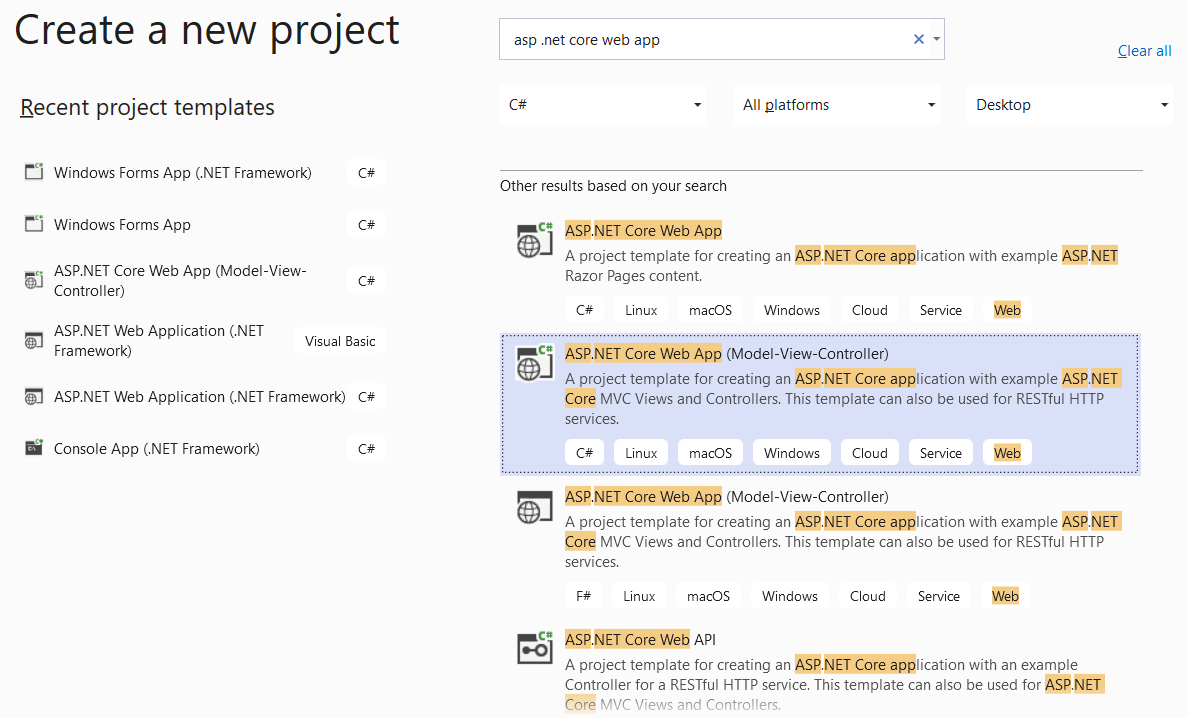
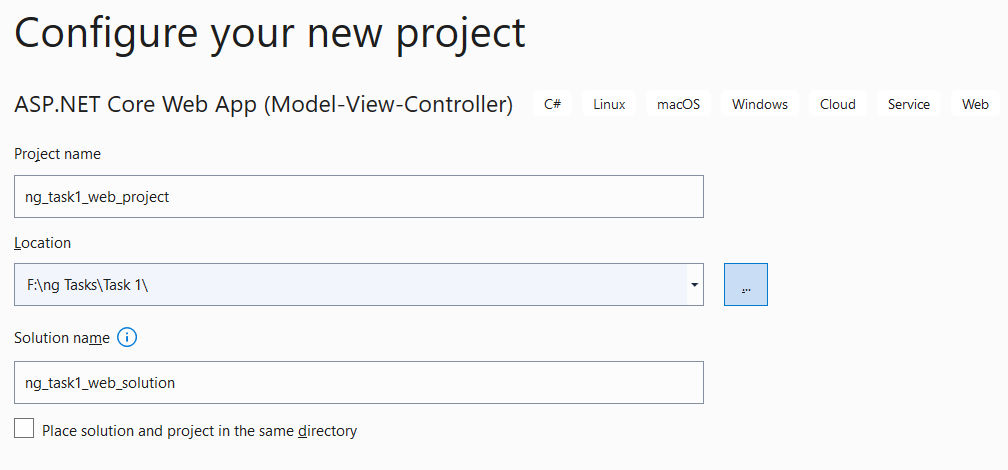
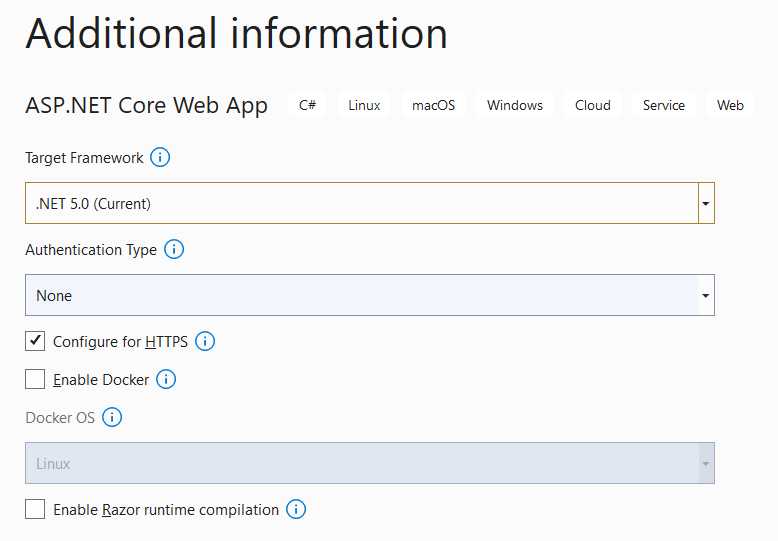
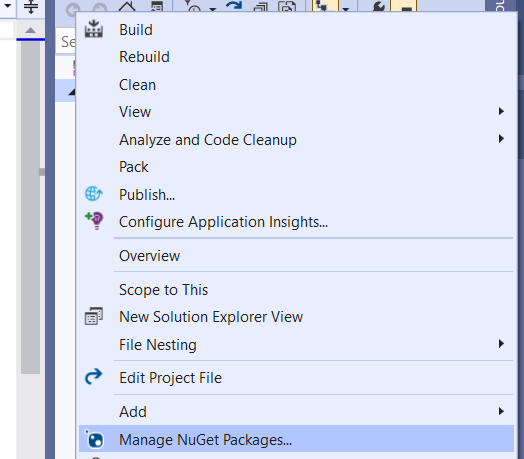
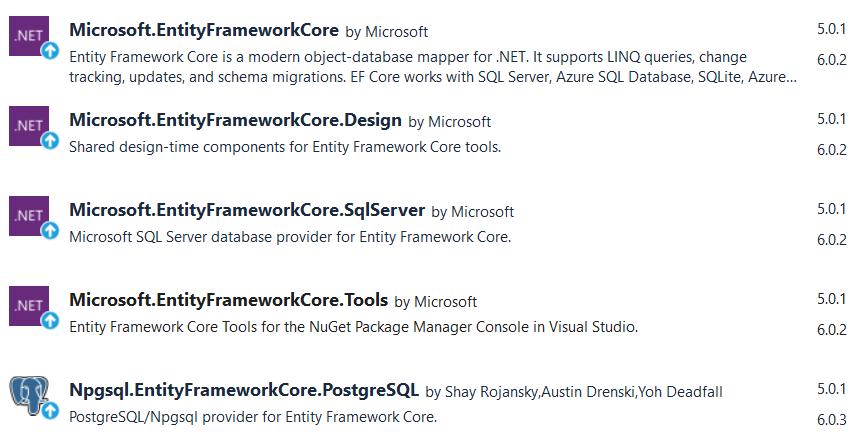
