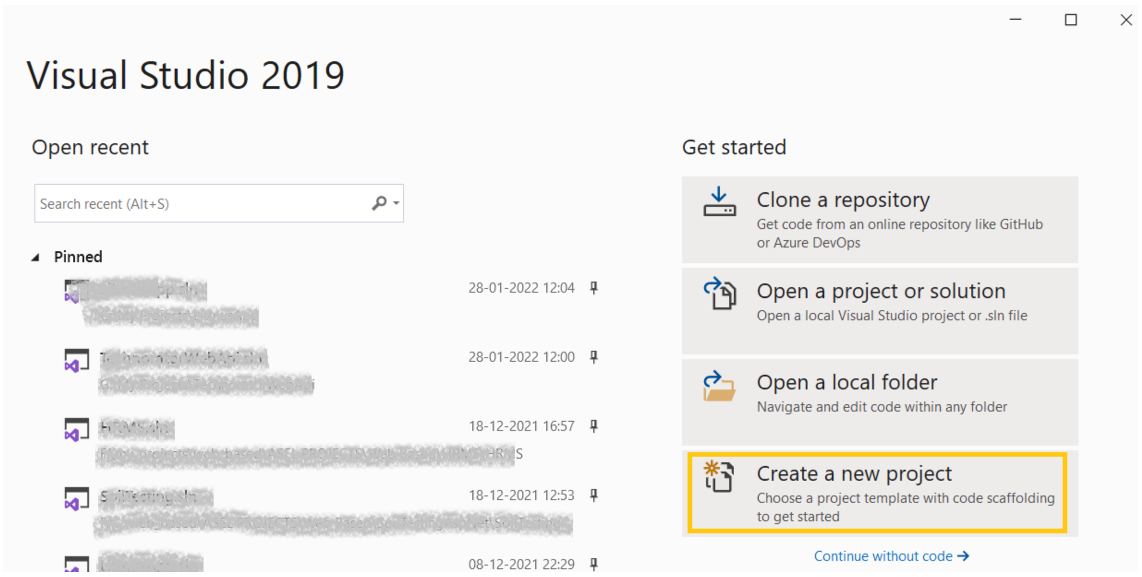
.Net Core 5.0 with Entity Framework Core (Database First Approach)

Open Visual Studio 2019 and click on ‘Create a New Project’.



Visual Studio 2019

Open recent

Search recent (Alt+S)

Pinned

[redacted]	28-01-2022 12:04
[redacted]	28-01-2022 12:00
[redacted]	18-12-2021 16:57
[redacted]	18-12-2021 12:53
[redacted]	08-12-2021 22:29

Get started

Clone a repository
Get code from an online repository like GitHub or Azure DevOps

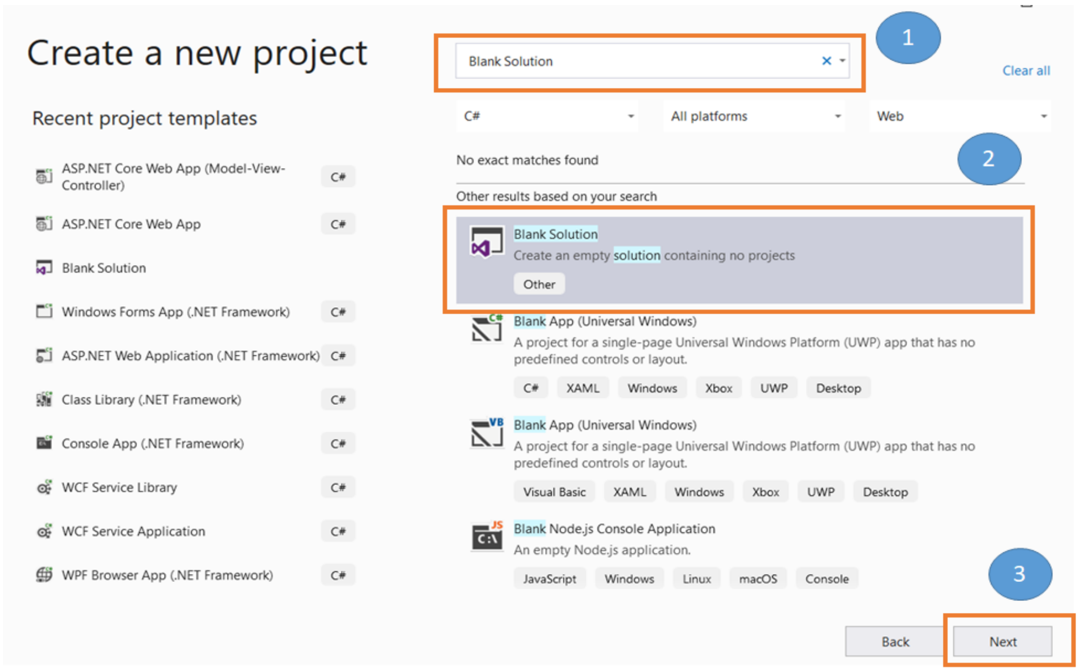
Open a project or solution
Open a local Visual Studio project or .sln file

Open a local folder
Navigate and edit code within any folder

Create a new project
Choose a project template with code scaffolding to get started

Continue without code →

Search for ‘Blank Solution’, select Blank Solution from the template and click on ‘Next’ button.



