Gabrielle Kirsch

CS 320 Project Two

7 December 2023

**Summary:**

**Unit Testing Approach:** For each feature, my unit testing approach was comprehensive. In the Contact class, tests ensured compliance with requirements, such as the 10-character limit for names. The Task class tests focused on the unique ID constraint, while the Service class tests covered a broad spectrum of functionality. Assertions and checks were strategically placed to cover various scenarios.

**Alignment with Software Requirements:** The unit testing approach meticulously aligned with software requirements. In the ContactTest class, for example, assertions checked name lengths, adhering to the specified 10-character limit. Similar attention to detail was applied in TaskTest for unique ID constraints. Each test method was crafted to address a specific requirement, ensuring compliance at every step.

**Overall Quality of JUnit Tests:** The quality of my JUnit tests is robust, validated through coverage percentages. The Service class, with the highest coverage percentage, exemplifies comprehensive testing. Positive coverage percentages affirm that the majority of code functions underwent successful testing, minimizing the likelihood of undetected issues.

**Experience Writing JUnit Tests:** Writing JUnit tests was an enlightening experience. It required meticulous consideration of each requirement and a strategic approach to cover diverse scenarios. The ContactTest class, for instance, involved crafting tests for non-null names and ensuring compliance with length limits.

**Ensuring Technical Soundness:** Technical soundness was assured through careful crafting of code snippets. In the Contact class, the line "public static List CONTACT = new ArrayList();" demonstrated the use of a list for contact management, enhancing the technical structure and aiding in test execution.

**Ensuring Efficiency:** Efficiency was prioritized through judicious use of resources. The modular approach, reflected in the coverage percentages, ensured that tests were efficient in targeting specific functionalities. For instance, the TaskTest class efficiently handled unique ID constraints.

**Reflection:**

**Testing Techniques Employed:** I utilized a step-by-step method, White Box Testing, and JUnit testing. The step-by-step method ensured systematic coverage, White Box Testing complemented JUnit for structural and logical scrutiny, and JUnit tests were the bedrock for functional validation.

**Unused Testing Techniques:** Static testing was not employed in this project. This technique involves code review without execution. While it's effective for early issue detection, I opted for active testing methods like JUnit to identify errors dynamically.

**Practical Uses and Implications:** Step-by-step testing ensures systematic coverage, valuable for complex projects. White Box Testing aids in revealing internal logic flaws. JUnit testing, being dynamic, is crucial for functional validation, ensuring that the software meets requirements during runtime.

**Mindset:**

**Adopted Mindset:** I embraced a dedicated mindset, understanding the importance of caution. Appreciating the complexity and interrelationships of the code was crucial to foresee potential issues. For example, in ContactTest, considering different name scenarios ensured comprehensive testing.

**Limiting Bias:** To limit bias, I maintained humility and openness to learning. If testing my own code, there could be a subconscious inclination to overlook flaws. Actively seeking diverse perspectives and employing varied testing techniques mitigated bias risks.

**Discipline and Commitment:** Discipline in code creation and testing is paramount. Cutting corners risks compromising the final product. Avoiding technical debt involves consistent vigilance, akin to maintaining the code's health. For instance, in JUnit tests, the careful arrangement of assertions reflects a disciplined commitment to quality.

In conclusion, a disciplined approach, diverse testing techniques, and a cautious mindset are indispensable for a software engineering professional to deliver high-quality, reliable code.