Java Web Application CI/CD Pipeline Documentation

1. Overview

This document explains how to set up a complete **CI/CD pipeline** for a Java Web Application using **GitHub, Jenkins, SonarQube, Nexus, and Tomcat**.

The process automates:

- Code integration from GitHub
- Static analysis using SonarQube
- Build and packaging using Maven
- Artifact upload to Nexus
- Deployment to Tomcat

CI/CD Architecture: GitHub → Jenkins → SonarQube → Nexus → Tomcat

2. Prerequisites

Before you begin, ensure the following are available:

- GitHub repository containing your Java web application source code.
- Jenkins master server (Ubuntu 22.04 LTS).
- SonarQube, Nexus, and Tomcat servers (Ubuntu 22.04 LTS).
- SSH connectivity between Jenkins master and all agent servers.
- GitHub Personal Access Token (PAT), SonarQube token, Nexus credentials, and Tomcat Manager credentials.

3. Jenkins Setup (Detailed)

Step 1: Install Java

```
sudo apt update && sudo apt install -y openjdk-17-jdk
```

Step 2: Install Jenkins

```
sudo apt install fontconfig openjdk-21-jre

java -version

sudo wget -O /etc/apt/keyrings/jenkins-keyring.asc \

  https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key

echo "deb [signed-by=/etc/apt/keyrings/jenkins-keyring.asc]" \
```

https://pkg.jenkins.io/debian-stable binary/ | sudo tee \

/etc/apt/sources.list.d/jenkins.list > /dev/null

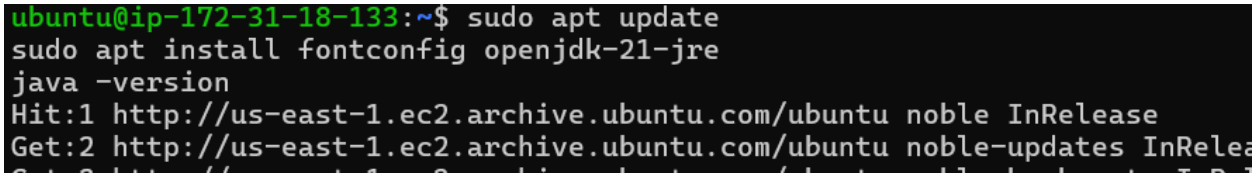
sudo apt update

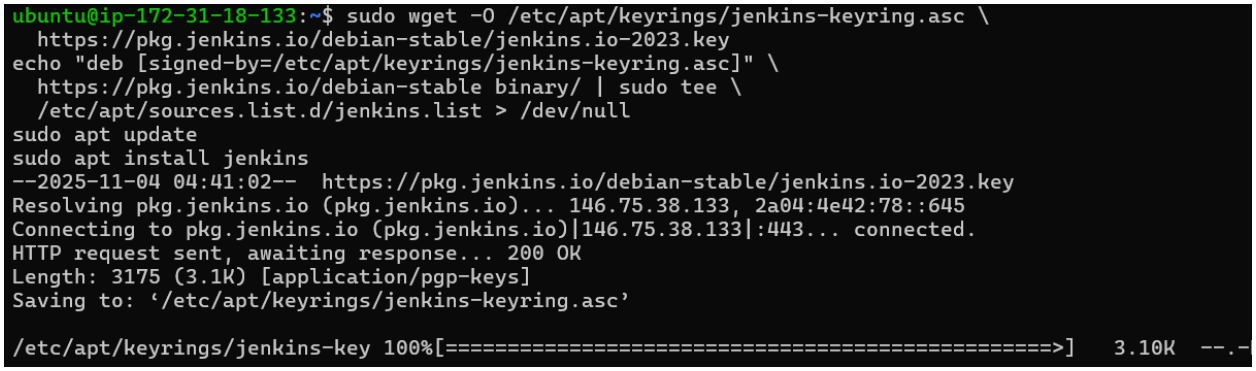
sudo apt install jenkins

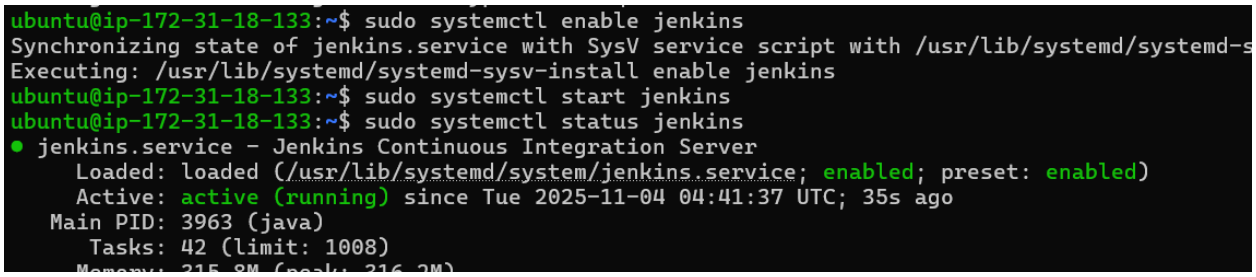
sudo systemctl enable jenkins

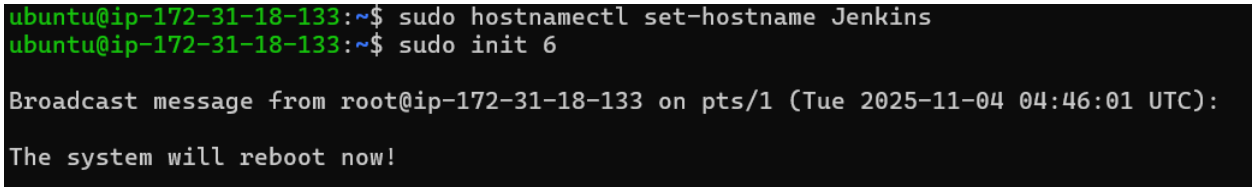
sudo systemctl start jenkins

sudo systemctl status jenkins







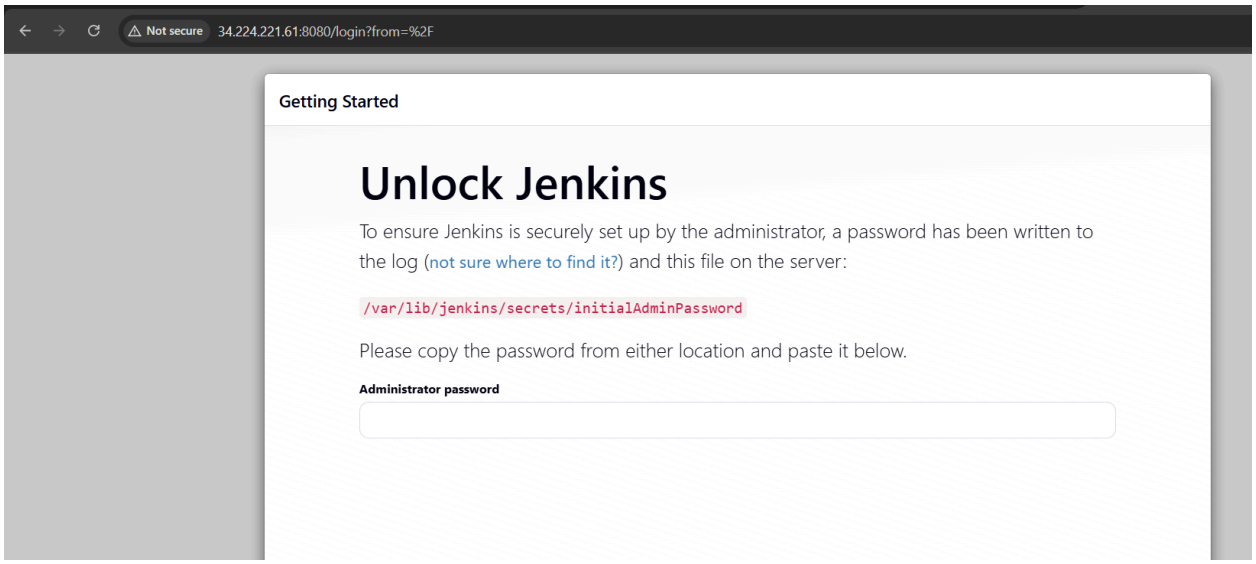


Step 3: Access Jenkins

Open in browser → http://<jenkins-ip>:8080

Get unlock password:

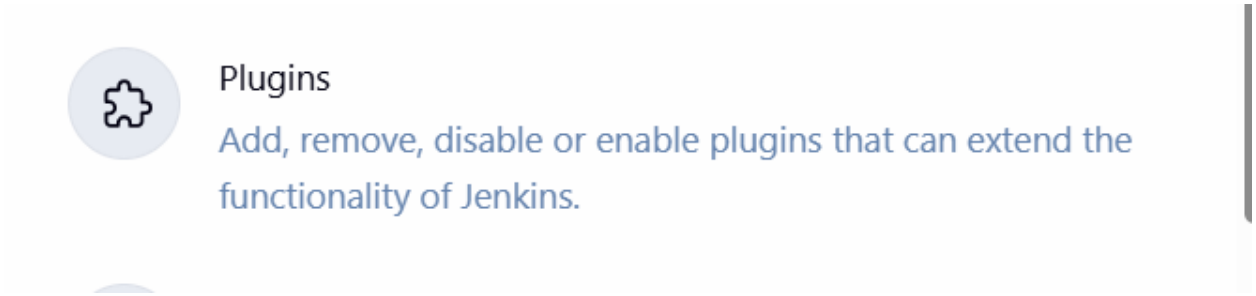
sudo cat /var/lib/jenkins/secrets/initialAdminPassword