Create a new project

Blank Solution

Clear all

C# | All platforms | Web

No exact matches found

Other results based on your search

Blank Solution
Create an empty solution containing no projects
Other

Blank App (Universal Windows)
A project for a single-page Universal Windows Platform (UWP) app that has no predefined controls or layout.
C# XAML Windows Xbox UWP Desktop

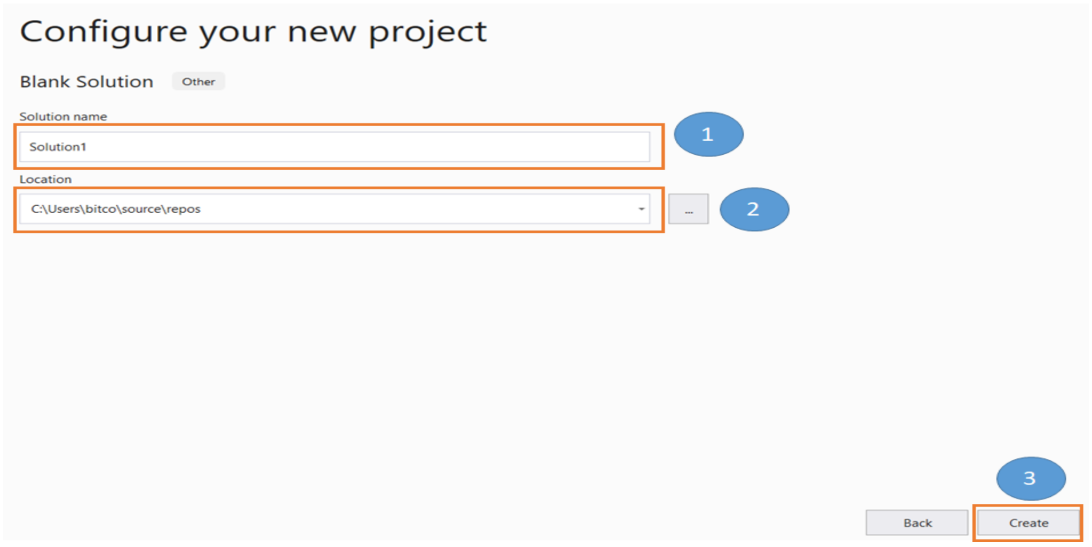
Blank App (Universal Windows)
A project for a single-page Universal Windows Platform (UWP) app that has no predefined controls or layout.
Visual Basic XAML Windows Xbox UWP Desktop

Blank Node.js Console Application
An empty Node.js application.
JavaScript Windows Linux macOS Console

Recent project templates

ASP.NET Core Web App (Model-View-Controller)	C#
ASP.NET Core Web App	C#
Blank Solution	
Windows Forms App (.NET Framework)	C#
ASP.NET Web Application (.NET Framework)	C#
Class Library (.NET Framework)	C#
Console App (.NET Framework)	C#
WCF Service Library	C#
WCF Service Application	C#
WPF Browser App (.NET Framework)	C#

Give an appropriate name to your solution, select a location where you want it to save and click on 'Create' button. This will create a blank solution. In this solution we will add our projects.

