**JENKINS PIPELINE CHALLENGE**

**1) Create one Declarative pipeline job**

**2) Create one Scripted pipeline job**

**3) Create one multi stage pipeline job**

**4) Create one parallel stage pipeline job**

**JENKINS PIPELINE CHALLENGE**

**1) Create one Declarative pipeline job**

## **1️⃣ Jenkins EC2 Configuration**

**EC2 Instance: Your Jenkins server**

**Actions & Configurations:**

1. **Installed Jenkins**
   * **Installed on EC2 (probably Amazon Linux or Ubuntu).**
   * **Installed Java (required for Jenkins & Maven).**
2. **Installed Tools in Jenkins**
   * **Maven (version 3.8.4) → configured in Manage Jenkins → Global Tool Configuration.**
   * **JDK (Java 17) → configured for builds.**
3. **Installed Plugins**
   * **Git Plugin → to clone repositories from GitHub.**
   * **Pipeline Plugin → to create declarative pipelines.**
   * **Maven Integration Plugin → to run Maven builds.**
   * **SonarQube Scanner Plugin → to run SonarQube analysis from Jenkins.**
   * **Slack Notification Plugin → to send Slack messages from Jenkins.**
   * **Credentials Plugin → to store passwords securely.**
   * **Nexus artifact loader**
   * **Stage view plugin**
4. **Created Credentials**
   * **Nexus Credentials → username/password stored in Jenkins (nexus-creds).**
   * **Tomcat Credentials → username/password stored in Jenkins (tomcat-credentials).**
   * **Credentials used in Jenkinsfile with withCredentials.**
5. **Configured Jenkins Pipeline**
   * **Declarative pipeline created using Jenkinsfile from GitHub.**
   * **Stages: Checkout SCM → Build → SonarQube → Deploy to Nexus → Deploy to Tomcat → Slack notification.**
6. **Environment Variables**
   * **Injected NEXUS\_USER/NEXUS\_PASS, TOMCAT\_USER/TOMCAT\_PASS securely using withCredentials.**

**CREDENTIALS**

****

## **2️⃣ SonarQube EC2 Configuration**

**EC2 Instance: SonarQube server**

**Actions & Configurations:**

1. **Installed SonarQube**
   * **Installed SonarQube using Docker (docker pull sonarqube:lts).**
   * **Ran on its EC2 instance.**
2. **Exposed SonarQube Web UI**
   * **Default port 9000 → http://<SonarQube-IP>:9000.**
3. **Configured Quality Profiles & Rules**
   * **Default “Sonar way” profiles were used for Java/JSP/XML.**
   * **Ensured analysis can run without custom configuration initially.**
4. **Connected Jenkins with SonarQube**
   * **In Jenkins → Manage Jenkins → Configure System → SonarQube servers.**
   * **Created a server configuration with name SonarQube.**
   * **Jenkins uses this to run analysis in the pipeline.**

## **3️⃣ Nexus EC2 Configuration**

**EC2 Instance: Nexus Repository server**

**Actions & Configurations:**

1. **Installed Nexus Repository Manager**
   * **Probably Nexus OSS version installed.**
   * **Default port 8081 → http://<Nexus-IP>:8081.**
2. **Created Repositories**
   * **releases repository for production artifacts.**
3. **Configured Credentials in Jenkins**
   * **Created username/password in Jenkins (nexus-creds).**
   * **Used in pipeline to upload artifacts (mvn deploy).**
4. **Configured Maven settings.xml**
   * **Included Nexus server credentials (optional if using Jenkins credentials).**

## **4️⃣ Tomcat EC2 Configuration**

**EC2 Instance: Tomcat server**

**Actions & Configurations:**

1. **Installed Tomcat**
   * **Probably version 9 or 10 installed at /opt/tomcat.**
   * **Default port 8080 → http://<Tomcat-IP>:8080.**
2. **Created Tomcat User**
   * **Username/password: deployer/deployer.**
   * **Role: manager-script in tomcat-users.xml.**
   * **This allows WAR deployment using the Tomcat Manager API.**
3. **Configured Remote Deployment**
   * **Jenkins uses curl to deploy WAR to Tomcat Manager:**

**curl -u $TOMCAT\_USER:$TOMCAT\_PASS -T target/hiring.war http://54.145.142.96:8080/manager/text/deploy?path=/hiring&update=true**

* **Path prefix /hiring used for the web application.**

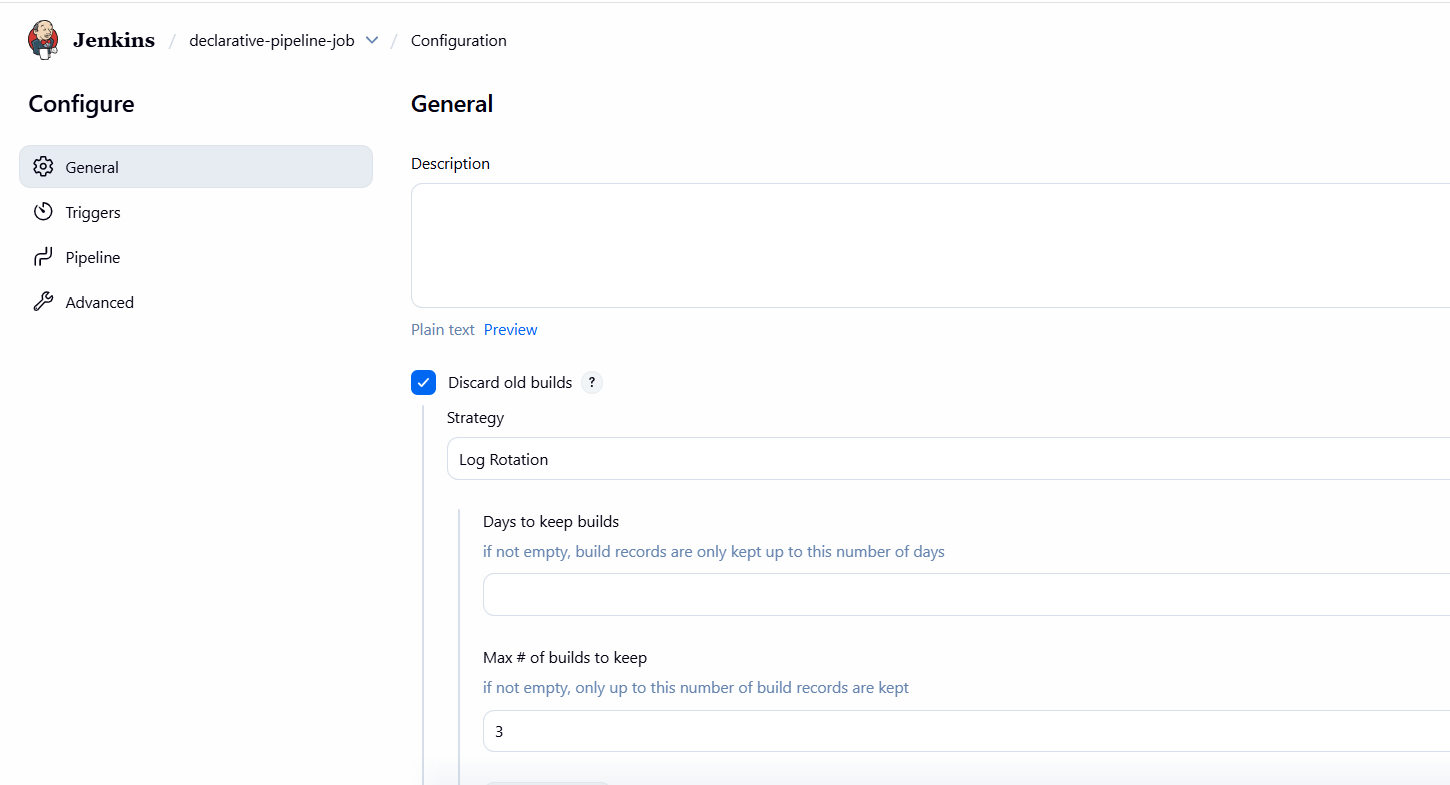
**Verified Deployment**

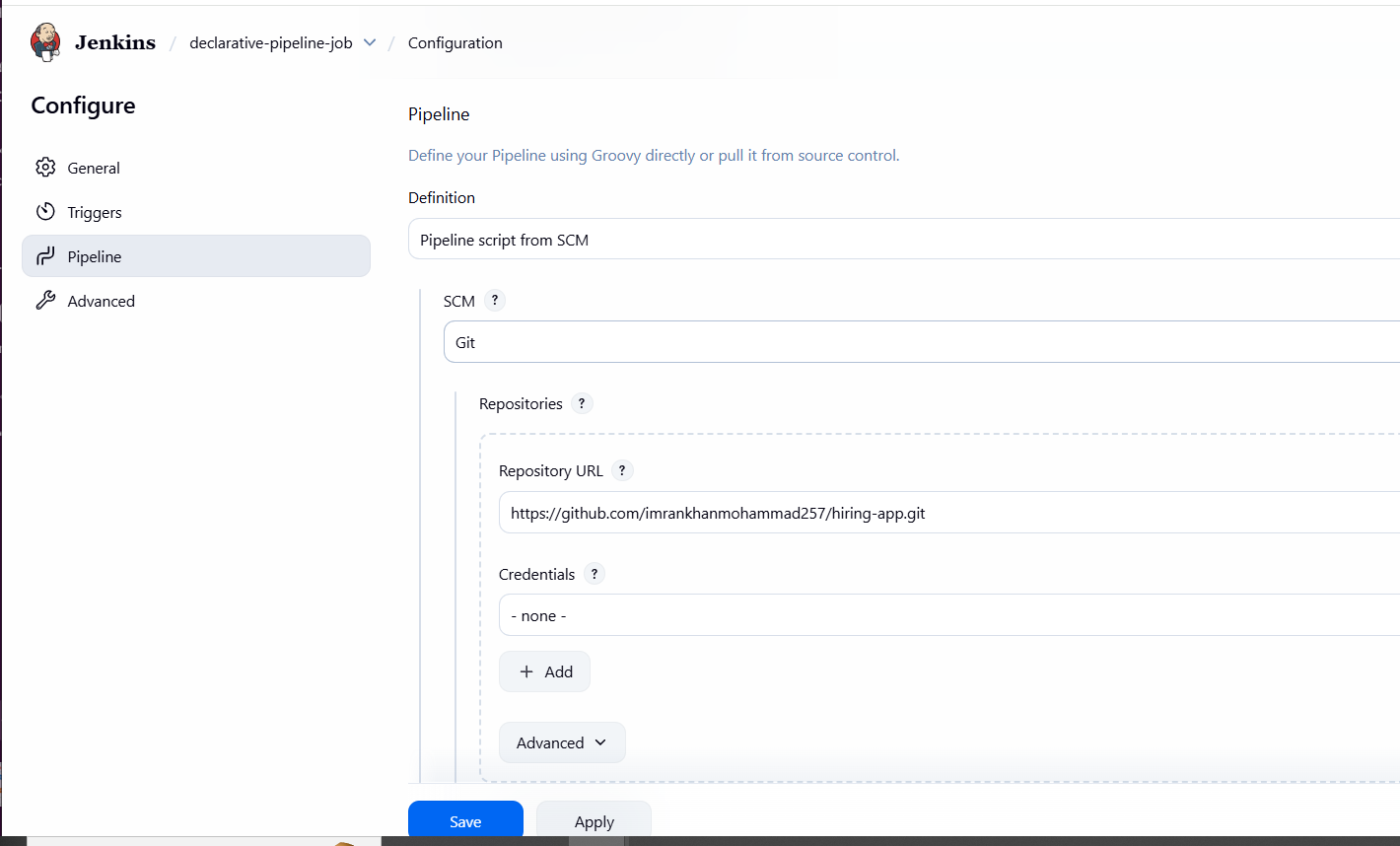
* **WAR file uploaded via Jenkins → visible at http://54.145.142.96:8080/hiring.**
* **Updated WAR reflects HTML changes pushed in GitHub.**