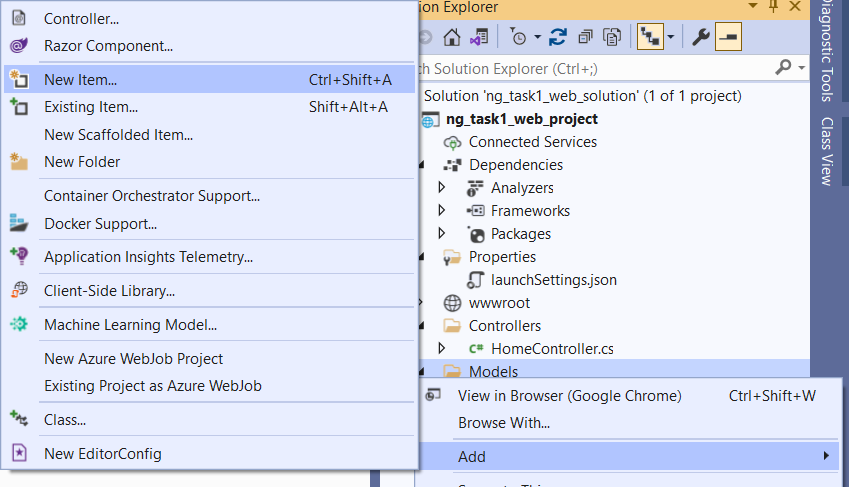
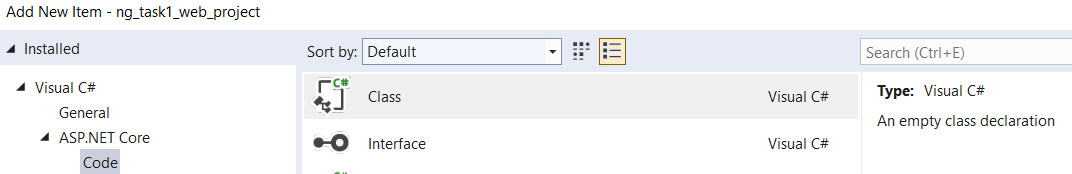
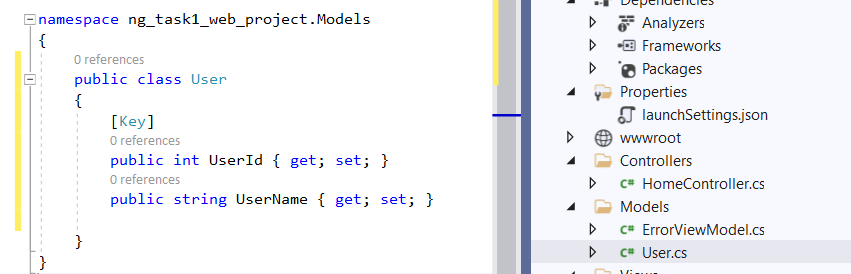
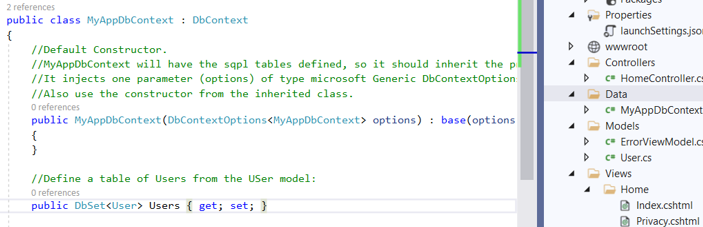
