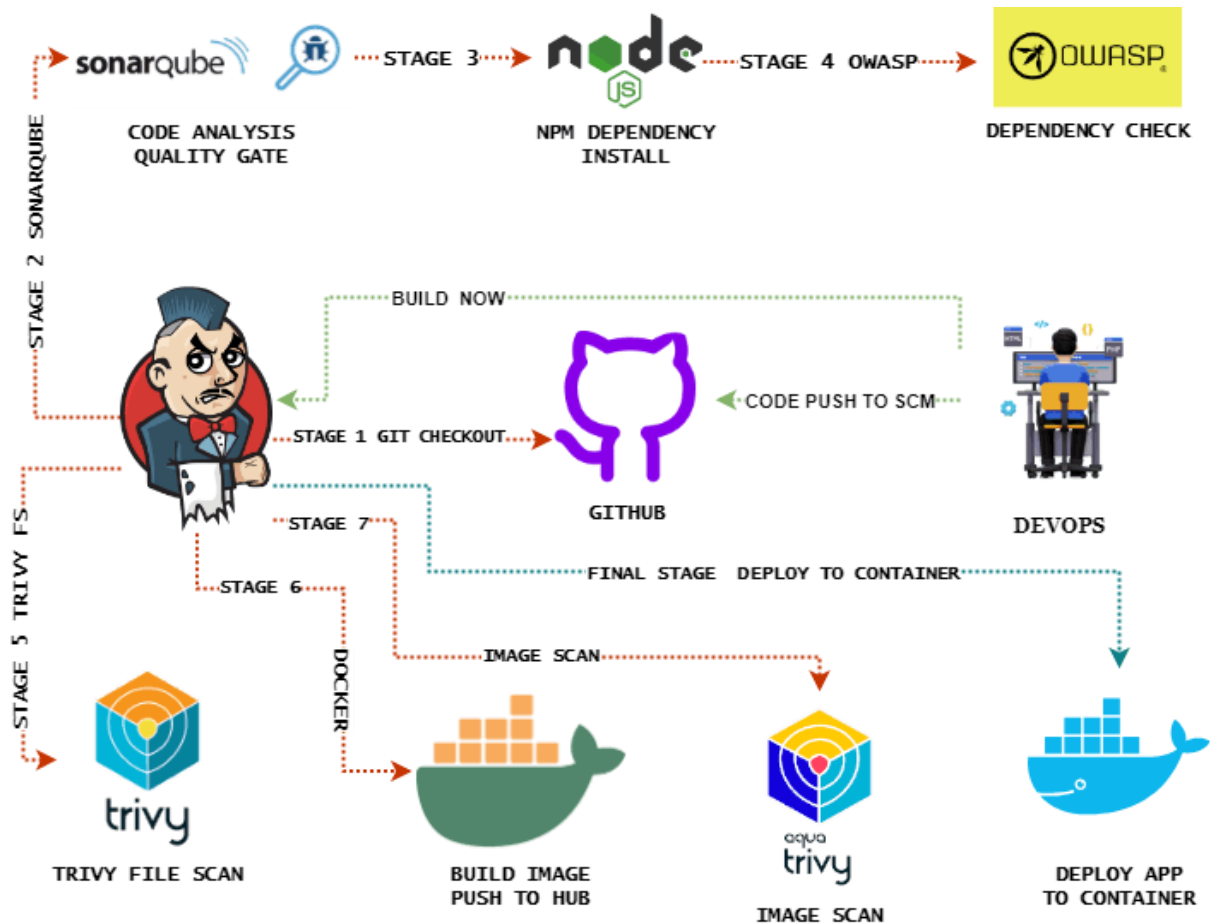


SECURE DEPLOYMENT OF ZOMATO CLONE WITH DEVSECOPS

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
```

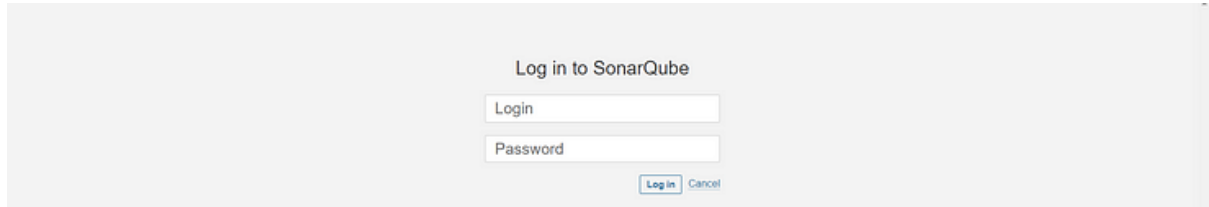
The screenshot shows an AWS Management Console terminal window. The terminal output displays the command `docker ps` and its results in a table format:

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
a76e197c9a51	sonarqube:lts-community	"/opt/sonarqube/dock..."	4 hours ago	Up 2 hours	0.0.0.0:9000->9000/tcp, :::9000->9000/tcp	sonar

The terminal prompt is `root@ip-172-31-13-218:~#`.

