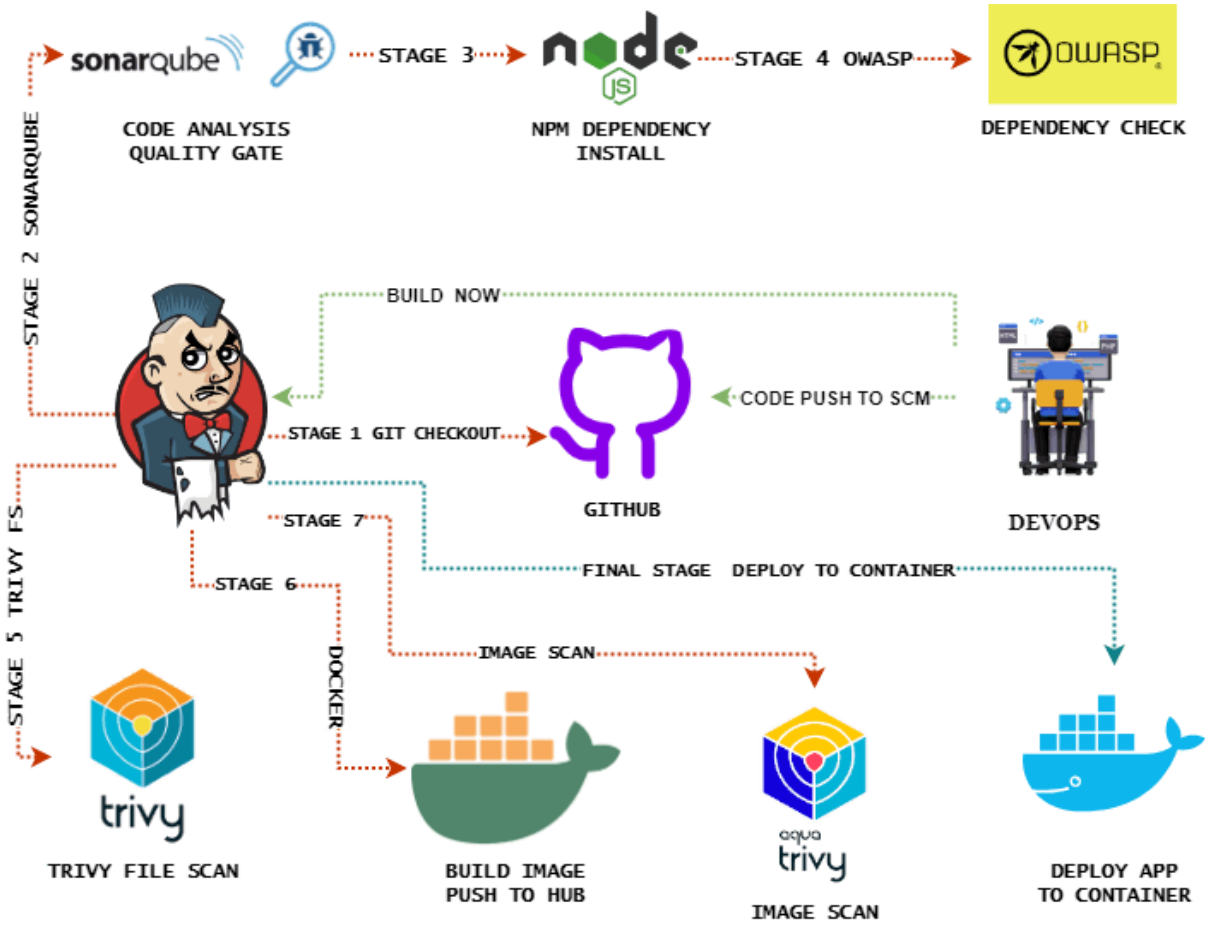
SECURE DEPLOYMENT OF ZOMATO CLONE WITH DEVSECOPS

BY

MANICHANDU K.



STEP-1(launching ubuntu instance):

- Go to the AWS console and launch an instance of type "t2.large" and an image of Ubuntu.
- Open all-traffic in the security group of the EC2 instance.
- Now SSH into the instance and continue to step-2.

STEP-2(Installing Jenkins, Trivy, and Docker):

- First become a root user by giving this command.

sudo -i

- And follow these commands.

apt update -y && apt install default-jdk -y

*sudo wget -O /usr/share/keyrings/jenkins-keyring.asc *

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

*echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]" *

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

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

sudo apt-get update

sudo apt-get install Jenkins -y

sudo systemctl enable Jenkins

sudo systemctl start Jenkins

sudo systemctl status Jenkins

- Now access the Jenkins server by giving the public-ip of the ec2 instance add port 8080 at the end.

  http://<public-ip>:8080

- Unlock Jenkins by giving the initial password.

  cat /var/lib/Jenkins/secrets/initialAdminPassword

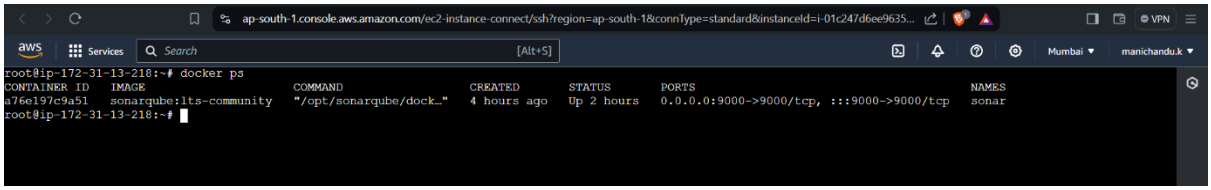
- Install the suggested plug-ins and you are ready to go.

