

## Mr Cloud Book

Home Blog DevSecOps Contact About Me Testimonials

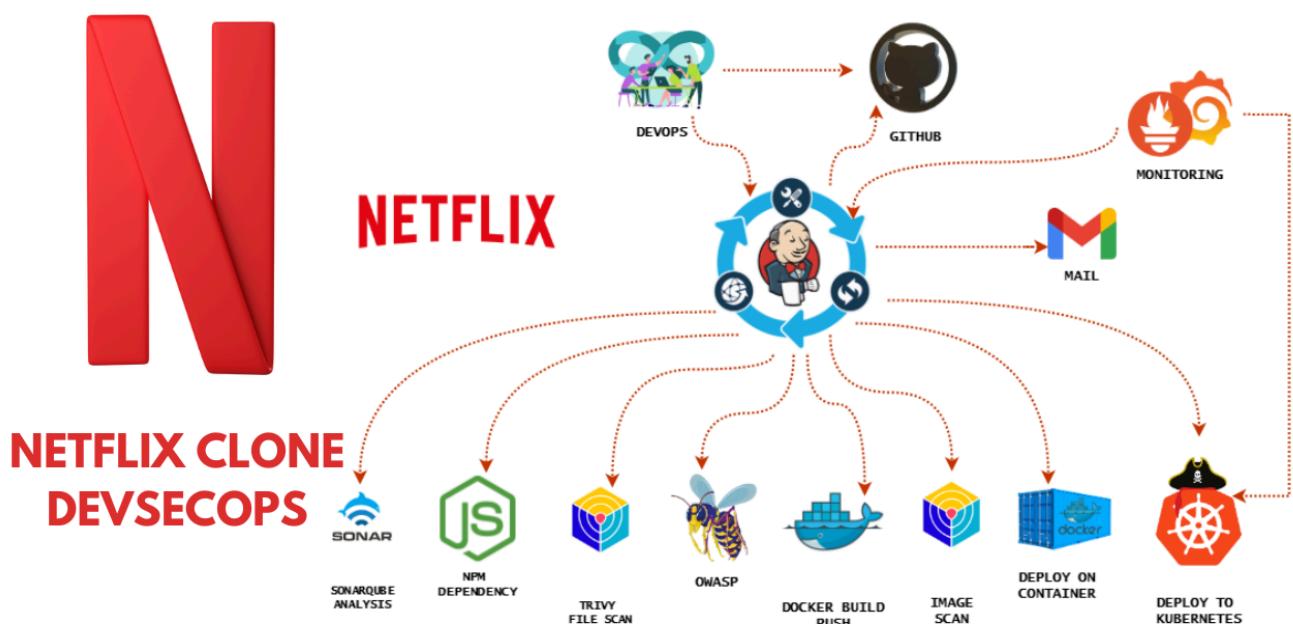
Search Blogs

DevOps

# Netflix Clone CI/CD with Monitoring | Jenkins | Docker| Kubernetes| Monitoring | DevSecOps



mrcloudbook.com · 8 January 2024



Hello friends, we will be deploying a Netflix clone. We will be using Jenkins as a CICD tool and deploying our application on a Docker container and Kubernetes Cluster and we will monitor the Jenkins and Kubernetes metrics using Grafana, Prometheus and Node exporter. I Hope this detailed blog is useful.

## [CLICK HERE FOR GITHUB REPOSITORY](#)

### **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 – Create a TMDB API Key.

Step 4 – Install Prometheus and Grafana On the new Server.

Step 5 – Install the Prometheus Plugin and Integrate it with the Prometheus server.

Step 6 – Email Integration With Jenkins and Plugin setup.

Step 7 – Install Plugins like JDK, Sonarqube Scanner, Nodejs, and OWASP Dependency Check.

Step 8 – Create a Pipeline Project in Jenkins using a Declarative Pipeline

Step 9 – Install OWASP Dependency Check Plugins

Step 10 – Docker Image Build and Push

Step 11 – Deploy the image using Docker

Step 12 – Kubernetes master and slave setup on Ubuntu (20.04)

Step 13 – Access the Netflix app on the Browser.

Step 14 – Terminate the AWS EC2 Instances.

**Now, let's get started and dig deeper into each of these steps:-**

## Contents [ hide ]

STEP1:Launch an Ubuntu(22.04) T2 Large Instance  
Step 2 – Install Jenkins, Docker and Trivy  
2A – To Install Jenkins  
2B – Install Docker  
2C – Install Trivy  
Step 3: Create a TMDB API Key  
Step 4 – Install Prometheus and Grafana On the new Server  
Install Node Exporter on Ubuntu 22.04  
Install Grafana on Ubuntu 22.04  
Step 5 – Install the Prometheus Plugin and Integrate it with the Prometheus server  
Step 6 – Email Integration With Jenkins and Plugin Setup  
Step 7 – Install Plugins like JDK, Sonarqube Scanner, NodeJs, OWASP Dependency Check  
7A – Install Plugin  
7B – Configure Java and Nodejs in Global Tool Configuration  
7C – Create a Job  
Step 8 – Configure Sonar Server in Manage Jenkins  
Step 9 – Install OWASP Dependency Check Plugins  
Step 10 – Docker Image Build and Push  
Step 11 – Kuberentes Setup  
Kubectl is to be installed on Jenkins also  
Part 1 -----Master Node-----  
-----Worker Node-----  
Part 2 -----Both Master & Node -----  
Part 3 ----- Master -----  
-----Worker Node-----  
Install Node\_exporter on both master and worker  
STEP 12:Access from a Web browser with  
Step 13: Terminate instances.  
Complete Pipeline

