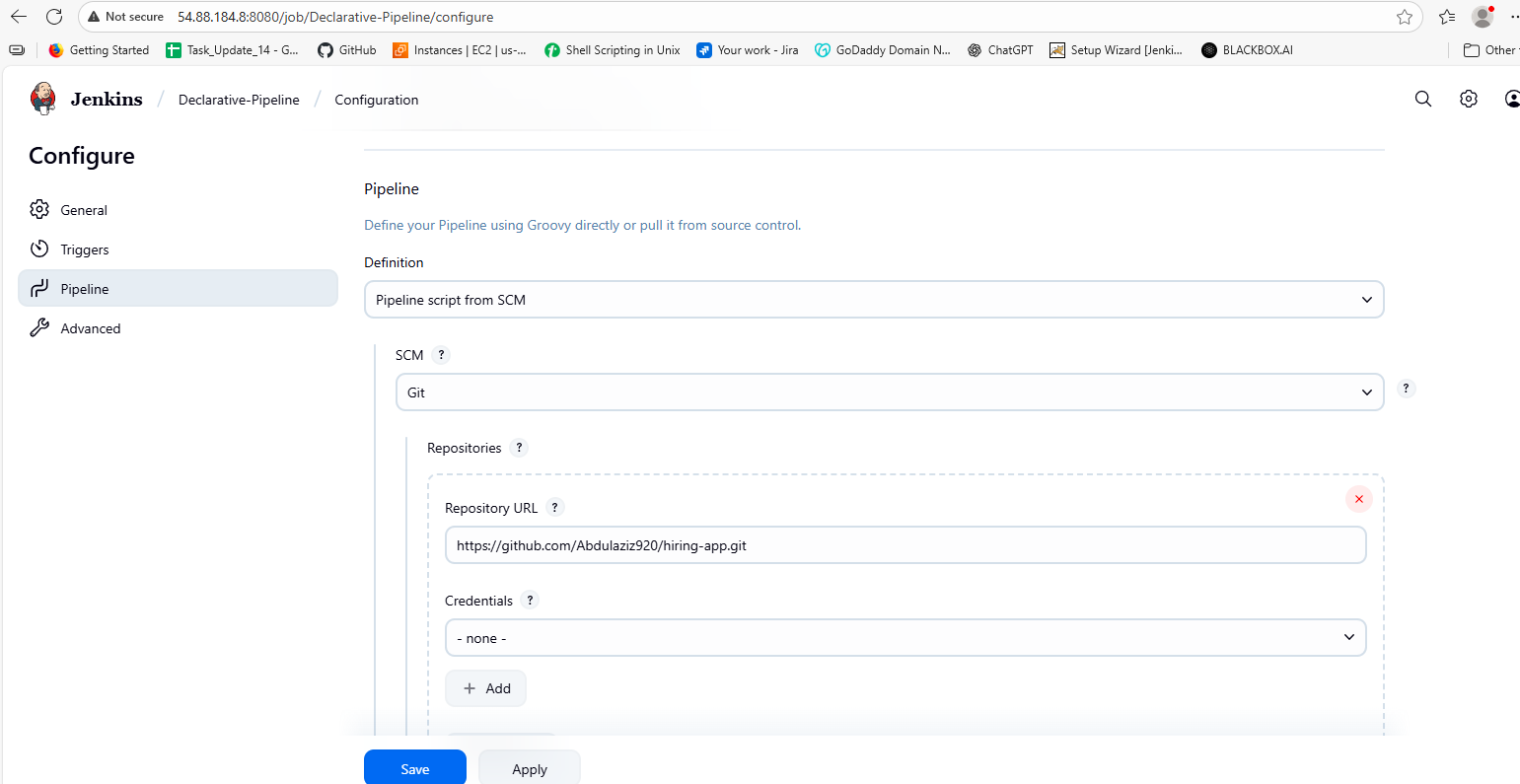
**Challenge-04**

**Tasks related to Jenkins pipelines:**

1. **Create one Declarative pipeline job**

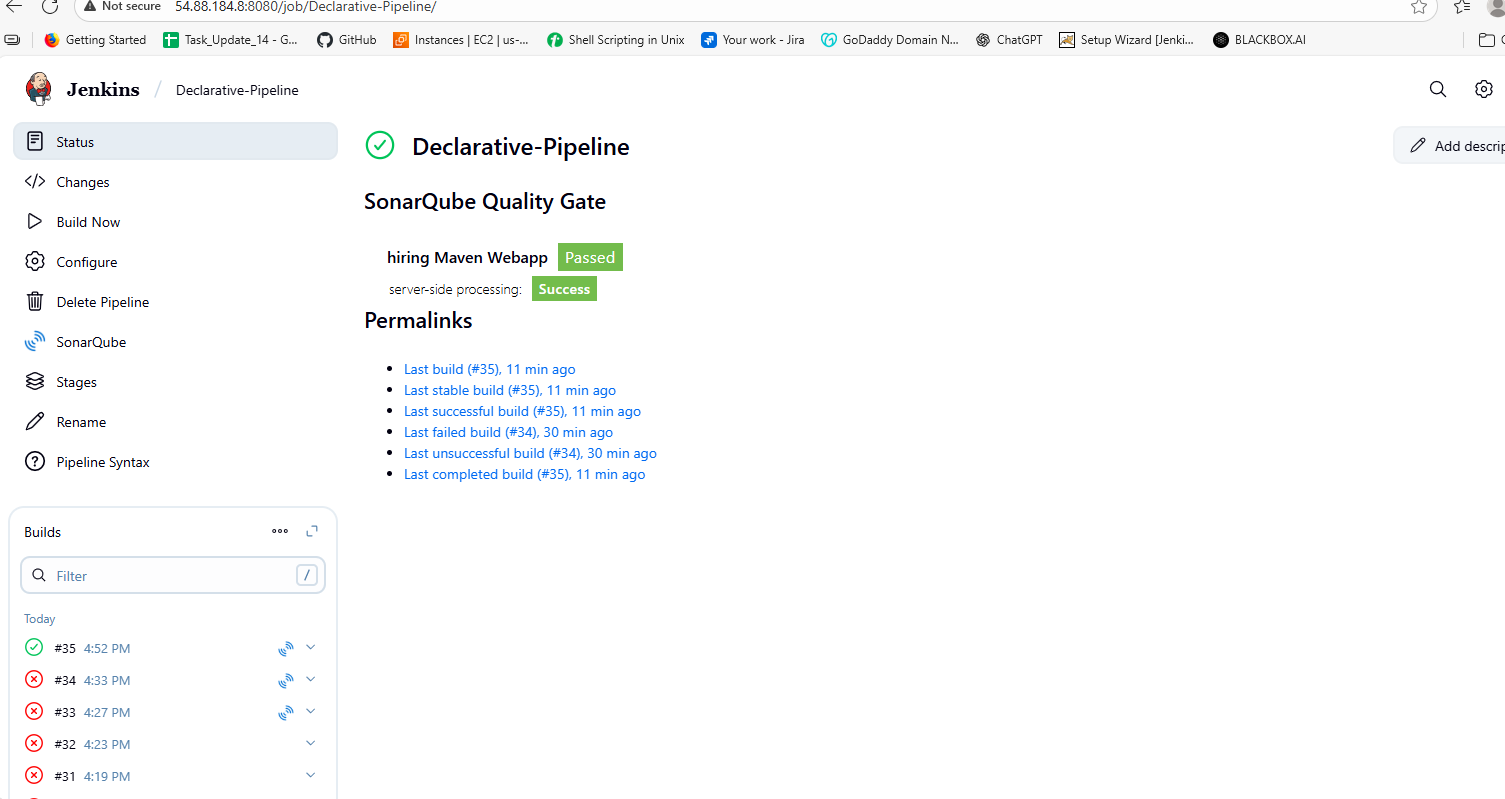
First we have to fork the given repo because in that we cant change anything that has some credentials like ready only access that’s why first we have to fork the repo then edit all stages as in the vm and Jenkins then .

