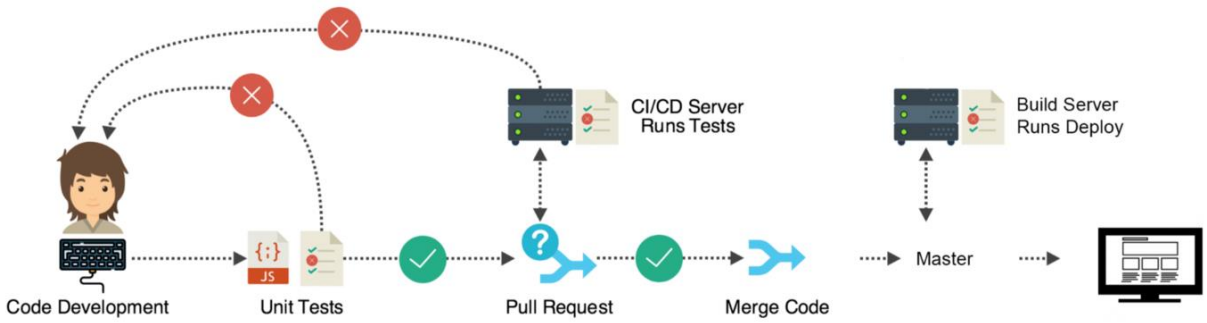
Fundamentals and Benefits of CI/CD

One of the key principles for implementing DevOps is the CI/CD pipeline, which connects the development and operations teams. Delivering code more frequently and reliably is the best practice for DevOps teams. It is the backbone of the modern development environment. Using CI/CD, teams may completely automate the related build, test, and deployment procedures.

Continuous Integration (CI) takes care of integrating the working copies of each individual in the development team into a shared mainline several times a day. Most of the work related to the code happens here. The purpose of CI is to produce a high quality and deployable artifact.

Continuous delivery (CD) picks up where continuous integration ends, and automates application delivery to selected environments, including production, development, and testing environments. Everything regarding the deployment of the artifact is consistent here. Continuous delivery is an automated way t o apply code changes to these environments.



Let's examine some of the advantages of the CICD below to understand the importance and need of the organisation.

1. Reduce Risk

Finding and fixing bugs late in the development process is costly and time intensive. This is particularly true when there are problems with functionality that have already been released to production.

With a CI/CD pipeline, you can test and deploy code more frequently, giving testers the ability to detect issues as soon as they occur and to fix them immediately. You're basically reducing risk in real time.

2. Delivery Faster

Organizations are moving toward releasing features multiple times a day. This is not an easy task. Not all businesses can do this. But with a smooth IC/CD pipeline, multiple daily releases can be achieved.

Teams can build, test and deploy functionality automatically without virtually any manual intervention. This saves time and boosts your business's revenue.

3. Decrease manual effort

Automation is required from the beginning if we are to conform to the shift-left paradigm. This is also a vital component of having a successful CI/CD implementation. Tests should start immediately after building features and checking in code to ensure that neither the new code nor the new features damage any already-existing features. Downtime from a deploy-related crash or major bug is reduced.

Following the completion of the tests, the code is deployed to various environments, such as QA, staging, and production. You will continuously receive messages regarding the development, test, and deploy cycles

during this process through various channels, providing you with a wealth of information.

4. Generate extensive logs

One of the most important aspects of DevOps and CI/CD integration is observability. If something is wrong, you must figure out why. A mechanism is required to study the system in production over time and identify key performance metrics. Observability is a technical solution that aids in this effort.

Logging information is an important aspect of observability. Logs are a rich source of information for understanding what is going on beneath the UI and studying application behavior.

A CI/CD pipeline generates a large amount of logging data at each stage of the development process. There are several tools available to effectively analyze these logs and provide immediate feedback about the system.

5. Facilitate roll-backs

One of the most significant benefits of a CI/CD pipeline is the ability to quickly roll back changes. If any new code changes break the production application, you can restore it to its previous state immediately. To avoid production outages, the most recent successful build is usually deployed right away.

Rapid release cycles are becoming the norm, and CI/CD pipelines have sped up the release pace. Such a pipeline can aid in improving customer satisfaction by enabling you to identify faults more quickly, carry out remedies right away, and prepare for them carefully.