Research and describe fully, what the true cost of finding a bug later in the SDLC process. Is it cheaper to find the bug during development or in production? Why? Cite resource's other than the book.

The cost of detecting and fixing software defects rises sharply over time in the development process. Addressing bugs after deployment is especially expensive and risky, often increasing costs by an order of magnitude because it can also hinder future clients who no longer trust your products. Detecting issues in code is much easier while developers are still writing it. When software reaches the testing phase, finding defects in a developer's local environment becomes a time-consuming challenge. Fundamental issues that slip through to the design phase often only become known during production. Fixing defects in the maintenance phase can be up to 30 times more expensive than addressing them earlier in the development life cycle because of the increased challenges and risks involved.

References

Functionize. (n.d.). *The cost of finding bugs later in the SDLC*. Functionize. Retrieved September 27, 2024, from https://www.functionize.com/blog/the-cost-of-finding-bugs-later-in-the-sdlc

Answer the following questions found on page. 389: 5, 7, 8, 9, and 13

5. Discuss why testing is so essential to the development of the new system.

Testing is crucial in developing a new system for several reasons. It ensures the software meets requirements and performs as expected, helping to identify defects early. This early detection is cost-effective, as fixing bugs during testing is much cheaper than addressing them after deployment. Thorough testing enhances user satisfaction by delivering a reliable and intuitive system It also assist in identify risks, such as security issues, before the system is deployed.

7. What is the primary goal of unit testing?

Unit testing verifies that individual components of code function correctly on their own. By testing each function developers can ensure that they work as

intended and meet specified requirements. This also helps with the debugging process.

8. How are test cases developed for unit tests?

Black-box testing focuses on reviewing the functional specifications and requirements. Identify what the system is supposed to do without considering the internal implementation. White-box testing looks at the actual source code for debugging errors.

9. What is the primary goal of integration testing?

The main goal of integration testing is to make sure different parts of a software system function together. After testing each part separately, integration testing checks for issues that arise when they interact. This helps ensure that data flows correctly between components.

13. Compare and contrast system testing and acceptance testing.

Both types of testing help create the best system possible, however, they each have different goals. System testing is a quality check on the technical side. System testing checks if the entire software works as intended, focusing on performance and security. In contrast, acceptance testing is a quality check on the user interface. Acceptance testing ensures the software meets the needs of end users and is performed by the users.