- After installing Jenkins now it is time to install docker and it is simply done by giving this command.

sudo apt install docker.io -y

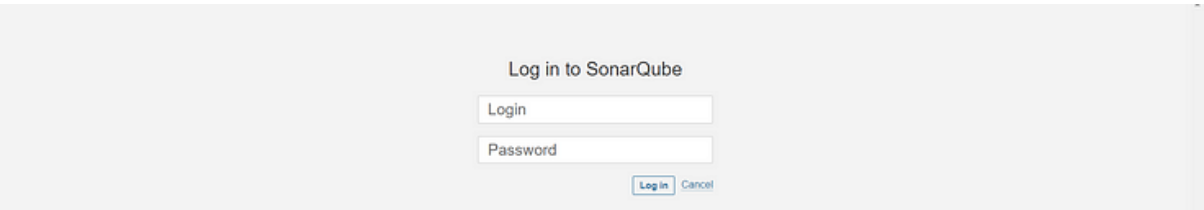
- After the docker installation, we create a sonarqube container.

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



- Now grab the public-ip of instance with port 9000 to access the sonarqube.

http://<public-ip>:9000



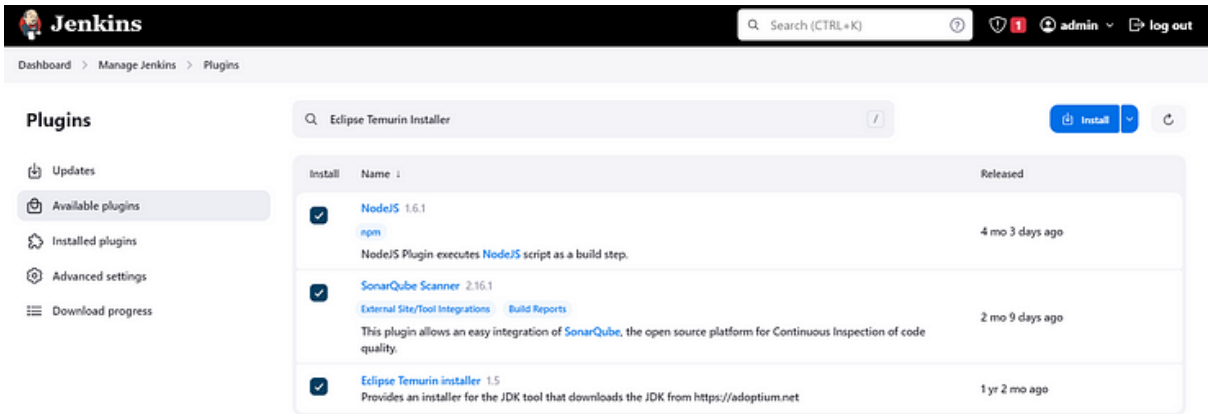
- Now give admin as both username and password. Change the password as you like later.

- Now install trivy in our instance/

sudo apt-get install wget apt-transport-https gnupg lsb-release -y
wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | gpg --dearmor | sudo tee /usr/share/keyrings/trivy.gpg > /dev/null
echo "deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasecurity.github.io/trivy-repo/deb $(lsb_release -sc) main" | sudo tee -a /etc/apt/sources.list.d/trivy.list
sudo apt-get update
sudo apt-get install trivy -y

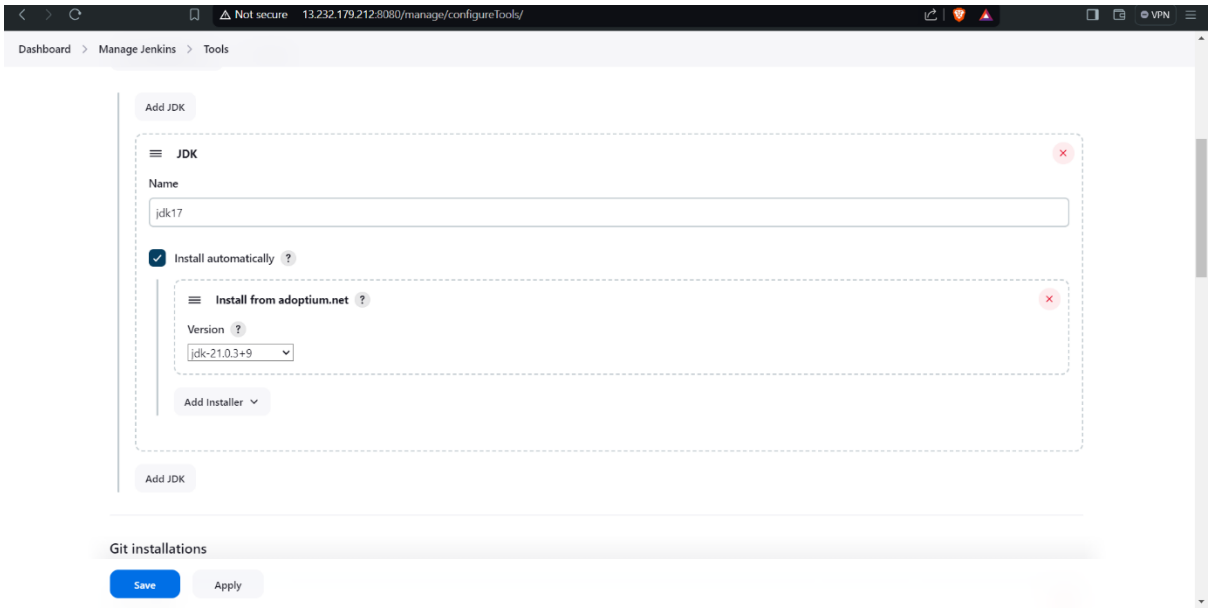
STEP-3:

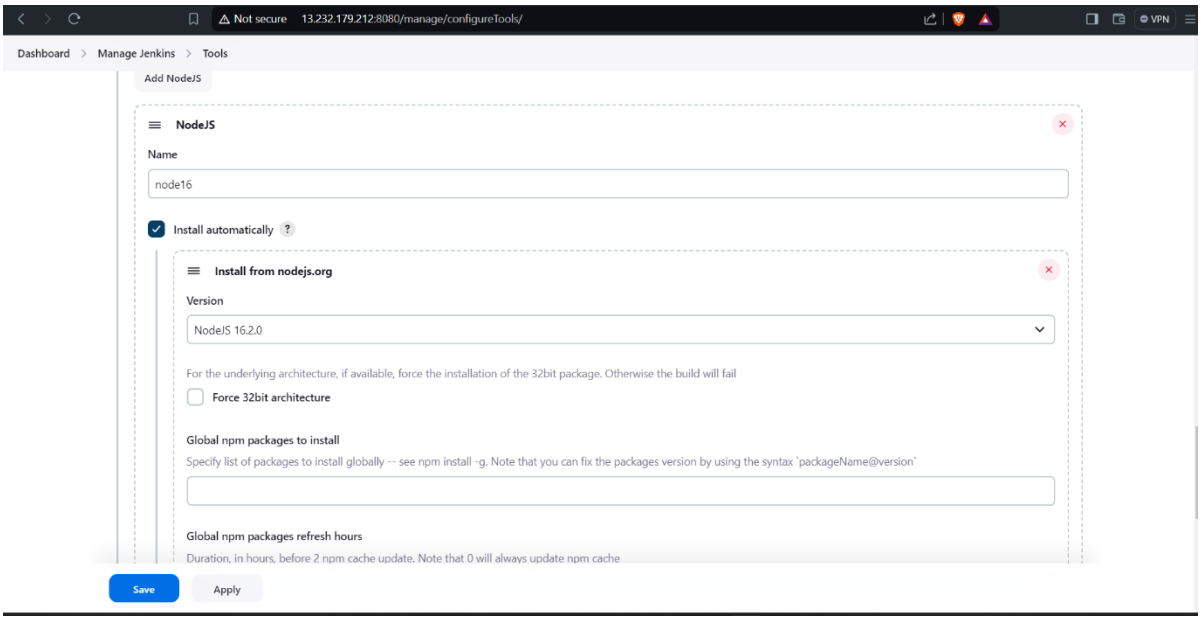
- Now we have to install the necessary plug-ins for the pipeline.



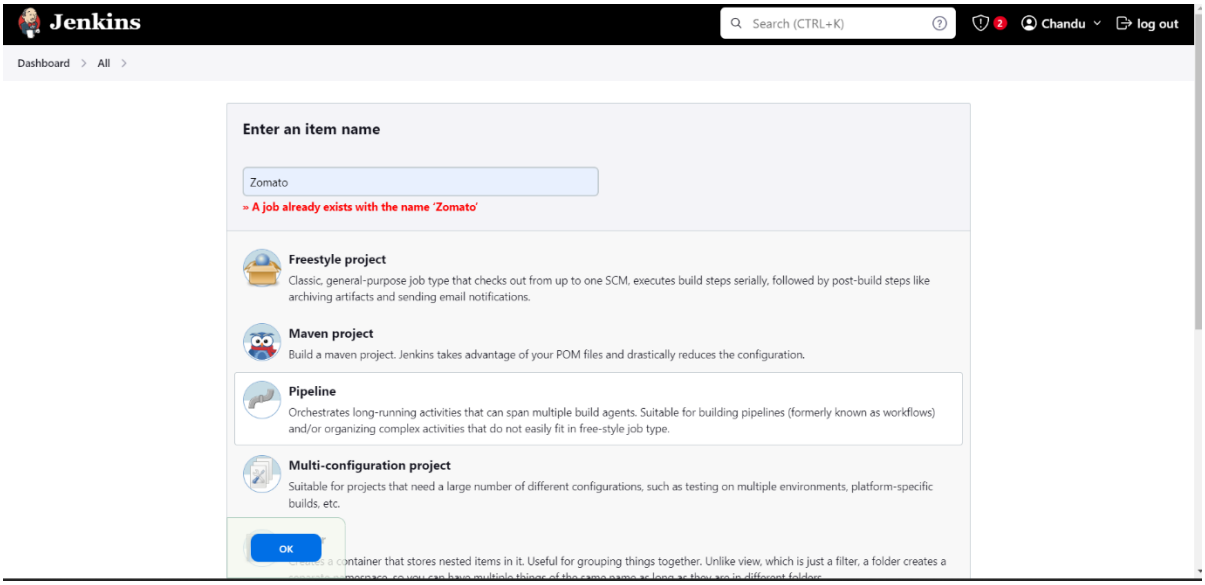
- Now configure nodejs and java in global tool configuration.

- Goto Manage Jenkins → Tools → Install JDK(17) and NodeJs(16)→ Click on Apply and Save.



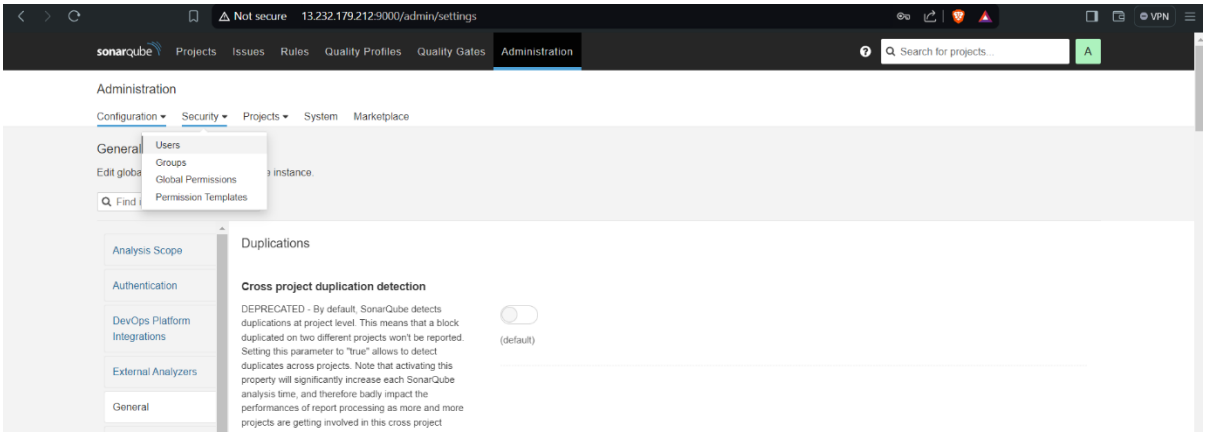


- Create a job as Zomato Name, select pipeline and click on ok.