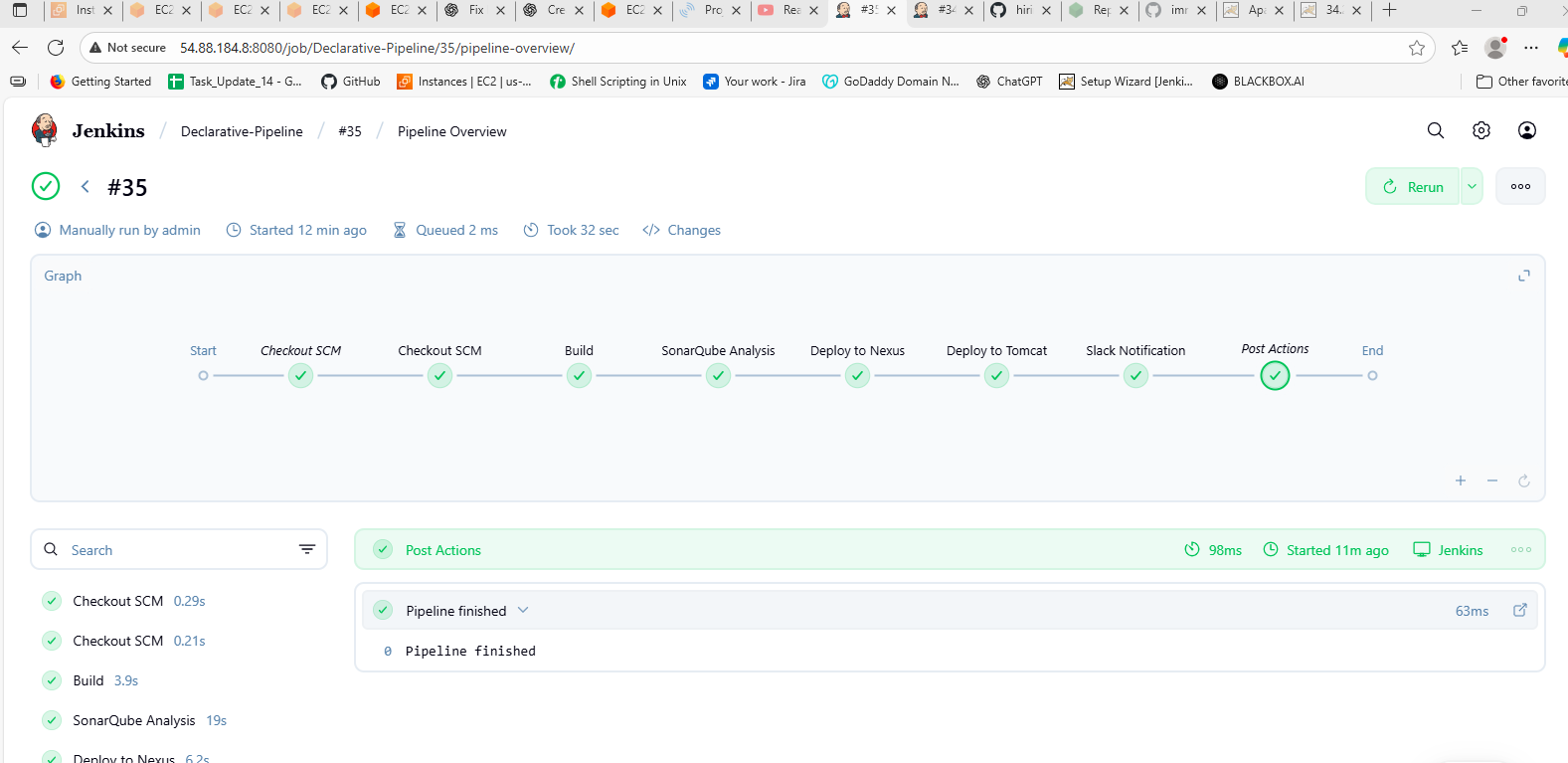
First I have created the job with selecting the pipeline and see and below picture.