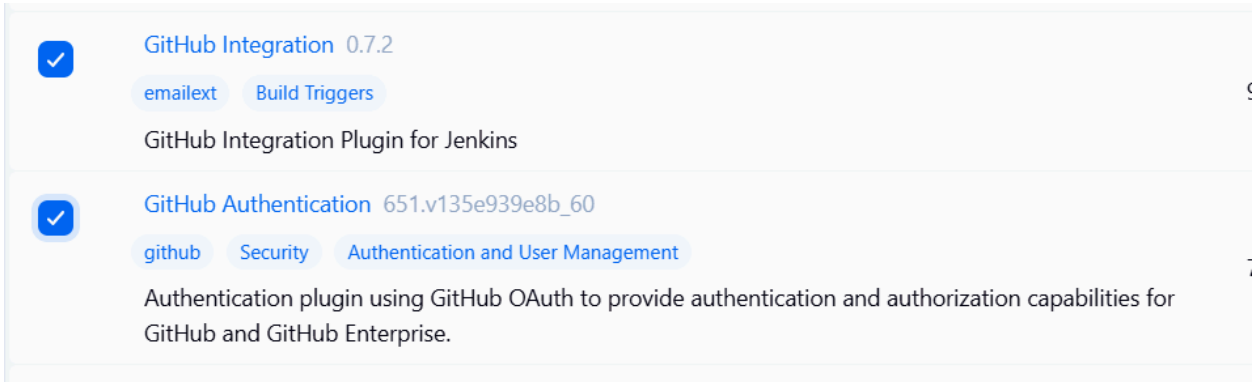



```
ubuntu@Jenkins:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
0a8d7f48882d453bb973bbeb418dd9bd
ubuntu@Jenkins:~$
```

Step 4: Install Plugins

Click **“Plugins”** after the login.





Generic Webhook Trigger 2.4.1

notification github webhook Build Parameters gitlab Build Triggers bitbucket bitbucket-server jira

Can receive any HTTP request, extract any values from JSON or XML and trigger a job with those values available as variables. Works with GitHub, GitLab, Bitbucket, Jira and many more.

Maven Integration 3.27

Build Tools

This plugin provides a deep integration between Jenkins and Maven. It adds support for automatic triggers between projects depending on SNAPSHOTs as well as the automated configuration of variou Jenkins publishers such as Junit.

Pipeline Maven Integration 1567.vb_2c3a_2116860

pipeline Maven

This plugin provides integration with Pipeline, configures maven environment to use within a pipeline job by calling sh mvn or bat mvn. The selected maven installation will be configured and prepended to the path.

SonarQube Scanner 2.18

External Site/Tool Integrations Build Reports

This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.

Sonar Quality Gates 352.vdcdb_d7994fb_6

Library plugins (for use by other plugins) analysis Other Post-Build Actions

Fails the build whenever the Quality Gates criteria in the Sonar 5.6+ analysis aren't met (the project Quality Gates status is different than "Passed")

Quality Gates 2.5

Fails the build whenever the Quality Gates criteria in the Sonar analysis aren't met (the project Quality Gates status is different than "Passed")

Warning: This plugin version may not be safe to use. Please review the following security notices:

- Credentials transmitted in plain text

Artifactory 4.0.8

pipeline

This plugin allows your build jobs to deploy artifacts and resolve dependencies to and from Artifactory, and then have them linked to the build job that created them. The plugin includes a vast collection of features, including a rich pipeline API library and release management for Maven and Gradle builds with Staging and Promotion.

Deploy to container 1.17

Nexus Artifact Uploader 2.14

Artifact Uploaders

This plugin to upload the artifact to Nexus Repository.

This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.

Artifact Deployer 1.3

Artifact Uploaders

This plugin makes it possible to deploy artifacts from workspace to output directories.

This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.

SSH 158.ve2a_e90fb_7319

Build Wrappers

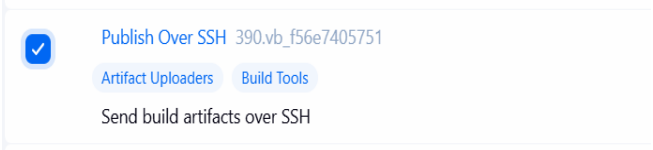
This plugin executes shell commands remotely using SSH protocol.

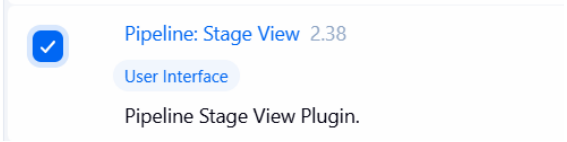
Warning: This plugin version may not be safe to use. Please review the following security notices:

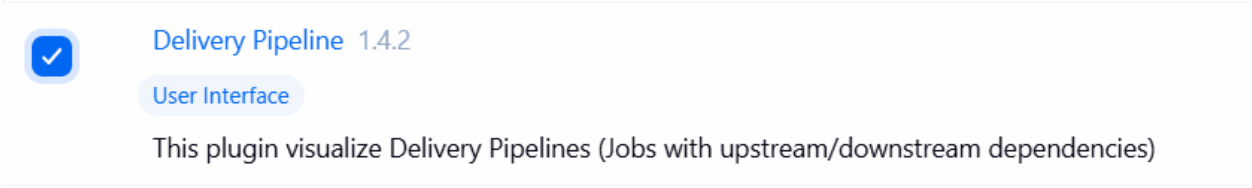
- CSRF vulnerability and missing permission checks allow capturing credentials
- Missing permission check allows enumerating credentials IDs

SSH Agent 386.v36cc0c7582f0

This plugin allows you to provide SSH credentials to builds via a ssh-agent in Jenkins.



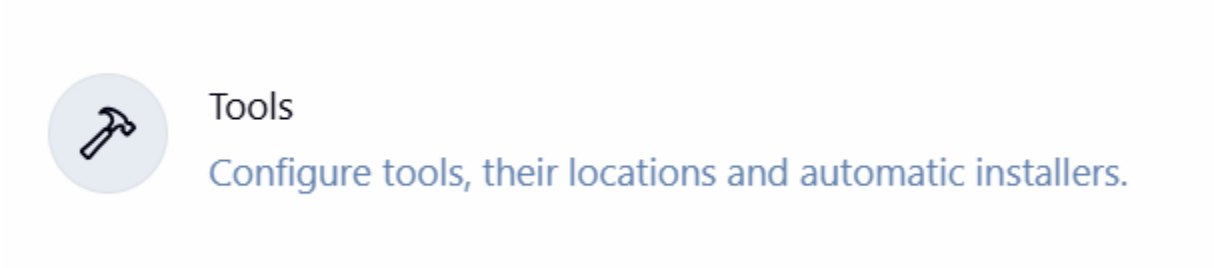




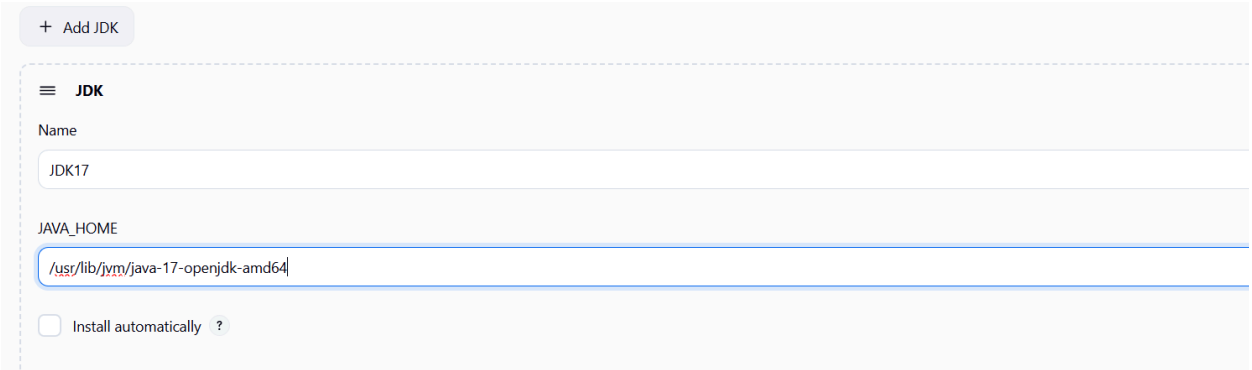
Step 5: Configure Global Tools

Go to **Manage Jenkins → Global Tool Configuration**

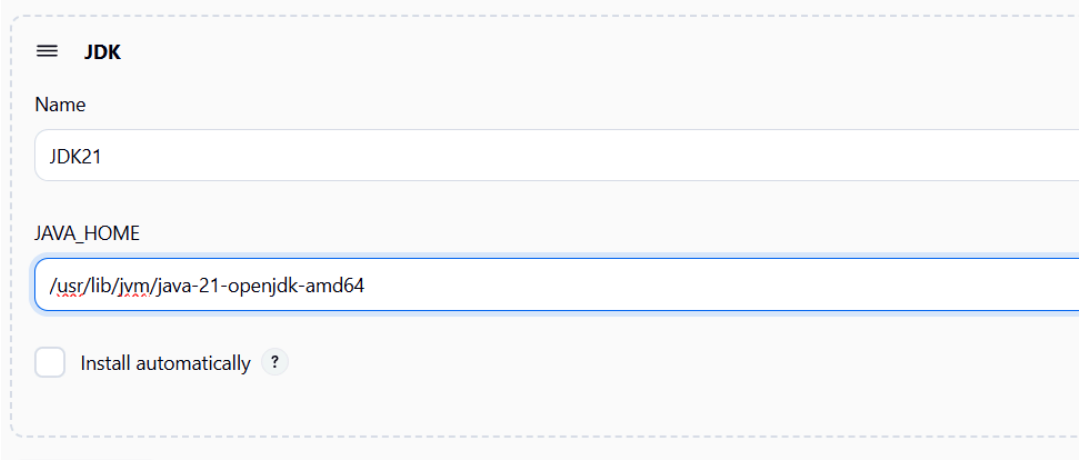
- Add JDK (Name: JDK17)
- Add JDK (Name: JDK21)
- Add Maven (Name: Maven)