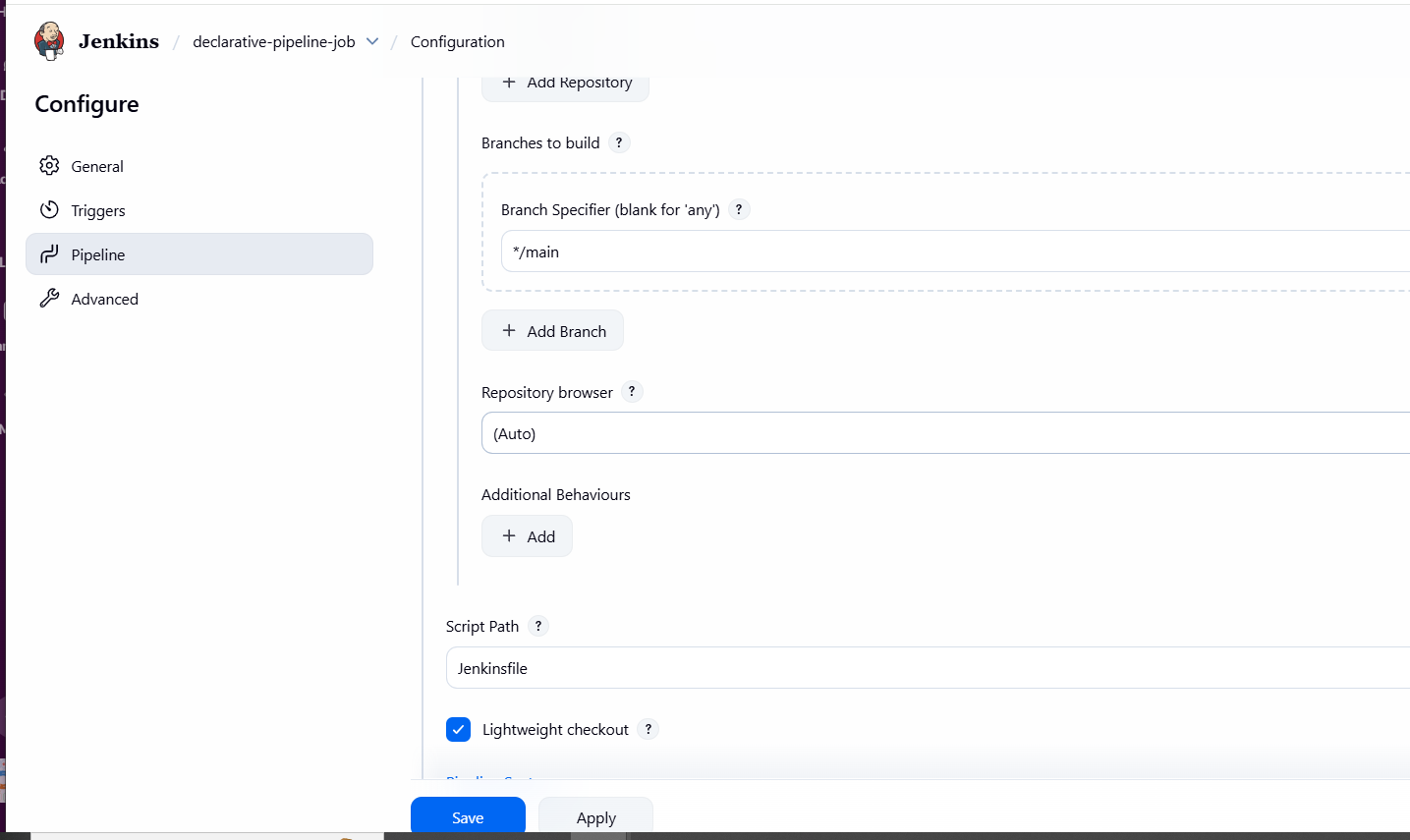
## **5️⃣ Jenkins Pipeline Flow**

**Stages & Steps Recap:**

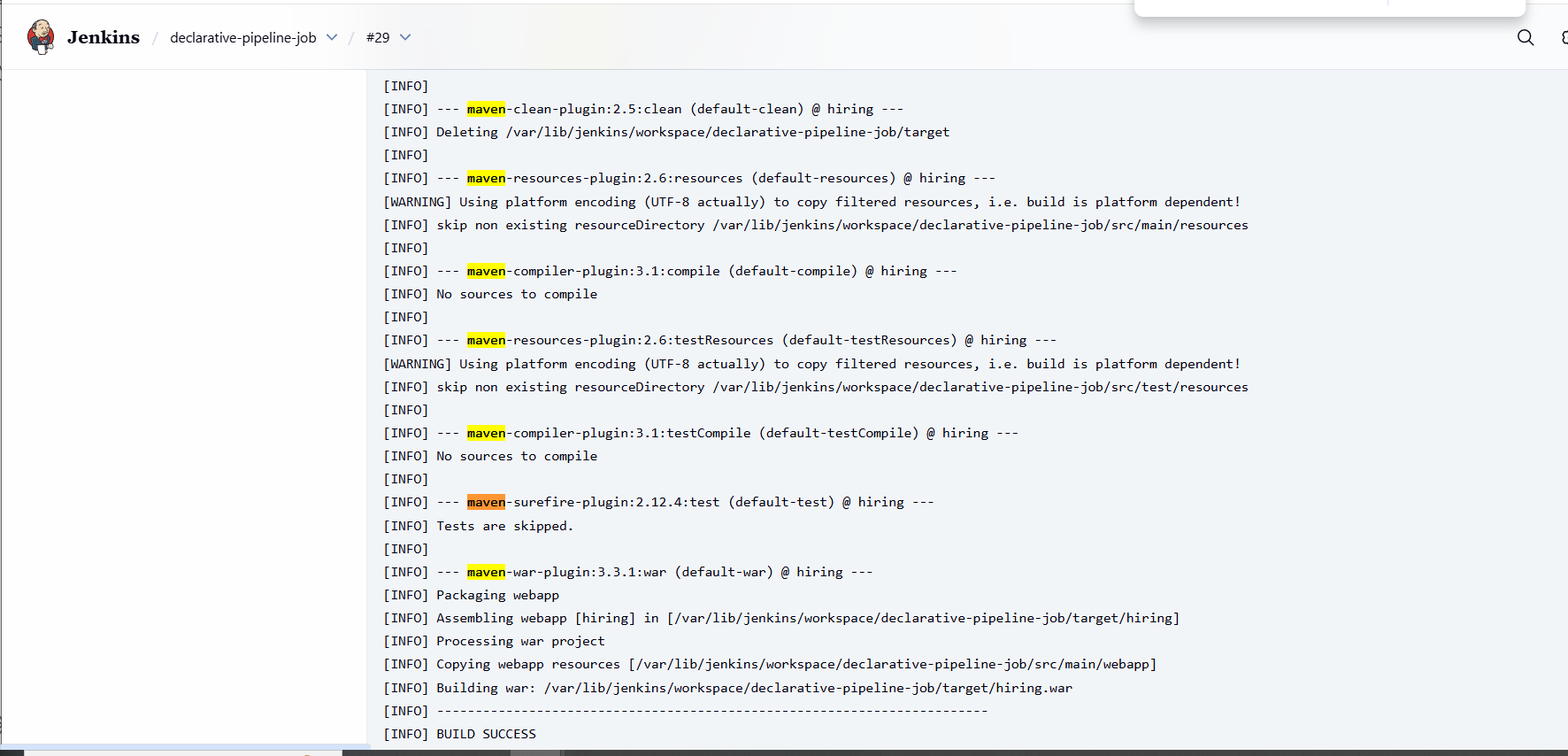
1. **Checkout SCM: Git clone the project.**

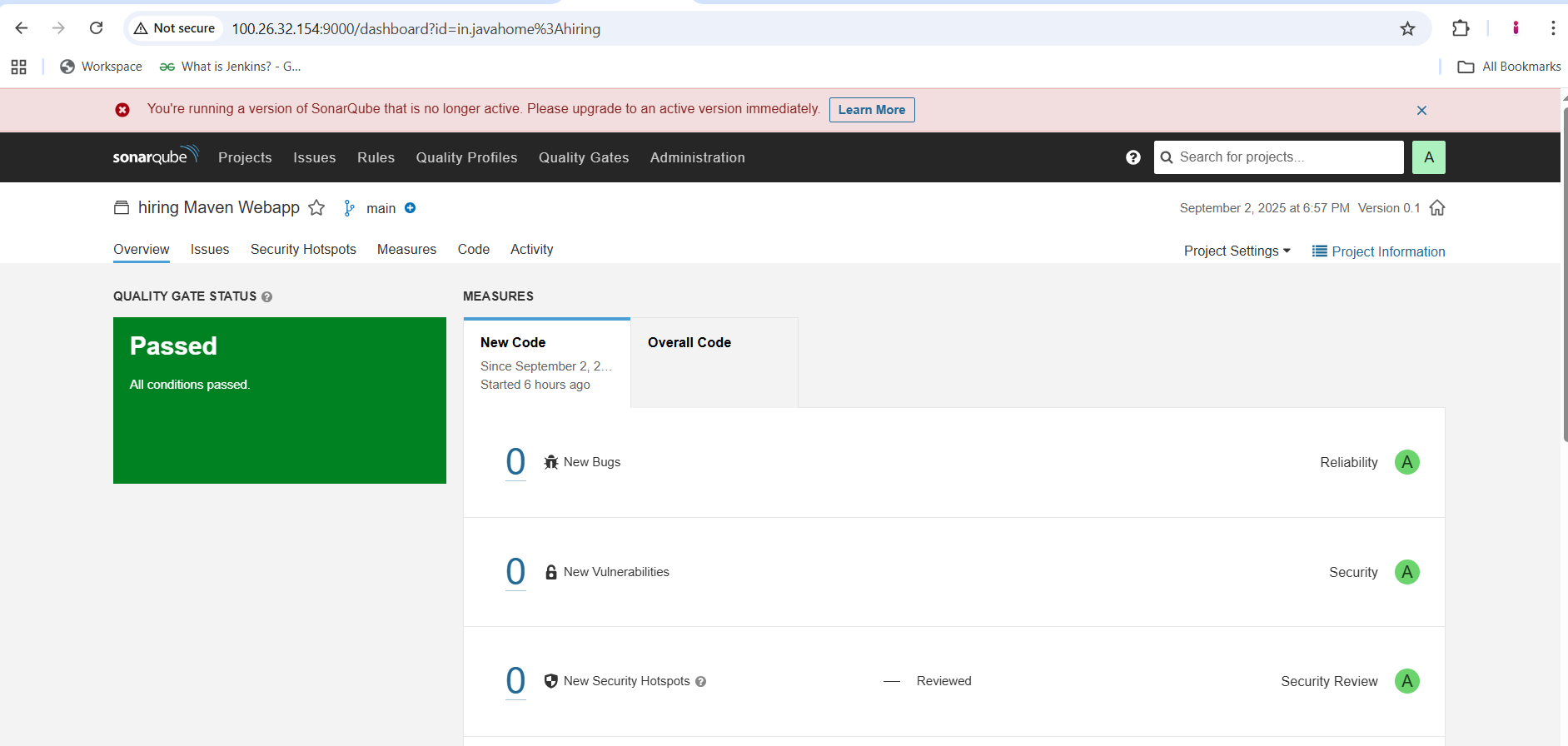
****

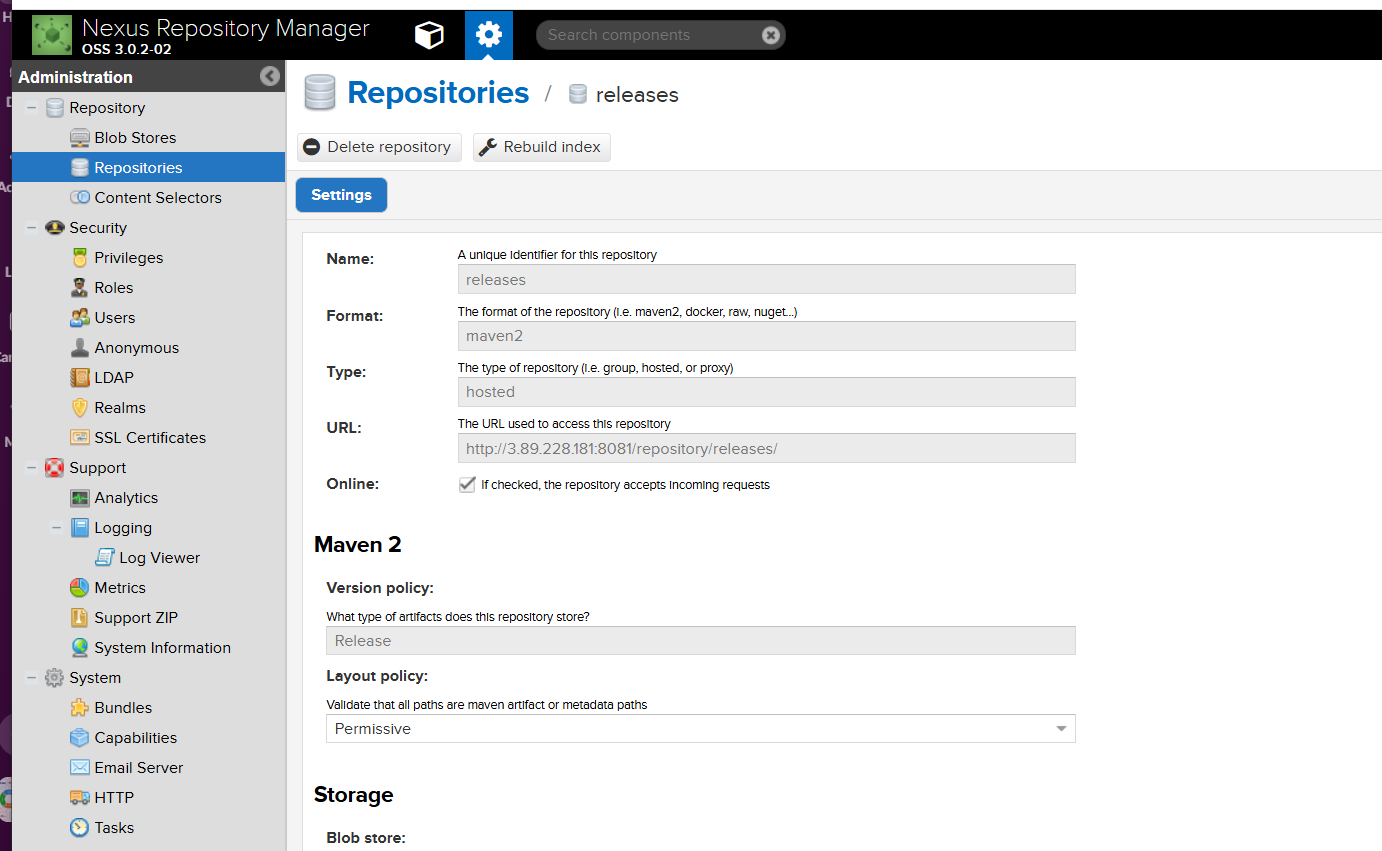
****

****

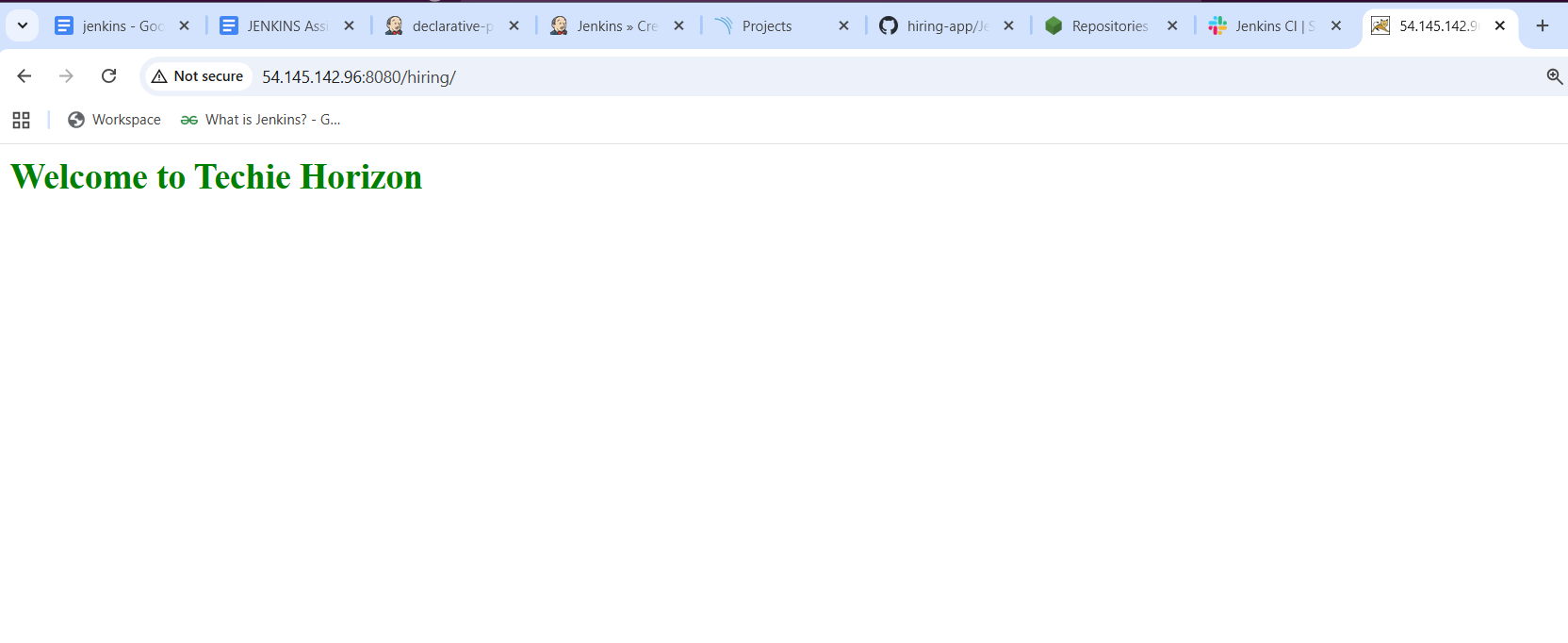
1. **Build: Maven clean package**

****

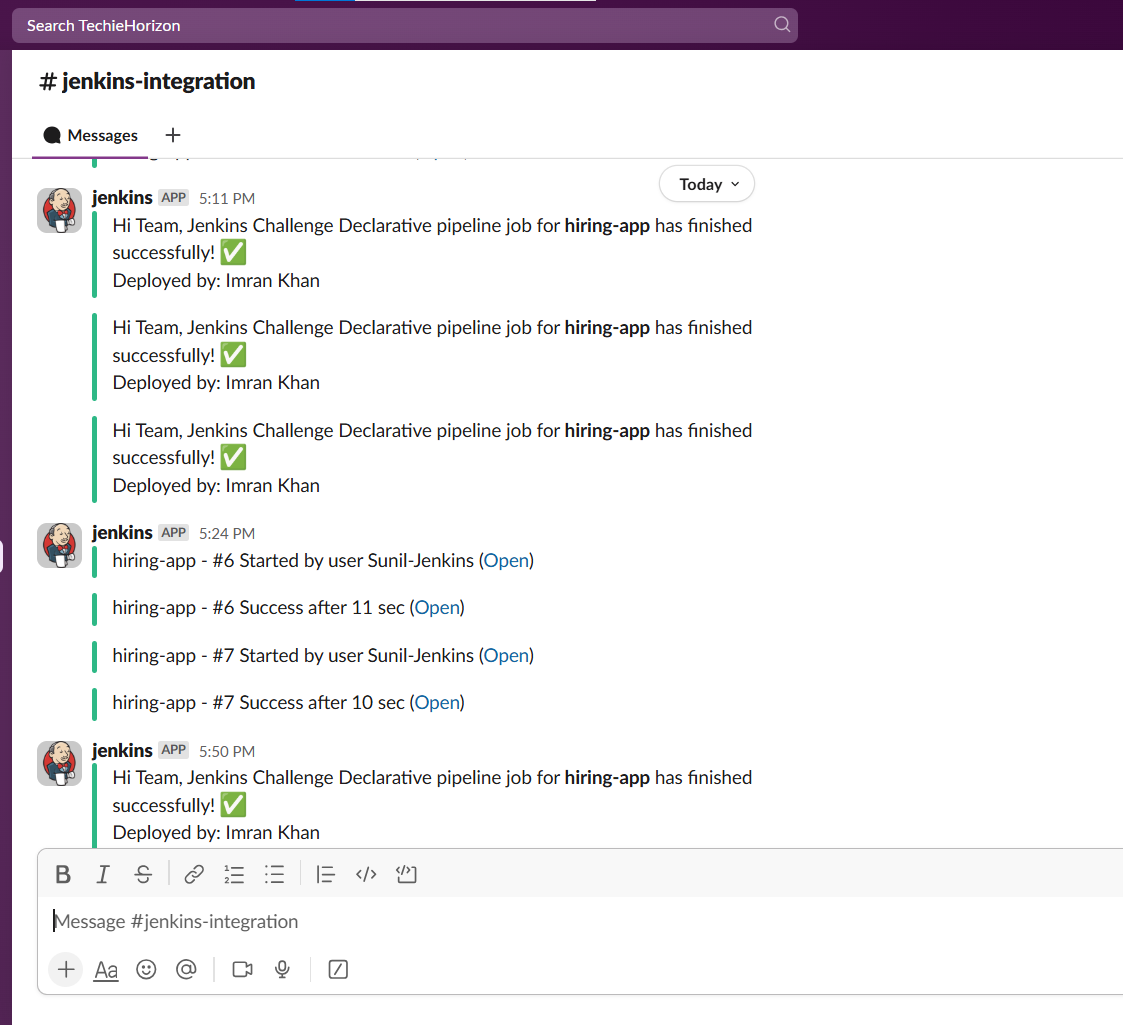
1. **SonarQube Analysis: Run static code analysis.**
2. ****
3. ****
4. **Deploy to Nexus: Upload WAR to Nexus repository.**