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

<http://<public-ip>:9000>



Log in to SonarQube

Login

Password

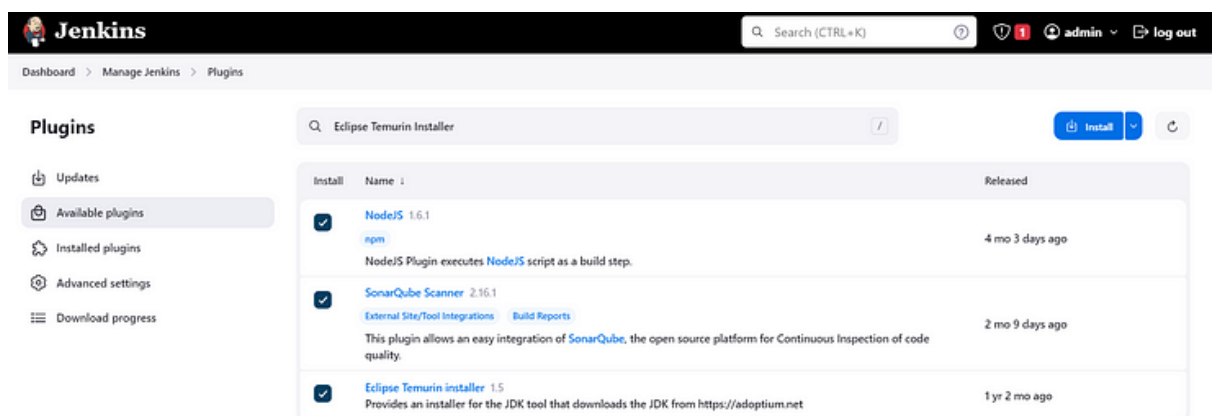
Log in Cancel

- 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.



Jenkins

Search (CTRL+K)

admin log out

Dashboard > Manage Jenkins > Plugins

Plugins

Updates

Available plugins

Installed plugins

Advanced settings

Download progress

Eclipse Temurin Installer

Install	Name	Released
✓	NodeJS 1.6.1 npm NodeJS Plugin executes NodeJS script as a build step.	4 mo 3 days ago
✓	SonarQube Scanner 2.16.1 External Site/Tool Integrations Build Reports This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.	2 mo 9 days ago
✓	Eclipse Temurin installer 1.5 Provides an installer for the JDK tool that downloads the JDK from https://adoptium.net	1 yr 2 mo ago

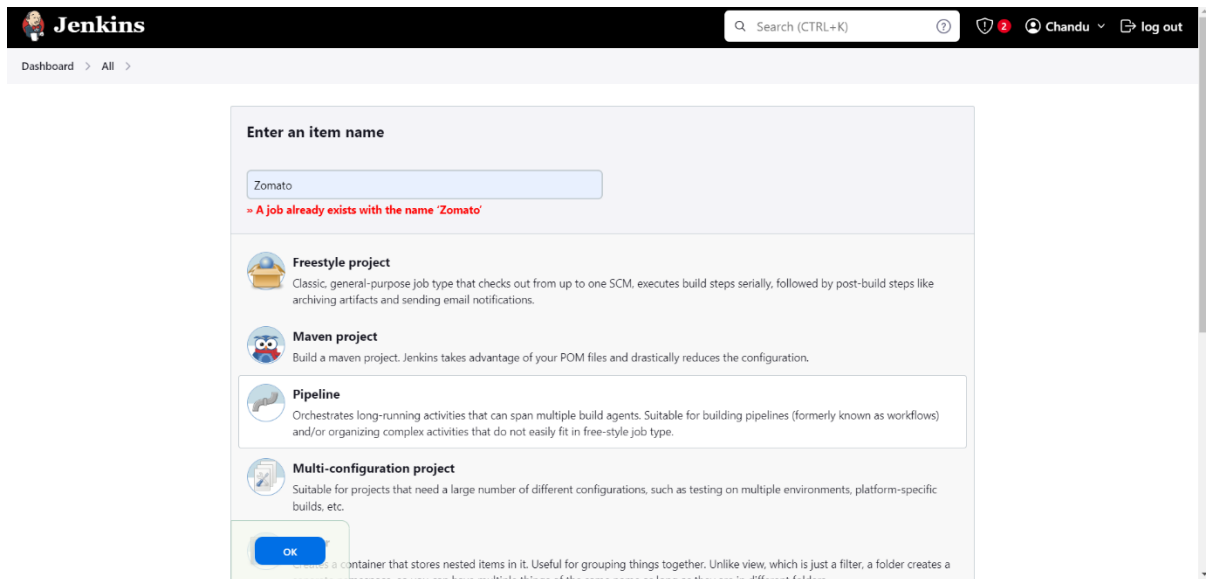
- Now configure nodejs and java in global tool configuration.