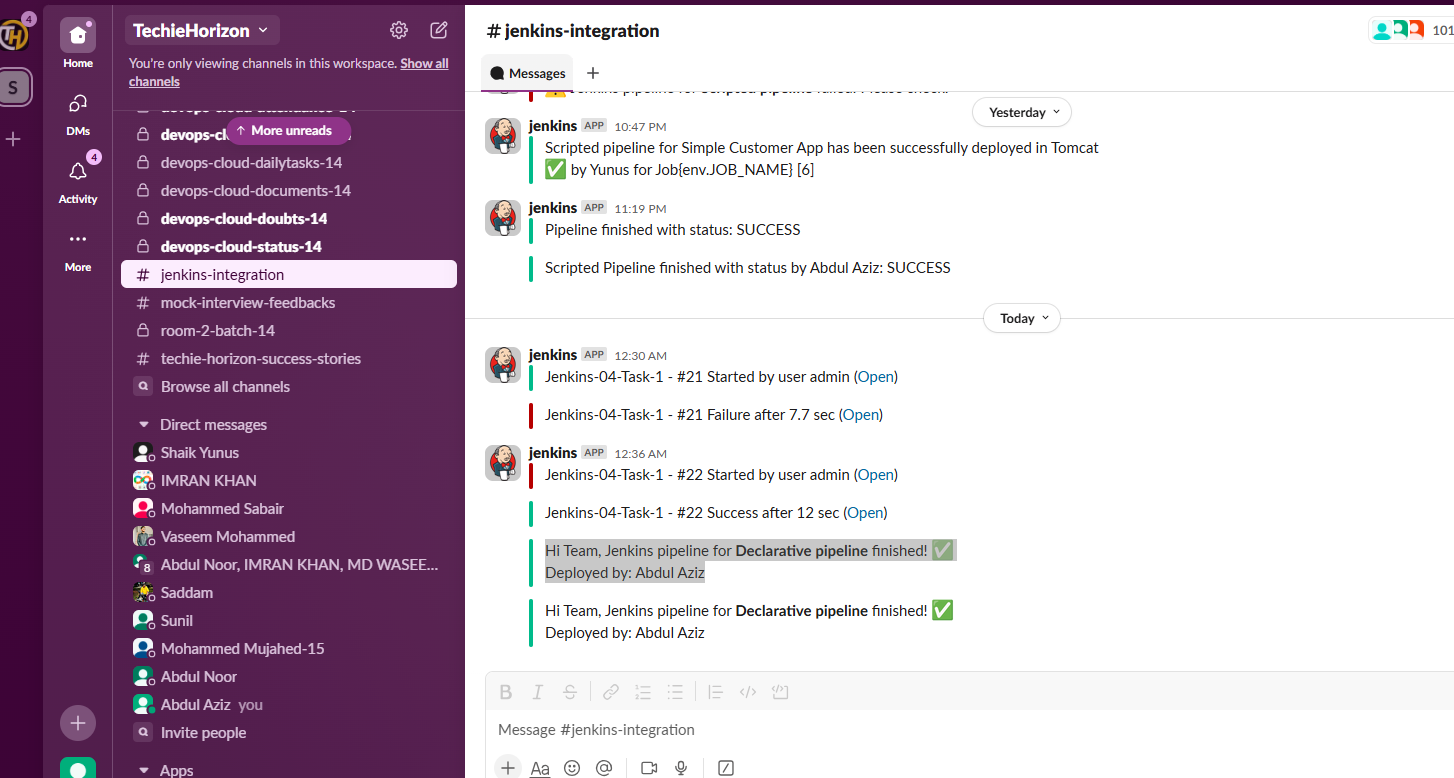


In pipeline you will see 2 options select the above one as SCM

Then paste the git url and branch name and click save.





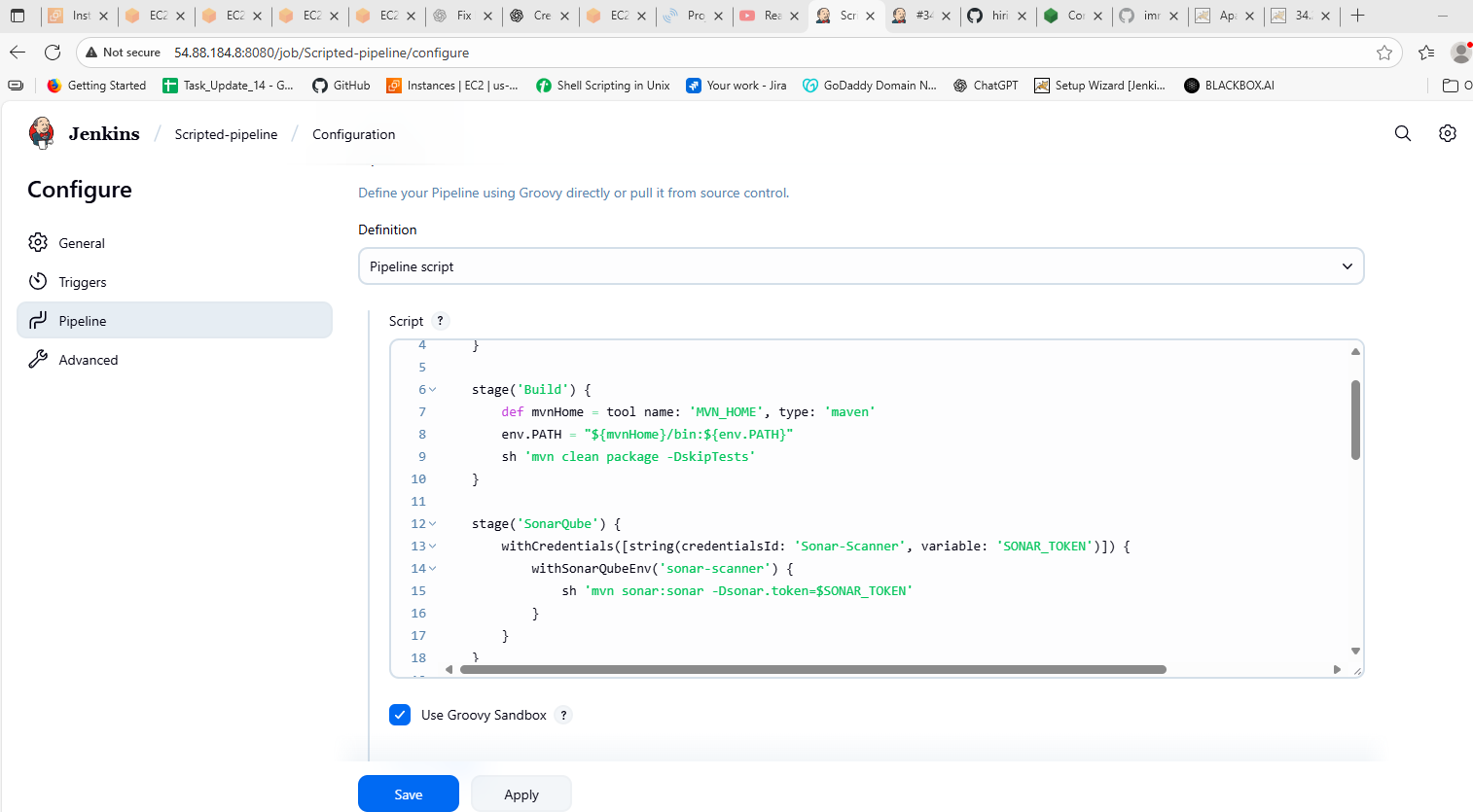


Successfully done with 1st one.