****

1. **Deploy to Tomcat: Upload WAR to remote Tomcat via curl.**

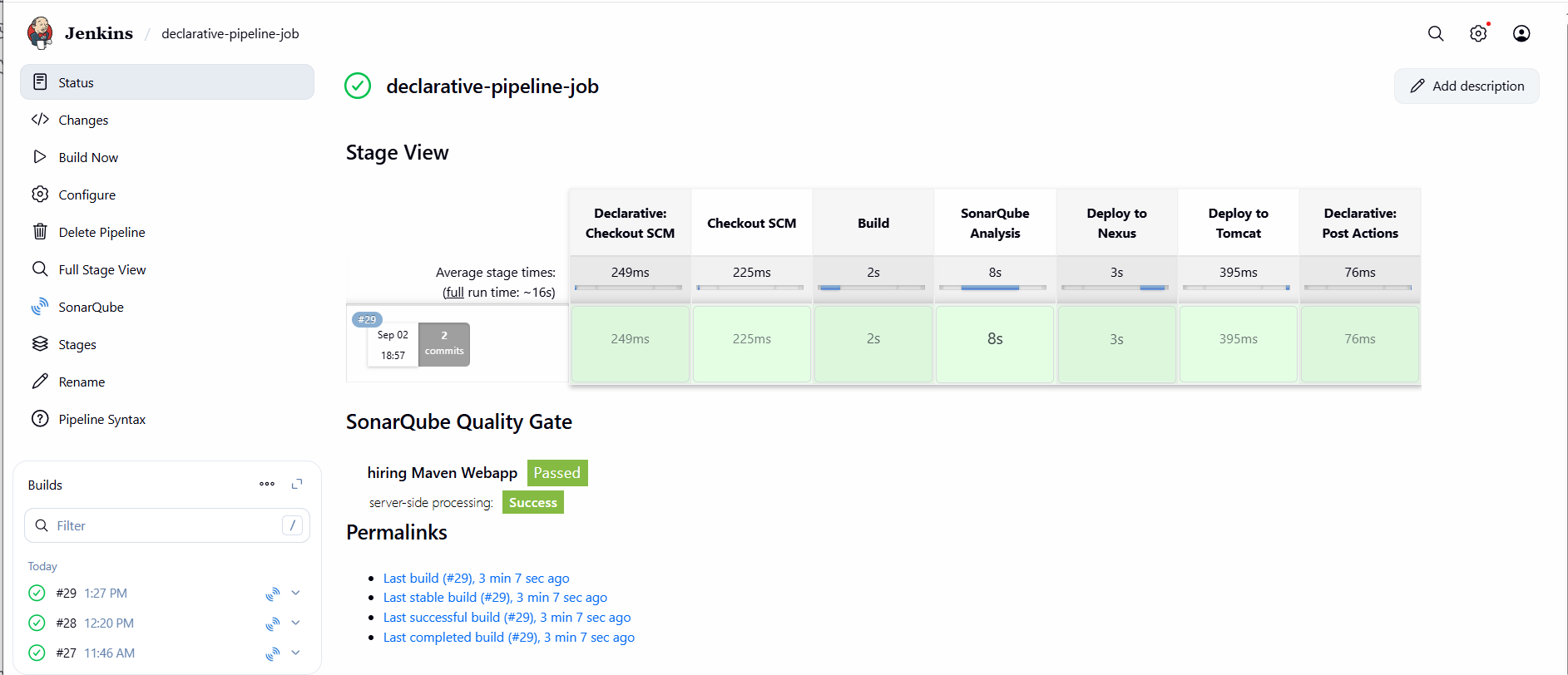
****

1. **Slack Notification: Send message to #jenkins-integration channel.**

****

1. **Post Actions: Echo “Pipeline finished” in Jenkins logs.**

**BUILD JOB**

****

**Declarative script of jenkins file in github repo**

**pipeline {**

**agent any**

**stages {**

**stage('Checkout SCM') {**

**steps {**

**git url: 'https://github.com/imrankhanmohammad257/hiring-app.git', branch: 'main'**

**}**

**}**

**stage('Build') {**

**steps {**

**tool name: 'Maven-3.8.4', type: 'maven'**

**sh 'mvn clean package -DskipTests'**

**}**

**}**

**stage('SonarQube Analysis') {**

**steps {**

**withSonarQubeEnv('SonarQube') {**

**sh 'mvn sonar:sonar'**

**}**

**}**

**}**

**stage('Deploy to Nexus') {**

**steps {**

**withCredentials([usernamePassword(credentialsId: 'nexus-creds', usernameVariable: 'NEXUS\_USER', passwordVariable: 'NEXUS\_PASS')]) {**

**sh 'mvn clean deploy -DskipTests -Dnexus.username=$NEXUS\_USER -Dnexus.password=$NEXUS\_PASS --settings /var/lib/jenkins/.m2/settings.xml'**

**}**

**}**

**}**

**stage('Deploy to Tomcat') {**

**steps {**

**withCredentials([usernamePassword(credentialsId: 'tomcat-credentials', usernameVariable: 'TOMCAT\_USER', passwordVariable: 'TOMCAT\_PASS')]) {**

**sh '''**

**curl -u $TOMCAT\_USER:$TOMCAT\_PASS \**

**-T target/hiring.war \**

**http://54.145.142.96:8080/manager/text/deploy?path=/hiring&update=true**

**'''**

**}**

**}**

**}**

**stage('Slack Notification') {**

**steps {**

**slackSend(**

**channel: '#jenkins-integration',**

**color: 'good',**

**message: "Hi Team, Jenkins Challenge Declarative pipeline job for \*hiring-app\* has finished successfully! ✅\nDeployed by: Imran Khan"**

**)**

**}**

**}**

**}**

**post {**

**always {**

**echo 'Pipeline finished'**

**}**

**}**

**}**

**2) Create one Scripted pipeline job**

# **📘 Jenkins Scripted Pipeline – Hiring App Deployment**

## **🔹 Objective**

**Automate the CI/CD pipeline for a simple Java Web Application (hiring-app) with the following integrations:**

1. **Source Code Management – GitHub**
2. **Build Tool – Maven**
3. **Code Quality – SonarQube**
4. **Artifact Repository – Nexus**
5. **Deployment – Apache Tomcat**
6. **Notification – Slack**

## **🔹 Infrastructure Setup**

* **Jenkins EC2: Installed Jenkins on an AWS EC2 instance (t2.large).**
* **SonarQube EC2: Installed Docker, pulled and ran the SonarQube container.**
* **Nexus EC2: Installed Nexus Repository Manager.**
* **Tomcat EC2: Installed Apache Tomcat 9 for WAR deployment.**

## **🔹 Jenkins Configuration**

1. **Installed Plugins:**
   * **Git Plugin**
   * **Pipeline Plugin**
   * **SonarQube Scanner Plugin**
   * **Slack Notification Plugin**
2. **Configured Tools:**
   * **Maven (Maven-3.8.4)**
   * **JDK (Java 11/17 depending on app requirements)**
3. **Credentials Added in Jenkins:**
   * **nexus-creds → Nexus username/password**
   * **tomcat-credentials → Tomcat manager username/password**
   * **Slack Webhook for notifications**
4. **SonarQube Integration:**
   * **Added SonarQube server details under Manage Jenkins → Configure System.**

**🔹 Scripted Pipeline (Jenkinsfile)**

**node {**

**stage('Checkout SCM') {**

**git branch: 'main',**

**url: 'https://github.com/imrankhanmohammad257/hiring-app.git'**

**}**

**stage('Build') {**

**def mvnHome = tool name: 'Maven-3.8.4', type: 'maven'**

**sh "${mvnHome}/bin/mvn clean package -DskipTests"**

**}**

**stage('SonarQube Analysis') {**

**def mvnHome = tool name: 'Maven-3.8.4', type: 'maven'**

**withSonarQubeEnv('SonarQube') {**

**sh "${mvnHome}/bin/mvn sonar:sonar"**

**}**

**}**

**stage('Deploy to Nexus') {**

**withCredentials([usernamePassword(credentialsId: 'nexus-creds', usernameVariable: 'NEXUS\_USER', passwordVariable: 'NEXUS\_PASS')]) {**

**sh """**

**mvn clean deploy -DskipTests \**

**-Dnexus.username=$NEXUS\_USER \**

**-Dnexus.password=$NEXUS\_PASS**

**"""**

**}**

**}**

**stage('Deploy to Tomcat') {**

**withCredentials([usernamePassword(credentialsId: 'tomcat-credentials', usernameVariable: 'TOMCAT\_USER', passwordVariable: 'TOMCAT\_PASS')]) {**

**sh '''**

**# First undeploy the old app**

**curl -u $TOMCAT\_USER:$TOMCAT\_PASS "http://54.87.222.232:8080/manager/text/undeploy?path=/hiring"**

**# Now deploy the new WAR**

**curl -u $TOMCAT\_USER:$TOMCAT\_PASS \**

**-T target/hiring.war \**

**"http://54.87.222.232:8080/manager/text/deploy?path=/hiring&update=true"**

**'''**

**}**

**}**

**stage('Slack Notification') {**

**slackSend(**

**channel: '#jenkins-integration',**

**color: 'good',**

**message: "Hi Team, Jenkins Scripted pipeline job for \*hiring-app\* has finished successfully! ✅\nDeployed by: Imran Khan"**

**)**

**}**

**}**

****

## **🔹 Execution Flow**

1. **SCM Checkout** → Code pulled from GitHub repo (hiring-app).
2. **Build with Maven** → Cleaned and packaged WAR file (target/hiring.war).
3. **SonarQube Analysis** → Code quality check performed.
4. **Deploy to Nexus** → Artifact pushed to Nexus Releases Repository.
5. **Deploy to Tomcat** → WAR deployed on Tomcat server.
6. **Slack Notification** → Team notified on Slack channel after successful deployment.

## **🔹 Testing the Deployment**

* Application accessible at:  
   👉 http://54.87.222.232:8080/hiring/
* Updated index.jsp customized with attractive UI and background.

## **🔹 Issues Faced & Fixes**

1. **401 Unauthorized (Nexus Deploy)**
   * Fixed by updating settings.xml with correct credentials using Jenkins credentials store.
2. **WAR Not Updating in Tomcat**
   * Used update=true flag in Tomcat deploy URL.
   * Cleared browser cache to load latest changes.
3. **Plain JSP Page**
   * Enhanced UI with gradient background, animations, and styled text.

✅ **End Result** → Fully automated CI/CD pipeline for hiring-app with **Git → Maven → SonarQube → Nexus → Tomcat → Slack**.

