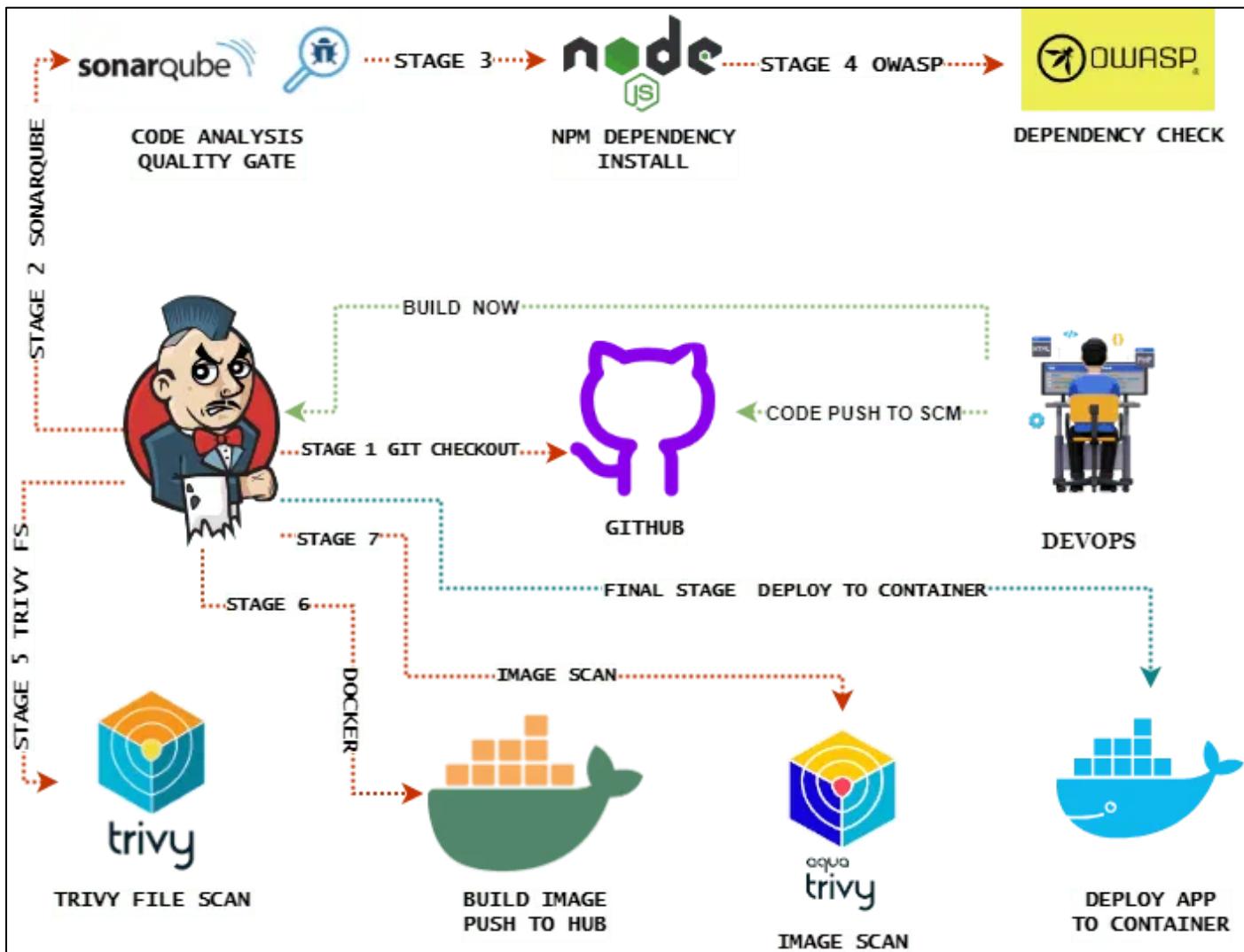


Project

# **Deploying Zomato Clone App with DevSecOps CI/CD**

[Ajay Gaur]

## # Zomato Clone Architecture –



### Steps:

- Step 1** — Launch an Ubuntu (22.04) T2 Large Instance
- Step 2** — Install Jenkins, Docker and Trivy. Create a Sonarqube Container using Docker.
- Step 3** — Install Plugins like JDK, Sonarqube Scanner, Nodejs, & OWASP Dependency Check.
- Step 4** — Create a Pipeline Project in Jenkins using a Declarative Pipeline
- Step 5** — Install OWASP Dependency Check Plugins
- Step 6** — Docker Image Build and Push
- Step 7** — Deploy the image using Docker
- Step 8** — Terminate the AWS EC2 Instances.

Now, let's get started and dig deeper into each of these steps.

## # Step 1 - Launch an Ubuntu (22.04) T2 Large Instance

Launch an AWS T2 Large Instance. Use the image as Ubuntu. You can create a new key pair or use an existing one. Enable HTTP and HTTPS settings in the Security Group and open all ports (not best case to open all ports but just for learning purposes it's okay).

The screenshot shows the AWS EC2 Instances page. On the left, there is a navigation sidebar with various options like EC2 Dashboard, Instances, Capacity Reservations, and more. The main area displays a table of instances. One instance is selected: 'zomato-app-clone' (Instance ID: i-044f7b2215c8b20ea). The instance is listed as 'Running' with the type 't2.large'. Below the table, a detailed view for the selected instance is shown. It includes sections for Instance summary, Details, Status and alarms, Monitoring, Security, Networking, Storage, and Tags. Under Instance summary, details such as Public IPv4 address (18.144.5.6), Instance state (Running), and Instance type (t2.large) are provided.

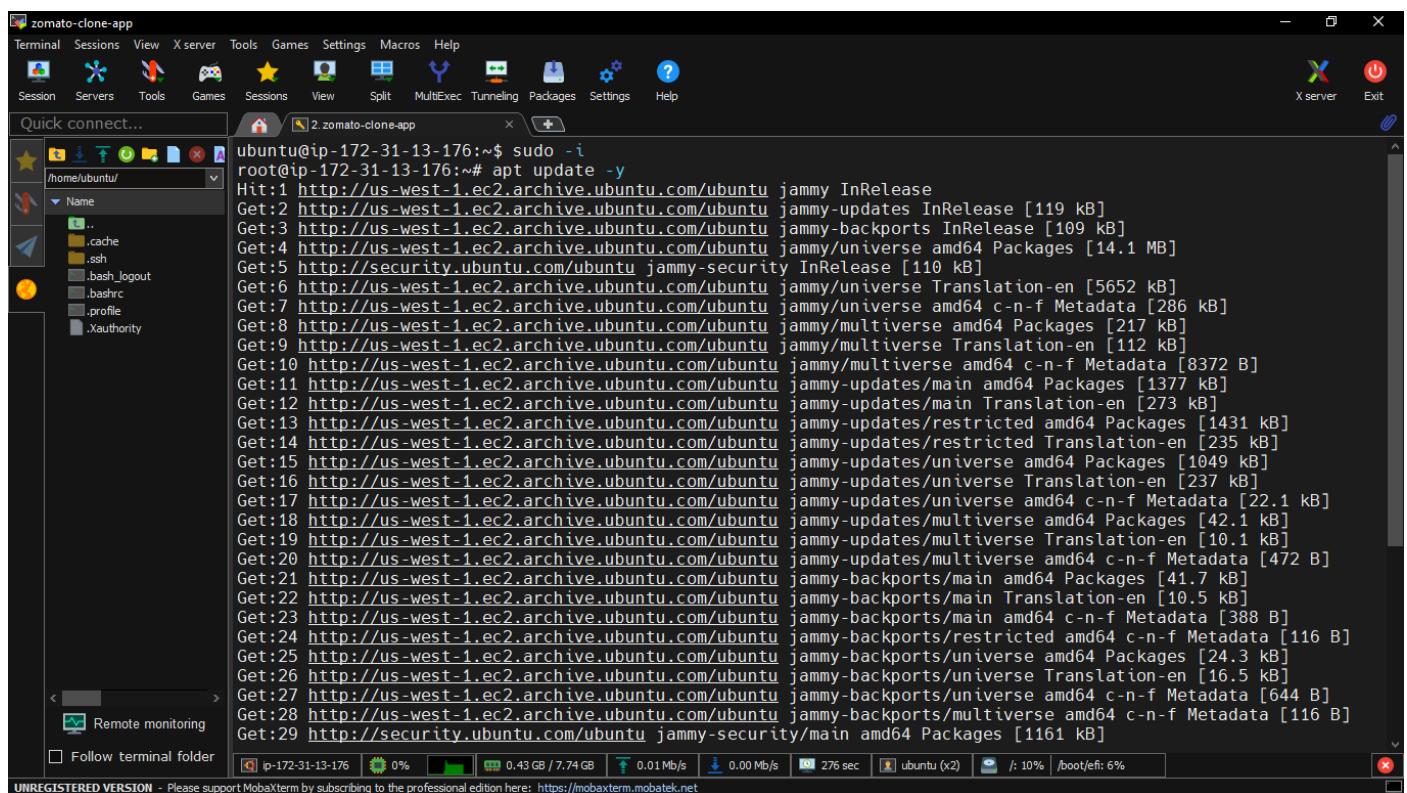
The screenshot shows the AWS EC2 Security Groups page. The security group 'sg-0983987dca7fb3518 - launch-wizard-5' is selected. The 'Details' tab is active, showing the security group name, owner, and VPC ID. The 'Inbound rules' tab is selected, displaying four rules. The rules are as follows:

Name	Security group rule...	IP version	Type	Protocol	Port range	Source
-	sgr-0b613c8a548b9b0f5	IPv4	SSH	TCP	22	0.0.0.0/0
-	sgr-0f2bd5f0cd0a89a71	IPv4	All traffic	All	All	0.0.0.0/0
-	sgr-05a875b828bcd905f	IPv4	HTTP	TCP	80	0.0.0.0/0
-	sgr-06fcf5dc276abb3f	IPv4	HTTPS	TCP	443	0.0.0.0/0

## Step 2 - Install Jenkins, Docker and Trivy

### → To Install Jenkins

```
> sudo apt update -y
> wget -O - https://packages.adoptium.net/artifactory/api/gpg/key/public | tee
  /etc/apt/keyrings/adoptium.asc
> echo "deb [signed-by=/etc/apt/keyrings/adoptium.asc]
  https://packages.adoptium.net/artifactory/deb $(awk -F=
  '/^VERSION_CODENAME/{print$2}' /etc/os-release) main" | tee
  /etc/apt/sources.list.d/adoptium.list
> sudo apt update -y
> sudo apt install temurin-17-jdk -y
> /usr/bin/java -version
> curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \
  /usr/share/keyrings/jenkins-keyring.asc > /dev/null
> 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 -y
> sudo apt-get install jenkins -y
> sudo systemctl start Jenkins
> sudo systemctl status Jenkins
```



```
ubuntu@ip-172-31-13-176:~$ sudo -i
root@ip-172-31-13-176:~# apt update -y
Hit:1 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:3 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Get:4 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]
Get:5 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:6 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy/universe Translation-en [5652 kB]
Get:7 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 c-n-f Metadata [286 kB]
Get:8 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 Packages [217 kB]
Get:9 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse Translation-en [112 kB]
Get:10 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 c-n-f Metadata [8372 B]
Get:11 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1377 kB]
Get:12 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [273 kB]
Get:13 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [1431 kB]
Get:14 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted Translation-en [235 kB]
Get:15 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1049 kB]
Get:16 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe Translation-en [237 kB]
Get:17 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 c-n-f Metadata [22.1 kB]
Get:18 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [42.1 kB]
Get:19 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse Translation-en [10.1 kB]
Get:20 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 c-n-f Metadata [472 B]
Get:21 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main amd64 Packages [41.7 kB]
Get:22 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main Translation-en [10.5 kB]
Get:23 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main amd64 c-n-f Metadata [388 B]
Get:24 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/restricted amd64 c-n-f Metadata [116 B]
Get:25 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 Packages [24.3 kB]
Get:26 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe Translation-en [16.5 kB]
Get:27 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 c-n-f Metadata [644 B]
Get:28 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/multiverse amd64 c-n-f Metadata [116 B]
Get:29 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [1161 kB]
```

```

root@ip-172-31-13-176:~# wget -O /etc/apt/keyrings/adoptium.asc https://packages.adoptium.net/artifactory/api/gpg/key/public
--2024-02-22 05:05:15-- https://packages.adoptium.net/artifactory/api/gpg/key/public
Resolving packages.adoptium.net (packages.adoptium.net)... 151.101.43.42
Connecting to packages.adoptium.net (packages.adoptium.net)|151.101.43.42|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1793 (1.8K) [text/plain]
Saving to: 'STDOUT'

[1793/1793] 1.75K --.-KB/s in 0s

2024-02-22 05:05:15 (51.5 MB/s) - written to stdout [1793/1793]

-----BEGIN PGP PUBLIC KEY BLOCK-----

mQENBGGTvtQBACAC6ey144n7CG8foaffF6mwIBN1fIm1ILZDuGS4tMr0/XI8pgJnT
QvsPxZWEvtSm7be0z0zJcXwBcJl1B0ui8k5kHMTI75gCmZPsokLFWIEpuRBQ
PBoCuSw80apDmlLnNDQLVQDFtEua5gaNa/fRw9YsmBoXbqvgrjFUIdGyWoQvH5+a
90YLWD9n5VV0gnVMB+aclwVzB/zJw3kHGSguMtlAheQiah78yomQn/UIX8yqDf
+11sP3+c87YcjKRqImRTtmKEdCtGPAIXC6SYA+uEEkbY0Fy0chkvtnWJ597fa
Epa14rnICU8z0J6X5z3v1aM2WerhX9oq9X8PABEBAAQ0QEfkB3B0aXVtIEDoRyBL
ZXkgKERFQi9SUE0gU2LnbmluZyBLZXkpIDx0ZW11cmuluWRldkBLy2xpcHNLLm9y
Zz6JAVIEewEIAdwWIQQ7BNdTyQUNm00PzmEPeilZfjwSwUCYZ09NAIBAwULCQgH
AgMiAgEGF0oJAsCBBYC AwECHgcCF4AACgkQhDxIpWX48Et4AggAjjzYwukV3nG
7ngInng18G/m9JoHr7BmwgcQXYhdy5hVkmCu5JLeXz2LMBUH/F2nD595hgjMabk
kvib20X8lq9RsNbdfc2hBCWU6qyhKxsIq14bo12/XDyEzzMyyZWWNGo/27C17Xmj
pwu31nh0pDdpqdyWDIKojbVnnxLCRY8as8Sm+1ufi709Kci4MuwhNsULCSwb/fju
NKeHkrHbLcHKUIIEcmTSKRWrpMYBzm1HYOGBz4xPuElwUfUp71ehfoyBZlp6RDRf
15TYI1FmCyHuvjNhrJgWv7b0Tcf8yObGY+TEUhzc4xQqCrF4ur9d3opvsuPB0sv+
Klqi5KSZgrkBd0Rhk700A0qAq14okly8cfRpYVenEQP iB75AUZfKRpMduiR6IxAj
SKCh7aSoFZ9AubUEBVpZsyT5svxoEPe1i4TdbF+m0FGy42Ec01La3ArLTj5H8FRL