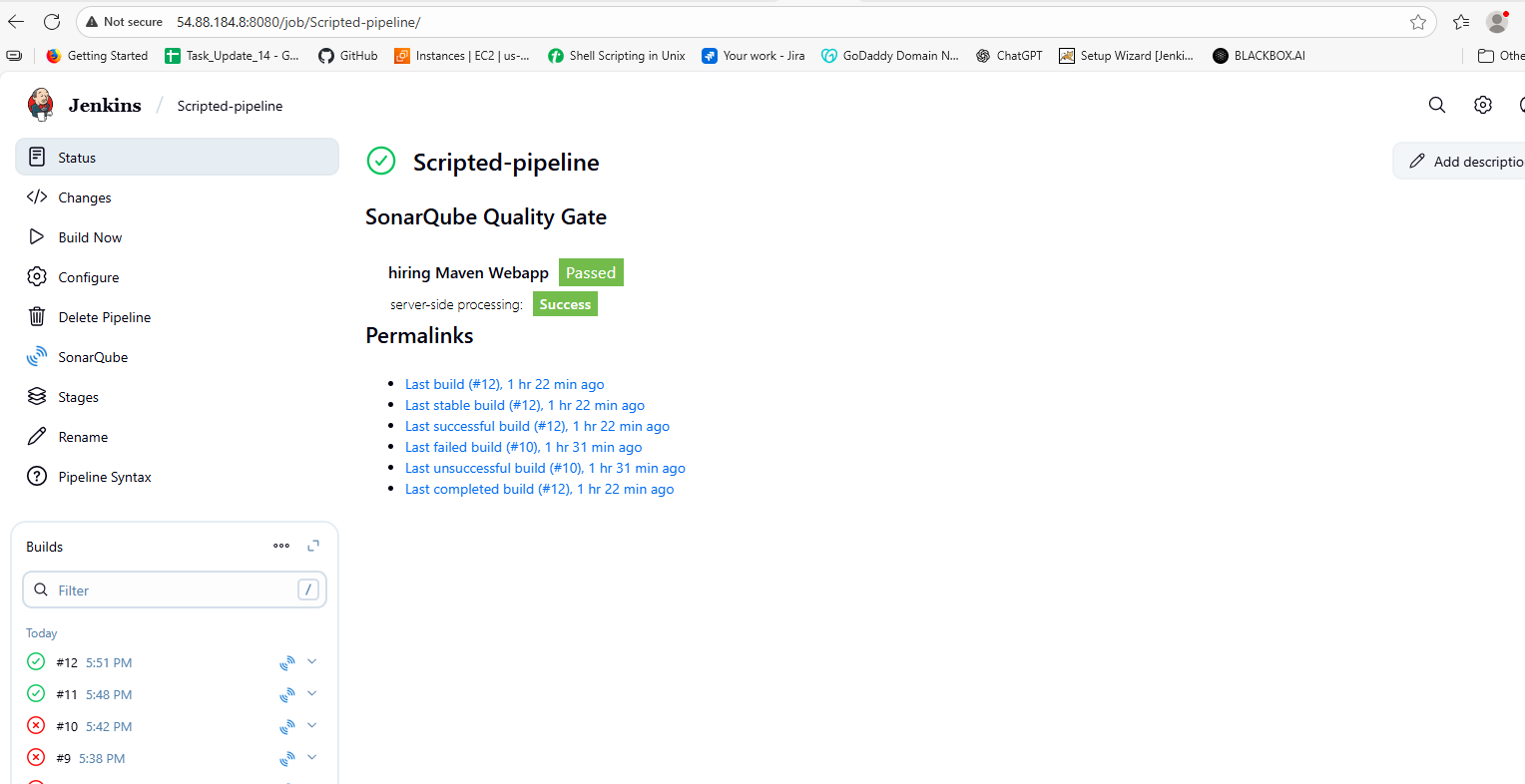
**2) Create one Scripted pipeline job**

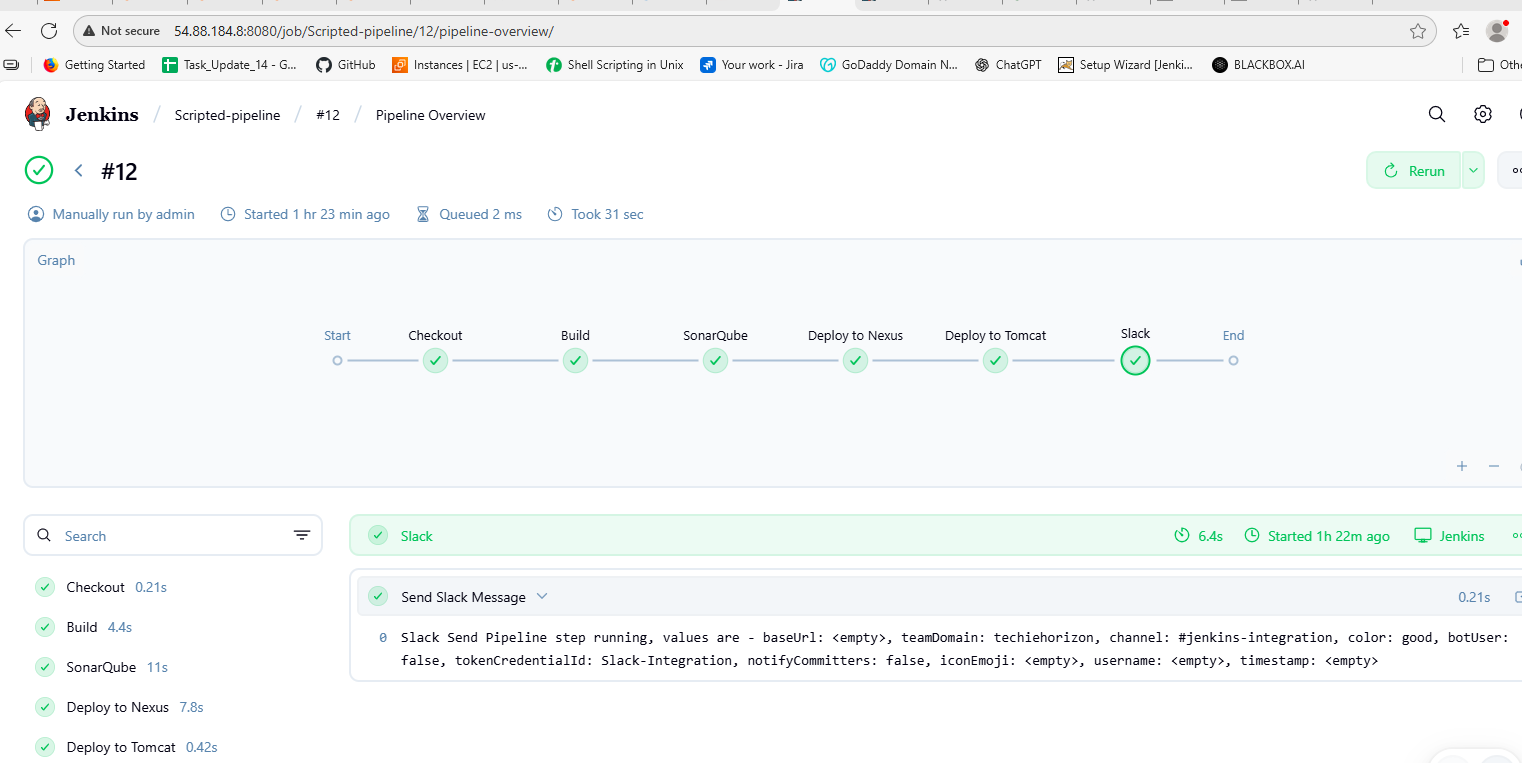
First we have to make the job as scripted name and then select the pieline then Ok

Go to configure and select the scripted pipline and copy the jenkinsfile from our forked repo and then save it ..

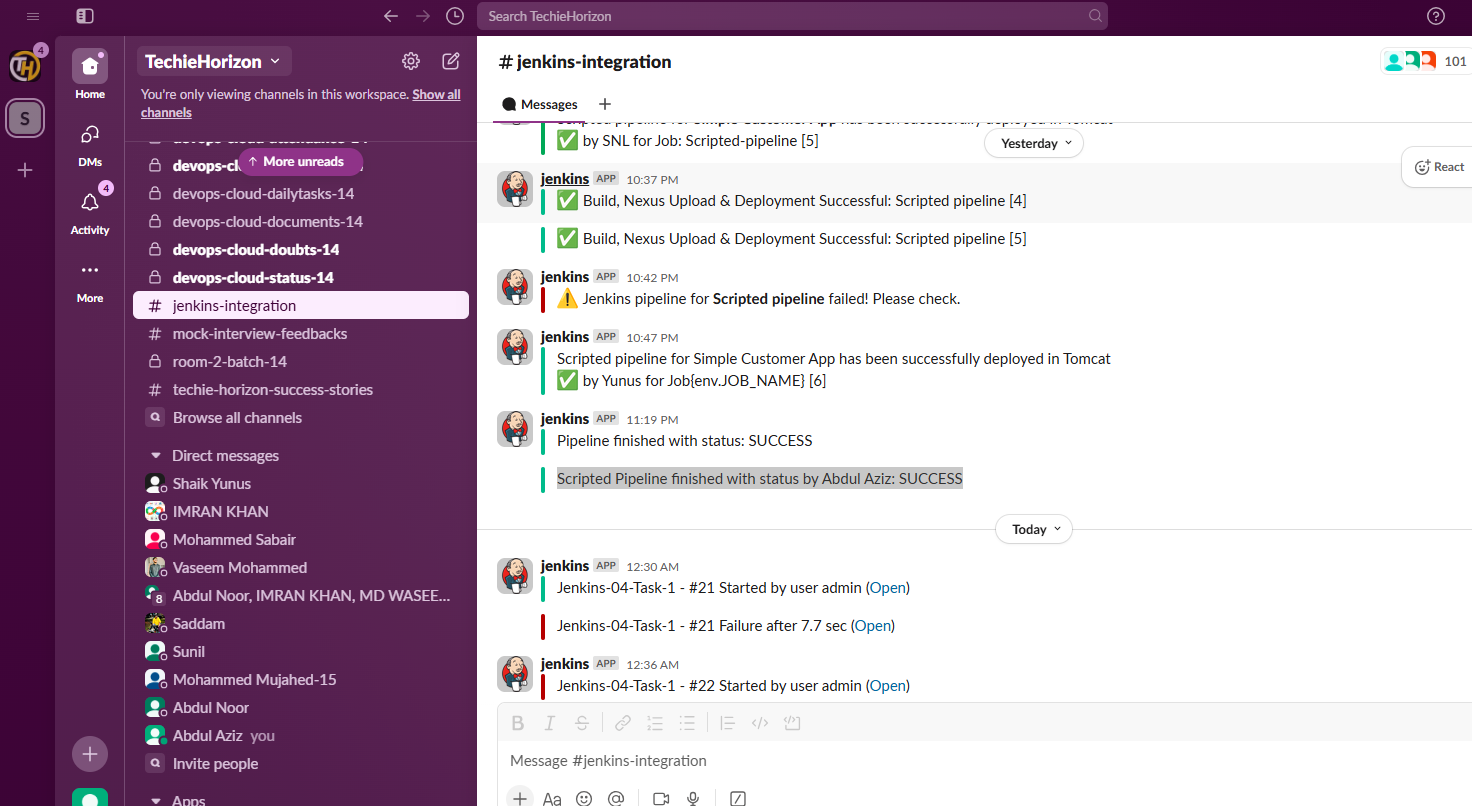


Click on save and build the job..





After this if the job is successfully build then we get the notification on the slack as well because in the script we are adding that too.As given below..



1. **Create one multi stage pipeline job**

**Step 1: Log in to Jenkins**

1. Open your Jenkins URL in a browser.
2. Log in with admin credentials.

**Step 2: Create a new pipeline job**

1. Click **“New Item”** in the Jenkins dashboard.
2. Enter a name for your job (e.g., Multistage-Pipeline).
3. Select **“Pipeline”**.
4. Click **“OK”**.

