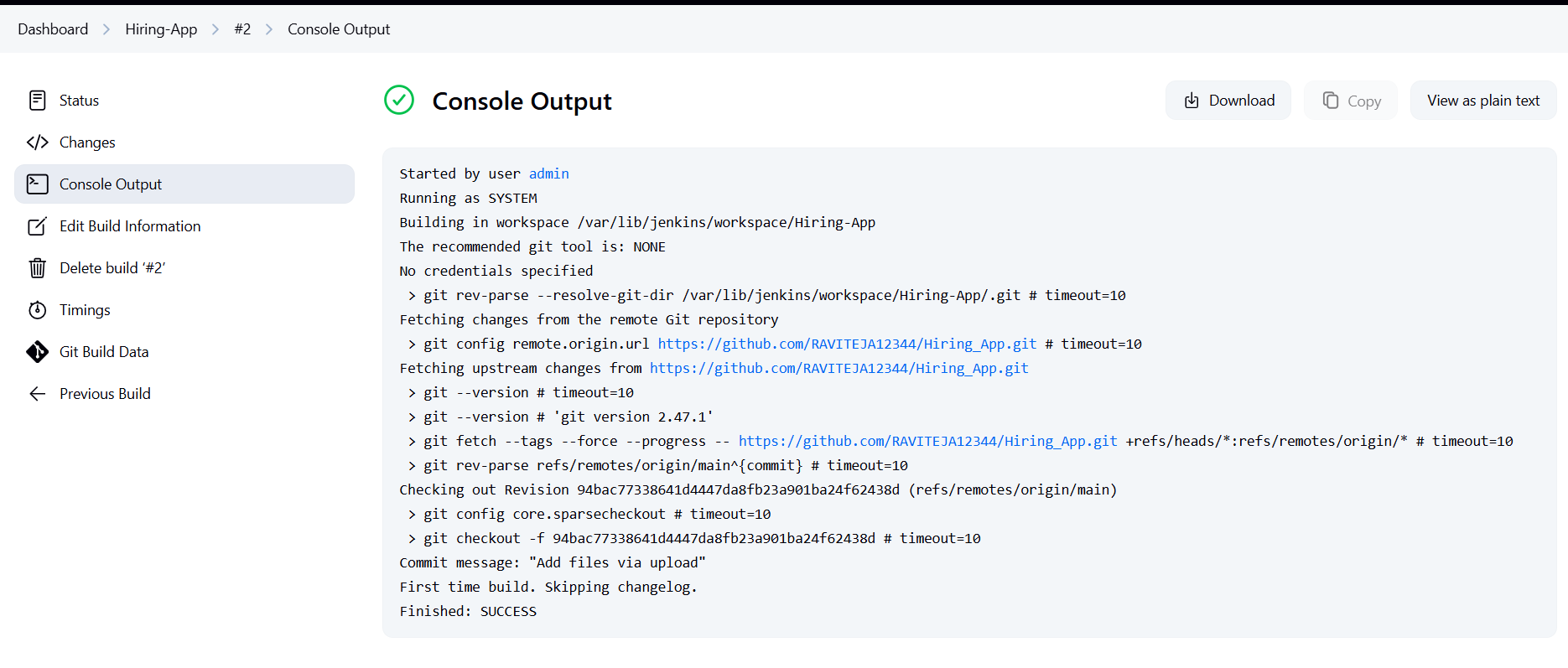
1) Setup jenkins CICD pipeline using freestyle job using Docker containers using below code.

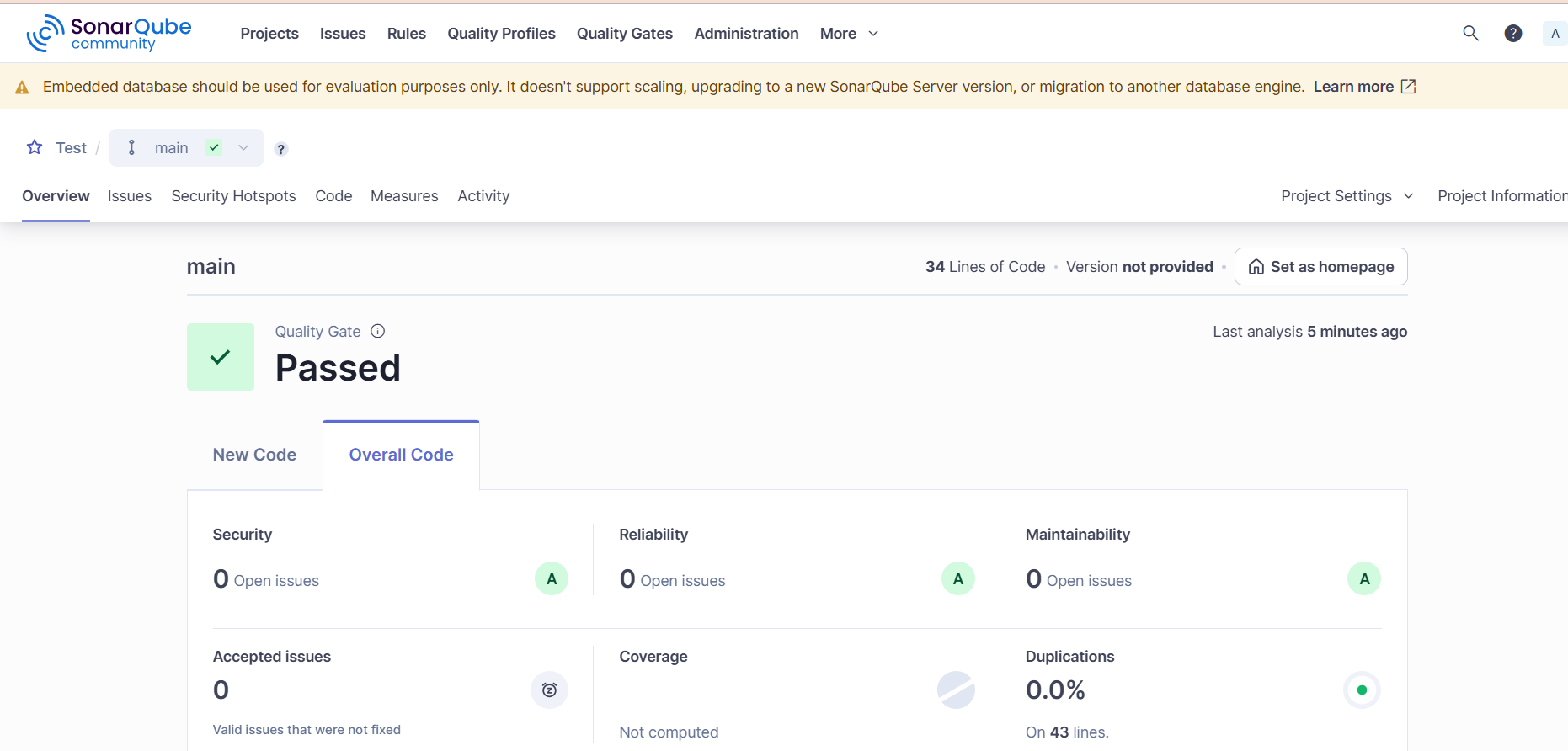
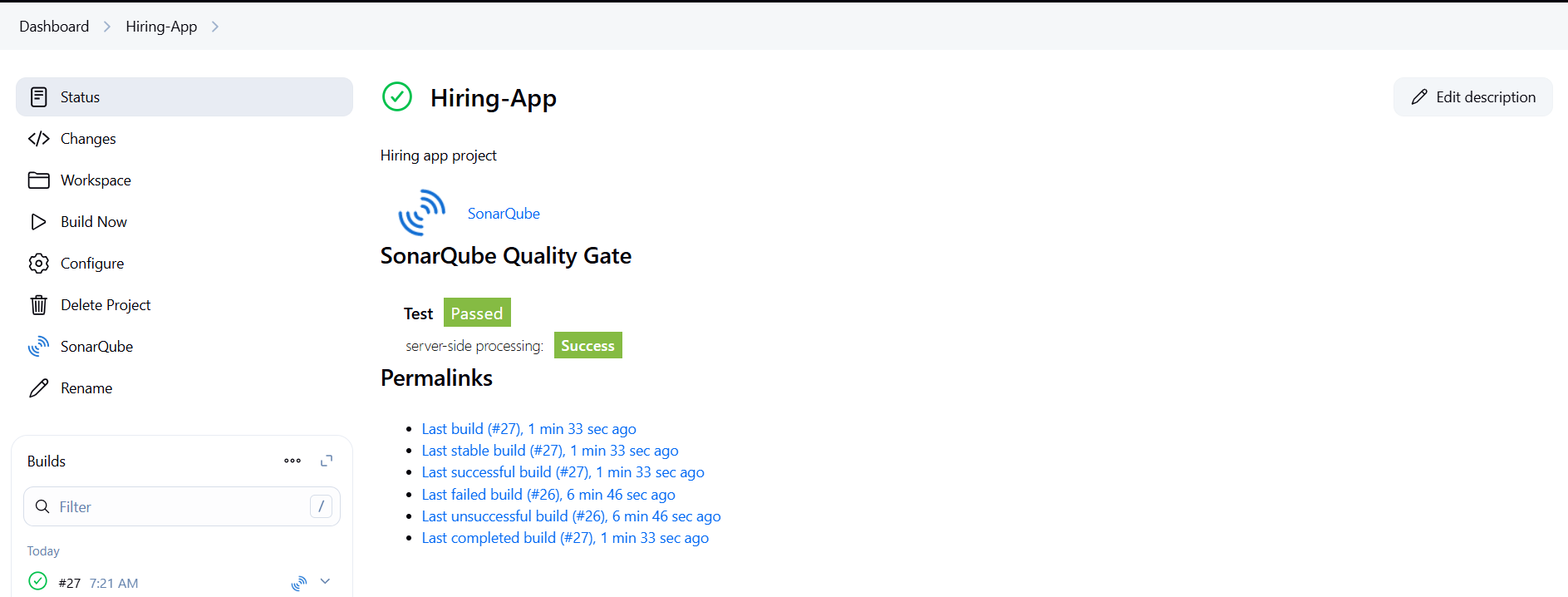
<https://github.com/betawins/hiring-app.git>

Stages:

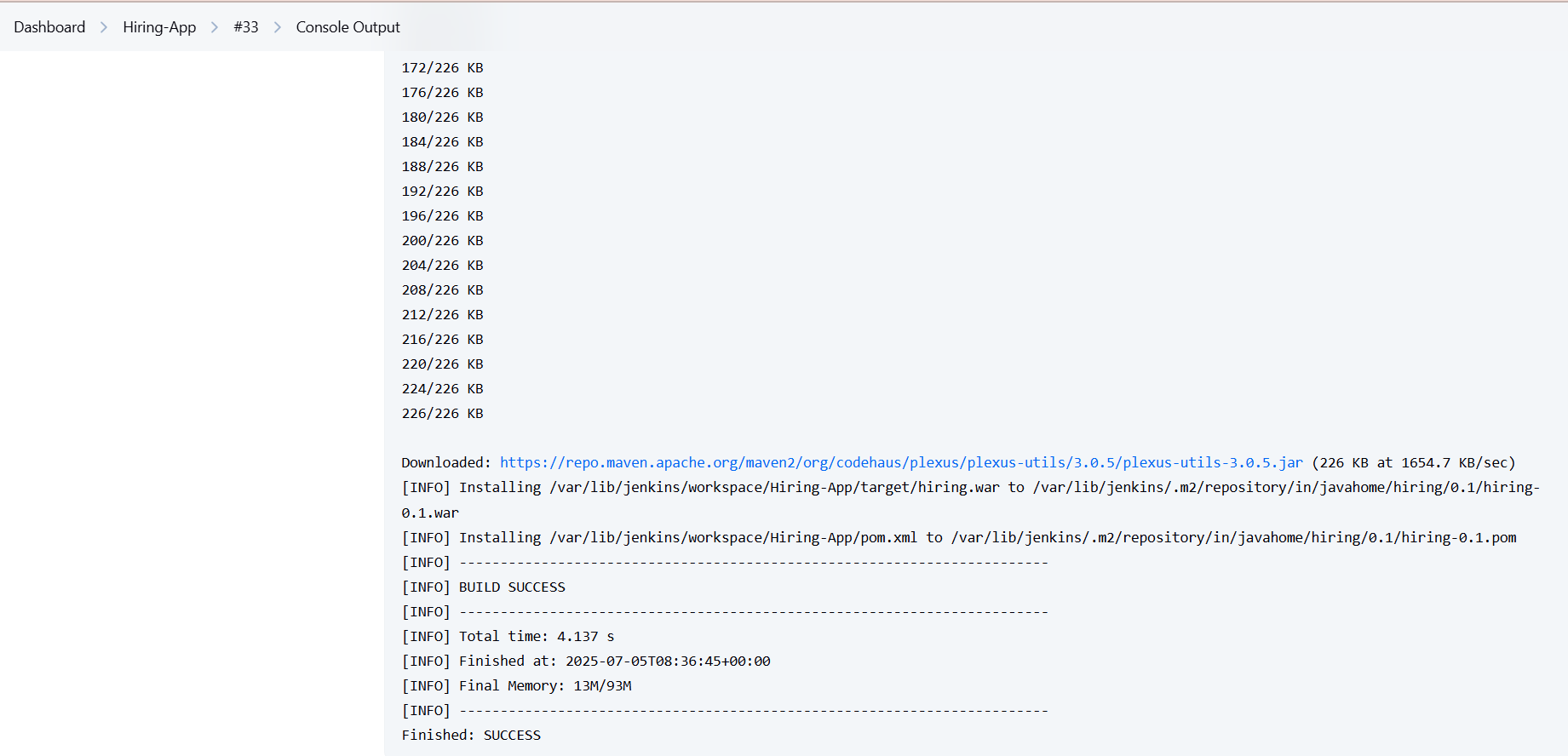
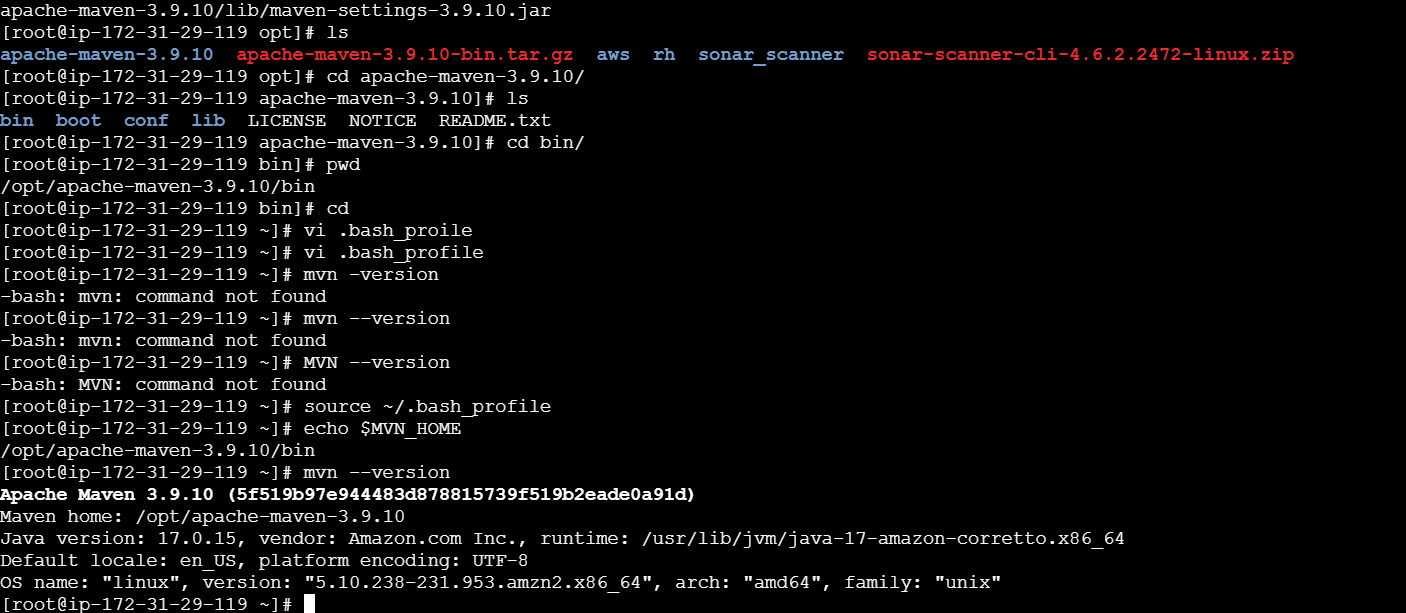
1. Git Clone



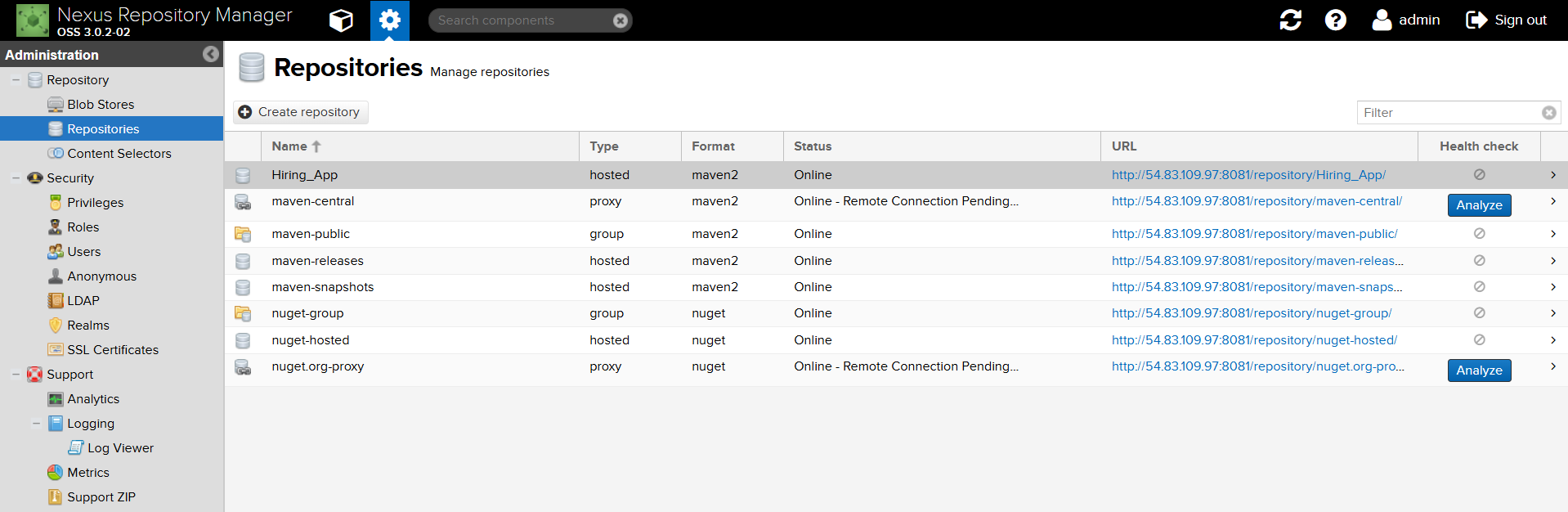
2) Sonarqube Integration

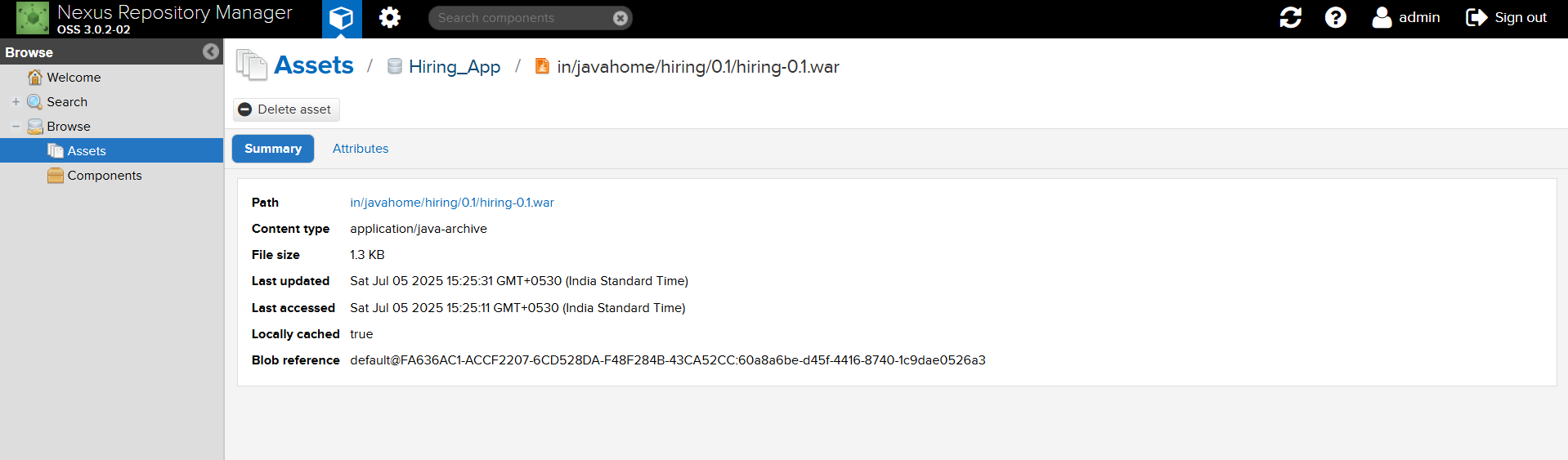
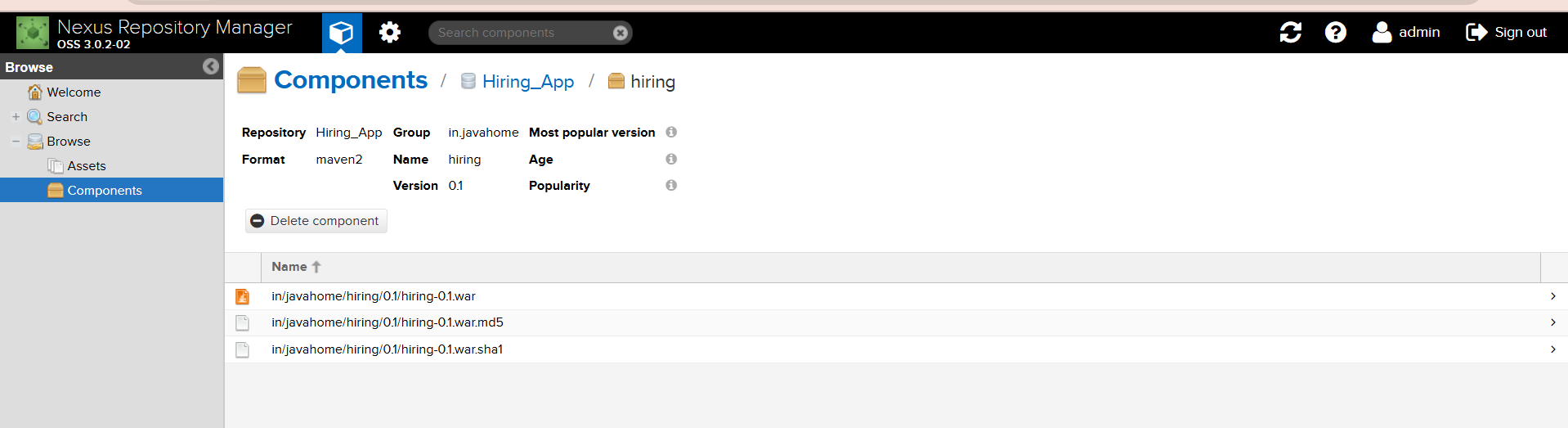


