UofMules – Database First application

Inorder to implement Database First application, and create a MVC EF6 core application for an existing database called UofMules, do the following steps:

1. Create a new project called [YourLastName]Week6 – C# - MVC-Web-ASP.NET core project. Delete the .\Views\Shared\Error.cshtml, and also delete the HomeController's `ILogger` Property, and constructor.

2. Create App_Data folder and copy the databases you want to use for the project inside the App_Data folder. (appdata folder provided for you on blackboard. Download and save the folder).

   You should save the App_Data folder inside the project folder. Make sure the files you saved have both .mdf and .ldf extensions. Once you have copied the App_Data folder, you will be able to see those files from the solution explorer.

3. Double click on the mdf file. If it straight away connects, you should be able to see the tables, etc, when you expand on the database name from dataconnections in the server explorer window. (If you do not already see it, you can access that from View-> ServerExplorer window menu.

   a. If you are not able to connect to the database, you will see it in dataconnections in the server explorer window, but with a crossout. (X). right click on it and select modify connections. Make sure you do the test connection and make sure the connection succeeds. Common errors and their remedies are given to you in the ppt.

   At this time, an important to verify the properties for the .mdf file.

   b. The Property should show "Copy if newer".

   c. The default is Do Not copy. This will cause errors.

   Make sure you build the application once, after this property is set.

4. Set the connection string in appSettings.json file as follows:

   a. Note: The entire appSettings.json is given below. The only entry you need is the ConnectionStrings. The rest of the file is given to you only for a reference of placement inside the file. It does not matter where you put it, but it has to be its own node at the root level inside the json file.

```
{
  "ConnectionStrings": {
    "NPGoldConnstring": "Server=(LocalDB)\\MSSQLLocalDB; AttachDBFilename=
|DataDirectory|\\App_Data\\UofMules.mdf;Trusted_Connection=True;
MultipleActiveResultSets=true"
  },
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*"
}
```

5. Dependency Injection: Inside your program.cs file, add the following code:

   var path = Directory.GetCurrentDirectory();

   In order to access the appsettings.json file where you set the connection string, add the following line:

   IConfiguration configuration = new ConfigurationBuilder().AddJsonFile("appsettings.json").Build();

   Install the following EntityFrameworkCore packages from either the Package Manager tabs, or from the Package Manager console: (the commands are also given for console use).

   ```
   Install-Package Microsoft.EntityFrameworkCore.SqlServer
   Install-Package Microsoft.EntityFrameworkCore.Design
   Install-Package Microsoft.EntityFrameworkCore.Tools
   ```

6. Once these are successfully installed, proceed to the next step of scaffolding the application. The scaffolding will create the dbContext and the classes inside the model folder for you.

   Scaffolding: using the CLI from **Tools >> NugetPackageManager >> PacketManager Console**

   Scaffold-DbContext "Name=ConnectionStrings:NPGConnString" Microsoft.EntityFrameworkCore.SqlServer -context NPGContext -OutputDir Models

7. After scaffolding is successfully done, go back to your Program.cs file and give the correct pat to the data directory using the pipes to create a portable application.

   Inside your program.cs file, add the following code:

   var path = Directory.GetCurrentDirectory();

   In order to access the appsettings.json file where you set the connection string, add the following line: (leave it as it is).

   IConfiguration configuration = new ConfigurationBuilder().AddJsonFile("appsettings.json").Build();

   Modify the builder.services line as follows: (this is your DI).

   builder.Services.AddDbContext<NPGContext>(options => options.UseSqlServer(configuration.GetConnectionString(“NPGConnString").Replace("|DataDirectory|", path)));

8. When and after the scaffolding succeeds, and you have set the DI, create controller and views as follows:

   Right click on the controller folder and select Add >> Controller. When the dialog opens select: MVC Controller with Views using Entity Framework Core option (do not choose the empty controller). Click add.

   Another dialog opens where you will need to specify several things:

   From the model class options :select the Model (or table) for which you want to create controller and views. From the Data Context Class option: select the DbContext class you just created. You can see checkboxes for create views, use reference Libraries and user layout are checked. When you click add, it will create controllers and views for the model & data context you have selected above.

   If there is no further coding – you will need to create the navigation menu items in the _Layout.cshtml file.