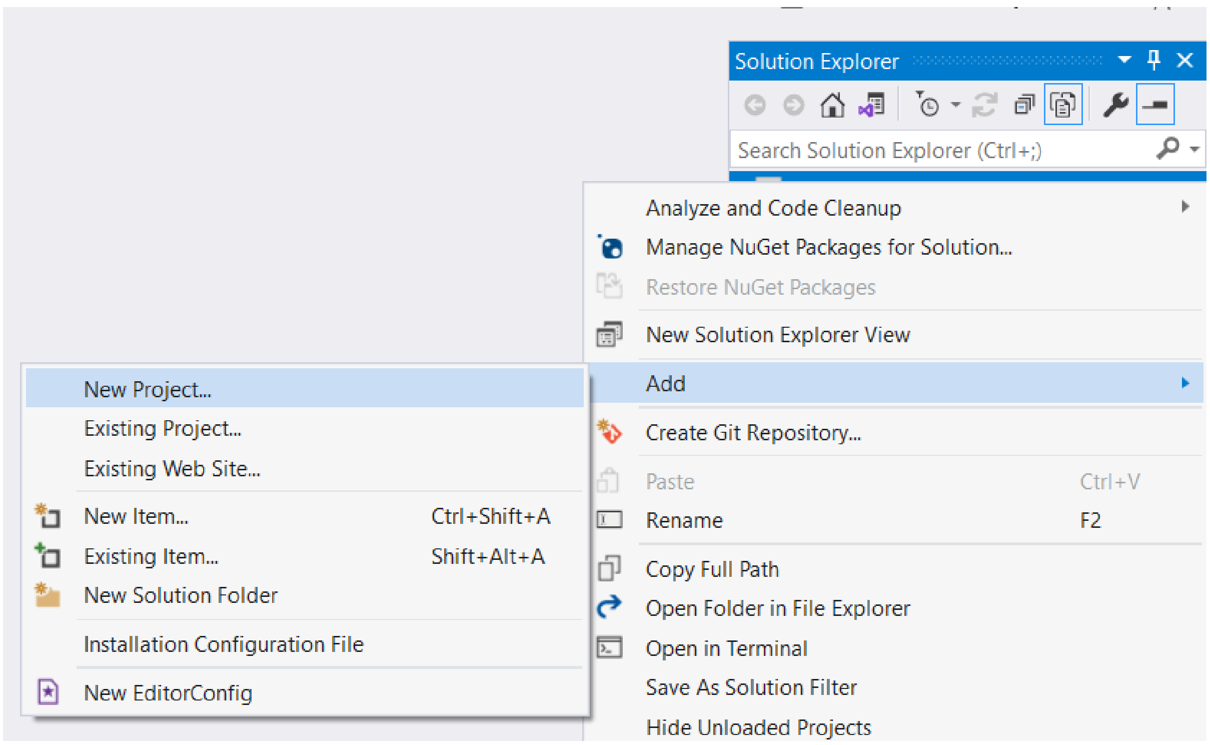

Configure your new project

Blank Solution Other

Solution name: Solution1

Location: C:\Users\bitco\source\repos

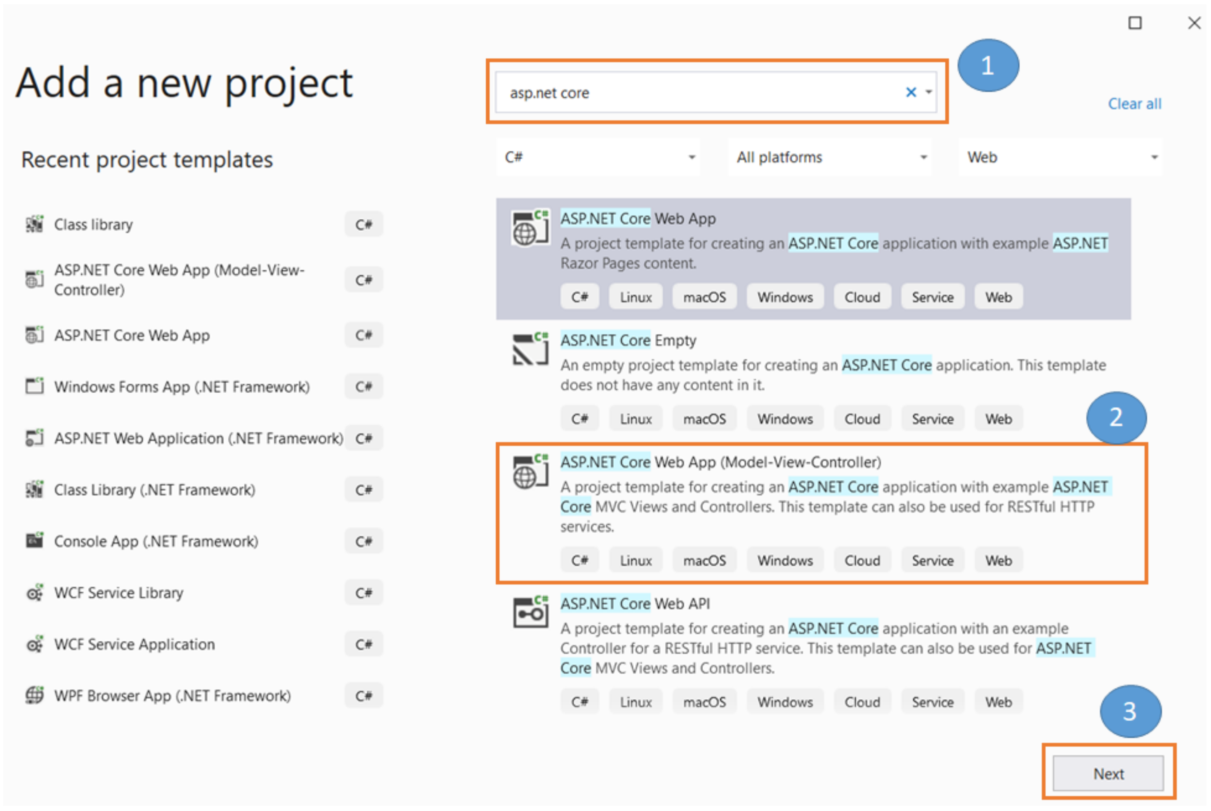
Right click on the solution and click on 'Add New Project'



A new window will open to select project template/type.

Search for 'asp.net core' and from the sorted list of templates, select 'ASP.NET CORE Web App (Model-View-Controller). Also, make sure you have selected project with right programming language, in our case it's **c#**.

Once you selected the template click on Next Button.


Add a new project

asp.net core

Clear all

C# | All platforms | Web

Recent project templates

- Class library C#
- ASP.NET Core Web App (Model-View-Controller) C#
- ASP.NET Core Web App C#
- Windows Forms App (.NET Framework) C#
- ASP.NET Web Application (.NET Framework) C#
- Class Library (.NET Framework) C#
- Console App (.NET Framework) C#
- WCF Service Library C#
- WCF Service Application C#
- WPF Browser App (.NET Framework) C#

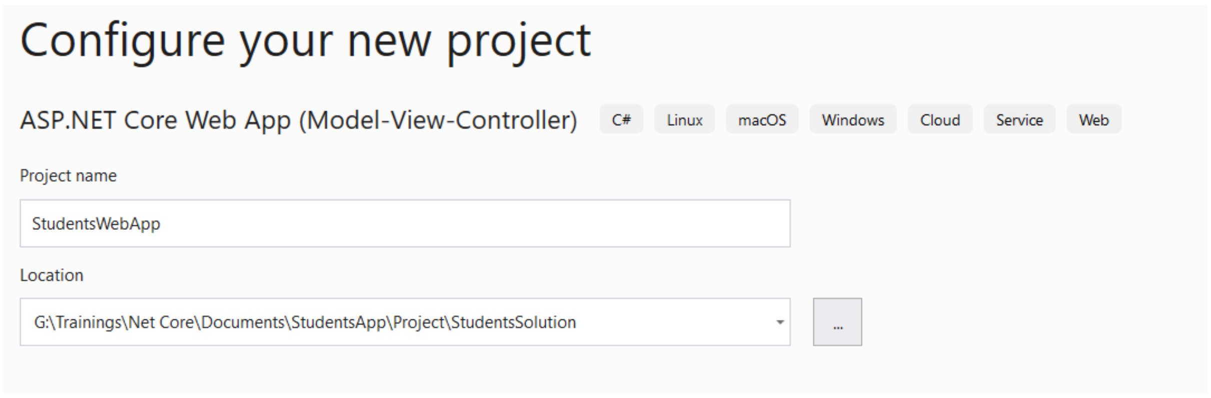
ASP.NET Core Web App
A project template for creating an ASP.NET Core application with example ASP.NET Razor Pages content.
C# Linux macOS Windows Cloud Service Web

ASP.NET Core Empty
An empty project template for creating an ASP.NET Core application. This template does not have any content in it.
C# Linux macOS Windows Cloud Service Web

ASP.NET Core Web App (Model-View-Controller)
A project template for creating an ASP.NET Core application with example ASP.NET Core MVC Views and Controllers. This template can also be used for RESTful HTTP services.
C# Linux macOS Windows Cloud Service Web

ASP.NET Core Web API
A project template for creating an ASP.NET Core application with an example Controller for a RESTful HTTP service. This template can also be used for ASP.NET Core MVC Views and Controllers.
C# Linux macOS Windows Cloud Service Web

In next window, you will be asked to 'Configure your new project', just give a meaning full name to your project and leave the location as it is (it is same location where you saved your solution). Click on 'Next' button.


