## **2. Setting Up a Basic CI Pipeline on AWS**

**Steps required to set up a CI pipeline using Jenkins on AWS**

Setting up a CI (Continuous Integration) pipeline using **Jenkins on AWS** involves several steps:

**Step 1: Set Up an AWS EC2 Instance for Jenkins**

1. **Log in to AWS Console** and navigate to **EC2**.
2. **Launch an EC2 Instance**

**Step 2:**

* Install Jenkins on the EC2 Instance
* Install Java
* Add the Jenkins repository and install Jenkins
* Start Jenkins and enable it on boot
* Check Jenkins status
* Access Jenkins

**Step 3: Install Required Jenkins Plugins**

**Step 4: Configure AWS Credentials in Jenkins**

**Step 5: Set Up a Jenkins Job for CI/CD**

**Step 6: Set Up a Jenkins file for Pipeline**

pipeline {

agent any

stages {

stage('Checkout') {

steps {

git 'https://github.com/your-repo.git'

}

}

stage('Build') {

steps {

sh 'mvn clean install'

}

}

stage('Test') {

steps {

sh 'mvn test'

}

}

stage('Deploy') {

steps {

sh './deploy.sh'

}

}

}

}

**Step 7: Automate Deployments Using AWS Services**

* Deploy to S3 (for static sites)
* Deploy to ECS (Docker-based apps)
* Deploy to Lambda (if serverless)

**Step 8: Set Up Webhooks for Automatic Builds**

**Step 9: Monitor and Optimize**

# **C. Define pipeline stages**

**Build:**

Step 1: Install and Set Up Jenkins

Step 2: Install Required Plugins

Step 3: Configure Python Environment

Step 4: Create a New Jenkins Job

Step 5: Run the Build

Step 6: Automate with a Pipeline

pipeline {

agent any

stages {

stage('Checkout') {

steps {

git 'https://github.com/your-repo/sample-python.git'

}

}

stage('Setup') {

steps {

sh 'python3 -m venv venv && source venv/bin/activate && pip install -r requirements.txt'

}

}

stage('Run Tests') {

steps {

sh 'python -m unittest discover tests'

}

}

stage('Deploy') {

steps {

sh 'python app.py'

}

}

}

}

**Test**

1. Choose a CI/CD Tool

Select a CI/CD platform such as:

* 1. GitHub Actions
  2. GitLab CI/CD
  3. Jenkins
  4. CircleCI
  5. Travis CI

2. Define Test Steps in CI Configuration

GitLab CI (.gitlab-ci.yml)

stages:

- test

test:

stage: test

script:

- pip install -r requirements.txt

- pytest --junitxml=test-results.xml

artifacts:

reports:

junit: test-results.xml

1. Use Test Reports & Notifications
2. Automate Test Execution on Pull Requests

**Deploy**

Step 1: Launch an EC2 Instance

Step 2: Connect to EC2 Instance

Step 3: Install Dependencies

Step 4: Transfer Your Application

Step 5: Install Application Dependencies

Step 6: Configure and Run the Application  
 Step 7: Configure Security and Firewalls

Step 8: Set Up a Reverse Proxy

Step 9: Set Up Auto Start on Boot

Step 10: Access the Application