## STEP1: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).

Instances (1) <a href="#">Info</a>		<a href="#">C</a>	Connect	Instance state	Actions	Launch instances		
	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 D
<input type="checkbox"/>	CI-CD	i-065c10200537a1eee	<span>Running</span>	t2.large	<span>2/2 checks passed</span>	No alarms	ap-south-1a	ec2-52-66-14

## Step 2 – Install Jenkins, Docker and Trivy

### 2A – To Install Jenkins

Connect to your console, and enter these commands to Install Jenkins

```
vi jenkins.sh #make sure run in Root (or) add at userdata while ec2 lau
```



```
#!/bin/bash
sudo apt update -y
#sudo apt upgrade -y
wget -O - https://packages.adoptium.net/artifactory/api/gpg/key/public
echo "deb [signed-by=/etc/apt/keyrings/adoptium.asc] https://packages.adoptium.net/artifactory/api/debian stable main" | sudo tee /etc/apt/sources.list.d/adoptium.list &gt;>
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
```



```
sudo chmod 777 jenkins.sh  
./jenkins.sh # this will install jenkins
```

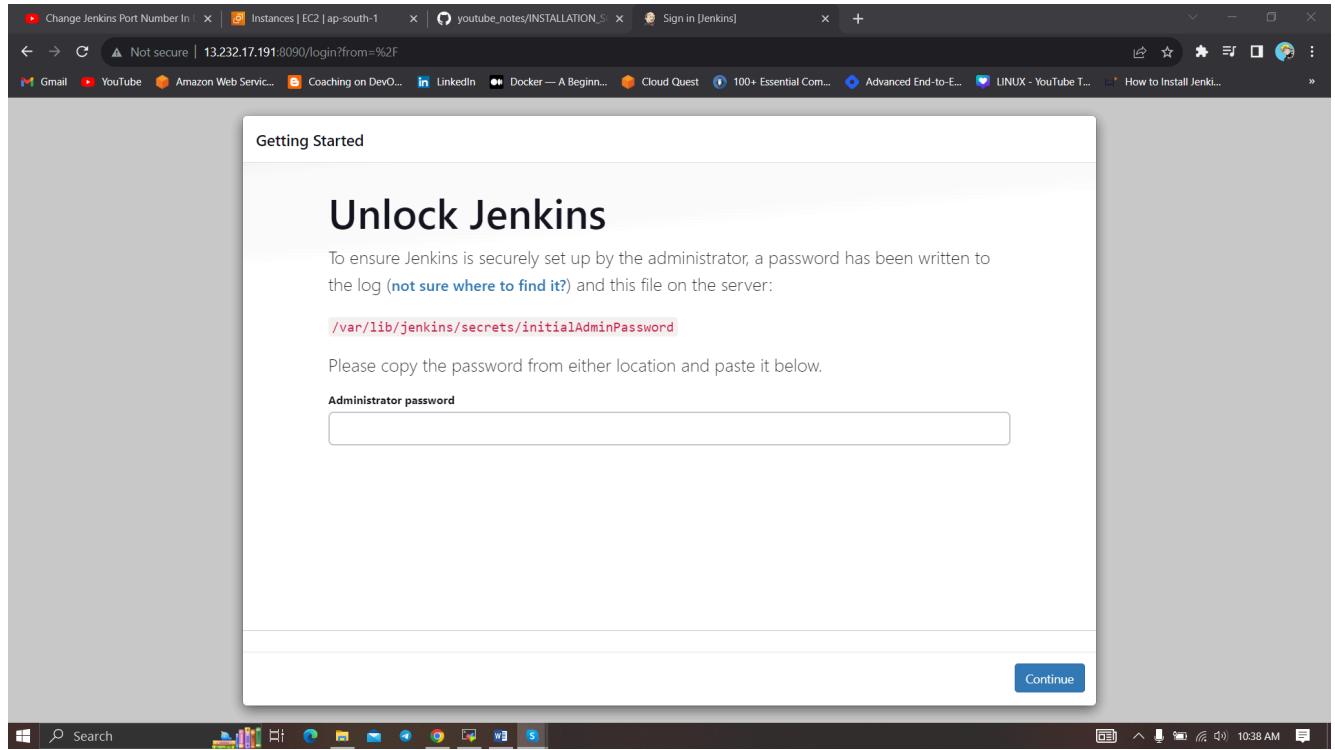


Once Jenkins is installed, you will need to go to your AWS EC2 Security Group and open Inbound Port 8080, since Jenkins works on Port 8080.