Configure your new project

ASP.NET Core Web App (Model-View-Controller) C# Linux macOS Windows Cloud Service Web

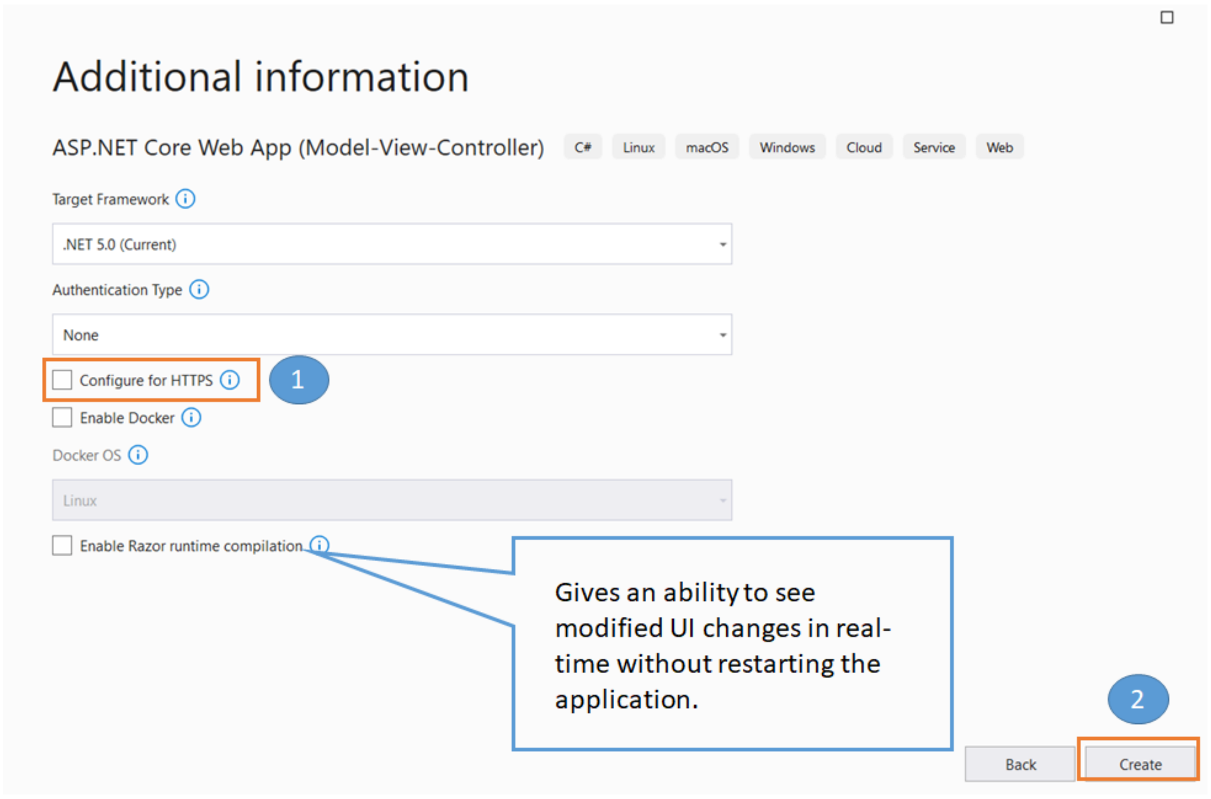
Project name

StudentsWebApp

Location

G:\Trainings\Net Core\Documents\StudentsApp\Project\StudentsSolution

After, clicking on next button, a new window will open to setup some additional but important setting for the project.


Additional information

ASP.NET Core Web App (Model-View-Controller) C# Linux macOS Windows Cloud Service Web

Target Framework: .NET 5.0 (Current)

Authentication Type: None

☐ Configure for HTTPS

☐ Enable Docker

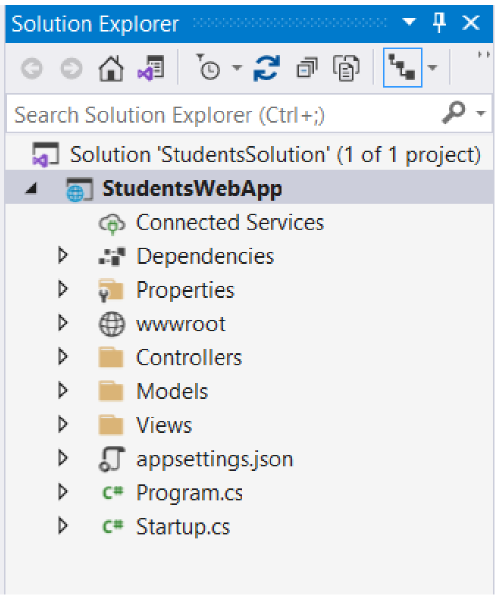
Docker OS: Linux

☐ Enable Razor runtime compilation

1
Gives an ability to see modified UI changes in real-time without restarting the application.
2
Back
Create

Select an appropriate .Net version, leave authentication type 'None' for the project (available options are individual account, Microsoft Identity Platform and Windows) and un-check 'Configure for HTTPS'. Click on 'Create' button.

