**User Guide for Maintainence Purposes**

First of all the following will be necessary for maintaining the code base effectively:

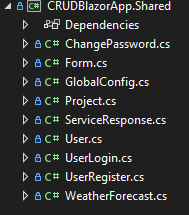
* Visual Studio 2022 with Blazor Web Assembly installed
* SQL Server
* SQL Server Management Studio

Visual Studio 2022 is where the codebase was developed in. The following dependencies were used in the development of this product:

* Blazored.LocalStorage V 4.4.0
* Buint V 1.23.9
* Coverlet.collector V 3.2.0
* FluentAssertions V 6.12.0
* Microsoft.AspNetCore.Authentication.JwtBearer V 6.0.18
* Microsoft.AspNetCore.Components.Authorization V 6.0.18
* Microsoft.AspNetCore.Components.WebAssembly V 6.0.16
* Microsoft.AspNetCore.Components.WebAssembly.DevServer V 6.0.16
* Microsoft.AspNetCore.Components.WebAssembly.Server V 6.0.16
* Microsoft.EntityFrameworkCore V 7.0.10
* Microsot.EntityFrameworkCore.Design V 7.0.10
* Microsoft.EntityFrameworkCore.InMemory V 6.0.18
* Microsoft.EntityFrameworkCore.SqlServer V 7.0.10
* Microsoft.NET.Test.Sdk V. 17.7.2
* Moq V 4.20.69
* Radzen.Blazor V 4.15.11
* Swashbuckle.AspNetCore V 6.5.0
* System.ComponentModel.Annotations V 5.5.0
* Xunit V 2.4.2
* Xunit.runner.visualstudio V 2.4.2

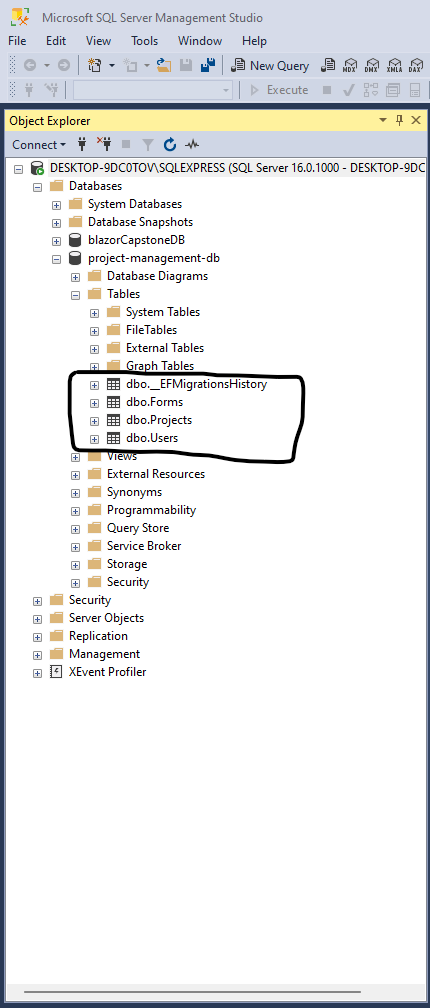
These dependencies allow the tests, UI functionality, interaction with the database, and api documentation to work properly. If updating these packages please save a version prior to updating dependencies as some of the integration may be thrown off by the updates.

All of the models that are stored in the SQL Server database are located in the CRUDBlazorApp.Shared folder (screenshot below):