**Step 3: Configure the pipeline**

1. Scroll to the **Pipeline** section.
2. Select **Definition: Pipeline script**.
3. Paste your **multistage Jenkinsfile** into the script area.below script :

pipeline {

agent any

tools {

maven "MVN\_HOME"

}

environment {

NEXUS\_URL = "52.91.90.35:8081"

NEXUS\_REPOSITORY = "Abdul"

NEXUS\_CREDENTIAL\_ID = "My-Nexus"

TOMCAT\_USER = "deployer" // Tomcat manager username

TOMCAT\_PASSWORD = "deployer" // Tomcat manager password

TOMCAT\_HOST = "184.72.143.25" // Tomcat EC2 public IP

TOMCAT\_PORT = "8080"

SLACK\_CHANNEL = "#jenkins-integration"

SLACK\_CREDENTIAL\_ID = "slack\_notification"

}

stages {

stage("Checkout") {

steps {

git 'https://github.com/Abdulaziz920/spring3-mvc-maven-xml-hello-world-1.git'

}

}

stage("Build & Test") {

parallel {

stage("Maven Clean & Compile") {

steps {

sh 'mvn clean compile'

}

}

stage("Unit Tests") {

steps {

sh 'mvn test'

}

}

}

}

stage("Package") {

steps {

sh 'mvn -B package -DskipTests'

}

}

stage("Publish to Nexus") {

steps {

script {

def pom = readMavenPom file: 'pom.xml'

def artifactVersion = pom.version

def groupId = pom.groupId

def artifactId = pom.artifactId

def warFiles = findFiles(glob: "target/${artifactId}-${artifactVersion}.war")

if (warFiles.length == 0) {

error "WAR file not found: target/${artifactId}-${artifactVersion}.war"

}

def warFile = warFiles[0].path

echo "Uploading artifact: ${warFile} (version: ${artifactVersion}) to Nexus"

nexusArtifactUploader(

artifacts: [[

artifactId: artifactId,

classifier: '',

file: warFile,

type: 'war'

], [

artifactId: artifactId,

classifier: '',

file: 'pom.xml',

type: 'pom'

]],

credentialsId: NEXUS\_CREDENTIAL\_ID,

groupId: groupId,

version: artifactVersion,

repository: NEXUS\_REPOSITORY

)

}

}

}

stage("Deploy to Tomcat") {

steps {

script {

def pom = readMavenPom file: 'pom.xml'

def artifactVersion = pom.version

def artifactId = pom.artifactId

def warFile = "target/${artifactId}-${artifactVersion}.war"

echo "Deploying ${warFile} to Tomcat at ${TOMCAT\_HOST}:${TOMCAT\_PORT}"

sh """

curl -u ${TOMCAT\_USER}:${TOMCAT\_PASSWORD} \

-T ${warFile} \

"http://${TOMCAT\_HOST}:${TOMCAT\_PORT}/manager/text/deploy?path=/${artifactId}&update=true"

"""

}

}

}

}

post {

success {

slackSend(

channel: SLACK\_CHANNEL,

color: 'good',

message: ":white\_check\_mark: ✅ Pipeline '${env.JOB\_NAME} [${env.BUILD\_NUMBER}]' completed successfully! <${env.BUILD\_URL}|Open Build>"

)

}

failure {

slackSend(

channel: SLACK\_CHANNEL,

color: 'danger',

message: ":x: ❌ Pipeline '${env.JOB\_NAME} [${env.BUILD\_NUMBER}]' failed! <${env.BUILD\_URL}|Open Build>"

)

}

always {

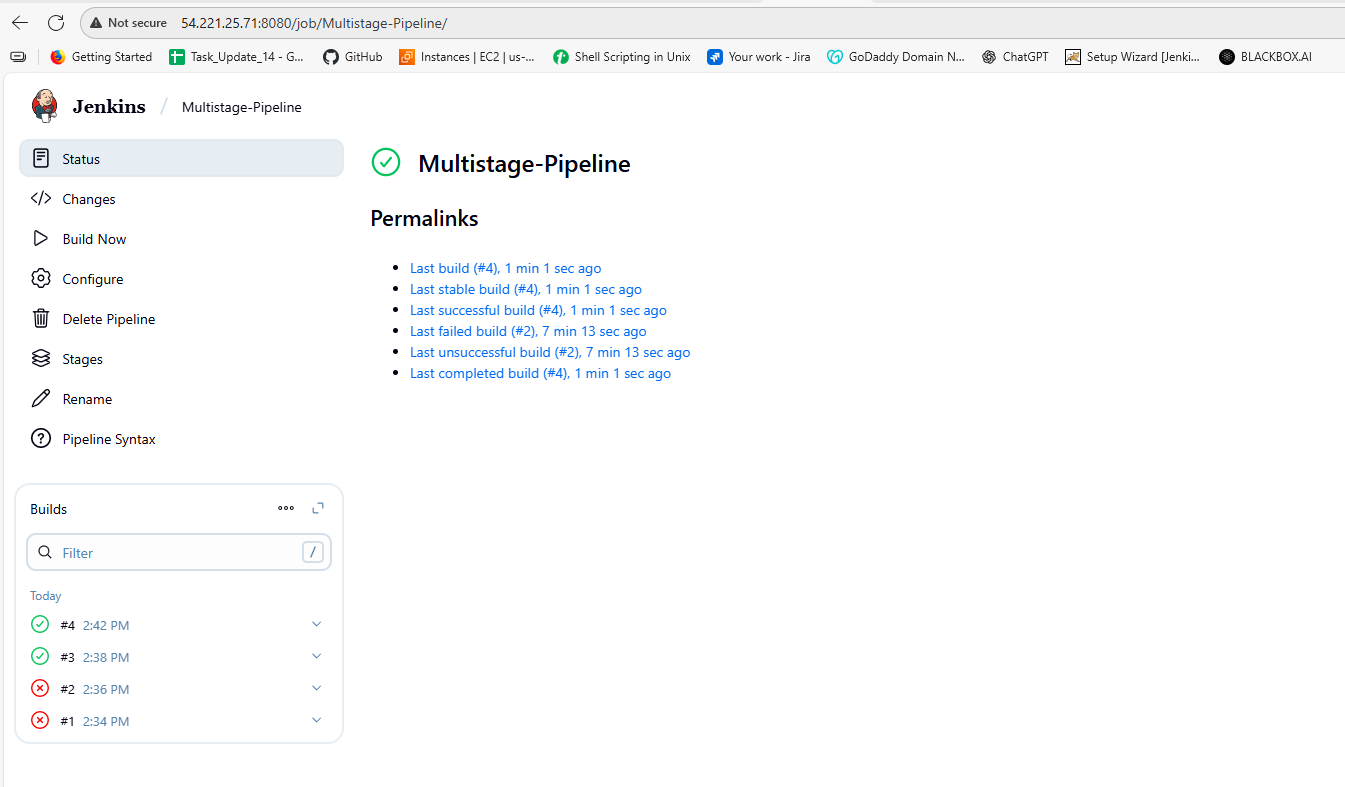
echo "Cleaning workspace..."

