Mini Project(Zomato)

Author: Krishna Prasad Janney
Email Id: Janneyk92@gmail.com

Clean: IT WILL CLEAN THE WORKSPACE IN JENKINS JOB

CODE: WE WILL GET THE CODE FROM GITHUB TO SERVER

SONAR: IT WILL SCAN THE ENTIER CODE AND DISPLAY THE BUGS AND CODE SMELLS

DEPENDENCIES INSTALL: AS WE ARE NEED NODE JS, THIS STAGE WILL DOWNLOAD ALL THE DEPENDENCIES FROM PACKAGE.JSON FILES

OWASP: OPEN WEB APPLICATION SECURE PROJECT- IT WILL CHECK THE DEPENDENCIES

TRIVY SCAN: IN THIS STAGE ALL THE FILES WILL GET COPED INTO TRIVYFS.TXT FILE

BUILD: WE WILL BUILD THE DOCKER IMAGE

PUSH: PUSH THE DOCKER IMAGE INTO DOCKER HUB

TRIVY IT WILL SCAN THE IMAGE

CONTAINER: CONTAINER WILL GET CREATED AND APPLICATION WILL GET DEPLOYE

Step 1: Launch EC2 INSTANCE with t2.Large

STEP 2: INSTALL JENKINS, GIT, DOCKER & TRIVY

Trivy:

wget https://github.com/aquasecurity/trivy/releases/download/v0.18.3/trivy_0.18.3_Linux-64bit.tar.gz

tar zxvf trivy_0.18.3_Linux-64bit.tar.gz

mv trivy /usr/local/bin/

Vim .bashrc

export PATH=$PATH:/usr/local/bin/

source .bashrc

trivy -version

JENKINS: https://pkg.jenkins.io/redhat-stable/

amazon-linux-extras install java-openjdk11 -y

sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo

sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key

yum install jenkins -y

systemctl start jenkins

systemctl status jenkins

Connect jenkins with instance PUBLIC-IP:8080

Click on Installed suggested plugins (user name-krishna, pass- K123456789)

Git & Docker:

yum install git docker -y

systemctl start docker

chmod 777 ///var/run/docker.sock

STEP 3: INSTALL THE FOLLOWING JENKINS PLUGINS

SonarQube Scanner

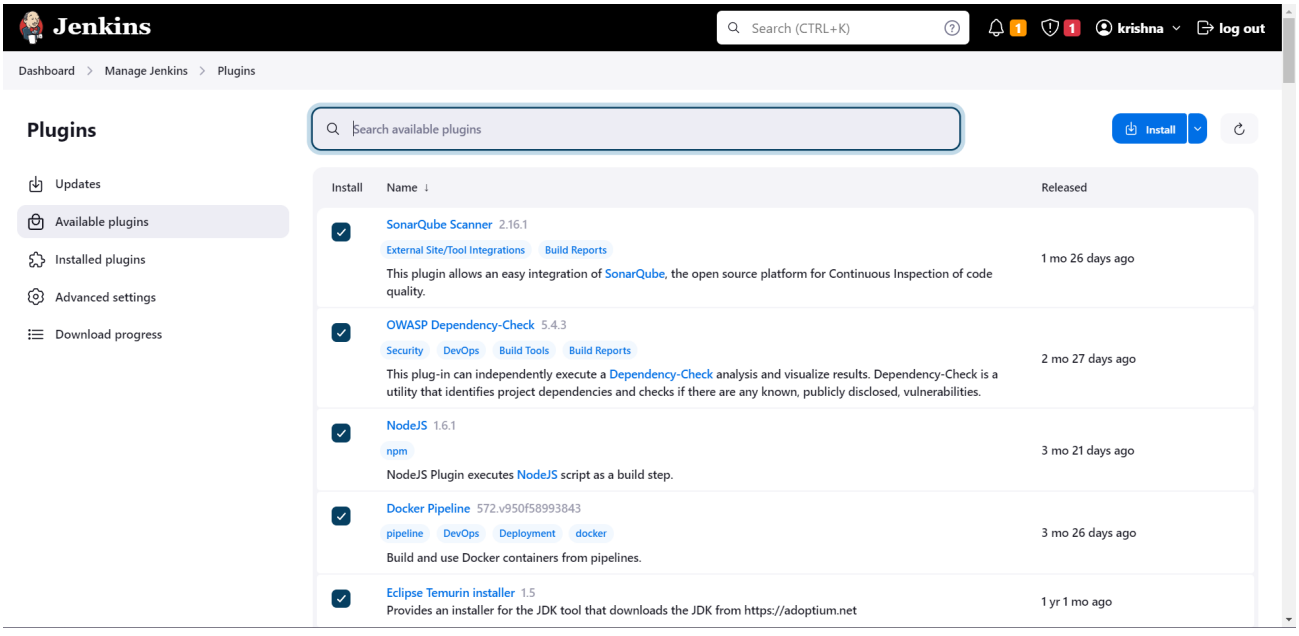
NODEJS

OWASP DEPENDENCY CHECK

DOCKER PIPELINE

Eclipse Temurin installer

Dashboard>Manage Jenkins>Plugins>Available plugins>



SETUP SONAR USING DOCKER(container):

Image-name:lts-community

docker run -d --name sonar -p 9000:9000 sonarqube:lts-community

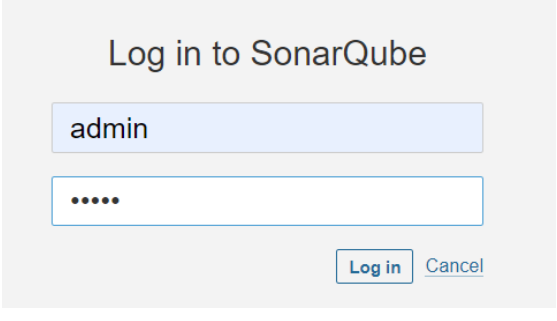
After creating the sonar container, access the sonarqube with Public-Ip:9000 port number.

Login to the sonar dashboard with the following and credentials After entering the credentials

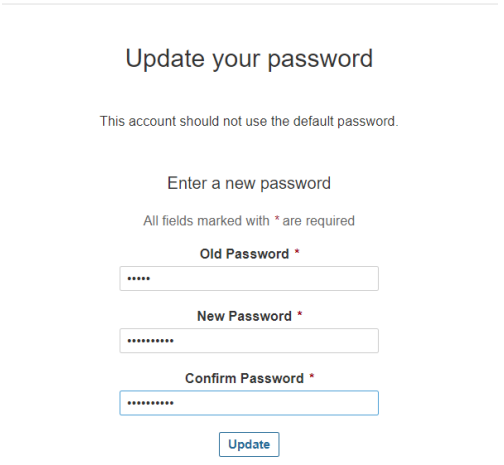
we have to set a new password.