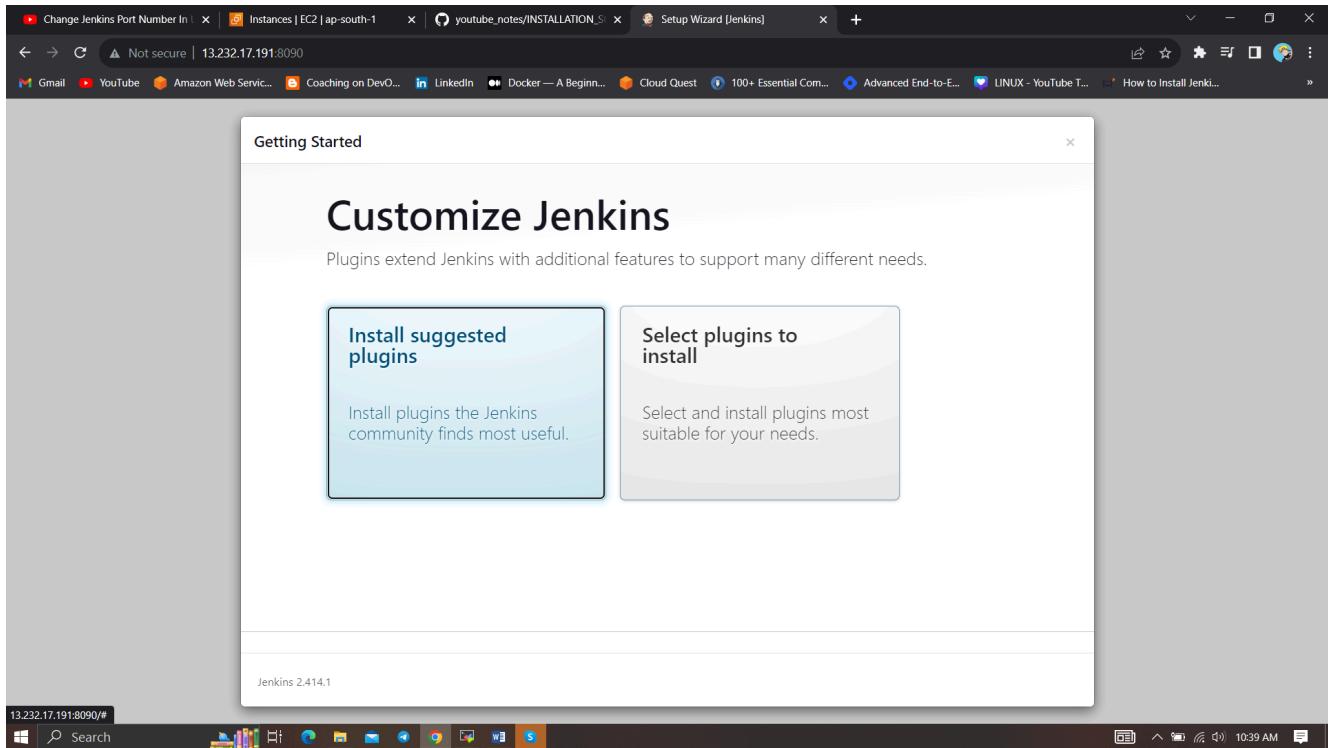
Now, grab your Public IP Address



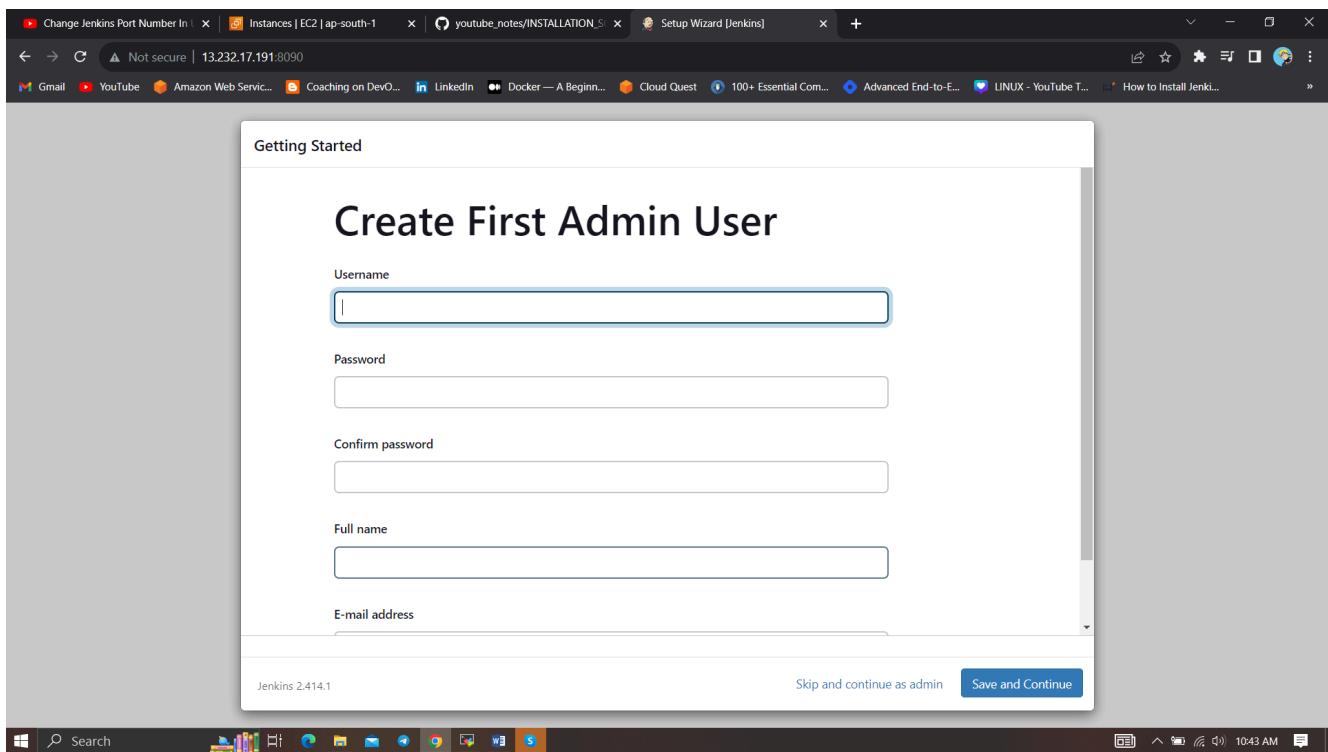
```
<EC2 Public IP Address:8080>  
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```



Unlock Jenkins using an administrative password and install the suggested plugins.



Jenkins will now get installed and install all the libraries.



Create a user click on save and continue.

Jenkins Getting Started Screen.

Dashboard >

+ New Item      Add description

People      Build History      Manage Jenkins

My Views

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Build Queue

No builds in the queue.

Build Executor Status

1 Idle      2 Idle

Create a job      →

Set up a distributed build

Set up an agent      →

Configure a cloud      →

Learn more about distributed builds      ↗

## 2B – Install Docker

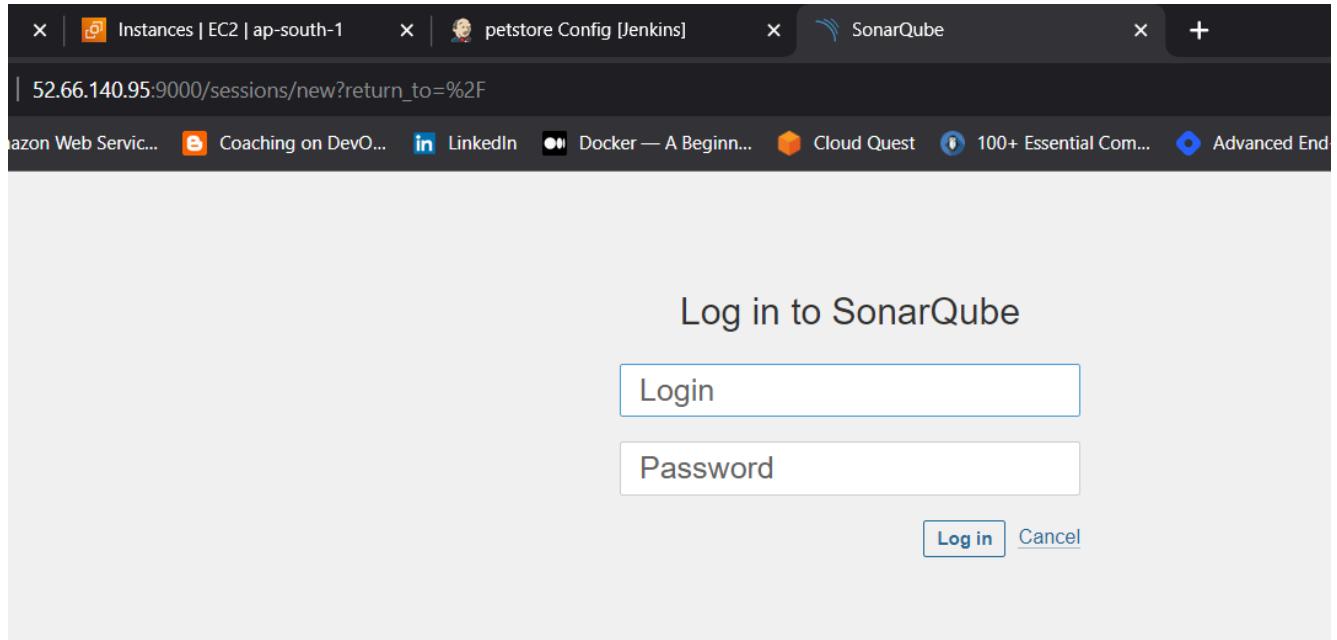
```
sudo apt-get update
sudo apt-get install docker.io -y
sudo usermod -aG docker $USER      #my case is ubuntu
newgrp docker
sudo chmod 777 /var/run/docker.sock
```

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
```

```
ubuntu@ip-172-31-42-253:~$ sudo chmod 777 /var/run/docker.sock
ubuntu@ip-172-31-42-253:~$ docker run -d --name sonar 9000:9000 sonarqube:lts-community
Unable to find image 'sonarqube:lts-community' locally