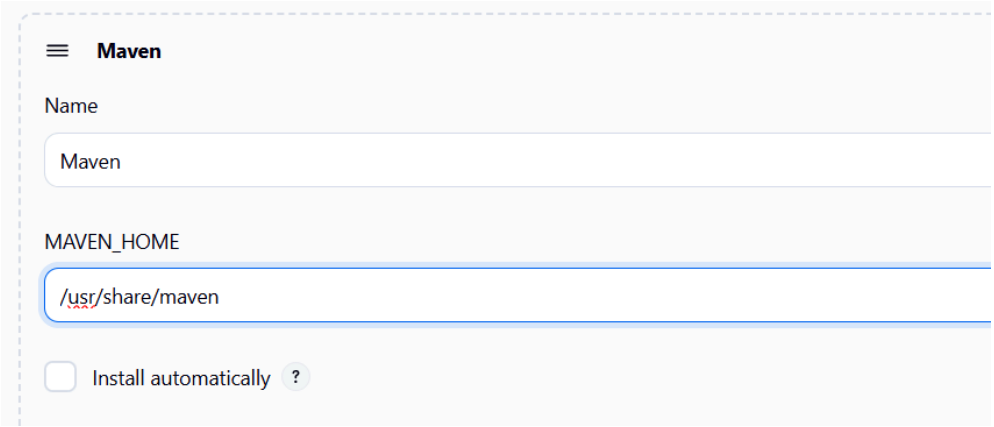

```
ubuntu@SonarQube:~$ readlink -f $(which java)
/usr/lib/jvm/java-17-openjdk-amd64/bin/java
ubuntu@SonarQube:~$
```




```
ubuntu@Jenkins:~$ readlink -f $(which java)
/usr/lib/jvm/java-21-openjdk-amd64/bin/java
ubuntu@Jenkins:~$
```




```
ubuntu@SonarQube:~$  readlink -f $(which mvn)
/usr/share/maven/bin/mvn
ubuntu@SonarQube:~$
```



Step 6: Create SSH Keys for Agents

Sudo su jenkins

ssh-keygen

Copy the public key into each agent server under:

~/.ssh/authorized_keys

Try to connect to the servers from the jenkins server:

Ssh ubuntu@publicip-of-the-agent-server

```
ubuntu@Jenkins:~$ sudo su jenkins
jenkins@Jenkins:/home/ubuntu$ cd
jenkins@Jenkins:~$ ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (/var/lib/jenkins/.ssh/id_ed25519):
Created directory '/var/lib/jenkins/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /var/lib/jenkins/.ssh/id_ed25519
Your public key has been saved in /var/lib/jenkins/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:Em+Lsswn1vilUO7YQt9a24lFEeIiwbxNBhTcoNYaIDg jenkins@Jenkins
The key's randomart image is:
+--[ED25519 256]--+
|+  =*=  . .       |
|E. o+.+. . .      |
| .+ o=o . .       |
| . 8...+  .       |
```

SonarQube:

```
  GNU nano 7.2                                    authorized_keys *
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDaRu0y1AEdXFKMcoYSjQ+QRSQhgDVEDyqP6m3PCpHCu3wdxtzTNsNDe1gP5L4sZYjNjISQ8SHlkvrSJjd>

ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIGvAupKMk3BO4B/iH+zY/PmVPtyjzGAShPSVRcL7OBOz jenkins@Jenkins
```

```
jenkins@Jenkins:~$ ssh ubuntu@18.208.131.176
The authenticity of host '18.208.131.176 (18.208.131.176)' can't be established.
ED25519 key fingerprint is SHA256:wgB1/LMd7ycwrO39wSHHfBJCMOVbyAzKXoiYNMhh7DU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '18.208.131.176' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1015-aws x86_64)
```

```
ubuntu@SonarQube:~$ exit
logout
Connection to 18.208.131.176 closed.
jenkins@Jenkins:~$
```

Tomcat:

```
jenkins@Jenkins:~$ ssh ubuntu@98.91.17.31
The authenticity of host '98.91.17.31 (98.91.17.31)' can't be established.
ED25519 key fingerprint is SHA256:D/IAnan7cjP0pmgEbu/tPZlwIgR27Rpk6MfqS0nZD5o.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '98.91.17.31' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1015-aws x86_64)

 * Documentation:  https://help.ubuntu.com
```

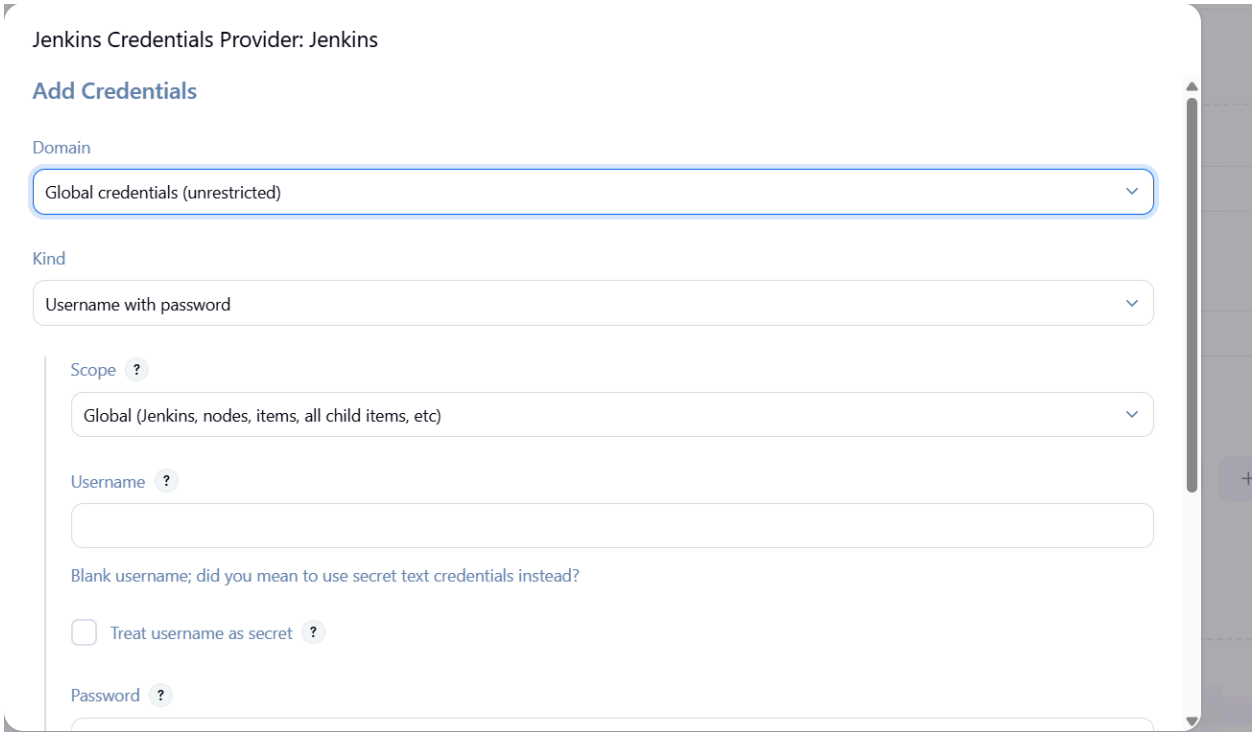
```
ubuntu@ip-172-31-23-240:~$ exit
logout
Connection to 98.91.17.31 closed.
jenkins@Jenkins:~$
```

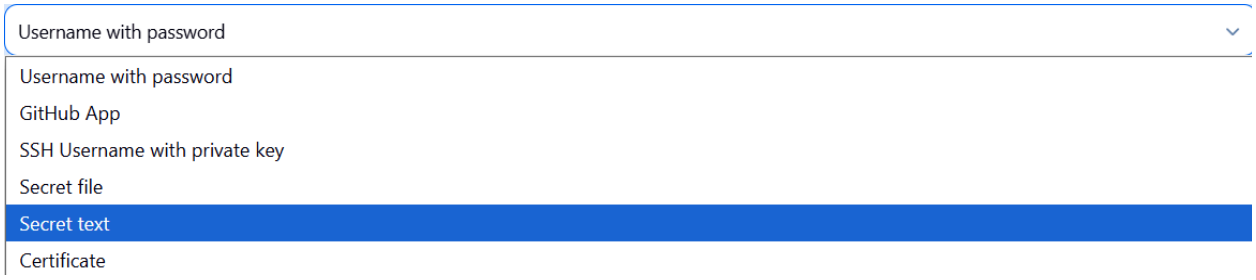
Step 7: Add Credentials in Jenkins

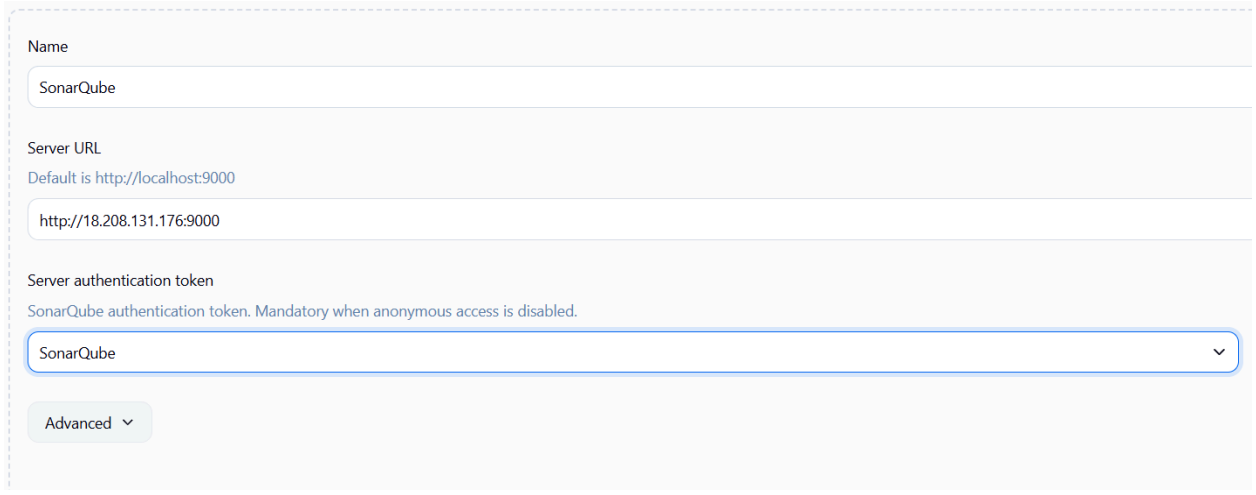
Go to **Manage Jenkins → Credentials → System → Global → Add Credentials**