username-admin

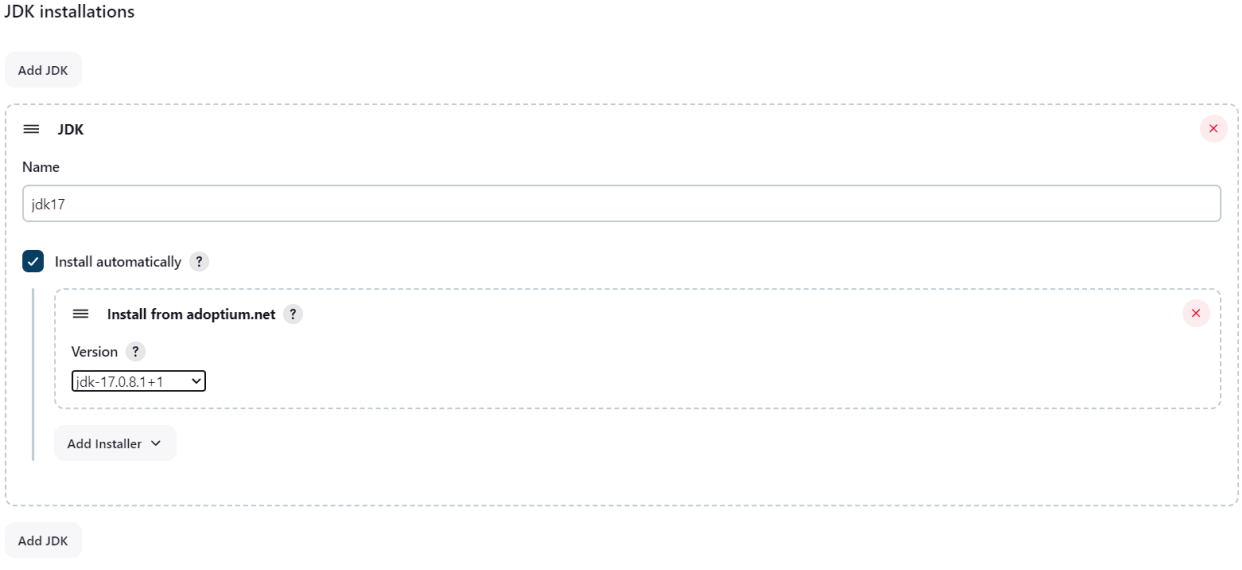
password-admin

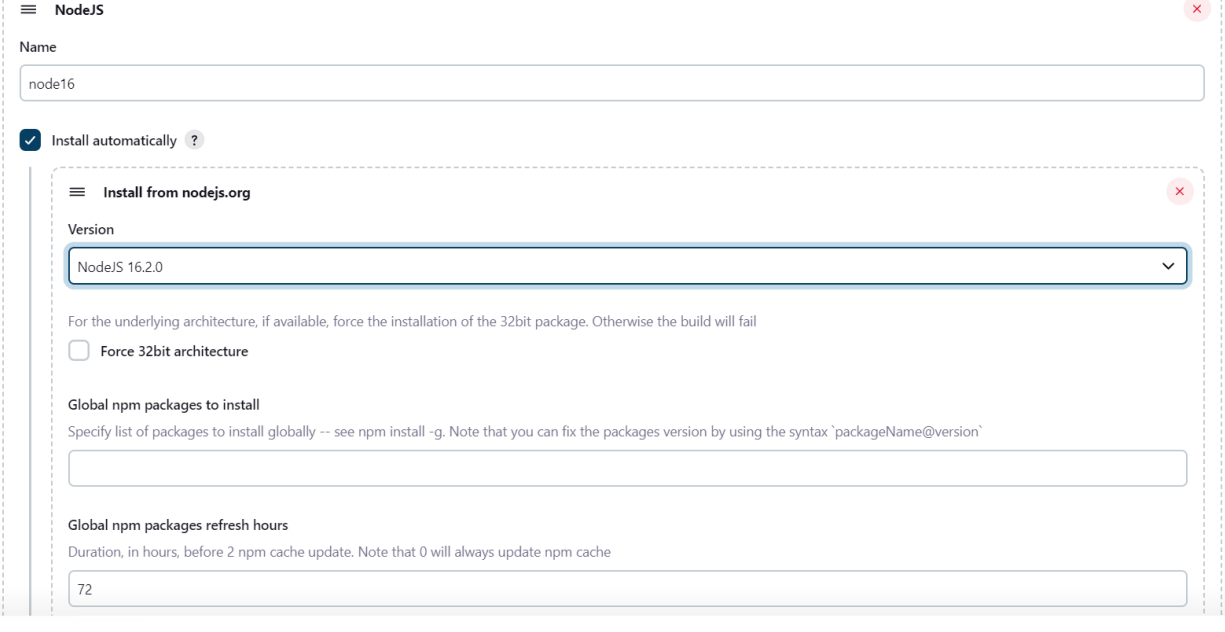


Update the password



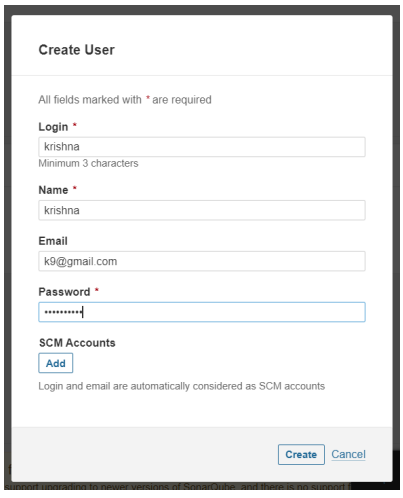
Now configure NodeJs, Java & DP-Check:

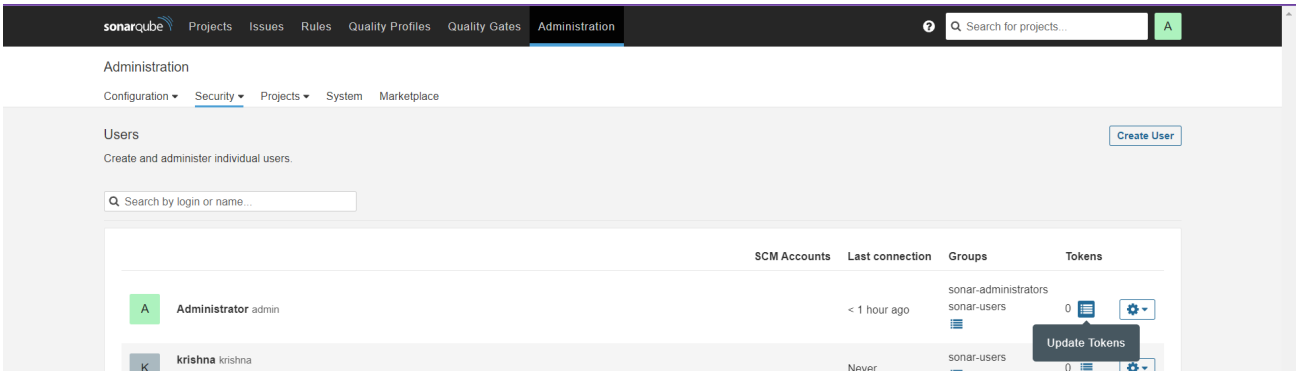


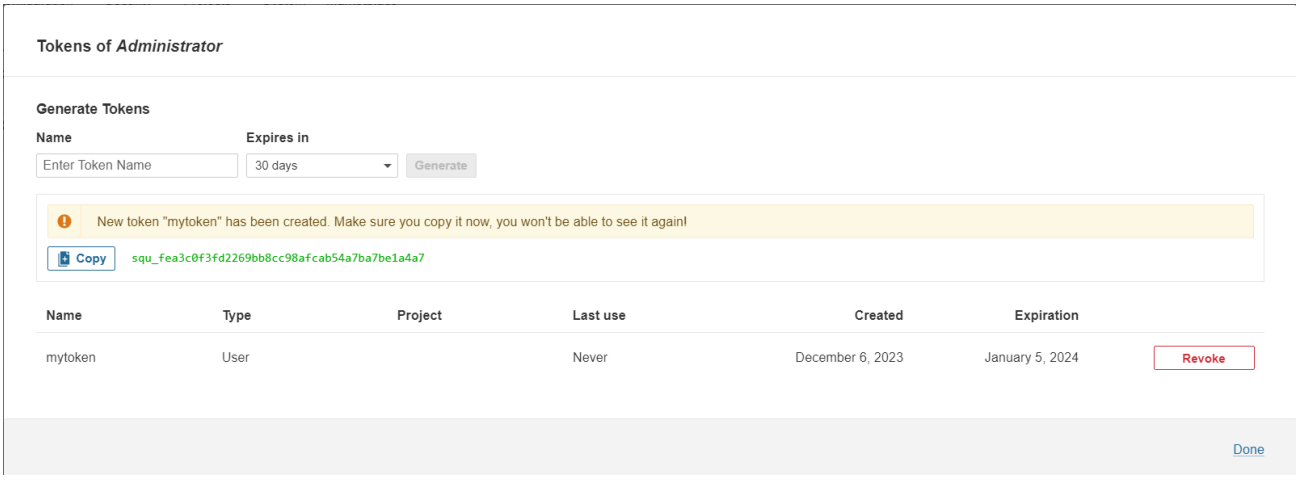


CONFIGURE ALL THE PLUGINS INTO JENKINS:

Goto your Sonarqube Server. Click on Administration ----→ Security ----→ Users → Click on Tokens and Update Token ----→ Give it a name ----→ and click on Generate Token.

