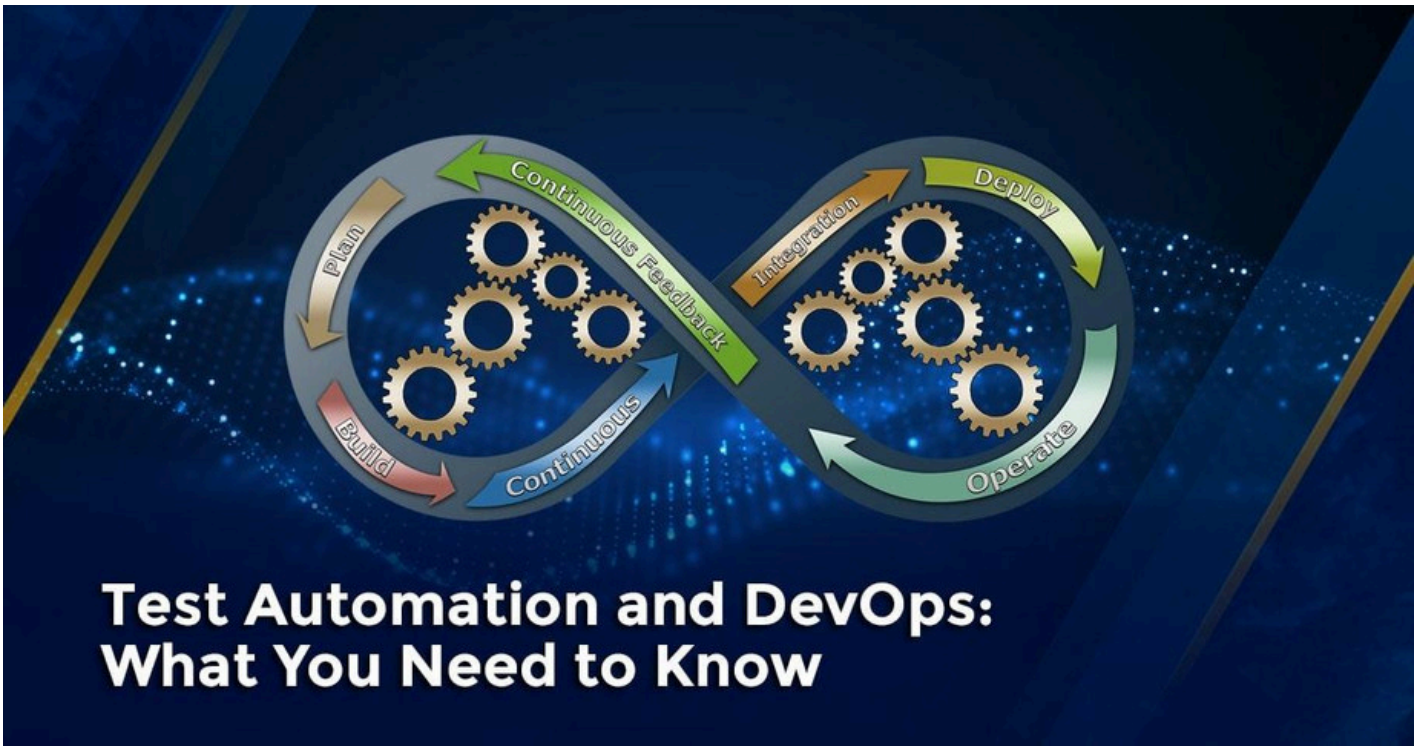

DevOps has become the go-to methodology for enterprises to accelerate software development and delivery, minimize or eliminate risks, reduce costs, and deliver superior customer experiences. However, it is not always easy to build a DevOps-driven process as it entails many things, such as continuous testing at higher speeds. This is where automation, especially DevOps test automation, kicks in to achieve continuous testing, an essential pillar of DevOps.

Why adopt DevOps automation in the SDLC?

DevOps runs on three principles - continuous integration (CI), continuous development (CD), and continuous testing (CT). Here, continuous testing means testing early and more often in the DevOps pipeline with the goal of ‘failing fast’. In other words, catching bugs or glitches early in the code under testing and fixing them to avoid costly fixes later. To understand the need to adopt continuous test automation in DevOps, consider the situation of manual testing:

A developer sends a piece of code into the code repository and begins working on another segment of the application. Now, when the code enters the build system and is tested against a set of parameters, it fails. The code is sent back to the developer for fixing, who at that point in time might have moved on to developing many other pieces of code. The developer, upon receiving the failed code, needs to recall and figure out the fault. After fixing, the code is sent back to the tester for rechecking, thus making the process extremely time-consuming, inefficient, and cost-intensive.

The above-mentioned situation can be avoided by adopting test automation where fast feedback loops are available and developers are notified within hours, if not minutes, about the presence of bugs in a code. Further, automation reduces the risk of missing out on identifying errors that can typically happen when testers are burdened with repetitive work. Business enterprises with voluminous sets of sensitive data can implement DevOps test automation to minimize risks by limiting human interaction from the equation.

Automation in testing can help continuous testing services to increase the scope of testing. Given that many applications need to be tested across environments, device configurations, operating systems, and browsers, test automation can help facilitate testing in parallel. In fact, automation allows more tests to be done quickly, at lower costs, and with fewer risks. This way, QA testers can take out valuable time for other tasks requiring critical thinking, such as test design and exploratory testing.

How to start automation in DevOps testing - the best practices

At the outset, the testers should map out the release pipeline by the following steps:

- Determining the stages of software release
- Identifying the requirements and gates for a build's journey into production
- Identifying the feedback mechanism for error detection and fixing
- Build the automation flow using a tool
- Choose the right tool that every tester is familiar with and can be reused

Build the automation flow and increase test coverage: Since repetitive flows are time-consuming and risk-prone, it is better to identify such flows for automation. Also, identify the build flows that are easy and convenient to automate.

Test one at a time: It is easy to handle one test at a time rather than bundling multiple tests within one test case. Also, the test automation tool should help build reusable components to minimize the time it takes to create new test cases.

Build independent and self-contained test cases: It is better to use separate and self-contained test cases in DevOps implementation so that at any given point in time, they can be scheduled for execution across different environments in parallel. This can reduce the test complexity and increase the speed.

Ownership of test automation: Continuous testing in DevOps can be successful with the collective knowledge of the team. Choose a test tool and a platform that every tester is familiar with. This is important to ensure automation becomes a natural aspect of everybody's work schedule.

Conclusion

With software applications becoming more complex and having multiple touchpoints across devices, browsers, operating systems, and networks, they need to be developed, tested, and deployed using the DevOps methodology to achieve speedy execution, minimal risks, enhanced quality, and better compliance across operating environments. It is only through a continuous testing strategy that software quality can be maintained at top-notch levels and businesses remain competitive.

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