lts-community: Pulling from library/sonarqube
44ba2882fb8e: Pull complete
2cabec57fa36: Pull complete
c20481384b6a: Pull complete
bf7b17ee74f8: Pull complete
38617faac714: Pull complete
706f20f58f5e: Pull complete
65a29568c257: Pull complete
Digest: sha256:1a118f8ab960d6c3d4ea8b4455a5a6560654511c88a6816f1603f764d5dcc77c
Status: Downloaded newer image for sonarqube:lts-community
4b66c96bf9ad3d62289436af7f752fdb04993092d0ca5065e2f2e32301b50139
ubuntu@ip-172-31-42-253:~$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
4b66c96bf9ad sonarqube:lts-community "/opt/sonarqube/dock..." 9 seconds ago Up 5 seconds 0.0.0.0:9000->9000/tcp, :::9000->9000/tcp sonar
ubuntu@ip-172-31-42-253:~$
```

Now our sonarqube is up and running



Enter username and password, click on login and change password



```
username admin
password admin
```



Instances | EC2 | ap-south-1 x petstore Config [Jenkins] x SonarQube x +

6.140.95:9000/account/reset\_password

Web Servic... Coaching on DevO... LinkedIn Docker — A Beginn... Cloud Quest 100+ Essential Com... Advanced End-to-E...

## Update your password

This account should not use the default password.

Enter a new password

All fields marked with \* are required

**Old Password \***

**New Password \***

**Confirm Password \***

**Update**

Update New password, This is Sonar Dashboard.