1. Maven Compilation



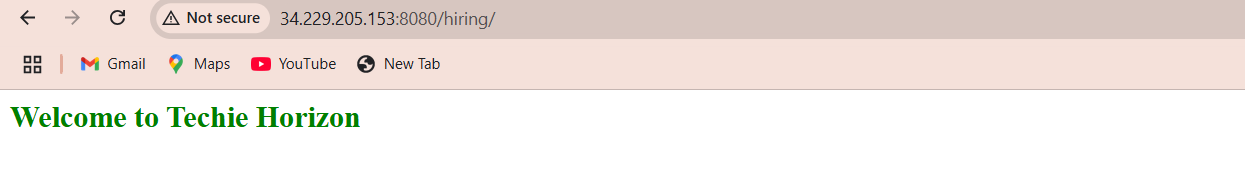
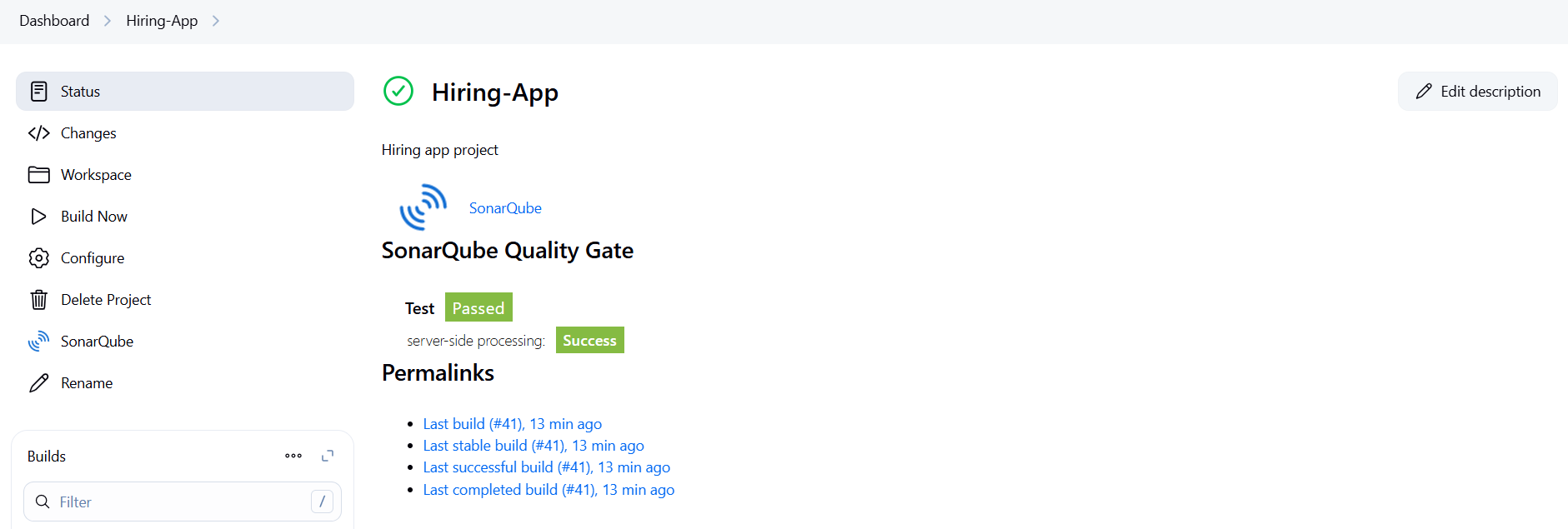
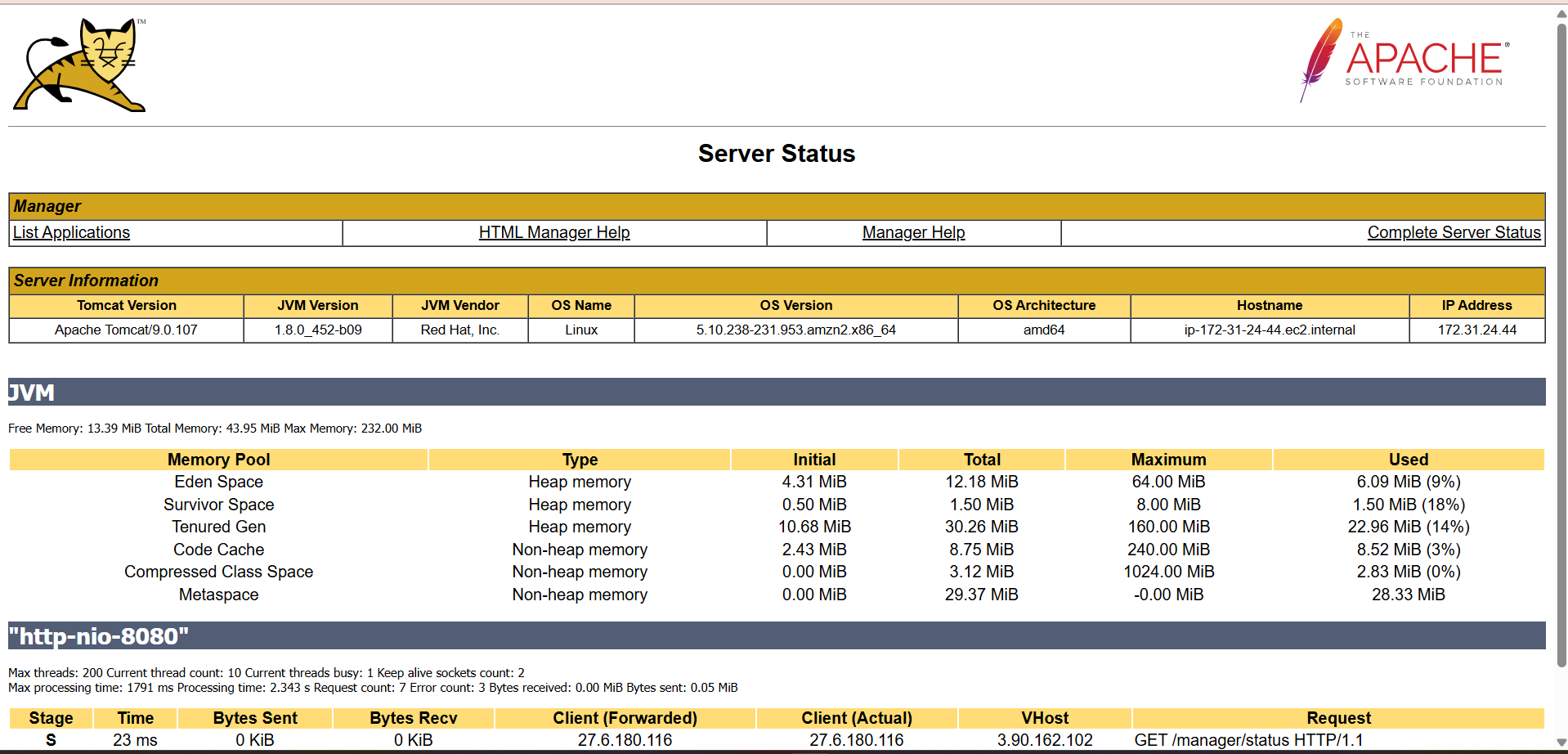
1. Nexus Artifactory





5) Slack Notification

6) Deploy On tomcat



2) Setup a jenkins CICD pipeline using Declarative pipeline using feature-1.1 branch.

<https://github.com/betawins/sabear_simplecutomerapp/tree/feature-1.1>

**CICD Pipeline using Declarative pipeline:**

pipeline{

agent any

tools{

maven 'MAVEN\_HOME'

}

stages{

stage('checkout the code'){

steps{

git url:'https://github.com/RAVITEJA12344/SampleCustomer.git ', branch: 'main'

}

}

stage('build the code'){

steps{

sh 'mvn clean package'

}

}

stage('Generate sonarqube-analysis'){

steps{

withSonarQubeEnv(installationName: 'sonarqube-server', credentialsId: 'Sonarqube-server-token') {

sh 'mvn sonar:sonar'

}

}

}

stage('Upload War file to Nexus'){

steps{

nexusArtifactUploader artifacts: [

[

artifactId: 'hiring',

classifier: '',

file: 'target/hiring.war',

type: 'war'

]

],

credentialsId: 'nexus-server',

groupId: 'in.javahome',

nexusUrl: '34.202.231.31:8081',

nexusVersion: 'nexus3',

protocol: 'http',

repository: 'Hiring\_App',

version: '0.1'

}

}

stage('Deploy to Tomcat') {

steps {

withCredentials([usernamePassword(credentialsId: 'Tomcat-server-1', usernameVariable: 'TOMCAT\_USER', passwordVariable: 'TOMCAT\_PASS')]) {

sh '''

curl -T target/hiring.war "http://${TOMCAT\_USER}:${TOMCAT\_PASS}@52.207.229.8:8080/manager/text/deploy?path=/customerapp&update=true"

'''

}

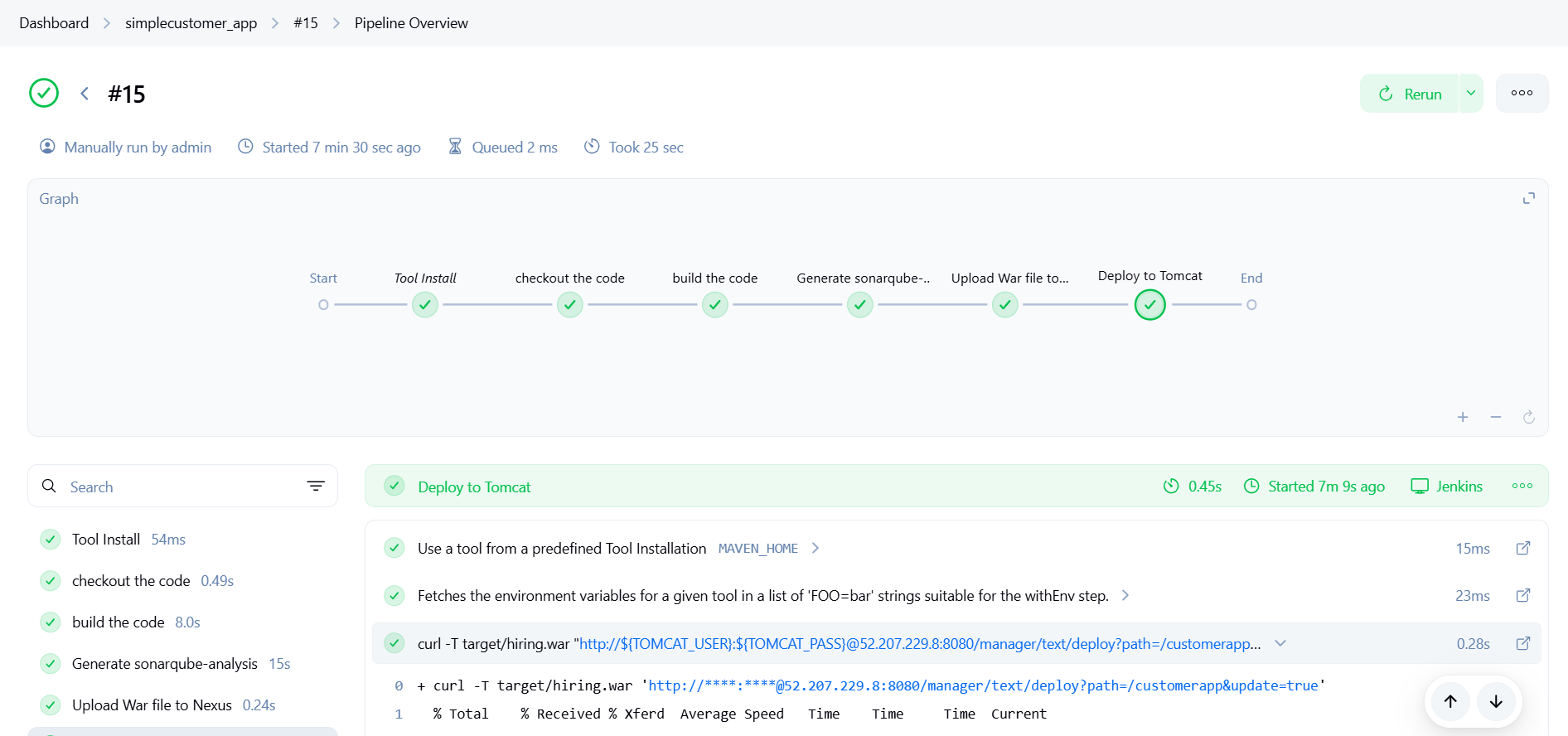
}

}

}

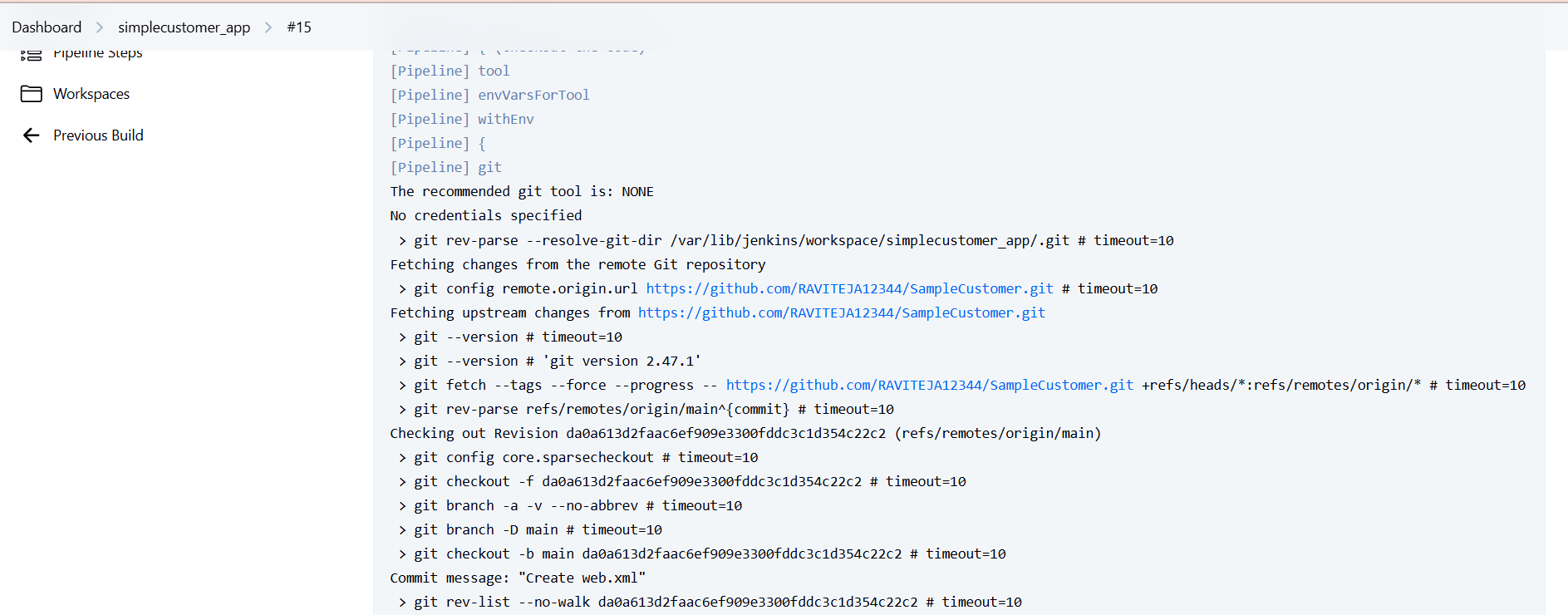
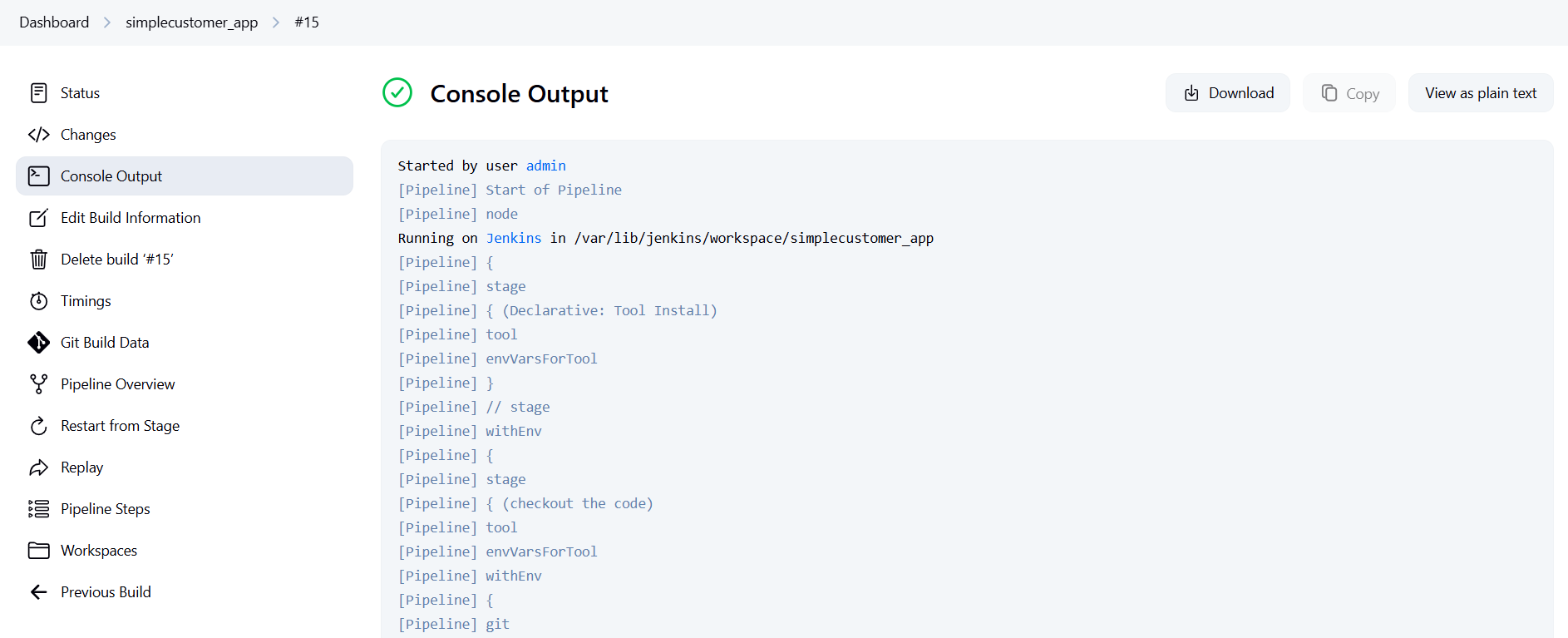
}