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

The screenshot shows the 'Add JDK' configuration page in Jenkins. The breadcrumb navigation is 'Dashboard > Manage Jenkins > Tools'. The page title is 'Add JDK'. There is a 'JDK' section with a 'Name' field containing 'jdk17'. The 'Install automatically' checkbox is checked. Below it, the 'Install from adoptium.net' section is expanded, showing a 'Version' dropdown set to 'jdk-21.0.3+9'. There is an 'Add Installer' button. At the bottom of the configuration area, there is another 'Add JDK' button. Below the configuration area, there is a 'Git installations' section with 'Save' and 'Apply' buttons.

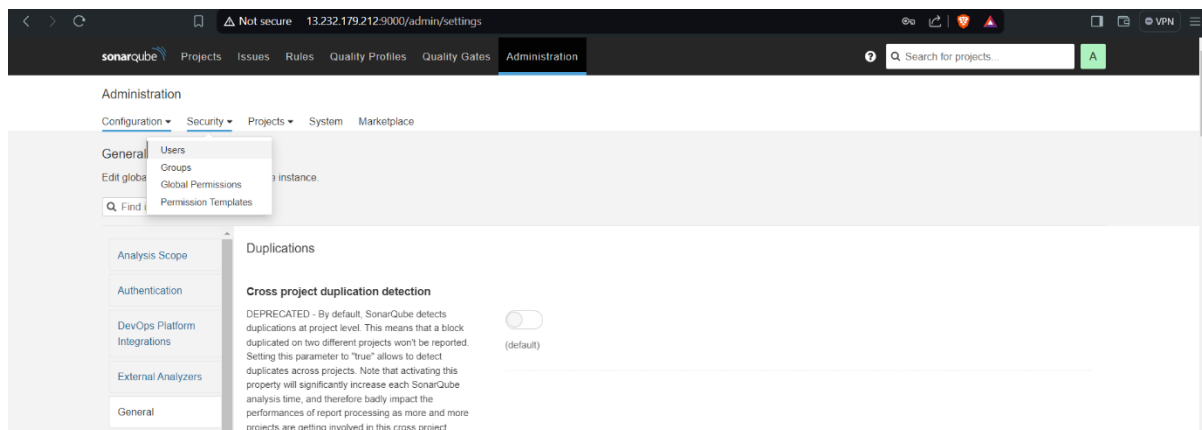
The screenshot shows the 'Add NodeJS' configuration page in Jenkins. The breadcrumb navigation is 'Dashboard > Manage Jenkins > Tools'. The page title is 'Add NodeJS'. There is a 'NodeJS' section with a 'Name' field containing 'node16'. The 'Install automatically' checkbox is checked. Below it, the 'Install from nodejs.org' section is expanded, showing a 'Version' dropdown set to 'NodeJS 16.2.0'. There is a checkbox for 'Force 32bit architecture' which is unchecked. Below that, there is a section for 'Global npm packages to install' with a text area for specifying packages. At the bottom, there is a section for 'Global npm packages refresh hours' with a text area for specifying the duration. At the bottom of the configuration area, there is another 'Add NodeJS' button. Below the configuration area, there is a 'Git installations' section with 'Save' and 'Apply' buttons.

