**CI/CD**

**CI (Continuous Integration):** Continuous Integration is a process where we integrate a set of tools/processes that we follow before delivering application to a customer

**CD (Continuous Delivery): -** continuous deliveryis a process where we deploy our application on a specific platform to the customer

Some standard steps that we follow are

1. Unit Testing
2. Static code analysis
3. Code quality/ vulnerability testing
4. Automation testing
5. Reporting
6. Deploying the application

**Flow of the CICD Pipeline**

* Assuming **GIT** as the version control system and **JENKINS** as the orchestrator and **Kubernetes** is for deploying the application

**CONTINUOUS INTEGRATION**

1. When the user (developer) commits the code to the GIT. The GitHub webhook triggers the pipeline in the Jenkins
2. Then the continuous integration (CI) starts. As part of CI there are multiple stages
3. **Checkout stage: -** As part of this stage we will checkout the code that the user has committed
4. **Build action along with the unit testing:** - Maven for building. And using unit testing framework to test the u **🡪 This is for java application**
5. **Code Scanning: -** As part of the code scanning, we use SonarQube to scan the code for any security related vulnerabilities, to ensure code is free from any security related issue.
6. **Image Build: -** As we are deploying the application in Kubernetes. We need to build docker image for that we need to have docker file in the GitHub repository
7. **Image Scanning:** - Check whether the docker image has any vulnerabilities
8. **Image Push:** - After image scanning, we need to push the docker image to docker hub

These are the multiple stages in the continuous integration and we write Jenkins file in Jenkins for orchestrating each of them.

* **We use the declarative Jenkins pipeline**

**CONTINUOUS DELIVERY**

1. When the image pushed to the repository then continuous delivery takes place
2. Using the same Jenkins pipeline, we will update the GIT repo with image that we have created in the last stage of the CI
3. Using Argo cd, we will deploy the new change on to the Kubernetes
4. Argo CD continuously watching the Git repository in which we have the image pushed and all the Kubernetes manifest files are there
5. Wherever we are pushing updated K8’s yaml manifests there we have to configure Argo CD to watch that repository and push the changes to the Kubernetes Cluster
6. We can use ansible, shell or python scripts in place of Argo CD if we wanted to

