Automate Testing and Deployments

Today, developers write automated scripts that can verify thousands of scenarios in minutes and then deploy updated code into production environments multiple times a day. They use automated performance tests which simulate surges in traffic to identify performance bottlenecks. While manual tests and quality assurance are still necessary, automated tests provide consistent and reliable protection against unintentional regressions, and make it possible for developers to confidently release frequent updates to the service.

Key Questions

1. What percentage of the code base is covered by automated tests?
2. How long does it take to build, test, and deploy a typical bug fix?
3. How long does it take to build, test, and deploy a new feature into production?
4. How frequently are builds created?
5. What test tools are used?
6. Which deployment automation or continuous integration tools are used?
7. What is the estimated maximum number of concurrent users who will want to use the system?
8. How many simultaneous users could the system handle, according to the most recent capacity test?
9. How does the service perform when you exceed the expected target usage volume? Does it degrade gracefully or catastrophically?
10. What is your scaling strategy when demand increases suddenly?

Checklist

Create automated tests that verify all user-facing functionality

Create unit and integration tests to verify modules and components

Run tests automatically as part of the build process

Perform deployments automatically with deployment scripts, continuous delivery services, or similar techniques

Conduct load and performance tests at regular intervals, including before public launch

---

Revision #1
Created 26 February 2024 16:20:04 by Tom O'Malley
Updated 26 February 2024 16:20:44 by Tom O'Malley