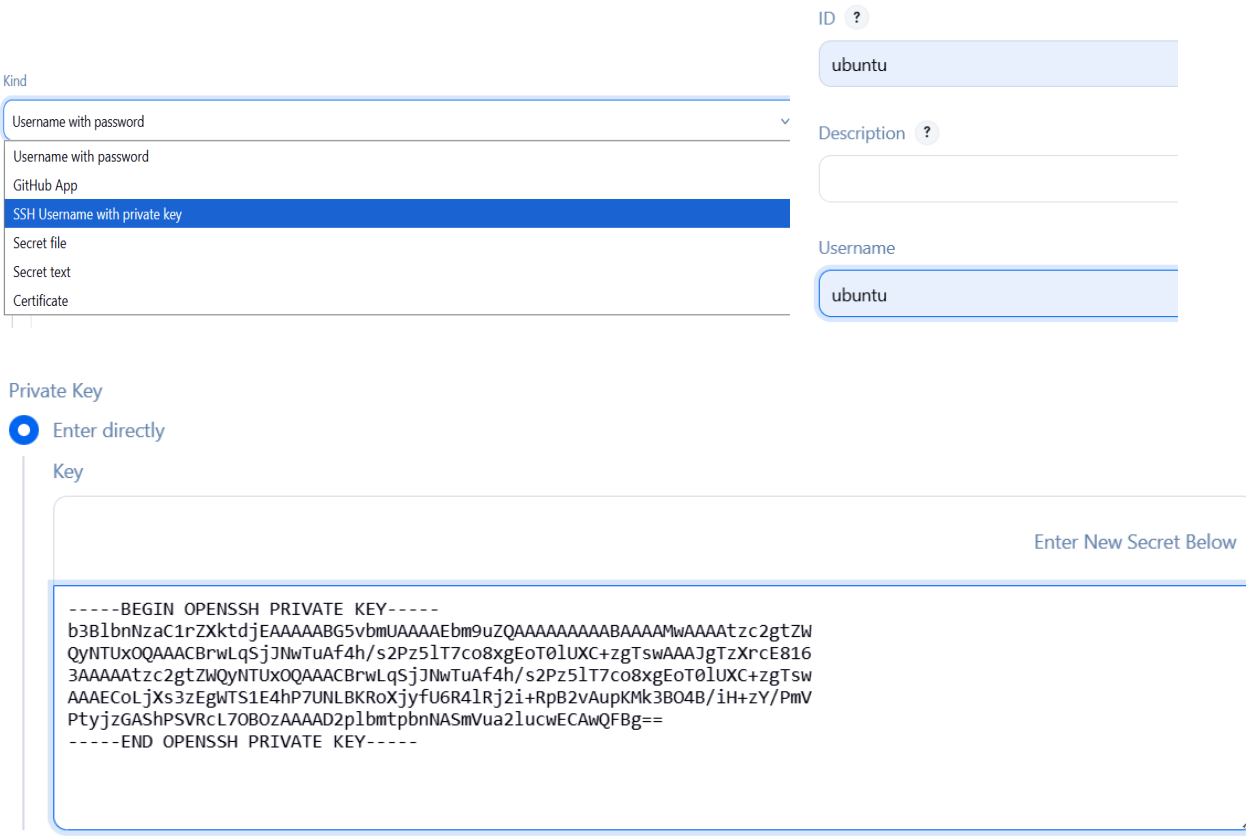
Create the following:

- ssh-ubuntu → SSH Username with private key
- nexus-creds → Username & password for Nexus
- tomcat-manager → Tomcat manager credentials
- github-token → GitHub PAT


Jenkins Credentials Provider: Jenkins
Add Credentials
Domain
Global credentials (unrestricted)
Kind
Username with password
Scope
Global (Jenkins, nodes, items, all child items, etc)
Username
Blank username; did you mean to use secret text credentials instead?
Treat username as secret
Password


Username with password
Username with password
GitHub App
SSH Username with private key
Secret file
Secret text
Certificate


Name
SonarQube
Server URL
Default is http://localhost:9000
http://18.208.131.176:9000
Server authentication token
SonarQube authentication token. Mandatory when anonymous access is disabled.
SonarQube
Advanced

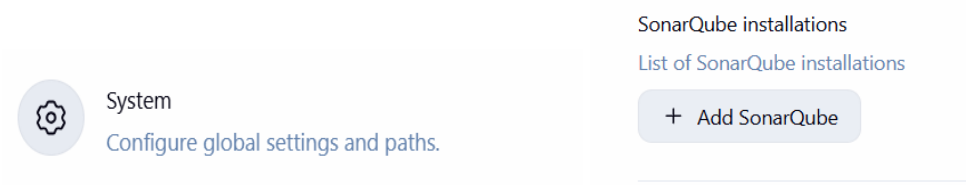


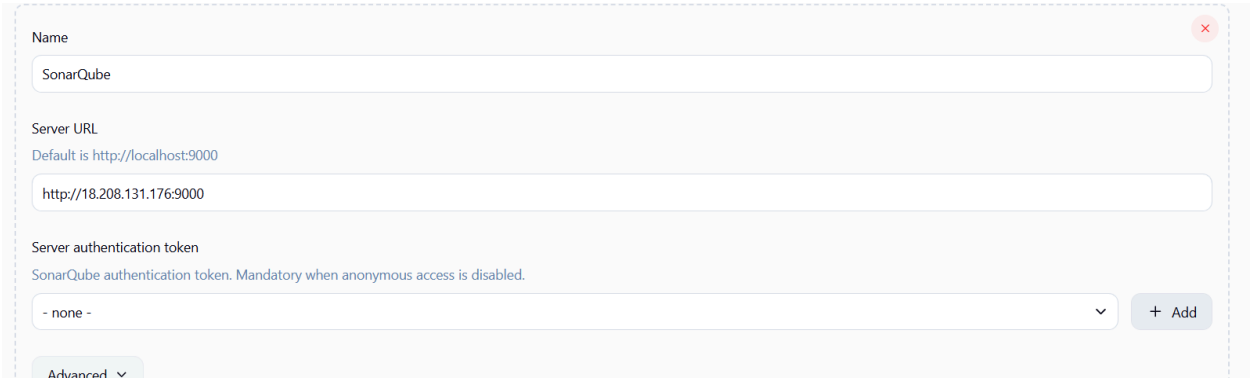
Step 8: Configure SonarQube in Jenkins

Go to **Manage Jenkins → Configure System → SonarQube servers**

Add:

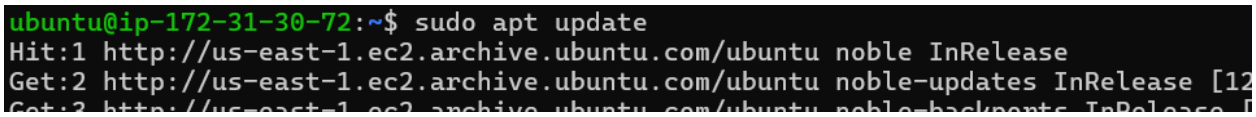
- Name: SonarQube
- Server URL: http://<sonar-ip>:9000
- Token: paste from SonarQube UI

