DevOps Assignment 1

CI-CD

1. What is CI/CD?

CI/CD is a method to frequently deliver apps to customers by introducing automation into the stages of app development. The main concepts attributed to CI/CD are continuous integration, continuous delivery, and continuous deployment. CI/CD is a solution to the problems integrating new code can cause for development and operations teams.

The acronym CI/CD has a few different meanings. The "CI" in CI/CD always refers to continuous integration, which is an automation process for developers. Successful CI means new code changes to an app are regularly built, tested, and merged to a shared repository. It's a solution to the problem of having too many branches of an app in development at once that might conflict with each other.

The "CD" in CI/CD refers to continuous delivery and/or continuous deployment, which are related concepts that sometimes get used interchangeably. Both are about automating further stages of the pipeline, but they're sometimes used separately to illustrate just how much automation is happening.

2. What are feature flags and how it is used?

Feature flags (also known as feature toggles or feature switches) are a software development technique that turns certain functionality on and off during runtime, without deploying new code. This allows for better control and more experimentation over the full lifecycle of features. The idea behind feature flags is to build conditional feature branches into code in order to make logic available only to certain groups of users at a time. If the flag is "on," new code is executed, if the flag is "off," the code is skipped. Also referred to

as or release toggles, feature flags are a best practice in DevOps, often occurring within distributed version control systems.

• Feature flags can be used to test new features with a subset of users before releasing them to the entire user base, or to gradually roll out a feature to all users.
• They can also be used to quickly disable a feature in the event of a problem or to experiment with different versions of a feature.
• In DevOps, feature flags are often used in conjunction with continuous delivery and deployment practices.
• Feature flags are typically implemented using a combination of code and configuration.
• They can be managed through a variety of tools and platforms, including feature flag management platforms, version control systems, and configuration management systems.

An example of how feature flags might be used in a DevOps workflow:
• A developer writes and tests a new feature on their local machine.
• The developer commits the code changes to a version control system (such as Git) and pushes them to a central repository.
• The code is automatically built and deployed to a staging environment.
• The developer creates a feature flag to toggle the availability of the new feature on or off.
• The developer uses the feature flag to enable the new feature for a small group of users, such as a subset of the staging environment or a group of beta testers.
• The team monitors the usage and performance of the new feature to ensure that it is working as expected.
• If the new feature is stable and performs well, the team can use the feature flag to roll it out to a wider audience. If there are any issues with the feature, the team can use the feature flag to quickly disable it.

3. Explain CI/CD pipeline with block diagram.

CI/CD, or Continuous Integration/Continuous Deployment, is a software development practice that aims to reduce the time it takes to deliver new features and updates to users by automating the build, test, and deployment processes.

In a CI/CD pipeline, code changes are automatically built, tested, and deployed to production as soon as they are committed to the source code repository. This allows development teams to deliver new features and bug fixesto users faster and with fewer errors, as the pipeline catches any issues at the early stages of development.

Here is a high-level overview of the steps involved in a typical CI/CD pipeline:**Code Commit**: Developers commit their code changes to a version control system (such as Git) and push them to a central repository.

Build: A build system retrieves the latest code from the repository and compilesit into a deployable package (such as a Docker image). The build process may also include tasks such as running tests and generating documentation.

Test: The build is deployed to a staging environment and automated tests arerun to ensure that the code is working as expected.

Deploy: If the tests pass, the code is deployed to production. If the tests fail, thedeployment process is halted and the developers are notified of the issue.

Monitor: Once the code is deployed to production, it is monitored for any issues or errors.

By automating these steps, a CI/CD pipeline allows development teams to quickly and confidently deliver new code changes tousers, while also ensuring that the code is of high quality.