Added data in POM.XML in github repo

**<distributionManagement>**

**<repository>**

**<id>nexus-releases</id>**

**<url>http://54.209.186.138:8081/repository/releases/</url>**

**</repository>**

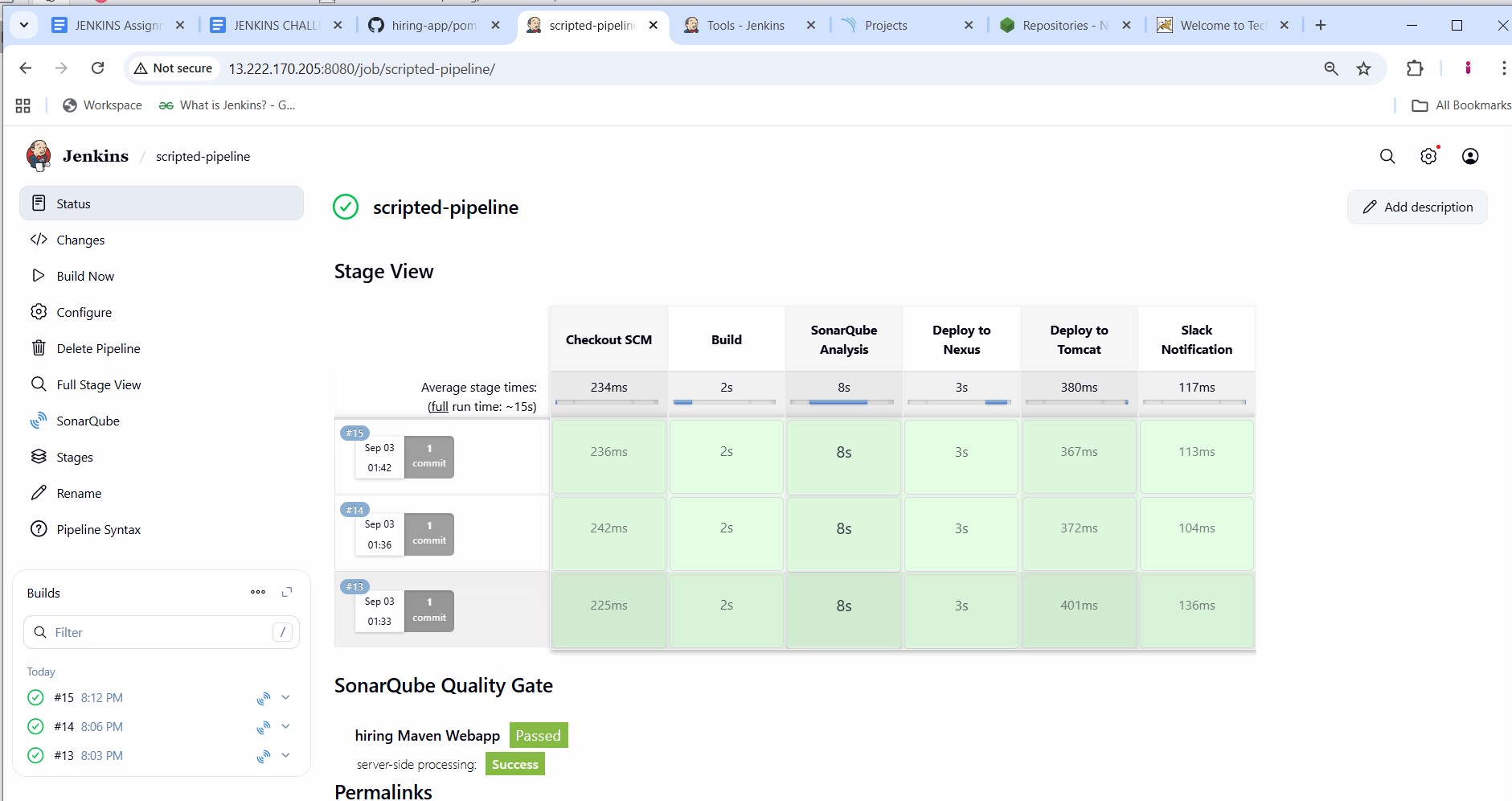
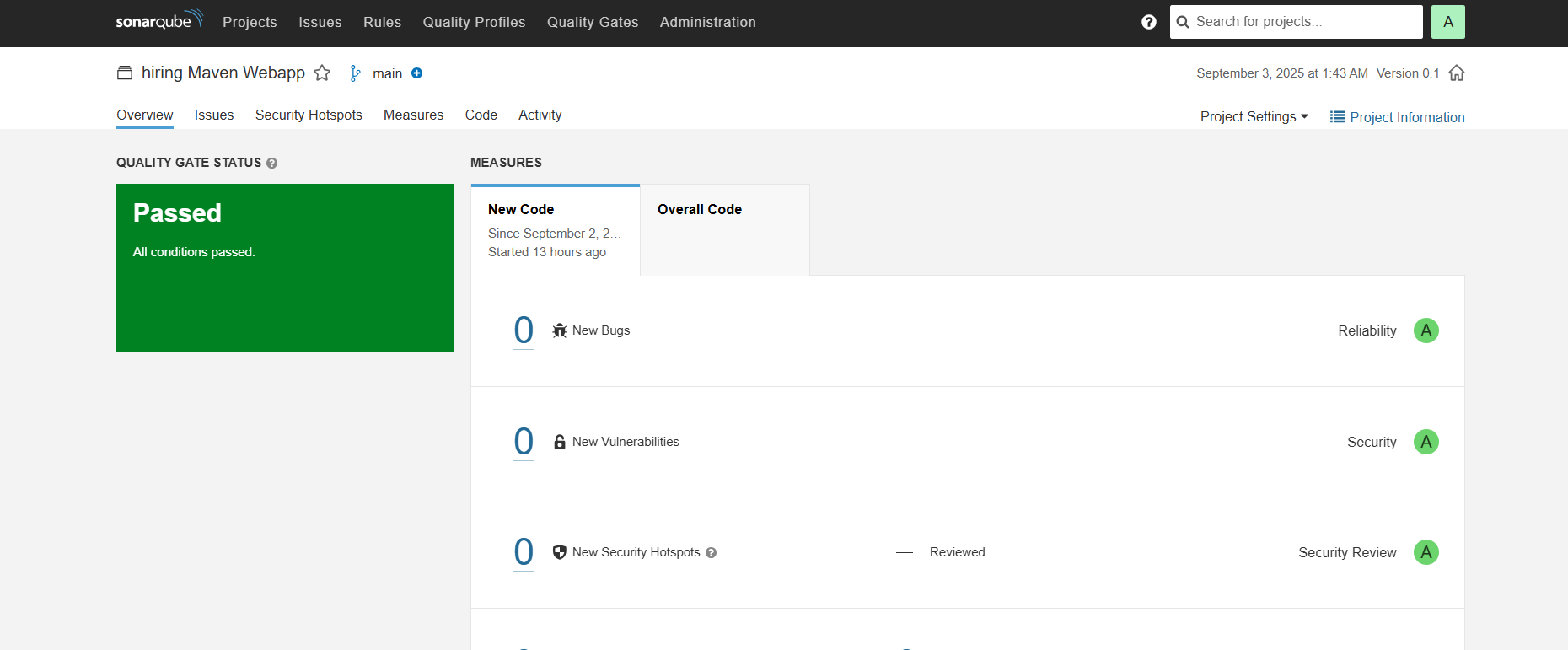
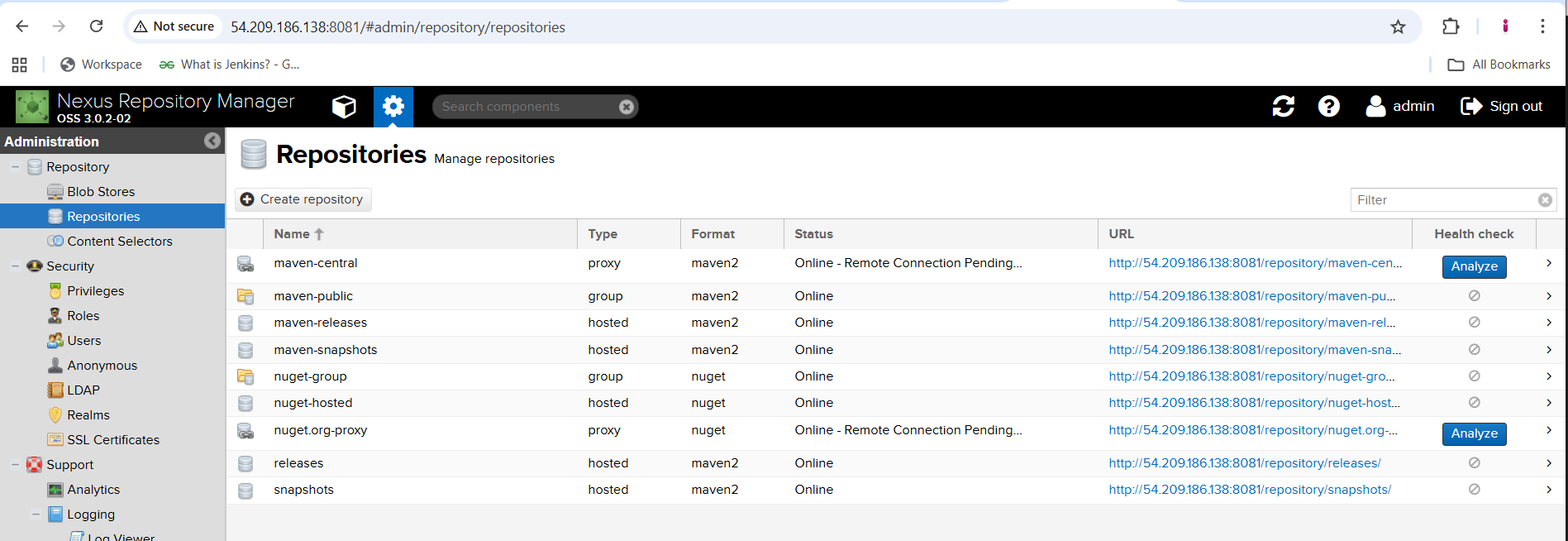
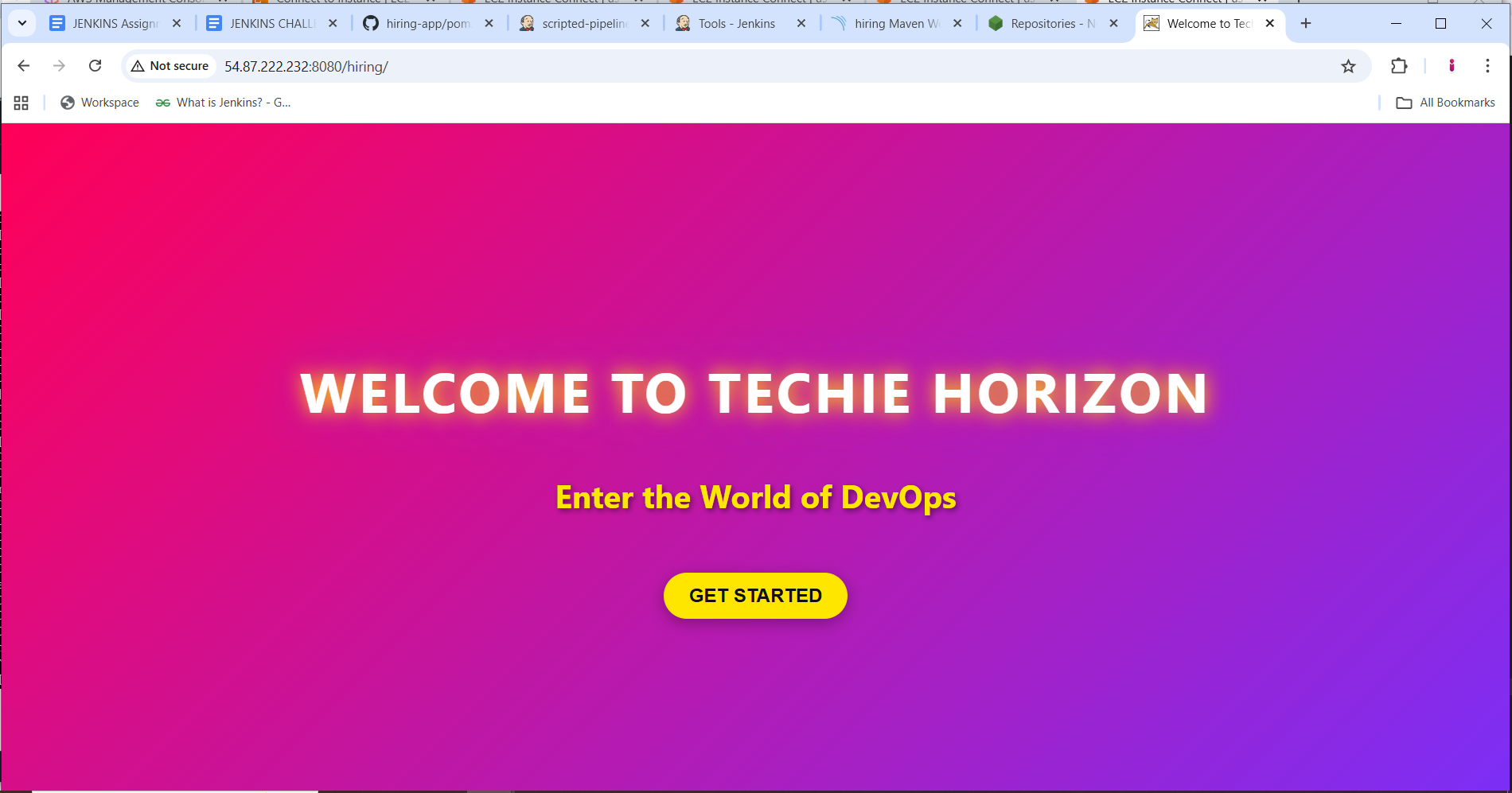
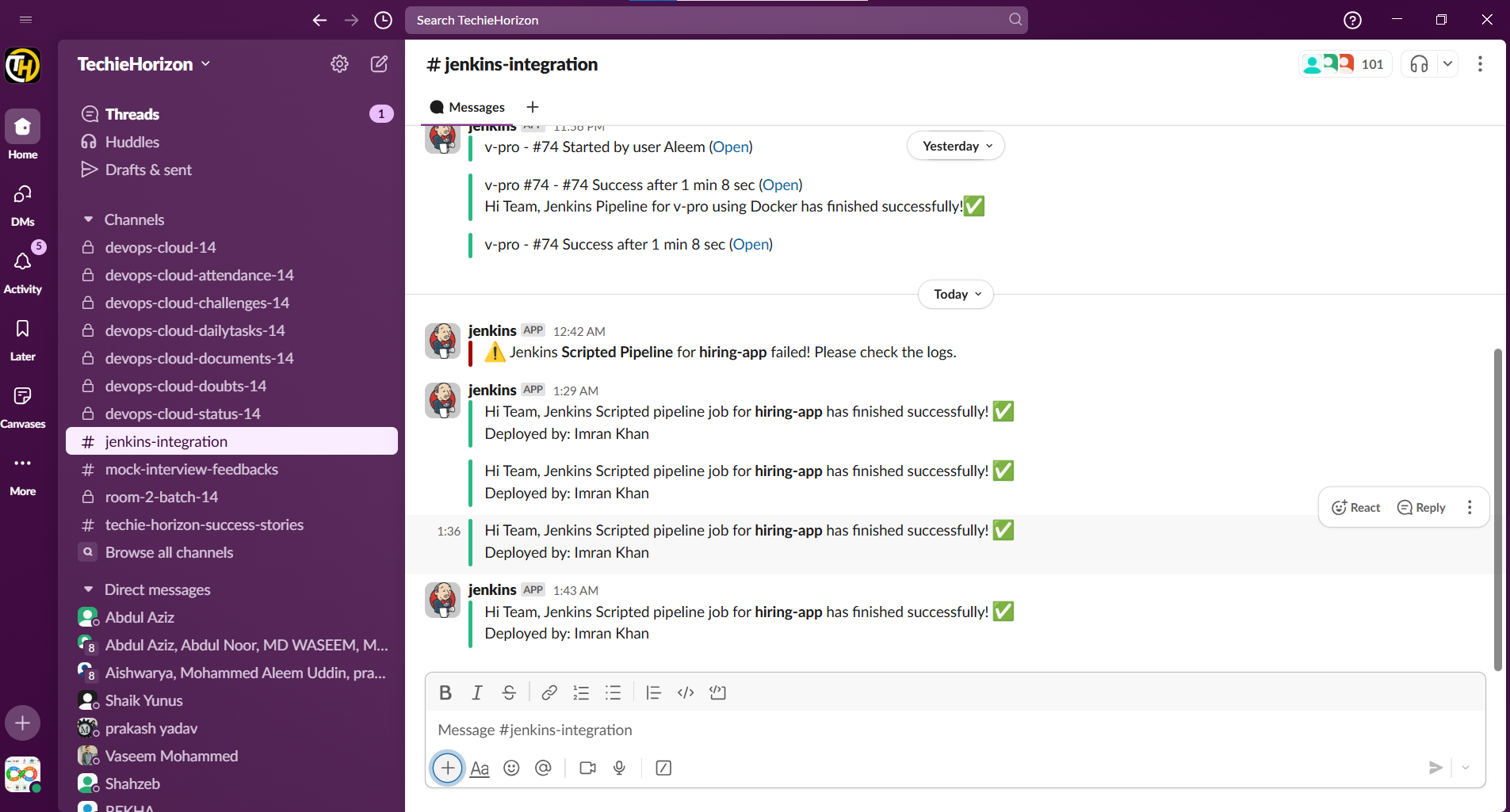
**<snapshotRepository>**

**<id>nexus-snapshots</id>**

**<url>http://54.209.186.138:8081/repository/snapshots/</url>**

**</snapshotRepository>**

**</distributionManagement>**

1. **Job build**
2. ****
3. **Sonarqube quality check**
4. ****
5. ****
6. **NEXUS repos created, releases and snapshots**
7. ****
8. **TOMCAT deployment**
9. ****
10. **SLACK NOTIFICATIONS**
11. ****

**3) Create one multi stage pipeline job**

# **Documentation: Multi-Stage Jenkins Pipeline for Hiring-App**

## **1. Objective**

**Create a Jenkins multi-stage scripted pipeline that:**

1. **Checks out code from GitHub.**
2. **Builds the Maven project.**
3. **Performs SonarQube code analysis.**
4. **Deploys artifacts to Nexus repository.**
5. **Deploys WAR file to Tomcat server.**
6. **Sends Slack notification after successful deployment.**

## **2. Prerequisites**

1. **Jenkins EC2 instance with:**
   * **Java (JDK 17 recommended)**
   * **Maven (configured either via Jenkins tool or manually)**
   * **Jenkins user with proper permissions**
2. **GitHub repository:  
    https://github.com/imrankhanmohammad257/hiring-app.git**
3. **SonarQube server (e.g., http://54.90.169.53:9000) configured in Jenkins.**
4. **Nexus repository (e.g., http://54.163.17.174:8081) with credentials in Jenkins.**
5. **Tomcat server for deployment, with manager credentials.**
6. **Slack webhook and channel for notifications (optional).**

## **3. Environment Setup on Jenkins EC2**

### **3.1 Install Maven (if not already installed)**

**cd /opt**

**wget https://dlcdn.apache.org/maven/maven-3/3.9.11/binaries/apache-maven-3.9.11-bin.tar.gz**

**tar -xvf apache-maven-3.9.11-bin.tar.gz**

**cd apache-maven-3.9.11/bin**

**Update .bash\_profile:**

**MVN\_HOME=/opt/apache-maven-3.9.11/bin**

**PATH=$PATH:$HOME/bin:$MVN\_HOME**

**export PATH**

**source ~/.bash\_profile**

**mvn -version**

### **3.2 Configure Jenkins Tools**

* **Go to Manage Jenkins → Global Tool Configuration → Maven**
* **Add Maven installation:**
  + **Name: Maven-3.8.4 (or your version)**
  + **Install automatically: No (use installed path)**