Now your solution explorer will look similar to below image.



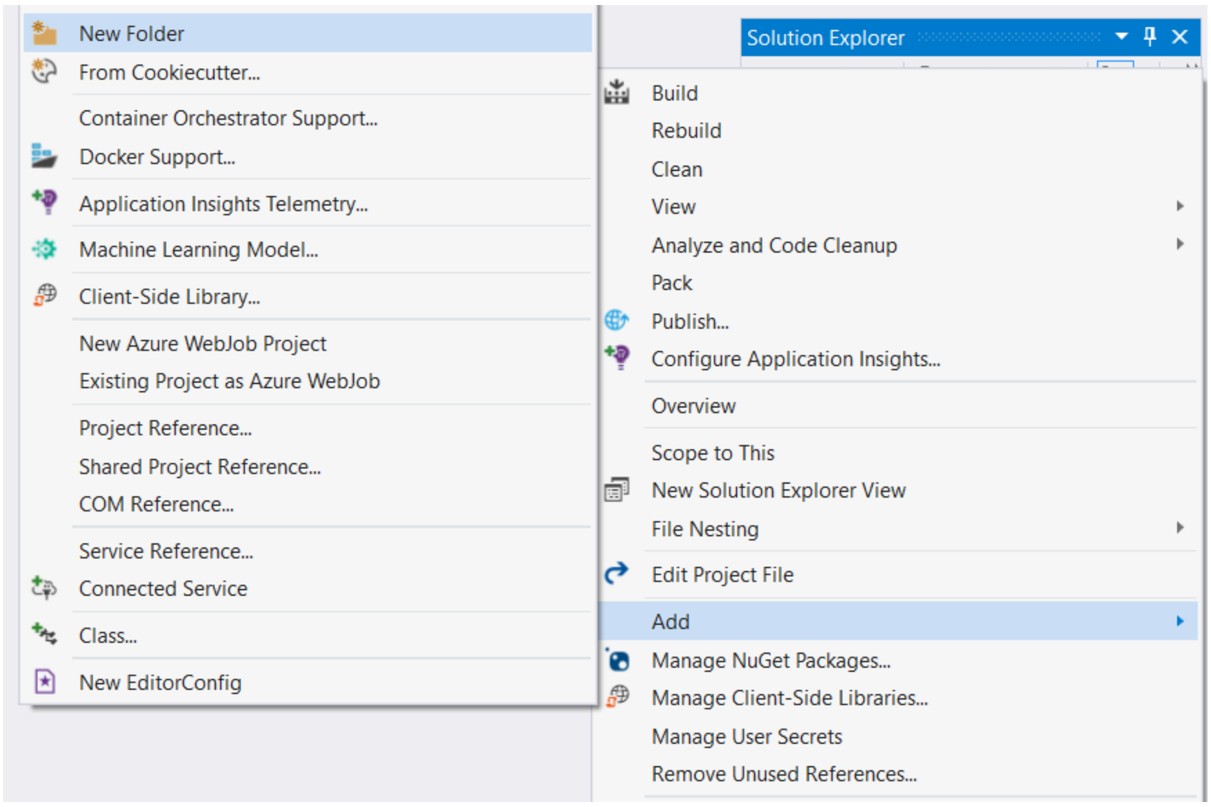
```
Solution 'StudentsSolution' (1 of 1 project)
StudentsWebApp
    Connected Services
    Dependencies
    Properties
    wwwroot
    Controllers
    Models
    Views
    appsettings.json
    Program.cs
    Startup.cs
```

Folder Structure

Folder / File Name	Description
Wwwroot	Contains static files of the project.
Controller	
Models	
Views	
Appsettings.json	
Program.cs	
Startup.cs	

Add a new folder to the project and name it as 'DAL', in this folder we will keep our database related operations/logics.

Right Click on Project -> Add -> New Folder



NuGet Packages for the project.

In this project we will require several 'NuGet' package List of packages and details are as under

Package Name	Details	Version
Microsoft.EntityFrameworkCore	Entity Framework Core is a modern object-database mapper for .NET. It supports LINQ queries, change tracking, updates, and schema migrations.	5.0.13
Microsoft.EntityFrameworkCore.sql	Microsoft SQL Server database provider for Entity Framework Core.	5.0.13
Microsoft.EntityFrameworkCore.Tool	Enables these commonly used commands: Add-Migration Bundle-Migration Drop-Database Get-DbContext Get-Migration Optimize-DbContext Remove-Migration Scaffold-DbContext Script-Migration Update-Database	5.0.13