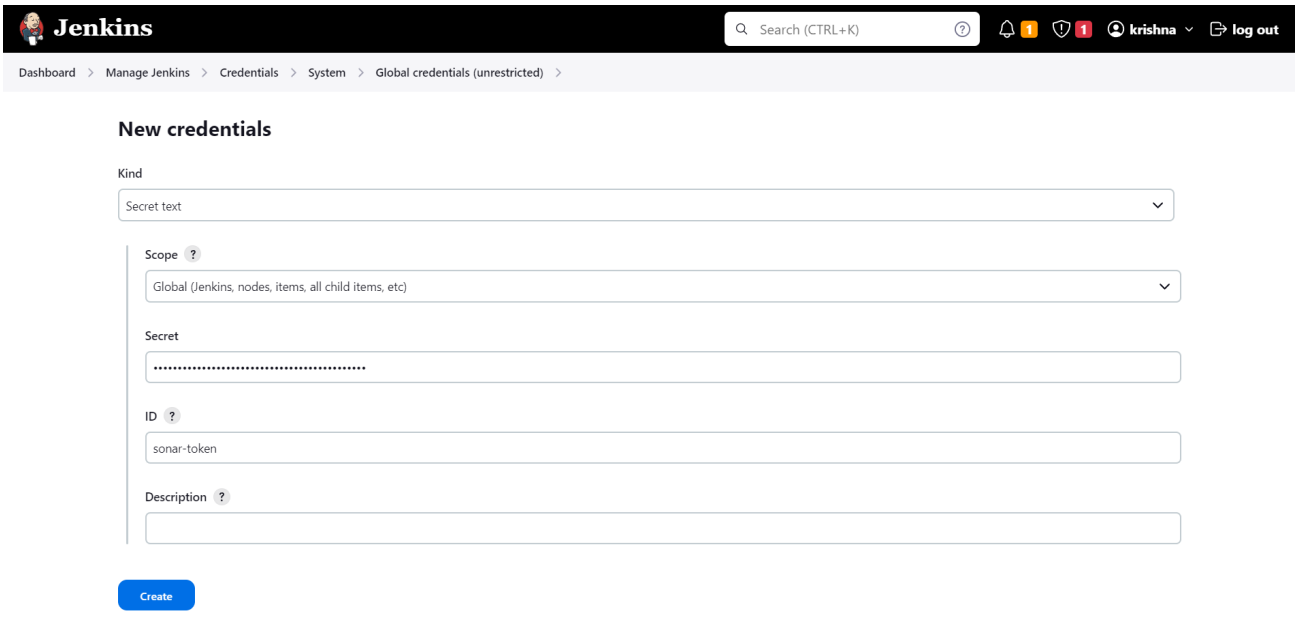




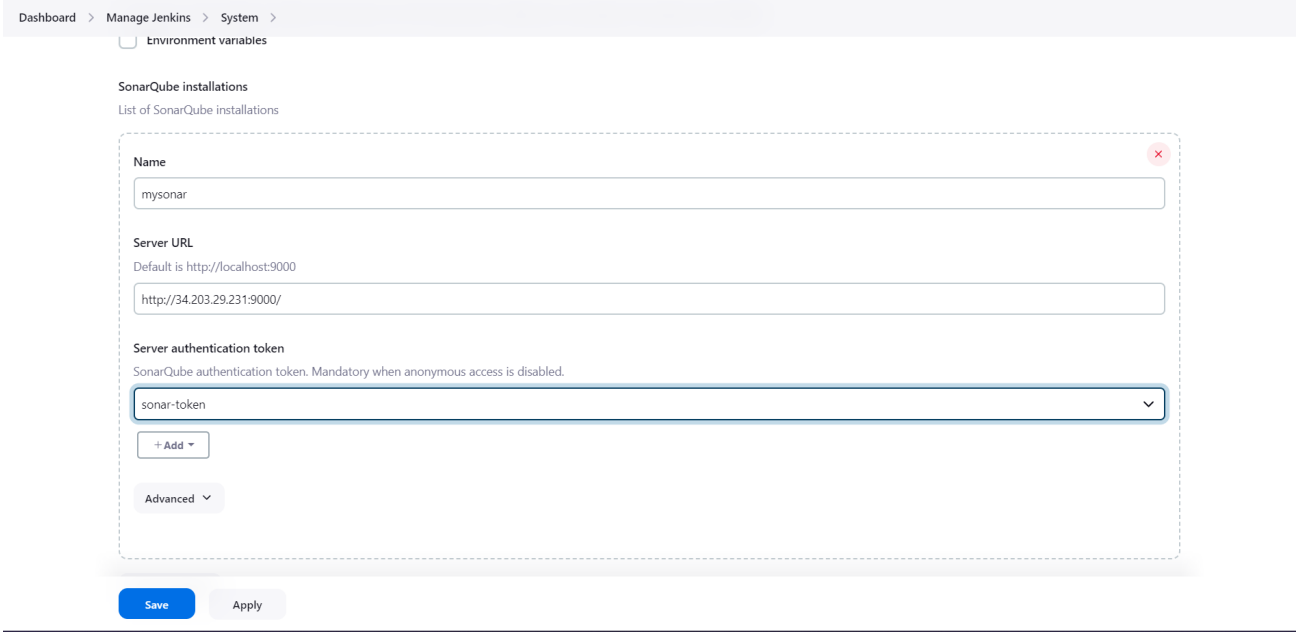


copy Token