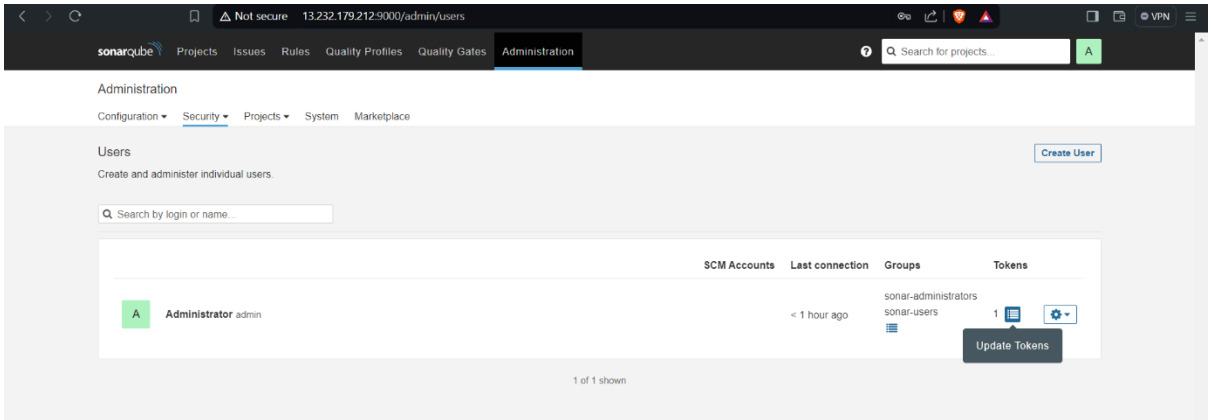


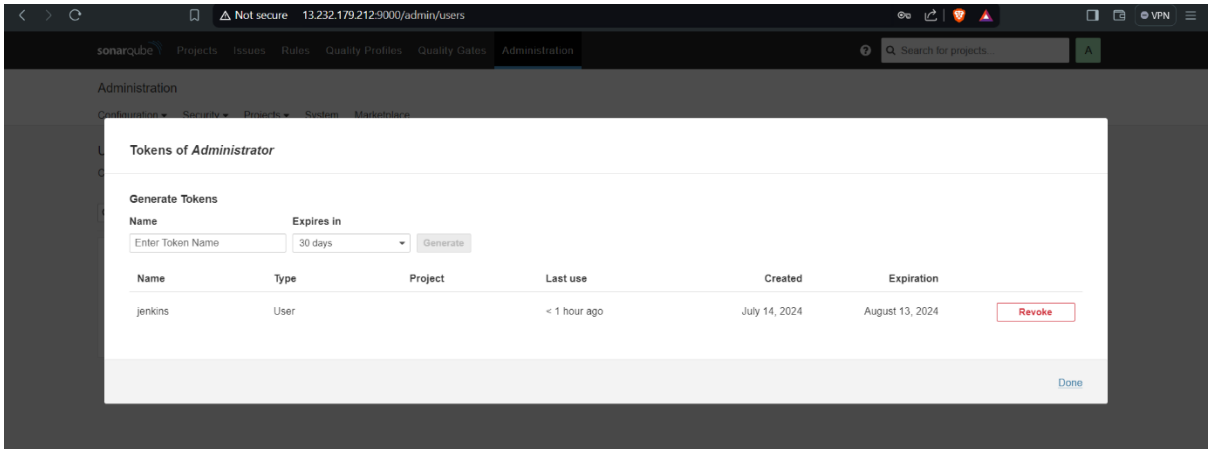
STEP-4:

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

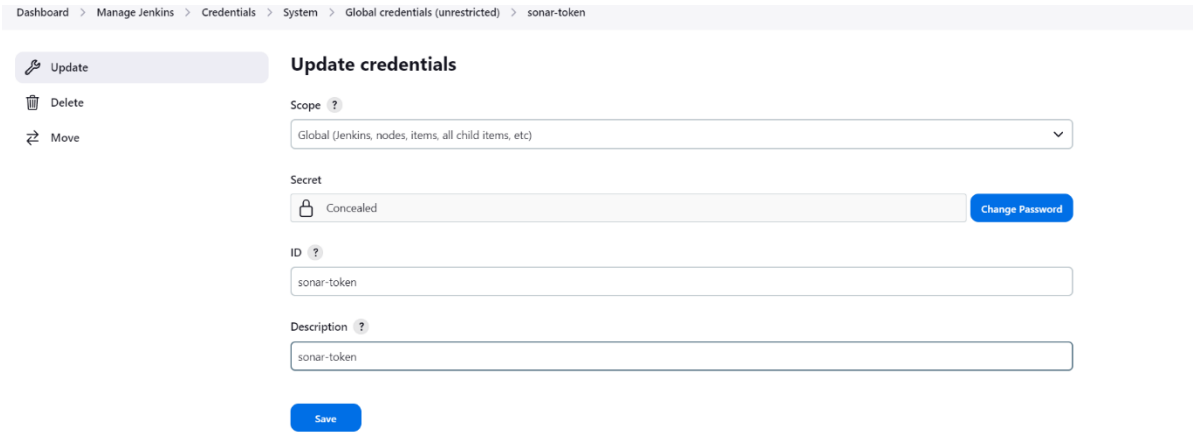


- Click on update token.
- Enter the token name as Jenkins and click on create.



