Making the TodoList Project

1. Creating the Repo

Head over to GitKraken and click on New to create the repo. Don’t forget to have a gitignore file for any files that are needed. If you select a template, don’t forget to delete all the data in that file and commit. You are able to edit the file by clicking on the edit button on the top right of the file.

1. Create the file structure

Go to the repo and create 3 folders: server, client, database-scripts

Go to the server folder and create another folder called Todo-API

Go to the client folder and make a folder called Todo-UI

1. Create the baseline API

Now go to rider and click on New solution. Create the solution under the Todo-API and name it: TodoList.Api.

Now add new projects TodoList.Core and TodoList.Infrastructure.SQL and their corresponding test projects ending in .Test.

To do this just click on the solution at the top and click on “New Project”

Now we want to add some more folder structure so put all the test projects under the “tests” folder and the other projects under the “src” folder. To do this go to the solution, right click and press “New Solution folder”.

Recall that solution folders are virtual folders that let you group projects easily in the explorer but are not physical folders in your computer’s directory

1. Configuring the initial API solution build.

On the top left you will see the TodoList.Api to run. That configuration name for the run will match the project name but this can be updated. Just go the top right where the 3 dots are and click on edit.

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

You can change the name in the name section and in the launch file you can specify which launch profile will be used to launch the api front page. Since we are using swagger we want to use the launch configuration from the API project hence my choice. The configurations including the port that will be used for hosting the API can be found in the launchsettings.json file.

In the launch settings under the profiles section you will find the different ports and names of launch configurations.

A computer screen shot of text

Description automatically generated

If you notice the application URL, there are two URLs, one of them if the public API url that can be accessed publically by other applications or users. The second url seems to be a private link.

Note the ASPNETCORE\_ENVIRONMENT variable is set to Development so that when the application is built, we check what environment we are in and decide to display the Swagger UI or not. Note that the below picture is from the Program.cs file. In .NET 6, we use the program file alone rather than a startup and program.cs file.

A screen shot of a computer code

Description automatically generated

1. Creating the models in the API

So here we should create any models that the API will use, not including any for the EntityFramework or repository/services. We normally add models for objects in the core project and in this case we have one basic model: TodoListItem.cs

Make the ID an init property since we don’t want to modify the ID, only on object creation

1. Set up the database

Take a look at the scripts under the database-scripts folder.

Lets take a look at the setup.sql first which sets up the new tables called “todo”

Here you can create the table with the three columns. Note that the id is IDENTITY(1,1). This means that the value of the id will always increase by 1 starting at 1

The data.sql file has the basic data for the table

If you like to take a look at the lucidchart diagrams with the outlines and tentative plan here is the link: <https://lucid.app/lucidchart/7a21d7c2-2e00-4ba1-bcdd-93dbd25b1cfc/edit?viewport_loc=-11%2C-11%2C1619%2C780%2C0_0&invitationId=inv_0ec8ca89-4340-4f58-bc53-ccfa72741957>