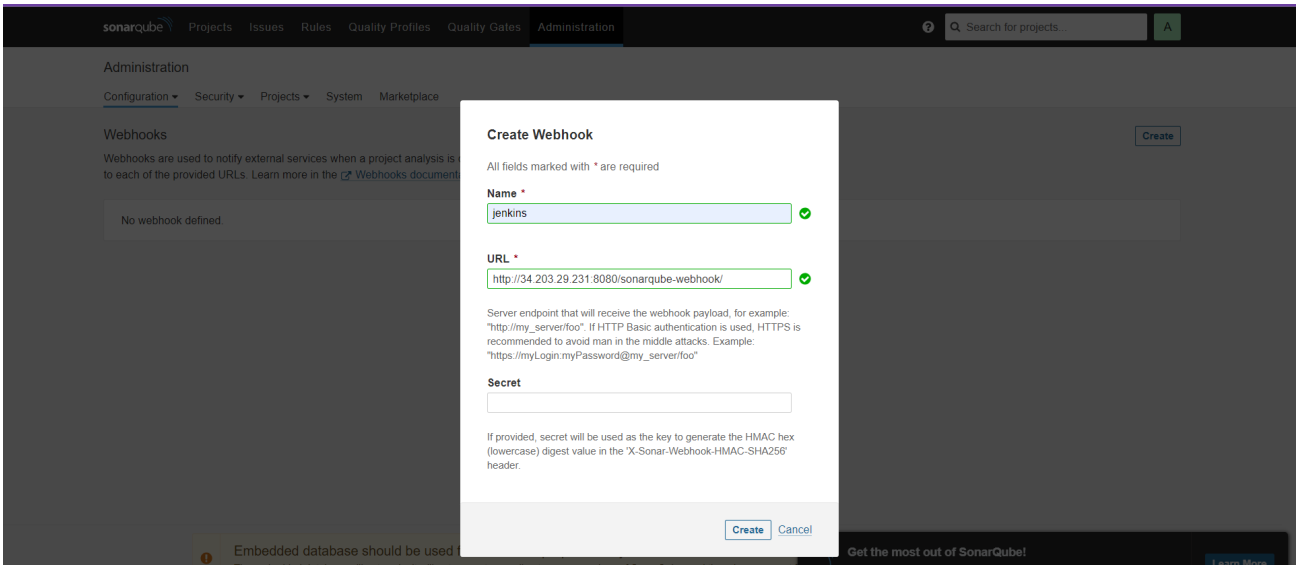
Goto Jenkins Dashboard ----> Manage Jenkins ----> Credentials ----> Add Secret Text with id