### **3.3 Configure Jenkins Credentials**

1. **Nexus Credentials**
   * **ID: nexus-creds**
   * **Type: Username with password**
   * **Username/Password: Nexus login**
2. **Tomcat Credentials**
   * **ID: tomcat-credentials**
   * **Type: Username with password**
   * **Username/Password: Tomcat manager**
3. **Slack Token (optional)**
   * **ID: slack-token**

### **3.4 Configure settings.xml for Maven (Optional)**

* **Location: /var/lib/jenkins/.m2/settings.xml**

**in jenkins ec2 change or create maven file**

**cd /var/lib/jenkins/.m2**

**sudo vi settings.xml**

**<settings>**

**<servers>**

**<server>**

**<id>releases</id>**

**<username>admin</username>**

**<password>admin123</password>**

**</server>**

**<server>**

**<id>snapshots</id>**

**<username>admin</username>**

**<password>admin123</password>**

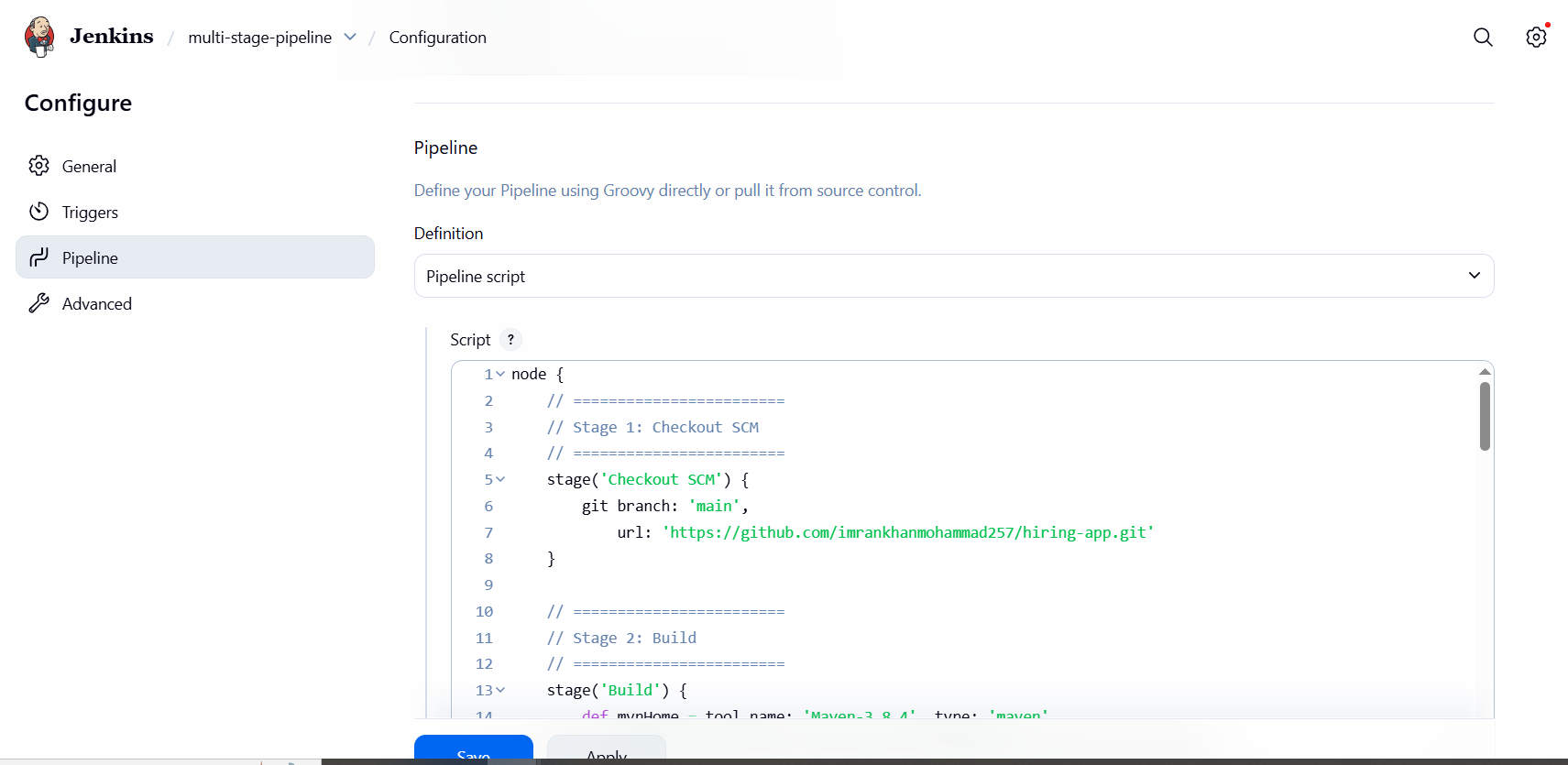
**</server>**

**</servers>**

**</settings>**

## **4. Pipeline Script**

**Here is the Scripted Pipeline for hiring-app:**

****

**node {**

**// ========================**

**// Stage 1: Checkout SCM**

**// ========================**

**stage('Checkout SCM') {**

**git branch: 'main',**

**url: 'https://github.com/imrankhanmohammad257/hiring-app.git'**

**}**

**// ========================**

**// Stage 2: Build**

**// ========================**

**stage('Build') {**

**def mvnHome = tool name: 'Maven-3.8.4', type: 'maven'**

**sh "${mvnHome}/bin/mvn clean package -DskipTests"**

**}**

**// ========================**

**// Stage 3: SonarQube Analysis**

**// ========================**

**stage('SonarQube Analysis') {**

**def mvnHome = tool name: 'Maven-3.8.4', type: 'maven'**

**withSonarQubeEnv('SonarQube') {**

**sh "${mvnHome}/bin/mvn sonar:sonar"**

**}**

**}**

**// ========================**

**// Stage 4: Deploy to Nexus**

**// ========================**

**stage('Deploy to Nexus') {**

**withCredentials([usernamePassword(credentialsId: 'nexus-creds', usernameVariable: 'NEXUS\_USER', passwordVariable: 'NEXUS\_PASS')]) {**

**def mvnHome = tool name: 'Maven-3.8.4', type: 'maven'**

**sh """**

**${mvnHome}/bin/mvn clean deploy -DskipTests \**

**-Dnexus.username=$NEXUS\_USER \**

**-Dnexus.password=$NEXUS\_PASS**

**"""**

**}**

**}**

**// ========================**

**// Stage 5: Deploy to Tomcat**

**// ========================**

**stage('Deploy to Tomcat') {**

**withCredentials([usernamePassword(credentialsId: 'tomcat-credentials', usernameVariable: 'TOMCAT\_USER', passwordVariable: 'TOMCAT\_PASS')]) {**

**sh """**

**# Undeploy old app**

**curl -s -u $TOMCAT\_USER:$TOMCAT\_PASS \**

**"http://3.82.42.125:8080/manager/text/undeploy?path=/hiring"**

**# Deploy new WAR**

**curl -s -u $TOMCAT\_USER:$TOMCAT\_PASS \**

**-T target/hiring.war \**

**"http://3.82.42.125:8080/manager/text/deploy?path=/hiring&update=true"**

**"""**

**}**

**}**

**// ========================**

**// Stage 6: Slack Notification**

**// ========================**

**stage('Slack Notification') {**

**slackSend(**

**channel: '#jenkins-integration',**

**color: 'good',**

**message: "Hi Team, Jenkins Scripted pipeline job for \*hiring-app\* has finished successfully! ✅\nDeployed by: Imran Khan"**

**)**

**}**

**}**

## **5. Pipeline Flow / Stages**

| **Stage** | **Purpose** |
| --- | --- |
| **Checkout SCM** | **Pull latest code from GitHub** |
| **Build** | **Compile & package the Maven project** |
| **SonarQube Analysis** | **Code quality check using SonarQube** |
| **Deploy to Nexus** | **Upload WAR file to Nexus repository** |
| **Deploy to Tomcat** | **Deploy WAR file to Tomcat server** |
| **Slack Notification** | **Inform team of successful deployment** |

## **6. Troubleshooting**

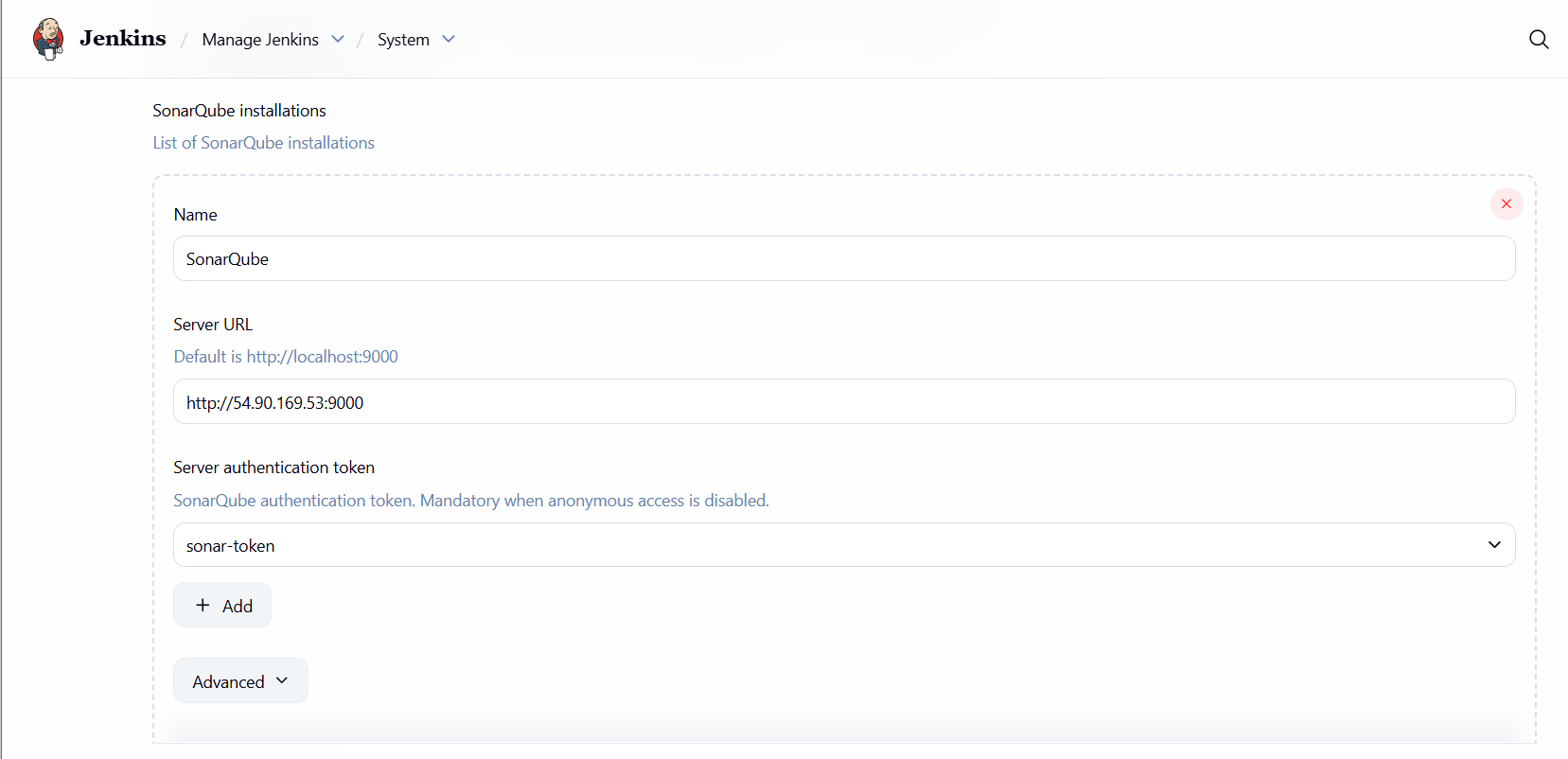
1. **401 Unauthorized on Nexus**
   * **Check Jenkins credential ID (nexus-creds) and username/password.**
   * **Verify <server> ID matches pom.xml.**
2. **SonarQube Analysis Fails**
   * **Ensure withSonarQubeEnv('SonarQube') matches Jenkins SonarQube installation name.**
   * **Ensure SonarQube server is running and accessible from EC2.**
3. **Tomcat Deploy Fails**
   * **Check Tomcat Manager credentials.**
   * **Ensure curl commands can reach Tomcat URL.**
4. **Maven Not Found**
   * **Verify Maven is installed and MVN\_HOME is correct.**
   * **Confirm tool name matches Jenkins Global Tool Configuration.**

## **7. References / Links**

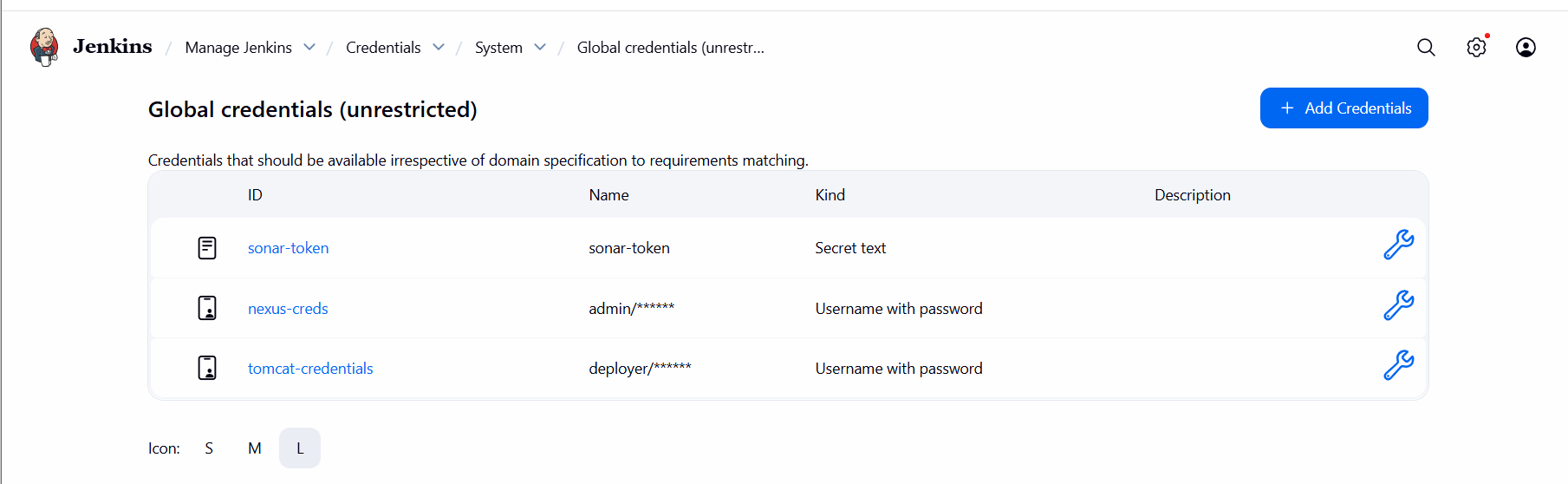
* **Maven:**[**https://maven.apache.org/**](https://maven.apache.org/)
* **Jenkins Pipeline:** [**https://www.jenkins.io/doc/book/pipeline/**](https://www.jenkins.io/doc/book/pipeline/)
* **SonarQube:** [**https://www.sonarqube.org/**](https://www.sonarqube.org/)
* **Nexus:** [**https://www.sonatype.com/nexus-repository-oss**](https://www.sonatype.com/nexus-repository-oss)
* **Tomcat Manager:** [**https://tomcat.apache.org/tomcat-9.0-doc/manager-howto.html**](https://tomcat.apache.org/tomcat-9.0-doc/manager-howto.html)