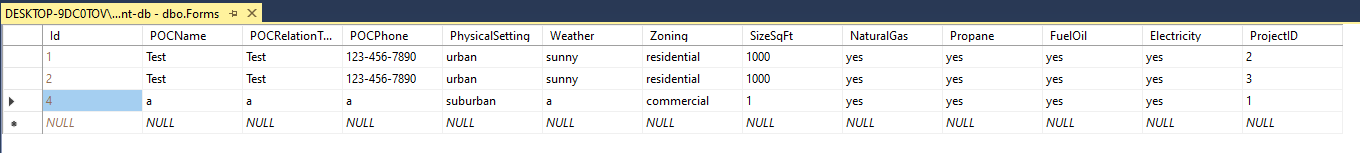


If any of these models need to be updated then please make a git branch and then begin your updates. To apply these changes to the database use the following commands within the package manager console: *dotnet ef add migrations ExampleMigration* followed by *dotnet ef update database.* This will apply changes to the database downstream. If there are new relationships being formed then the DotNet EntityFramework should pick them up. This can be seen within the Server Migrations folder for more detail.

If you need to view the database in detail use the SQL Server Management Studio to view the database. The connection string is located within the CRUDBlazorApp.Server/appsettings.json file. Once hooked into the database you will be able to view the various database tables.Screenshot example provided below:

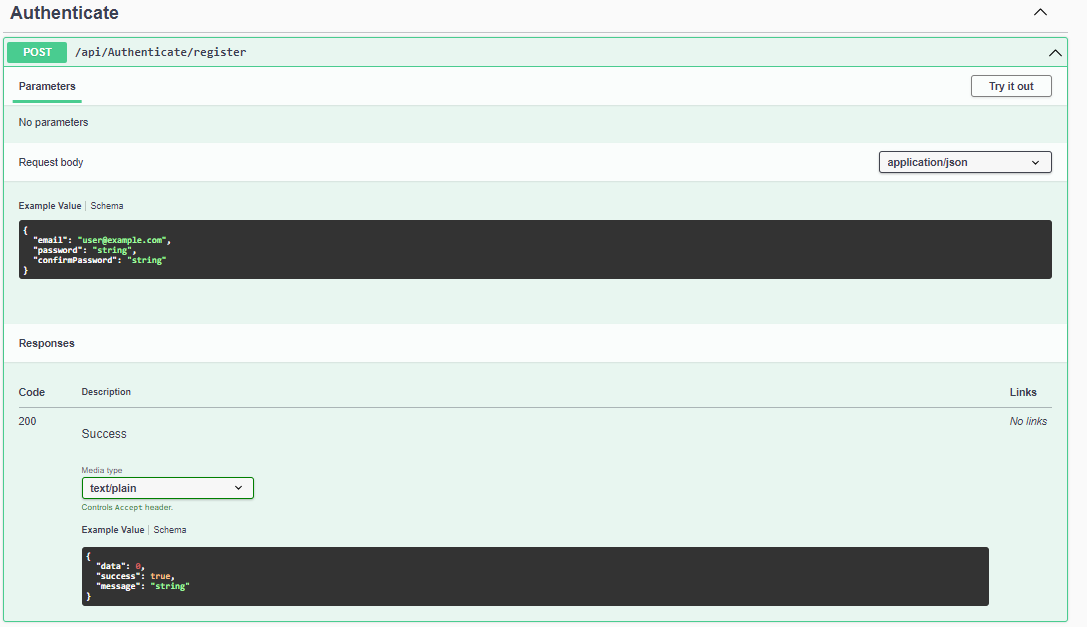


Right click the table you’d like to view and select Edit top 200 rows to see a snippet of the available data in each table. The view provided after selecting this looks as following:



The management studio also allows maintenance to be performed on the database outside of the web application.

In addition, there may be a time where the information being displayed on the application to the users isn’t being performed. To test this information you can go to the website path website/swagger in order to see how all of the API calls operate and test them through the try it out feature. Below is an example of one of the many API calls within the application:



If a status code of 500 appears then I would recommend rebooting the application as sometimes the webassembly doesn’t render correctly.

On the client side, there shouldn’t be any maintenance issues as all the information being fed to the client side is from the API and the database.

If you have any questions or concerns, please feel free to reach out to me at [wlee90@wgu.edu](mailto:wlee90@wgu.edu).

Thank you,  
Bill Lee