**Journal: Software Testing Techniques**

Justin Starr

Department of STEM

CS 320 – Software Test Automation & QA

Professor Ramsey Kraya

October 1, 2023

**Journal: Software Testing Techniques**

The software testing technique that was employed for each of the milestones was Junit testing, a form of white-box testing where the test cases were based on the internal structure (Jakubiak, 2022). For all three milestones, I was able to choose inputs that exercise specific paths through each of the classes, and I configured assertions that would then validate the output. Another way that white-box testing was used was to test component structures, which at this level, program structures such as program statements and decisions were tested (Hambling, B., et al., 2019). Statement tests that were written exercised the executable statements in the code. It forces the program to execute specific statements by setting the program to a start state and then giving it input that forces the program to execute the required statements (Hambling, B., et al., 2019). Similarly to statement tests being conducted, decision tests were conducted as well where specific tests were created to test the ability of the program to make appropriate decisions based on input that the tests were given.

Software testing techniques that were not used were black-box testing. This form of testing is done when the tester does not know how the code works or what the structure of the code is that is being tested (BrowserStack, 2023). Black-box testing examines the external behavior of the software and then determines, based on specified requirements, whether or not the software performs or functions as it should \*BrowserStack, 2023). Creating Junit tests can sometimes be thought of as black-box testing; however, it depends on how the tests were created. In the case of modules 3 through 5, we didn’t consider this black box testing because we were concerned about the inner workings of the code. The tests were created after each of the classes was written, which then proved it met the requirements or specifications and functionality. Had the tests been written first, then we could say that we absolutely developed or created the tests first based solely on the requirements documentation.

When considering when it would be useful to implement white-box testing, its practical uses, and its implications for different software development projects and situations we would consider situations where it is essential to know what the inner workings of the software are. It is helpful when trying to identify problems with the developer's code and if it does what it is supposed to do like making the correct decisions at the correct times. It can help to identify code that is necessary but perhaps written in a way that is unreachable. This helps ensure that all parts of the code are tested. Another way white-box testing is useful is that it can help identify defects early on in the developmental process, which can help to save money, and also depending on the type of system being developed, it can help minimize the risk of catastrophic failures that could result in projects being scrapped or even worse, the potential for it to cause harm.

Black-box testing is practical in situations when we are concerned with user-focused testing (Sereda, I, 2023). This is essentially making sure that the software performs how users would expect the software to work, for example when testing the software’s UI (User interface). Again, black-box testing tests the behavior of a system and isn’t concerned with the inner workings of the software or its code. Another example of when black-box testing could be beneficial to a project is if the project is running out of time and needs to be able to have testing conducted quickly. This again, is possible because tests are written not based on the code but rather on if the software does what it is supposed to do based on behavior. Because black-box testing requires less technical expertise, it can be performed by testers with less technical knowledge (Sereda, I., 2023).

**References**

BrowserStack. (2023, July 31). *What is Black Box Testing: Types, Tools & Examples*. App &

Browser Testing Made Easy. <https://www.browserstack.com/guide/black-box-testing>

Hambling, B., Morgan, P., Samaroo, A., Thompson, G., & Williams, P. (2019). *Software testing*

*: An istqb-bcs certified tester foundation guide - 4th edition*. BCS Learning & Development Limited.

Jakubiak, N. (2022, December 6). *JUnit Tutorial with Examples: Setting Up, Writing, and*

*Running Java Unit Tests*. Parasoft. <https://www.parasoft.com/blog/junit-tutorial-setting-up-writing-and-running-java-unit-tests/>

Sereda, I. (2023, July 5). *Black-Box vs White-Box Testing: The Pros and Cons of Each*. Solvd.

<https://www.solvd.com/blog/black-box-vs-white-box-testing>