In jenkins



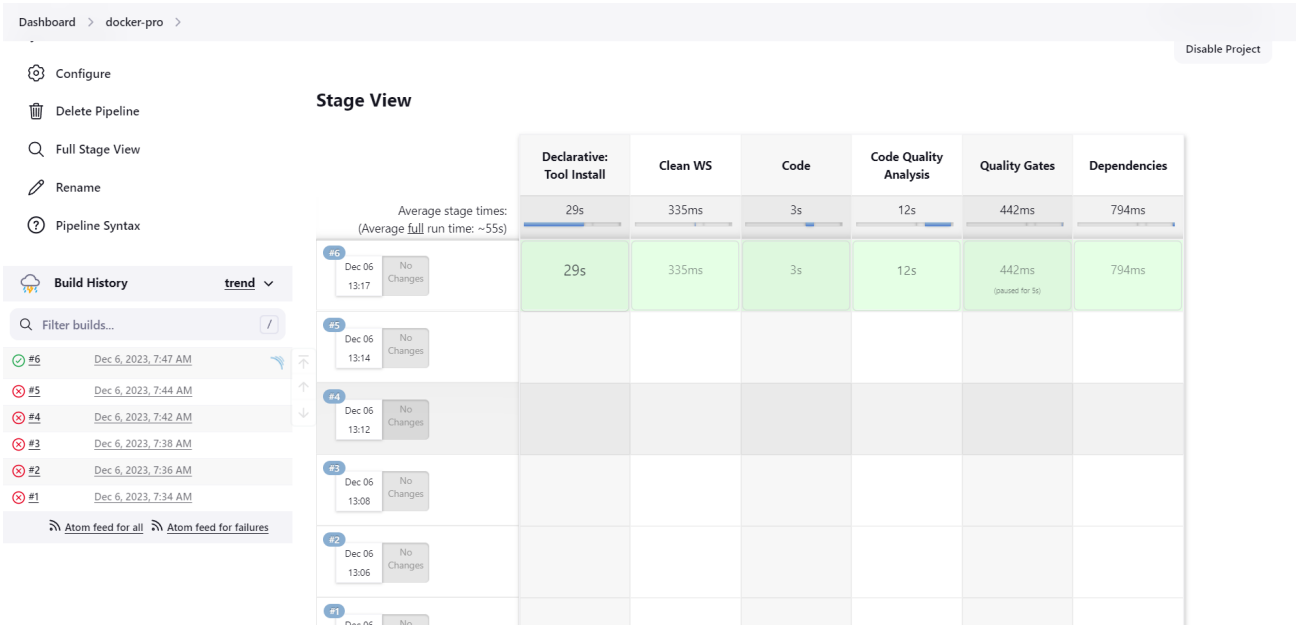
Create webhook on Sonarqube

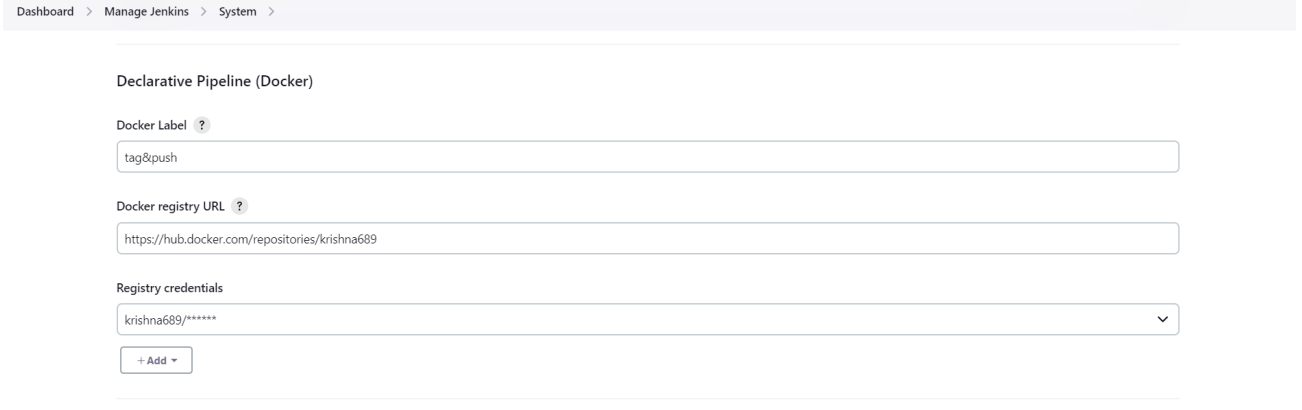


Configure script

```
pipeline {
    agent any
    tools {
        jdk 'jdk17'
        nodejs 'node16'
    }
    environment {
        SCANNER_HOME=tool 'mysonar'
    }
    stages {
        stage ("Clean WS") {
            steps {
                cleanWs ()
            }
        }
        stage ("Code") {
```

```
        steps {
            git branch: 'main', url: 'https://github.com/devops0014/Zomato-Project.git'
        }
    }
    stage ("Code Quality Analysis") {
        steps {
            withSonarQubeEnv('mysonar') {
                sh ''' $SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=zomato \
                -Dsonar.projectKey=zomato '''
            }
        }
    }
    stage ("Quality Gates") {
        steps {
            script {
                waitForQualityGate abortPipeline: false; credentialsId: 'sonar-token'
            }
        }
    }
    stage ("Dependencies") {
        steps {
            sh 'npm install'
        }
    }
  }
}
```





Install all docker plugins:
Dashboard ----> Manage Jenkins ----> Available Plugins

```
pipeline {
  agent any
  tools {
    jdk 'jdk17'
    nodejs 'node16'
  }
  environment {
    SCANNER_HOME=tool 'mysonar'
```

```
}
stages {
    stage ("Clean WS") {
        steps {
            cleanWs ()
        }
    }
    stage ("Code") {
        steps {
