# CI/CD ASSIGNMENT

1) What is Continuous Integration?

   The build and unit testing phases of the software release process are referred to as continuous integration. Every time a revision is committed, an automated build and test are started. Code updates are automatically built, tested, and ready for release to production when using continuous delivery.In other words, automating the merging of code changes made by various contributors into a single software project is known as continuous integration (CI).

2) What is Continuous Deployment?

   Code updates to an application are automatically pushed into the production environment as part of a software development method called continuous deployment. A set of specified tests serve as the engine for this automation. The mechanism directly delivers updates to users of the software after they pass those tests. When teams rely on a fully automated pipeline, this is essentially known as continuous deployment.

3) Can you describe an example of a CI (and/or CD) process starting the moment a
   Developer submitted a change/PR to a repository?

   Possible stages are:

   - Compile
   - Build
   - Install
   - Configure
   - Update
   - Test

   To a project, a developer sent a pull request. Two jobs were started by the PR (pull request) (or one combined job). One job is to run a lint test on the submitted change, and the other is to build a package containing the change and run various api/scenario tests using that package. The change is merged/pushed to the repository once all tests have passed and it has received the maintainer's or core's approval. The change will not be accepted for merging or pushing to the repository if part of the tests failed. The developer sends code to a repository, which then starts a workflow to create a container image and push it to the registry, according to a completely different response or CI process. Once in the registry, the updated changes are applied to the k8s cluster.

4) What is Continuous Delivery?

A development approach that involves delivering code to QA and Operations often for testing. This calls for a staging area with features similar to a production environment, where changes can only be approved for production following a manual review. When opposed to continuous deployment, there is typically a gap between release and review due to this human involvement, which makes it slower and more error-prone.

5) What is difference between Continuous Delivery and Continuous Deployment?

All of the commits made to a single codebase are collected by continuous integration (CI). Even when several people are simultaneously working on the same piece of code, integration with the main trunk is continuous.

Continuous Delivery (CD) pulls commits automatically through the building, testing, and packaging for deployment stages of the pipeline. When prepared, it can either be released manually or automatically.

Without requiring human intervention, Continuous Deployment (CD) pushes a deployment right into the production environment. In contrast to CD, which leverages automation to speed up testing, staging, and validation so that code may be deployed at the touch of a button, CI enhances cooperation throughout the development process. Quality gates assess commits against the crucial standard at various stages.

6) What CI/CD best practices are you familiar with? Or what do you consider as CI/CD best practice?

Continuous integration, delivery and deployment are software development practices born out of the DevOps movement. They make the process of building, testing and releasing software more efficient and get working software into the hands of users more quickly than traditional methods.Automated process of building, testing and deploying software

Commit and test often Testing/Staging environment should be a clone of production environment

7)You are given a pipeline and a pool with 3 workers: virtual machine, baremetal and a container. How will you decide on which one of them to run the pipeline?

Bare-metal containers reduce the number of layers to manage versus VM-based containers, and because bare metal is more efficient, you need less hardware to run the same number of containers, reducing the total amount of equipment under management.

8) Where do you store CI/CD pipelines? Why?

- App Repository – store them in the same repository of the application they are building or testing
- Central Repository – store all organization's/project's CI/CD pipelines in one separate repository (perhaps the best approach when multiple teams test the same set of projects and they end up having many pipelines)
- CI repo for every app repo – you separate CI related code from app code but you don't put everything in one place (perhaps the worst option due to the maintenance)

9)How do you perform plan capacity for your CI/CD resources? (e.g. servers, storage,etc.)

The CI/CD pipeline combines continuous integration, delivery and deployment into four major phases: source, build, test and deploy. Each phase uses highly detailed processes, standards, tools and automation.

10) How would you structure/implement CD for an application which depends on several other applications?

Code: Checked into the repository.

Build: Build is triggered and deployed in a test environment.

Test: Automated tests are executed.

Deploy: Code is deployed to stage, and production environments.

11) How do you measure your CI/CD quality? Are there any metrics or KPIs you are using for measuring the quality?

DevOps organizations monitor their CI/CD pipeline across three groups of metrics:

- Automation performance

- Speed
- Quality

Metrics:

- Agile CI/CD Pipeline
- Time to value

12) What is Jenkins? What have you used it for?

Jenkins is an open-source free automation tool used to build and test software projects. The tool makes it painless for developers to integrate changes to the project. Jenkins' primary focus is to keep track of the version control system and initiate and monitor a build system if there are any changes.