What Are the Main Elements of Continuous Testing in DevOps?

The need to deliver quality at speed by leveraging flexible delivery models with quick feedback loops has become critical in today's dynamic business environment. This is where DevOps incorporating CI/CD and test automation can enable businesses to quickly adapt to changing market scenarios. With digital transformation being the buzzword to enhance efficiencies and competitiveness of businesses, DevOps continuous testing has become an essential element in the transition. It helps to minimize any associated business risks and delivers superior-quality products with quick turnarounds. Let us understand the need for continuous testing in the SDLC.

Continuous testing and its importance

Quality has become the cornerstone for business enterprises to achieve customer acceptance and increase brand visibility. The traditional software development approach of testing the integrated modules of a software application after development does not make it entirely free of bugs or vulnerabilities. However, the continuous testing approach of testing early and

consistently throughout the SDLC using automation can mitigate the risks and improve the quality of the product. It is important for the following reasons:

- Reduces the cost of rework by detecting and fixing bugs early
- Enables developers to build features quickly by automating the QA process
- Speeds up the testing process as continuous test automation tools reduce manual testing
- Improves test coverage as automated tests are executed quickly and evaluate all features of the software application
- Enables quicker and frequent release of quality applications
- Brings consistency in testing as the configured environment allows the execution of comparative tests
- Enhances the value of the software, allows accurate reporting of the errors, failures, and successes of the tests, and creates greater visibility for the stakeholders (developers and testers)
- Facilitates coordination among teams with quick feedback and reduces the time for code review

The key to successful continuous testing in DevOps

DevOps continuous testing is implemented in the following ways:

Develop a correct test code: A continuous testing strategy entails building the right test script to generate accurate test results. In other words, it would mean the difference between a high-value test suite and a wasteful and erroneous one. Testers should treat the test suite the same way they treat the production code.

Plan and execute a risk-based test automation strategy: User experience can determine whether a software application will be a success or otherwise in the market. Hence, issues leading to bad user experiences need to be identified and addressed. This calls for employing a risk-based DevOps testing approach to detect, evaluate, and mitigate the issues surrounding bad user experiences.

Provide fast feedback: Continuous testing services generate vast amounts of data that provide stakeholders (testers and developers) with deep and fresh insights into issues. These

also provide the management with dashboard-based intelligence or perspective to choose “go or no go” for individual features.

Maintain test data: The DevOps approach to testing can generate mission-critical test data applicable to real-world scenarios. This data can help QA experts develop insights into the need for updating the application based on the evolving scenarios, customer preferences, and cost considerations.

Access to a stable test environment: To ensure the optimal performance of the test suite, it must be backed by a stable continuous testing framework. Further, the test environments, APIs, and third-party tools should be made available to the testers without downtime or latency.

Better collaboration between developers and testers: In the DevOps scheme of things, all silos must disappear and there should be enhanced collaboration between processes (development, testing, and operations). The teams should be small in size, and all test reports should be centrally located to be accessed by the stakeholders.

Prioritize automation: Instead of removing manual testing, automation should be prioritized across processes. Thus, focus on processes that need to run repeatedly by mapping out the SDLC and identifying the opportunities for automation.

Small design: Run iterative tests on small increments that are easier to design. Small designs allow easy automation and ready deployment.

Track everything: Set up test cases with the proper metrics and pass-fail criteria. Continuous testing can identify if there are issues with the code or if things are working or not.

Use the right tools: For continuous testing, get the right tools that can run in the working environment. Use tools that have community-based support to get better use cases and solutions.

Conclusion

Continuous testing allows the execution of automated tests in the SDLC to obtain immediate feedback on errors, bugs, vulnerabilities, or business risks. It facilitates the delivery of quality software applications seamlessly.

Resource

James Daniel is a software Tech enthusiastic & works at Cigniti Technologies. I'm having a great understanding of today's software testing quality that yields strong results and always happy to create valuable content & share thoughts.

Article Source: dev.to