Rukhsar Amin

7-2 Project Two

Professor Joseph Martinez

June 22nd, 2025

**Summary:**

To be honest, I had no idea how thorough proper testing was until I began working on the Contact, Task, and Appointment services. I was initially only concerned with finishing the bare minimum. Slowly, though, I started considering all the possible ways a user could engage with these services, both positive and negative as well as unexpected. Making my code ready for everything became my objective. I ensured that the requirements of the project were directly linked to my test cases. For example, I confirmed that a past date or an excessively lengthy description could not be used to create an appointment. This is an illustration from my test:

assertThrows(IllegalArgumentException.class, () -> new Appointment("A1", pastDate, "Valid description")

My codes were effective, and I don't just say that because they worked. They forced me to examine my own presumptions, which is why I say that. They put my mindset to the test in addition to my techniques. In order to ensure that my reasoning was sound, I included both expected inputs and edge cases. In addition to being satisfying, seeing all green bars at the end was comforting. Testing my own work initially felt weird, almost like evaluating oneself in front of a mirror. However, my confidence increased slightly as I completed each test. I verified every setter and constraint and organized my tests using reusable setup techniques. Writing meaningful tests was more important for efficiency than writing fewer ones. I also made sure that every test had a purpose.

**Reflection:**

I utilized JUnit 5 for performing unit tests. It enabled me to evaluate the specific functionalities of each class, ensuring they worked precisely as intended. I did not incorporate integration or UI testing for this project because it was primarily centered around service logic; however, I gained insight into these methods through my research. I now recognize the significance of these techniques as projects expand and integrate with front-end systems or external APIs.

This project highlighted the importance of unit testing as the cornerstone of effective software development. It also helped me value how different testing methods serve distinct purposes much like assembling a puzzle where every testing type holds its unique role.

I approached this project with great care, not out of fear, but because I have learned how easy it is to overlook minor details that can lead to larger problems. I allowed myself to take my time, verify my work, and ask, 'what if?' This subtle adjustment made a significant impact.

I recognized that because I was the one who wrote the code, I might accidentally miss its shortcomings. Therefore, I compelled myself to input invalid values bad IDs, excessively long strings, nulls anything that might cause a failure. Evaluating it from this perspective helped me eliminate my developer blind spots and adopt the mindset of a genuine user (or a meticulous bug hunter).

For me, discipline meant avoiding shortcuts. It involved persevering through moments of fatigue or uncertainty, choosing to complete the task with integrity. This approach is not solely about grades; it reflects the person I am becoming as a developer. Someone who cares, takes pride in their work, and aspires to create enduring solutions.

This project not only enhanced my skills but also shifted my mindset. It served as a reminder that software quality hinges on being deliberate at every stage from how I write code, to how I test it, to how I handle errors. One aspect I would like to emphasize is how much this experience made me realize that testing is about more than just code: it is fundamentally about trust. Whether it is a user relying on your application to keep their data safe or a colleague depending on your function to produce the correct output, those small snippets of code carry significant responsibility. There were moments during this project when I felt a bit unsure or exasperated, particularly when Eclipse would show unexpected errors and it would take quiet some time to figure out. However, instead of admitting defeat, I continued to persevere. I used search engines, sought help, revisited the requirements, and most importantly, I showed myself some compassion. To me, that was just as crucial as crafting a clean line of code.

Also, I think that each class I tested provided me with a sense of accomplishment. By the time I reached the AppointmentService Test, I had gained confidence in my structure and logic. I understood what to expect, and I had the tools to manage exceptions effectively. This personal development feels like a progress not only because I completed the task, but also because I transformed into someone capable of taking on greater challenges. Looking ahead, I want to support this perspective as I embark on future projects. I aspire to be a developer who not only addresses issues but also proactively prevents them through careful design and testing. Despite meeting challenges, it feels rewarding to persevere and appear stronger in the end.