cleanWs()

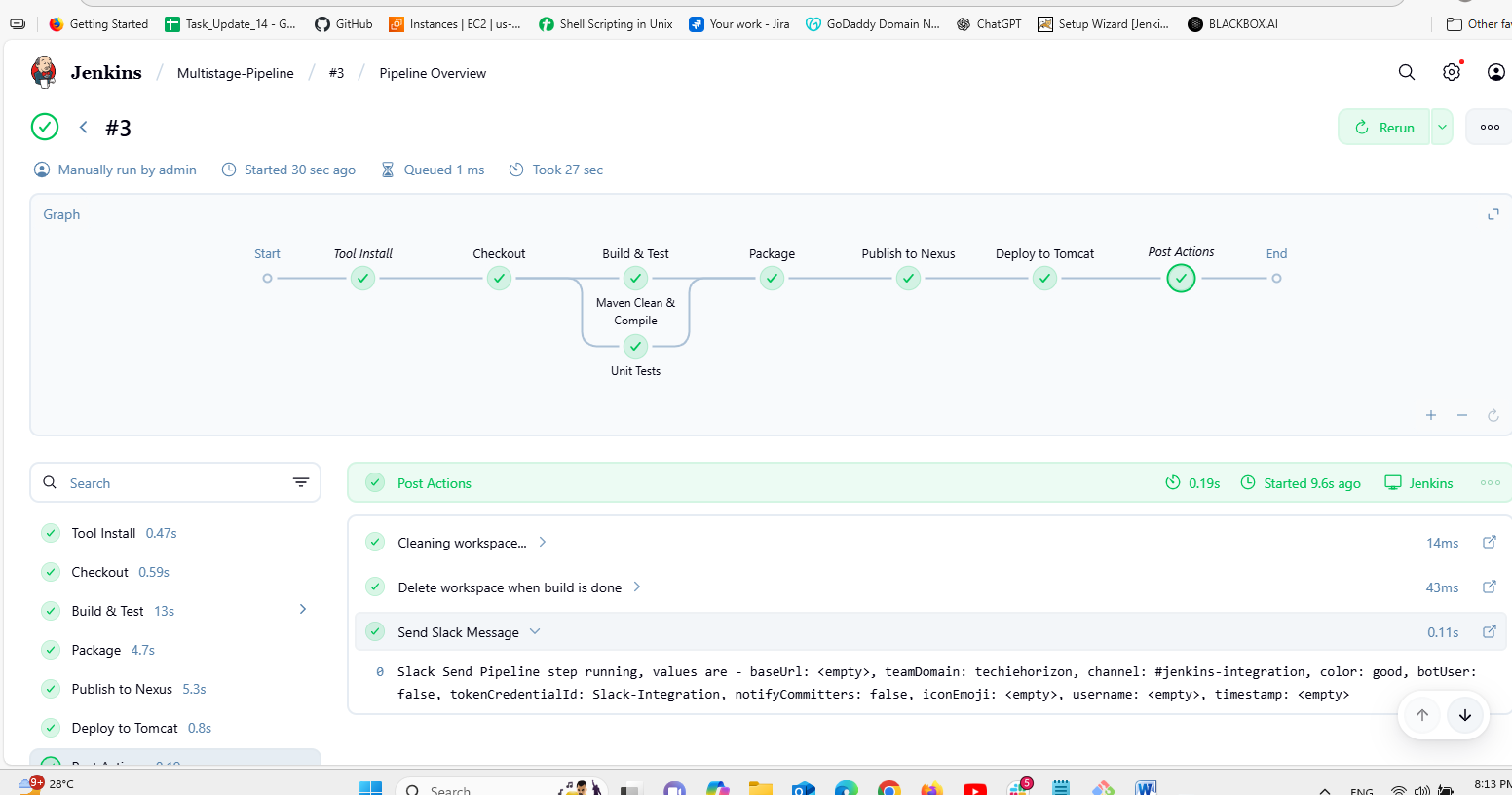
}

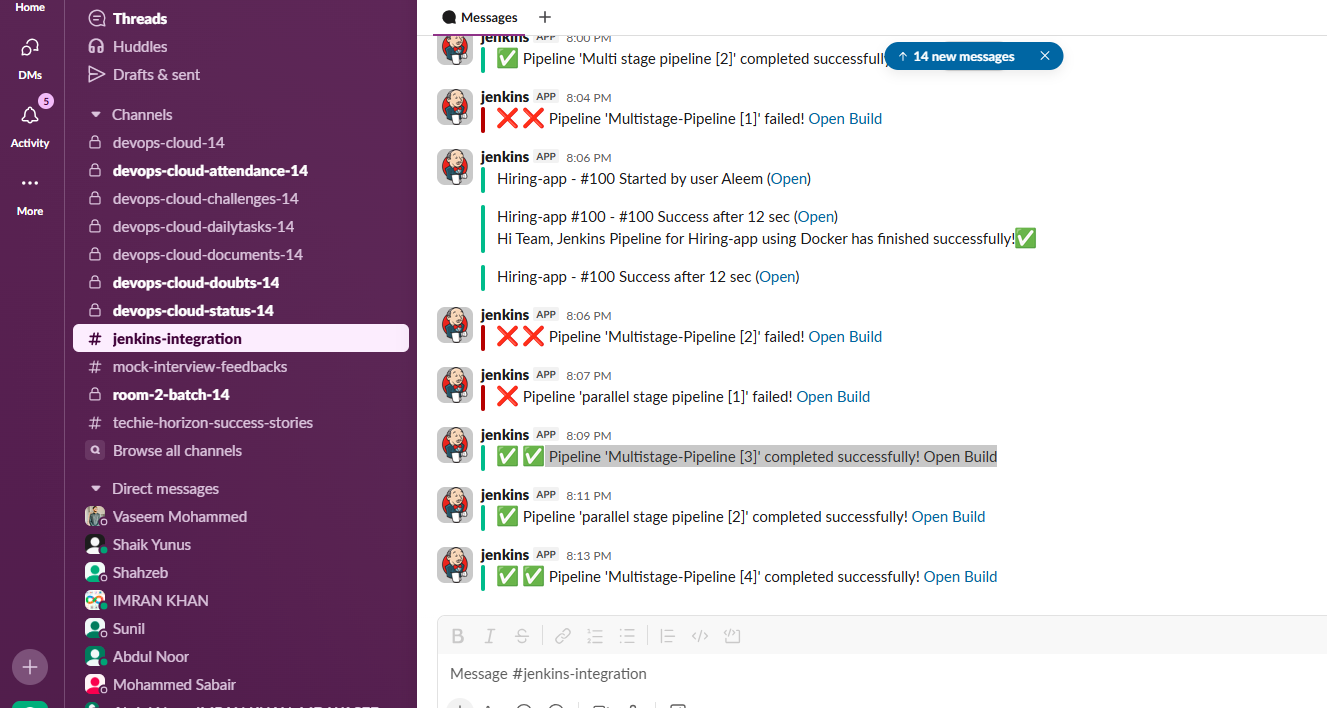
}

}



In configuration we just need to add the script . Then





* 1. **Create one parallel stage pipeline job**

**Step 1: Log in to Jenkins**

1. Open your Jenkins URL in a browser.
2. Log in with your admin credentials.

**Step 2: Create a new pipeline job**

1. Click **“New Item”** on the Jenkins dashboard.
2. Enter a name for your job (e.g., Parallel-Pipeline).
3. Select **“Pipeline”**.
4. Click **“OK”**.

**Step 3: Configure the pipeline script**

1. Scroll to the **Pipeline** section.
2. Choose **Definition: Pipeline script**.
3. Paste a multistage pipeline with **parallel stages**, e.g.:

pipeline {

agent any

tools {

maven "MVN\_HOME"

}

environment {

NEXUS\_URL = "52.91.90.35:8081"

NEXUS\_REPOSITORY = "Abdul"

NEXUS\_CREDENTIAL\_ID = "My-Nexus"

TOMCAT\_USER = "deployer"

TOMCAT\_PASSWORD = "deployer"

TOMCAT\_HOST = "184.72.143.25"

TOMCAT\_PORT = "8080"

SLACK\_CHANNEL = "#jenkins-integration"

SLACK\_CREDENTIAL\_ID = "slack\_notification"

}

stages {

stage("Clone code") {

steps {

git 'https://github.com/Abdulaziz920/spring3-mvc-maven-xml-hello-world-1.git'

}

}

stage("Build & Test") {

parallel {

stage("Maven Build") {

steps {

sh 'mvn clean compile'

}

}

stage("Unit Tests") {

steps {

sh 'mvn test'

}

}

}

}

stage("Package") {

steps {

sh 'mvn -B -Dmaven.test.failure.ignore=true package'

}

}

stage("Publish to Nexus") {

steps {

script {

def pom = readMavenPom file: 'pom.xml'

def artifactVersion = pom.version

def groupId = pom.groupId

def artifactId = pom.artifactId

def warFiles = findFiles(glob: "target/${artifactId}-${artifactVersion}.war")

if (warFiles.length == 0) {

error "WAR file not found: target/${artifactId}-${artifactVersion}.war"

}

def warFile = warFiles[0].path

echo "Uploading artifact: ${warFile} (version: ${artifactVersion}) to Nexus"

nexusArtifactUploader(

artifacts: [[

artifactId: artifactId,