```

```

root@ip-172-31-13-176:~# echo "deb [signed-by=/etc/apt/keyrings/adoptium.asc] https://packages.adoptium.net/artifactory/deb ${_VERSION_CODENAME} main" | tee /etc/apt/sources.list.d/adoptium.list
root@ip-172-31-13-176:~# sudo apt update -y
Hit:1 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:4 https://packages.adoptium.net/artifactory/deb jammy InRelease [6635 B]
Hit:5 http://security.ubuntu.com/ubuntu jammy-security InRelease
Get:6 https://packages.adoptium.net/artifactory/deb jammy/main amd64 Packages [6965 B]
Fetched 13.6 kB in 1s (25.0 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
74 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ip-172-31-13-176:~#

```

```
zomato-clone-app
Terminal Sessions View Xserver Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
X server Exit
Quick connect...
2.zomato-clone-app
root@ip-172-31-13-176:~# sudo apt install temurin-17-jdk -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  adoptum-ca-certificates alsal-topology-conf alsal-ucm-conf fontconfig-config fonts-dejavu-core
  fonts-dejavu-extra java-common libasound2 libasound2-data libfontconfig1 libxi6 libxrender1 libxtst6 p11-kit
  p11-kit-modules x11-common
Suggested packages:
  default-jre libasound2-plugins alsal-utils
The following NEW packages will be installed:
  adoptum-ca-certificates alsal-topology-conf alsal-ucm-conf fontconfig-config fonts-dejavu-core
  fonts-dejavu-extra java-common libasound2 libasound2-data libfontconfig1 libxi6 libxrender1 libxtst6 p11-kit
  p11-kit-modules temurin-17-jdk x11-common
0 upgraded, 17 newly installed, 0 to remove and 74 not upgraded.
Need to get 170 MB of archives.
After this operation, 346 MB of additional disk space will be used.
Get:1 https://packages.adoptium.net/artifactory/deb jammy/main amd64 adoptum-ca-certificates all 1.0.2-1 [2304 kB]
Get:2 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 p11-kit-modules amd64 0.24.0-6build1 [223 kB]
Get:3 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 p11-kit amd64 0.24.0-6build1 [101 kB]
Get:4 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 alsal-topology-conf all 1.2.5.1-2 [15.5 kB]
Get:5 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 libasound2-data all 1.2.6.1-1ubuntu1 [19.1 kB]
Get:6 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 libasound2 amd64 1.2.6.1-1ubuntu1 [390 kB]
Get:7 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 alsal-ucm-conf all 1.2.6.3-1ubuntu1.10 [43.4 kB]
Get:8 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 fonts-dejavu-core all 2.37-2build1 [1041 kB]
Get:9 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 fontconfig-config all 2.13.1-4.2ubuntu5 [2 kB]
Fetched 170 MB in 1s (59.5 kB/s)
Reading package lists... Done
root@ip-172-31-13-176:~#
```

```
zomato-clone-app
Terminal Sessions View Xserver Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
X server Exit
Quick connect...
2.zomato-clone-app
Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-13-176:~# /usr/bin/java --version
openjdk 17.0.10 2024-01-16
OpenJDK Runtime Environment Temurin-17.0.10+7 (build 17.0.10+7)
OpenJDK 64-Bit Server VM Temurin-17.0.10+7 (build 17.0.10+7, mixed mode, sharing)
root@ip-172-31-13-176:~# curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \
  /usr/share/keyrings/jenkins-keyring.asc > /dev/null
root@ip-172-31-13-176:~# 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
root@ip-172-31-13-176:~# sudo apt-get update -y
Hit:1 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Ign:4 https://pkg.jenkins.io/debian-stable binary/ InRelease
Get:5 https://pkg.jenkins.io/debian-stable binary/ Release [2044 B]
Get:6 https://pkg.jenkins.io/debian-stable binary/ Release.gpg [833 B]
Get:7 https://packages.adoptium.net/artifactory/deb jammy InRelease [6635 B]
Hit:8 http://security.ubuntu.com/ubuntu jammy-security InRelease
Get:9 https://pkg.jenkins.io/debian-stable binary/ Packages [26.4 kB]
Fetched 35.9 kB in 1s (59.5 kB/s)
Reading package lists... Done
root@ip-172-31-13-176:~#
```

```

zomato-clone-app
Terminal Sessions View Xserver Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
Quick connect...
z.zomato-clone-app
/home/ubuntu/
Name
.. .cache .ssh .bash_logout .bashrc .profile .Xauthority
root@ip-172-31-13-176:~# sudo apt-get install jenkins -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  net-tools
The following NEW packages will be installed:
  jenkins net-tools
0 upgraded, 2 newly installed, 0 to remove and 74 not upgraded.
Need to get 86.1 MB of archives.
After this operation, 87.4 MB of additional disk space will be used.
Get:1 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 net-tools amd64 1.60+git20181103.0eebece-1
ubuntu5 [204 kB]
Get:2 https://pkg.jenkins.io/debian-stable binary/ jenkins 2.440.1 [85.8 MB]
Fetched 86.1 MB in 3s (33.3 MB/s)
Selecting previously unselected package net-tools.
(Reading database ... 66125 files and directories currently installed.)
Preparing to unpack .../net-tools_1.60+git20181103.0eebece-1ubuntu5_amd64.deb ...
Unpacking net-tools (1.60+git20181103.0eebece-1ubuntu5) ...
Selecting previously unselected package jenkins.
Preparing to unpack .../jenkins_2.440.1_all.deb ...
Unpacking jenkins (2.440.1) ...
Setting up net-tools (1.60+git20181103.0eebece-1ubuntu5) ...
Setting up jenkins (2.440.1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /lib/systemd/system/jenkins.service.
Processing triggers for man-db (2.10.2-1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

```

ip-172-31-13-176 1% 2.52 GB / 7.74 GB 0.01 Mb/s 0.00 Mb/s 9 min ubuntu (x2) /: 13% /boot/efi: 6%

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

```

zomato-clone-app
Terminal Sessions View Xserver Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
Quick connect...
z.zomato-clone-app
/home/ubuntu/
Name
.. .cache .ssh .bash_logout .bashrc .profile .Xauthority
No containers need to be restarted.
No user sessions are running outdated binaries.
No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-13-176:~# sudo systemctl start jenkins
root@ip-172-31-13-176:~# sudo systemctl enable jenkins
Synchronizing state of jenkins.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable jenkins
root@ip-172-31-13-176:~# sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
    Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enabled)
    Active: active (running) since Thu 2024-02-22 05:09:16 UTC; 1min 3s ago
      Main PID: 5441 (java)
        Tasks: 51 (limit: 9498)
       Memory: 2.1G
          CPU: 41.033s
        CGroup: /system.slice/jenkins.service
                └─ 5441 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cach

Feb 22 05:09:00 ip-172-31-13-176 jenkins[5441]: a7fe574d38e74376b128603f6958438f
Feb 22 05:09:00 ip-172-31-13-176 jenkins[5441]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPw
Feb 22 05:09:00 ip-172-31-13-176 jenkins[5441]: ****
Feb 22 05:09:00 ip-172-31-13-176 jenkins[5441]: ****
Feb 22 05:09:00 ip-172-31-13-176 jenkins[5441]: ****
Feb 22 05:09:16 ip-172-31-13-176 jenkins[5441]: 2024-02-22 05:09:16.137+0000 [id=32] INFO jenkins>
Feb 22 05:09:16 ip-172-31-13-176 jenkins[5441]: 2024-02-22 05:09:16.156+0000 [id=24] INFO hudson.>
Feb 22 05:09:16 ip-172-31-13-176 systemd[1]: Started Jenkins Continuous Integration Server.
Feb 22 05:09:16 ip-172-31-13-176 jenkins[5441]: 2024-02-22 05:09:16.494+0000 [id=49] INFO h.m.Down>
Feb 22 05:09:16 ip-172-31-13-176 jenkins[5441]: 2024-02-22 05:09:16.495+0000 [id=49] INFO hudson.>
Lines 1-20/20 (END)

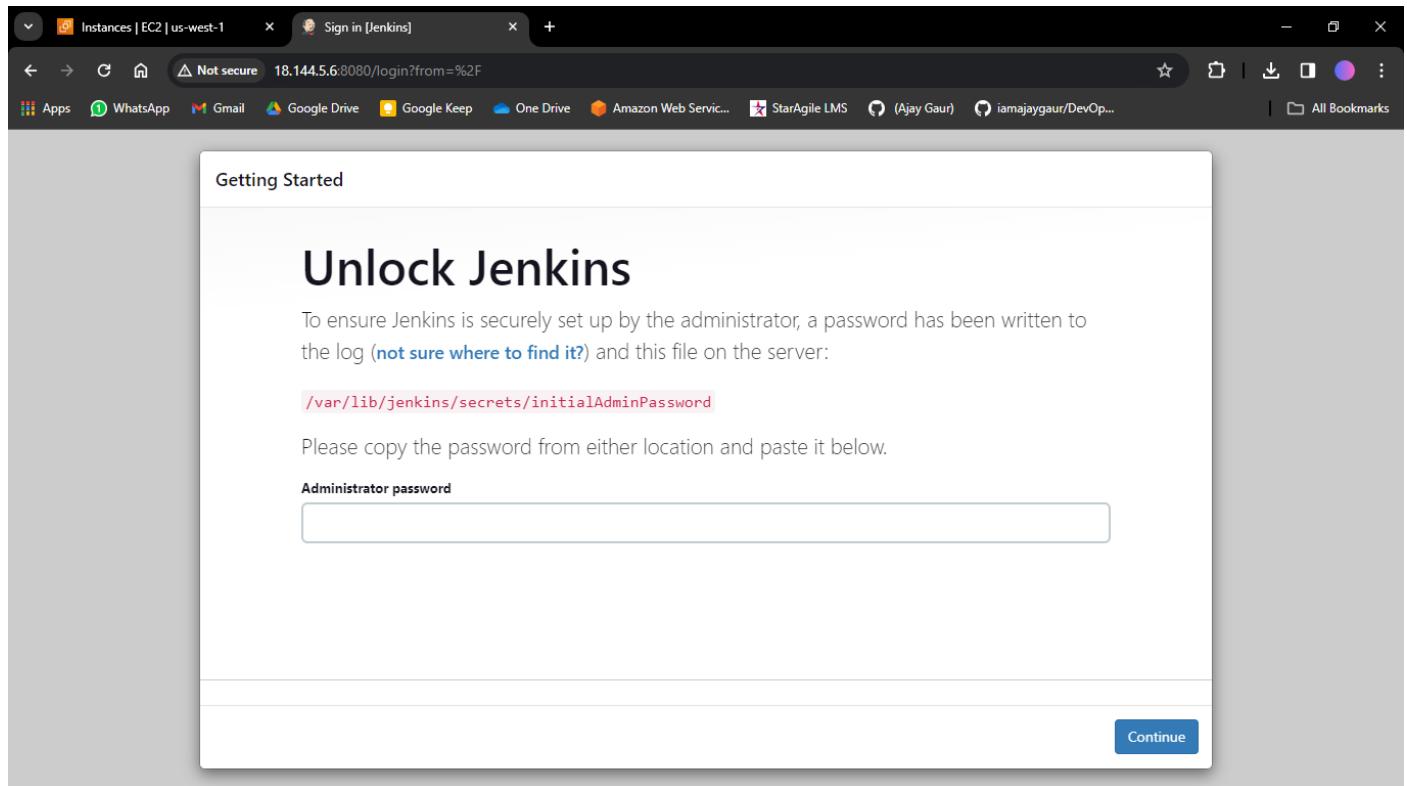
```

ip-172-31-13-176 0% 2.53 GB / 7.74 GB 0.01 Mb/s 0.00 Mb/s 10 min ubuntu (x2) root /: 13% /boot/efi: 6%