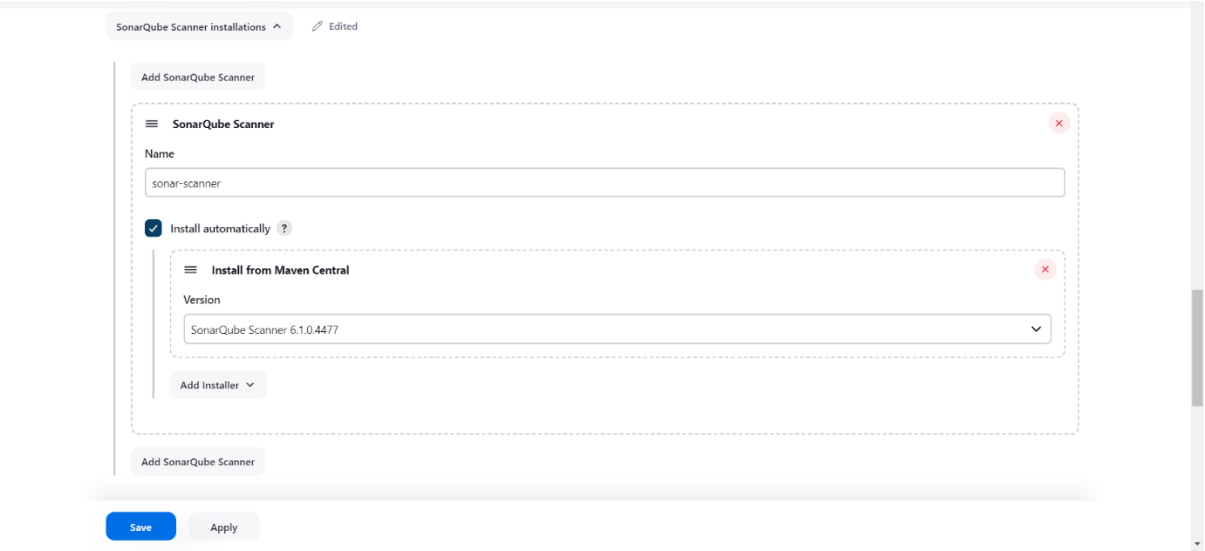


- Copy the token and come to the Jenkins server, go to Jenkins Dashboard → Manage Jenkins → Credentials → Add Secret Text. It should look like this

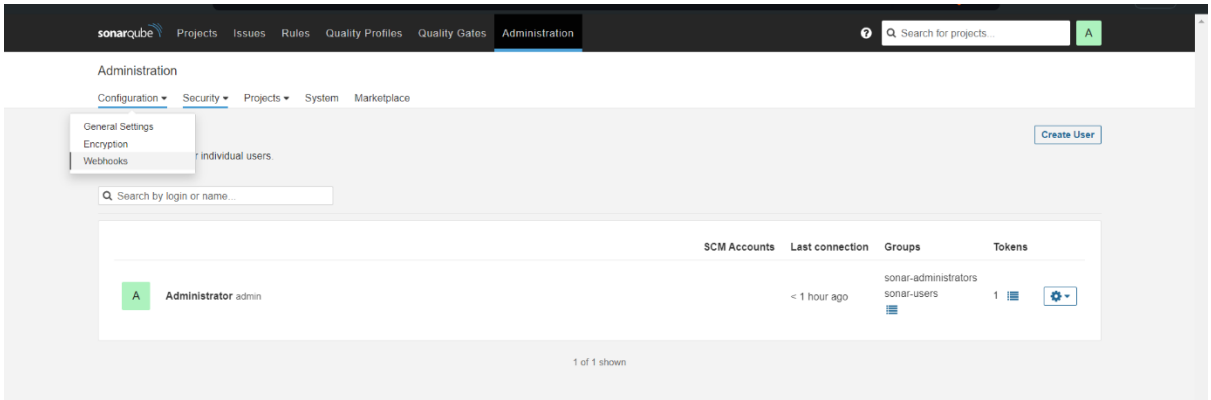


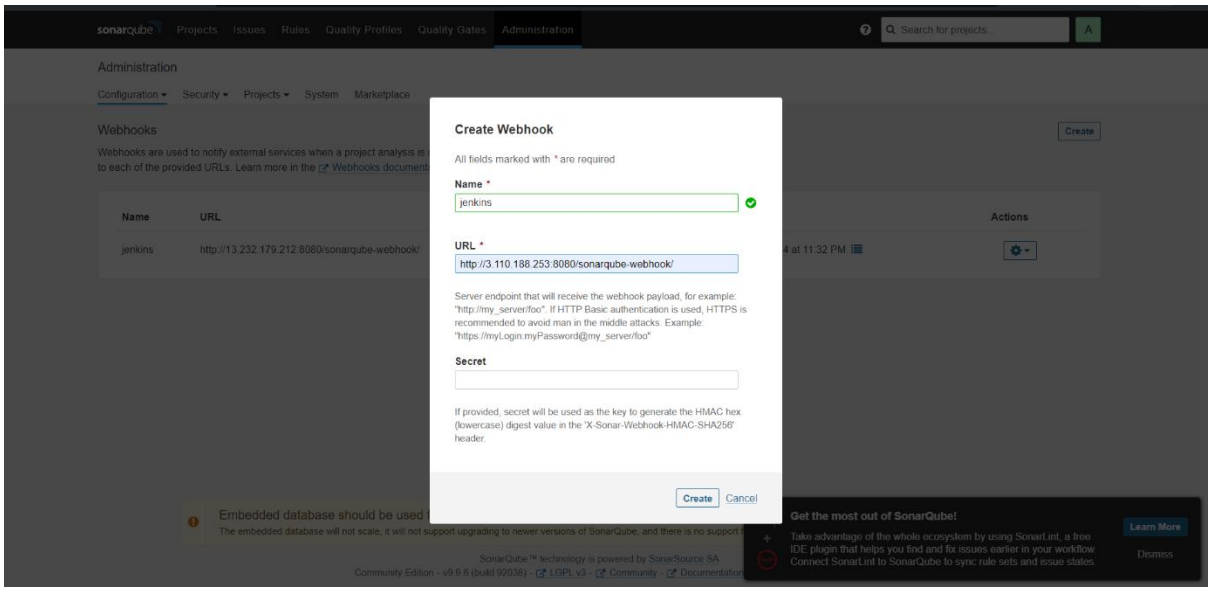
- Now, go to Dashboard → Manage Jenkins → System and Add like the below image.