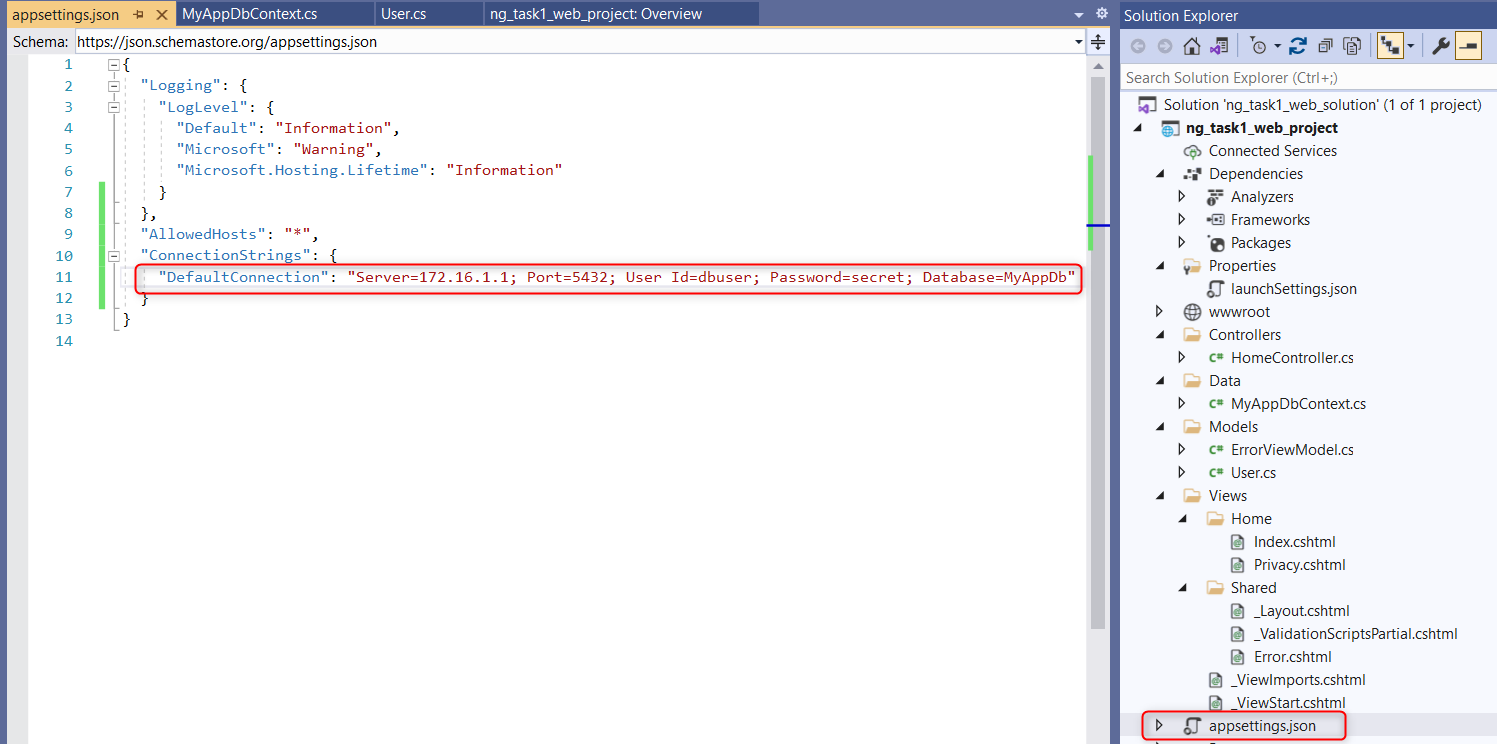
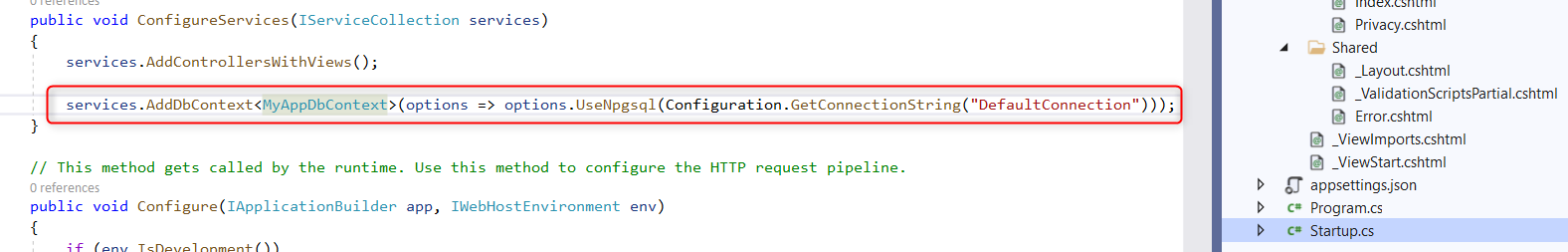
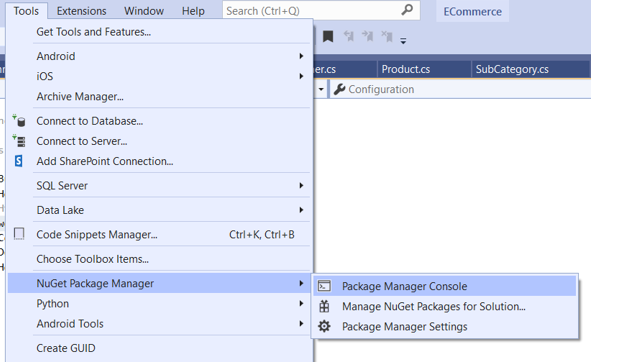
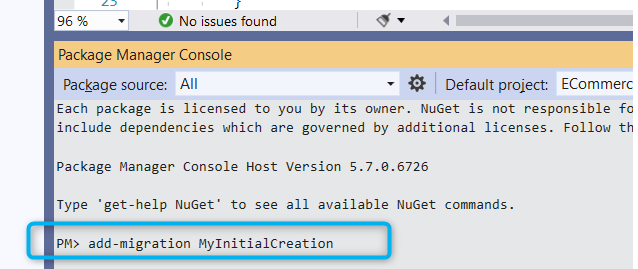
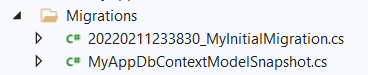
