Jenkins

Continuous Integration

Continuous Delivery

Continuous Deployment

Delivery Pipeline

Developer commits code

Build

Test

Stage

Deploy to QA

Continuous Integration = commit + build

Continuos Delivery =

Has Manual Step

commit + build + tests are run on stage (mimic prod) and automate test case (approval required from devops architect) has manual test

Continuous Deployment =

No manual step

everything automated commit to customer

Source Build Test Deploy Monitor

Commit Compile Integration test Pre-Prod HealthChecks

Code Review Unit Test Load test Prod

Program War/Jar System test

Docker Image UI test

JENKINS (continuous deployment, plugin based tool, port 8080)

1. Code and Commit (SVN, GitHub, GitLab)
2. Build and Configuration (Ant, Maven, Gradle, Docker)
3. Testing (JUnit, Sonar, Selenium, Cucumber)
4. Release (Release Management Tools: XL Release, uDeploy, UrbanCode)
5. Deploy (Plugins for AWS cloud, GCP Cloud, OpenStack, OpenShift)
6. Monitor

Jenkins Dashboard

1. Plugins
2. Security
3. Create Jobs
4. Jenkins Pipeline for project

Jenkins require java to work

Setup Jenkins on AWS Linux EC2 AMI instance

sudo --i

yum install Jenkins

Jenkins run on port 8080

Jenkins is plugin based (ant, maven, gradle)

Jenkins uses Jenkins dashboard

Create a new Jenkins job

Manage Different user permission (admin, developers)

Build History (Jobs run/failed)

Manage Jenkins (Configure Path for variables, Manage Plugins)

First Jenkins Job

1. Create a shell script hello.sh

echo “Hello Jenkins!!”

1. chmod 777 hello.sh
2. ./hello.sh
3. “Create new Job” in Jenkins Dashboard
4. Choose “Freestyle Project”
5. Source Code Management
6. Build Triggers (how often it should poll)
7. Build Environment (env variable)
8. Build (execute shell) Commnd : /root/hello.sh
9. Post Build Actions
10. Build Now
11. Success -> Blue, Fail -> Red
12. Console Output

Java Maven Jenkins Job

1. Choose “Freestyle Project”
2. Give path of github repository source code URL
3. mvn clean install
4. Build success or Failure

Build Trigger

1. cron job \* \* \* \* \*

Create more users and add Global Security

Security Deploy War File on EC2 instance

1. Create a new EC2 linux instance
2. Install tomcat on the machine (yum install tomcat)
3. Launch a new linux EC2 instance
4. service start tomcat
5. Install a Deployment Plugin in Jenkins (plugin: deploy to container)

Jenkins Job Java Web Auto Deploy

1. Add Github Repo URL (Pulls project from git and deploy to tomcat)
2. Provide Username and Password for github
3. Build: mvn clean package
4. Post Build Action:
5. WAR File location
6. Add Tomcat Version
7. Username and Password
8. Tomcat URL : <http://amazoninstance:8080/>
9. Build Now

Web Hooks (Notify to the Amazon EC2 Instance where tomcat is running)

1. How to tell that a commit has happened?
2. Add a webhook in github and Jenkins will automatically trigger a job (mvn clean install package test)
3. Trigger build using webhook

Jenkins Master/Slave Architecture (If more load)

Jenkins Pipeline (commit to deployment) (JenkinsFile) (Jenkins Plugin)

Stage 1: Commit

Stage 2: Build

Stage 3: Test

Stage 4. Stage

Stage 5: Deploy Dev/QA

Stage 6: Deploy to Prod

First Jenkins Pipeline

1. Pipeline Project
2. Pipeline Script

Stages{

Step 1:

Step 2:

}

1. Build Now

Multi Stage Jenkins Pipeline

1. Checkout Stage
2. Build Stage
3. Test
4. Deploy

Define environment variables from jobs