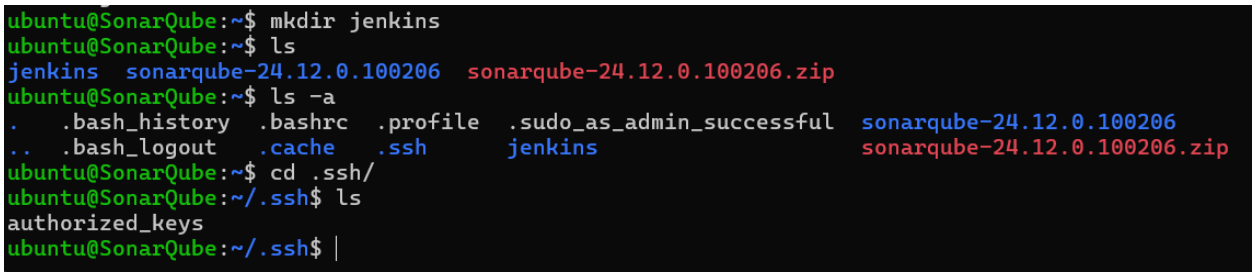


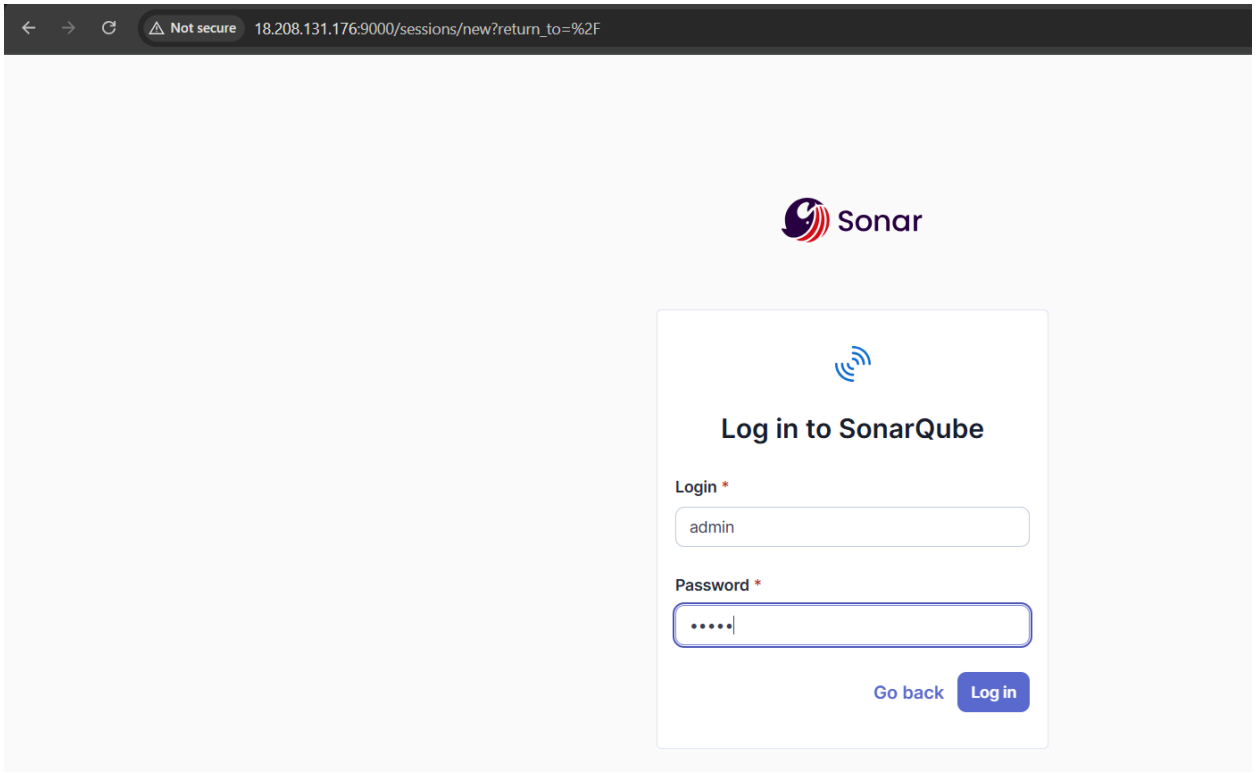


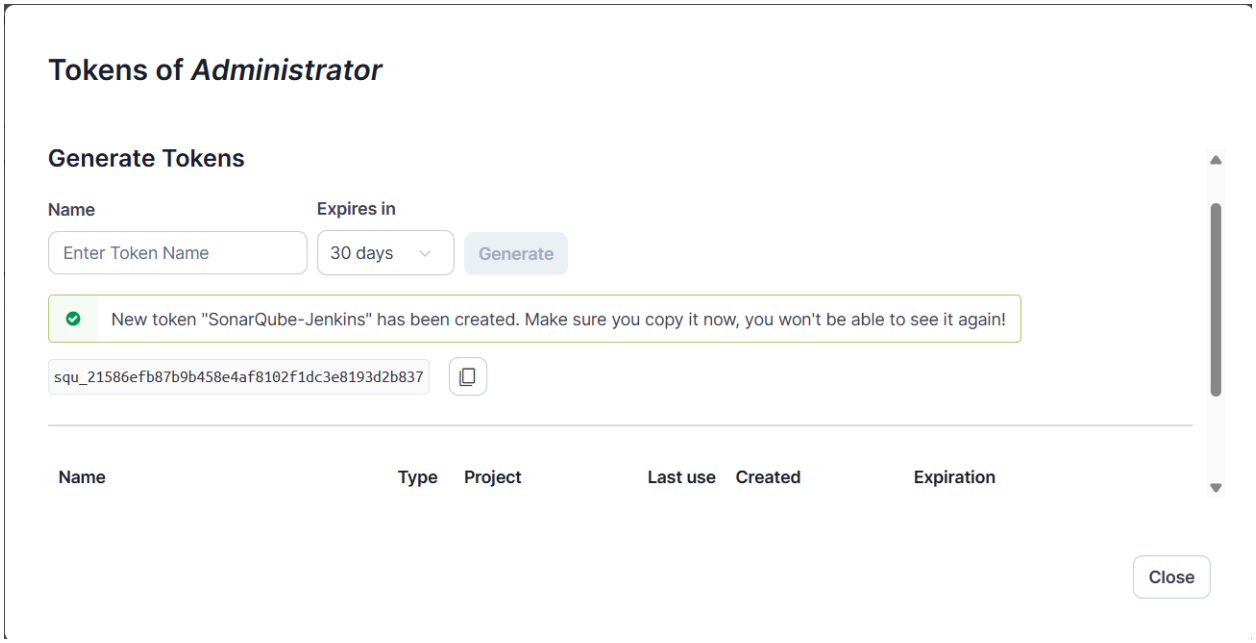
4. SonarQube Setup (Summary)

- Install SonarQube with Java 17.
- Create a jenkins directory so that jenkins can use that as the workspace.
- Extract the package.
- Access SonarQube at http://<sonar-ip>:9000.
- Login with admin/admin, create a new token, and use it in Jenkins.









5. Nexus Setup (Summary)

- Install Nexus Repository OSS with Java 17.
- Start the service and access http://<nexus-ip>:8081.
- Login with admin credentials and create a hosted repository named maven-releases.
- Note the repository URL for your Maven settings.

6. Tomcat Setup (Summary)

- Install Tomcat9 and Tomcat9-admin:

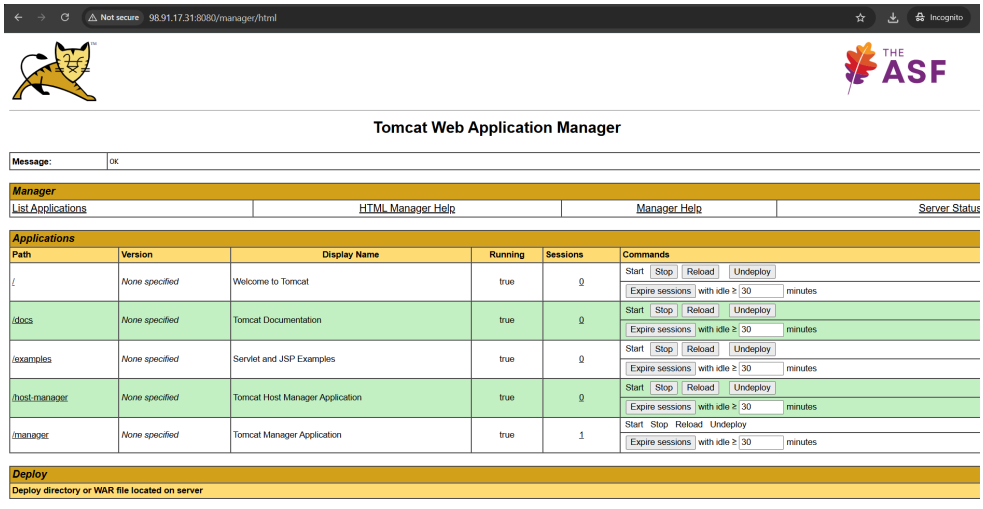
  sudo apt install -y tomcat9 tomcat9-admin

- Add the manager user to /etc/tomcat9/tomcat-users.xml.
- Restart Tomcat and test http://<tomcat-ip>:8080/manager.
- Create a jenkins directory so that jenkins can use that as the workspace.

```
ubuntu@ip-172-31-30-72:~$ sudo apt update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [12
```

```
-->
<role rolename="manager-gui"/>
<role rolename="manager-script"/>
<user username="tomcat" password="s3cret" roles="manager-gui"/>
</tomcat-users>
```

```
ubuntu@ip-172-31-23-240:~$ mkdir jenkins
ubuntu@ip-172-31-23-240:~$ ls
apache-tomcat-9.0.111  apache-tomcat-9.0.111.tar.gz  jenkins
ubuntu@ip-172-31-23-240:~$ cd .ssh/
ubuntu@ip-172-31-23-240:~/.ssh$ ls
authorized_keys
ubuntu@ip-172-31-23-240:~/.ssh$ sudo nano authorized_keys
ubuntu@ip-172-31-23-240:~/.ssh$
```

