CI/CD Interview Questions with Detailed Answers

Q: What is CI/CD? How do you implement it in AWS?

A: CI/CD stands for Continuous Integration and Continuous Delivery/Deployment.

CI: Developers regularly merge code changes into a central repository where automated builds and tests run.

CD: Automatically deploy code to production (Deployment) or staging (Delivery) environments after successful tests.

In AWS, CI/CD can be implemented using:

- AWS CodeCommit (source control)
- AWS CodeBuild (build and test)
- AWS CodeDeploy (deployment to EC2, Lambda, ECS)
- AWS CodePipeline (orchestration of above tools)

Real-time example: In my previous project, we used CodePipeline to connect CodeCommit CodeBuild CodeDeploy for a Spring Boot application hosted on EC2 with automated unit and integration testing.

Q: How does AWS CodePipeline work? Real-time implementation?

A: CodePipeline is a fully managed CI/CD service that automates the build, test, and deploy phases of release.

Components:

- Source Stage: Triggers on code changes (CodeCommit/GitHub)
- Build Stage: Compiles code (CodeBuild)

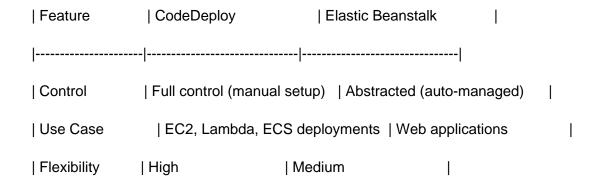
 Test Stage: Rui 	n automated tests
-------------------------------------	-------------------

- Deploy Stage: Deploy using CodeDeploy, Elastic Beanstalk, etc.

Real-time example: In one pipeline, the source was GitHub. CodeBuild used a `buildspec.yml` file to install dependencies, run unit tests, and package the app. CodeDeploy handled EC2 deployment across multiple AZs.

Q: Difference between CodeDeploy and Elastic Beanstalk?

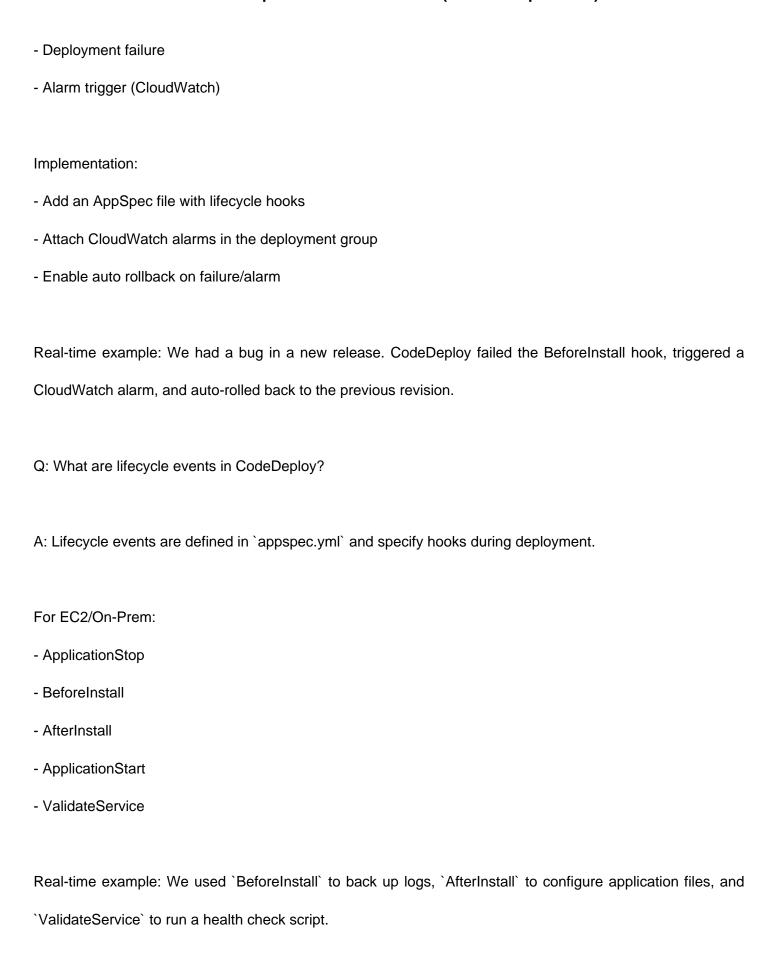
A: CodeDeploy is a deployment automation service. Elastic Beanstalk is a PaaS that manages the complete app lifecycle.



Real-time example: We used CodeDeploy in a project where infrastructure was manually provisioned (VPC, EC2, ASG). For another microservice, Elastic Beanstalk simplified deployment and environment management.

Q: How do you automate rollback in CodeDeploy?

A: You can configure CodeDeploy to automatically roll back on:



Q: How do you test a pipeline before production deployment?
A: To test pipelines safely:
- Use staging/pre-prod environments with identical infra
- Add approval stage in CodePipeline
- Use canary or blue/green deployments
- Perform integration tests in build/test stage
Real-time example: Our pipeline had a manual approval stage after deployment to UAT. Once the QA team
validated the app, they approved the production deployment.
Q: What is a Blue/Green Deployment in AWS?
A: Blue/Green Deployment involves two environments:
- Blue: Current running version
- Green: New version
Traffic is routed to green once it passes testing, minimizing downtime and rollback risk.
In AWS:
- Use CodeDeploy with ELB and Auto Scaling groups
- Configure Blue/Green settings in deployment group
Real-time example: We deployed a Node.js app using CodeDeploy. After validation in green ASG

CodeDeploy rerouted ELB traffic from blue to green instances.

Q: Integrating Jenkins with AWS CodeDeploy how and when?
A: Jenkins can integrate with CodeDeploy for customized pipelines.
Steps:
1. Use CodeDeploy plugin in Jenkins
2. Configure Jenkins job to push code to S3 or Git
3. Trigger CodeDeploy using AWS CLI or plugin
Use it when:
- You need custom scripting/logic
- You already use Jenkins for other projects
Real-time example: In a hybrid cloud setup, Jenkins built Docker images, pushed them to ECR, and triggered
CodeDeploy to deploy the updated app to ECS.