**Configurations in JENKINS PAGE**

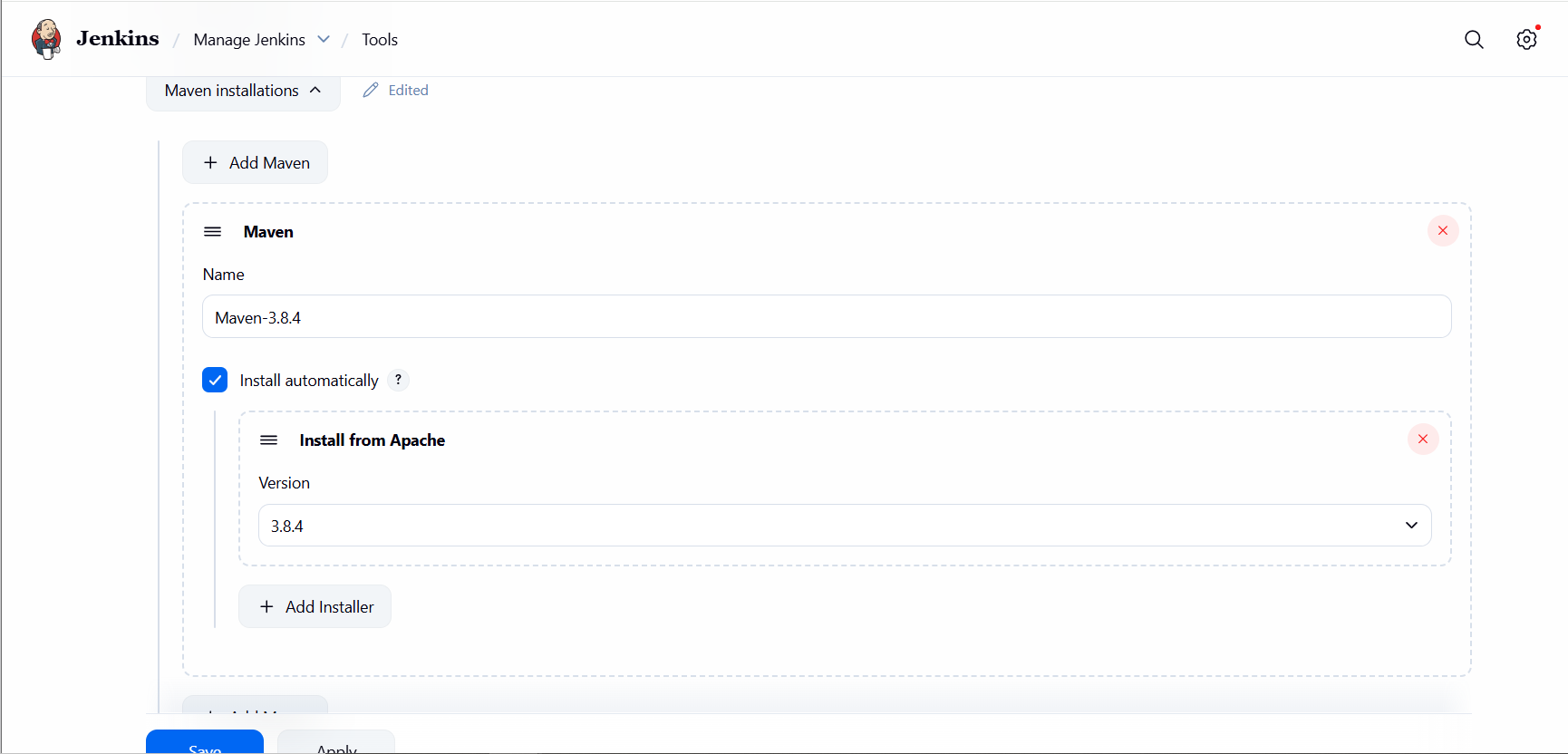
**SonarQube**

****

**Credentials**

****

**Maven**

****

**In pom.xml of git repo**

**<distributionManagement>**

**<repository>**

**<id>nexus-releases</id>**

**<url>http://54.209.186.138:8081/repository/releases/</url>**

**</repository>**

**<snapshotRepository>**

**<id>nexus-snapshots</id>**

**<url>http://54.209.186.138:8081/repository/snapshots/</url>**

**</snapshotRepository>**

**</distributionManagement>**

**TOMCAT USER**

**Tomcat change files comment at valve**

**vi /opt/tomcat/webapps/docs/META-INF/context.xml**

**vi /opt/tomcat/webapps/examples/META-INF/context.xml**

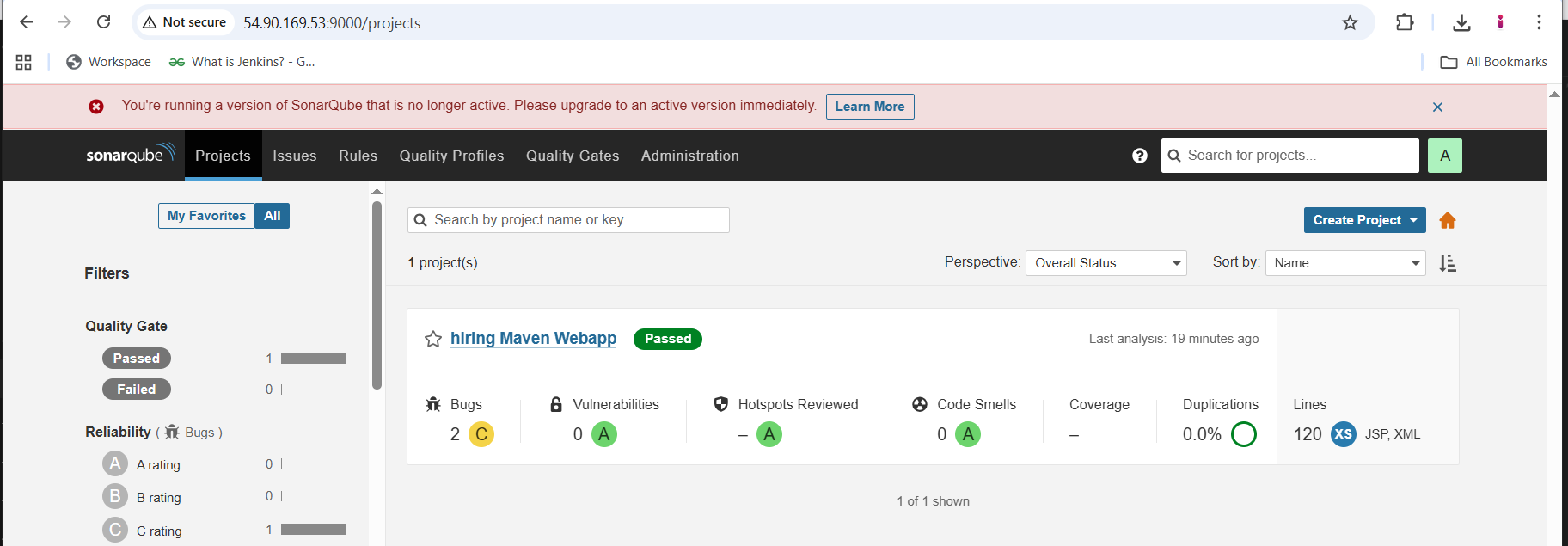
**vi /opt/tomcat/webapps/host-manager/META-INF/context.xml**

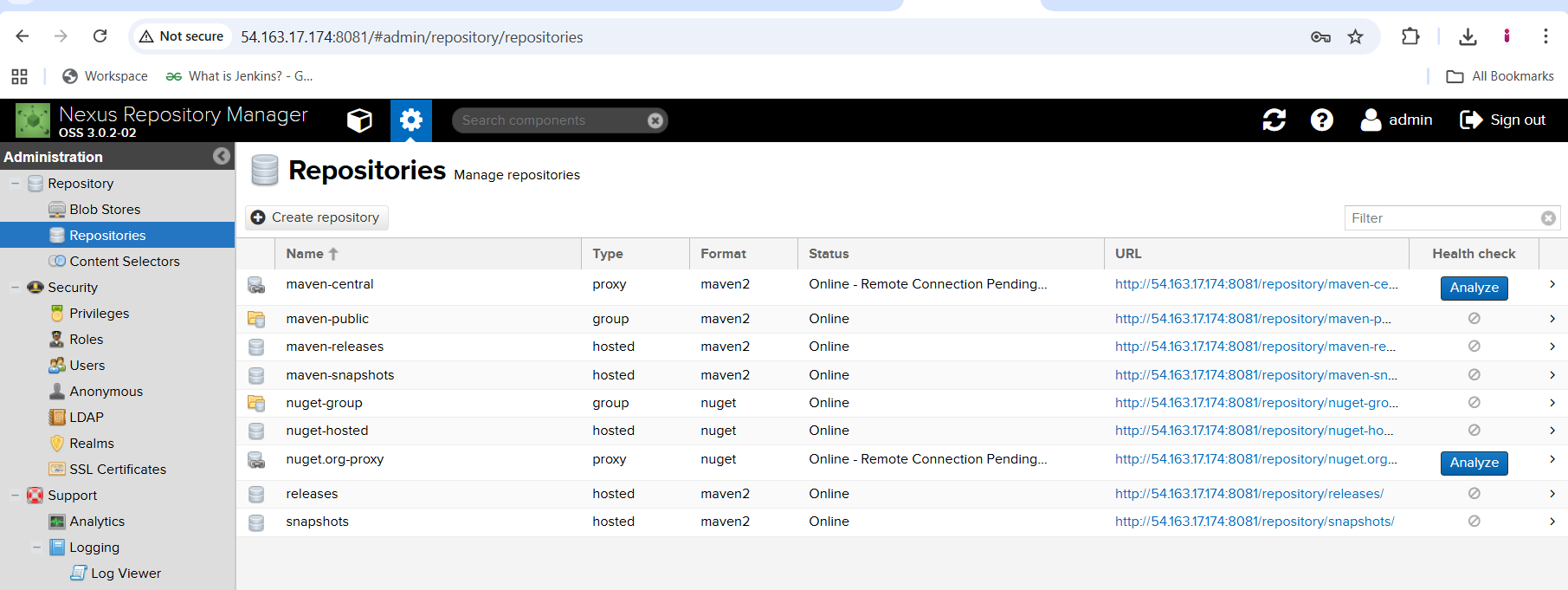
**vi /opt/tomcat/webapps/manager/META-INF/context.xml**

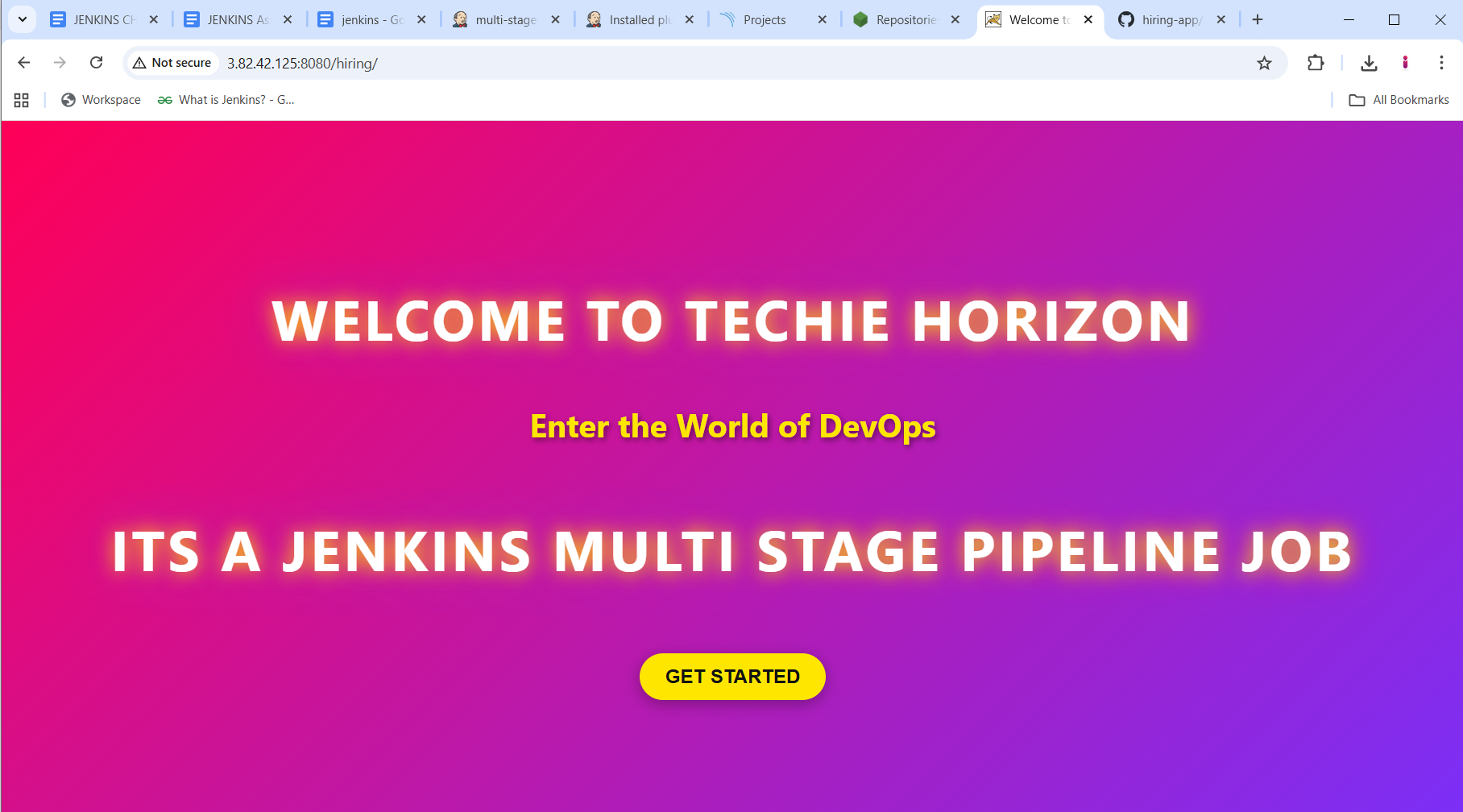
**cd /opt/tomcat/conf**

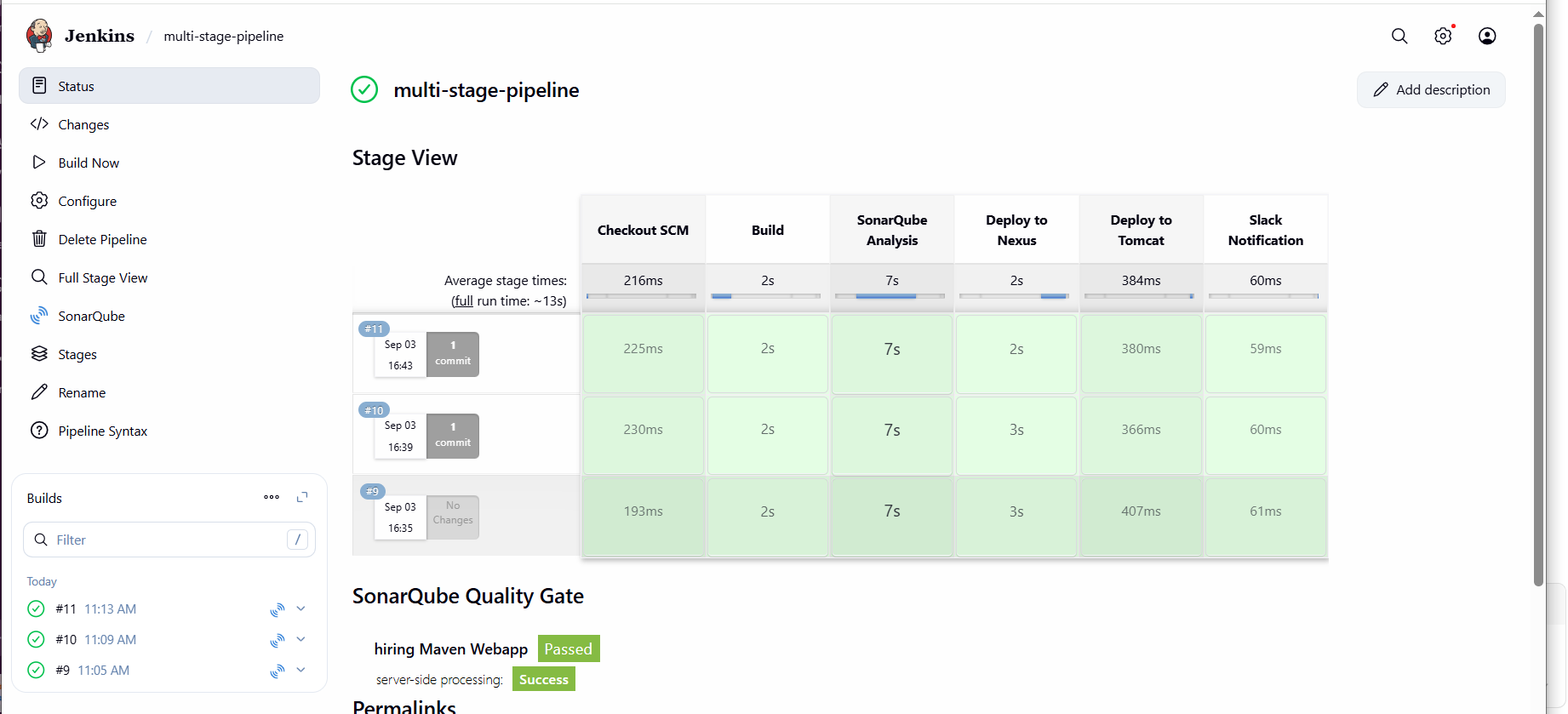
**vi tomcat-users.xml**

****

****

****

****

****

****

**4) Create one parallel stage pipeline job**

# **Documentation: Creating a Parallel Stage Pipeline Job in Jenkins**

## **1. Prerequisites**

Before creating the pipeline, ensure you have:

* A running **Jenkins server** (EC2 in this case).
* **Java 17+** installed on Jenkins server.
* **Maven** installed and configured in **Global Tool Configuration**.
* A **SonarQube server** installed and reachable from Jenkins.
* A **Nexus repository** installed and reachable.
* **Tomcat server** running with Manager app enabled.
* **Slack App/Webhook** configured with Jenkins Slack plugin.
* GitHub repo: [hiring-app](https://github.com/imrankhanmohammad257/hiring-app.git?utm_source=chatgpt.com)

## **2. Create a New Pipeline Job**

1. Open Jenkins URL: http://<EC2-IP>:8080.
2. Log in with your Jenkins credentials.
3. Click **“New Item”** (top-left).
4. Enter job name: Hiring-App-Parallel-Pipeline.
5. Select **Pipeline** → Click **OK**.