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

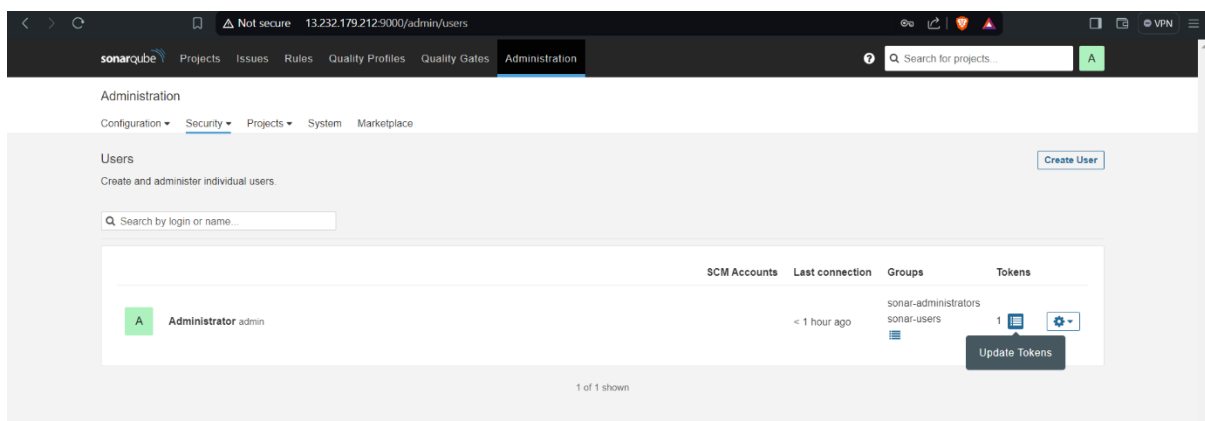


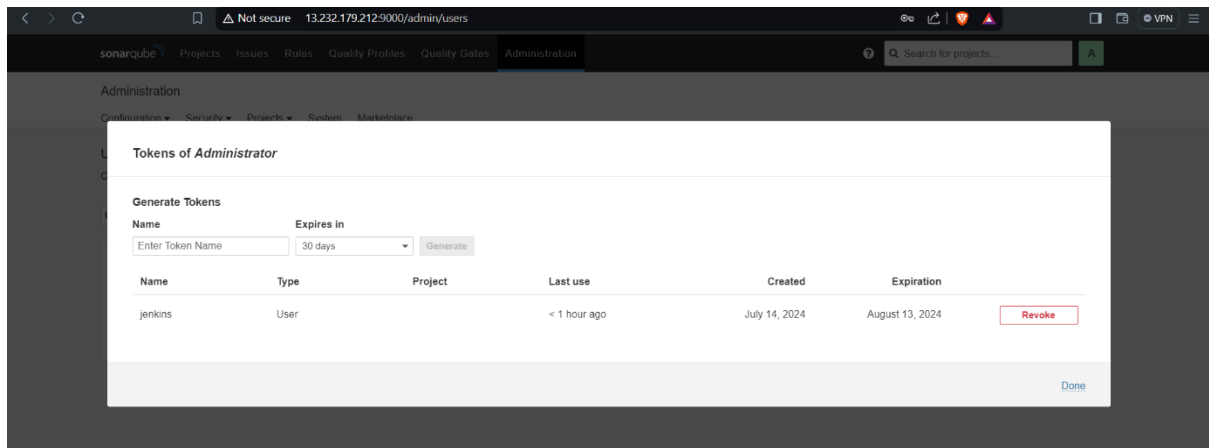
STEP-4:

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

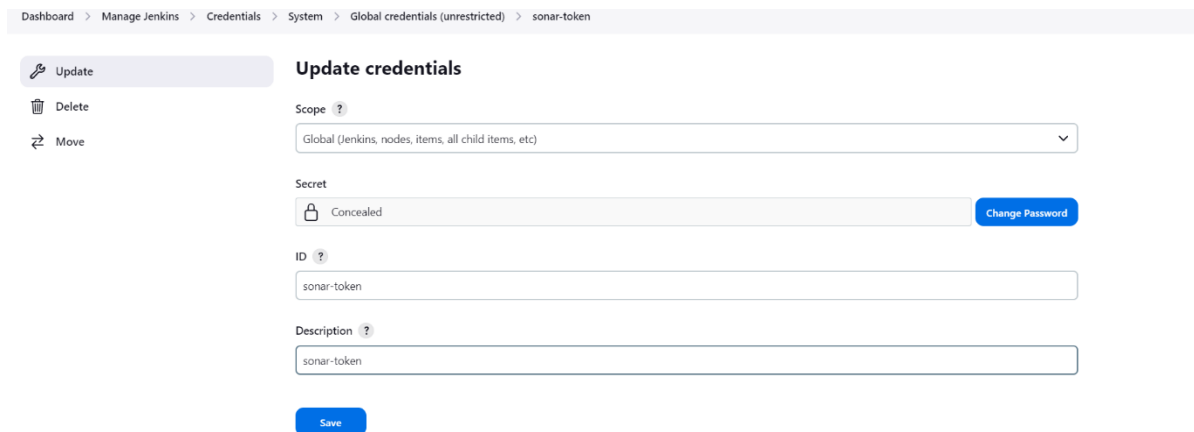


- Click on update token.
