Griffin Hood

CS-320

Project 2

June 12, 2023

* **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.**
    - My approach was very well aligned to the software requirements. I ensured that each part of the requirements was tested by converting each requirement itself into a test. I did this for all 3 features of the application. For example, in the ContactTest.java file, I tested the contact ID validation by creating an assertion for each of the 3 guidelines given in the requirements.
  + **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?**
    - I know that my unit tests were effective because the coverage of the tests were very high. This tells me that not only did the tests pass, but also that I tested the majority of the new code that should have been tested.
* **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.**
    - I ensure that my code was technically sound by testing the fundamentals of each feature. By testing the very low-level components of the code, I was confident that the higher-level components would work. For example, in ContactTest.java, on lines 21 and 22, I ensured that exceptions would be thrown if the contact ID was either null, or if it was too long.
  + **How did you ensure that your code was efficient? Cite specific lines of code from your tests to illustrate.**
    - One example of how my code was efficient was in ContactService.java in the findContactById method. I made sure to return from the function once the correct contact was found, rather than storing that contact and continuing through the rest of the list unnecessarily.
* **Testing Techniques**
  + **What were the software testing techniques that you employed in this project? Describe their characteristics using specific details.**
    - One of the testing techniques that I used in this project was unit testing. I created separate test suites to test each individual component, or unit, of code. There were 6 units in all, contacts, tasks, appointments, and their respective service classes.
  + **What are the other software testing techniques that you did not use for this project? Describe their characteristics using specific details.**
    - I did not use integration testing during this project. Integration testing is a testing technique where you test how well a new component fits in and works with an existing system. Because there was no “existing system” there was no need for integration testing.
  + **For each of the techniques you discussed, explain the practical uses and implications for different software development projects and situations.**
    - Unit testing can be used in almost every project. In every project, there is going to be a distinction between different units of code, even if there is only just one unit. Integration testing can be useful when you are slowly rewriting different parts of legacy code. You don’t want to do that without integration testing.
* **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.**
    - Assuming the mindset of a software tester, I made sure that each of the requirements was translated into a test. This helped to ensure that each requirement was tested properly and that it was functioning as expected. In the TaskTest.java file, in the testTaskDescription test, I made sure to test a valid description, and both types of invalid descriptions to make sure that code coverage was 100%.
  + **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.**
    - I could definitely imagine that bias would be a concern for testing my own code. As a tester, I tried to remove any preferences of style or convention from my testing process. This helped to ensure that I was focused solely on testing the functionality of the code and not the way in which the code was written. For the majority of the code, I decided to throw exceptions for invalid arguments. As a tester, I didn’t question this process and made sure to test the code as it was written.
  + **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.**
    - Quality of code should be one of, if not the top, concern of a software engineer. Code quality ensures several things down the line. One of those things is that it makes the code reliable in production. Reliable code is something that is very important, and it helps to not have to think about whether something *will* work because you *know* it will work. I plan to employ different testing techniques in the field to make my code more reliable and to reduce tech debt for myself and others that may maintain the code in the future.