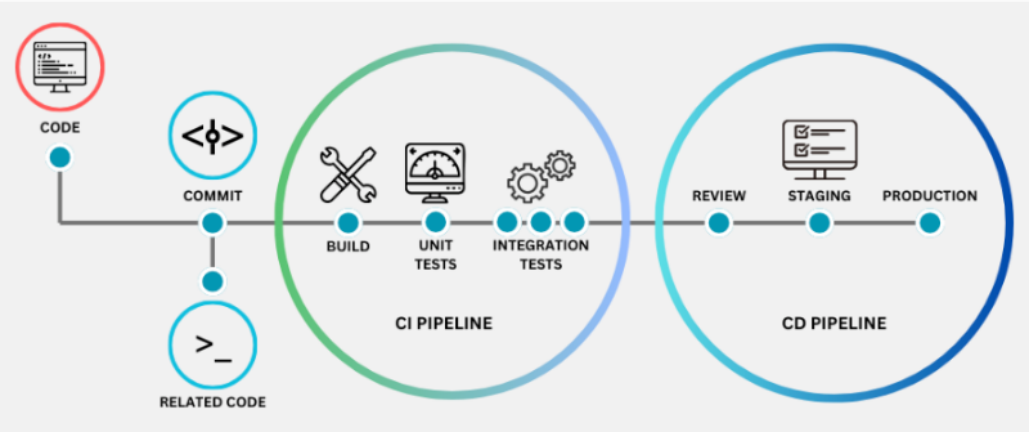

How to quickly set up a CI/CD Pipeline to Run Automated Tests

Introduction

In Continuous integration, developers keep integrating their developed code in a common and shared repository. In Continuous delivery or “Continuous Deployment” the codes that are merged by the developers are continuously delivered to the live environment.

Why do you need continuous testing in such a system?

- The development team works in parallel and the team members keep integrating their code in the repository
- This keeps updating the master branch regularly. And so there arises a need to continue testing the code to keep the software quality. It has to be ensured that the testing goes on at the same pace.
- Manual testing will not be a good choice in such a scenario. Automated testing though can promise you successful testing in CI/CD pipeline.
- Since the CI model supports the parallel testing of the code it helps in early identification of problems.

Why Automate Testing in CI/CD?

In CI, the testing has to keep up with the code development. Builds has to be tested regularly for any changes and modifications. To keep up the pace and deal with the regular testing requirements, automation testing is preferred in the CI model.

Testing Stages in Continuous Integration/Continuous Delivery Model

The testing process goes through the following stages in the CI/CD model:

- **Develop:** the development team develops the code as per the requirements.
- **Writing tests:** After code is developed, the tests are written for that code.
- **Local Testing**: The local testing of the code is done to ensure the quality of the code.
- **Rebase and Resolve conflict**: in the CI model, the developers keep updating the branches with the code. This process is called “rebasing”. These codes need to be tested and the code is passed back to the developers to fix the defects
- **Commit:** When the code passes the test, it is committed.
- **Build:** The source code is combined and deployed to many testing environments.

- **UAT:** Developed code is deployed to a test server.
- **Merge:** if the commit under test if passed it is merged with the master branch
- **Production deployment:** after merging the code is passed on for the production.

This process is repeated for every build.

Where Does Automation Testing Fall in This CI/CD Pipeline?

Automated tested is done when the build is completed and code is read7y to be deployed. At this stage, the automated unit tests, integration tests and UI tests can be conducted.

Tests can also be parallel done to fasten testing speed.

Tools Used for CI/CD

- **Jenkins:** Jenkins is a commonly used testing tool for continuous integration. It is open-source and free of cost.
- **Travis CI:** Hosted by GitHub, Travis CI is a free and open-source testing tool for continuous integration approach.
- **Gitlab:** Available in both paid and free versions, Gitlab has its own cloud-based CI methodology. Gitlab can be supported by multiple platforms.
- **Bamboo:** Bamboo is a popular testing tool for continuous integration provided by JIRA. Bamboo is a good option for companies using JIRA.

Best Practices

- **Incremental changes**: following an incremental approach is always beneficial. Breaking down the complex functionalities into smaller ones can help accomplish the testing task faster. It also helps in finding the root cause of the defects (if found) easily.
- **Identify what can be automated**: Automating everything may seem to be an easier approach, but it is

not a viable and a smarter thing to do. Identify and pick what should be automated to get maximum benefit.

- **Parallel Tests**: running the test cases in parallel is a big thumps up. Executing the test cases in parallel can save you a lot of time and initiates more efficient and timely testing. Apart from the parallel execution of test cases, scaling the size of the servers plays an equally important role to increase the speed of the testing.

Setting up the CI/CD pipeline for automated testing

- Setup continuous integration (CI) servers

Setup a build server/ continuous integration (CI) server. You can opt for continuous integration (CI) server. After setting up the server, set up the first build. A build contains:

- Application's source code link
- Command to build the application

This is set up whenever a change is done to the source code.

- Setup test suites

Setup the test suite to test your build. A separate build is set for your test suites.

- Add a deployment step

The deployment has to be linked to the test suite so that whenever your build passes all the tests the changes are deployed to production.

Practices to avoid

- **Duplication:** Do not duplicate test scenarios as it will slow down the test pipelines. No waits for the feedback if the test time crosses a limit and hence the complete testing process becomes of no use.
- **Limited automation**: Don't just go on automating everything, be wise in what to automate and what not.
- **Only QAs owning test pipelines**: Because of regular changes, the tests start failing after a specific commit.

So developers should monitor automation pipelines so that they can fix bugs as in when they come.

Benefits

Here are few benefits of using automated testing in CI/CD model:

- **Low Cost**: since the testing takes place in parallel with the development, the bugs are found earlier and are fixed at an early stage. It saves a lot of cost of finding and fixing bugs at a later stage.
- **Faster Release Cycles**: Automation testing logically increases the speed of testing and hence increasing the complete software process and speeding the release cycle.
- **High Quality:** the constant updates and automated regression and parallel tests make sure that the code is bug-free and compatible with multiple environments. It also ensures higher customer satisfaction.
- **Risk Reduction**: CI supports regular addition, testing and fixing of bugs. This reduces the risk of finding bugs at a later stage drastically.
- **Better ROI**: With regular bug fixation businesses ensures a higher quality product in reduced time and hence promising better ROI

Conclusion

CI/CD model is a very popular and helpful model in software development. Automation testing adds to its value by enhancing the testing of the software. But any faulty practice can increase your work and can decrease the efficiency of the complete process. So be very wise while setting up and performing automation testing in CI/CD model.
