**To check dotNet version: -**

dotnet --version #cmd

**To Check angular version: -**

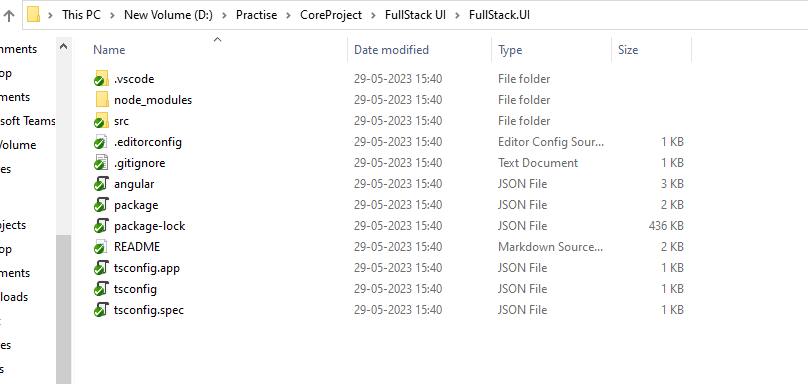
ng version #cmd

**To Check node version: -**

node -v #cmd

**To Create a new angular project: -**

ng new ProjectName #cmd



All the files will be created.

Now go in the UI project that we created.

A picture containing text, font, screenshot

Description automatically generated

**To open visual studio code, write: -**

code . #cmd

A screenshot of a computer

Description automatically generated

The starting point of our project will be ‘index.html’ file.

The index.html file contains an app root element which is also present in ‘app.component.ts’ file’s selector component.

**Open the terminal and write in vs code: -**

To run the project

ng serve #cmd

After running the project, the html that we see is from app.component.html file. So, to make our application we need to clear the preexisting code leaving the router outlet element which is there at the end of this file. We will use this element later for routing.

**To add css in our project: -**

Go to Bootstrap 5 website and copy the css and bundle scripts and paste them both in index.html file after the links which are already there.

Add navbar from bootstrap into app.component.html file. Do the minor changes required in it for UI.

**To create a new component: -**

ng g c componentName #cmd

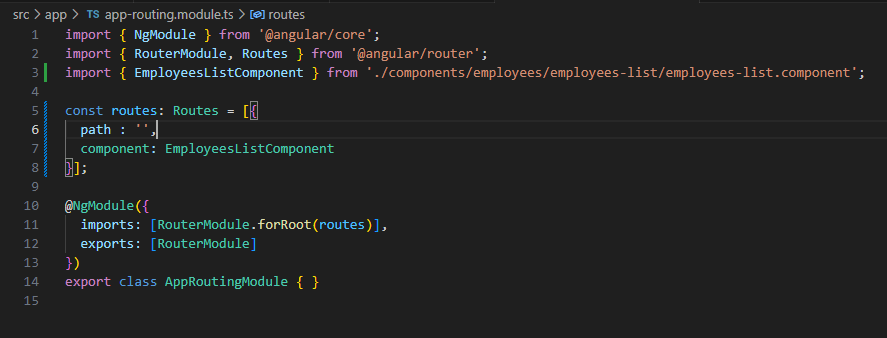
where,

g -> generate

c -> component

**To add our component** on home page we will have to give routing in app-routing.module.ts file

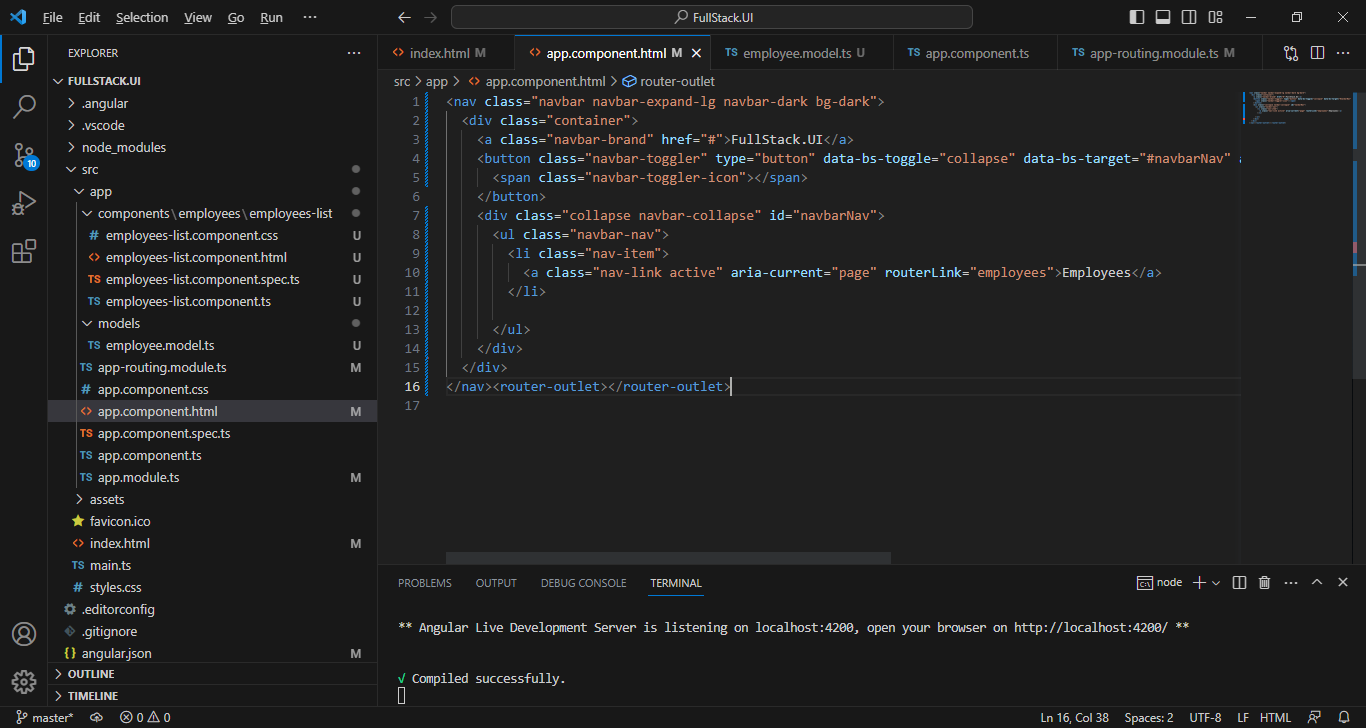
Where we will add two things, one is the path which is in string format and the other is the component name.