- Enter the token name as Jenkins and click on create.





- 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.

SonarQube Scanner installations Edited

Add SonarQube Scanner

SonarQube Scanner

Name
sonar-scanner

☒ Install automatically ?

Install from Maven Central

Version
SonarQube Scanner 6.1.0.4477

Add Installer ▼

Add SonarQube Scanner

Save Apply

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

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration Search for projects... A

Administration

Configuration ▼ Security ▼ Projects ▼ System Marketplace

General Settings
Encryption
Webhooks

Create User

Search by login or name...

	SCM Accounts	Last connection	Groups	Tokens
A Administrator admin		< 1 hour ago	sonar-administrators sonar-users	1 ⌵ ⚙️

1 of 1 shown

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration Search for projects... A

Administration

Configuration ▼ Security ▼ Projects ▼ System Marketplace

Webhooks

Webhooks are used to notify external services when a project analysis is finished. Learn more in the [Webhooks document](#).

Name	URL
jenkins	http://3.232.179.212:8080/sonarqube-webhook/

Create Webhook

All fields marked with * are required

Name *
jenkins ✓

URL *
http://3.110.188.253:8080/sonarqube-webhook/

Server endpoint that will receive the webhook payload, for example: "http://my_server/foo". If HTTP Basic authentication is used, HTTPS is recommended to avoid man in the middle attacks. Example: "https://myLogin.myPassword@my_server/foo"

Secret

If provided, secret will be used as the key to generate the HMAC hex (lowercase) digest value in the 'X-Sonar-Webhook-HMAC-SHA256' header.

Create Cancel

Embedded database should be used
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for it.

Get the most out of SonarQube!
Take advantage of the whole ecosystem by using SonarLint, a free IDE plugin that helps you find and fix issues earlier in your workflow. Connect SonarLint to SonarQube to sync rule acts and issue states.

Learn More

Dismiss

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

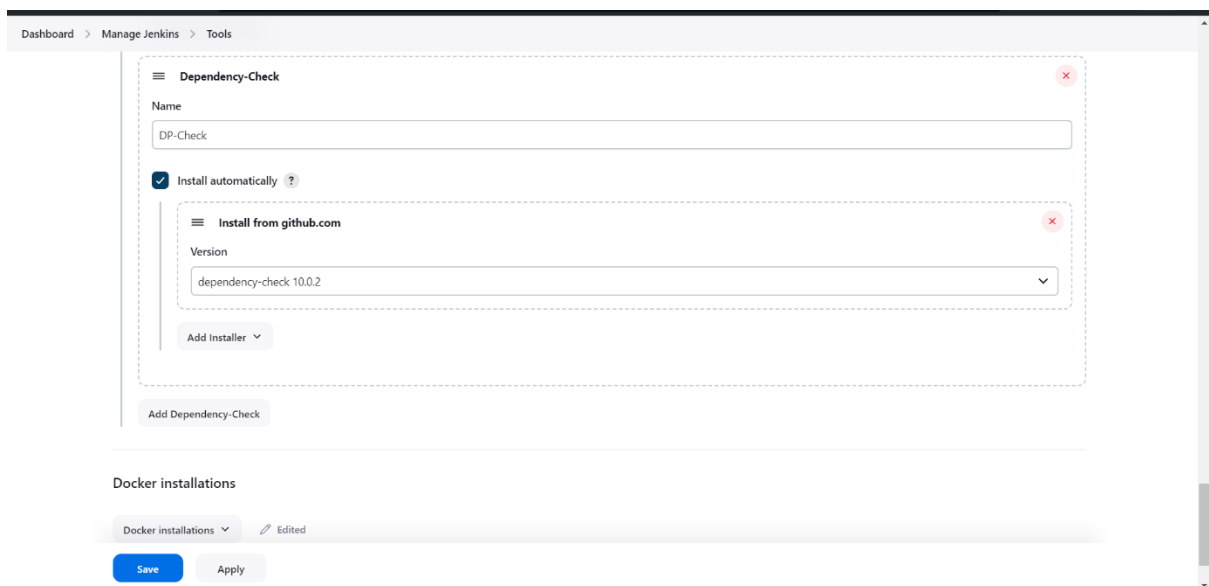
<http://jenkins-public-ip:8080>/sonarqube-webhook/

STEP-5(Install OWASP dependency):

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



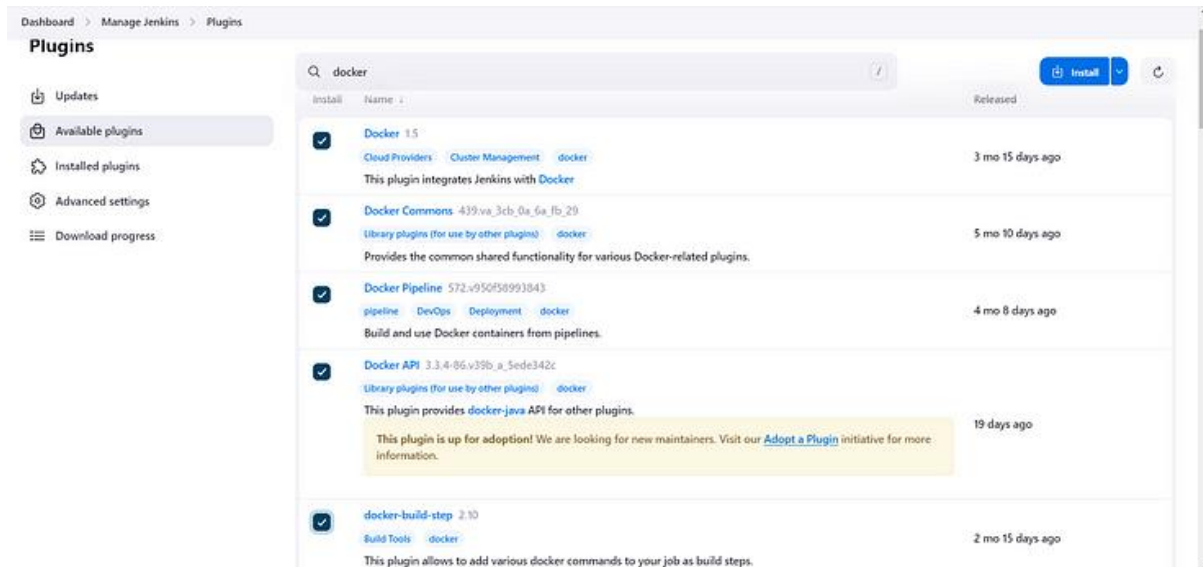
- Now configure the dependency tool in our global tools.



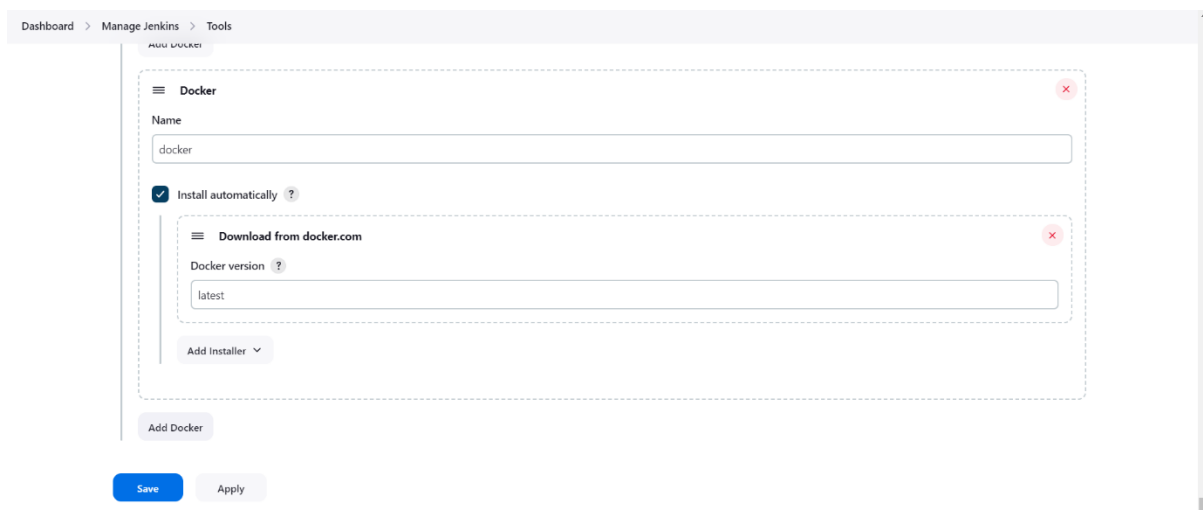
- Click on apply and save.