← → C Not secure | 52.66.140.95:9000/projects/create

Gmail YouTube Amazon Web Service... Coaching on DevO... LinkedIn Docker — A Beginn... Cloud Quest 100+ Essential Com... Advanced End-to-E... LINUX - YouTube T... How to Install Jenk...

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

How do you want to create your project?

Do you want to benefit from all of SonarQube's features (like repository import and Pull Request decoration)? Create your project from your favorite DevOps platform. First, you need to set up a DevOps platform configuration.

From Azure DevOps Set up global configuration	From Bitbucket Server Set up global configuration	From Bitbucket Cloud Set up global configuration	From GitHub Set up global configuration	From GitLab Set up global configuration
--	--	---	--	--

## 2C – Install Trivy



```
vi trivy.sh
```



```
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 > /usr/share/keyrings/trivy.gpg  
echo "deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasecurity.github.io/trivy-repo/deb/ /" | sudo tee /etc/apt/sources.list.d/trivy.list  
sudo apt-get update  
sudo apt-get install trivy -y
```



## Step 3: Create a TMDB API Key

Next, we will create a TMDB API key

Open a new tab in the Browser and search for TMDB

Google

TMDB

All Images News Videos Shopping More Tools SafeSearch

About 4,66,00,000 results (0.45 seconds)

The Movie Database

<https://www.themoviedb.org>

The Movie Database (TMDB)

The Movie Database (TMDB) is a popular, user editable database for movies and TV shows.

API Reference

Welcome to version 3 of The Movie Database (TMDB) API. This is ...

API key

TMDB Talk - API Terms of Use - API Reference - OAS - Basics

Login to your account

The Movie Database (TMDB) is a popular, user editable database ...

Popular Movies

Now Playing - Top Rated Movies - Upcoming Movies - ...