## **3. Configure Pipeline Job**

### **General Section**

* Add a description:  
   *“This job builds, analyzes, deploys, and notifies for hiring-app using parallel stages.”*

### **Build Triggers (Optional)**

* Choose one:  
  + **Poll SCM** (H/15 \* \* \* \*) → Check GitHub every 15 min.
  + **GitHub Webhook** → Trigger on every push.

### **Pipeline Section**

* **Definition** → Select **Pipeline script**.
* Paste the following script:

**node {**

**// ========================**

**// Stage 1: Checkout SCM**

**// ========================**

**stage('Checkout SCM') {**

**git branch: 'main',**

**url: 'https://github.com/imrankhanmohammad257/hiring-app.git'**

**}**

**// ========================**

**// Stage 2: Build**

**// ========================**

**stage('Build') {**

**def mvnHome = tool name: 'Maven-3.8.4', type: 'maven'**

**sh "${mvnHome}/bin/mvn clean package -DskipTests"**

**}**

**// ========================**

**// Stage 3: Parallel Execution**

**// ========================**

**stage('Parallel Tasks') {**

**parallel(**

**'SonarQube Analysis': {**

**def mvnHome = tool name: 'Maven-3.8.4', type: 'maven'**

**withSonarQubeEnv('SonarQube') {**

**sh "${mvnHome}/bin/mvn sonar:sonar"**

**}**

**},**

**'Deploy to Nexus': {**

**withCredentials([usernamePassword(credentialsId: 'nexus-creds', usernameVariable: 'NEXUS\_USER', passwordVariable: 'NEXUS\_PASS')]) {**

**def mvnHome = tool name: 'Maven-3.8.4', type: 'maven'**

**sh """**

**${mvnHome}/bin/mvn clean deploy -DskipTests \**

**-Dnexus.username=$NEXUS\_USER \**

**-Dnexus.password=$NEXUS\_PASS**

**"""**

**}**

**}**

**)**

**}**

**// ========================**

**// Stage 4: Deploy to Tomcat**

**// ========================**

**stage('Deploy to Tomcat') {**

**withCredentials([usernamePassword(credentialsId: 'tomcat-credentials', usernameVariable: 'TOMCAT\_USER', passwordVariable: 'TOMCAT\_PASS')]) {**

**sh """**

**# Undeploy old app**

**curl -s -u $TOMCAT\_USER:$TOMCAT\_PASS \**

**"http://3.82.42.125:8080/manager/text/undeploy?path=/hiring"**

**# Deploy new WAR**

**curl -s -u $TOMCAT\_USER:$TOMCAT\_PASS \**

**-T target/hiring.war \**

**"http://3.82.42.125:8080/manager/text/deploy?path=/hiring&update=true"**

**"""**

**}**

**}**

**// ========================**

**// Stage 5: Slack Notification**

**// ========================**

**stage('Slack Notification') {**

**slackSend(**

**channel: '#jenkins-integration',**

**color: 'good',**

**message: "Hi Team, Jenkins parallel pipeline job for \*hiring-app\* has finished successfully! ✅\nDeployed by: Imran Khan"**

**)**

**}**

**}**

## **4. Execution Flow**

1. **Checkout SCM**
   * Pulls source code from GitHub branch main.
2. **Build**
   * Compiles and packages the project with Maven.
   * WAR file is created in target/.
3. **Parallel Tasks**
   * **SonarQube Analysis**: Sends code quality report to SonarQube dashboard.
   * **Deploy to Nexus**: Publishes the WAR to Nexus repository.
4. **Deploy to Tomcat**
   * Undeploys old app.
   * Deploys new WAR into Tomcat server.
5. **Slack Notification**
   * Sends success notification to Slack channel.

## **5. Expected Output**

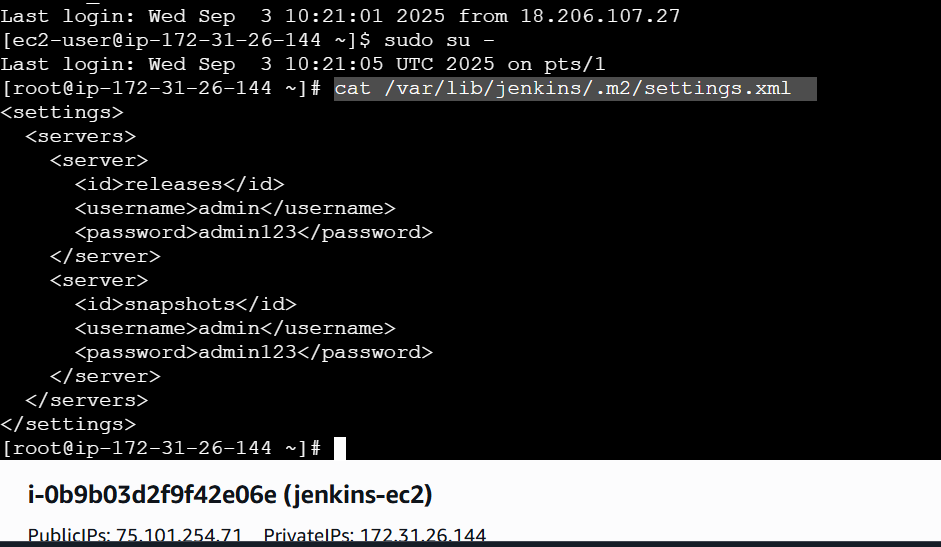
* **Console Output in Jenkins** will show:  
  + Parallel execution of *SonarQube Analysis* and *Deploy to Nexus*.
  + WAR uploaded successfully to Nexus.
  + Application deployed on Tomcat.
  + Slack message delivered to team channel.
* **SonarQube Dashboard**: Code analysis report.
* **Nexus Repository**: New WAR artifact uploaded.
* **Tomcat Server**: Application running at http://<Tomcat-IP>:8080/hiring.
* **Slack Channel**: Notification message sent.

## **6. Advantages of Parallel Stages**

* Saves build time by running independent tasks at the same time.
* Improves CI/CD efficiency.
* Flexible structure: you can add more parallel stages (e.g., run tests, code scanning).

**Created or edited settings.xml in jenkins ec2**

cat /var/lib/jenkins/.m2/settings.xml

****

**Added below content in pom.xml in github repo**

**<distributionManagement>**

**<repository>**

**<id>releases</id>**

**<url>http://54.163.17.174:8081/repository/releases/</url>**

**</repository>**

**<snapshotRepository>**

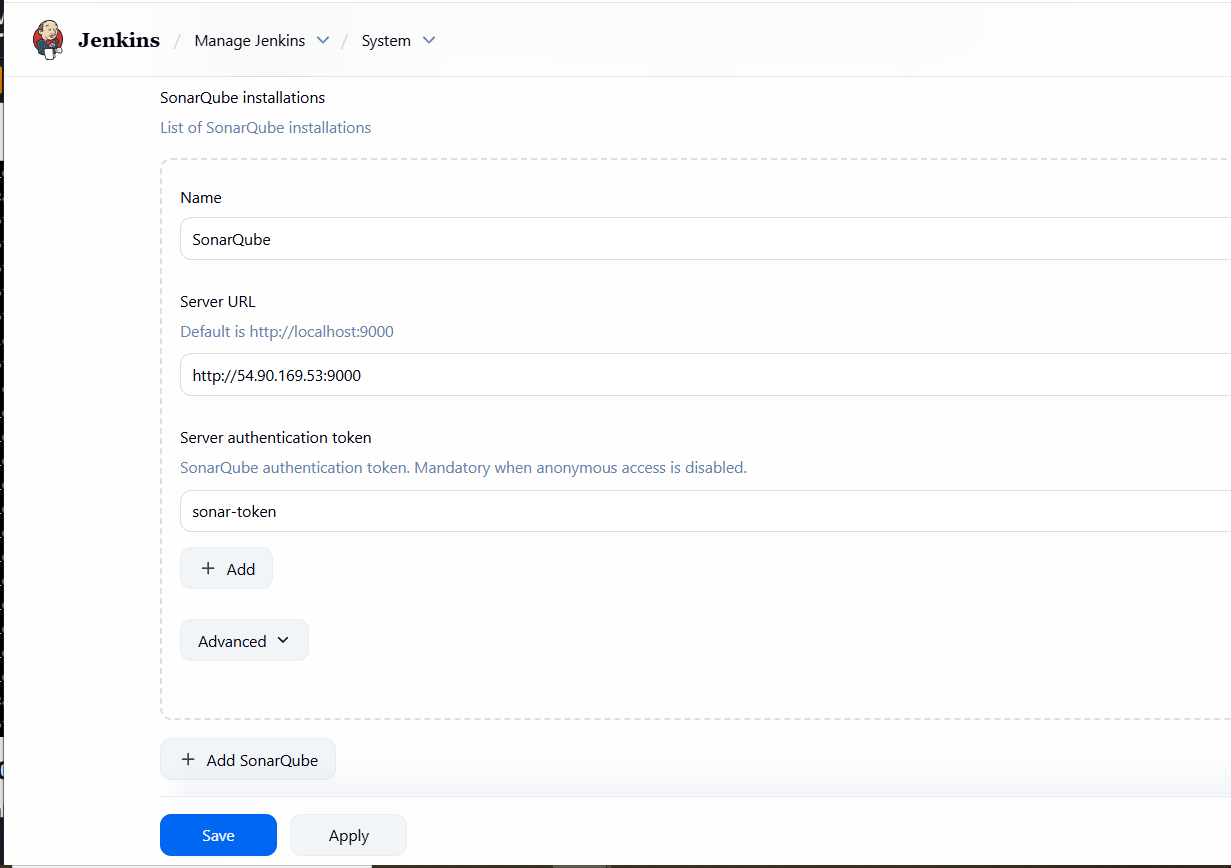
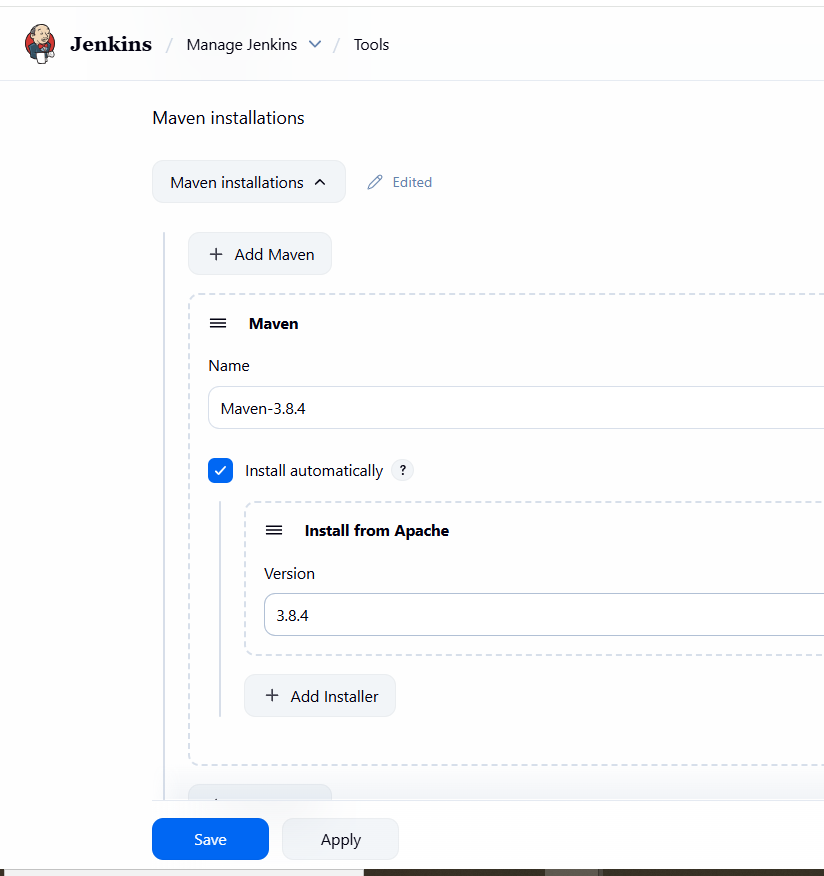
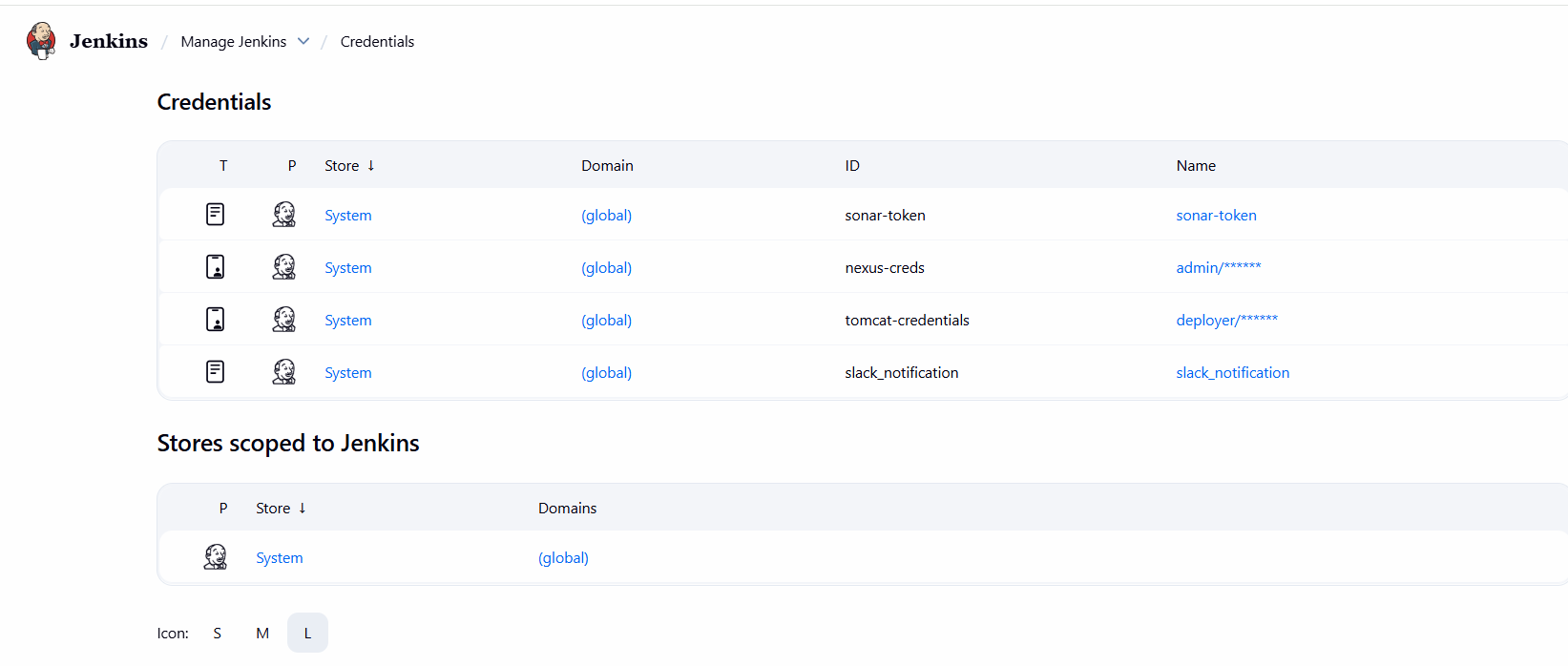
**<id>snapshots</id>**

