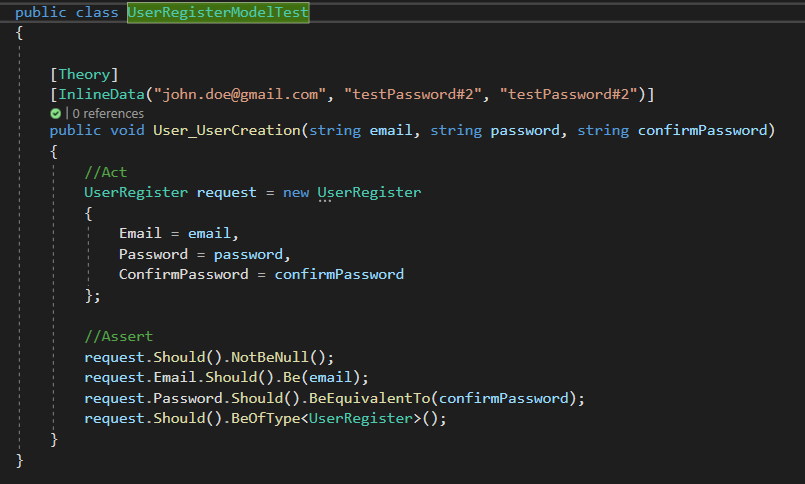
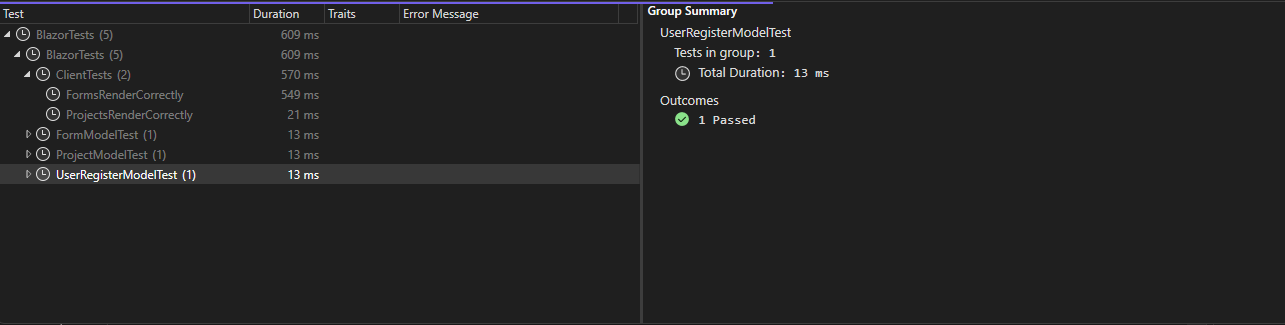
**Testing Plan**

The goal of the testing plan is to identify key areas to ensure that the application can be run effectively for the use of the customer or organization. In this application, unit test were created on both the server and client sides of the application. API testing can be achieved through the /swagger/index.html page of the website. For the unit testing, each of the major models (user register, project, and form) were tested to ensure that objects could be created using only the required fields for each model.

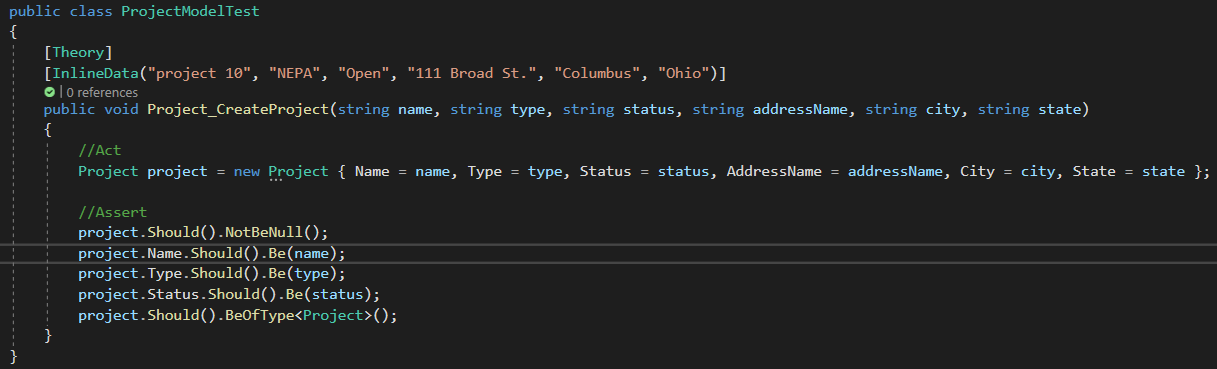
The test script for the user registration model is screenshot below:



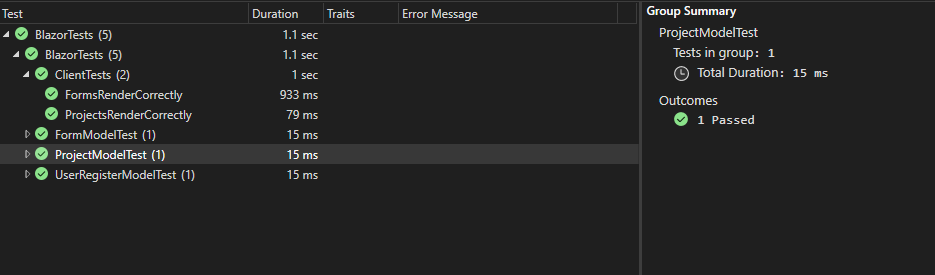
This test ensures that when the object is created that the object is of the correct type, is not a null value and that the password and confirm password both match. Below is a screenshot of the test results in Visual Studio 2022:



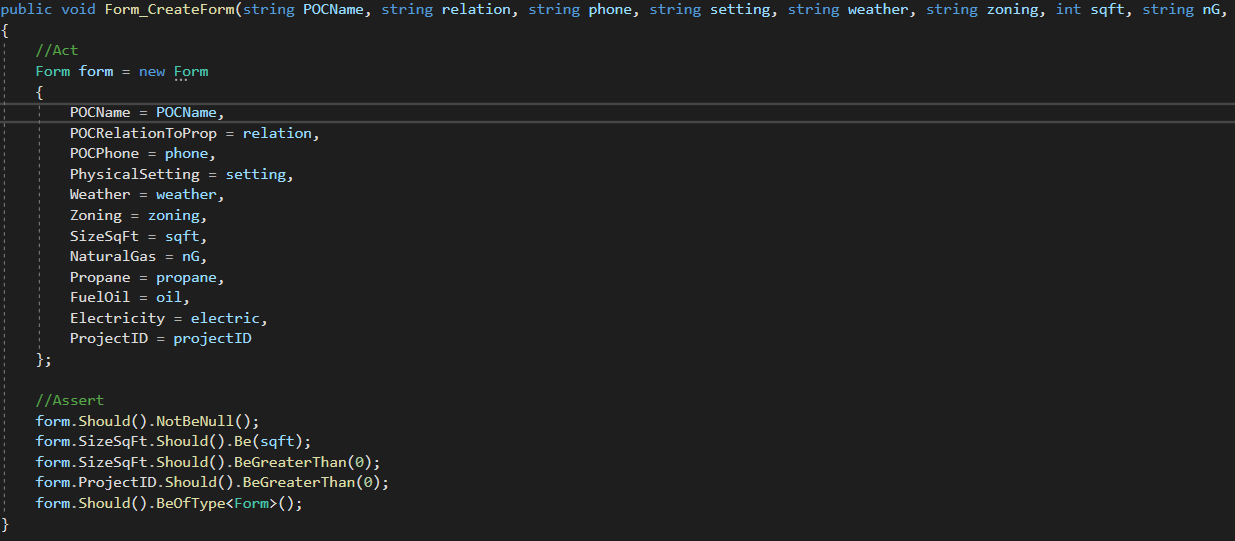
The next test script is for the project model. The screenshot of this script is below:



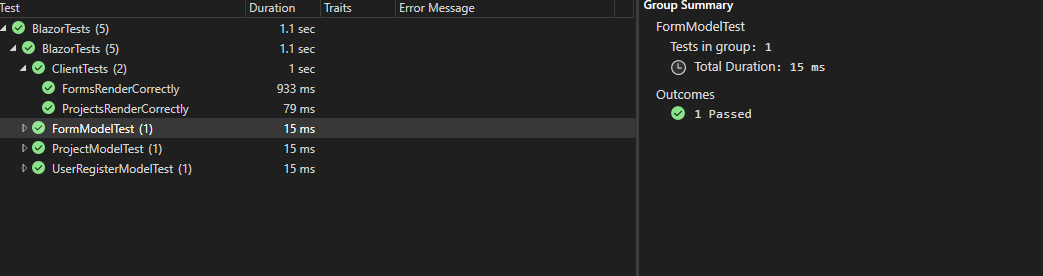
This test ensures that when the object is created that the object is of the correct type, is not a null value and that several of the driving values within the model are being correctly input to the new object from the class. Below is a screenshot of the test results in Visual Studio 2022:



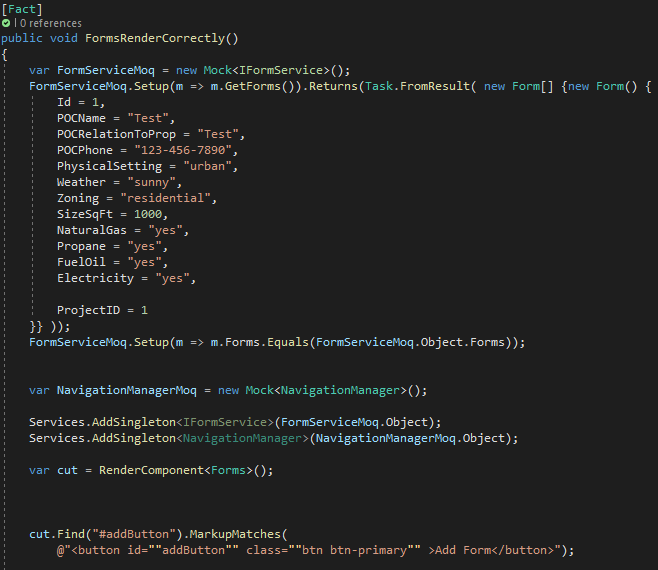
The next test script is for the form model. The screenshot of this script is below:



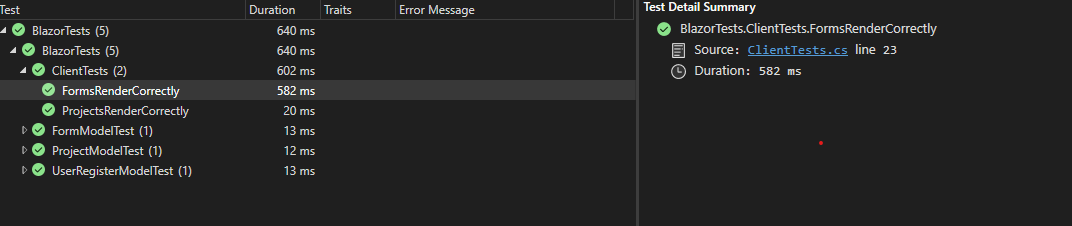
This test ensures that when the object is created that the object is of the correct type, is not a null value. Due to there being important integer values additional checks were put into place to ensure that these numbers where within a logical range. For instance if there was a negative value in one of these it would throw an exception. Below is a screenshot of the test results in Visual Studio 2022:



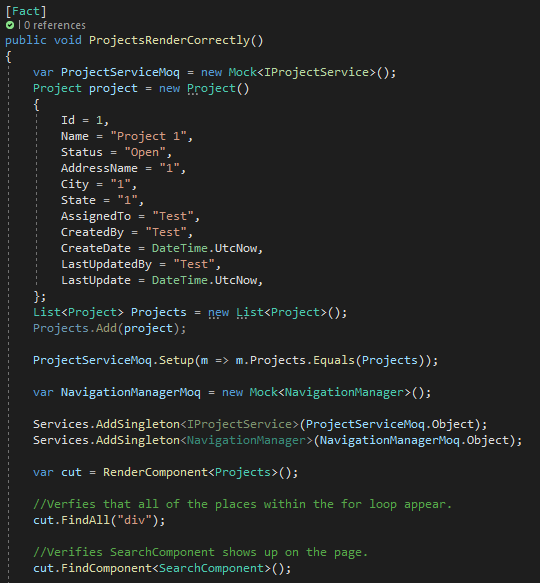
On the client side, I ran two tests within the ClientTests.cs file. The first test was to see if the correct button displayed on the forms.razor page. Below is a screenshot of the script:



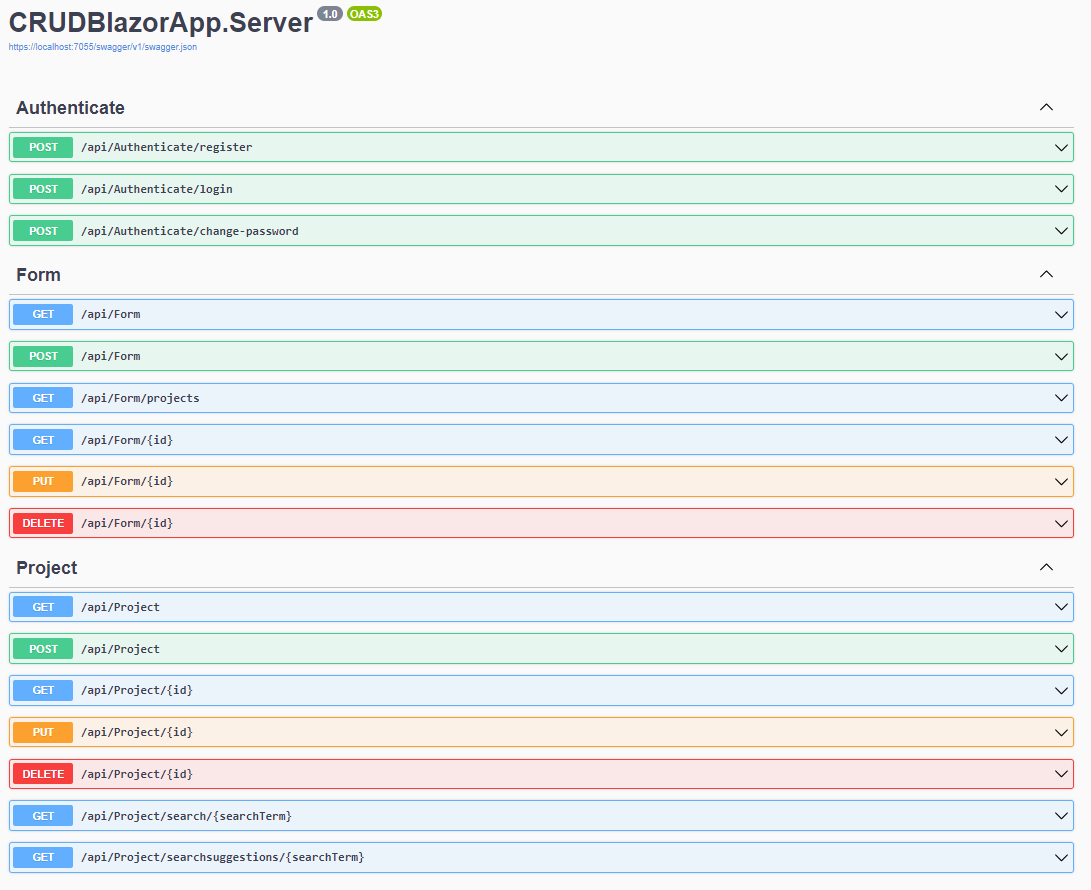
The main purpose for this is to ensure that the user can get to the button to input a new form into the system. As this button drives the clients ability to organize and analyze specific data regarding each site survey. Below is a screenshot of the successful test:



The second test script I ran on the client site was to ensure that the Projects.razor file rendered properly. Below is a screenshot of the script:



I ensured the projects component was rendering properly by finding all of the div’s within the for loop were properly being pulled since there were only <div> elements within the for loop. Following this I wanted to ensure that the <SearchComponent /> was being rendered on this page. This component is important because it allows the user to search the database for specific data within the projects component. Both of these come after injecting several mock services to ensure proper functioning. Lastly, is a screenshot for the API testing built-in with SwaggerUI:



This interface allowed me to manual test the controllers within the server to ensure that data was being relayed from the Server to the Client appropriately.

After the testing was complete I did not end up changing anything, but rather double checked that the correct portions of the different class models that were required should still be required. I determined after these tests that the required fields equated to the minimal information required for the application to continue to function smoothly. Additional fields may be required depending on user testing.