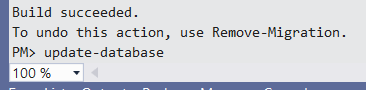
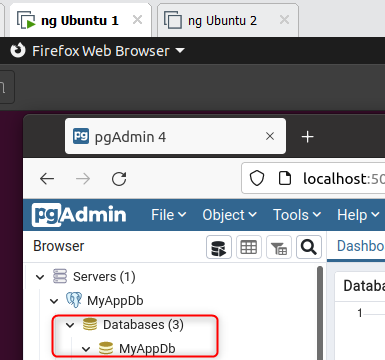
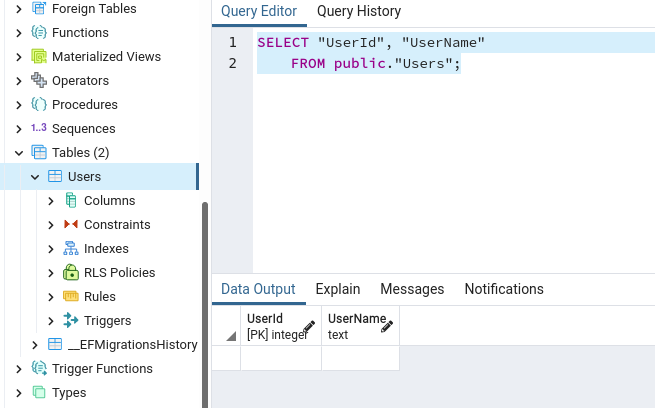
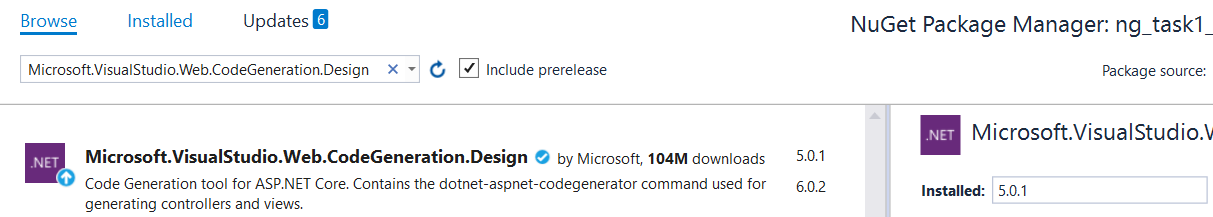
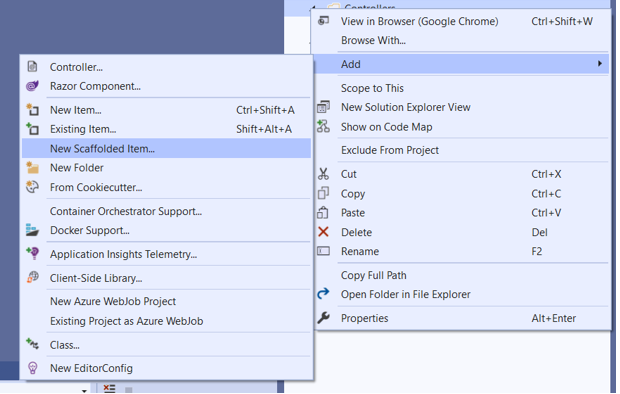