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.

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) > manichandu09/***** (docker)

Update credentials

Update

Delete

Move

Scope ?
Global (Jenkins, nodes, items, all child items, etc)

Username ?
manichandu09

☐ Treat username as secret ?

Password ?
Concealed [Change Password](#)

ID ?
docker

Description ?
docker

[Save](#)

- Now add this script to the pipeline.

```
pipeline{
  agent any
  tools{
    idk '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 ./ --disableVarnAudit --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.

The Jenkins Dashboard shows a list of builds for the 'Zomato' job. The most recent build, #11, is in a 'Success' state (green checkmark) and completed 23 minutes ago. The build history table is as follows:

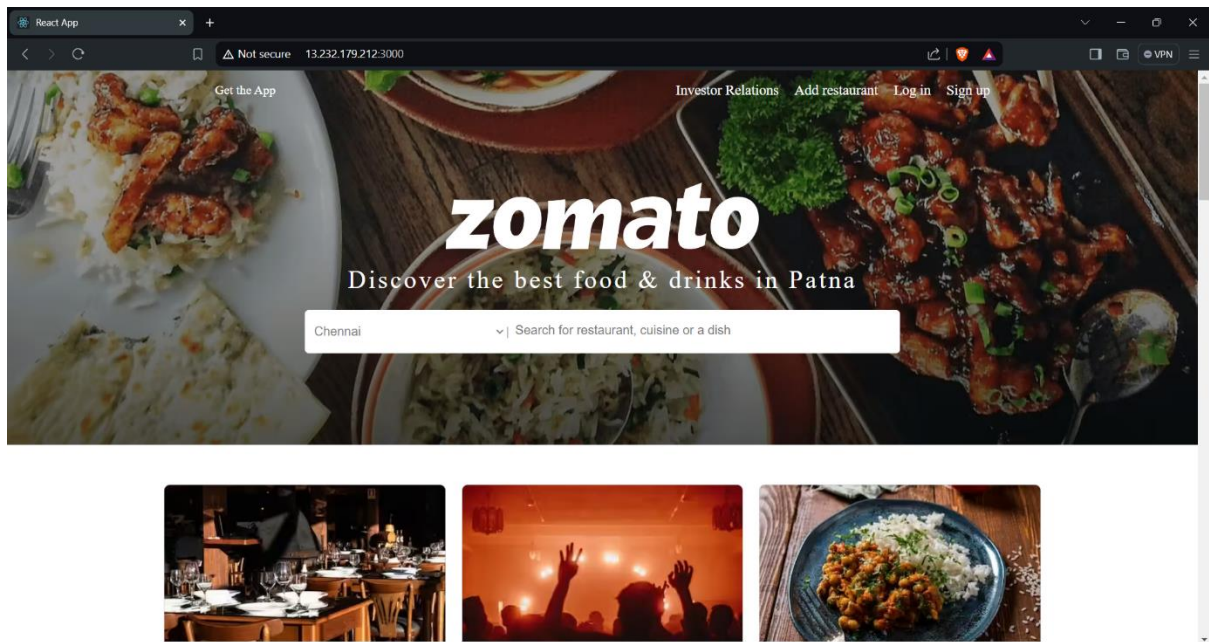
S	W	Name	Last Success	Last Failure	Last Duration
✓	☁	Zomato	23 min #11	48 min #10	18 min

On the left sidebar, the 'Build Queue' section indicates 'No builds in the queue.' and the 'Build Executor Status' section shows 'built-in node (0 of 2 executors busy)'.

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

The Docker Hub 'Repositories' page for the user 'manichandu09' shows a repository named 'zomato'. The repository was created 6 minutes ago and is currently 'Public'. The page includes a search bar, filters, and a 'Create repository' button.

- 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.