**✅ Docker Interview Questions & Answers**

**1. What is Docker and why is it used?**

**Answer:** Docker is a platform for developing, shipping, and running applications inside lightweight containers. It provides consistent environments, faster deployments, and better scalability.

**2. What are Docker images and containers?**

* **Image**: A read-only template with instructions to create a container.
* **Container**: A running instance of an image.

**3. How do you create a Docker image?**

**Answer:**  
Using a Dockerfile:

dockerfile

CopyEdit

FROM openjdk:17

COPY ./target/app.jar /app/app.jar

CMD ["java", "-jar", "/app/app.jar"]

Build with:

bash

CopyEdit

docker build -t myapp:latest .

**4. Difference between COPY and ADD in Dockerfile?**

* COPY: Copies files/directories.
* ADD: Same as COPY, but also supports auto-extraction of tar files and URLs.

**5. How do you optimize Docker images?**

* Use multi-stage builds
* Remove unused packages
* Minimize layers
* Use .dockerignore

**6. How do volumes work in Docker?**

**Answer:** Volumes are persistent storage outside the container's filesystem. Useful for databases or shared configuration.

bash

CopyEdit

docker run -v myvolume:/data myapp

**7. How do you troubleshoot container issues?**

* Use docker logs
* Run shell inside: docker exec -it container\_id sh
* Check resource usage: docker stats

**8. How do you secure Docker containers?**

* Use minimal base images (e.g., Alpine)
* Use Docker Bench for Security
* Avoid running containers as root
* Sign images with Docker Content Trust

**✅ Kubernetes Interview Questions & Answers**

**1. What is Kubernetes and why is it needed?**

**Answer:** Kubernetes is a container orchestration tool that automates deployment, scaling, and management of containerized apps. It ensures high availability and auto-recovery.

**2. What are main Kubernetes components?**

* **Master node**:
  + kube-apiserver, etcd, kube-controller-manager, kube-scheduler
* **Worker node**:
  + kubelet, kube-proxy, container runtime (Docker/CRI-O)

**3. What is a Pod in Kubernetes?**

**Answer:** Smallest deployable unit in K8s. A pod can have one or more containers that share storage and network.

**4. How does Kubernetes handle service discovery?**

* **Services** expose Pods using internal DNS (e.g., my-service.default.svc.cluster.local)
* LoadBalancer and Ingress for external access

**5. What is the difference between Deployment and StatefulSet?**

* **Deployment**: For stateless apps, allows rolling updates.
* **StatefulSet**: For stateful apps like databases, retains persistent identity.

**6. What is a ConfigMap and a Secret?**

* **ConfigMap**: Stores config data as key-value pairs.
* **Secret**: Stores sensitive data, Base64-encoded.

**7. How do you perform rolling updates in Kubernetes?**

yaml

CopyEdit

spec:

strategy:

type: RollingUpdate

rollingUpdate:

maxUnavailable: 1

maxSurge: 1

**8. How do you monitor a Kubernetes cluster?**

* **Tools**: Prometheus, Grafana, ELK, Fluentd
* **Health Probes**: liveness, readiness, startup probes

**9. How does Kubernetes handle auto-scaling?**

* **Horizontal Pod Autoscaler (HPA)**: Scales pods based on CPU/memory.
* **Vertical Pod Autoscaler (VPA)**: Adjusts resource requests/limits.
* **Cluster Autoscaler**: Scales nodes based on workloads.

**✅ Jenkins Interview Questions & Answers**

**1. What is Jenkins and how does it fit into DevOps?**

**Answer:** Jenkins is an open-source CI/CD tool used to automate building, testing, and deploying applications.

**2. How do you configure a Jenkins pipeline?**

Using Jenkinsfile:

groovy

CopyEdit

pipeline {

agent any

stages {

stage('Build') {

steps {

sh 'mvn clean install'

}

}

stage('Test') {

steps {

sh 'mvn test'

}

}

}

}

**3. What are the types of Jenkins pipelines?**

* **Declarative pipeline** (preferred, readable syntax)
* **Scripted pipeline** (Groovy-based, flexible)

**4. What is the difference between freestyle and pipeline jobs?**

* **Freestyle**: UI-driven, limited flexibility
* **Pipeline**: Code-driven, reusable, version-controlled via Jenkinsfile

**5. How do you implement CI/CD in Jenkins?**

* Trigger builds on git push
* Build and run tests
* Create artifacts (JAR/WAR)
* Deploy using plugins (e.g., Docker, Kubernetes)

**6. What plugins are essential in Jenkins?**

* Git
* Pipeline
* Docker Pipeline
* Blue Ocean
* Kubernetes
* SonarQube Scanner

**7. How do you secure Jenkins?**

* Enable authentication (LDAP, SSO)
* Use Role-based Strategy Plugin
* Regular backups
* Audit logs
* Agent-to-master encryption

**8. How do you integrate Jenkins with Docker/Kubernetes?**

* Docker agent inside Jenkinsfile:

groovy

CopyEdit

agent {

docker {

image 'maven:3.8.5-jdk-17'

}

}

* Kubernetes plugin to run Jenkins agents in a K8s cluster.

