Alexander Vera

SNHU

6/23/2022

Project 2

CS320

1. Summary
2. Describe your unit testing approach for each of the three features.

* To what extent was your approach aligned to the software requirements? Support your claims with specific evidence.

Each coding package, Contact Service, Task Service, and Appointment Service had certain requirements for each class inside. Contact class object required a contact ID string that cannot be longer than 10 characters, not be null and shall not be updatable. That the first name attribute within cannot be longer that 10 characters and not null, the last name attribute cannot be longer than 10 characters and not null. The phone number must be exactly 10 digits and shall not be null. The address field shall not be longer than 30 and not be null. Contact Service class file within that package also held certain requirements. Fulfilling these requirements were fulfilled specifically in each project. A test run was of creating the object and adding all the proper information was done.

* Defend the overall quality of your JUnit tests. In other words, how do you know your JUnit tests were effective based on the coverage percentage?

Each package was tested by creating an object and adding it to a list. The way I coded the objects is that if the incorrect information was inputted, the program would display a message telling the user that the wrong information was entered. Each time I tested the objects within these packages, it was 100 percent complete.

1. Describe your experience writing the JUnit tests.

* How did you ensure that your code was technically sound? Cite specific lines of code from your tests to illustrate.

Junit uses the @ and certain keywords to active the code in the following lines. This in be tried in a small method. By creating a method that simply uses the package already created, we can create and object and see if it works. If the testing code is written correctly Junit will run, if it written incorrectly, you will receive a syntax error before you even run the code. If the syntax is correct but the test failed, you must look at what exactly you’re trying to test. If you expected, it to fail then you want to set the test up that way for the failure to be counted as a success.

* How did you ensure that your code was efficient? Cite specific lines of code from your tests to illustrate.

Making sure the test was successful through the Junit. When a test is performed, the number of tests is shown, and how many were passed.

2. Reflection

a. Testing Techniques

* What were the software testing techniques that you employed in this project? Describe their characteristics using specific details.

Testing different objects in each package, creating fail safes in the code to catch mistakes beforehand, creating different scenarios like to test each attribute of the object and the update and delete methods.

* What are the other software testing techniques that you did not use for this project? Describe their characteristics using specific details.

I was not able to successfully use the assert methods. For some reason I was receiving error messages every time I tried to run assert methods. I tried true, false, and expected. I do not know if I had the proper junit library package added.

* For each of the techniques you discussed, explain the practical uses and implications for different software development projects and situations.

The assert methods can be used to test different types of projects. A more mathematical project may use expected to check if a math equation in a method works correctly.

1. Mindset

* Assess the mindset that you adopted working on this project. In acting as a software tester, to what extent did you employ caution? Why was it important to appreciate the complexity and interrelationships of the code you were testing? Provide specific examples to illustrate your claims.

I think it is important to note from what perspective a tester is taking or coming from. If it is the developer who is testing their code. then they might not be as familiar with all the ways testing can be done. From someone who concentrates on just testing software, they may not necessarily know how the program might function.

* Assess the ways you tried to limit bias in your review of the code. On the software developer side, can you imagine that bias would be a concern if you were responsible for testing your own code? Provide specific examples to illustrate your claims.

The best way to handle coder bias is by having someone look at the code and give their honest assessment of it. What errors they see and how it can be better. Testing is better off being down by a different group more focused on testing.

* Finally, evaluate the importance of being disciplined in your commitment to quality as a software engineering professional. Why is it important not to cut corners when it comes to writing or testing code? How do you plan to avoid technical debt as a practitioner in the field? Provide specific examples to illustrate your claims.
  + Having coding integrity comes with more practice and time. Often it is a race to get something done because jobs are literally on the line. To make sure corners are not being cut, developers should aim to test the corners might have been cut. If a corner is cut, their may be a piece of data the program might not be able to handle because of said cut corner.