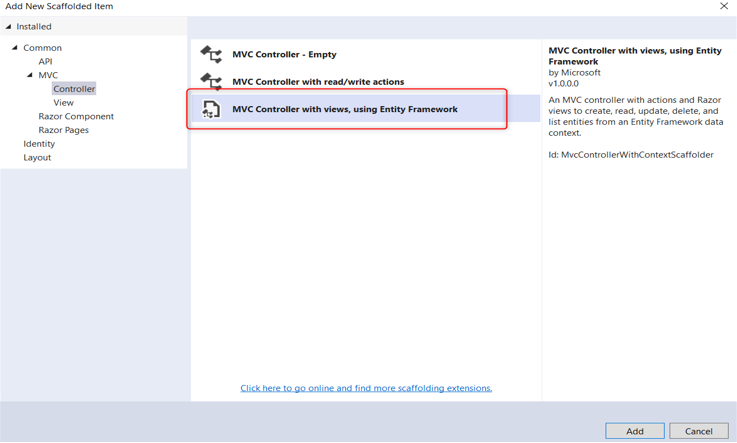
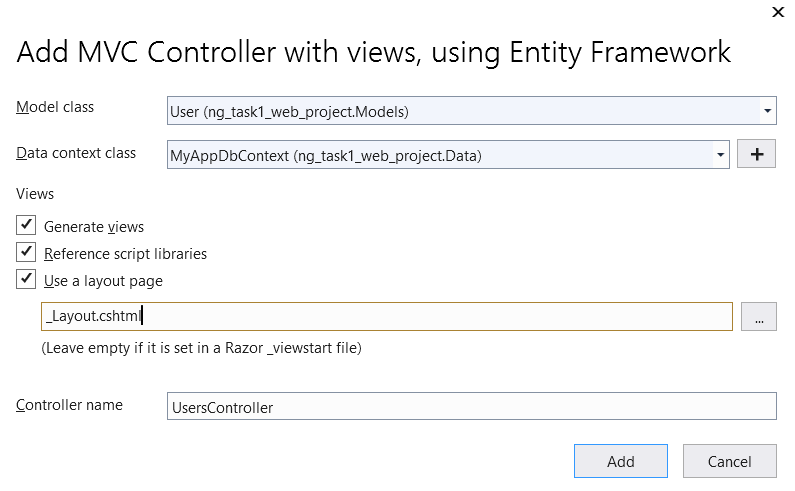
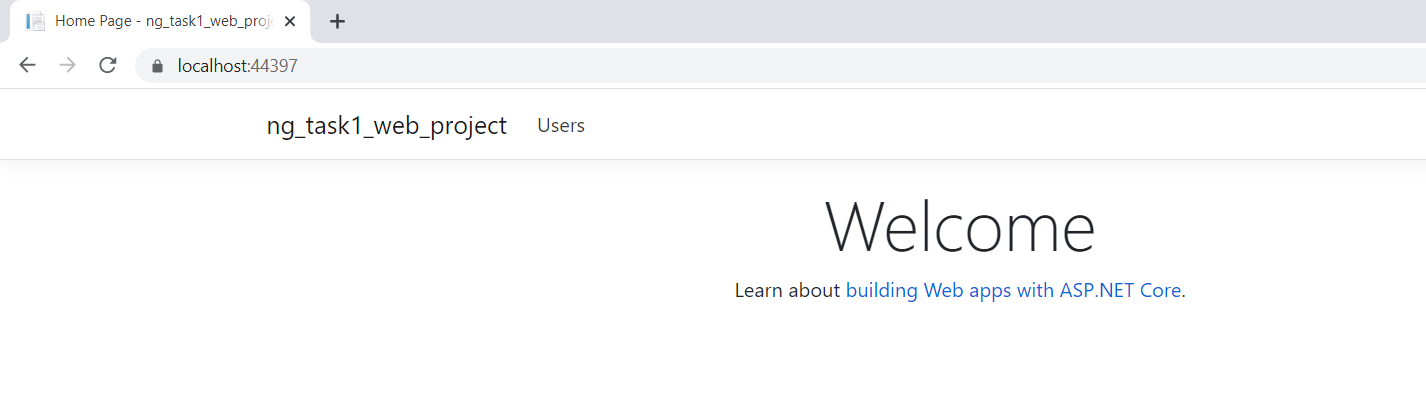
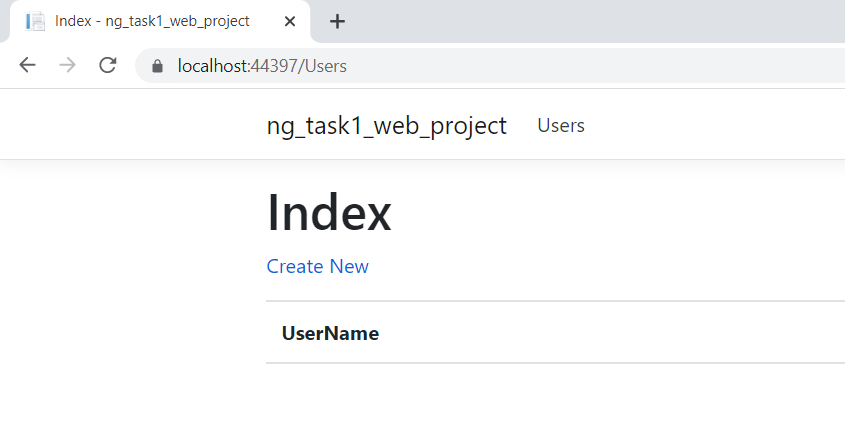
1. Open Visual Studio 2019 as an administrator.
2. The application will be a GUI with a button to open a table, then perform CRUD (create, read, update, delete) operations on it.
3. Choose:
   1. 
   2.   
        