**THE MOVIE DB**

LARAVEL TMDB

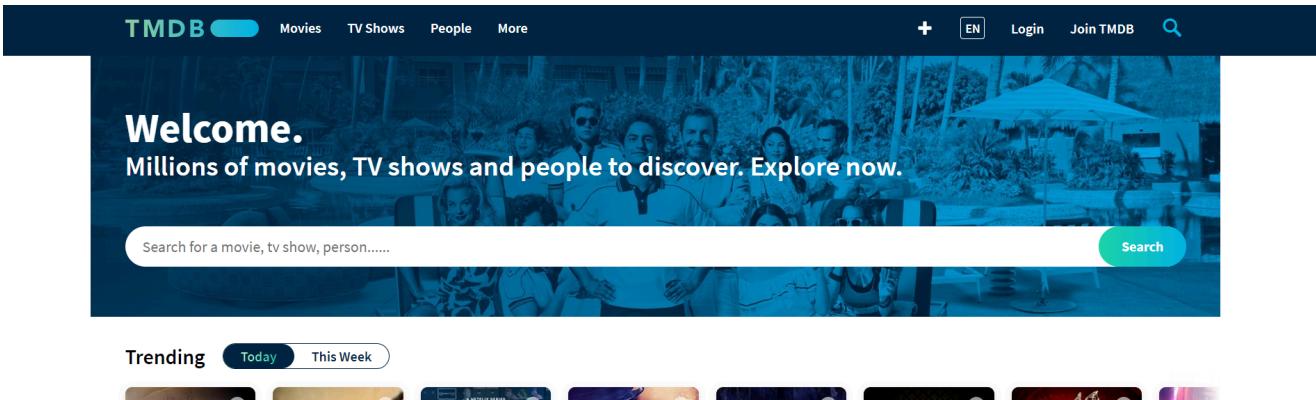
themoviedb.org

Category: film database

Date launched: 2008

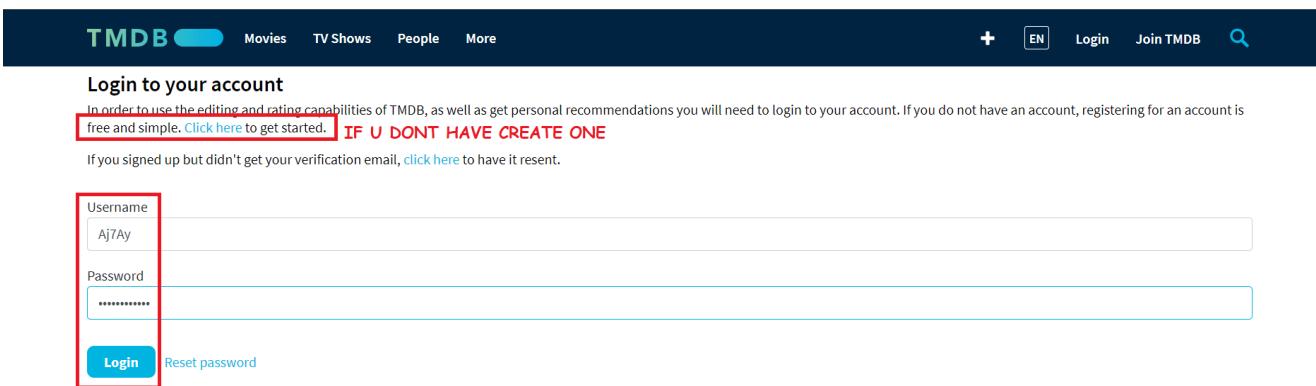
Owner: Fan TV

Click on the first result, you will see this page

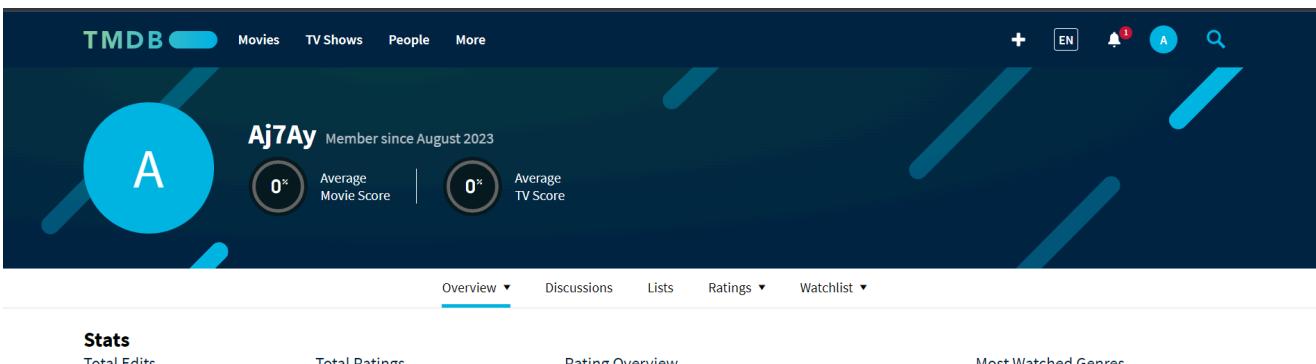


Click on the Login on the top right. You will get this page.

You need to create an account here. click on click here. I have account that's why i added my details there.



once you create an account you will see this page.



Let's create an API key, By clicking on your profile and clicking settings.

Welcome.  
Millions of movies, TV shows and people to discover. Explore now.

Search for a movie, tv show, person.....

Trending Today This Week

Loki (82)  
The Equalizer (72)  
LUPIN (78)  
Totally Killer (68)  
The Haunted Mansion (68)  
Nuuk (70)  
Pet Sematary: Bloodlines (65)  
Ball (75)

Aj7Ay View profile  
Discussions  
Lists  
Ratings  
Watchlist  
Edit Profile  
**Settings**  
Logout

Now click on API from the left side panel.

TMDB Movies TV Shows People More + EN A 🔍

Aj7Ay

**Settings**

**Account Settings**

Default Language: English (en-US)  
Fallback Language: None (Don't Fallback)  
Country: India  
Timezone - Auto detect?  Asia - Kabul  
Include Adult Items in Search?: No  
Filter Profanity?: Yes

**API**

Delete Account

Now click on create

**Settings**

- Edit Profile
- Account Settings
- Streaming Services
- Notification Settings
- Blocked Users
- Import List
- Sharing Settings
- Sessions
- API**

**API** Overview **Create**  

TMDb offers a powerful API service that is free to use as long as you properly attribute us as the source of the data and/or images you use. You can find the logos for attribution [here](#).

**Documentation**

Our primary documentation is located at [developer.themoviedb.org](#).

**Support**

If you have questions or comments about the information covered here, please create a post on our [support forums](#).

**Request an API Key**

To generate a new API key, [click here](#).

## Click on Developer

**Settings**

**API** Overview **Create**  

What type of API key do you wish to register?

**Developer**  

- You are an individual
- Your project is still in development
- Your project is non profit
- Your project is ad supported

**Professional**

- You represent a company
- Your project is for profit (not ad supported)
- You are an OEM or hardware vendor

## Now you have to accept the terms and conditions.

**12. General Terms**

- Relationship of the Parties.** Notwithstanding any provision hereof, for all purposes of the Terms of Use, you and TMDb shall be and act independently and not as partner, joint venturer, agent, employee or employer of the other. You shall not have any authority to assume or create any obligation for or on behalf of TMDb, express or implied, and you shall not attempt to bind TMDb to any contract.
- Invalidity of Specific Terms.** If any provision of the Terms of Use is found by a court of competent jurisdiction to be invalid, the parties nevertheless agree that the other provisions of this agreement will remain in full force and effect and the court should endeavor to give effect to the parties' intentions as reflected in the invalid provision.
- Location of Lawsuit and Choice of Law.** THE TERMS OF USE AND THE RELATIONSHIP BETWEEN YOU AND TMDb SHALL BE GOVERNED BY THE LAWS OF THE STATE OF CALIFORNIA WITHOUT REGARD TO ITS CONFLICT OF LAW PROVISIONS. YOU AND TMDb AGREE TO SUBMIT TO THE PERSONAL JURISDICTION OF THE COURTS LOCATED WITHIN THE COUNTY OF SAN MATEO, CALIFORNIA.
- No Waiver of Rights by TMDb.** TMDb's failure to exercise or enforce any right or provision of the Terms of Use shall not constitute a waiver of such right or provision.
- No Transfer.** The rights and obligations of these Terms of Use are personal to you and may not be transferred by you, either voluntarily or by operation of law.
- Notice.** Any notice to be sent to you under these Terms of Use may be sent via email, post, or any other reasonable means, at the contact information provided by you to TMDb from time to time. It is your obligation to insure that this information is current.

**Miscellaneous.** The section headings and subheadings contained in this agreement are included for convenience only, and shall not limit or otherwise affect the terms of the Terms of Use. Any construction or interpretation to be made of the Terms of Use shall not be construed against the drafter. The Terms of Use constitute the entire agreement between TMDb and you with respect to the subject matter hereof.

This Agreement was last updated on: July 28, 2014.

