🚀 SCRIPTED PIPELINE — COMPLETE STEP-BY-STEP DOCUMENT

Environment reminder (your setup):

```
Jenkins runs inside Docker
Tomcat runs inside Docker
SonarQube on EC2 #2
Nexus3 on EC2 #3
Workspace: /var/jenkins_home/workspace/
Branch: feature-1.1
Project: Maven WAR (SimpleCustomerApp)
```

☑ STEP 1 — Git Clone ONLY

Script

```
node {

    stage('Git Clone') {
        git branch: 'feature-1.1',
        url: 'https://github.com/betawins/sabear_simplecutomerapp.git'
    }

}
```

What it does

Downloads project into Jenkins container workspace.

✅ STEP 2 — Add SonarQube Integration (mvn sonar:sonar)

```
node {

    stage('Git Clone') {
        git branch: 'feature-1.1',
        url: 'https://github.com/betawins/sabear_simplecutomerapp.git'
    }

    stage('SonarQube Integration') {
        withSonarQubeEnv('SonarQube') {
            sh 'mvn sonar:sonar'
        }
    }

}
```

Why we use mvn sonar

```
Project is Maven-based.
Maven sends build context → Sonar does full analysis.
```

☑ STEP 3 — Add Maven Compilation

Script

```
node {

    stage('Git Clone') {
        git branch: 'feature-1.1',
        url: 'https://github.com/betawins/sabear_simplecutomerapp.git'
    }

    stage('SonarQube Integration') {
        withSonarQubeEnv('SonarQube') {
            sh 'mvn sonar:sonar'
        }
    }

    stage('Maven Compilation') {
```

```
        sh 'mvn clean install -DskipTests'
    }

}
```

Result

Creates:

`target/SimpleCustomerApp-<BUILD_NUMBER>-SNAPSHOT.war`

☑ STEP 4 — Add Nexus Artifactory Upload

```
node {

    stage('Git Clone') {
        git branch: 'feature-1.1',
        url: 'https://github.com/betawins/sabear_simplecutomerapp.git'
    }

    stage('SonarQube Integration') {
        withSonarQubeEnv('SonarQube') {
            sh 'mvn sonar:sonar'
        }
    }

    stage('Maven Compilation') {
        sh 'mvn clean install -DskipTests'
    }

    stage('Nexus Artifactory') {
        nexusArtifactUploader(
            nexusVersion: 'nexus3',
            protocol: 'http',
            nexusUrl: 'YOUR_NEXUS_PRIVATE_IP:8081',
            groupId: 'com.javatpoint',
            version: "${env.BUILD_NUMBER}-SNAPSHOT",
            repository: 'Project-02',
            credentialsId: 'Nexus',
```

```
            artifacts: [
                [artifactId: 'SimpleCustomerApp',
                 classifier: '',
                 file: "target/SimpleCustomerApp-${env.BUILD_NUMBER}-
SNAPSHOT.war",
                 type: 'war']
            ]
        )
    }

}
```

Important fixes included

✓ credentialsId must be Nexus
✓ Use private IP
✓ Match pom.xml coordinates

☑ STEP 5 — Add Deploy On Tomcat

Script

```
node {

    stage('Git Clone') {
        git branch: 'feature-1.1',
        url: 'https://github.com/betawins/sabear_simplecutomerapp.git'
    }

    stage('SonarQube Integration') {
        withSonarQubeEnv('SonarQube') {
            sh 'mvn sonar:sonar'
        }
    }

    stage('Maven Compilation') {
        sh 'mvn clean install -DskipTests'
```

```
    }

    stage('Nexus Artifactory') {
        nexusArtifactUploader(
            nexusVersion: 'nexus3',
            protocol: 'http',
            nexusUrl: 'YOUR_NEXUS_PRIVATE_IP:8081',
            groupId: 'com.javatpoint',
            version: "${env.BUILD_NUMBER}-SNAPSHOT",
            repository: 'Project-02',
            credentialsId: 'Nexus',
            artifacts: [
                [artifactId: 'SimpleCustomerApp',
                 classifier: '',
                 file: "target/SimpleCustomerApp-${env.BUILD_NUMBER}-
SNAPSHOT.war",
                 type: 'war']
            ]
        )
    }

    stage('Deploy On Tomcat') {
        sh """
        docker cp target/SimpleCustomerApp-${env.BUILD_NUMBER}-SNAPSHOT.war
tomcat-container:/usr/local/tomcat/webapps/
        """
    }

}
```

Important fix we applied

❌ Don't use:

```
target/*.war
```

✓ Use exact WAR name to avoid duplicate deployment.

☑ STEP 6 — FINAL VERSION (Add Slack Notification)

FINAL COMPLETE SCRIPTED PIPELINE

```
node {

    try {

        stage('Git Clone') {
            git branch: 'feature-1.1',
            url: 'https://github.com/betawins/sabear_simplecutomerapp.git'
        }

        stage('SonarQube Integration') {
            withSonarQubeEnv('SonarQube') {
                sh 'mvn sonar:sonar'
            }
        }

        stage('Maven Compilation') {
            sh 'mvn clean install -DskipTests'
        }

        stage('Nexus Artifactory') {
            nexusArtifactUploader(
                nexusVersion: 'nexus3',
                protocol: 'http',
                nexusUrl: 'YOUR_NEXUS_PRIVATE_IP:8081',
                groupId: 'com.javatpoint',
                version: "${env.BUILD_NUMBER}-SNAPSHOT",
                repository: 'Project-02',
                credentialsId: 'Nexus',
                artifacts: [
                    [artifactId: 'SimpleCustomerApp',
                     classifier: '',
                     file: "target/SimpleCustomerApp-${env.BUILD_NUMBER}-
SNAPSHOT.war",
                     type: 'war']
                ]
            )
```

```
        }

        stage('Deploy On Tomcat') {
            sh """
            docker cp target/SimpleCustomerApp-${env.BUILD_NUMBER}-SNAPSHOT.war
tomcat-container:/usr/local/tomcat/webapps/
            """
        }

        slackSend channel: '#jenkins',
        message: "Build SUCCESS ${env.JOB_NAME} #${env.BUILD_NUMBER}"

    } catch (err) {

        slackSend channel: '#jenkins',
        message: "Build FAILED ${env.JOB_NAME} #${env.BUILD_NUMBER}"

        throw err
    }
}
```

🔑 QUICK PROOF CHECKS (No More Guessing)

After pipeline runs:

Check WAR built

```
docker exec -it jenkins-container ls /var/jenkins_home/workspace/<job>/target/
```

Check Tomcat deployment

```
docker exec -it tomcat-container ls /usr/local/tomcat/webapps/
```

Correct URL

http://IP:8082/SimpleCustomerApp-XX-SNAPSHOT

(Versioned context path is normal.)

🎯 FINAL FLOW YOU BUILT

```
GitHub
   ↓
Scripted Jenkins Pipeline
   ↓
mvn sonar:sonar → SonarQube
   ↓
mvn clean install → WAR
   ↓
Upload → Nexus3
   ↓
docker cp → Tomcat container
   ↓
Slack notification
```

👉 **Interview one-liner:**

“A scripted Jenkins pipeline uses Groovy-based stages inside a node block to clone code, run Maven Sonar analysis, build a WAR, publish to Nexus, deploy to Dockerized Tomcat, and send Slack alerts.”