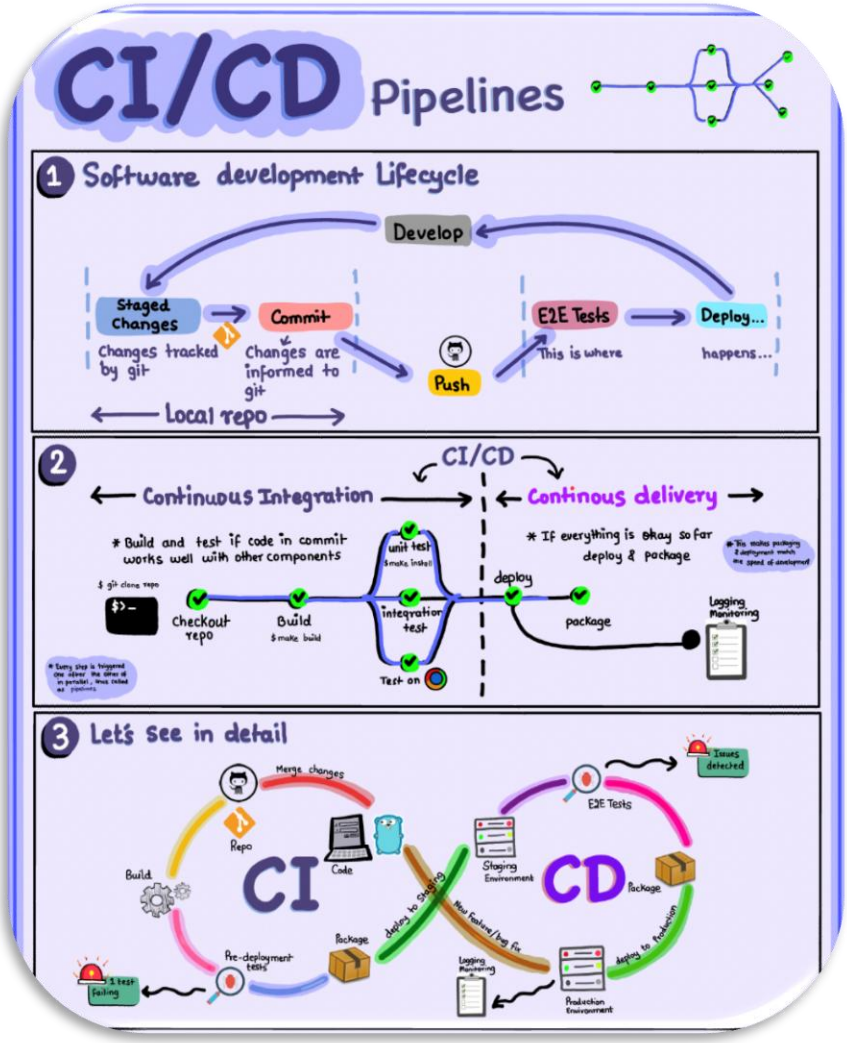
CI/CD Pipeline

What is CI/CD?

Use CI/CD to automate your software development workflows and deploy better quality code, more often. Using a continuous and iterative process to build, test, and deploy helps avoid bugs and code failures.



CI/CD falls under DevOps (the joining of development and operations teams) and combines the practices of continuous integration and continuous delivery. CI/CD automates much or all of the manual human intervention traditionally needed to get new code from a commit into production, encompassing the build, test (including integration tests, unit tests, and regression tests), and deploy phases, as well as infrastructure provisioning. With a CI/CD pipeline, development teams can make changes to code that are then automatically tested and pushed out for delivery and deployment. Get CI/CD right and downtime is minimized and code releases happen faster.

Why it is important?

In today's fast-paced technological landscape, Continuous Integration and Continuous Delivery (CI/CD) are more than just industry buzzwords—they represent a crucial foundation for a modern software development process.

CI/CD is crucial because it automates the software development process, from coding through deployment. This automation means teams can release new features and fixes faster and more frequently, enhancing the product's responsiveness to user needs. By continuously integrating and deploying, errors are detected sooner, reducing downtime and improving software quality.

CI/CD also allows for quicker feedback loops with stakeholders, ensuring that the final product aligns closely with user expectations. Overall, it's a foundational practice for any team aiming for high-speed, high-quality software development.

What is continuous integration(CI)?

Continuous integration is the practice of integrating all your code changes into the main branch of a shared source code repository early and often, automatically testing each change when you commit or merge them, and automatically kicking off a build. With continuous integration, errors and security issues can be identified and fixed more easily, and much earlier in the development process.

By merging changes frequently and triggering automatic testing and validation processes, you minimize the possibility of code conflict, even with multiple developers working on the same application. A secondary advantage is that you don't have to wait long for answers and can, if necessary, fix bugs and security issues while the topic is still fresh in your mind.