The above code means that when we open the empty path provided that is our home page, then Employee Component should be called.

**How to link Navbar components to other page links?**

In app.component.html if u have given other navbar components on which you have to provide routing then in the html element add routerLink for the same, then in app-routing.module.ts add the routing in component section as we did previously.

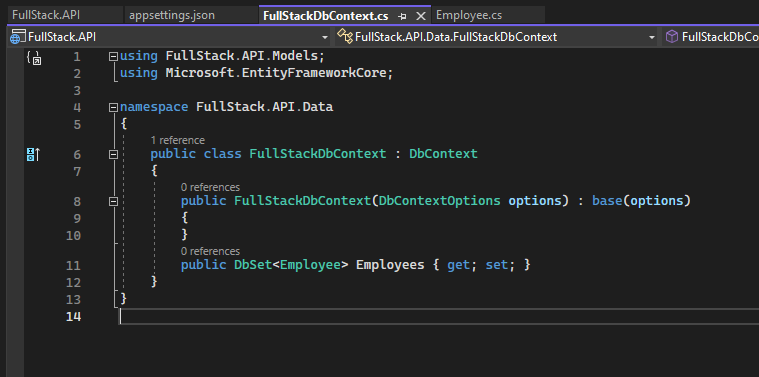


A screenshot of a computer program

Description automatically generated with medium confidence

**API Project**

1. Install Visual Studio 2022.
2. Create New Project.
3. Select ASP .NET Core Web API.
4. Fill the rest of the details.
5. Go to Dependencies > Manage Nuget Packages > Search
   1. Microsoft.EntityFrameworkCore.SqlServer
   2. Microsoft.EntityFrameworkCore.Tools
6. Add new folder (Models) in project.
7. Add a new class (Employee.cs) in that folder.
8. Add our properties in that class which we have defined in our UI project.
9. To remove the squiggles that we see under our properties, open the project properties file (double click on our project name) and in Nullable tag write disable.
10. To add a db context which is used for database connectivity we will create a new folder “Data” in our project folder.
11. In this folder add a new class named FullStackDbContext.
12. In this class we call a constructor and add a property of type DbSet. In this property we then give it type Employee which we made in Models folder. Entity framework will look at this property as a whole table and we will be able to access it altogether.



**To create a new server on our local system: -**

Write in cmd

sqllocaldb create "dbnamehere" #cmd

A screenshot of a computer

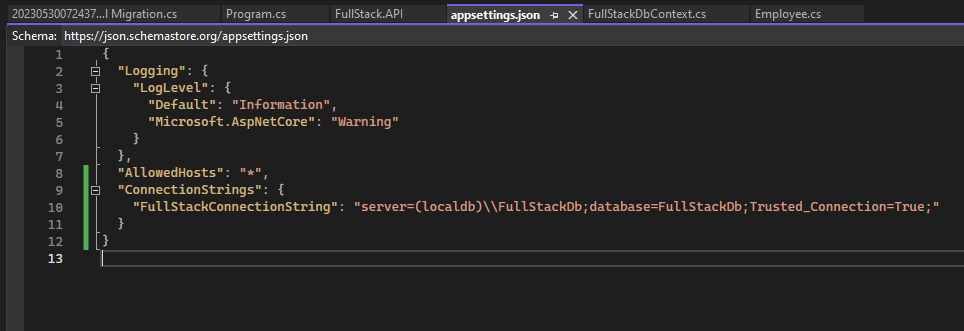
Description automatically generated with medium confidence