Stage overviews:



stages:

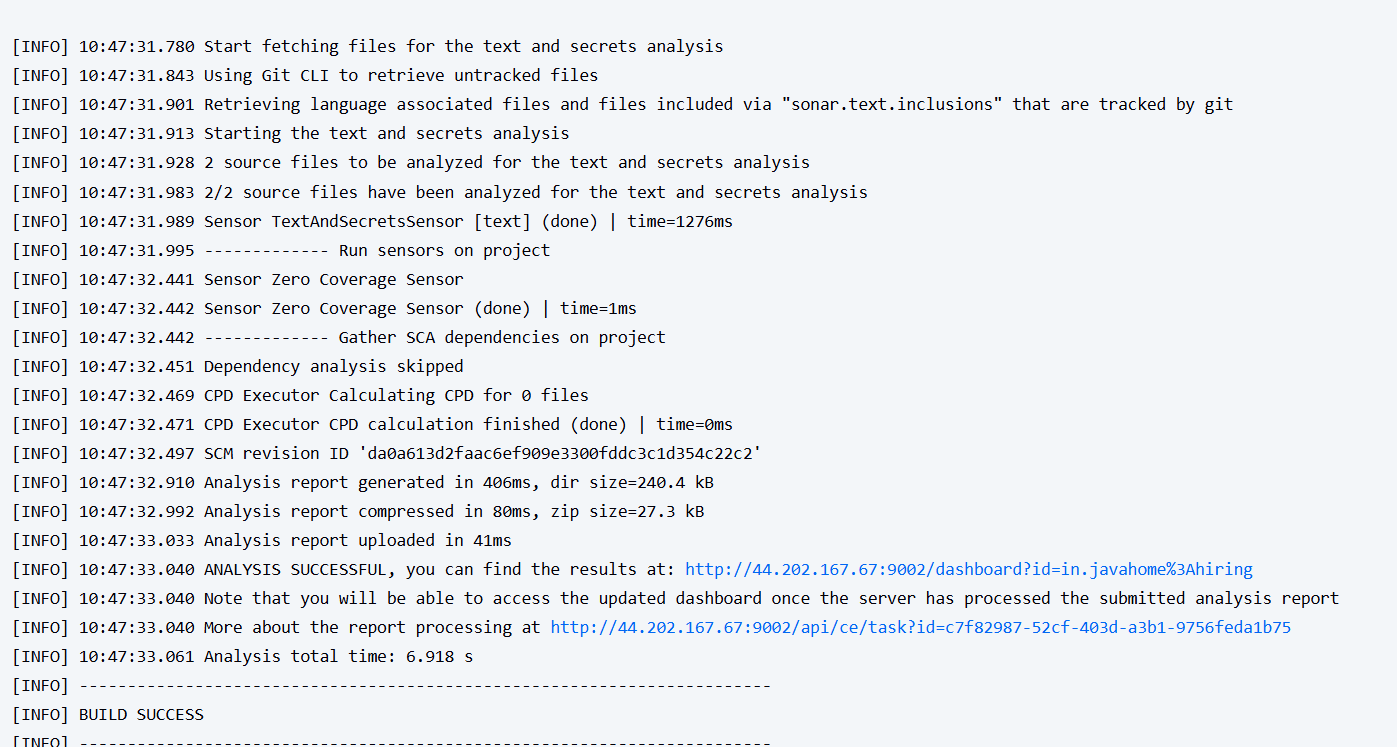
1. Git Clone



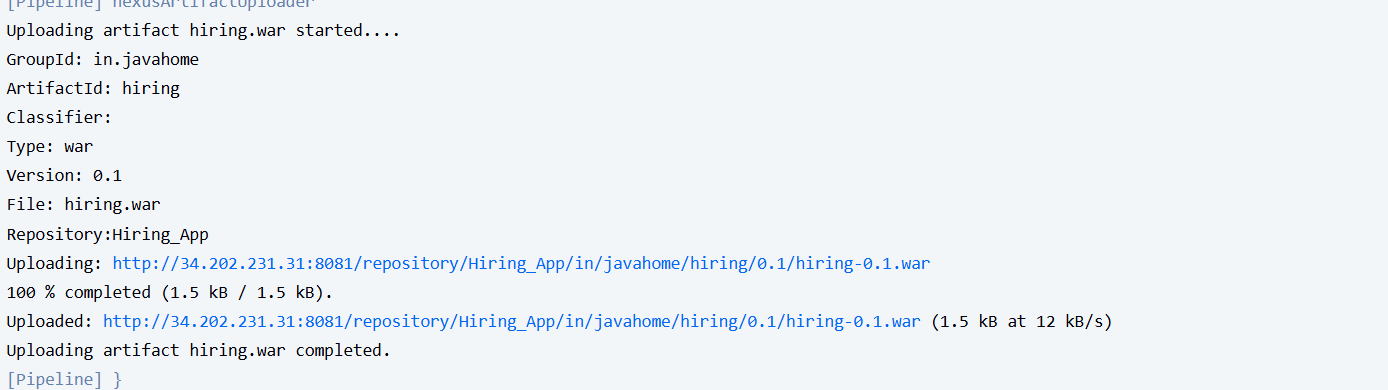
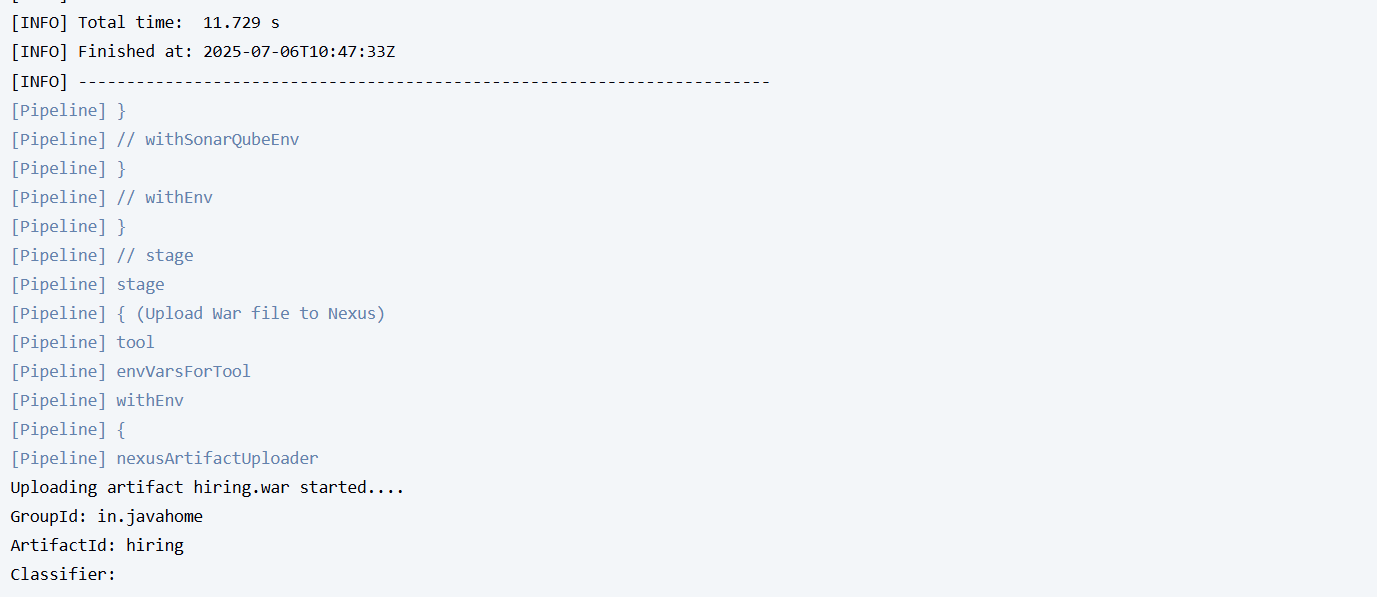
1. Sonarqube Integration



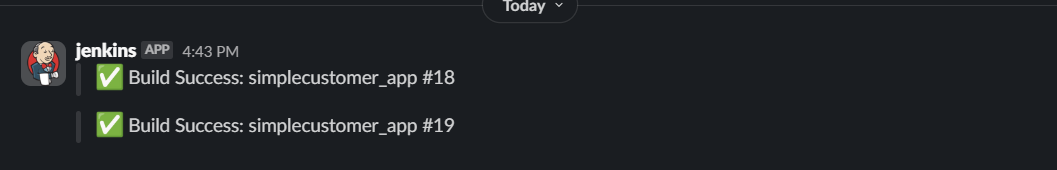
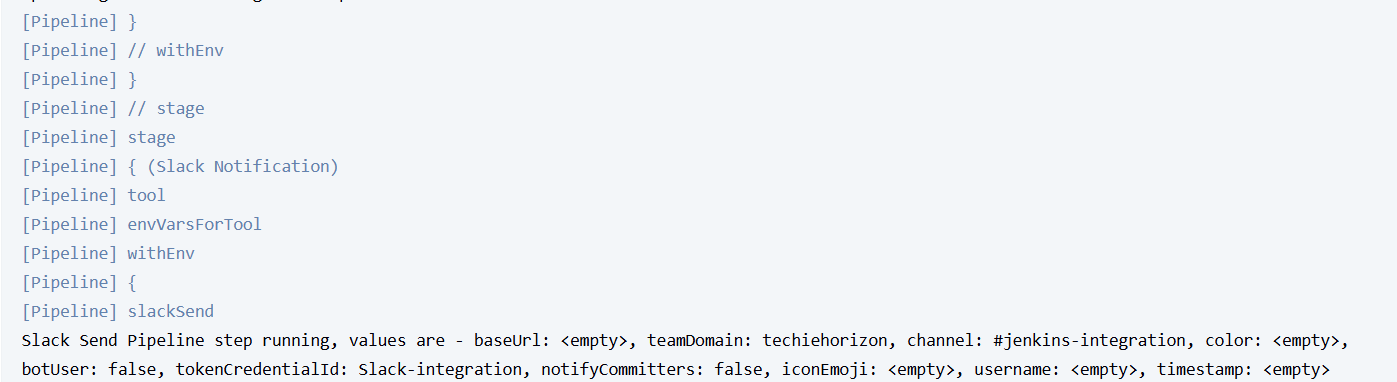
1. Maven Compilation