**9. How do you manage Jenkins at scale?**

* Use **Shared Libraries**
* Use **Jenkins Operations Center** (CloudBees)
* Configure **Dockerized Jenkins with persistent volume**
* Run agents dynamically on Kubernetes

**🔁 Cross Questions (Follow-Ups)**

| **Primary** | **Cross Question** |
| --- | --- |
| Docker Image | How do you scan Docker images for vulnerabilities? |
| Kubernetes Deployment | What happens if a rolling update fails in K8s? |
| Jenkins Pipeline | How do you parallelize stages in Jenkins? |
| Jenkins + K8s | How to trigger a K8s job from Jenkins pipeline? |
| CI/CD | How do you implement Blue-Green or Canary deployments using Jenkins + K8s? |

Would you like a:

✅ **PDF version**  
✅ **GitHub-ready Jenkinsfile example with Docker + Kubernetes integration**  
✅ **Real-world project setup example**

Let me know what format you'd like or if you want system design + architecture questions related to DevOps.

**You said:**

✅ GitHub-ready Jenkinsfile example with Docker + Kubernetes integration

**ChatGPT said:**

Here’s a **GitHub-ready Jenkinsfile** demonstrating **CI/CD with Docker and Kubernetes integration** using a real-world style example. This script includes:

* Git checkout
* Docker image build and push
* Kubernetes deployment (via kubectl)
* Use of credentials for DockerHub and K8s config
* Optional parallel test stage

**✅ Jenkinsfile: Docker + Kubernetes Pipeline**

groovy

CopyEdit

pipeline {

agent any

environment {

DOCKER\_IMAGE = "yourdockerhubusername/myapp"

IMAGE\_TAG = "${BUILD\_NUMBER}"

REGISTRY\_CREDENTIALS = 'dockerhub-credentials-id'

KUBECONFIG\_CREDENTIALS = 'kubeconfig-secret-id'

}

stages {

stage('Checkout') {

steps {

git url: 'https://github.com/your-org/your-repo.git', branch: 'main'

}

}

stage('Build') {

steps {

sh 'mvn clean package -DskipTests'

}

}

stage('Test (Parallel)') {

parallel {

stage('Unit Tests') {

steps {

sh 'mvn test'

}

}

stage('Static Code Analysis') {

steps {

sh 'mvn sonar:sonar'

}

}

}

}

stage('Docker Build & Push') {

steps {

script {

docker.withRegistry('', REGISTRY\_CREDENTIALS) {

def image = docker.build("${DOCKER\_IMAGE}:${IMAGE\_TAG}")

image.push()

image.push("latest")

}

}

}

}

stage('Deploy to Kubernetes') {

steps {

withCredentials([file(credentialsId: KUBECONFIG\_CREDENTIALS, variable: 'KUBECONFIG')]) {

sh '''

export KUBECONFIG=$KUBECONFIG

kubectl set image deployment/myapp-deployment myapp-container=${DOCKER\_IMAGE}:${IMAGE\_TAG} -n mynamespace

kubectl rollout status deployment/myapp-deployment -n mynamespace

'''

}

}

}

}

post {

always {

echo 'Cleaning up...'

}

success {

echo "Deployment successful: ${DOCKER\_IMAGE}:${IMAGE\_TAG}"

}

failure {

echo 'Pipeline failed!'

}

}

}

**🔐 Jenkins Credentials Setup**

* dockerhub-credentials-id: Jenkins credentials (Username + Password) for DockerHub
* kubeconfig-secret-id: Jenkins secret file containing your kubeconfig for the target cluster

**🧪 Kubernetes Deployment YAML Sample (for reference)**

yaml

CopyEdit

apiVersion: apps/v1

kind: Deployment

metadata:

name: myapp-deployment

namespace: mynamespace

spec:

replicas: 2

selector:

matchLabels:

app: myapp

template:

metadata:

labels:

app: myapp

spec:

containers:

- name: myapp-container

image: yourdockerhubusername/myapp:latest

ports:

- containerPort: 8080

**📦 GitHub Repo Structure**

css

CopyEdit

.

├── Jenkinsfile

├── Dockerfile

├── src/

├── pom.xml

└── k8s/

└── deployment.yaml