**<url>http://54.163.17.174:8081/repository/snapshots/</url>**

**</snapshotRepository>**

**</distributionManagement>**

**In JENKINS HOME PAGE:**

1. **SYSTEM**
2. ****
3. **SLACK**
4. ****
5. **TOOLS**
6. ****
7. **CREDENTIALS**
8. ****

# **✅ Required Jenkins Plugins**

### **1. Pipeline Related**

* **Pipeline (workflow-aggregator) → Enables scripted/Declarative pipelines.**
* **Pipeline: GitHub Groovy Libraries → Allows using shared libraries.**
* **Pipeline Utility Steps → Extra pipeline functions (file ops, JSON parsing, etc.).**

### **2. Git Integration**

* **Git Plugin → To fetch code from GitHub repo.**
* **GitHub Plugin → For GitHub webhooks & integration.**

### **3. Build Tools**

* **Maven Integration Plugin → To configure Maven in *Global Tool Configuration*.**

### **4. Code Quality**

* **SonarQube Scanner for Jenkins → Required for withSonarQubeEnv('SonarQube').**

### **5. Artifact Repository**

* **Nexus Artifact Uploader (optional) → You are already using Maven deploy, so not strictly required, but useful.**

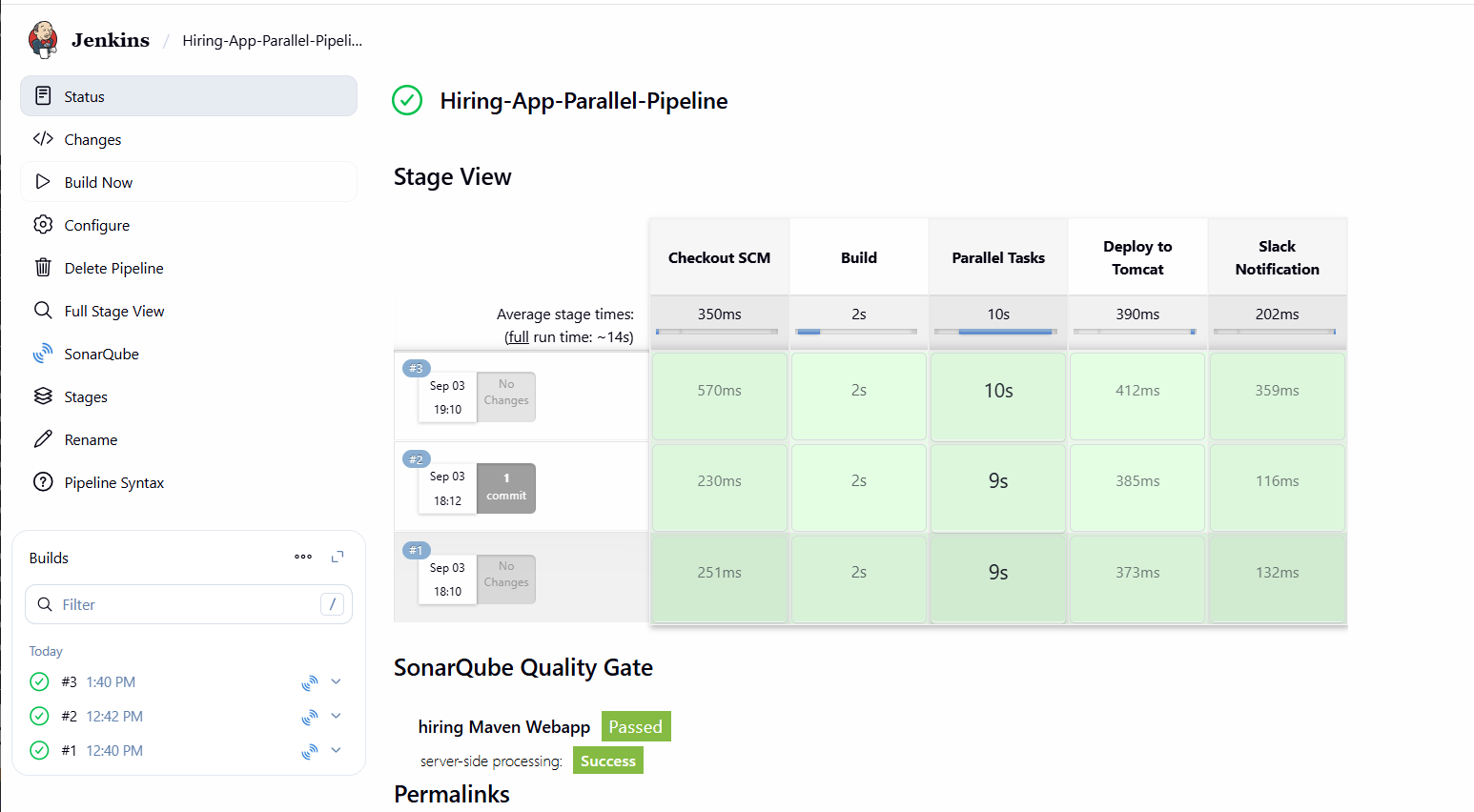
### **6. Deployment**

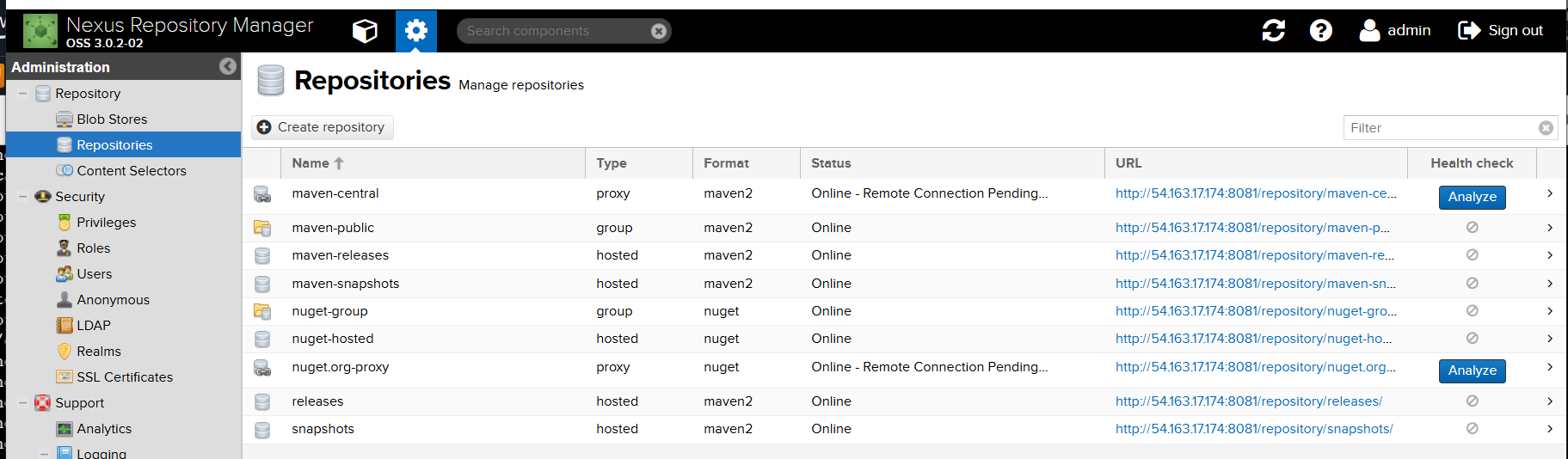
* **Deploy to container Plugin (optional) → You are using curl for Tomcat, so plugin not mandatory.**

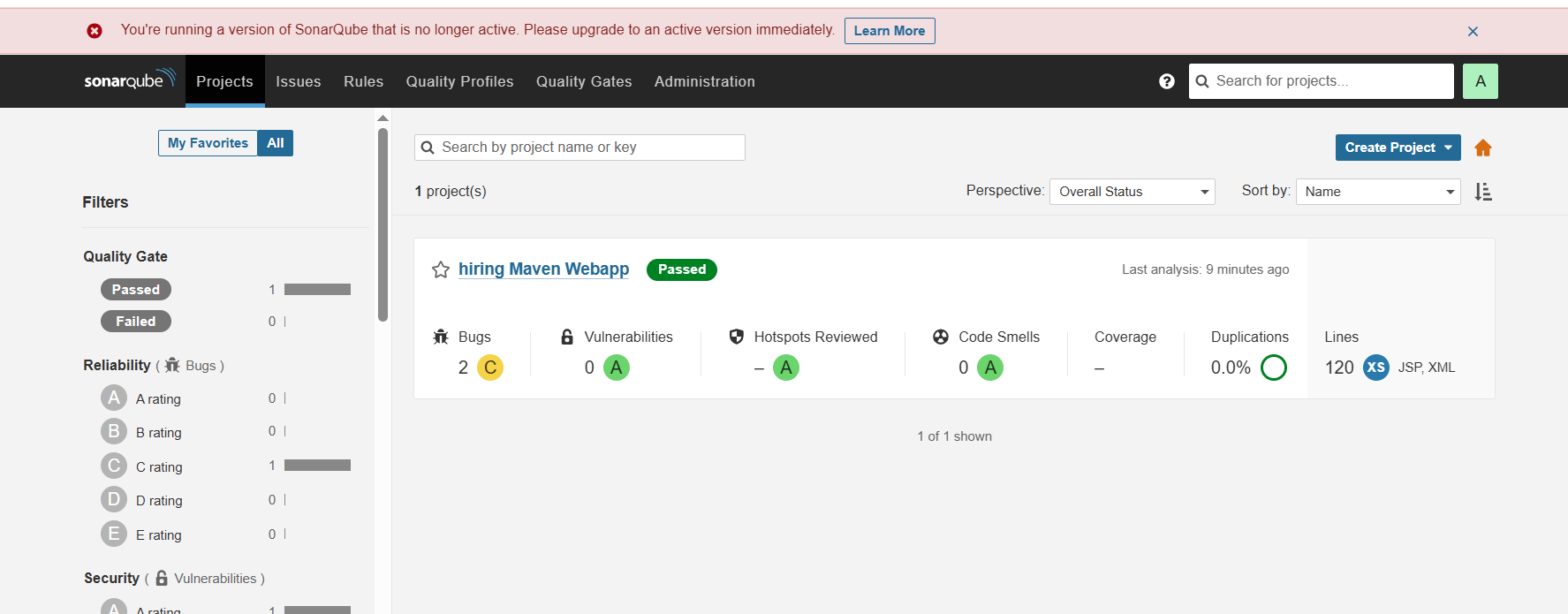
### **7. Notifications**

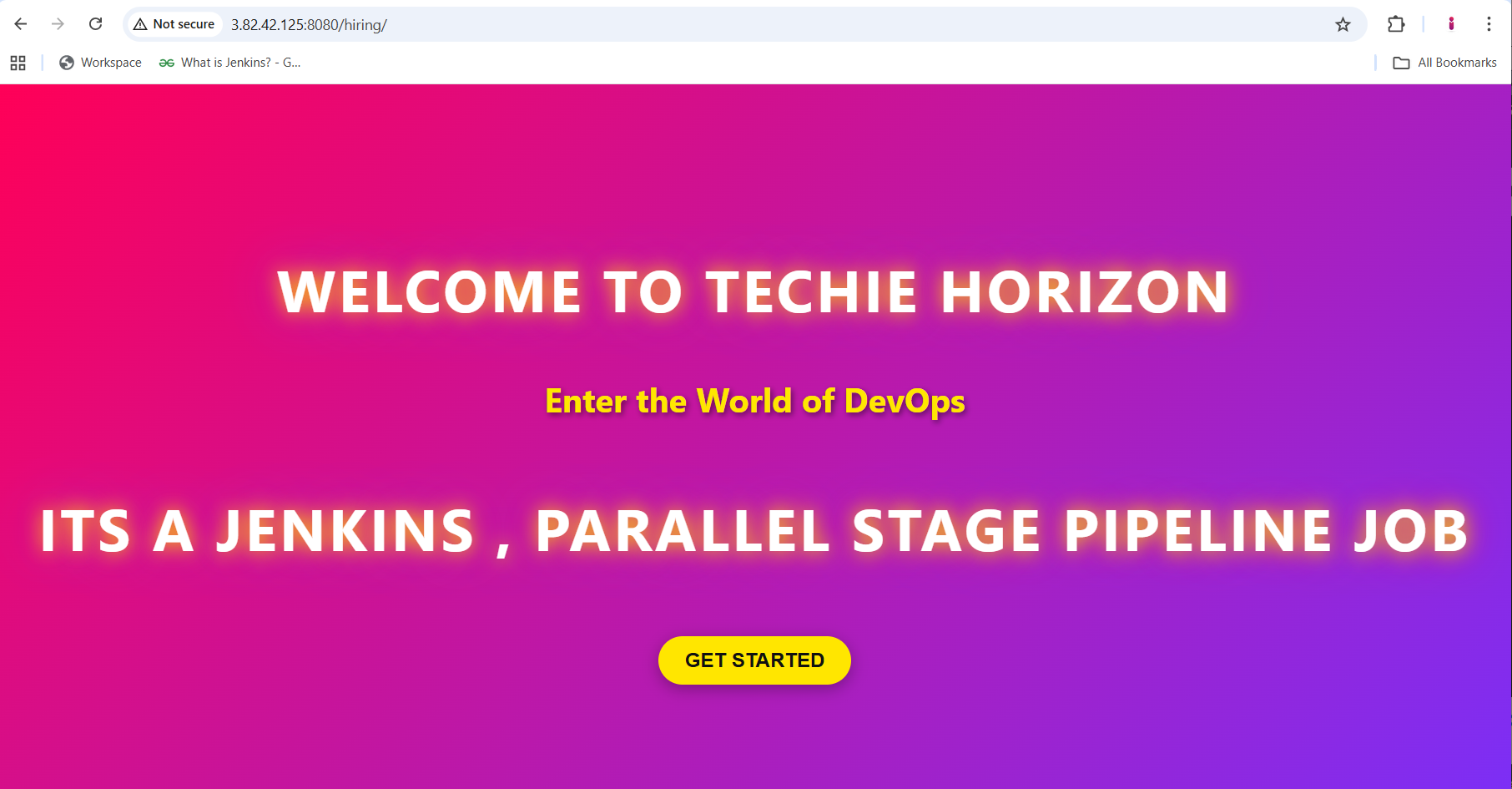
* **Slack Notification Plugin → Required for slackSend(...).**

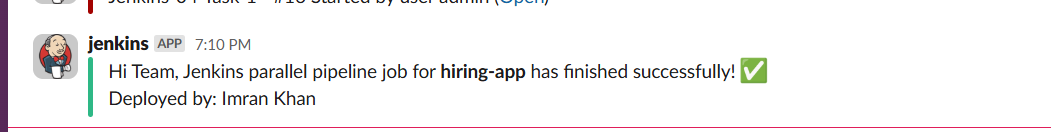
**SNAPS:**

****

****

****

****

****