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

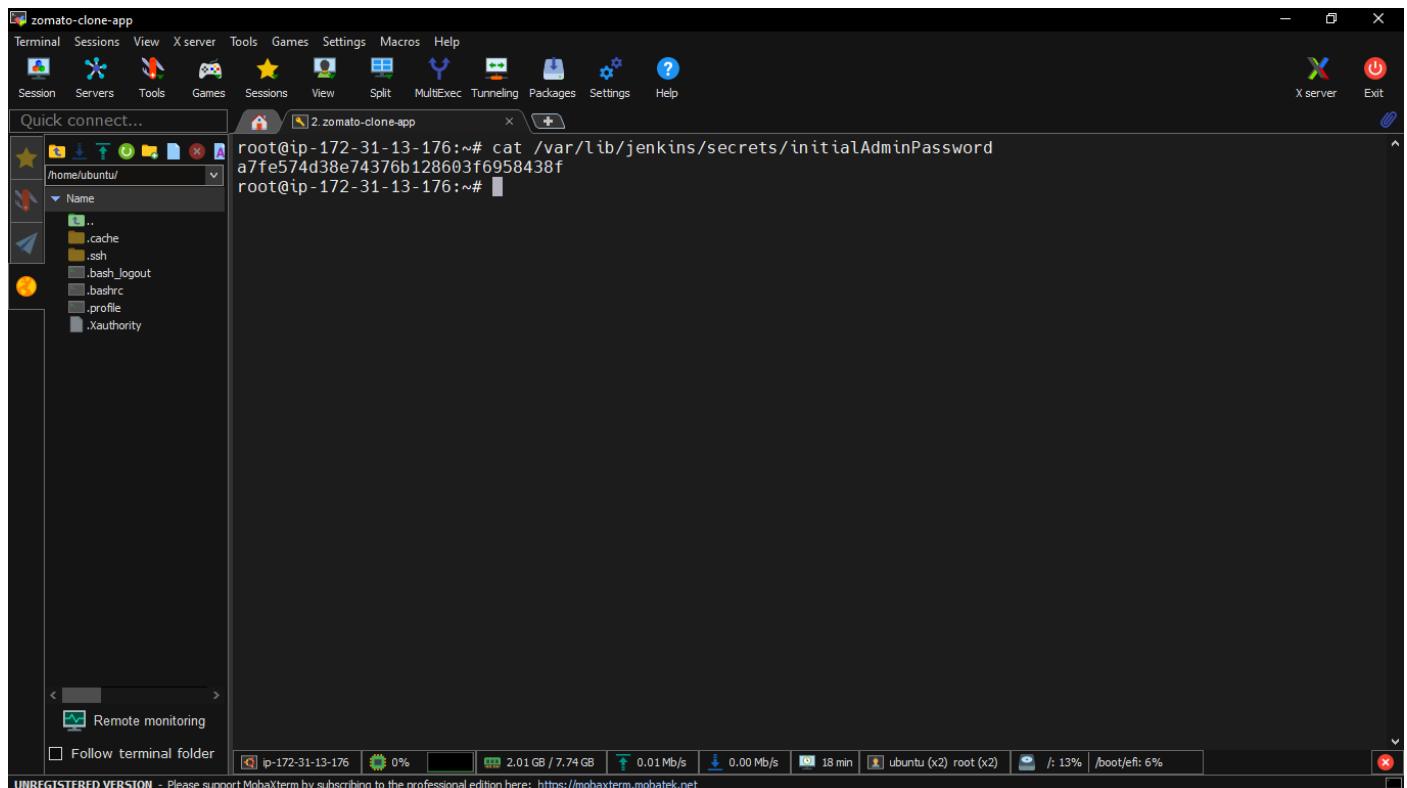
Once Jenkins is installed, you will need to go to your AWS EC2 Security Group and open Inbound Port 8080, since Jenkins works on Port 8080.

→ [http://<public\\_ip>:8080/](http://<public_ip>:8080/)



```
> cat /var/lib/jenkins/secrets/initialAdminPassword
```

→ Get the Password and put into Jenkins for login.



The screenshot shows a web browser window titled "Sign in [Jenkins]" with the URL "18.144.5.6:8080/login?from=%2F". The page is titled "Getting Started" and features a large heading "Unlock Jenkins". A text block explains that a password has been written to the log and provides the path "/var/lib/jenkins/secrets/initialAdminPassword". Below this, a placeholder text "Administrator password" is followed by a redacted input field. At the bottom right is a blue "Continue" button.

→ Install the suggested plugins.

The screenshot shows a web browser window titled "Setup Wizard [Jenkins]" with the URL "18.144.5.6:8080". The page is titled "Getting Started" and features a large heading "Customize Jenkins". A text block states that plugins extend Jenkins with additional features. Two options are presented: "Install suggested plugins" (blue box) and "Select plugins to install" (grey box). Both boxes contain descriptive text. At the bottom left is a note "Jenkins 2.44.0" and at the bottom center is the URL "18.144.5.6:8080/#".

→ Create a user click on save and continue.

The screenshot shows the Jenkins Setup Wizard 'Getting Started' page. It contains fields for entering a Username ('ajay'), Password ('...'), Confirm password ('...'), Full name ('Ajay Gaur'), and E-mail address ('ajay94932@gmail.com'). At the bottom, there are two buttons: 'Skip and continue as admin' and a highlighted 'Save and Continue' button.

The screenshot shows the Jenkins Setup Wizard 'Instance Configuration' page. It features a large heading 'Instance Configuration'. A 'Jenkins URL:' field is populated with 'http://18.144.5.6:8080/'. Below the field, explanatory text states: 'The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the BUILD\_URL environment variable provided to build steps.' Another note says: 'The proposed default value shown is not saved yet and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.' At the bottom, there are three buttons: 'Not now', 'Save and Finish' (highlighted), and 'Save and Continue'.

The screenshot shows a web browser window titled "Setup Wizard [Jenkins]" with the URL "18.144.5.6:8080". The main content area is titled "Getting Started" and "Instance Configuration". A form field labeled "Jenkins URL:" contains the value "http://18.144.5.6:8080/". Below the field is a descriptive text block: "The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the BUILD\_URL environment variable provided to build steps." It also notes that the proposed default value is not saved yet and is generated from the current request if possible. At the bottom of the page, it says "Jenkins 2.440.1" and provides "Not now" and "Save and Finish" buttons.

The screenshot shows a web browser window titled "Setup Wizard [Jenkins]" with the URL "18.144.5.6:8080". The main content area is titled "Getting Started" and displays the message "Jenkins is ready!". It states "Your Jenkins setup is complete." and features a blue "Start using Jenkins" button. At the bottom of the page, it says "Jenkins 2.440.1".

## → Jenkins Getting Started Screen.

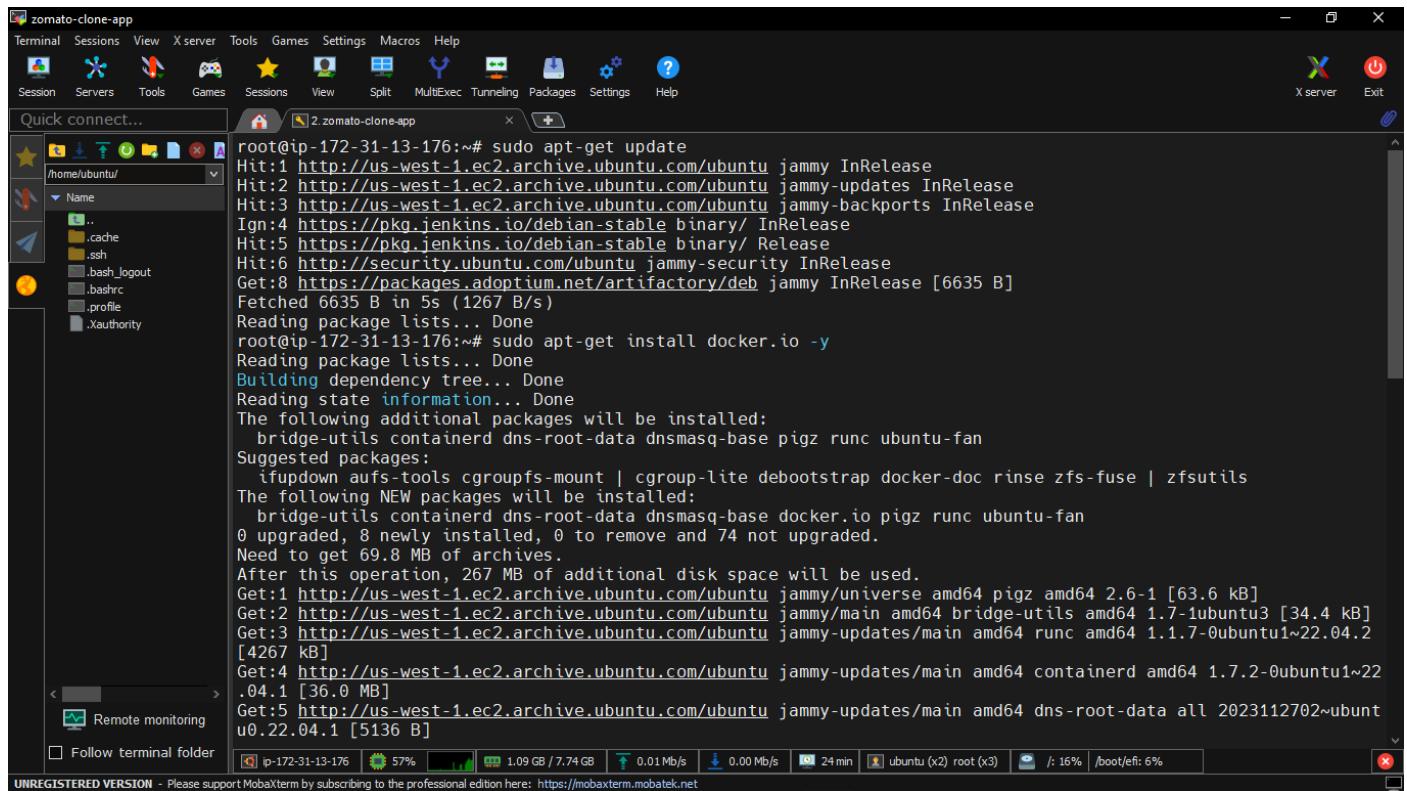
The screenshot shows the Jenkins dashboard. At the top, there's a navigation bar with tabs for 'Instances | EC2 | us-west-1' and 'Dashboard [Jenkins]'. The address bar shows 'Not secure 18.144.5.6:8080'. Below the bar are various links to Google services like Apps, WhatsApp, Gmail, etc. On the right, there's a search bar, a notification icon with '1', and a user profile for 'Ajay Gaur'.

The main content area has a title 'Welcome to Jenkins!' and a sub-section 'Start building your software project'. It includes a 'Create a job' button and three links: 'Set up a distributed build', 'Set up an agent', and 'Configure a cloud'. There are also sections for 'Build Queue' (empty) and 'Build Executor Status' (1 Idle, 2 Idle).