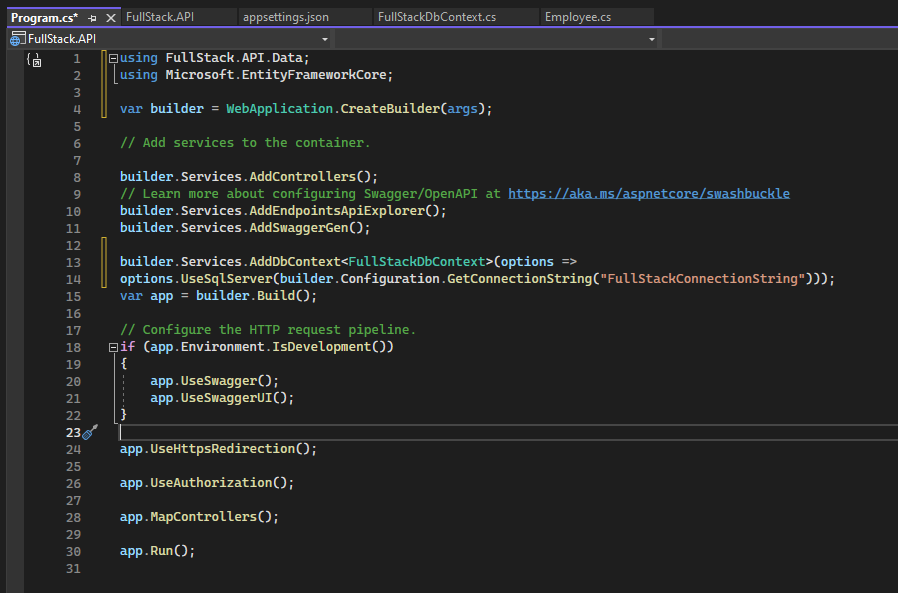
Connect the created server in the above given manner. Here put your server name in place of FullStackDb.

**Now we will do sql server connection: -**

1. Go to appsettings.json file and add a new field “ConnectionStrings” object and write the details for connection string in it.



1. We will now have to inject DbContext into our Program.cs file to be able to use it in our controllers to talk to our database.



Ref line 13-14

We are now set up to create our database using Entity framework core.

1. Go to Tools > Nuget Package Manager > Package Manager Console. This will open a console, in which write the following commands: -
   1. Add-Migration “Initial Migration” #cmd (This command will create a Migration folder for us which contains the class files)
   2. Update-Database #cmd (This command will create the db in sql server that we gave the name in our json file)

**To perform CRUD Operations: -**

1. Create EmployeesController.cs in Controller folder.
2. Add [ApiController] on top of the class in the controller that we created.
3. .net core comes with a built-in swagger application to test our APIs. So, when we run the API project, we can directly test our APIs.
4. Complete the Create and Read functions in the controller.

**Connecting API and UI Project: -**

In Angular, we use services to call any external API.

**To create new service: -**

First make a new folder named services in app folder.

Then route to services folder from terminal and write the following command

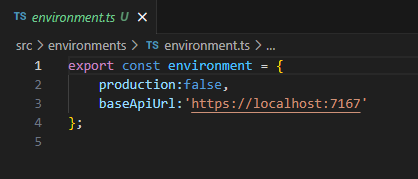
ng g s employees #cmd

**Steps:-**

1. Now, to talk to our .Net API we need to make method in our employees.service.ts file in our angular project.
2. Then create environment folder with the following command.

ng g environments

1. Give the Api path from the swagger link in environment.ts file. Also add the same in environment.development.ts file.



1. Then make changes in employees.service.ts file.

A screen shot of a computer program

Description automatically generated with low confidence

1. After this we will use our employees Service in employee-list.component.ts file

A screen shot of a computer program

Description automatically generated with low confidence

1. To be able to make Http calls add HttpClientModule in app.module.ts file.

A screen shot of a computer program

Description automatically generated with low confidence

If we run the project now and see, we will be having a CORS (Cross Origin Resource Sharing) error in the console. This is because CORS in angular does not allow API port (7167) to talk to UI port (4200). To enable CORS we have to make small changes in Program.cs file in our API project.

A computer screen shot of a program

Description automatically generated with low confidence

Ref line 26.

Here we are enabling CORS to access any header, method and origin in our Angular project.

Now if we run and see the console, we will get an array with the employee data from our db.

1. To show the data that we got in array form on our employees page we will add in employees list page. Make changes in employees-list.component.ts file and run the project, then you will be able to see the data on your page.

A screen shot of a computer program

Description automatically generated with low confidence

Ref line 18.

**Creating Add component: -**

1. Browse to employees folder in components from terminal to add a new component in that folder.

A picture containing screenshot, font

Description automatically generated

1. To add routing for this component, go to app-routing.module.ts and add another object in Routes as given below:-

A screen shot of a computer program

Description automatically generated with low confidence

Ref line 15-18.

1. To add a new url inside our header (frontend html), go to app.component.html file and do as follows: -

A screen shot of a computer

Description automatically generated with medium confidence

Ref line 12-14.

**Common steps to add a new functionality: -**

1. Create a new component in the components folder.
2. Add routing for the newly created component in app-routing.module.ts.
3. Add html for the new component’s html file.
4. To bind the model with html, add methods in the component.ts file of the new component and call the methods in html.
5. When we want to call a method from API project, we first need to call the method in service and then call it in component.ts file.