classifier: '',

file: warFile,

type: 'war'

], [

artifactId: artifactId,

classifier: '',

file: 'pom.xml',

type: 'pom'

]],

credentialsId: NEXUS\_CREDENTIAL\_ID,

groupId: groupId,

version: artifactVersion,

repository: NEXUS\_REPOSITORY

)

}

}

}

stage("Deploy to Tomcat") {

steps {

script {

def pom = readMavenPom file: 'pom.xml'

def artifactVersion = pom.version

def artifactId = pom.artifactId

def warFile = "target/${artifactId}-${artifactVersion}.war"

echo "Deploying ${warFile} to Tomcat at ${TOMCAT\_HOST}:${TOMCAT\_PORT}"

sh """

curl -u ${TOMCAT\_USER}:${TOMCAT\_PASSWORD} \

-T ${warFile} \

"http://${TOMCAT\_HOST}:${TOMCAT\_PORT}/manager/text/deploy?path=/${artifactId}&update=true"

"""

}

}

}

}

post {

success {

slackSend(

channel: SLACK\_CHANNEL,

color: 'good',

message: ":white\_check\_mark: Pipeline '${env.JOB\_NAME} [${env.BUILD\_NUMBER}]' completed successfully! By Abdul Aziz <${env.BUILD\_URL}|Open Build>"

)

cleanWs()

}

failure {

slackSend(

channel: SLACK\_CHANNEL,

color: 'danger',

message: ":x: Pipeline '${env.JOB\_NAME} [${env.BUILD\_NUMBER}]' failed! <${env.BUILD\_URL}|Open Build>"

)

cleanWs()

}

always {

echo "Cleaning workspace..."

cleanWs()

}

}

}

**Step 4: Configure tools**

1. Go to **Manage Jenkins → Global Tool Configuration**.
2. Add **Maven** (name it MVN\_HOME) if referenced in your pipeline.
3. Ensure **Java** is installed on the node and available in PATH.

**Step 5: Configure credentials (if needed)**

1. Go to **Manage Jenkins → Credentials**.
2. Add:
   * Git credentials (if private repo)
   * Nexus credentials
   * Slack token
   * Tomcat manager credentials
3. Make sure the **IDs match** what you used in environment {} in your pipeline.

**Step 6: Save the job**

1. Click **Save** at the bottom of the job configuration page.

**Step 7: Run the pipeline**

1. Click **Build Now** on the job page.
2. Watch the **parallel stages** run concurrently in the console output or Blue Ocean view.