- Click on apply and save. Now we also have to add sonarqube tool to our global tools.



- In sonarqube we also have to add a quality gate. Go to Administration → Configuration →Webhooks in sonarqube server.





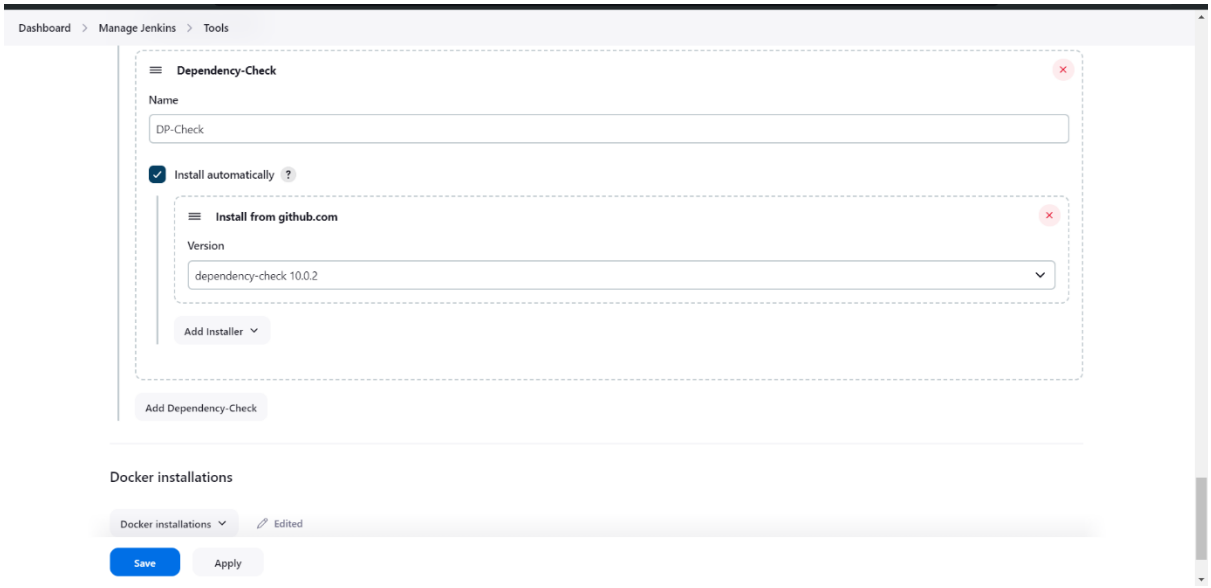
- Click on create and add in url section of quality gate

<http://jenkins-public-ip:8080>/sonarqube-webhook/

STEP-5(Install OWASP dependency):

- Go to Dashboard → Manage Jenkins → Plugins → OWASP Dependency-Check. Click on it and install it without restart.

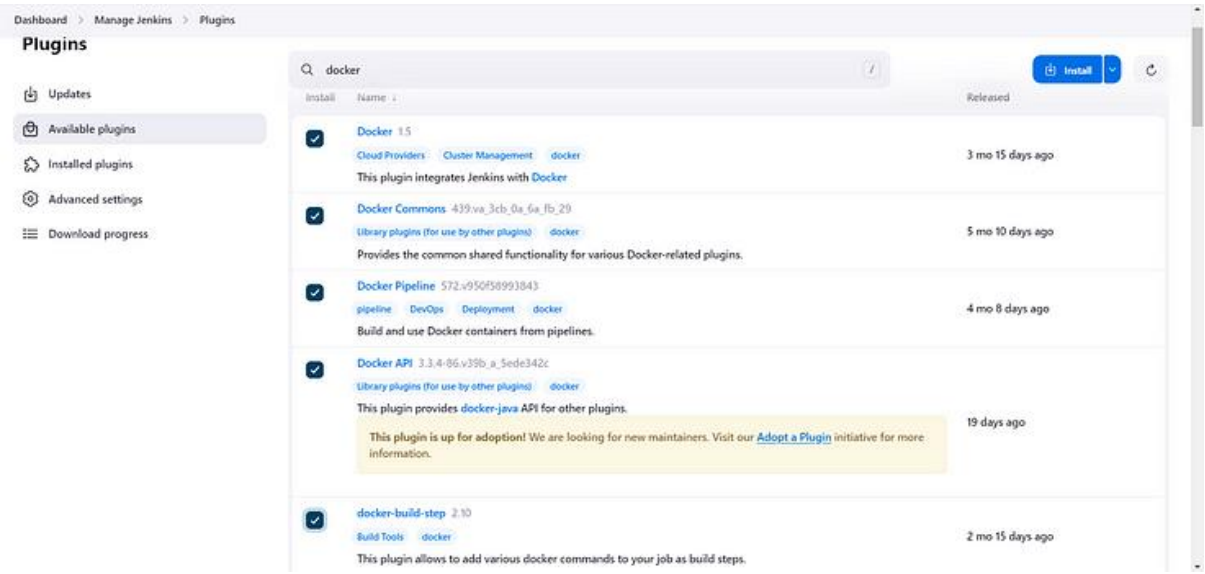
- Now configure the dependency tool in our global tools.