**Cancel**   **Accept**

## Provide basic details

The screenshot shows the 'API' settings page. On the left sidebar, 'API' is selected. The main area displays application details: Type of Use (Desktop Application), Application Name (dEM), Application URL (NOT AVAILABLE), and an Application Summary (jdf). Below this, a sharing section contains a text input with the value 'jdfjkjkjkjkjkjk'. The right side shows form fields for First Name (Code), Last Name (Word), Email Address (redacted), Address 1 (ddddd), City (fffffssss), Zip Code (225588), and a dropdown menu for Phone Number (no parenthesis or dashes) with options like Mali, Malta, Marshall Islands, Martinique, Mauritania, Mauritius, Mavotte, and India.

**Sharing Settings**

**API**

**First Name** Code      **Last Name** Word

**Email Address** [REDACTED]

**Address 1** dddddd

**City** ffffffssss

**Zip Code** 225588

**Phone Number (no parenthesis or dashes)**

- Mali
- Malta
- Marshall Islands
- Martinique
- Mauritania
- Mauritius
- Mavotte
- India

**Submit**

Click on submit and you will get your API key.

The screenshot shows the 'API' settings page. On the left sidebar, 'API' is selected. In the main area, there is a section titled 'API Key' which contains the value '8f757ea1c6e3dc2deef92fb4682d7e78', highlighted with a red box. Below this, there is a section titled 'API Read Access Token' containing a long, complex token string.

**Notification Settings**

**Blocked Users**

**Import List**

**Sharing Settings**

**Sessions**

**API**

**Support**

If you have questions or comments about the information covered here, please create a post on our [support forums](#).

**API Details**

If you'd like to edit the details of your app, [click here](#).

**API Key**

8f757ea1c6e3dc2deef92fb4682d7e78

**API Read Access Token**

eyJhbGciOiJIUzI1NiJ9.eyJhdWQiOiI4Zjc1N2VhMWM2ZTNkYzJkZWVmOTJmYjQ2ODJkN2U3OCIsInN1YiI6IjY0NzE5NWRhODgxM2U0MDEwMzU2ZGNmNCIsInNjb3BlcyI6WyJhcGlfcmVhZCJdLCJ2ZXJzaW9uljoxfQ.K6Pu7To0wWp9C4PVW-ZvoViRjh-CsRpefc5CZt6ObX0

## Step 4 – Install Prometheus and Grafana On the new Server

First of all, let's create a dedicated Linux user sometimes called a system account for Prometheus. Having individual users for each service serves two main purposes:

It is a security measure to reduce the impact in case of an incident with the service.

It simplifies administration as it becomes easier to track down what resources belong to which service.

To create a system user or system account, run the following command:

```
sudo useradd \
--system \
--no-create-home \
--shell /bin/false prometheus
```



```
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ sudo useradd \
--system \
--no-create-home \
--shell /bin/false prometheus
ubuntu@ip-172-31-38-156:~$ █
```

-system – Will create a system account.

-no-create-home – We don't need a home directory for Prometheus or any other system accounts in our case.

-shell /bin/false – It prevents logging in as a Prometheus user.

Prometheus – Will create a Prometheus user and a group with the same name.

Let's check the latest version of Prometheus from the [download page](#).

You can use the curl or wget command to download Prometheus.

```
wget https://github.com/prometheus/prometheus/releases/download/v2.47.1,
```

```
ubuntu@ip-172-31-38-156:~$ wget https://github.com/prometheus/prometheus/releases/download/v2.47.1/prometheus-2.47.1.linux-amd64.tar.gz
--2023-10-06 08:51:59-- https://github.com/prometheus/prometheus/releases/download/v2.47.1/prometheus-2.47.1.linux-amd64.tar.gz
Resolving github.com (github.com)... 20.207.73.82
Connecting to github.com (github.com)|20.207.73.82|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 95713066 (91M) [application/octet-stream]
Saving to: 'prometheus-2.47.1.linux-amd64.tar.gz'

prometheus-2.47.1.linux-amd64.tar.gz      100%[=====]  91.28M  4.01MB/s   in 88s

2023-10-06 08:53:28 (1.03 MB/s) - 'prometheus-2.47.1.linux-amd64.tar.gz' saved [95713066/95713066]

ubuntu@ip-172-31-38-156:~$ ls
ubuntu@ip-172-31-38-156:~$ ls
prometheus-2.47.1.linux-amd64.tar.gz
ubuntu@ip-172-31-38-156:~$
```

Then, we need to extract all Prometheus files from the archive.

```
tar -xvf prometheus-2.47.1.linux-amd64.tar.gz
```

```
ubuntu@ip-172-31-38-156:~$ ls
prometheus-2.47.1.linux-amd64.tar.gz
ubuntu@ip-172-31-38-156:~$ tar -xvf prometheus-2.47.1.linux-amd64.tar.gz
prometheus-2.47.1.linux-amd64/
prometheus-2.47.1.linux-amd64/LICENSE
prometheus-2.47.1.linux-amd64/NOTICE
prometheus-2.47.1.linux-amd64/prometheus.yml
prometheus-2.47.1.linux-amd64/consoles/
prometheus-2.47.1.linux-amd64/consoles/prometheus.html
prometheus-2.47.1.linux-amd64/consoles/prometheus-overview.html
prometheus-2.47.1.linux-amd64/consoles/node-cpu.html
prometheus-2.47.1.linux-amd64/consoles/index.html.example
prometheus-2.47.1.linux-amd64/consoles/node.html
prometheus-2.47.1.linux-amd64/consoles/node-disk.html
prometheus-2.47.1.linux-amd64/consoles/node-overview.html
prometheus-2.47.1.linux-amd64/promtool
prometheus-2.47.1.linux-amd64/console_libraries/
prometheus-2.47.1.linux-amd64/console_libraries/prom.lib
prometheus-2.47.1.linux-amd64/console_libraries/menu.lib
prometheus-2.47.1.linux-amd64/prometheus
ubuntu@ip-172-31-38-156:~$
```

