CS 320 Project Two

Tomas Palma Alvarado

Southern New Hampshire University

CS 320 Software Test, Automation, QA

Professor Nicholas Shepard

October 20th, 2024

1. **Summary**
   1. Describe your unit testing approach for each of the three features.
      1. To what extent was your approach **aligned to the software requirements**? Support your claims with specific evidence.

A blue and green text on a black background

Description automatically generatedThroughout my development of the classes and their respective test files for this project, I used the given requirements for each as a rubric to ensure they were being fulfilled. Thus, it is accurate to say my development approach was directly influenced by the software requirements provided. This claim can be supported through the cross examination of the requirements and my test files. For example, the Contact class had the following requirement, “The contact object shall have a required firstName String field that cannot be longer than 10 characters.”, which led me to develop this test:

* + 1. 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?

My test files leverage different methods, such as assertEquals, assertThrows, assertNotNull , and @BeforeEach, to provide full coverage on all six classes. Through the use of these tools my coverage reached 100%, as shown in the following:

A screenshot of a graph

Description automatically generated

* 1. Describe your experience writing the JUnit tests.
     1. How did you ensure that your code was **technically sound**? Cite specific lines of code from your tests to illustrate.

I ensured my code was technically sound through extensive testing of each individual component of each class, as well as ensuring components worked across classes when necessary. For example, here is the test that I wrote to ensure the Contact class was functioning properly through its constructor, as well as a test for cross-class functionality orchestrated through the A screen shot of a computer code

Description automatically generatedA screen shot of a computer code

Description automatically generatedContactService class:

* + 1. How did you ensure that your code was **efficient**? Cite specific lines of code from your tests to illustrate.

A computer screen shot of text

Description automatically generatedEfficiency in my code can be seen in two key components. The first, seen in the Contact, Task, and Appointment classes, is the use of setter methods in the component. Through this mechanism, we can avoid duplicate code in the constructor and setters that check the validity of the inputted parameters. The second is through the use of the HashMap data structure which allows for rapid look up with a time complexity of O(1) for get() and put() operations due to its key-value pairings in storage. (Builtin, 2024). Here is an example from my code of how the HashMaps were implemented:

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

The three main software testing techniques employed in this project were unit testing, regression testing, and validation testing. Unit testing involves testing each individual component for functionality. A unit is described as the smallest functional unit of code (Amazon Web Services, n.d.). Regression testing is the practice of testing previous code in a file once new code is added as to ensure no regression in functionality. This was conducted specifically when adding new functionality to my code when implementing instructor guided corrections with my constructors. When the new code was added, I rechecked my other methods for proper functionality. Lastly, validation testing comprises of confirming that the code meets requirements set by the client. This is completed through having a test for each requirement provided and ensuring proper output.

* + 1. What are the **other software testing techniques** that you did not use for this project? Describe their characteristics using specific details.

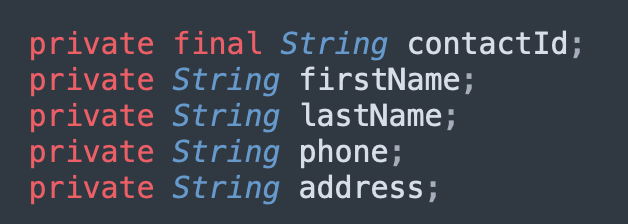
Two testing techniques that were not deployed in the application were integration testing and stress testing. Integration testing involves examining the interaction between modules to identify any problems when they are integrated into the same environment. Stress testing involves testing the reliability and durability of a system under extreme conditions (GeeksForGeeks, 2024). It can involve putting a large amount of traffic and requests through a system to see how it deals with it and if it has a viable exit strategy if things do end up going wrong.

* + 1. For each of the techniques you discussed, explain the **practical uses and implications** for different software development projects and situations.

Unit testing has the practicality of being easy to implement and leads to the early detection of bugs within units. Regression testing is helpful in codebases that undergo change and helps maintain the functionality in the product. Validation testing is conducted towards the end of development of code and ensures that the product is fulfilling the needs to the client. Integration testing is critical in multifaceted projects which have a lot of different components to them. It ensures that different parts of the codebase and system work harmoniously and as expected. Stress testing is critical for projects where the product could potentially be forced to handle a large load of traffic. This form of testing is great for ensuring the reliability and stability of the product which, depending on the sector it deals with, could be of the upmost importance.

* 1. Mindset
     1. 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.

My mindset throughout the completion of this project was to ensure that my code met all requirements provided to me using industry best standards. Notably, I was sure to exercise caution as I developed my code due to its interconnectivity and general nature of the material. If this project were to be used beyond schoolwork, things such as IDs, names, and addresses would need to be properly encapsulated as to not leak sensitive data to attackers. Also, since there exists interconnectivity between classes in the project proper planning needed to be done to ensure the task would be successfully completed. An example of caution can be seen in the initialization of my fields:



* + 1. 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.

It is critical to have an unbiased approach when testing one’s code in order to properly test it. Issues could certainly arise from a tester that is not being thorough due to personal feelings. This could lead to potential bugs making it past the testing stage and causing issues in production code. Personally, I limit bias by remaining objective to my goal of producing functional code and tests. While testing, I am as thorough as if I were inspecting someone else’s code and do my best to try not to get too down on myself when I find issues. Acknowledging these issues helps one produce better code in the future and helps develop one’s testing abilities. For example, in my code my instructor pointed out that I had repeated code in my constructor and setters which highlighted an inefficiency in my code. I recognized this and implemented the solution, and my code was absolutely the better for it.

* + 1. 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.

Discipline with the production of software is comprised of multiple components. It involves a double ended effort in the generation of quality software that is followed by a rigorous testing phase. Both are needed to ultimately produce the highest quality code as possible which works as intended. Cutting corners ultimately does not save developers effort and energy as issues must be resolved and can grow in magnitude as the codebase progresses. All of this information has caused me to commit to trying to abide by best industry standards whenever possible and to scrutinize all pieces of code equally, regardless of if it is mine or not. I believe through these practices I can minimize the addition of technical debt to whatever projects I work on in the future.

Works Cited

Amazon Web Services. (n.d.). What is unit testing? AWS. <https://aws.amazon.com/what-is/unit-testing/>

Builtin. (2024, January 18). Hashmap in Java: A complete guide. Built In. <https://builtin.com/articles/hashmap-in-java#:~:text=Because%20of%20its%20effective%20retrieval,their%20position%20within%20the%20collection>.

GeeksforGeeks. (2024, September 25). Stress Testing in Software Testing. GeeksforGeeks. Retrieved October 18, 2024, from <https://www.geeksforgeeks.org/stress-testing-software-testing/>