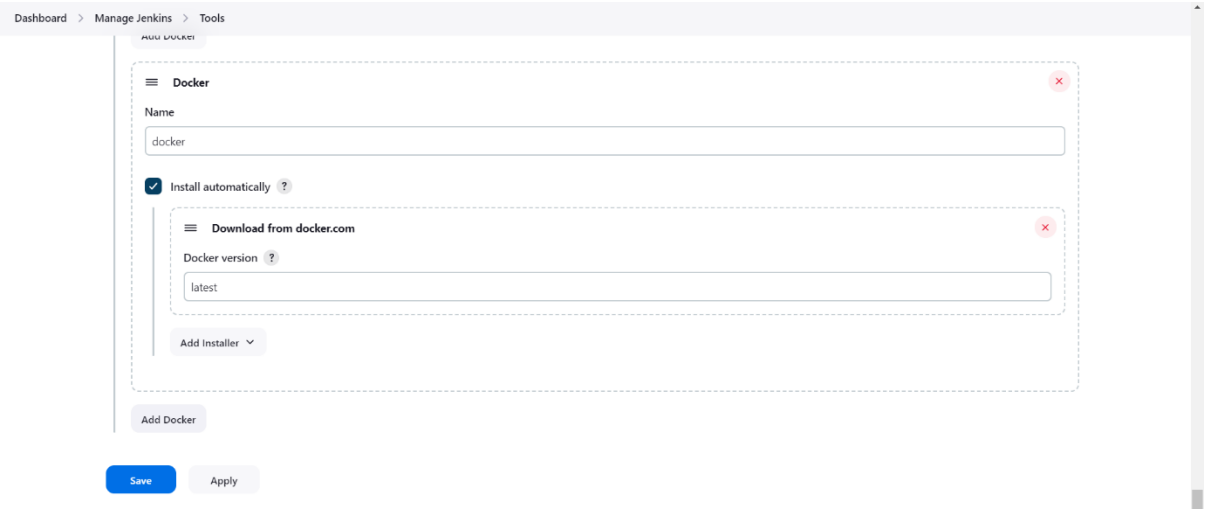
- Click on apply and save.

STEP-6(Docker Integration):

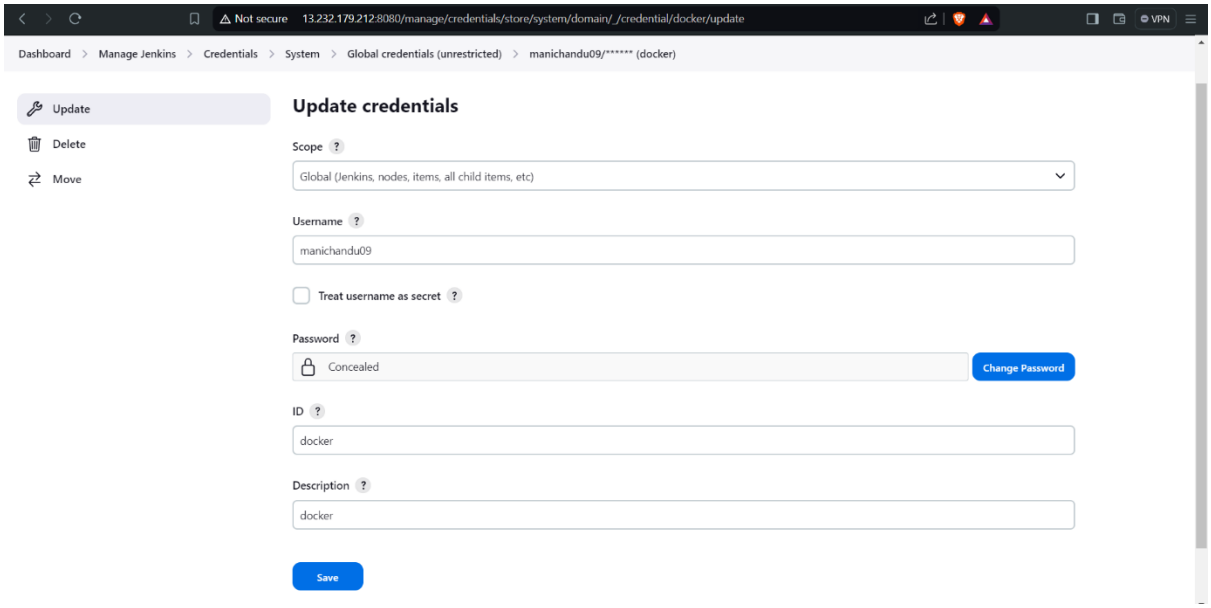
- We need to install the Docker tool in our system, Goto Dashboard → Manage Plugins → Available plugins → Search for Docker and install these plugins



- Click on Install and configure docker tool in global tools.



- Go to the credentials section and add docker-hub credentials with username and password.

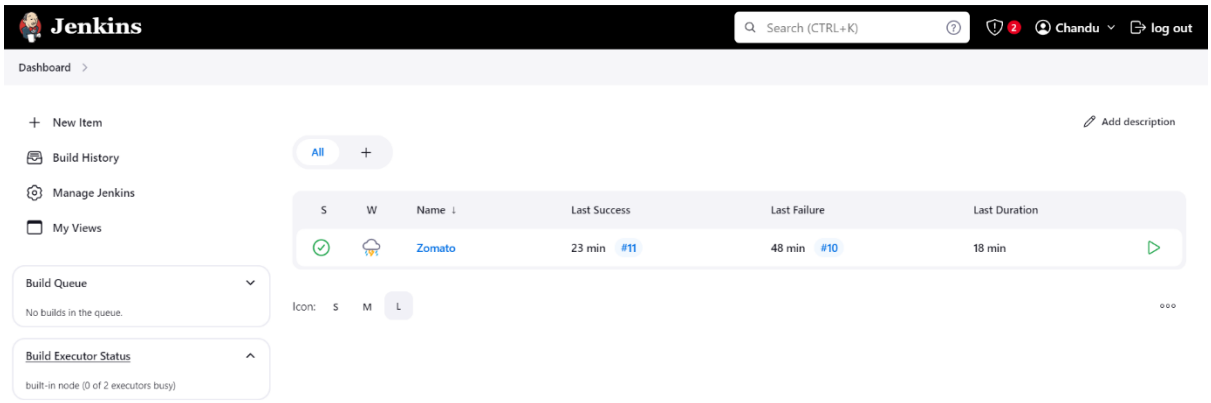


- Now add this script to the pipeline.