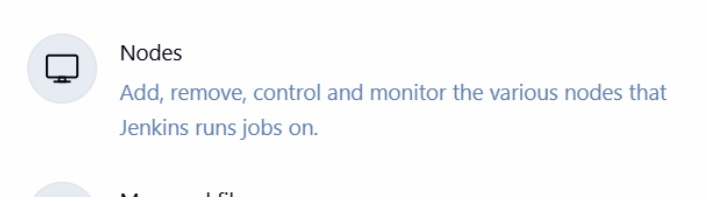


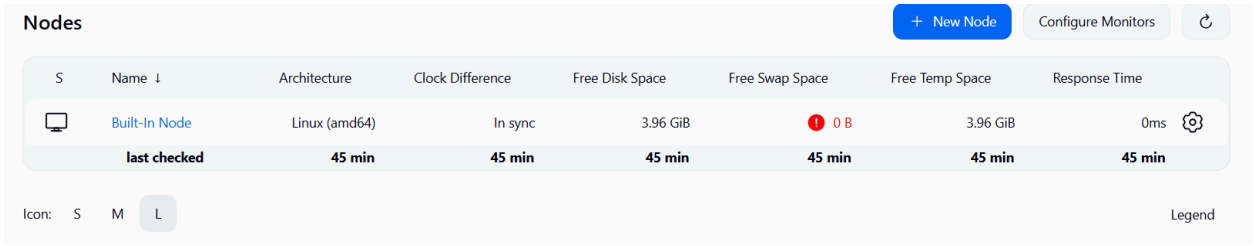
7.Jenkins Nodes configuration

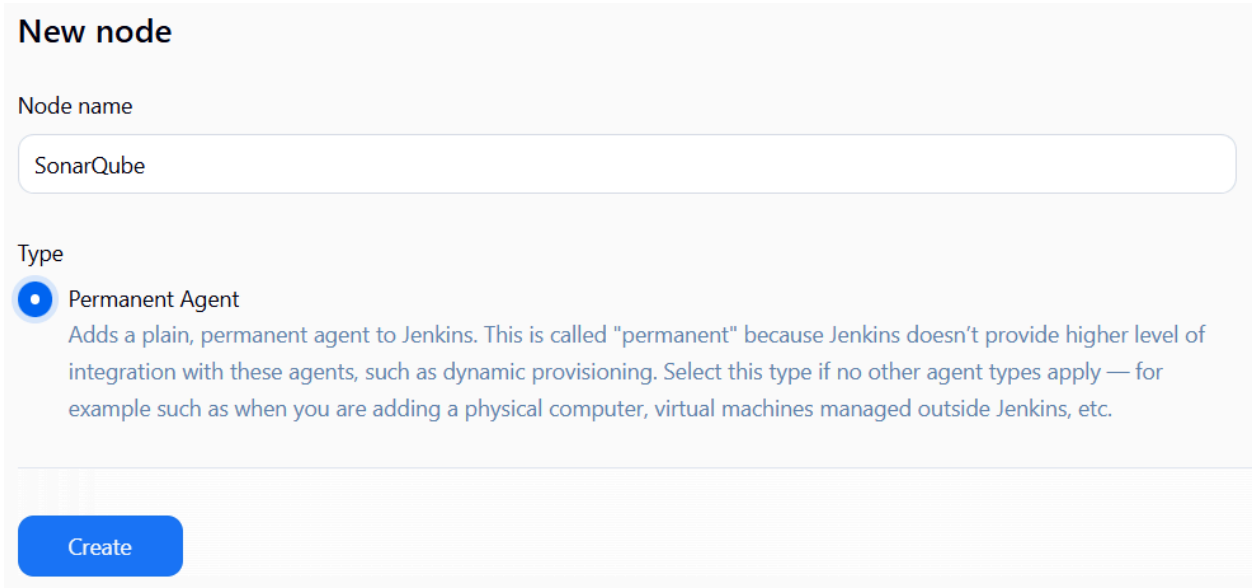
Go to **Manage Jenkins → Configure Nodes → New Nodes**

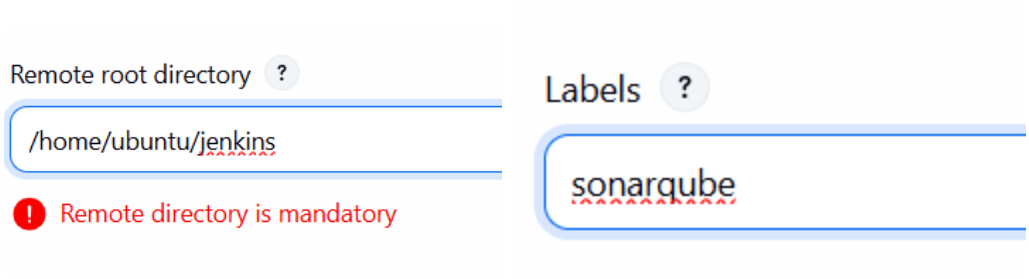
Add SonarQube:

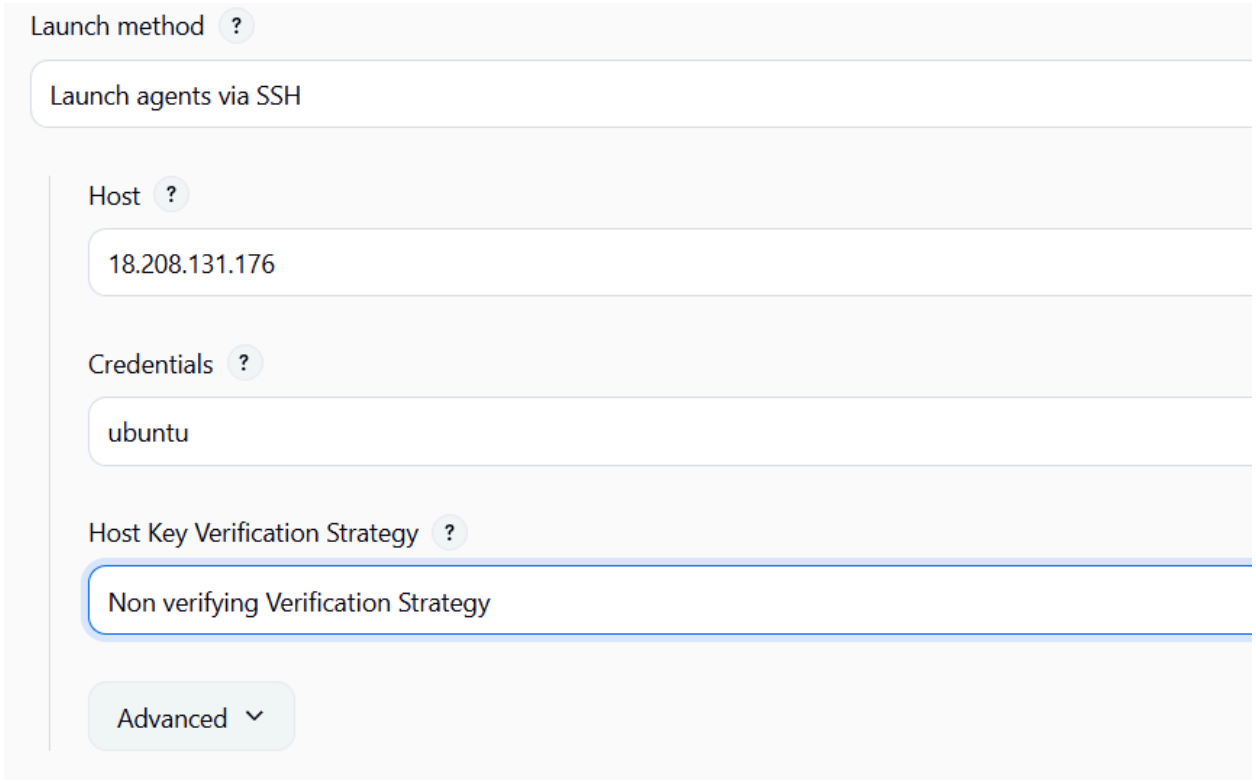
- Name: SonarQube
- Type: Permanent Agent
- Remote root directory: /home/ubuntu/jenkins
- Labels: sonarqube
- Launch method: Launch agents via SSH
- Host: sonar-Public ip
- Credentials: ubuntu or your own id name
- Host key Verification strategy: Non verifying Verification Strategy.