            git 'https://github.com/devops0014/Zomato-Project.git'
        }
    }
    stage ("Code Quality Analysis") {
        steps {
            withSonarQubeEnv('mysonar') {
                sh ''' $SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=zomato \
                -Dsonar.projectKey=zomato '''
            }
        }
    }
    stage ("Quality Gates") {
        steps {
            script {
                waitForQualityGate abortPipeline: false; credentialsId: 'sonar-token'
            }
        }
```

```
    }
    stage ("Dependencies") {
      steps {
        sh 'npm install'
      }
    }
    stage ("OWASP") {
      steps {
        dependencyCheck additionalArguments: '--scan ./ --disableYarnAudit --
disableNodeAudit',odcInstallation: 'DP-Check'
        dependencyCheckPublisher pattern: '**/dependency-check-report.xml'
      }
    }
    stage ("trivy") {
      steps {
        sh 'trivy fs . > trivyfs.txt'
      }
    }
    stage ("Build") {
      steps {
        sh 'docker build -t image1 .'
      }
    }
    stage ("Tag&Push") {
      steps {
        script {
          withDockerRegistry(credentialsId: 'DockerPass') {
```

```
            sh 'docker tag image1 krishna689/mydockerproject:myzomatoimage'
            sh 'docker push krishna689/mydockerproject:myzomatoimage'
        }
      }
   }
}
stage ("Scan the image") {
   steps {
      sh 'trivy image krishna689/mydockerproject:myzomatoimage'
```