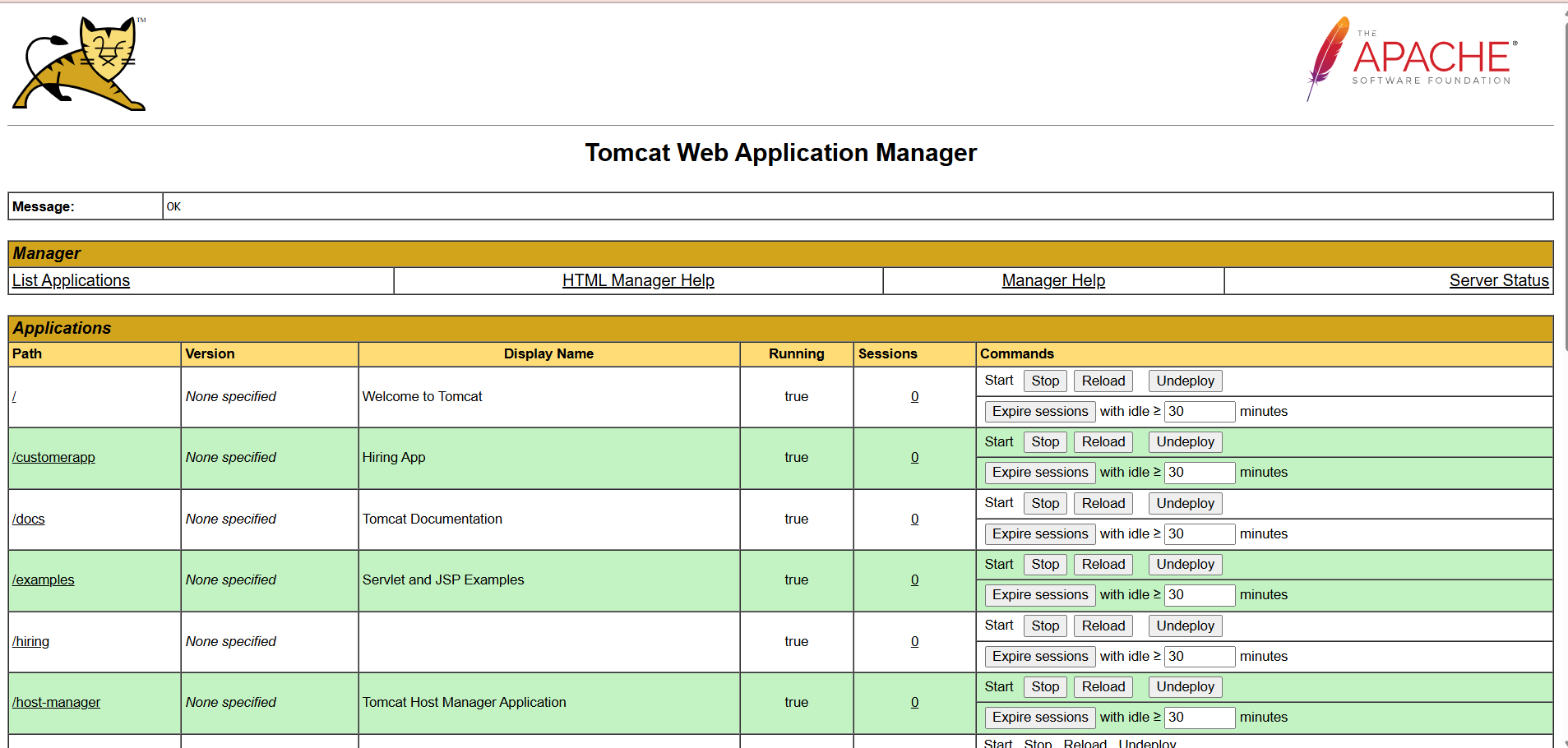
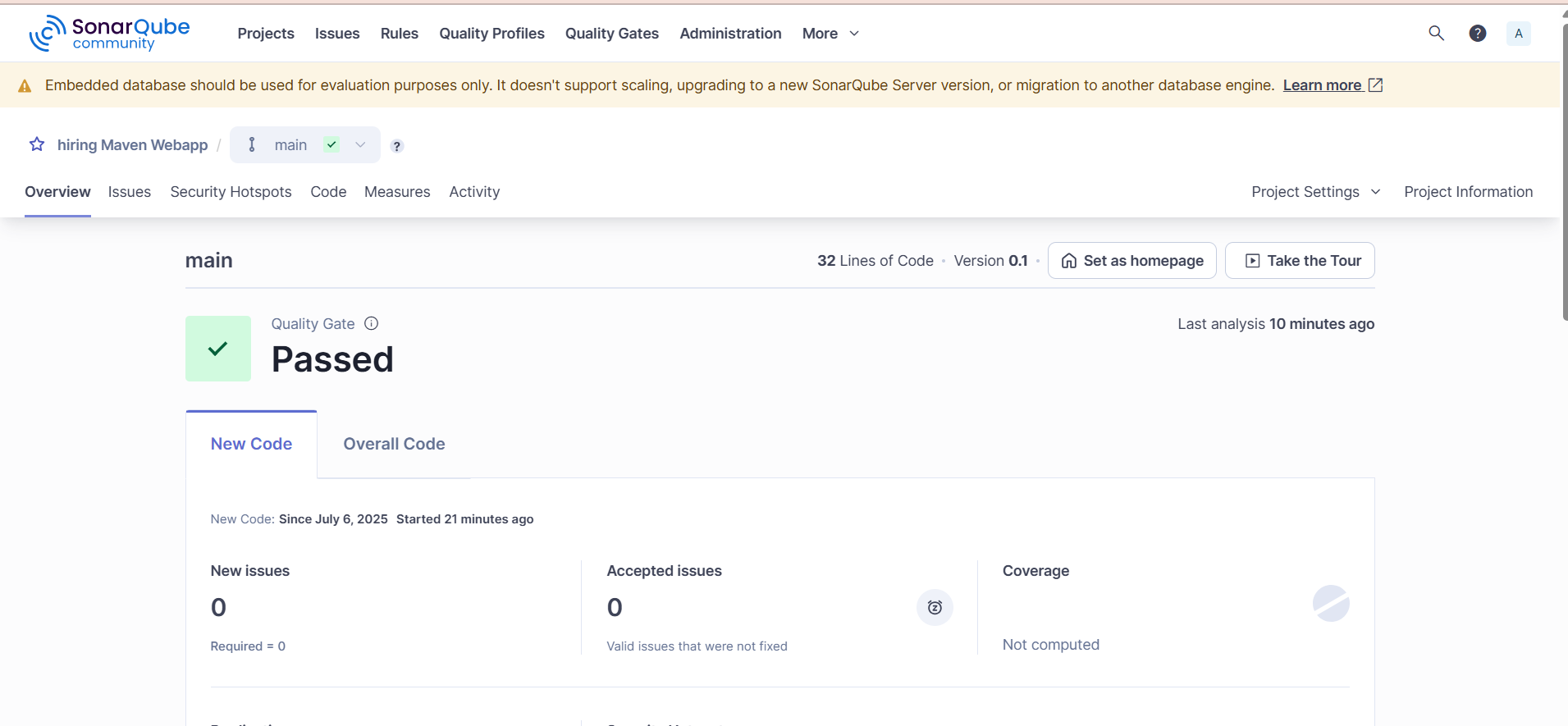
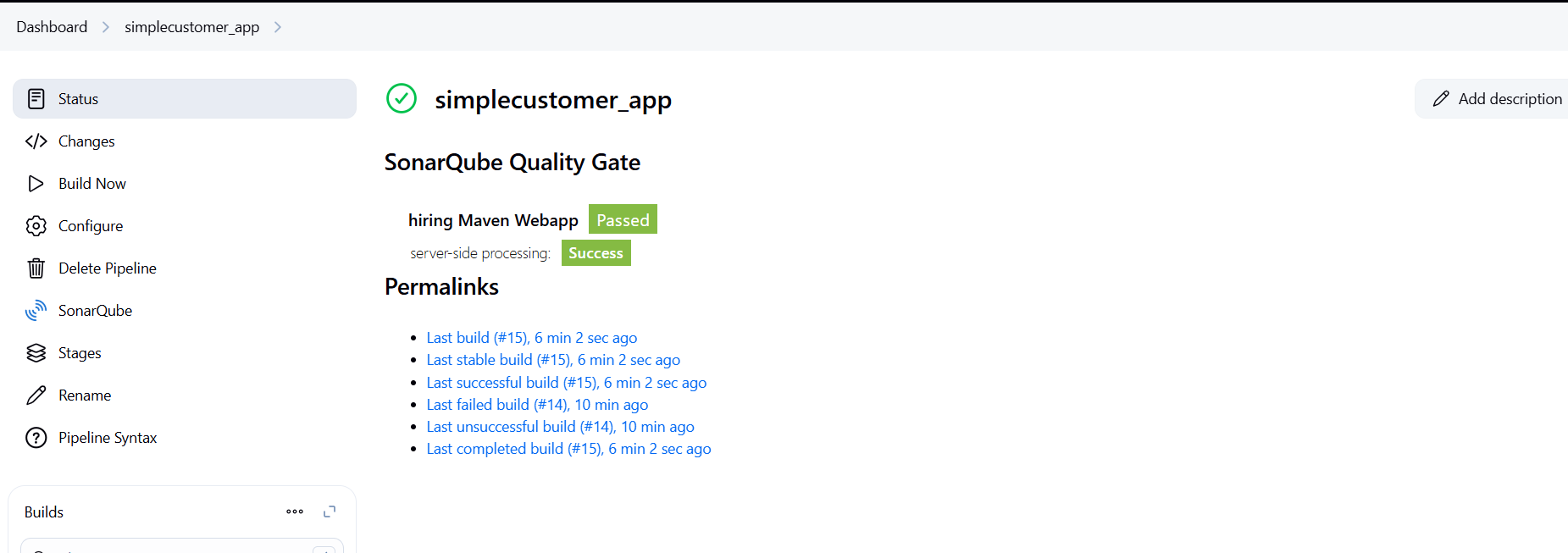
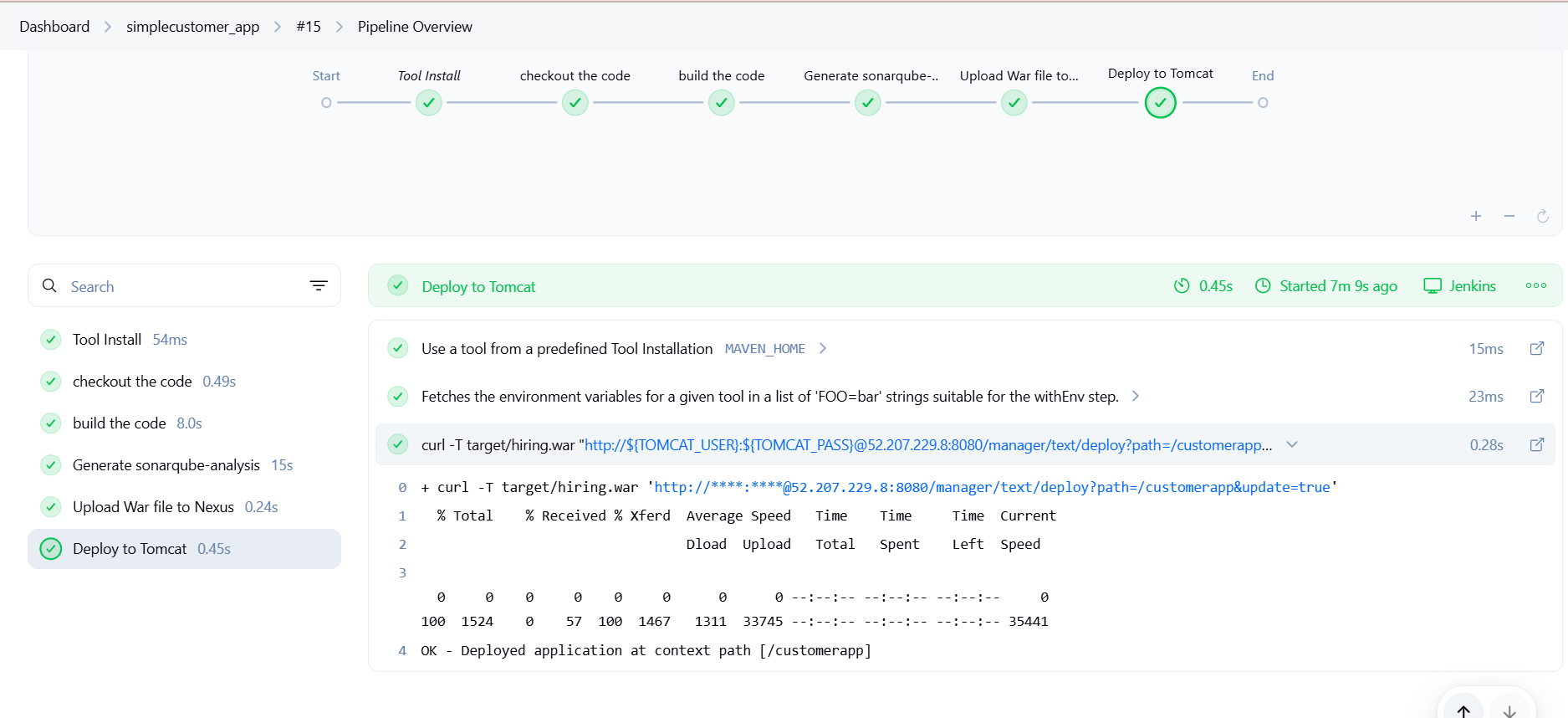
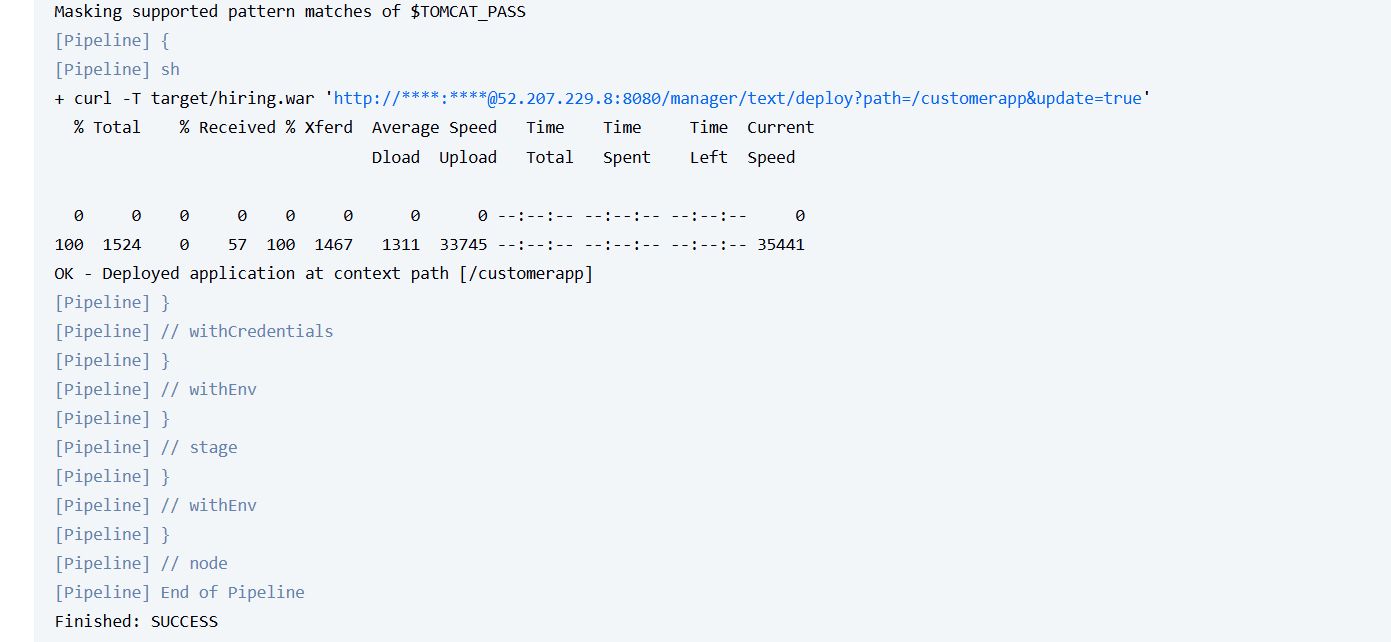
4) Nexus Artifactory