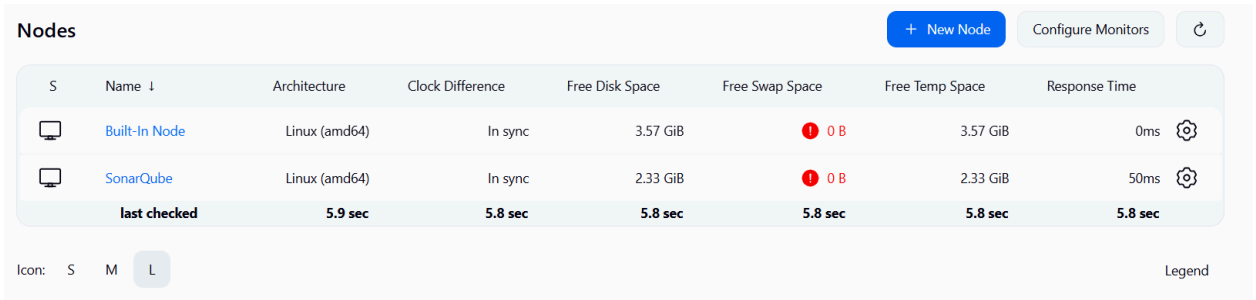








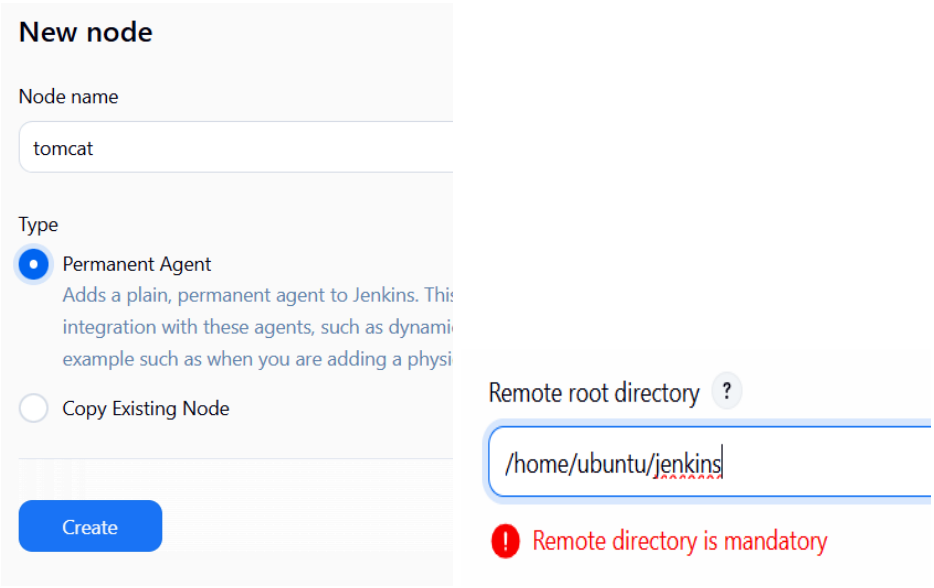


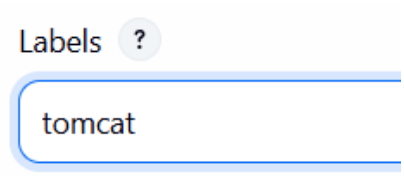


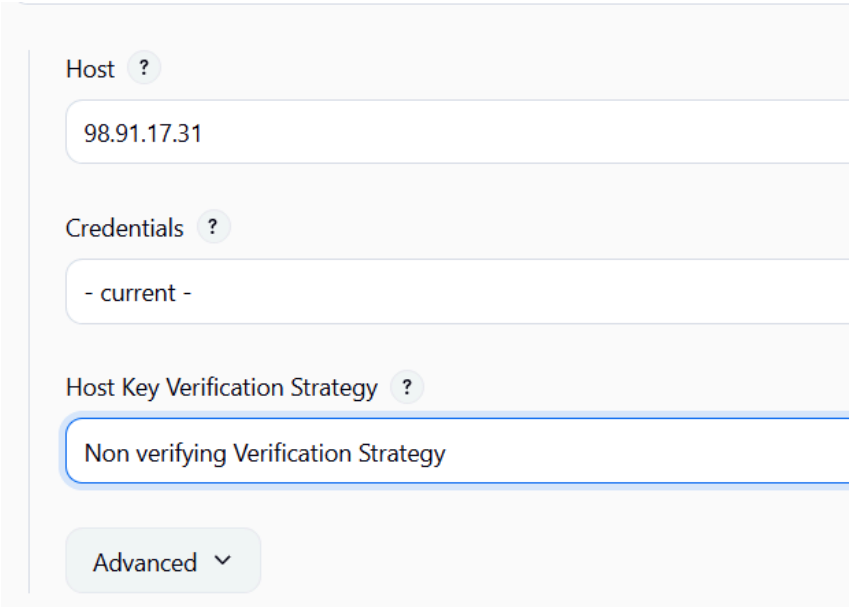
Add Tomcat:

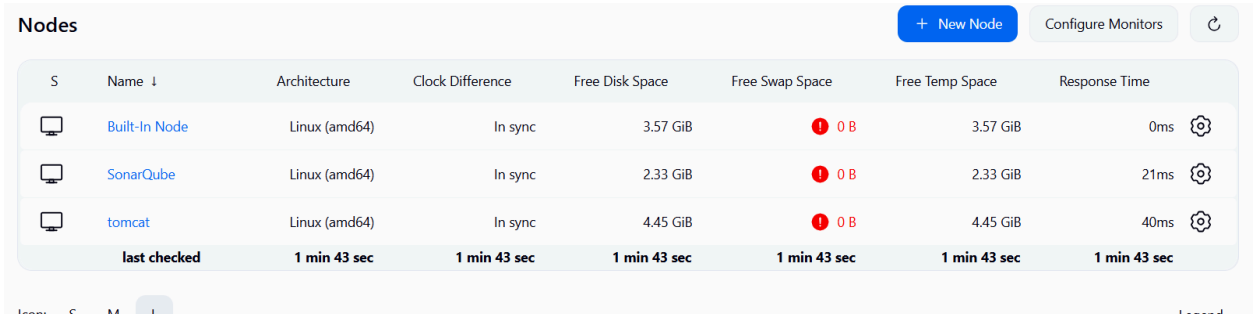
- Name: tomcat
- Type: Permanent Agent
- Remote root directory: /home/ubuntu/jenkins
- Labels: tomcat
- Launch method: Launch agents via SSH

- Host: tomcat-Public ip
- Credentials: ubuntu or your own id name
- Host key Verification strategy: Non verifying Verification Strategy.



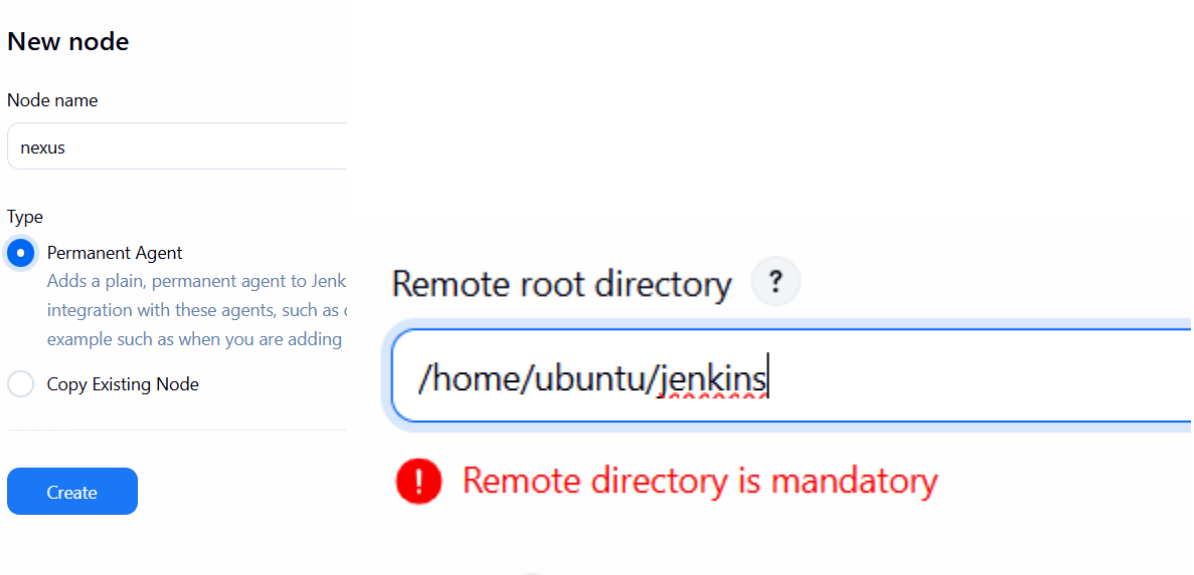


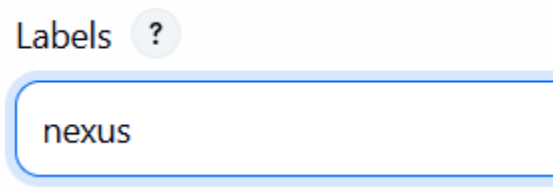


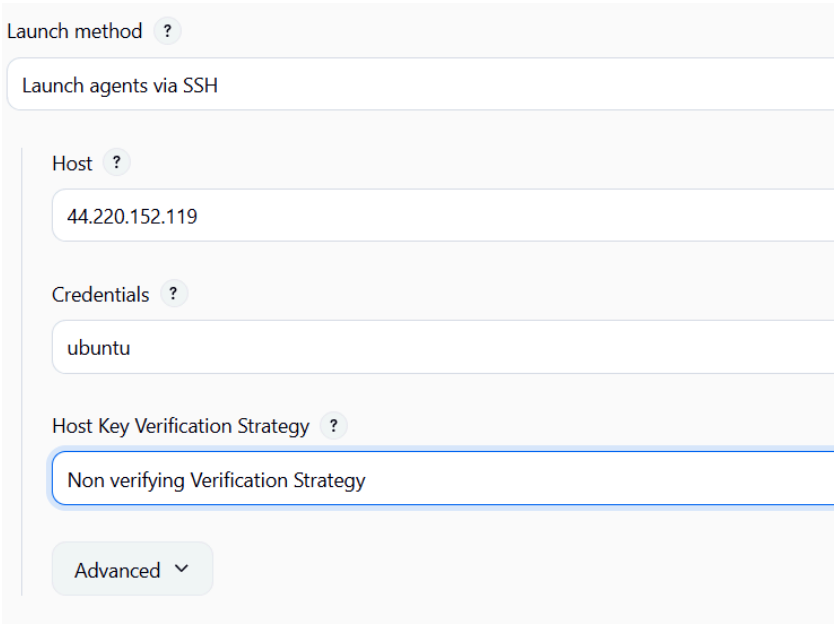


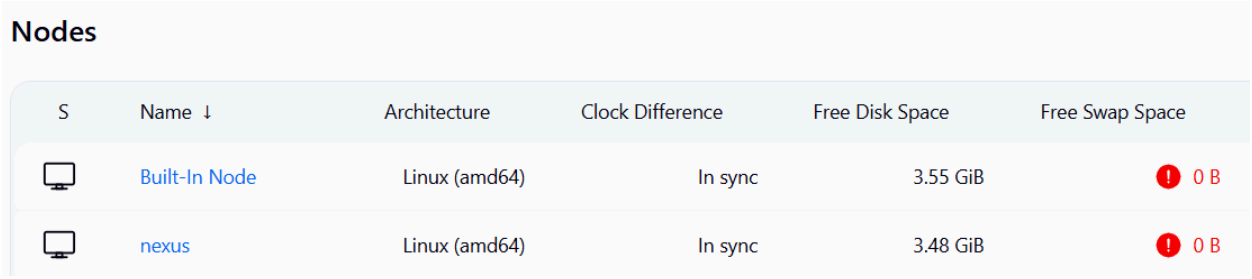
Add Nexus:

- Name: nexus
- Type: Permanent Agent
- Remote root directory: /home/ubuntu/jenkins
- Labels: nexus
- Launch method: Launch agents via SSH
- Host: nexus-Public ip
- Credentials: ubuntu or your own id name
- Host key Verification strategy: Non verifying Verification Strategy.