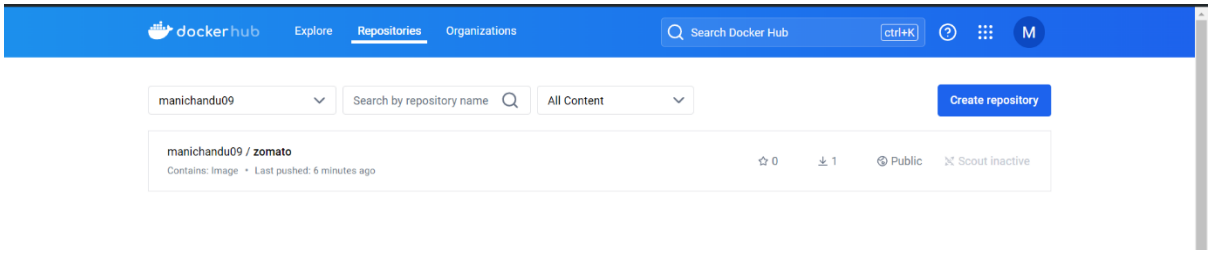
```
pipeline{
    agent any
    tools{
        jdk 'jdk17'
        nodejs 'node16'
    }
    environment {
        SCANNER_HOME=tool 'sonar-scanner'
    }
    stages {
        stage('clean workspace'){
            steps{
                cleanWs()
            }
        }
        stage('Checkout from Git'){
            steps{
                git branch: 'main', url: 'https://github.com/mudit097/Zomato-Clone.git'
            }
        }
        stage("Sonarqube Analysis "){
            steps{
                withSonarQubeEnv('sonar-server') {
                    sh ''' $SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=zomato \
                    -Dsonar.projectKey=zomato '''
                }
            }
        }
        stage("quality gate"){
           steps {
                script {
                    waitForQualityGate abortPipeline: false, credentialsId: 'sonar-token'
                }
            }
        }
        stage('Install Dependencies') {
            steps {
                sh "npm install"
            }
        }
```

```
        }
        stage('Install Dependencies') {
            steps {
                sh "npm install"
            }
        }
        stage('OWASP FS SCAN') {
            steps {
                dependencyCheck additionalArguments: '--scan ./ --disableYarnAudit --disableNodeAudit', odcInstallation: 'DP-Check'
                dependencyCheckPublisher pattern: '**/dependency-check-report.xml'
            }
        }
        stage('TRIVY FS SCAN') {
            steps {
                sh "trivy fs . > trivyfs.txt"
            }
        }
        stage("Docker Build & Push"){
            steps{
                script{
                   withDockerRegistry(credentialsId: 'docker', toolName: 'docker'){
                       sh "docker build -t zomato ."
                       sh "docker tag zomato manichandu09/zomato:latest "
                       sh "docker push manichandu09/zomato:latest "
                   }
                }
            }
        }
        stage("TRIVY"){
            steps{
                sh "trivy image manichandu09/zomato:latest > trivy.txt"
            }
        }
        stage('Deploy to container'){
     steps{
          sh 'docker run -d --name zomato -p 3000:3000 manichandu09/zomato:latest'
        }
      }
    }
}
```

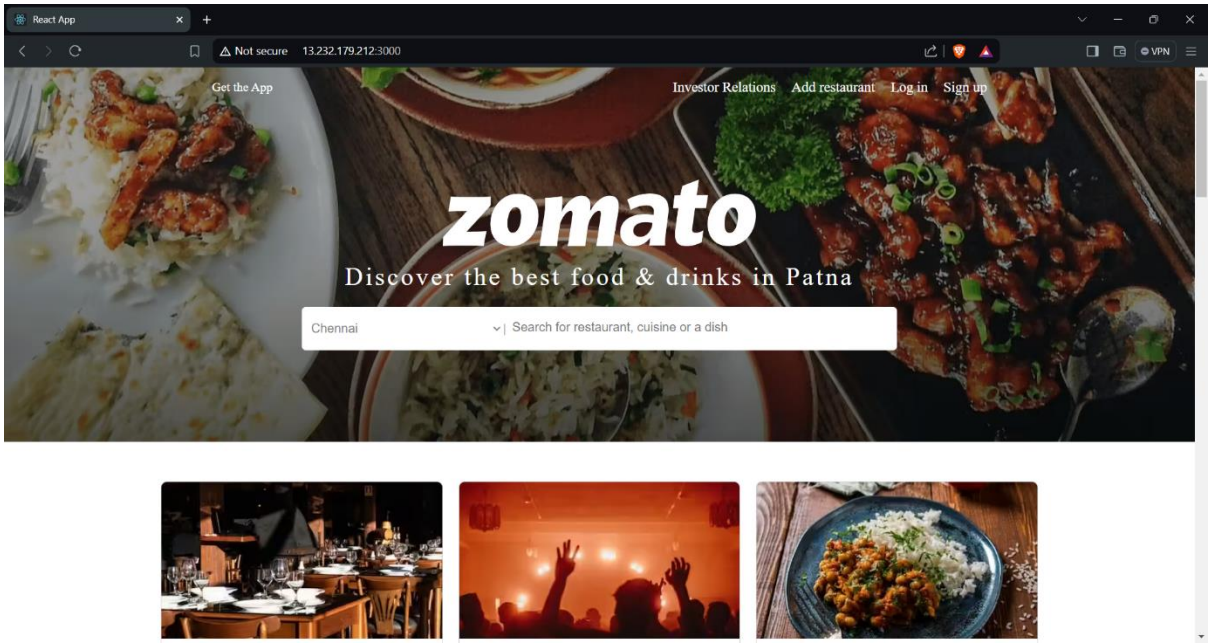
- Configure the pipeline based on your docker credentials and click on build now.



- After getting build success, you will get an output where an image will be pushed into the docker hub.



- Now copy the public ip of the server and paste it in browser with port number 3000.



- You will see the react js app deployed in the docker.