Ci/cd pipeline using jenkins

1. What is Jenkins?

Jenkins is a tool that is used for automation and it’s a open source server that allows the developers to build, test and deploy software . By using Jenkins we can make continuous integration of (projects) or end -to-endpoint automation. Jenkins is entirely developed using Java.

1. Why we have to use only Jenkins?

Jenkins is a popular **open-source automation server** that provides a wide range of features and plugins for continuous integration and continuous delivery (CI/CD) pipelines. It allows you to automate the build, test, and deployment processes, and can be easily integrated with various tools and platforms.

By using Jenkins we can reduce our time to 25-30% of time,

One of the **main advantages of Jenkins is its flexibility and customizability**. With its extensive plugin ecosystem and support for various programming languages and technologies, Jenkins can be tailored to meet the specific needs of your project. Additionally, **Jenkins is well-documented, has a large user community, and provides robust security features.**

That being said, there are other automation tools and platforms available that you may want to consider, depending on your specific requirements and preferences. Some examples include Travis CI, Circle CI, GitLab CI/CD, and Azure DevOps.

1. What is continuous integration?

Continuous Integration is a software development practice that involves frequently integrating code changes to the shared code repository where automatic builds and tests are performed.

Main goal of this continuous integration is to catch and fix the issues early in the development process, reducing the risk of integration issues and ensuring that the software is always in a releasable state.

**Real time example**:

A real-time example for continuous integration could be a software development team working on an e-commerce website.

The team uses a continuous integration tool, such as Jenkins to automatically build and test the website every time a code change is committed to the code repository.

For instance, when a developer submits new code to the website, the continuous integration tool automatically pulls the code from the repository, builds the application, and runs a suite of automated tests to verify that the changes have not introduced any issues.

If any issues are found, the team is notified immediately, and they can fix the issues before they become more significant problems.

**MAJOR BENEFIT:**

By using continuous integration, the development team can catch issues early in the development process, ensure that the website is always in a working state, and improve the overall quality of the software.

1. Why only continuous Integration?

If we did not use CI , we have to check all the things manually, such as pulling code from the repo, building the code, run the code, test the code, finding the bugs.

Simply by using the CI we can perform all the things using automation.

1. What is Continuous Deployment/Delivery?

Continuous Deployment is a software development practice where code changes are automatically deployed to **production environments** after passing through automated testing and validation stages, ensuring that software updates can be released to end-users quickly and reliably.

To implement Cd, organizations typically use a combination of automation tools, infrastructure-as-code, and version control systems. This enables them to automate the entire deployment process and achieve high levels of reliability, scalability, and efficiency.

**Real time example:**

A real-time example for continuous delivery and deployment could be a software development team working on a mobile application for a bank.

**The team uses a continuous delivery and deployment tool, such as Circle CI or GitLab CI/CD, to automatically build, test, and deploy the mobile application every time a code change is committed to the code repository**.

For instance, when a developer submits new code to the mobile application, the continuous delivery tool automatically pulls the code from the repository, builds the application, runs a suite of automated tests, and packages the application for release to production.

**Once the application is built, tested, and packaged, the continuous deployment tool deploys the application to the production environment, making it available for use by bank customers.**

By using continuous delivery and deployment, the development team can quickly and reliably release new features and updates to the mobile application, ensuring that the bank's customers have access to the latest and most secure version of the application