1. Slack Notification

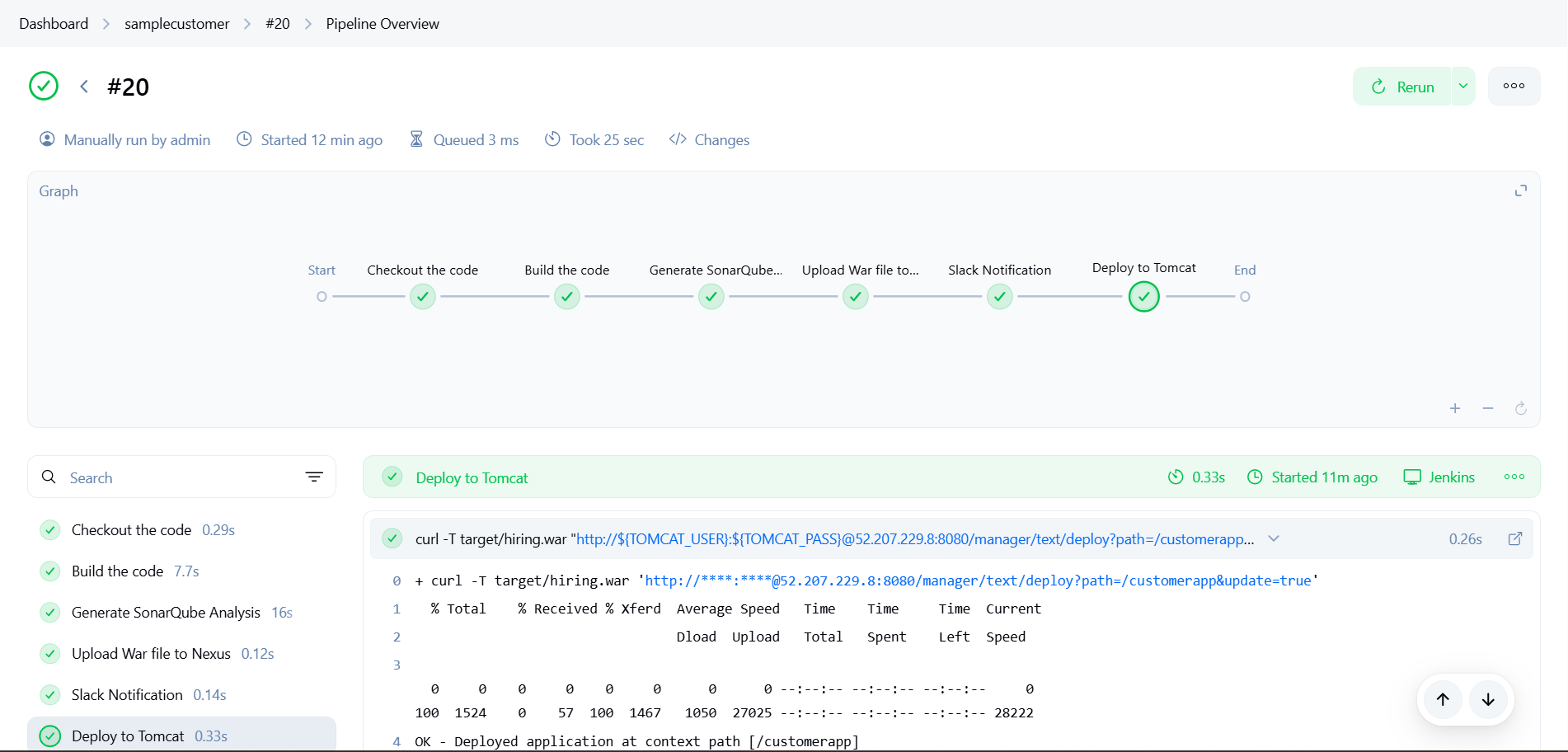


6) Deploy On tomcat



3) Setup a jenkins CICD pipeline using Scripted pipeline using feature-1.1 branch.

<https://github.com/betawins/sabear_simplecutomerapp/tree/feature-1.1>



**jenkins CICD pipeline using Scripted pipeline:**

node {

// Set Maven Tool

def mvnHome = tool name: 'MAVEN\_HOME'

stage('Checkout the code') {

git url: 'https://github.com/RAVITEJA12344/SampleCustomer.git', branch: 'main'

}

stage('Build the code') {

sh "${mvnHome}/bin/mvn clean package"

}

stage('Generate SonarQube Analysis') {

withSonarQubeEnv('sonarqube-server') {

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

}

}

stage('Upload War file to Nexus') {

nexusArtifactUploader(

artifacts: [[

artifactId: 'hiring',

classifier: '',

file: 'target/hiring.war',

type: 'war'

]],

credentialsId: 'nexus-server',

groupId: 'in.javahome',

nexusUrl: '34.202.231.31:8081',

nexusVersion: 'nexus3',

protocol: 'http',

repository: 'Hiring\_App',

version: '0.1'

)

}

stage('Slack Notification') {

slackSend(

channel: '#jenkins-integration',

message: "✅ Build Success: ${env.JOB\_NAME} #${env.BUILD\_NUMBER}",

tokenCredentialId: 'Slack-integration'

)

}

stage('Deploy to Tomcat') {

withCredentials([usernamePassword(credentialsId: 'Tomcat-server-1', usernameVariable: 'TOMCAT\_USER', passwordVariable: 'TOMCAT\_PASS')]) {

sh '''

curl -T target/hiring.war "http://${TOMCAT\_USER}:${TOMCAT\_PASS}@52.207.229.8:8080/manager/text/deploy?path=/customerapp&update=true"

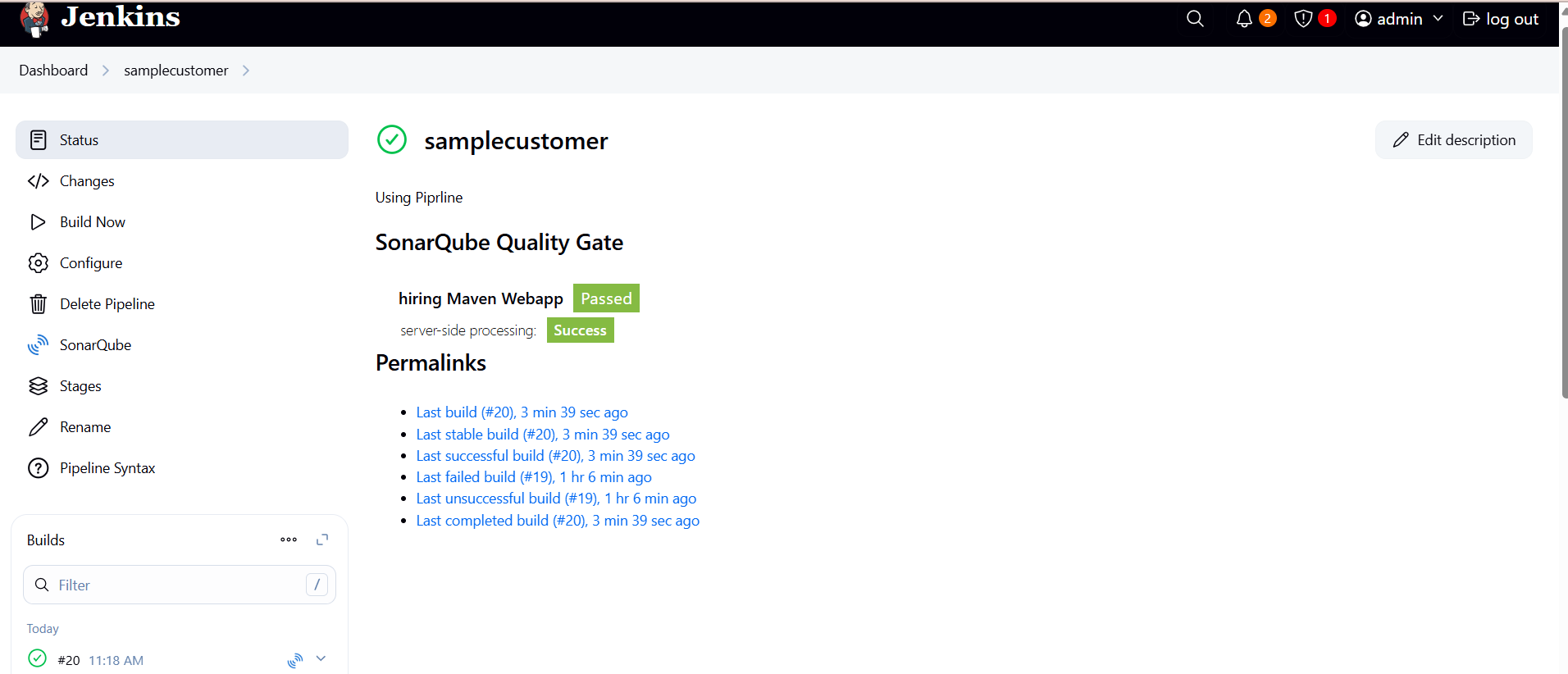
'''

}

}

}

stages:



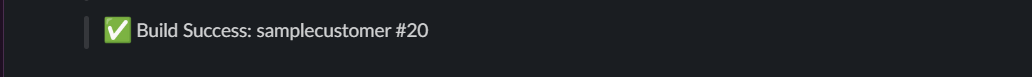
1) Git Clone

2) Sonarqube Integration

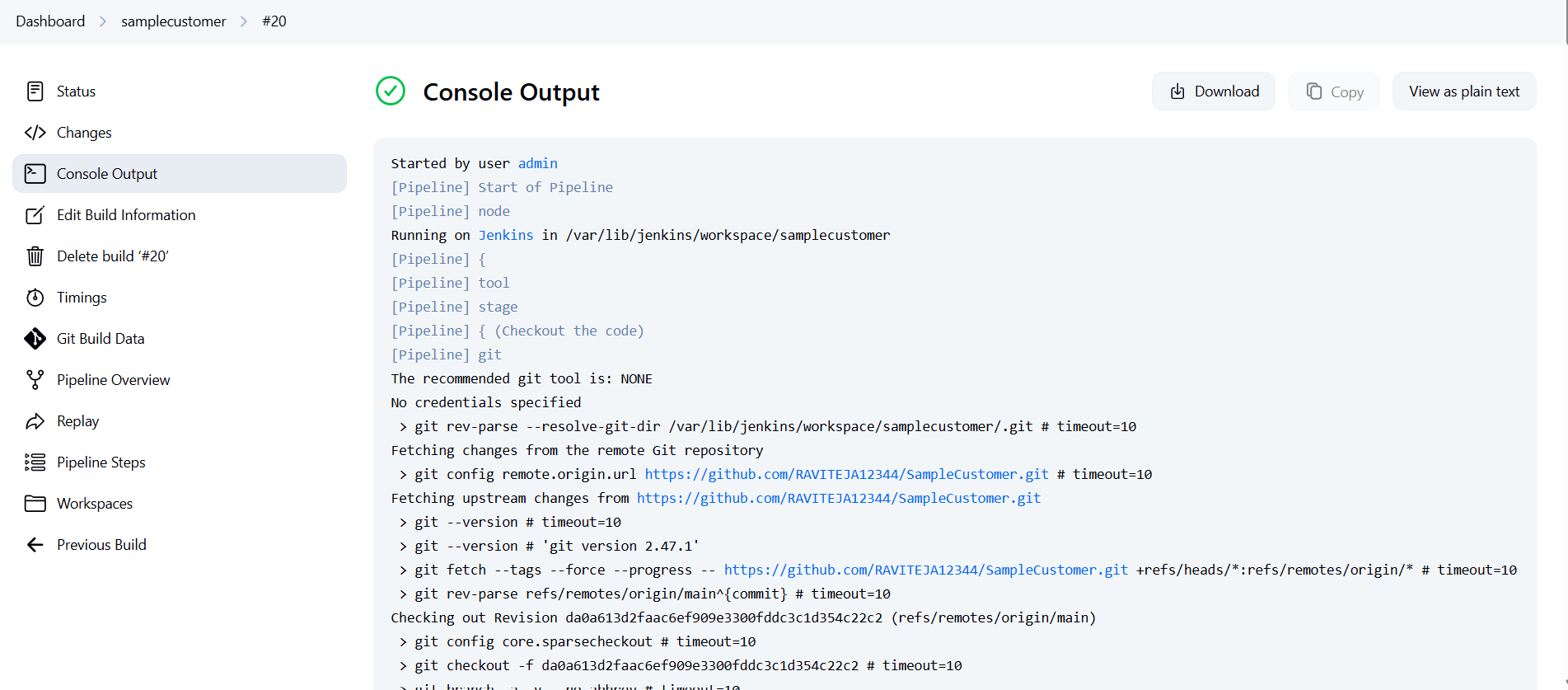
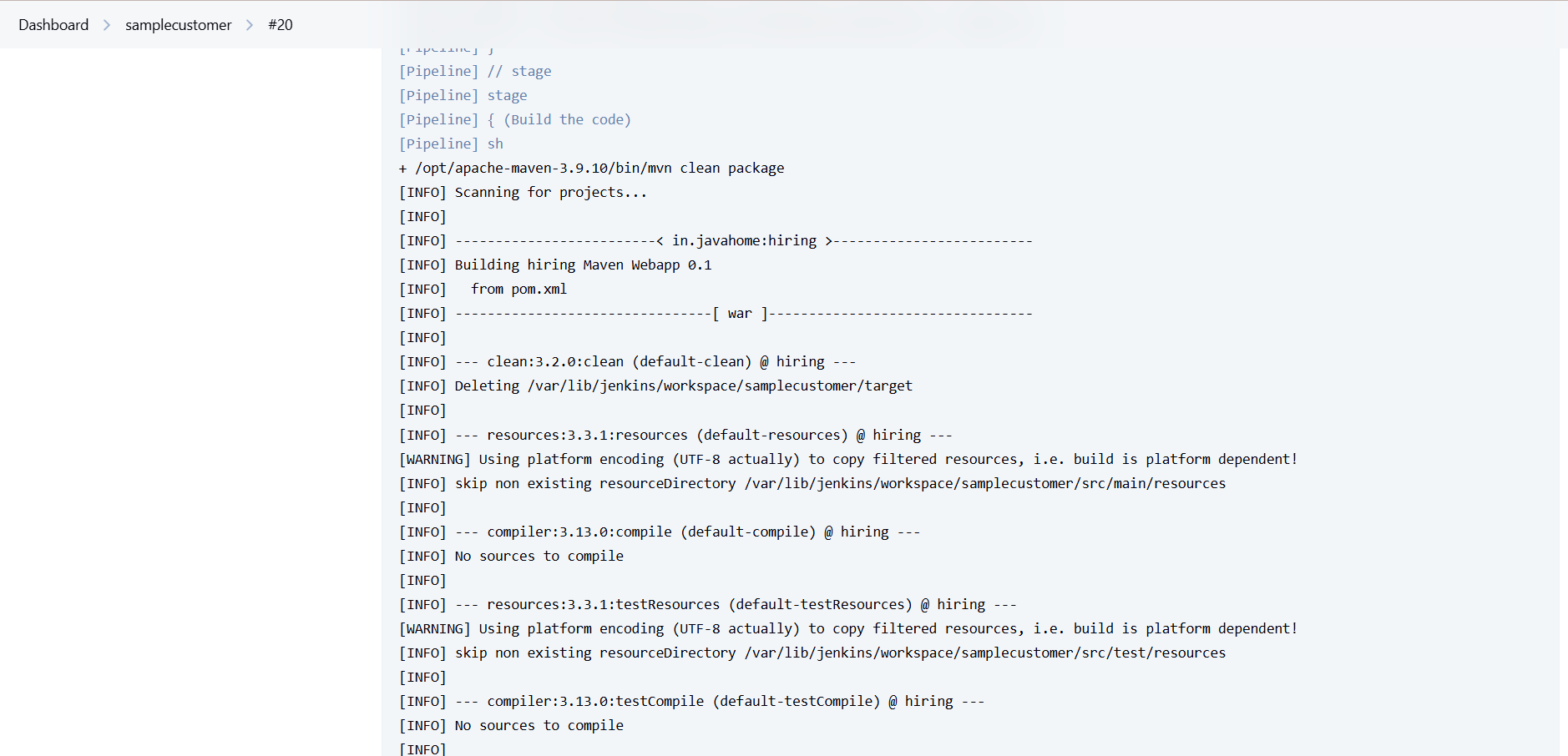
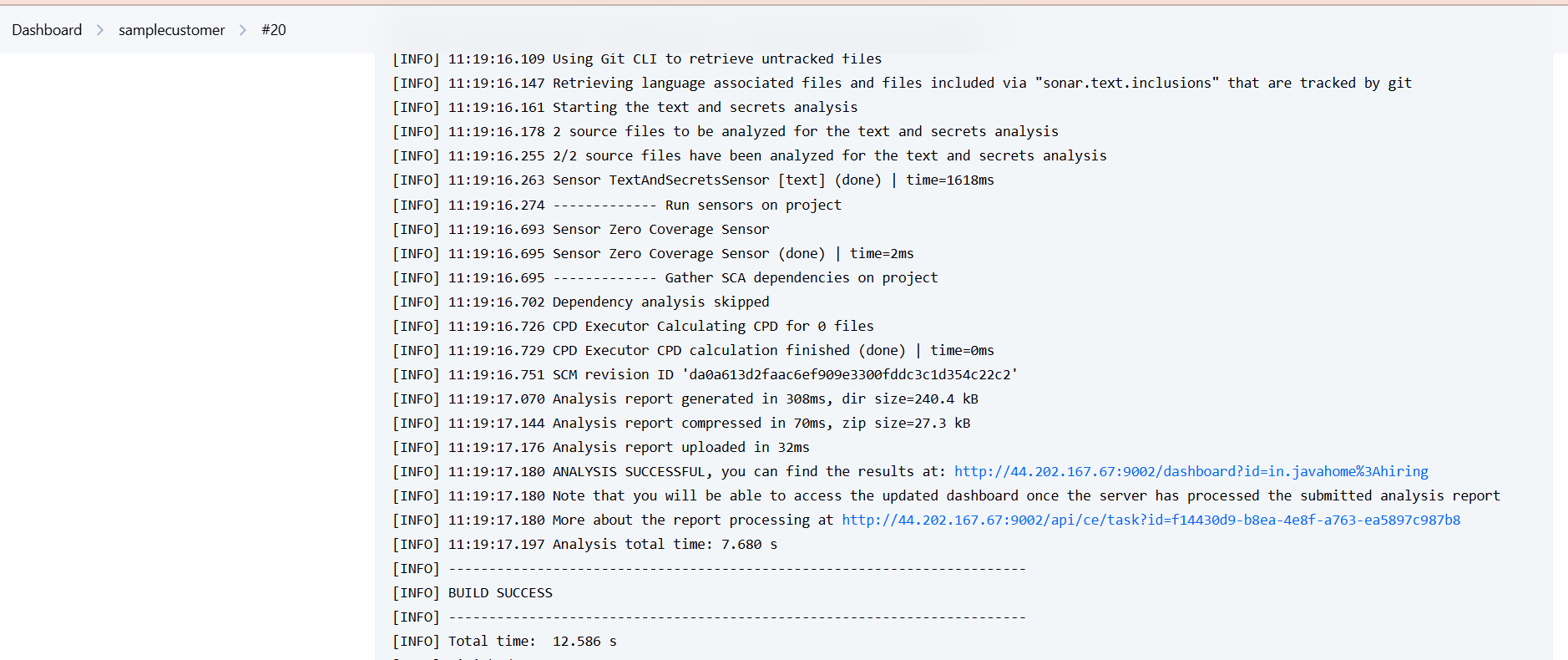
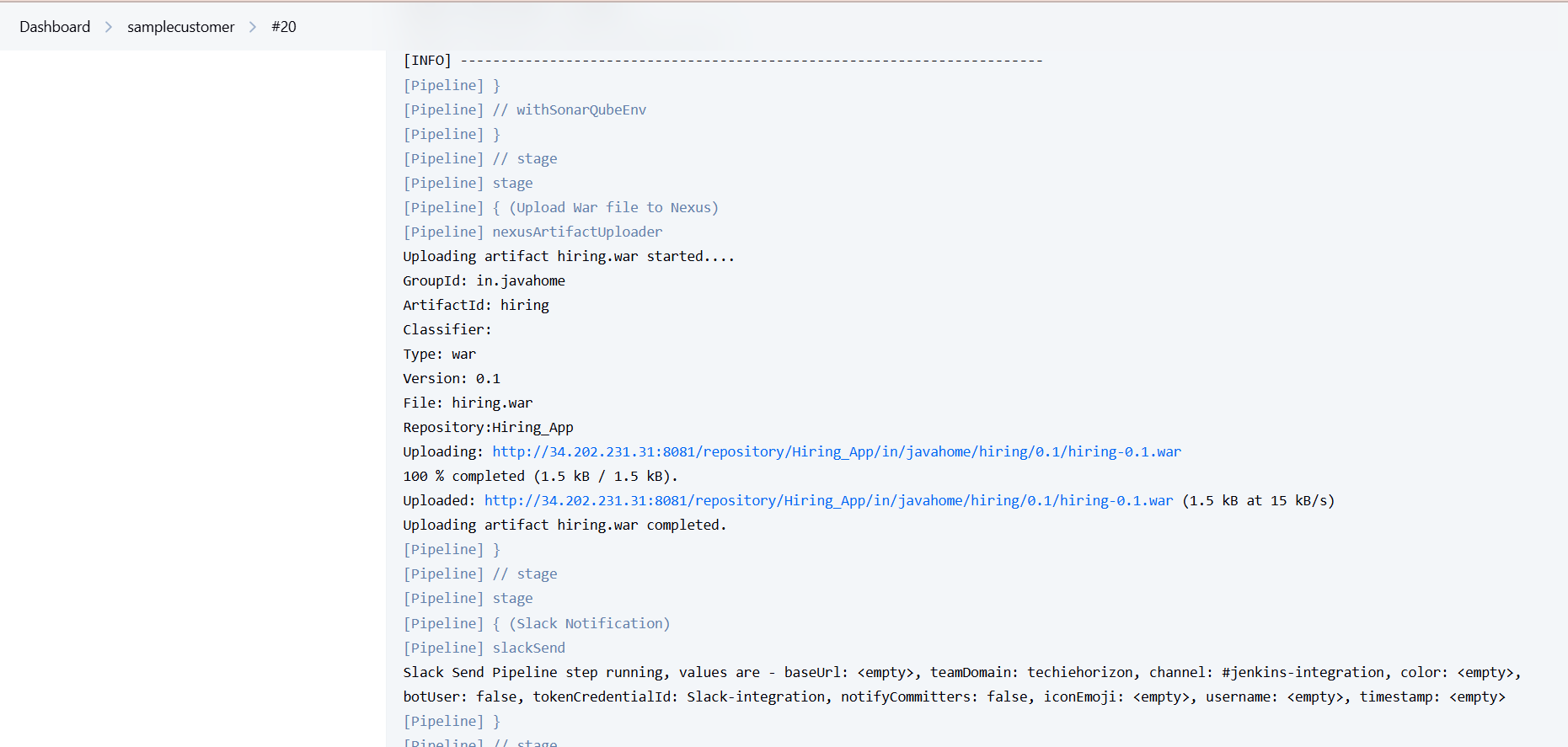
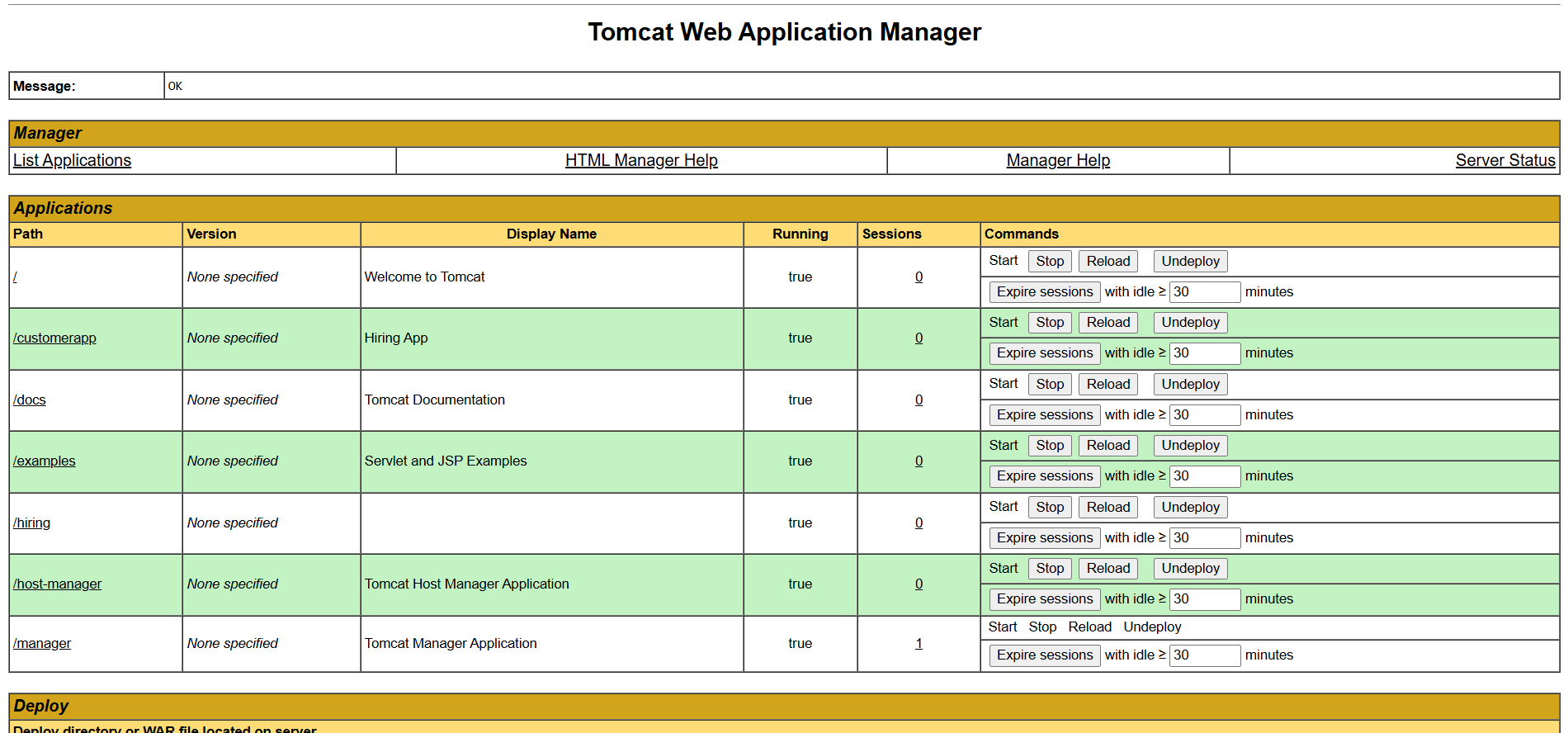
3) Maven Compilation