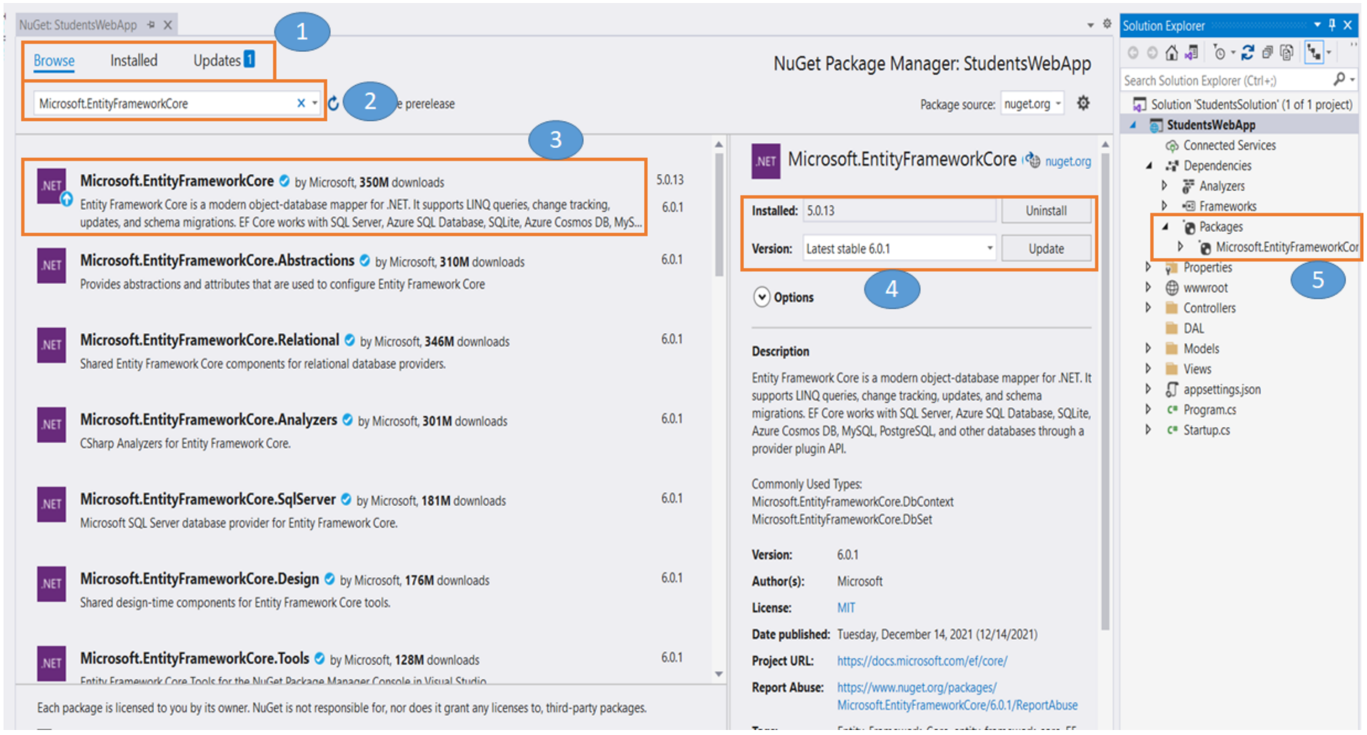
To install these package, follow below steps.

1. Right click on your project
2. Click on 'Manage NuGet Packages…'.
3. Select 'Browse' tab
4. Search for the package by typing package name in search box.

5. From the left hand side package list, select the appropriate package from the list.
6. On the right hand side select a version of the package and click on install.
7. It may show a window ‘Preview Changes’ click on OK.
8. On next window, click on **I accept**.

NOTE: - Make sure all package version must be same or support each other.

Once the packages are installed successfully, all these package will be visible in ‘Installed’ tab of NuGet package manager, also, you can see them under **Dependencies -> Package** in your solution.
For reference, NuGet package manager window after installing ‘Microsoft.EntityFrameworkCore’ package will look similar to below image.


NuGet: StudentsWebApp
1
Browse
Installed
Updates 1
2
3
4
5
NuGet Package Manager: StudentsWebApp
Package source: nuget.org
prerelease
Solution Explorer
Search Solution Explorer (Ctrl+;)

Microsoft.EntityFrameworkCore

Microsoft.EntityFrameworkCore by Microsoft, **350M** downloads — 5.0.13 / 6.0.1
Entity Framework Core is a modern object-database mapper for .NET. It supports LINQ queries, change tracking, updates, and schema migrations. EF Core works with SQL Server, Azure SQL Database, SQLite, Azure Cosmos DB, MyS...

Microsoft.EntityFrameworkCore.Abstractions by Microsoft, **310M** downloads — 6.0.1
Provides abstractions and attributes that are used to configure Entity Framework Core

Microsoft.EntityFrameworkCore.Relational by Microsoft, **346M** downloads — 6.0.1
Shared Entity Framework Core components for relational database providers.

Microsoft.EntityFrameworkCore.Analyzers by Microsoft, **301M** downloads — 6.0.1
CSharp Analyzers for Entity Framework Core.

Microsoft.EntityFrameworkCore.SqlServer by Microsoft, **181M** downloads — 6.0.1
Microsoft SQL Server database provider for Entity Framework Core.

Microsoft.EntityFrameworkCore.Design by Microsoft, **176M** downloads — 6.0.1
Shared design-time components for Entity Framework Core tools.

Microsoft.EntityFrameworkCore.Tools by Microsoft, **128M** downloads — 6.0.1
Entity Framework Core Tools for the NuGet Package Manager Console in Visual Studio

Each package is licensed to you by its owner. NuGet is not responsible for, nor does it grant any licenses to, third-party packages.

Microsoft.EntityFrameworkCore nuget.org

Installed: 5.0.13 Uninstall
Version: Latest stable 6.0.1 Update

Options

Description
Entity Framework Core is a modern object-database mapper for .NET. It supports LINQ queries, change tracking, updates, and schema migrations. EF Core works with SQL Server, Azure SQL Database, SQLite, Azure Cosmos DB, MySQL, PostgreSQL, and other databases through a provider plugin API.