At the bottom right, there are links for 'REST API' and 'Jenkins 2.440.1'.

## → To Install Docker

```
> sudo apt-get update  
> sudo apt-get install docker.io -y  
> sudo usermod -aG docker $USER #my case is ubuntu  
> newgrp docker  
> sudo chmod 777 /var/run/docker.sock
```



The screenshot shows a MobaXterm window titled "zomato-clone-app". The terminal session is running as root on an Ubuntu system. The command sequence for installing Docker is shown, along with the resulting package list and dependency resolution. The terminal window includes a file browser sidebar and various status indicators at the bottom.

```
root@ip-172-31-13-176:~# sudo apt-get update  
Hit:1 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease  
Hit:2 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease  
Hit:3 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease  
Ign:4 https://pkg.jenkins.io/debian-stable binary/ InRelease  
Hit:5 https://pkg.jenkins.io/debian-stable binary/ Release  
Hit:6 http://security.ubuntu.com/ubuntu jammy-security InRelease  
Get:8 https://packages.adoptopenjdk.net/artifactory/deb jammy InRelease [6635 B]  
Fetched 6635 B in 5s (1267 B/s)  
Reading package lists... Done  
root@ip-172-31-13-176:~# sudo apt-get install docker.io -y  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following additional packages will be installed:  
  bridge-utils containerd dns-root-data dnsmasq-base pigz runc ubuntu-fan  
Suggested packages:  
  ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-doc rinse zfs-fuse | zfsutils  
The following NEW packages will be installed:  
  bridge-utils containerd dns-root-data dnsmasq-base docker.io pigz runc ubuntu-fan  
0 upgraded, 8 newly installed, 0 to remove and 74 not upgraded.  
Need to get 69.8 MB of archives.  
After this operation, 267 MB of additional disk space will be used.  
Get:1 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 pigz amd64 2.6-1 [63.6 kB]  
Get:2 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 bridge-utils amd64 1.7-1ubuntu3 [34.4 kB]  
Get:3 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 runc amd64 1.1.7-0ubuntu1~22.04.2 [4267 kB]  
Get:4 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 containerd amd64 1.7.2-0ubuntu1~22.04.1 [36.0 MB]  
Get:5 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 dns-root-data all 2023112702~ubuntu-0.22.04.1 [5136 B]
```

```

Setting up bridge-utils (1.7-1ubuntu3) ...
Setting up pigz (2.6-1) ...
Setting up containerd (1.7.2-0ubuntu1~22.04.1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /lib/systemd/system/containerd.service.
Setting up ubuntu-fan (0.12.16) ...
Created symlink /etc/systemd/system/multi-user.target.wants/ubuntu-fan.service → /lib/systemd/system/ubuntu-fan.service.
Setting up docker.io (24.0.5-0ubuntu1~22.04.1) ...
Adding group `docker` (GID 123) ...
Done.
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
Processing triggers for dbus (1.12.20-2ubuntu4.1) ...
Processing triggers for man-db (2.10.2-1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-13-176:~# sudo usermod -aG docker $USER
root@ip-172-31-13-176:~# newgrp docker
root@ip-172-31-13-176:~# sudo chmod 777 /var/run/docker.sock
root@ip-172-31-13-176:~#

```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

After the docker installation, we create a sonarqube container (Remember to add 9000 ports in the security group).

```

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

```

```

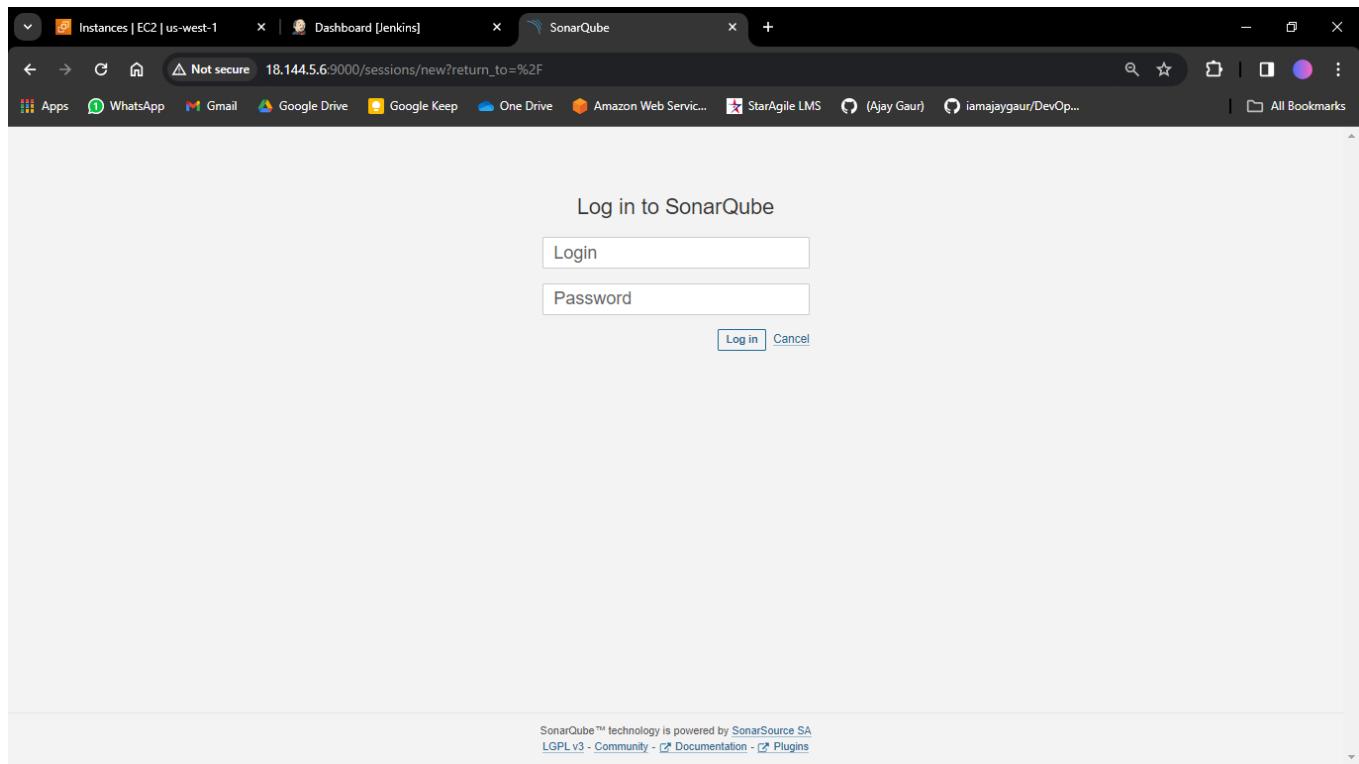
root@ip-172-31-13-176:~# docker run -d --name sonar -p 9000:9000 sonarqube:lts-community
Unable to find image 'sonarqube:lts-community' locally
lts-community: Pulling from library/sonarqube
d66d6a6a3687: Pull complete
18f947fdc0fc: Pull complete
5374706a264d: Pull complete
c9aeff6b625d: Pull complete
82faf7b7220d: Pull complete
86c5dcfeabbd: Pull complete
3d8c05302d61: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:f5f3d1d45484581d381b63f7daef56a6887a24cb4b068b8da561c84703833013c
Status: Downloaded newer image for sonarqube:lts-community
4844e9088132255a9691a7db9e42e58472f7d1b47b2ac3655036e3babd371881
root@ip-172-31-13-176:~# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
NAMES
4844e9088132      sonarqube:lts-community   "/opt/sonarqube/dock..."   2 minutes ago       Up 2 minutes      0.0.0.0:9000->9
000/tcp, ::9000->9000/tcp   sonar
root@ip-172-31-13-176:~#

```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

→ Now our sonarqube is up and running

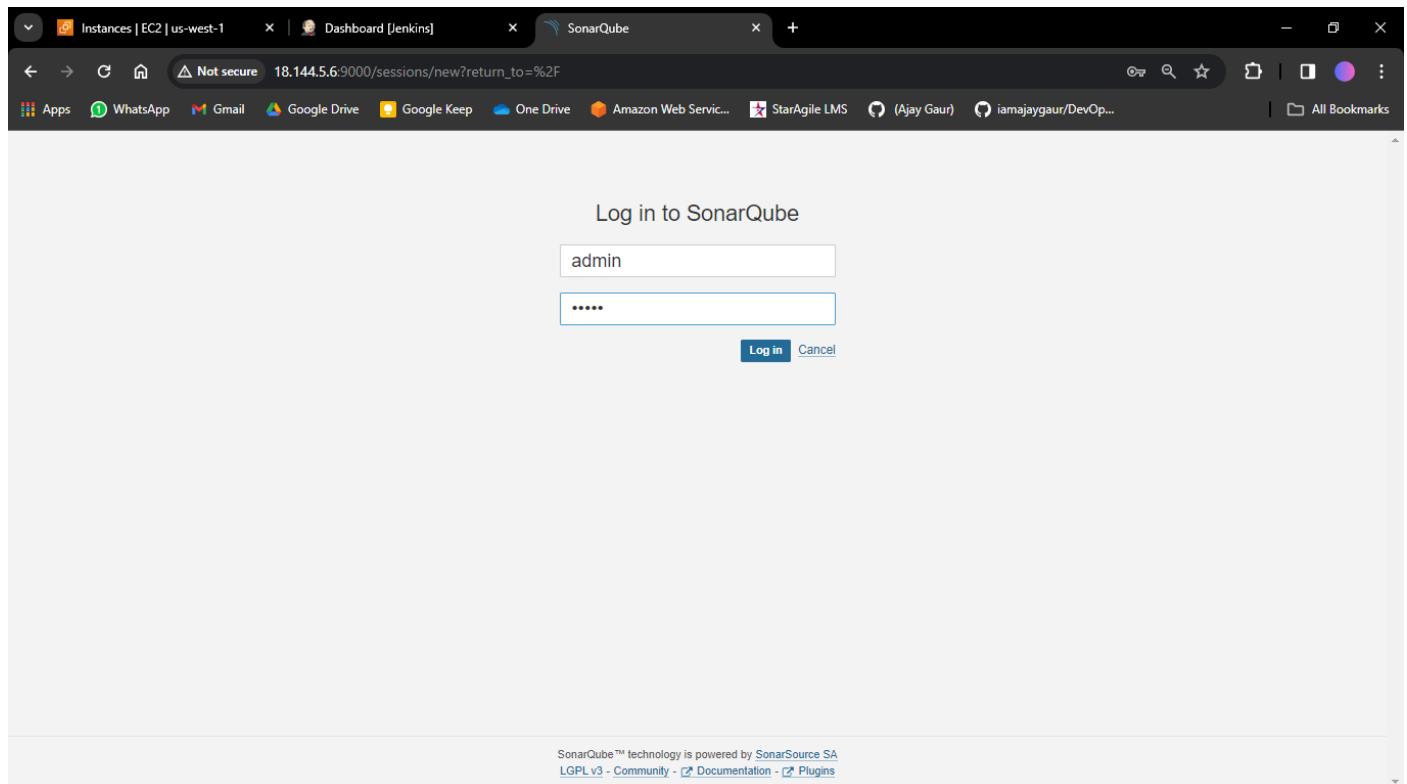
→ [http://<public\\_ip>/sessions/new?return\\_to=%2F](http://<public_ip>/sessions/new?return_to=%2F)



→ Enter username and password, click on login and change password

→ username - admin

→ password – admin



## → Update New password

The screenshot shows a web browser window with the URL [18.144.5.6:9000/account/reset\\_password](http://18.144.5.6:9000/account/reset_password). The page title is "Update your password". A note at the top says, "This account should not use the default password." Below it, instructions say, "Enter a new password" and "All fields marked with \* are required". There are three input fields: "Old Password \*", "New Password \*", and "Confirm Password \*". Each field has a placeholder with three dots. A blue "Update" button is at the bottom.

## → This is Sonar Dashboard.

The screenshot shows a web browser window with the URL [18.144.5.6:9000/projects/create](http://18.144.5.6:9000/projects/create). The page title is "How do you want to create your project?". It asks if the user wants to benefit from SonarQube's features by creating a project from a DevOps platform. It lists five options: "From Azure DevOps", "From Bitbucket Server", "From Bitbucket Cloud", "From GitHub", and "From GitLab", each with a "Set up global configuration" link. Below this, it says, "Are you just testing or have an advanced use-case? Create a project manually." with a "Manually" link and a left-right navigation icon.

