Gilbert Hernandez

Professor Philips

CS-320

21 June 2024

7-2 Project Two Submission

Throughout the development of the mobile application for our client at Grand Strand Systems, I adopted tailored unit testing approaches for the contact, task, and appointment services. Each feature was thoroughly tested to ensure compliance with the software requirements document (SRD). For example, in the ContactService, I meticulously checked that functionalities like contact creation, deletion, and updates adhered strictly to the specified rules, such as unique contact IDs and field length limits. This was validated through extensive JUnit tests that covered diverse scenarios, ensuring robust service performance.

Ensuring the quality of my JUnit tests was pivotal. I achieved comprehensive coverage of over 85% across all Java files, indicating thorough testing of critical paths and edge cases. Tests included validating updates to contact details and ensuring accuracy in the ContactService. Writing these tests was both challenging and enlightening, focusing not only on functionality but also on reliability and maintainability. For instance, in the ContactService, I ensured that setters for attributes handled edge cases effectively, maintaining code integrity.

To uphold technical soundness, I employed defensive programming techniques in my tests, especially in handling potential invalid inputs. This approach, seen in the Contact class where constructors and setters were designed to manage null inputs and character limit breaches, aimed to enhance application stability under varied conditions. Efficiency was also prioritized through streamlined test executions, such as using parameterized tests in JUnit to validate multiple data sets efficiently.

My testing strategies encompass unit testing, integration testing, and boundary testing. Unit tests were instrumental in isolating components like the ContactService to verify individual functionalities. Integration tests ensured smooth interactions between services, while boundary testing helped validate limits such as character lengths and numeric ranges, safeguarding against edge case failures.

While exploratory testing and usability testing weren't directly employed, they remain valuable for uncovering user experience issues and real-world application scenarios. Exploratory testing, for example, provides insights into end-user interactions, identifying usability concerns that automated tests might overlook.

A cautious mindset was crucial in this project, appreciating the intricacies and dependencies within the codebase. Understanding how changes in one part of the application could impact others, as observed in testing the TaskService, helped in crafting thorough test cases. Bias mitigation in code review involved maintaining objectivity, and ensuring all scenarios were equally considered to uncover potential defects.

Maintaining discipline in code writing and testing is fundamental to delivering reliable software solutions. Cutting corners during testing risks unresolved bugs surfacing later, leading to disruptions and increased costs. Upholding thoroughness in testing and adherence to coding standards mitigates technical debt and fosters sustainable software development practices.

In conclusion, by applying rigorous testing strategies and committing to quality, I aim to deliver robust and user-centric software solutions, meeting the expectations of our clients at Grand Strand Systems.