Commonly Used Types:
Microsoft.EntityFrameworkCore.DbContext
Microsoft.EntityFrameworkCore.DbSet

Version: 6.0.1
Author(s): Microsoft
License: MIT
Date published: Tuesday, December 14, 2021 (12/14/2021)
Project URL: https://docs.microsoft.com/ef/core/
Report Abuse: https://www.nuget.org/packages/Microsoft.EntityFrameworkCore/6.0.1/ReportAbuse

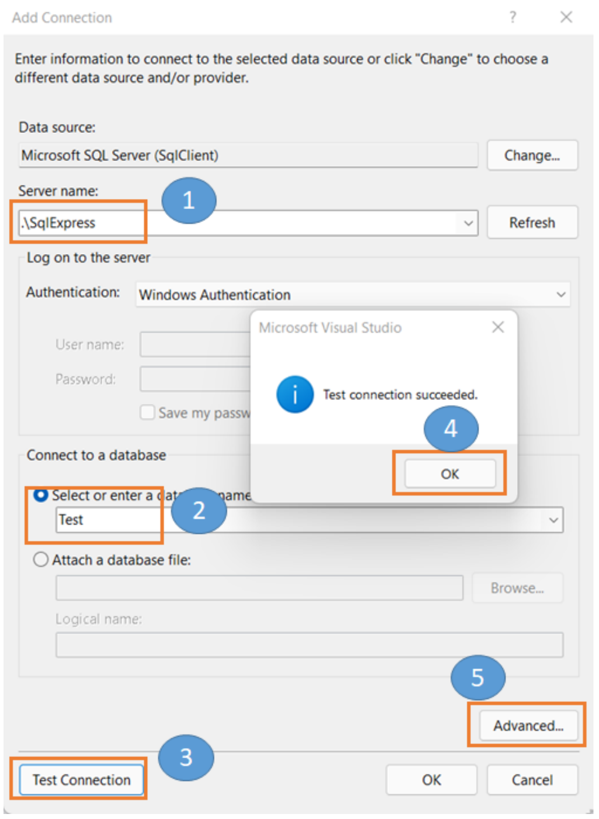
Solution 'StudentsSolution' (1 of 1 project)
- StudentsWebApp
  - Connected Services
  - Dependencies
    - Analyzers
    - Frameworks
    - Packages
      - Microsoft.EntityFrameworkCor
  - Properties
  - wwwroot
  - Controllers
  - DAL
  - Models
  - Views
  - appsettings.json
  - Program.cs
  - Startup.cs

Database Configuration

To get connection string of the database, go to ‘Tools’ menu and click on ‘Connect to Database…’.

Provide Server Name, Authentication type, user credentials (if any), Database name. Click on **Test Connection**, click on OK, if connection was successful, click on advance button and copy Connection string from there.

Now, in solution explorer, open **appsettings.json** file and connection string as shown in below image.



Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

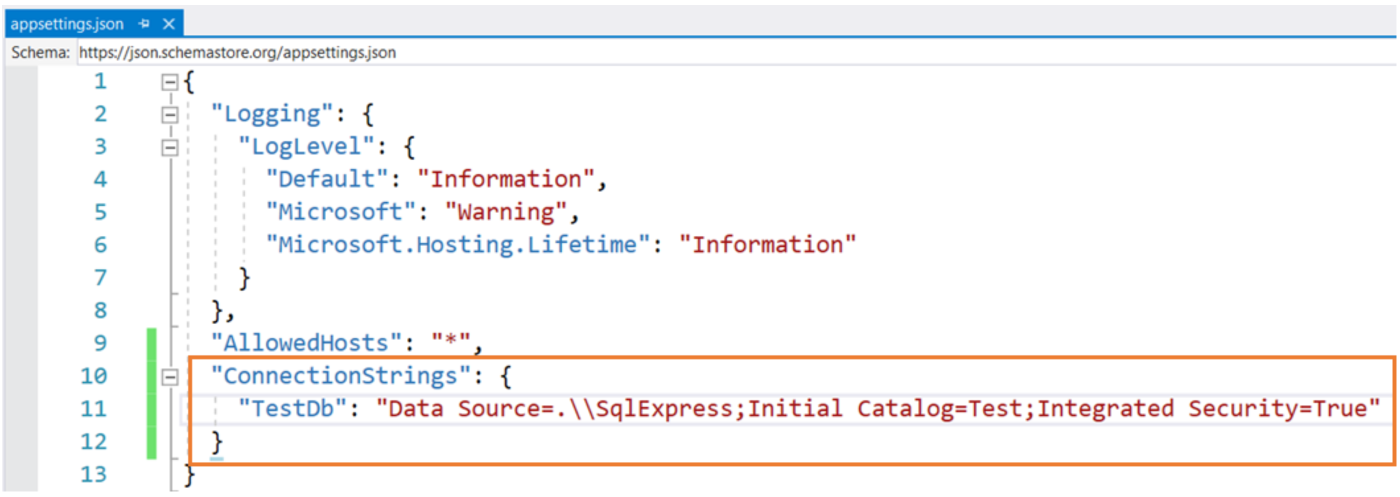
Data source:
Microsoft SQL Server (SqlClient)

Server name:
.\SqlExpress

Log on to the server
Authentication: Windows Authentication
User name:
Password:
Save my passw

Connect to a database
Select or enter a data... name:
Test
Attach a database file:
Logical name:

Microsoft Visual Studio
Test connection succeeded.
OK

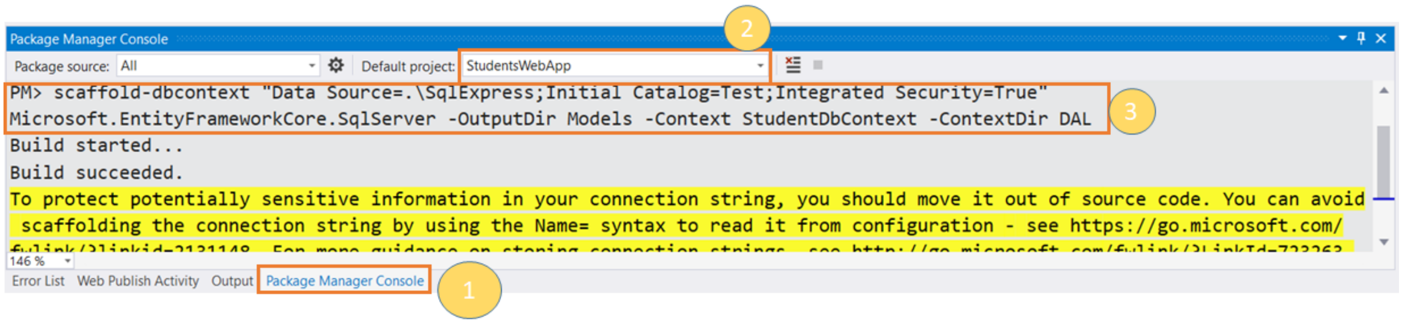


Schema: https://json.schemastore.org/appsettings.json

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Information"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "TestDb": "Data Source=.\\SqlExpress;Initial Catalog=Test;Integrated Security=True"
  }
}
```

Open ‘Package Manager Console’ window and execute below command

`scaffold-dbcontext "Data Source=.\SqlExpress;Initial Catalog=Test;Integrated Security=True" Microsoft.EntityFrameworkCore.SqlServer -OutputDir Models -Context DAL`



Package source: All
Default project: StudentsWebApp

```
PM> scaffold-dbcontext "Data Source=.\SqlExpress;Initial Catalog=Test;Integrated Security=True"
Microsoft.EntityFrameworkCore.SqlServer -OutputDir Models -Context StudentDbContext -ContextDir DAL
Build started...
Build succeeded.
To protect potentially sensitive information in your connection string, you should move it out of source code. You can avoid
 scaffolding the connection string by using the Name= syntax to read it from configuration - see https://go.microsoft.com/
fwlink/?linkid=2131148. For more guidance on storing connection strings, see http://go.microsoft.com/fwlink/?LinkId=723263.
```

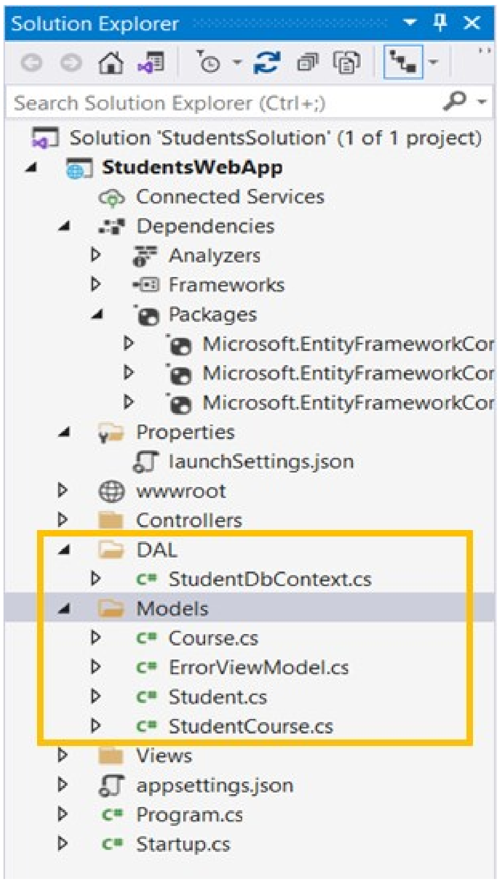
146 %

What is in above Command

`scaffold-dbcontext <DbConnectionString> <Provider> -OutputDir <DirectoryName> -ContextDir <DirectoryName> -Context <ContextFileName>`

Scaffold-dbcontext	Is command for scaffolding
`<DbConnectionString>`	It is connection string of database (only single backslah is there in command)
`<Provider>`	It is Database Provider
`-OutputDir <DirectoryName>`	The directory use to output the files. Paths are relative to the target project directory. Defaults to "Migrations".
`-ContextDir <DirectoryName>`	The directory to put the `DbContext` file in. Paths are relative to the project directory
`-Context <ContextFileName>`	Name of DbContext File.

After successful execution of the command, folder structure in solution explorer will look as below image.

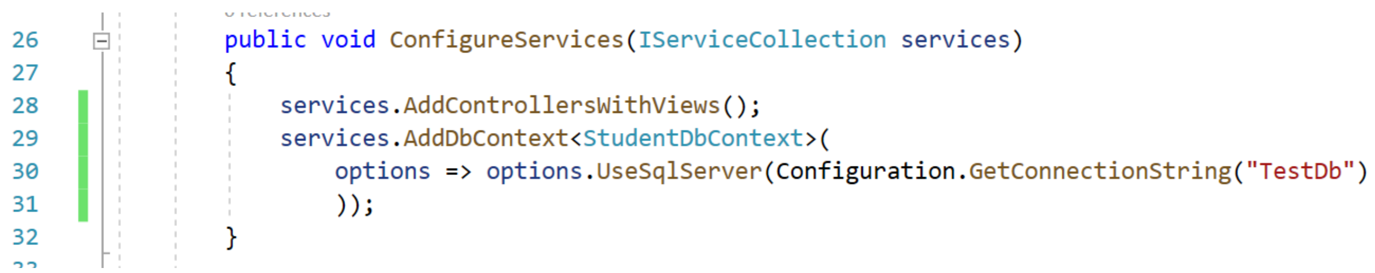


Now, our project has one warning. Basically it is stating that we should not keep our sensitive information like connection string in class file or source code files.

When we executed 'scaffold-dbcontext' command, it saved database connection string into 'StudentsDbContext' file. Let's delete it from there.

Now, let's open 'Startup.cs' file and DbContext to services.

Import
```
using StudentsWebApp.DAL;
using Microsoft.EntityFrameworkCore;
```



```
public void ConfigureServices(IServiceCollection services)
{
    services.AddControllersWithViews();
    services.AddDbContext<StudentDbContext>(
        options => options.UseSqlServer(Configuration.GetConnectionString("TestDb")
        ));
}
```