## → To Install Trivy

- > 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

The screenshot shows a MobaXterm window titled 'zomato-clone-app'. It displays a terminal session with the following command and output:

```
root@ip-172-31-13-176:~# sudo apt-get install wget apt-transport-https gnupg lsb-release -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
lsb-release is already the newest version (11.1.0ubuntu4).
lsb-release set to manually installed.
wget is already the newest version (1.21.2-2ubuntu1).
wget set to manually installed.
gnupg is already the newest version (2.2.27-3ubuntu2.1).
gnupg set to manually installed.
The following NEW packages will be installed:
  apt-transport-https
0 upgraded, 1 newly installed, 0 to remove and 74 not upgraded.
Need to get 1510 B of additional disk space will be used.
After this operation, 170 kB of additional disk space will be used.
Get:1 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 apt-transport-https all 2.4.11 [1510 B]
Fetched 1510 B in 0s (84.1 kB/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 66552 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_2.4.11_all.deb ...
Unpacking apt-transport-https (2.4.11) ...
Setting up apt-transport-https (2.4.11) ...
Scanning processes...
Scanning linux images...
Running kernel seems to be up-to-date.
No services need to be restarted.
No containers need to be restarted.
```

The terminal window includes a status bar at the bottom showing network activity and battery level.

The screenshot shows a MobaXterm window titled 'zomato-clone-app'. It displays a terminal session with the following commands and output:

```
root@ip-172-31-13-176:~# wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | gpg --dearmor | sudo tee /usr/share/keyrings/trivy.gpg > /dev/null
root@ip-172-31-13-176:~# 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
deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasecurity.github.io/trivy-repo/deb jammy main
Hit:1 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Ign:2 https://pkg.jenkins.io/debian-stable binary/ InRelease
Hit:3 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:4 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:5 https://pkg.jenkins.io/debian-stable binary/ Release
Get:6 https://aquasecurity.github.io/trivy-repo/deb jammy InRelease [3061 B]
Get:7 https://packages.adoptium.net/artifactory/deb jammy InRelease [6635 B]
Get:8 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:10 https://aquasecurity.github.io/trivy-repo/deb jammy/main amd64 Packages [367 B]
Fetched 120 kB in 1s (146 kB/s)
Reading package lists... Done
root@ip-172-31-13-176:~# 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
deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasecurity.github.io/trivy-repo/deb jammy main
root@ip-172-31-13-176:~# sudo apt-get update
Hit:1 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://us-west-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Ign:4 https://pkg.jenkins.io/debian-stable binary/ InRelease
Hit:5 https://aquasecurity.github.io/trivy-repo/deb jammy InRelease
Hit:6 https://pkg.jenkins.io/debian-stable binary/ Release
Get:7 https://packages.adoptium.net/artifactory/deb jammy InRelease [6635 B]
Hit:8 http://security.ubuntu.com/ubuntu jammy-security InRelease
Fetched 6635 B in 1s (10.3 kB/s)
Reading package lists... Done
```

The terminal window includes a status bar at the bottom showing network activity and battery level.

```
root@ip-172-31-13-176:~# sudo apt-get install trivy -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  trivy
0 upgraded, 1 newly installed, 0 to remove and 74 not upgraded.
Need to get 54.6 MB of archives.
After this operation, 209 MB of additional disk space will be used.
Get:1 https://aquasecurity.github.io/trivy-repo/deb jammy/main amd64 trivy amd64 0.49.1 [54.6 MB]
Fetched 54.6 MB in 1s (72.1 MB/s)
Selecting previously unselected package trivy.
(Reading database ... 66556 files and directories currently installed.)
Preparing to unpack .../trivy_0.49.1_amd64.deb ...
Unpacking trivy (0.49.1) ...
Setting up trivy (0.49.1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-13-176:~# trivy --version
Version: 0.49.1
root@ip-172-31-13-176:~#
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

→ Next, we will log in to Jenkins and start to configure our Pipeline in Jenkins.

## # Step 3 - Install Plugins like JDK, Sonarqube Scanner, NodeJs, OWASP Dependency Check

→ Goto Manage Jenkins → Plugins → Available Plugins

Install below plugins

1. Eclipse Temurin Installer (Install without restart)
2. SonarQube Scanner (Install without restart)
3. NodeJs Plugin (Install Without restart)

The screenshot shows the Jenkins plugin manager interface. The left sidebar has tabs for 'Updates', 'Available plugins' (which is selected), 'Installed plugins', 'Advanced settings', and 'Download progress'. The main area has a search bar with 'Eclipse Temurin Installer' and an 'Install' button. A table lists the plugin: 'Eclipse Temurin installer 1.5' by 'Provides an installer for the JDK tool that downloads the JDK from https://adoptium.net'. It shows 'Released 1 yr 4 mo ago'. At the bottom right of the page are links for 'REST API' and 'Jenkins 2.440.1'.

The screenshot shows the Jenkins plugin manager interface. The left sidebar has tabs for 'Updates', 'Available plugins' (selected), 'Installed plugins', 'Advanced settings', and 'Download progress'. The main area has a search bar with 'NodeJS Plugin' and an 'Install' button. A table lists the plugin: 'NodeJS 1.6.1' by 'npm'. It shows 'Released 6 mo 9 days ago'. At the bottom right of the page are links for 'REST API' and 'Jenkins 2.440.1'.

The screenshot shows the Jenkins Plugins page. On the left, there's a sidebar with links: Updates, Available plugins, **Installed plugins** (which is selected and highlighted in grey), Advanced settings, and Download progress. The main area has a search bar at the top with the word "Sona". Below it, a table lists the "SonarQube Scanner for Jenkins" plugin, version 2.17.2. The table includes columns for Name, Description, Enabled status (with a blue toggle switch), and a trash icon. A note below the table states: "This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality. Report an issue with this plugin". At the bottom right of the page, there are links for REST API and Jenkins 2.440.1.

## → To Configure Java and Nodejs in Global Tool Configuration

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

The screenshot shows the Jenkins Tools page. Under the "JDK installations" section, there's a form to add a new JDK. It has fields for "Name" (set to "jdk17") and "Install automatically" (checkbox checked). Below this, there's a sub-section for "Install from adoptium.net" with a dropdown for "Version" set to "jdk-17.0.8.1+1". There's also a "Add Installer" button. At the bottom of the form are "Save" and "Apply" buttons.

The screenshot shows the Jenkins 'Tools' configuration page. A new tool named 'node16' is being created. The 'Install automatically' checkbox is checked. The 'Version' dropdown is set to 'NodeJS 16.2.0'. There is a note about forcing the 32-bit architecture if available. Under 'Global npm packages to install', there is a placeholder for specifying a list of packages. Under 'Global npm packages refresh hours', the value '72' is entered. At the bottom are 'Save' and 'Apply' buttons.

## → To Create a Job

→ create a job as Zomato Name, select pipeline and click on ok.

The screenshot shows the Jenkins 'New Item' creation page. A new item named 'zomato-app-clone' is being created. The 'Freestyle project' type is selected. Other options shown are 'Pipeline' and 'Multi-configuration project'. The 'OK' button is visible at the bottom left of the dialog.

## # Step 4 - Configure Sonar Server in Manage Jenkins

→ Grab the Public IP Address of your EC2 Instance, Sonarqube works on Port 9000, so <Public IP>:9000. Goto your Sonarqube Server.

Click on Administration → Security → Users → Click on Tokens and Update Token → Give it a name → and click on Generate Token

The screenshot shows the Sonarqube Administration interface. The top navigation bar includes tabs for sonarqube, Projects, Issues, Rules, Quality Profiles, Quality Gates, and Administration. The Administration tab is selected. Below the navigation, there are sub-tabs: Configuration, Security, Projects, System, and Marketplace. The main content area is titled 'Administration' and contains a sidebar with various configuration sections like General, Authentication, DevOps Platform Integrations, External Analyzers, General, Housekeeping, JaCoCo, Languages, and New Code. The 'Users' section is currently selected, as indicated by a dropdown menu. The main content area displays settings for 'Cross project duplication detection', which is described as deprecated and detects duplicates at the project level. A toggle switch is shown, with '(default)' next to it. Below this, there are sections for 'Email' and 'SMTP host'. At the bottom of the page, a URL '18.144.5.6:9000/admin/users' is visible.

→ click on update Token

The screenshot shows the Sonarqube Administration interface, specifically the 'Users' section. The top navigation bar and sub-tabs are identical to the previous screenshot. The main content area shows a table of users. One user, 'Administrator admin', is listed with columns for SCM Accounts, Last connection, Groups, and Tokens. The 'Tokens' column for this user shows 'sonar-administrators' and 'sonar-users'. To the right of the tokens, there is a 'Tokens' icon with a gear and a '0' count, and a prominent 'Update Tokens' button. At the bottom of the page, a warning message in a yellow box states: 'Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.' The footer of the page provides SonarQube version information: 'SonarQube™ technology is powered by SonarSource SA Community Edition - Version 9.9.4 (build 87374) - [LGPL v3](#) - [Community](#) - [Documentation](#) - [Plugins](#) - [Web API](#)'.

→ Create a token with a name and generate and copy token.

The screenshot shows the SonarQube Administration interface. In the center, a modal window titled "Tokens of Administrator" is open. It displays a "Generate Tokens" section where a new token named "jenkins" has been created. A success message says, "New token 'jenkins' has been created. Make sure you copy it now, you won't be able to see it again!" Below this, a "Copy" button is visible next to the token ID "squ\_ae6d123c0afce8737ddbecc34f2ec2bb861c1a3". A table lists the token details:

Name	Type	Project	Last use	Created	Expiration	Action
jenkins	User		Never	February 22, 2024	March 23, 2024	Revoke

At the bottom of the modal, there is a note: "Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine." A "Done" button is at the bottom right.

→ Goto Jenkins Dashboard → Manage Jenkins → Credentials → System → Global credentials (unrestricted) Add the Following things

**Kind** – Select Secret text

**Secret** – post the generated token

**ID** – sonar

**Description** – sonar

The screenshot shows the Jenkins "New credentials" configuration page. The "Kind" dropdown is set to "Secret text". The "Scope" dropdown is set to "Global (Jenkins, nodes, items, all child items, etc)". The "Secret" field contains a masked password. The "ID" field is set to "sonar". The "Description" field is set to "sonar". At the bottom left, a "Create" button is visible.

→ You will get this page once you click on create

The screenshot shows the Jenkins Global credentials (unrestricted) page. At the top, there is a search bar and a 'Add Credentials' button. Below the header, a table lists a single credential:

ID	Name	Kind	Description
sonar	sonar	Secret text	sonar

Below the table, there is a 'Icon:' section with options S, M, and L. At the bottom right, there are links for 'REST API' and 'Jenkins 2.440.1'.

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

→ Click Apply and Save

The screenshot shows the Jenkins System configuration page under the SonarQube installations section. It displays a form to add a new SonarQube installation:

**Name:** sonar-server

**Server URL:** http://18.144.5.6:9000 (Default is http://localhost:9000)

**Server authentication token:** sonar

At the bottom, there are 'Save' and 'Apply' buttons.

→ The Configure System option is used in Jenkins to configure different server. Global Tool Configuration is used to configure different tools that we install using Plugins.

→ We will install a sonar scanner in the tools.

→ Go to Dashboard → Manage Jenkins → Tools

→ Click on Apply and Save

The screenshot shows the Jenkins 'Tools' configuration page at <http://18.144.5.6:8080/manage/configureTools/>. The 'SonarQube Scanner installations' section is open, displaying a form to add a new scanner. The 'Name' field contains 'sonar-scanner'. The 'Install automatically' checkbox is checked. Under 'Install from Maven Central', the 'Version' dropdown is set to 'SonarQube Scanner 5.0.1.3006'. Below the form are 'Save' and 'Apply' buttons.

→ In the Sonarqube Dashboard add a quality gate also

→ Go to Administration → Configuration → Webhooks

The screenshot shows the SonarQube Administration page at <http://18.144.5.6:9000/admin/users>. The 'Administration' tab is selected. The 'Webhooks' option under 'General Settings' is highlighted. The main table lists a user named 'Administrator admin'. A warning message at the bottom states: 'Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.' The footer includes links to SonarQube™ technology powered by SonarSource SA, Community Edition - Version 9.9.4 (build 87374) - LGPL v3 - Community - Documentation - Plugins - Web API.

→ Click on Create

The screenshot shows the SonarQube administration interface for webhooks. The top navigation bar includes links for Instances | EC2 | us-west-1, Manage Jenkins [Jenkins], and Webhooks - Administration. The main menu has tabs for sonarqube, Projects, Issues, Rules, Quality Profiles, Quality Gates, and Administration, with Administration selected. Below the menu is a search bar and a 'Create' button. The 'Webhooks' section displays a message: 'No webhook defined.' A warning message in a yellow box states: 'Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.' At the bottom, footer information includes: SonarQube™ technology is powered by SonarSource SA, Community Edition - Version 9.9.4 (build 87374) - [GPL v3](#) - [Community](#) - [Documentation](#) - [Plugins](#) - [Web API](#).

→ Add details

#in url section of quality gate

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

→ Click on Create

The screenshot shows the 'Create Webhook' dialog box overlaid on the SonarQube administration page. The dialog has fields for Name (jenkins), URL (http://18.144.5.6:8080/sonarqube-webhook/), and Secret. A note below the URL field explains: '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://my.Login.myPassword@my\_server/foo"'.

**Create Webhook**

All fields marked with \* are required

Name \*

 ✓

URL \*

 ✓

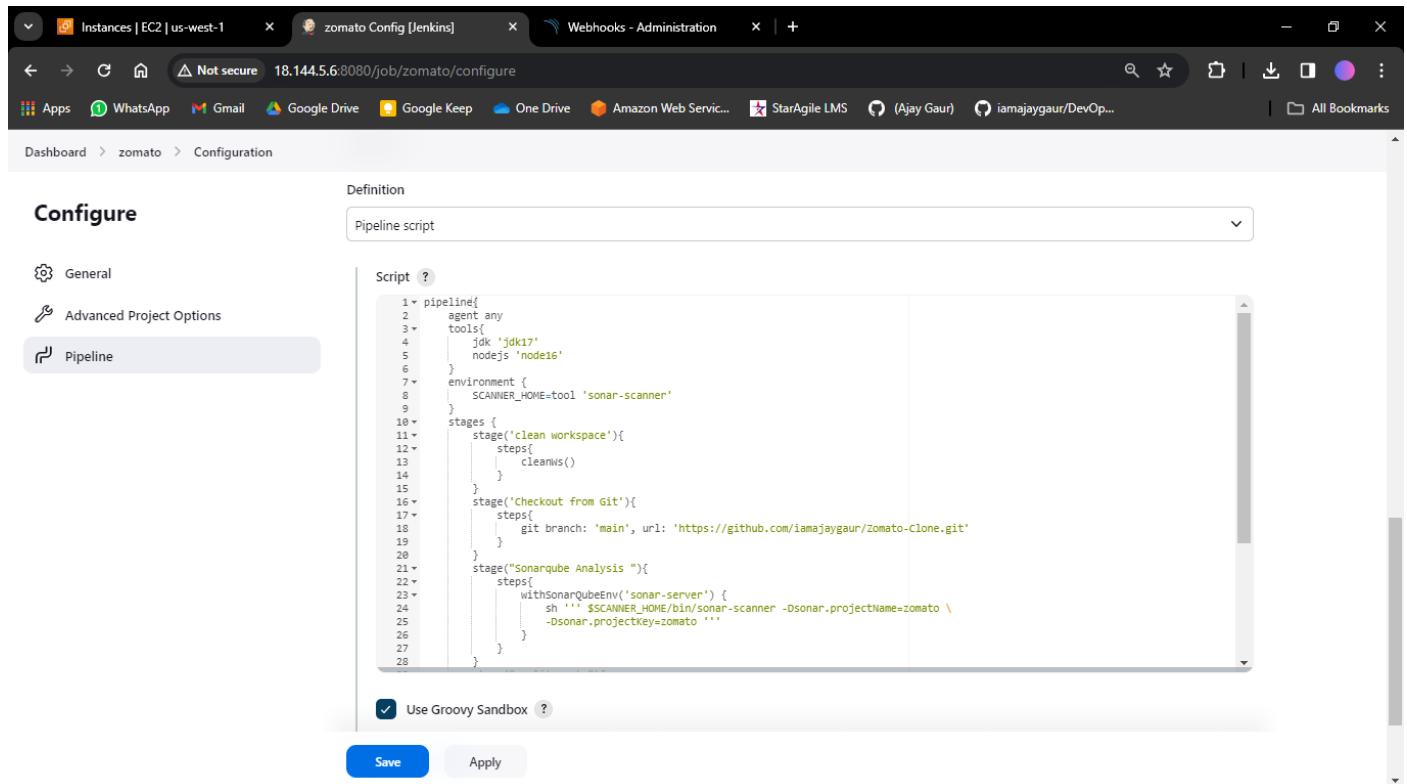
Secret

 ✓

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**

- Let's go to our Pipeline and add the script in our Pipeline Script.
- Add pipeline script from file name jenkinsfile from github repository
- <https://github.com/iamajaygaur/Zomato-Clone.git>
- Go to Dashboard → Zomato → Configuration and Paste the Script
- Then Click on Apply and Save



The screenshot shows the Jenkins Pipeline configuration page for a project named 'zomato'. The 'Pipeline' tab is selected. In the 'Script' editor, the Jenkinsfile is pasted:

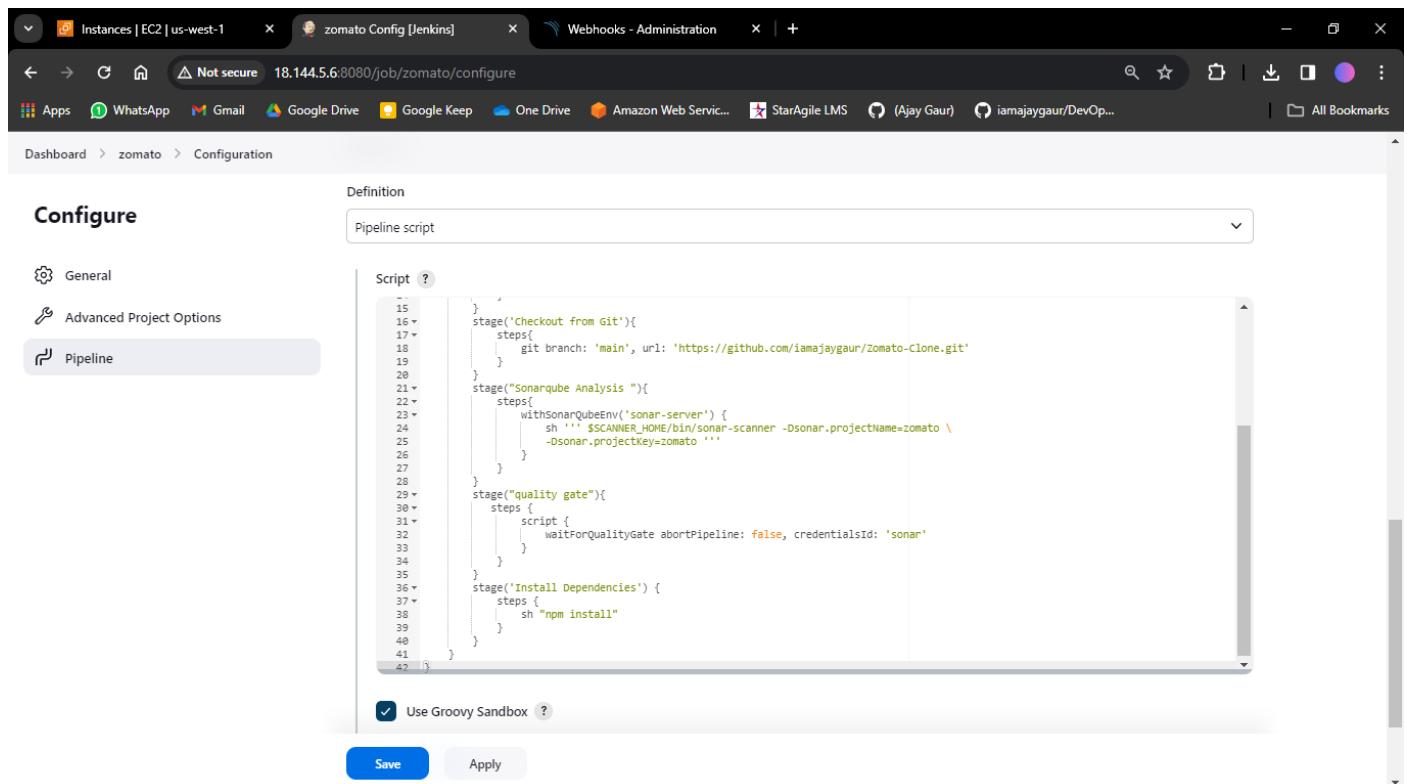
```

