Setting up Unit Testing with NUnit and Moq

GitHub Repository: https://github.com/PrashastVats1/Practice-Projects/tree/Phase3NUnit

Unit testing is an essential part of software development to ensure the correctness and reliability of code. In this guide, we will walk through the steps of setting up unit testing using NUnit and Moq for a school data management system. The system includes classes for managing schools, students, and subjects.

Step 1: Create a Windows Class Library Project

Open Visual Studio 2022.

Click on "File" > "New" > "Project...".

Select "Class Library" as the project template and name it "School."

Within the "School" project, add the following class files: Teacher.cs, Student.cs, and Subject.cs.

Step 2: Add References

Right-click on "References" in the Solution Explorer of your "RainbowSchoolLib" Class Library project.

Click on "Add Reference..." to open the reference manager.

Add the necessary references for your project, which may include references to Entity Framework, databases, or any other dependencies your classes require.

Step 3: Create a Unit Testing Project

Right-click on your solution in the Solution Explorer.

Select "Add" > "New Project...".

Choose "NUnit Test Project" as the project template and name it "NUnitTestProj"

Step 4: Add References to the Test Project

In the "NUnitTestProj" project, right-click on "References" in the Solution Explorer.

Click on "Add Reference..." to open the reference manager.

Under the "Projects" tab, select your "RainbowSchoolLib.dll" and click "OK."

Step 5: Write NUnit Test Cases

Create test classes within the "NUnitTestProj" project for each class you want to test. For example, you can create TeacherTest.cs, StudentTest.cs, and SubjectTest.cs for testing the Teacher, Student, and Subject classes, respectively.

Write test methods within these test classes to validate the behavior of the properties and functionalities of your classes.

Utilize NUnit assertions (e.g., Assert.AreEqual, Assert.IsTrue, etc.) to check if the actual results match the expected results in your test methods.

Step 6: Running Tests

In the Visual Studio menu bar, click on "Test."

Select "Test Explorer" to open the Test Explorer window, where we can view all our tests.

Click on the "Run All" button in the Test Explorer to execute all the tests.

Visual Studio will build the solution, run the tests, and display the test results in the Test Explorer.

With these steps completed, we have successfully set up unit testing using NUnit and Moq for our school data management system. We can now write and run tests to verify the correctness of our classes, ensuring they function as expected. This approach helps identify and address issues early in the development process, leading to more robust and maintainable code.