🟡 STEP 1 — JENKINS SERVER (EC2 #1)

👉 **Install on:** Jenkins EC2 instance

☑ Install Docker

```
sudo dnf install docker -y
sudo systemctl start docker
sudo systemctl enable docker
sudo usermod -aG docker ec2-user
exit
```

Login again.

☑ Run Jenkins (Docker)

```
docker run -d \
--name jenkins \
-p 8080:8080 -p 50000:50000 \
-v jenkins_home:/var/jenkins_home \
-v /var/run/docker.sock:/var/run/docker.sock \
jenkins/jenkins:lts
```

Open:

http://JENKINS-IP:8080

Unlock:

```
docker logs jenkins
```

Install suggested plugins.

🗀 Jenkins HOME

Inside container:

```
docker exec -it jenkins bash
cd /var/jenkins_home
```

Host location:

```
docker volume inspect jenkins_home
```

Example:

```
/var/lib/docker/volumes/jenkins_home/_data
```

Workspace:

```
/var/jenkins_home/workspace/
```

☑ Install Jenkins Plugins

Manage Jenkins → Plugins:

- Git
- Maven Integration
- SonarQube Scanner
- Slack Notification

Restart Jenkins.

STEP 2 — SONARQUBE SERVER (EC2 #2)

👉 **Install on:** SonarQube EC2

☑ Install Docker

```
sudo dnf install docker -y
sudo systemctl start docker
sudo usermod -aG docker ec2-user
exit
```

☑ Run SonarQube

```
docker run -d \
--name sonarqube \
-p 9000:9000 \
sonarqube:lts
```

Open:

http://SONAR-IP:9000

Login:

`admin/admin`

Create token.

🗀 SonarQube HOME

Inside container:

```
docker exec -it sonarqube bash
cd /opt/sonarqube
```

Data folder:

`/opt/sonarqube/data`

🔵 STEP 3 — NEXUS SERVER (EC2 #3)

👉 **Install on:** Nexus EC2

☑ Install Docker

```
sudo dnf install docker -y
sudo systemctl start docker
sudo usermod -aG docker ec2-user
exit
```

☑ Run Nexus

```
docker run -d \
--name nexus \
-p 8081:8081 \
-v nexus-data:/nexus-data \
sonatype/nexus3
```

Wait 2–3 minutes.

Open:

http://NEXUS-IP:8081

Get password:

```
docker exec nexus cat /nexus-data/admin.password
```

Create repository:

```
Recipe = maven2 (hosted)
Version policy = Release
```

🗀 Nexus HOME

Inside container:

```
docker exec -it nexus bash
cd /nexus-data
```

Host path:

```
/var/lib/docker/volumes/nexus-data/_data
```

🟢 STEP 4 — MAVEN MANUAL INSTALL (Inside Jenkins Container)

👉 Maven MUST exist where Jenkins builds run.

Enter Jenkins:

```
docker exec -it jenkins bash
```

Install Maven:

```
apt update
apt install maven -y
```

Check:

```
mvn -v
```

You will see:

```
Maven home: /usr/share/maven
```

🗀 Maven HOME

```
/usr/share/maven
```

👉 This is what you enter in Jenkins MAVEN_HOME.

Exit container:

```
exit
```

🟢 STEP 5 — TOMCAT SERVER (Deploy Server)

👉 **Install on:** Jenkins EC2 (simplest)

(You can also use separate EC2 if needed.)

☑ Install Tomcat (Docker)

```
docker run -d \
--name tomcat-container \
-p 8082:8080 \
tomcat
```

Open:

http://JENKINS-IP:8082

🗀 Tomcat HOME

Inside container:

```
docker exec -it tomcat-container bash
cd /usr/local/tomcat
```

Deploy folder:

`/usr/local/tomcat/webapps/`

Logs:

`/usr/local/tomcat/logs/`

STEP 6 — CONNECT JENKINS → SONARQUBE

Manage Jenkins → Configure System

Server URL:

`http://SONAR-IP:9000`

Add token credential.

STEP 7 — CREATE FREESTYLE PIPELINE

Dashboard → New Item → Freestyle.

Git Clone

`https://github.com/betawins/hiring-app.git`

Workspace:

`/var/jenkins_home/workspace/JOB_NAME`

☑ SonarQube Scan

Add Build Step:

`Execute SonarQube Scanner`

☑ Maven Build

```
clean package
```

WAR output:

`target/*.war`

☑ Upload to Nexus

```
curl -u admin:PASS -T target/*.war http://NEXUS-IP:8081/repository/maven-releases/
```

☑ Deploy to Tomcat

```
docker cp target/*.war tomcat-container:/usr/local/tomcat/webapps/
```

☑ Slack Notification

Post-build → Slack.

EXTRA — DOCKER COMMANDS YOU ASKED

Enter container:

```
docker exec -it jenkins bash
```

Logs:

```
docker logs jenkins
```

Restart:

```
docker restart jenkins
```

Volumes:

```
docker volume inspect jenkins_home
```

HOME DIRECTORY SUMMARY

Service	Home Directory
Jenkins	`/var/jenkins_home`
SonarQube	`/opt/sonarqube`
Nexus	`/nexus-data`
Maven	`/usr/share/maven`
Tomcat	`/usr/local/tomcat`

Interview One-Liner

“I deployed Jenkins, SonarQube, Nexus and Tomcat as Docker containers on separate EC2 instances, manually installed Maven inside Jenkins, and used container home paths to build, scan, store and deploy the WAR artifact.”