4) Nexus Artifactory

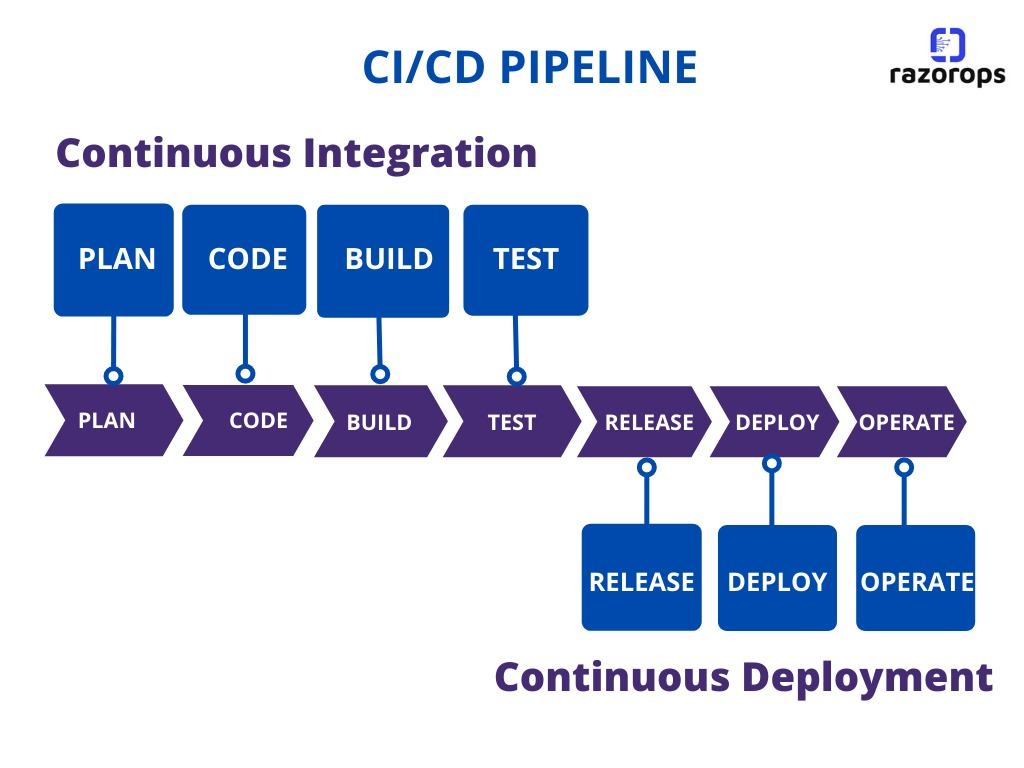
5) Slack Notification



6) Deploy On tomcat

1. Write sample skeleton of pipelines.



🟩 Declarative Pipeline Skeleton

pipeline {

agent any

environment {

// Define global environment variables

APP\_VERSION = '1.0.0'

}

options {

// Optional: timeout, timestamps, etc.

timeout(time: 10, unit: 'MINUTES')

timestamps()

}

parameters {

// Optional: Parameterized build inputs

string(name: 'BRANCH', defaultValue: 'main', description: 'Git branch to build')

}

tools {

// Reference Jenkins-managed tools

maven 'Maven'

jdk 'JDK17'

}

stages {

stage('Checkout') {

steps {

git branch: "${params.BRANCH}", url: 'https://github.com/your/repo.git'

}

}

stage('Build') {

steps {

sh 'mvn clean package'

}

}

stage('Test') {

steps {

sh 'mvn test'

}

}

stage('Deploy') {

steps {

echo "Deploying version ${env.APP\_VERSION}"

}

}

}

post {

success {

echo '✅ Pipeline completed successfully'

}

failure {

echo '❌ Pipeline failed'

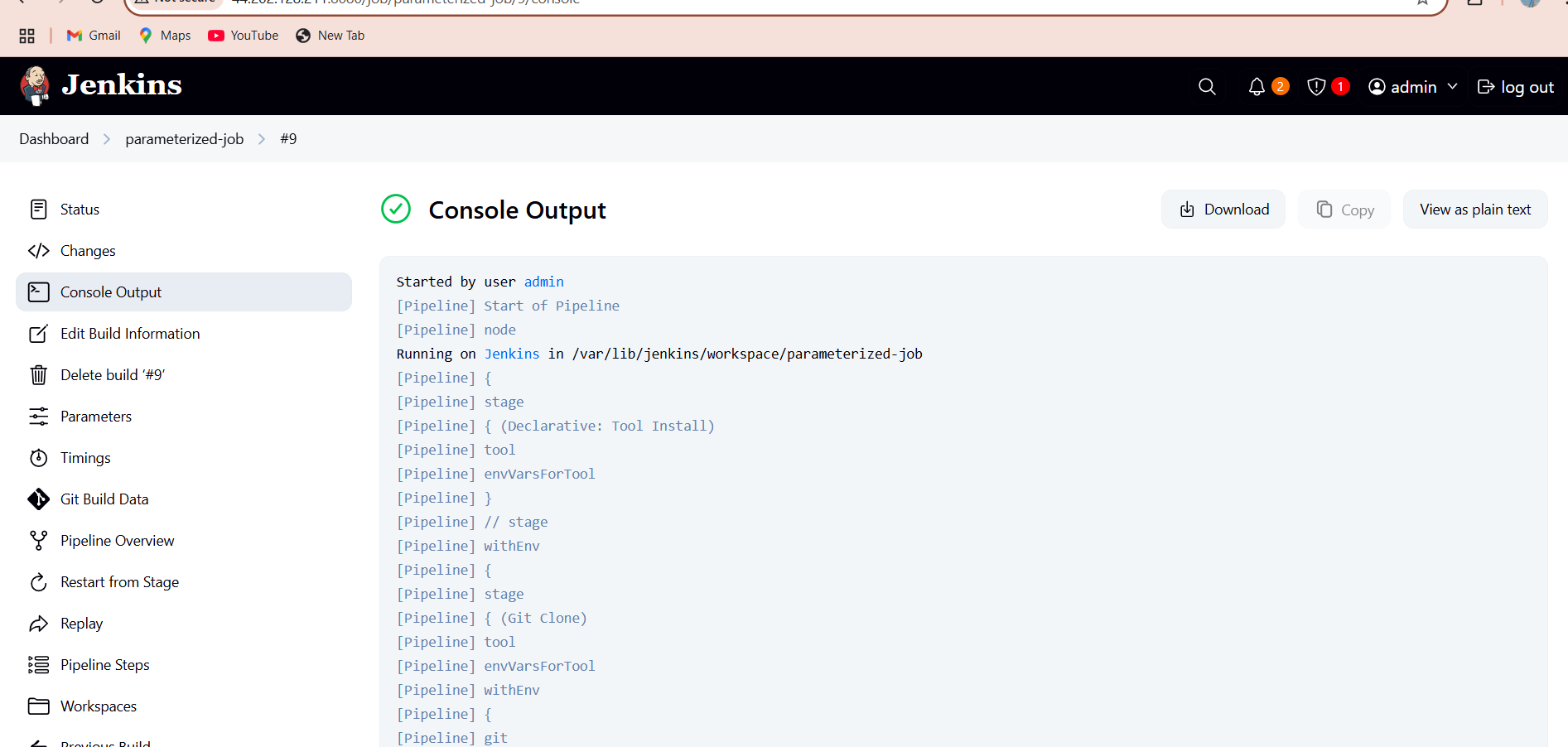
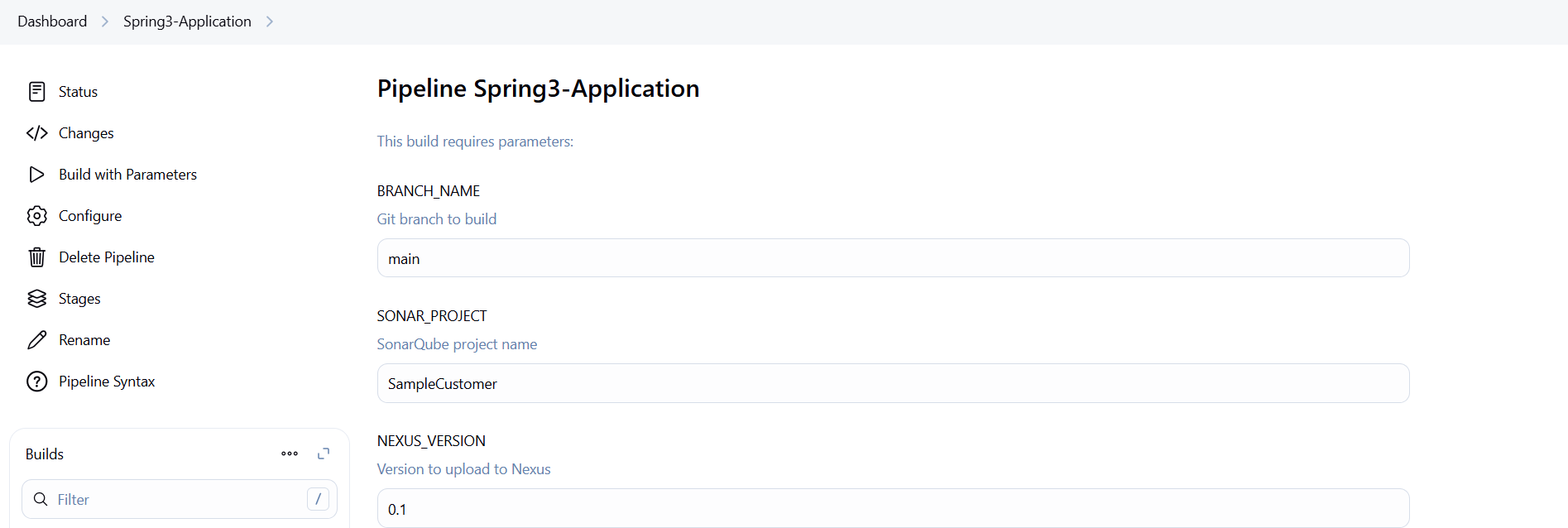
}

}

}

5) Create a parametirized job in jenkins.

<https://github.com/betawins/spring3-mvc-maven-xml-hello-world-1.git>



1. Setup one slave machine for jenkins.