Common code validation processes start with a static code analysis that verifies the quality of the code. Once the code passes the static tests, automated CI routines package and compile the code for further automated testing. CI processes should have a version control system that tracks changes so you know the version of the code used.

What is continuous delivery(CD)?

Continuous delivery is a software development practice that works in conjunction with CI to automate the infrastructure provisioning and application release process.

Once code has been tested and built as part of the CI process, CD takes over during the final stages to ensure it's packaged with everything it needs to deploy to any environment at any time. CD can cover everything from provisioning the infrastructure to deploying the application to the testing or production environment.

With CD, the software is built so that it can be deployed to production at any time. Then you can trigger the deployments manually or move to continuous deployment, where deployments are automated as well.

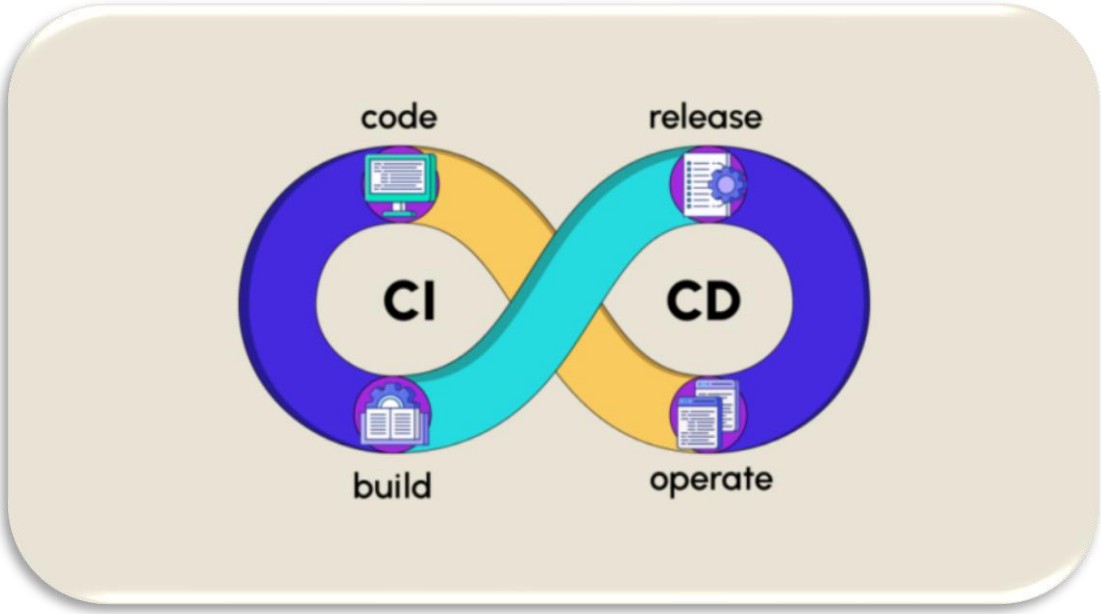
What is continuous deployment(CD)?

Continuous deployment enables organizations to deploy their applications automatically, eliminating the need for human intervention. With continuous deployment, DevOps teams set the criteria for code releases ahead of time and when those criteria are met and validated, the code is deployed into the production environment. This allows organizations to be more nimble and get new features into the hands of users faster.

While you can do continuous integration without continuous delivery or deployment, you can't really do CD without already having CI in place. That's because it would be extremely difficult to be able to deploy to production at any time if you aren't practicing CI fundamentals like integrating code to a shared repo, automating testing and builds, and doing it all in small batches on a daily basis.

What are CI/CD pipelines?

A CI/CD pipeline is an automated process utilized by software development teams to streamline the creation, testing and deployment of applications. "CI" represents continuous integration, where developers frequently merge code changes into a central repository, allowing early detection of issues. "CD" refers to continuous deployment or continuous delivery, which automates the application's release to its intended environment, ensuring that it is readily available to users. This pipeline is vital for teams aiming to improve software quality and speed up delivery through regular, reliable updates.



Integrating a CI/CD pipeline into your workflow significantly reduces the risk of errors in the deployment process. Automating builds and tests ensures that bugs are caught early and fixed promptly, maintaining high-quality software.

How does CI/CD fits into DevOps framework?

CI/CD is a cornerstone practice within the DevOps framework. It bridges the gap between development (Dev) and operations (Ops) through automation and continuous processes. By automating the build, test, and deployment phases, CI/CD enables rapid, reliable software releases. Due to this, it aligns closely with DevOp's goals of improving collaboration, efficiency, and product quality.

As an indispensable component of DevOps and modern software development, CI/CD leverages a purpose-built platform to optimize productivity, increase efficiency, and streamline workflows via automation, testing, and collaboration. This is particularly beneficial as applications scale, helping to simplify development complexity. Moreover, integrating CI/CD with other DevOps practices—such as enhancing security measures early in the development process and tightening feedback loops—enables organizations to overcome development silos, scale operations securely, and maximize the benefits of CI/CD.