1+ pipeline{
2+   agent any
3+   tools{
4+     jdk 'jdk17'
5+     nodejs 'node16'
6+   }
7+   environment {
8+     SCANNER_HOME=tool 'sonar-scanner'
9+   }
10+  stages {
11+    stage('clean workspace'){
12+      steps{
13+        cleanWs()
14+      }
15+    }
16+    stage('Checkout from Git'){
17+      steps{
18+        git branch: 'main', url: 'https://github.com/iamajaygaur/Zomato-Clone.git'
19+      }
20+    }
21+    stage("Sonarqube Analysis"){
22+      steps{
23+        withSonarqubeEnv('sonar-server') {
24+          sh """$SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=zomato \
25+            -Dsonar.projectKey=zomato"""
26+        }
27+      }
28+    }
29+  }
30+}
31+
32+stage('Checkout from Git'){
33+  steps{
34+    git branch: 'main', url: 'https://github.com/iamajaygaur/Zomato-Clone.git'
35+  }
36+}
37+stage("Sonarqube Analysis"){
38+  steps{
39+    withSonarqubeEnv('sonar-server') {
40+      sh """$SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=zomato \
41+            -Dsonar.projectKey=zomato"""
42+    }
43+  }
44+}
45+
46+stage("quality_gate"){
47+  steps{
48+    script {
49+      waitForQualityGate abortPipeline: false, credentialsId: 'sonar'
50+    }
51+  }
52+}
53+
54+stage('Install Dependencies') {
55+  steps {
56+    sh "npm install"
57+  }
58+}

```

Use Groovy Sandbox

**Save** **Apply**



The screenshot shows the Jenkins Pipeline configuration page for the same project 'zomato'. The 'Pipeline' tab is selected. The Jenkinsfile now includes several new stages:

```

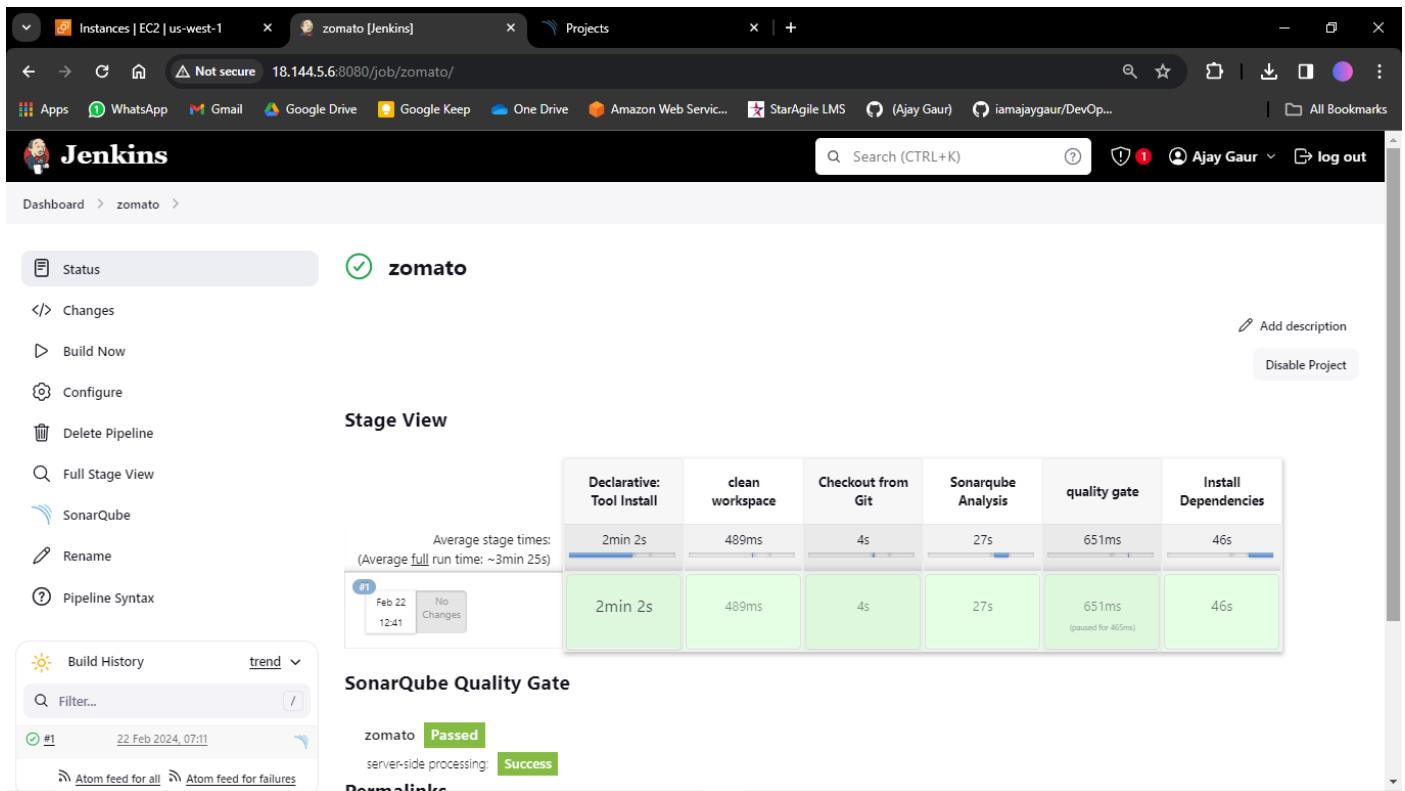
1+ pipeline{
2+   agent any
3+   tools{
4+     jdk 'jdk17'
5+     nodejs 'node16'
6+   }
7+   environment {
8+     SCANNER_HOME=tool 'sonar-scanner'
9+   }
10+  stages {
11+    stage('clean workspace'){
12+      steps{
13+        cleanWs()
14+      }
15+    }
16+    stage('Checkout from Git'){
17+      steps{
18+        git branch: 'main', url: 'https://github.com/iamajaygaur/Zomato-Clone.git'
19+      }
20+    }
21+    stage("Sonarqube Analysis"){
22+      steps{
23+        withSonarqubeEnv('sonar-server') {
24+          sh """$SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=zomato \
25+            -Dsonar.projectKey=zomato"""
26+        }
27+      }
28+    }
29+    stage("quality_gate"){
30+      steps{
31+        script {
32+          waitForQualityGate abortPipeline: false, credentialsId: 'sonar'
33+        }
34+      }
35+    }
36+    stage('Install Dependencies') {
37+      steps {
38+        sh "npm install"
39+      }
40+    }
41+  }
42+}
43+
44+stage('Checkout from Git'){
45+  steps{
46+    git branch: 'main', url: 'https://github.com/iamajaygaur/Zomato-Clone.git'
47+  }
48+}
49+stage("Sonarqube Analysis"){
50+  steps{
51+    withSonarqubeEnv('sonar-server') {
52+      sh """$SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=zomato \
53+            -Dsonar.projectKey=zomato"""
54+    }
55+  }
56+}
57+
58+stage("quality_gate"){
59+  steps{
60+    script {
61+      waitForQualityGate abortPipeline: false, credentialsId: 'sonar'
62+    }
63+  }
64+}
65+
66+stage('Install Dependencies') {
67+  steps {
68+    sh "npm install"
69+  }
70+}

```

Use Groovy Sandbox

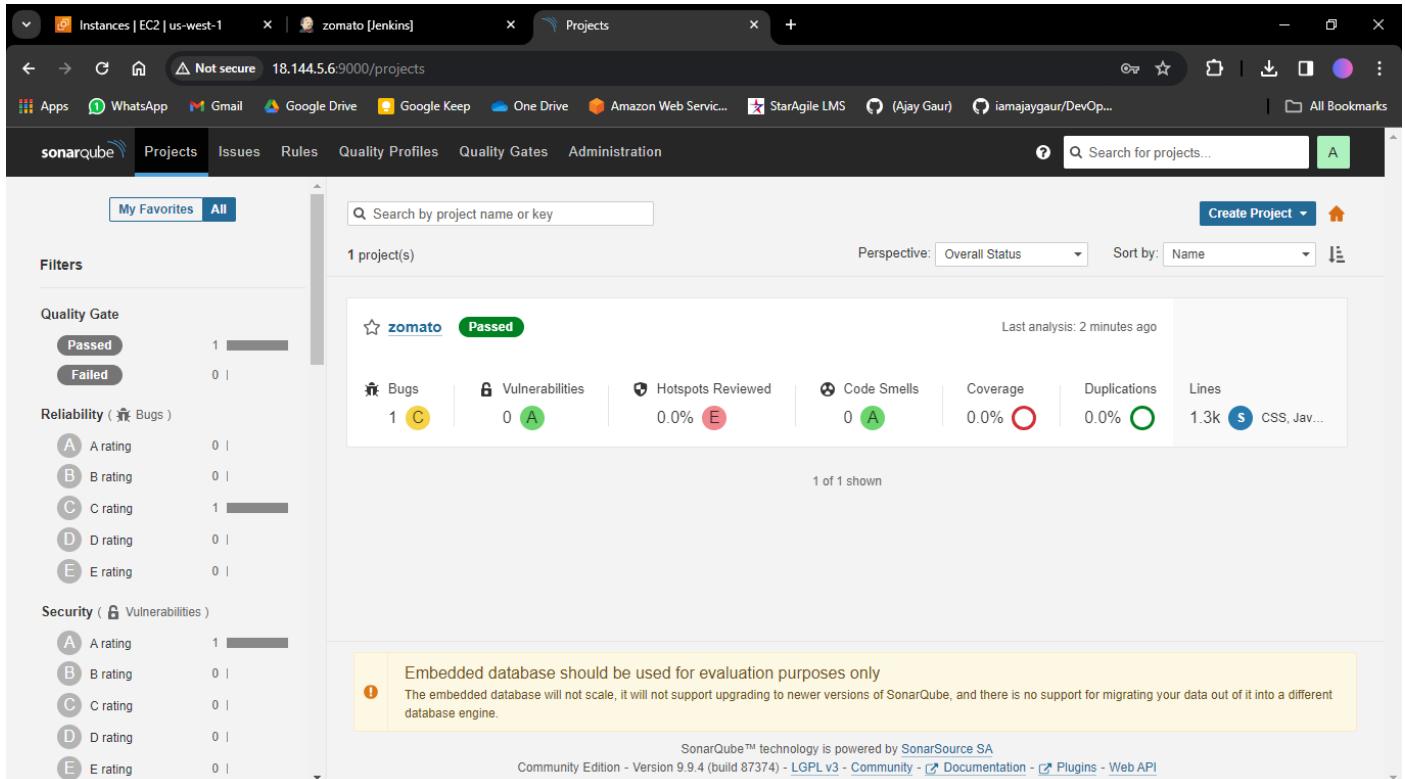
**Save** **Apply**

→ Click on Build now, you will see the stage view like this



The screenshot shows the Jenkins interface for the 'zomato' pipeline. On the left, there's a sidebar with various options: Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, SonarQube, Rename, Pipeline Syntax, Build History, Filter..., and Atom feeds. The main area is titled 'Stage View' and displays a timeline of stages: Declarative: Tool Install (2min 2s), clean workspace (489ms), Checkout from Git (4s), Sonarqube Analysis (27s), quality gate (651ms), and Install Dependencies (46s). Below this, a 'SonarQube Quality Gate' section shows the status as 'Passed' with a green button labeled 'Success'. A note says 'server-side processing: Success'. At the bottom, there's a 'Downstream' link.

→ To see the report, you can go to Sonarqube Server and go to Projects.



The screenshot shows the SonarQube interface for the 'zomato' project. The top navigation bar includes links for sonarqube, Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, and a search bar. The main content area shows a summary for the 'zomato' project, which has passed its quality gate. It displays metrics: Bugs (1 C), Vulnerabilities (0 A), Hotspots Reviewed (0.0% E), Code Smells (0 A), Coverage (0.0% O), Duplications (0.0% G), and Lines (1.3k S CSS, Java). On the left, there are filters for Quality Gate (Passed, Failed), Reliability (A-E rating), and Security (A-E rating). A warning message at the bottom states: 'Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.' The footer includes the text 'SonarQube™ technology is powered by SonarSource SA Community Edition - Version 9.9.4 (build 87374) - LGPL v3 - Community - Documentation - Plugins - Web API'.

You can see the report has been generated and the status shows as passed. You can see that there are 1.3k lines. To see a detailed report, you can go to issues.

## # Step 5 - Install OWASP Dependency Check Plugins

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

The screenshot shows the Jenkins Plugin Manager interface. On the left, there's a sidebar with options like 'Updates', 'Available plugins' (which is selected), 'Installed plugins', 'Advanced settings', and 'Download progress'. The main area has a search bar at the top with the query 'OWASP Dependency-Check'. Below the search bar, there's a table with columns 'Install', 'Name ↴', and 'Released'. A single row is shown for 'OWASP Dependency-Check 5.4.6', which is categorized under 'Security', 'DevOps', 'Build Tools', and 'Build Reports'. The description states: 'This plug-in can independently execute a Dependency-Check analysis and visualize results. Dependency-Check is a utility that identifies project dependencies and checks if there are any known, publicly disclosed, vulnerabilities.' To the right of the table, it says '14 days ago'. At the bottom right of the main area, there are links for 'REST API' and 'Jenkins 2.440.1'. At the very bottom of the page, there are 'Save' and 'Apply' buttons.

First, we configured the Plugin and next, we had to configure the Tool

→ Goto Dashboard → Manage Jenkins → Tools

→ Click Apply and Save

The screenshot shows the Jenkins Tools configuration screen. The title is 'Dependency-Check installations'. There's a 'Add Dependency-Check' button. Below it, there's a section for 'Dependency-Check' with a 'Name' field containing 'DP-Check' and a checked 'Install automatically' checkbox. Underneath, there's a 'Install from github.com' section with a 'Version' dropdown set to 'dependency-check 6.5.1'. At the bottom, there are 'Save' and 'Apply' buttons.

→ Now go configure → Pipeline and add this stage to your pipeline apply and save.

The screenshot shows the Jenkins Pipeline configuration page. The pipeline script contains two stages: 'OWASP FS SCAN' and 'TRIVY FS SCAN'. The 'OWASP FS SCAN' stage uses dependencyCheck with specific arguments and an ODC installation. The 'TRIVY FS SCAN' stage runs a shell command to generate a report. A 'Use Groovy Sandbox' checkbox is checked. Below the script, there are 'Save' and 'Apply' buttons.