      This is an MVC template project. It is one project where we have frontend and backend with one project. I could have for example developed angular frontend Restful API consumer, then .NET 5.0 backend Restful API producer, then containerized each one alone. But will skip that and use one simple project.
   3. Microsoft new web frameworks in order: .NET Core 2.2 🡪 .NET Core 3.1 🡪 .NET 5.0  
      (5.0 does not have the keyword "Core") (do not choose enable docker now)  
      
4. Install required dependencies:  
   we will use the "Code First" concept which means no sql commands will be used, no direct interaction with sql will be done, thus, saving a lot of time and effort.  
   The database schema will be created by C# with EntityFrameWork.   
   Then migrated to the sql DB to get created as a table there.
   1. Right click on the project name on the right panel of visual studio, then:  
      Go to NuGet package manager to install the following packages:  
      
   2. Make sure those are installed with the following versions 5.0.1:  
      
5. Create a database model:
   1. Add the following to the models folder:  
        
      Then:  
      
   2. Define the schema as the following:  
        
      This illustrates what a one record in an sql table should have.  
      UserId will be the sql primary key.
   3. Create the DbContext in which we define a table of a model:  
      
6. Now create this table in the PostgreSQL using C# EntityFrameWork:  
   we have 3 steps:  
   -Define the connection string with the SQL server.

-Add my Context (DB) as a service in the Startup.cs file.

-From NuGet package manager, do the migration process to create this DB

* 1. Connection string:  
     -This application will run on ng-vm2, and the database will be in ng-vm1, so I added ng-vm1 IP here. A better practice is to write ng-vm1 hostname, then resolve it to an IP in /etc/hosts file in ng-vm2 where this application will run. But I will skip that.  
     -Also, this application is being developed on my host (laptop) (Windows 10) and remember that it is also connected to ng-vm1 using vNet3 as explained before, so the sql DB is reachable from here as well.   
     -The connection string has the DB username and password. So usually we add this connection string during development phase in appsettingsDevelopment.json instead of appsettings.json, and we do not commit that file to GitHub. Anyway, I will skip those recommendations for now.   
     
  2. Add a service in the application services container to define an sql service using the DbContext we created before and the connection string we have, in the startup.cs:  
     
  3. Now, to create the sql table in the sql DB which is running in ng-vm1, do the process of "Migration":  
       
       
       
     Then Migration files are created:  
       
       
     Now perform the update-database command from the same place:  
       
       
       
     we go to ng-vm1 pgadmin, and notice the database is created in the PostgreSQL:  
       
       
       
     Table:  
       
     

1. Now, we need to control that sql table from the web app, we will create a controller for that table, with Create, Read, Update and Delete Views to do those operations on that table:
   1. First, install this package from NuGet package manager:  
      
   2. In the controller folder:  
        
        
        
        
        
      (  
      When you press "Add", the visual studio might not perform adding a control successfully, so use ..NET (dotnet) cli using PowerShell:  
      cd to the project file, then  
      >dotnet aspnet-codegenerator controller --model User --dataContext MyAppDbContext --layout ~/Views/Shared/\_Layout.cshtml --force --controllerName UsersController --useAsyncActions -outDir Controllers  
      )
   3. Create a button for it in the \_Layout.cshtml, and remove the old buttons for the default pages of Home and Privacy:  
      
2. Run the project using the visual studio built in IIS Express web server:  
     
   click on Users:  
     
     
   This will be opened:  
   