**Journal: The Role of Testing in the Software Development Life Cycle**

Justin Starr

Department of STEM

CS 320 – UI/UX Software Test Automation & QA

Professor Ramsey Kraya

September 3, 2023

**Journal: The Role of Testing in the Software Development Life Cycle**

No matter what methodology (waterfall, agile, DevOps, etc.) is implemented as part of a software development life cycle (SDLC), testing is a critical phase that is conducted which seeks to identify any errors that lead to defects or faults in software which can ultimately cause observed failures (Hambling, B. et al., 2019). The following is a short analysis of what occurs during the testing stage of the SDLC, why the testing stage is vital to a successful SDLC, and if any exceptions exist in which the testing stage would occur either earlier or later than it typically does in the SDLC.

During the testing stage of the SDLC, whether it's being done during or after development, both functional and non-functional requirements are used to determine acceptance criteria and a definition of “done”. Based on these acceptance criteria, a test basis is then created, which is the starting point or the body of knowledge for what is to be tested and how. (Hambling, B., et al., 2019). Test analysis and design begin which seeks to develop test conditions, which is a portion of the test basis needed to achieve specific test objectives. Once test objectives have been identified, test cases and test procedures can be written for test objectives. User stories and similar artifacts can be used to help create test cases and aid in determining whether or not the software performs as intended or if it fails testing, and needs to go back to development. Once the software or portion of the software that is being developed is ready for testing, the software is tested using the established test procedures for each of the defined test cases. A variation of testing is performed such as component, integration, system, and acceptance (Hambling, B., et al., 2019). This helps ensure that any new portion of the software works correctly and that it also does not negatively affect any other portion of the software that has already been developed. Once all testing has been completed, the product can be pushed through to production.

The reason why the testing phase is important to the SDLC is because as software developers we want to create working software that is reliable and secure. The testing of software seeks to determine that the software does not work because that essentially identifies the portions or aspects of our software that do not meet requirements or function as intended (Hambling, B., et al., 2019). Without this phase of the SDLC, it is possible to still produce a working product, however, oversights can occur which lead to major defects and failures such as the exposure of sensitive or critical information. It also leads to problems such as security vulnerabilities being exploited or even costly re-work or refactoring needing to be completed because errors in the software were not previously identified before the product launches. In certain situations, if problems are not identified, it could lead to entire projects having to be scrapped because the project runs out of money.

Exceptions do exist in which the testing stage could occur either earlier or later than it typically does in the SDLC. An example of early testing might be if the development team needed to test the user interface and user experience before the code development began. This is often done with additional user interviews often incorporating the use of prototypes. Regardless of which phase of development a project is in, the more testing that can be incorporated will ultimately reduce and help prevent further errors later in the SDLC. An example of when testing may occur later in the SDLC is when maintenance needs to occur on a system or when upgrades/updates occur. Any time changes happen to the software, the software needs to be tested again to ensure no new problems have been created or that the new software has not changed the functionality of the previously developed software. Another example of when testing might occur later in the SDLC is when security vulnerabilities are identified that have an impact on the software. The software would have to be modified to include patches or fixes, thus requiring the need for additional testing.

**References**

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.