

Perfect 👉 — here is your **FINAL FULL SETUP** again with EVERYTHING included:

- ✅ Multi-instance
- ✅ ec2-user
- ✅ Docker-based
- ✅ Maven manual install
- ✅ Tomcat install
- ✅ **Docker CLI inside Jenkins** (your new step)
- ✅ Home directories + where to install

Order kept exactly:

👉 Jenkins → SonarQube → Nexus → Maven → Tomcat → Pipeline

## **STEP 1 — JENKINS SERVER (EC2 #1)**

👉 Install on **Jenkins EC2**

### **Install Docker on EC2**

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

Login again.

## Run Jenkins Container

```
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:

```
/var/jenkins_home
```

Host path:

```
/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.

## **NEW STEP — Install Docker CLI inside Jenkins (IMPORTANT)**

Enter container as root:

```
docker exec -u 0 -it jenkins bash
```

Install docker command:

```
apt update
apt install docker.io -y
docker --version
```

Exit:

```
exit
```

 Now Jenkins can run:

```
docker cp
```

## **STEP 2 — SONARQUBE SERVER (EC2 #2)**

Install Docker:

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

Install Sonarqube:

```
docker run -d --name sonarqube \
  -p 9000:9000 \
  -v sonarqube_data:/opt/sonarqube/data \
  -v sonarqube_extensions:/opt/sonarqube/extensions \
  -v sonarqube_logs:/opt/sonarqube/logs \
  sonarqube:25.1.0.102122-community
```

Open:

<http://SONAR-IP:9000>

Login:

admin/admin

Create token.

## SonarQube HOME

/opt/sonarqube



## STEP 3 — NEXUS SERVER (EC2 #3)

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
```

Open:

<http://NEXUS-IP:8081>

Get password:

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

Create repo:

```
maven2 (hosted)  
Release
```

## **Nexus HOME**

/nexus-data

Host:

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

## **STEP 4 — MAVEN MANUAL INSTALL (Inside Jenkins)**

Enter:

```
docker exec -u 0 -it jenkins bash
```

Install:

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

Check:

```
mvn -v
```

Maven HOME:

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

Exit.

## **STEP 5 — TOMCAT SERVER (Install on Jenkins EC2)**

Run:

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

Open:

<http://JENKINS-IP:8082>

## **Tomcat HOME**

/usr/local/tomcat

Deploy folder:

/usr/local/tomcat/webapps/

## **STEP 6 — CONNECT JENKINS → SONAR**

Manage Jenkins → Configure System:

<http://SONAR-IP:9000>

Add token.

## **STEP 7 — FREESTYLE PIPELINE**

Create Freestyle Job.

### **Git Clone**

<https://github.com/betawins/hiring-app.git>

Workspace:

/var/jenkins\_home/workspace/Project-01

## ✓ SonarQube Scan

Add:

Execute SonarQube Scanner

## ✓ Maven Build

clean install

WAR output:

target/hiring.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/

(No restart needed)

Open:

<http://JENKINS-IP:8082/hiring>



## ✓ Slack Notification

Post-build → Slack.



## FINAL HOME DIRECTORY SUMMARY

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



## Interview One-Liner

**“I built a multi-instance Docker CI/CD setup where Jenkins with Docker CLI builds using Maven, scans via SonarQube, uploads artifacts to Nexus and deploys WAR files to a Tomcat container.”**