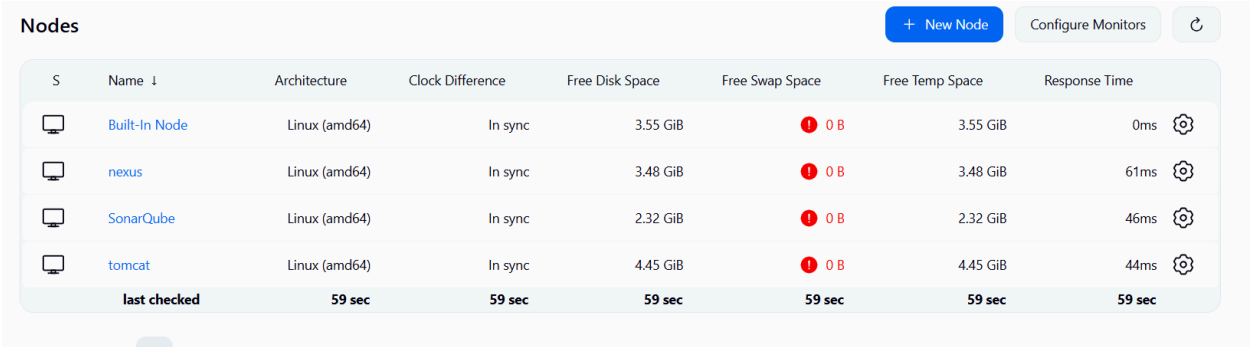








After building all the nodes this is how your Nodes should look line:

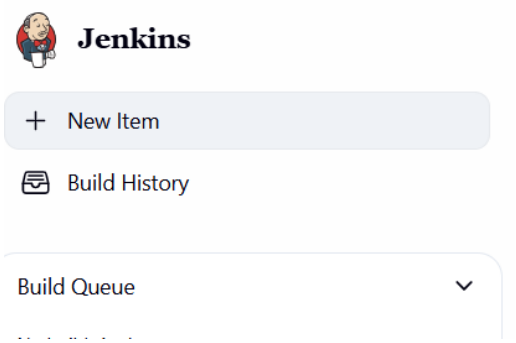


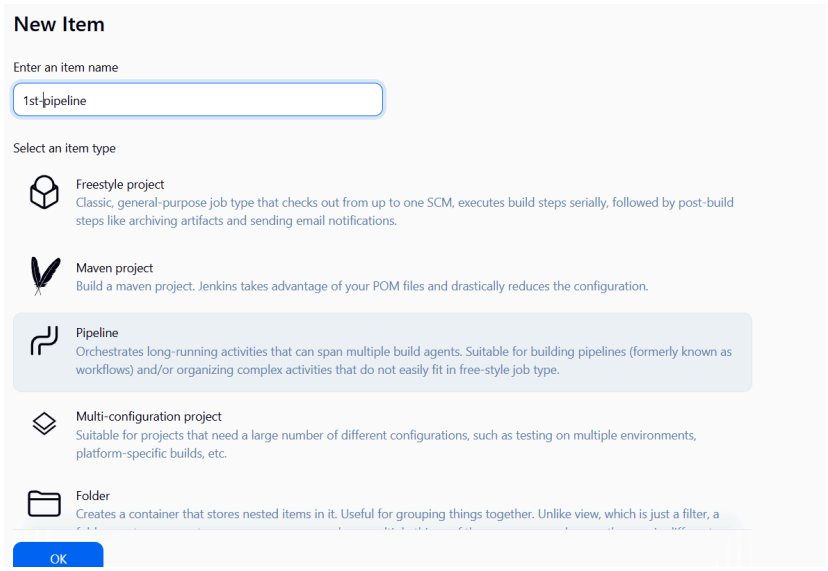
8. Jenkins Pipeline Configuration

In Jenkins:

- Create a new item → **Pipeline**
- Definition: *Pipeline script from SCM*
- SCM: Git
- Repository URL: https://github.com/<your-repo>.git

- Branches: */main
- Script Path: Jenkinsfile






```
pipeline{
    agent none
    stages{
        stage('sonar node'){
            agent {label 'sonar'}
            steps{
                script{
                    //run the commands
                    echo 'hello from the sonar'
                    sh 'whoami'
                    sh 'hostname -i'
                }
            }
        }

```

```
                sh 'df -h'
            }
        }
    }
    stage('nexus node'){
        agent {label 'nexus'}
        steps{
            script{
                //run the commands
                echo 'hello from the nexus'
                sh '''whoami
                      hostname -i
                      df-h'''
            }
        }
```

```
        }
        stage('tomcat node'){
            agent {label 'tomcat'}
            steps{
                script{
                    //run the commands
                    echo 'hello from tomcat'
                    sh '''whoami
                          hostname -i
                          df -h'''
                }
            }
        }
    }
}
```

8. GitHub Webhook Integration

- Go to **GitHub → Repo → Settings → Webhooks → Add Webhook**
- Payload URL: http://<jenkins-public-ip>:8080/github-webhook/
- Content Type: application/json
- Event: **Push event**
- Save Webhook

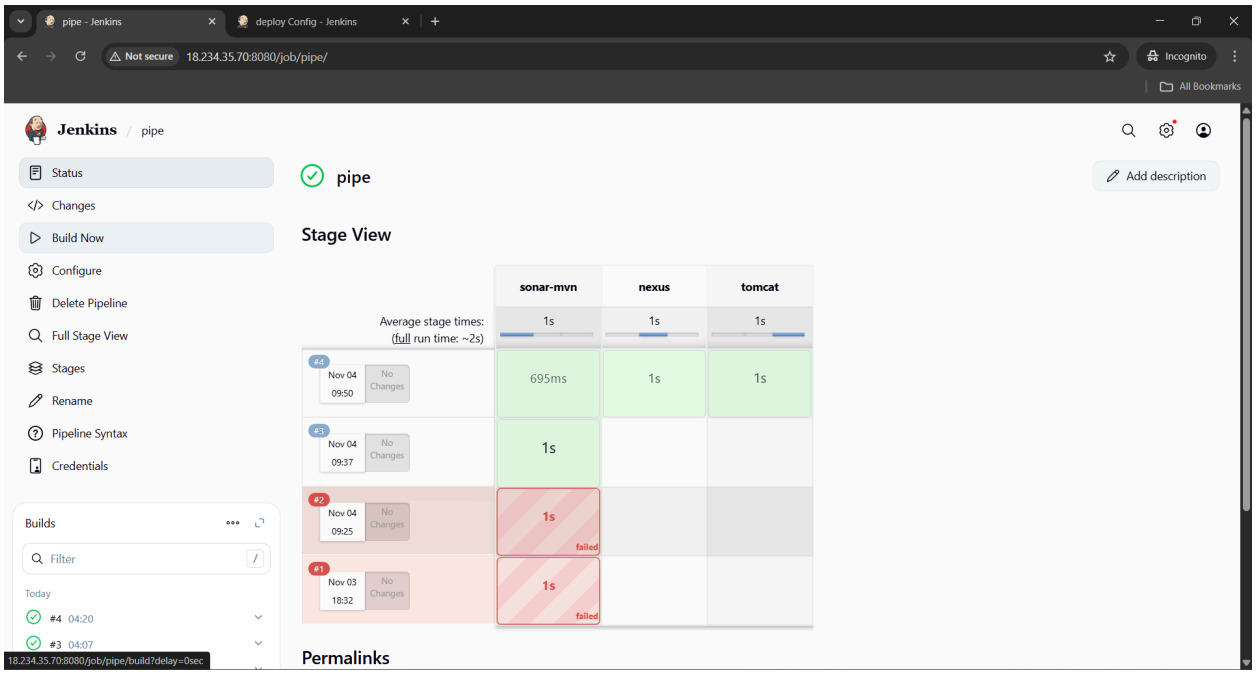
📸 **[[SCREENSHOT: GitHub webhook setup page]]**

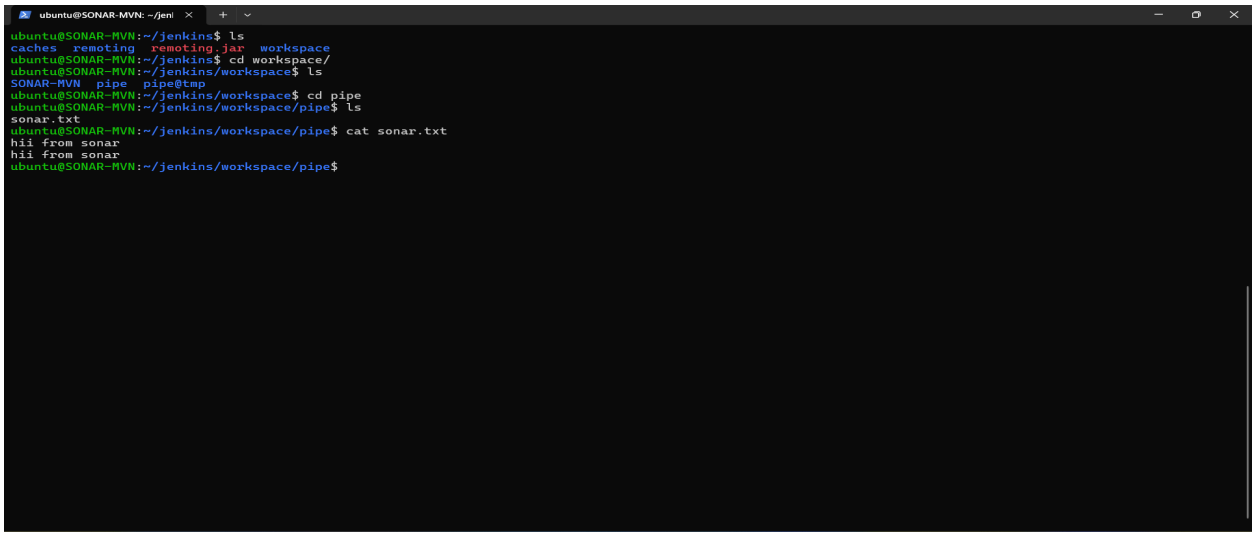
9 Validation

- Push code to the main branch — Jenkins should start automatically.
- Observe stages: Checkout → SonarQube → Build → Nexus Upload → Deploy to Tomcat.

Verify results:

- SonarQube: http://<sonar-ip>:9000
- Nexus: http://<nexus-ip>:8081
- Tomcat: http://<tomcat-ip>:8080/<app-name>/





10. Final Checklist

- [ ] GitHub repo and Jenkinsfile verified
- [ ] Jenkins configured with tools & credentials
- [ ] SonarQube, Nexus, and Tomcat running
- [ ] SSH connectivity verified
- [ ] GitHub webhook added and working
- [ ] Successful build and deployment confirmed.