CI/CD/Deploy

✅ CI / CD / Deployment Tool Matrix

Tool CI (Continuous Integration) CD (Continuous Delivery) Deployment

Git ✅

GitHub ✅ ✅

Jenkins ✅ ✅ ✅

Docker ✅ ✅

Kubernetes ✅ ✅ ✅

**✅ 1. Git (CI)**

✔️ What it does:

Version control — tracking changes, branching, merging.

📄 Code (Git CLI commands):# Initialize a Git repo

git init

# Add your source code

git add .

# Commit your code

git commit -m "Initial commit"

# Add GitHub remote

git remote add origin https://github.com/Lokeshkhadse/jenkinsDemo.git

# Push to GitHub

git push -u origin master

**✅ 2. GitHub (CI + CD)**

✔️ What it does:

CI: Triggers build on push

CD: Hosts code + triggers Jenkins via webhook

📄 Code: GitHub Webhook to Jenkins

Go to your GitHub repo → Settings → Webhooks → Add webhook

Payload URL: <http://<your-jenkins-server>/github-webhook/>

Content type: application/json

Select Just the push event

**✅ 3. Jenkins (CI + CD + Deployment)**

✔️ What it does:

CI: Build, test

CD: Docker build & push

Deploy: To Docker/Kubernetes

📄 Jenkinsfile example:

pipeline {

agent any

tools {

maven 'Maven 3.8.1'

jdk 'Java 17'

}

environment {

DOCKERHUB\_USER = 'lokeshkhadse'

IMAGE\_NAME = 'jenkinsdemo-app'

}

stages {

stage('Checkout') {

steps {

git branch: 'master', url: '<https://github.com/Lokeshkhadse/jenkinsDemo.git>'

}

}

stage('Build') {

steps {

bat 'mvn clean compile'

}

}

stage('Test') {

steps {

bat 'mvn test'

}

}

stage('Package') {

steps {

bat 'mvn package -DskipTests'

}

}

stage('Docker Build & Push') {

steps {

script {

def tag = new Date().format("yyyyMMddHHmmss")

env.IMAGE\_TAG = "${DOCKERHUB\_USER}/${IMAGE\_NAME}:${tag}"

bat "docker build -t ${IMAGE\_TAG} ."

withCredentials([usernamePassword(credentialsId: 'dockerhub-creds', usernameVariable: 'USERNAME', passwordVariable: 'PASSWORD')]) {

bat "echo %PASSWORD% | docker login -u %USERNAME% --password-stdin"

bat "docker push ${IMAGE\_TAG}"

}

}

}

}

stage('Deploy') {

steps {

script {

// Stop and remove old container if running

bat """

docker stop jenkinsdemo || echo "No container to stop"

docker rm jenkinsdemo || echo "No container to remove"

"""

// Run the new image

bat "docker run -d -p 8080:8080 --name jenkinsdemo ${IMAGE\_TAG}"

}

}

}

}

post {

success {

echo '✅ Build, Push, and Deployment Successful!'

}

failure {

echo '❌ Something went wrong during CI/CD!'

}

}

}

**✅ 4. Docker (CD + Deployment)**

✔️ What it does:

CD: Build & push Docker image

Deployment: Run app as a container

📄 Dockerfile:

FROM eclipse-temurin:17

WORKDIR /app

COPY target/jenkinsDemo-0.0.1-SNAPSHOT.jar /app/app.jar

ENTRYPOINT ["java", "-jar", "app.jar"]

📄 Docker Commands:

docker build -t lokeshkhadse/jenkinsdemo-app:latest .

docker login -u lokeshkhadse

docker push lokeshkhadse/jenkinsdemo-app:latest

docker run -p 8080:8080 lokeshkhadse/jenkinsdemo-app:latest

**✅ 5. Kubernetes (Deployment**)

✔️ What it does:

Deploys containers to a cluster.

📄 Deployment YAML:

apiVersion: apps/v1

kind: Deployment

metadata:

name: jenkinsdemo-deployment

spec:

replicas: 2

selector:

matchLabels:

app: jenkinsdemo

template:

metadata:

labels:

app: jenkinsdemo

spec:

containers:

- name: jenkinsdemo

image: lokeshkhadse/jenkinsdemo-app:latest

ports:

- containerPort: 8080

---

apiVersion: v1

kind: Service

metadata:

name: jenkinsdemo-service

spec:

type: NodePort

selector:

app: jenkinsdemo

ports:

- port: 80

targetPort: 8080

nodePort: 30036

📄 Commands:

kubectl apply -f deployment.yaml

kubectl get pods

kubectl get services