Ayush Patil - 4NI19IS020 - A SECTION

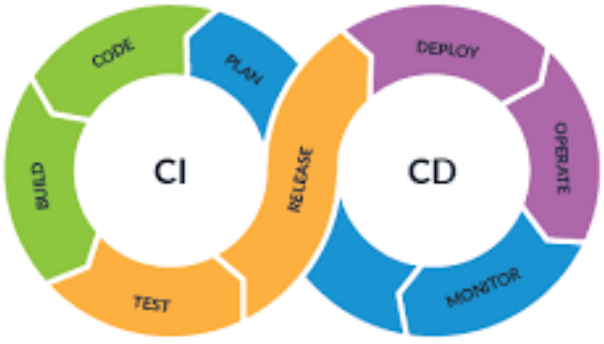
ASSIGNMENT – 1

What is CI/CD?

CI/CD stands for Continuous Integration/Continuous Delivery. It is a software development practice that involves regularly integrating code changes into a shared repository, testing those changes automatically, and then deploying them to production. The goal of CI/CD is to reduce the time and effort required to release new features and updates to customers, while also increasing the quality and reliability of the software. This is typically accomplished through the use of automated tools and processes that handle the build, test, and deployment phases of the software development lifecycle.

There are several advantages to using CI/CD in software development:

Faster deployment: By automating the build, test, and deployment process, teams can deploy new features and updates to customers more quickly, reducing the time it takes to get new features into the hands of users. **Improved quality:** CI/CD enables teams to catch and fix issues early in the development process, which can help prevent defects from being released to customers. **Increased collaboration:** By integrating code changes regularly, teams can work more closely together and ensure that everyone is working on the most up-to-date version of the software.**Better customer satisfaction:** By releasing new features and updates more frequently, teams can keep customers happy and engaged with their products.



A real-life example of a company using CI/CD is **Netflix.** The company has implemented a continuous delivery pipeline that allows them to deploy new code changes to production multiple times per day. This has helped them reduce the time it takes to release new features, improve the reliability of their software, and increase customer satisfaction.

What is CI/CD Pipeline?

A CI/CD pipeline is a series of automated processes that handle the build, test, and deployment phases of the software development lifecycle. It typically includes the following steps:

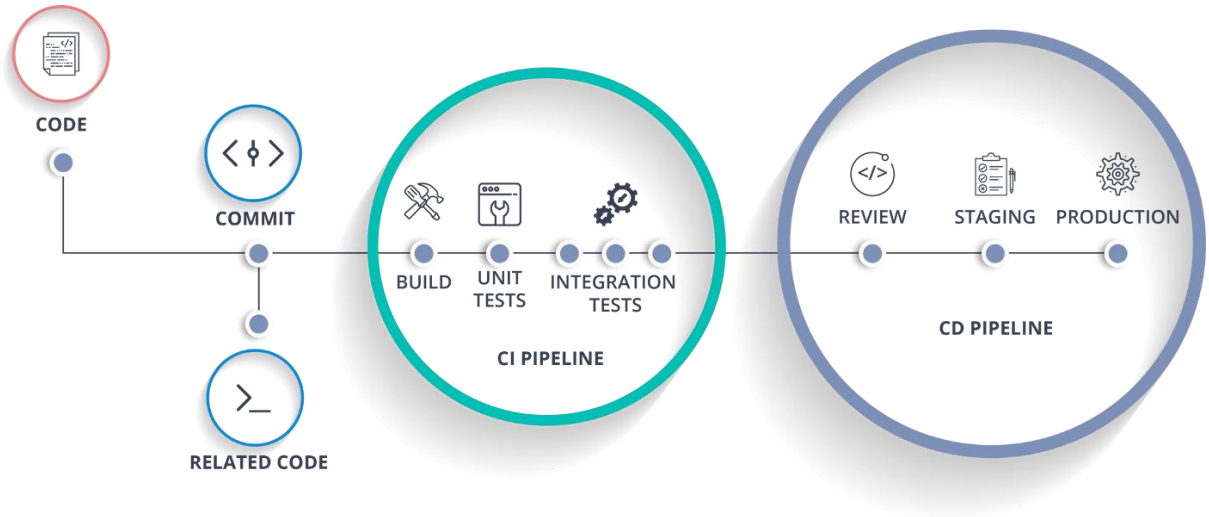
Continuous integration: This involves regularly integrating code changes into a shared repository, where they are automatically built and tested.

Testing: This can include automated testing to ensure that the code changes do not introduce any new defects or issues.

Deployment: Once the code has been successfully tested, it can be deployed to production or to staging environments for further testing and validation.

Continuous delivery: This involves automatically deploying code changes to production once they have been successfully tested, without the need for manual intervention.

The goal of a CI/CD pipeline is to automate the software development process as much as possible, so that teams can focus on creating new features and improvements rather than spending time on manual processes. It also helps ensure that the software is of high quality and can be released to customers quickly and reliably.



A real-life example of a CI/CD pipeline is the one used by Amazon Web Services (AWS). AWS has implemented a pipeline that automatically builds, tests, and deploys new code changes to production several times per day. This has helped the company reduce the time it takes to release new features and updates, while also improving the reliability and quality of their software.

Feature Flags - Feature flags, also known as feature toggles, are a technique used in software development to enable or disable specific features or functionality in a software application.

What is the use of CI/CD Pipeline?

Automating the build, test, and deployment process: By automating these tasks, teams can reduce the time and effort required to release new features and updates to customers.

Improving the quality and reliability of software: By automatically testing code changes and deploying them to production, teams can catch and fix issues early in the development process, which can help prevent defects from being released to customers.

Increasing collaboration and agility: By integrating code changes regularly and deploying them to production multiple times per day, teams can work more closely together and respond more quickly to customer needs.

Reducing the risk of deployment: By automating the deployment process and using feature flags, teams can roll out new features and updates to a small subset of users before rolling them out to all users. This can help reduce the risk of deployment and allow teams to gather feedback and data on the impact of new features.

Improving customer satisfaction: By releasing new features and updates more frequently, teams can keep customers happy and engaged with their products.

Some Real Time use cases are :

Netflix: Netflix has implemented a continuous delivery pipeline that allows them to deploy new code changes to production multiple times per day. This has helped them reduce the time it takes to release new features and updates, while also improving the reliability and quality of their software.

Amazon Web Services (AWS): AWS has a CI/CD pipeline that automatically builds, tests, and deploys new code changes to production several times per day. This has helped the company release new features and updates more quickly and with less risk, while also improving the reliability and quality of their software.

Google: Google has implemented a CI/CD pipeline for its cloud services platform, which allows teams to deploy new code changes to production multiple times per day. This has helped the company release new features and updates more quickly and with less risk, while also improving the reliability and quality of their software.

Uber: Uber has implemented a CI/CD pipeline for its ride-hailing platform, which allows teams to deploy new code changes to production multiple times per day. This has helped the company release new features and updates more quickly and with less risk, while also improving the reliability and quality of their software.

Example of CI/CD Tools –

1) Jenkins 2) Travis CI 3) CircleCi 4) GitLab CI 5) Azure Devops