```
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"
    }
}
```

→ Click on Build Now

→ The stage view would look like this,

The screenshot shows the Jenkins Stage View. It displays average stage times for a recent build. The stages and their average times are:

Stage	Average Time
Declarative: Tool Install	1min 1s
clean workspace	421ms
Checkout from Git	3s
Sonarqube Analysis	25s
quality gate	532ms
Install Dependencies	35s
OWASP FS SCAN	9min 34s
TRIVY FS SCAN	43s

Below the table, the SonarQube Quality Gate status is shown as 'Passed' with a green button. The Permalinks section lists several recent builds.

Average stage times:  
(Average full run time: ~7min 18s)

#	Date	Changes	Time	Step	Time	Step	Time	Step	Time	Step	Time	Step	Time	Step	Time	Step	
#2	Feb 22 12:57	No Changes	172ms	clean workspace	421ms	Checkout from Git	3s	Sonarqube Analysis	25s	Install Dependencies	35s	OWASP FS SCAN	9min 34s	TRIVY FS SCAN	43s		
#1	Feb 22 12:41	No Changes	2min 2s	Declarative: Tool Install	1min 1s	clean workspace	421ms	Checkout from Git	3s	Sonarqube Analysis	25s	Install Dependencies	35s	OWASP FS SCAN	9min 34s	TRIVY FS SCAN	43s

**SonarQube Quality Gate**

**Permalinks**

- Last build (#1), 15 min ago
- Last stable build (#1), 15 min ago
- Last successful build (#1), 15 min ago
- Last completed build (#1), 15 min ago

→ You will see that in status, a graph will also be generated and Vulnerabilities.

→ Refresh the page after build.

The screenshot shows the Jenkins Pipeline Status page for the 'zomato' project. It displays the following information:

- Pipeline Syntax:** Shows the pipeline syntax with two builds: #2 (Feb 22, 12:57) and #1 (Feb 22, 12:41). Both builds show "No Changes".
- Average stage times:** Average full run time: ~7min 18s. Stage times: 1min 1s, 421ms, 3s, 25s, 532ms, 35s, 9min 34s, 43s.
- SonarQube Quality Gate:** zomato Passed (server-side processing: Success).
- Permalinks:** Links to the latest dependency check findings.

The URL in the address bar is 18.144.5.6:8080/job/zomato/lastCompletedBuild/dependency-check-findings/

→ Click on the Latest Dependency-Check

The screenshot shows the Jenkins Dependency-Check Report page for the 'zomato' project, specifically for build #2. It displays the following information:

- Status:** Dependency-Check
- Severity Distribution:** A chart showing the distribution of vulnerabilities across severity levels (5, 8, 6).
- Dependency-Check Results:** A table listing vulnerabilities found in various files. The columns include File Name, Vulnerability, Severity, and Weakness.

File Name	Vulnerability	Severity	Weakness
css-what:3.4.2	OSSINDEX CVE-2022-21222	High	CWE-1333
ejs:3.1.8	OSSINDEX CVE-2023-29827	High	CWE-74
follow-redirects:1.15.2	NVD CVE-2023-26159	Medium	CWE-601
json5:1.0.1	NVD CVE-2022-46175	High	CWE-1321
jsonpointer:5.0.1	NVD CVE-2022-4742	Critical	CWE-1321
nth-check:1.0.2	NVD CVE-2021-3803	High	CWE-1333
parseuri:1.3.3	NVD CVE-2022-0722	High	CWE-200
parseuri:1.3.3	NVD CVE-2022-2216	Critical	CWE-918
parseuri:1.3.3	NVD CVE-2022-2217	Medium	CWE-79
parseuri:1.3.3	NVD CVE-2022-2218	Medium	CWE-79

## # Step 6 - Docker Image Build and Push

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

→ Docker

→ Docker Commons

→ Docker Pipeline

→ Docker API

→ docker-build-step

and click on install for all plugins without restart

The screenshot shows the Jenkins plugin manager interface. The search bar at the top is set to 'Docker'. On the left, there's a sidebar with options: 'Updates' (unchecked), 'Available plugins' (selected and highlighted in grey), 'Installed plugins' (unchecked), 'Advanced settings' (unchecked), and 'Download progress' (unchecked). The main area displays a list of Docker-related plugins:

- Docker** 1.6  
Cloud Providers | Cluster Management | docker  
This plugin integrates Jenkins with Docker  
Released 10 days ago
- Docker Commons** 439.va\_3cb\_0a\_6a\_fb\_29  
Library plugins (for use by other plugins) | docker  
Provides the common shared functionality for various Docker-related plugins.  
Released 7 mo 16 days ago
- Docker Pipeline** 572.v950f58993843  
pipeline | DevOps | Deployment | docker  
Build and use Docker containers from pipelines.  
Released 6 mo 14 days ago
- Docker API** 3.3.4-86.v39b\_a\_5ede342c  
Library plugins (for use by other plugins) | docker  
This plugin provides docker-java API for other plugins.  
This plugin is up for adoption! We are looking for new maintainers. Visit our [Adopt a Plugin](#) initiative for more information.  
Released 2 mo 25 days ago
- docker-build-step** 2.11  
Build Tools | docker  
This plugin allows to add various docker commands to your job as build steps.  
Released 1 mo 18 days ago

→ Now, Goto Dashboard → Manage Jenkins → Tools

→ Click on Apply and Save

The screenshot shows the Jenkins 'Docker installations' configuration page. A 'Docker' item is selected under the 'Tools' section. A new configuration card for 'docker' is being added. The 'Name' field contains 'docker'. The 'Install automatically' checkbox is checked. Under 'Download from docker.com', the 'Docker version' is set to 'latest'. There is a 'Save' button at the bottom left and an 'Apply' button at the bottom right.

→ Now, Goto Dashboard → Manage Jenkins → Credentials → System → Global credentials (unrestricted)

→ Add DockerHub Username and Password under Global Credentials

→ Click on Create

The screenshot shows the Jenkins 'New credentials' creation page for a 'Username with password' credential. The 'Kind' dropdown is set to 'Username with password'. The 'Scope' dropdown is set to 'Global (Jenkins, nodes, items, all child items, etc)'. The 'Username' field contains 'iamajaygaur'. The 'Password' field is filled with a redacted password. The 'ID' field is set to 'docker'. The 'Description' field is set to 'docker'. At the bottom, there is a 'Create' button.

Screenshot of the Jenkins Global credentials (unrestricted) page.

**Global credentials (unrestricted)**

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
sonar	sonar	Secret text	sonar
docker	iamajaygaur/******** (docker)	Username with password	docker

Icon: S M L

REST API Jenkins 2.440.1

→ Add this stage to Pipeline Script

→ Click on Apply and Save

Screenshot of the Jenkins Pipeline Configuration page for the zomato job.

**Configure**

Definition: Pipeline script

Script:

```

53-
54-
55-
56-
57-
58-
59-
60-
61-
62-
63-
64-
65-
66-
67-
68-
69-

```

```

steps{
    script{
        withDockerRegistry(credentialsId: 'docker', toolName: 'docker'){
            sh "docker build -t zomato ."
            sh "docker tag zomato iamajaygaur/zomato:latest"
            sh "docker push iamajaygaur/zomato:latest"
        }
    }
}
stage("TRIVY"){
    steps{
        sh "trivy image iamajaygaur/zomato:latest > trivy.txt"
    }
}

```

try sample Pipeline...  Use Groovy Sandbox

Pipeline Syntax

Save Apply

REST API Jenkins 2.440.1

→ Build Now

→ You will see the output below, with a dependency trend.

→ Refresh the page

The screenshot shows the Jenkins interface for the 'zomato' project. The top navigation bar includes links for Instances, Jenkins, Projects, and a search bar. The main dashboard features a 'Status' card for the 'zomato' job, which is currently green. Below this are links for Changes, Build Now, Configure, Delete Pipeline, Full Stage View, SonarQube, Rename, and Pipeline Syntax. To the right is a 'Dependency-Check Trend' chart showing a single point at level 0. The 'Stage View' section displays a grid of stages: Declarative: Tool Install (40s), clean workspace (395ms), Checkout from Git (2s), Sonarqube Analysis (25s), quality gate (474ms), Install Dependencies (31s), OWASP FS SCAN (8min 14s), TRIVY FS SCAN (38s), Docker Build & Push (4min 25s), and TRIVY (1min 38s). Each stage has a progress bar and a tooltip indicating the average stage time.

The screenshot shows a Jenkins pipeline interface. On the left, there's a sidebar with 'Build History' showing three recent builds (#3, #2, #1) with their run times (22 Feb 2024, 07:59, 22 Feb 2024, 07:27, 22 Feb 2024, 07:11). Below it are links for 'Atom feed for all' and 'Atom feed for failures'. The main area is titled 'Stage View' and displays a table of stage times for a declarative pipeline. The table has columns for 'Declarative: Tool Install', 'clean workspace', 'Checkout from Git', 'Sonarqube Analysis', 'quality gate', 'Install Dependencies', 'OWASP FS SCAN', 'TRIVY FS SCAN', 'Docker Build & Push', and 'TRIVY'. The first row shows average times: 40s, 395ms, 2s, 25s, 474ms, 31s, 8min 14s, 38s, 4min 25s, and 1min 38s. Subsequent rows show specific execution times for each stage. The 'Sonarqube Analysis' stage in the first row is highlighted in red, indicating a failure. The 'Install Dependencies' stage in the second row is also red. The 'Docker Build & Push' and 'TRIVY' stages in the third row are green. At the bottom, there's a summary for the 'SonarQube Quality Gate' which is 'Passed' (green button) and 'Success' (green button). A 'Latest Dependency-Check' icon is also present.

→ When you log in to Dockerhub, you will see a new image is created.

The screenshot shows the Dockerhub interface. At the top, there are tabs for 'Instances | EC2 | us-west-1', 'zomato [Jenkins]', 'Projects', and 'iamajaygaur/zomato general'. Below the tabs, the URL is hub.docker.com/repository/docker/iamajaygaur/zomato/general. The main navigation bar has 'dockerhub', 'Explore', 'Repositories', 'Organizations', a search bar 'Search Docker Hub', and a 'ctrl+K' button. Under 'Repositories', it shows 'iamajaygaur / Repositories / zomato / General'. A message says 'Using 0 of 1 private repositories. [Get more](#)'. Below this, there are tabs for 'General', 'Tags', 'Builds', 'Collaborators', 'Webhooks', and 'Settings'. The 'General' tab is selected. On the left, there's a 'Tags' section showing one tag: 'latest' (Image type, pushed 5 minutes ago). On the right, there's a 'Docker commands' section with a 'Public View' button and a command box containing 'docker push iamajaygaur/zomato:tagname'. Below this is an 'Automated Builds' section with a 'Upgrade' button.

→ Now Run the container to see if the app coming up or not by adding the below stage

→ Click on Apply and Save

The screenshot shows the Jenkins configuration page for a job named 'zomato Config [Jenkins]'. The URL is 18.144.5.6:8080/job/zomato/configure. The page has tabs for 'Dashboard', 'zomato', and 'Configuration'. In the 'Configuration' tab, under 'Advanced Project Options', the 'Pipeline' tab is selected. The 'Definition' dropdown is set to 'Pipeline script'. The 'Script' editor contains the following Groovy code:

```
stage('Deploy to container'){
    steps{
        sh 'docker run -d --name zomato -p 3000:3000 iamajaygaur/zomato:latest'
    }
}
```

Below the script, there's a checkbox 'Use Groovy Sandbox' which is checked. At the bottom, there are 'Save' and 'Apply' buttons.

→ Build Now

→ You can see the Container is deployed successfully in the below stages

The screenshot shows the Jenkins Pipeline Stage View. On the left, there's a sidebar with 'Build History' and a 'trend' dropdown. Below it is a 'Filter...' search bar. The main area displays a grid of build stages for four recent builds (#4, #3, #2, #1). Each row contains a timestamp and a 'No Changes' note. The columns represent various stages: Declarative: Tool Install, clean workspace, Checkout from Git, SonarQube Analysis, quality gate, Install Dependencies, OWASP FS SCAN, TRIVY FS SCAN, Docker Build & Push, and Deploy to container. The 'Deploy to container' stage is highlighted in green. Below the grid, a summary says 'Average stage times: (Average full run time: ~10min 24s)'. At the bottom, there's a 'SonarQube Quality Gate' section showing 'zomato' has passed with a green 'Passed' button and 'server-side processing: Success'. There's also a 'Latest Dependency-Check' link.

→ You can see now the Zomato app is deployed Successfully

→ <http://<Jenkins-public-ip>:3000>

→ You will get this output

The screenshot shows the Zomato website homepage. The header features the Zomato logo and the tagline 'Discover the best food & drinks in Patna'. Below the header is a search bar with 'Chennai' selected. The main background image shows various food items like chicken wings and a bowl of soup. At the bottom, there are three cards: 'Order Online' (showing a restaurant interior), 'Nightlife and Clubs' (showing silhouettes of people at night), and 'Dinning' (showing a plate of food).

Order Online  
Stay home and order to your doorstep

Nightlife and Clubs  
Explore the city's top nightlife outlets

Dinning  
Views the city's favourite venues

## Collection

Explore curated lists of top restaurants, cafes, pubs, and bars in Ahmedabad, based on trends

[All collection in Ahmedabad](#)



Bodakdev 345 Places	Setellite 336 Places	Gurukul 83 Places	Navrangpura 302 Places
Vastrapur 217 Places	Thaltej 222 Places	Prahalad Nagar 181 Places	C G Road 94 Places

See more

## Get the Zomato app

We will send you a link, open it on your phone to download the app

Email    Phone

Email

Share App Link

Download app from



The screenshot shows a search interface with three dropdown menus:

- Popular cuisines near me
- Popular restaurant types near me
- Top Restaurant Chains

Below the search interface, there is a screenshot of the Zomato website:

**Zomato**

ABOUT ZOMATO

- Who We Are
- Blog
- Work With Us
- Investor Relations
- Report Fraud
- Contact Us

ZOMAVERSE

- Zomato
- Blinkit
- Feeding India
- HyperPure
- Zomaland

FOR RESTAURANTS

- Partner With Us
- Apps For You
- For Enterprises
- Zomato For Work

LEARN MORE

- Privacy
- Security
- Terms
- Sitemap

SOCIAL LINKS

- India
- English

Links to social media and app stores:

- Facebook
- Twitter
- Instagram
- YouTube
- App Store
- Google Play

## # Step - 8: Terminate instances.

The screenshot shows the AWS EC2 Instances page. A success message at the top says "Successfully terminated i-044f7b2215c8b20ea". The main table displays one instance:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
zomato-app-clone	i-044f7b2215c8b20ea	Shutting-d...	t2.large	2/2 checks passed	View alarms +	us-west-1b

The instance details page for "zomato-app-clone" shows the following information:

**Details**

**Instance summary**

Instance ID	Public IPv4 address	Private IPv4 addresses
i-044f7b2215c8b20ea (zomato-app-clone)	18.144.5.6 [open address]	172.31.13.176
IPv6 address	Instance state	Public IPv4 DNS
-	Shutting-down	ec2-18-144-5-6.us-west-1.compute.amazonaws.com [open address]
Hostname type	Private IP DNS name (IPv4 only)	Elastic IP addresses
IP name: ip-172-31-13-176.us-west-1.compute.internal	ip-172-31-13-176.us-west-1.compute.internal	-
Answer private resource DNS name	Instance type	
IPv4 (A)	t2.large	