Usually, you would have a disk mounted to the data directory. For this tutorial, I will simply create a /data directory. Also, you need a folder for Prometheus

configuration files.

```
sudo mkdir -p /data /etc/prometheus
```



```
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ sudo mkdir -p /data /etc/prometheus  
ubuntu@ip-172-31-38-156:~$
```

Now, let's change the directory to Prometheus and move some files.

```
cd prometheus-2.47.1.linux-amd64/
```



```
= ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ cd prometheus-2.47.1.linux-amd64/  
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$  
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$  
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$ ls -l  
total 236852  
-rw-r--r-- 1 ubuntu ubuntu 11357 Oct 4 11:05 LICENSE  
-rw-r--r-- 1 ubuntu ubuntu 3773 Oct 4 11:05 NOTICE  
drwxr-xr-x 2 ubuntu ubuntu 4096 Oct 4 11:05 console_libraries  
drwxr-xr-x 2 ubuntu ubuntu 4096 Oct 4 11:05 consoles  
-rwxr-xr-x 1 ubuntu ubuntu 124158156 Oct 4 10:35 prometheus  
-rw-r--r-- 1 ubuntu ubuntu 934 Oct 4 11:05 prometheus.yml  
-rwxr-xr-x 1 ubuntu ubuntu 118343283 Oct 4 10:38 promtool  
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
```

First of all, let's move the Prometheus binary and a promtool to the /usr/local/bin/. promtool is used to check configuration files and Prometheus rules.

```
sudo mv prometheus promtool /usr/local/bin/
```



```
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$  
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$  
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$ sudo mv prometheus promtool /usr/local/bin/  
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
```

Optionally, we can move console libraries to the Prometheus configuration directory. Console templates allow for the creation of arbitrary consoles using the Go templating language. You don't need to worry about it if you're just getting started.



```
sudo mv consoles/ console_libraries/ /etc/prometheus/
```



```
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$  
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$  
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$  
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$ sudo mv consoles/ console_libraries/ /etc/prometheus/  
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
```

Finally, let's move the example of the main Prometheus configuration file.



```
sudo mv prometheus.yml /etc/prometheus/prometheus.yml
```



```
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$  
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$  
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$ sudo mv prometheus.yml /etc/prometheus/prometheus.yml  
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$
```

To avoid permission issues, you need to set the correct ownership for the /etc/prometheus/ and data directory.



```
sudo chown -R prometheus:prometheus /etc/prometheus/ /data/
```



```
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$ 
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$ 
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$ sudo chown -R prometheus:prometheus /etc/prometheus/ /data/
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$ 
```

You can delete the archive and a Prometheus folder when you are done.



```
cd
rm -rf prometheus-2.47.1.linux-amd64.tar.gz
```



```
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$ 
ubuntu@ip-172-31-38-156:~/prometheus-2.47.1.linux-amd64$ cd ..
ubuntu@ip-172-31-38-156:~$ ll
total 93516
drwxr-x--- 5 ubuntu ubuntu 4096 Oct  6 08:54 .
drwxr-xr-x  3 root   root  4096 Oct  6 08:49 ..
-rw-----  1 ubuntu ubuntu 62 Oct  6 08:50 .Xauthority
-rw-r--r--  1 ubuntu ubuntu 220 Jan  6 2022 .bash_logout
-rw-r--r--  1 ubuntu ubuntu 3771 Jan  6 2022 .bashrc
drwx----- 2 ubuntu ubuntu 4096 Oct  6 08:50 .cache/
-rw-r--r--  1 ubuntu ubuntu 807 Jan  6 2022 .profile
drwx----- 2 ubuntu ubuntu 4096 Oct  6 08:49 .ssh/
-rw-r--r--  1 ubuntu ubuntu 0 Oct  6 08:50 .sudo_as_admin_successful
-rw-rw-r--  1 ubuntu ubuntu 165 Oct  6 08:53 .wget-hsts
drwxr-xr-x  2 ubuntu ubuntu 4096 Oct  6 08:56 prometheus-2.47.1.linux-amd64/
-rw-rw-r--  1 ubuntu ubuntu 95713066 Oct  4 11:15 prometheus-2.47.1.linux-amd64.tar.gz
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ rm -rf prometheus-2.47.1.linux-amd64.tar.gz
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ ls
prometheus-2.47.1.linux-amd64
ubuntu@ip-172-31-38-156:~$ 
```

Verify that you can execute the Prometheus binary by running the following command:



```
prometheus --version
```



```
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ prometheus --version
prometheus, version 2.47.1 (branch: HEAD, revision: c4d1a8beff37cc004f1dc4ab9d2e73193f51aaeb)
  build user:      root@4829330363be
  build date:    20231004-10:31:16
  go version:    go1.21.1
  platform:      linux/amd64
  tags:          netgo,builtinassets,stringlabels
ubuntu@ip-172-31-38-156:~$ 
```

To get more information and configuration options, run Prometheus Help.

```
prometheus --help
```



We're going to use some of these options in the service definition.

We're going to use Systemd, which is a system and service manager for Linux operating systems. For that, we need to create a Systemd unit configuration file.

```
sudo vim /etc/systemd/system/prometheus.service
```



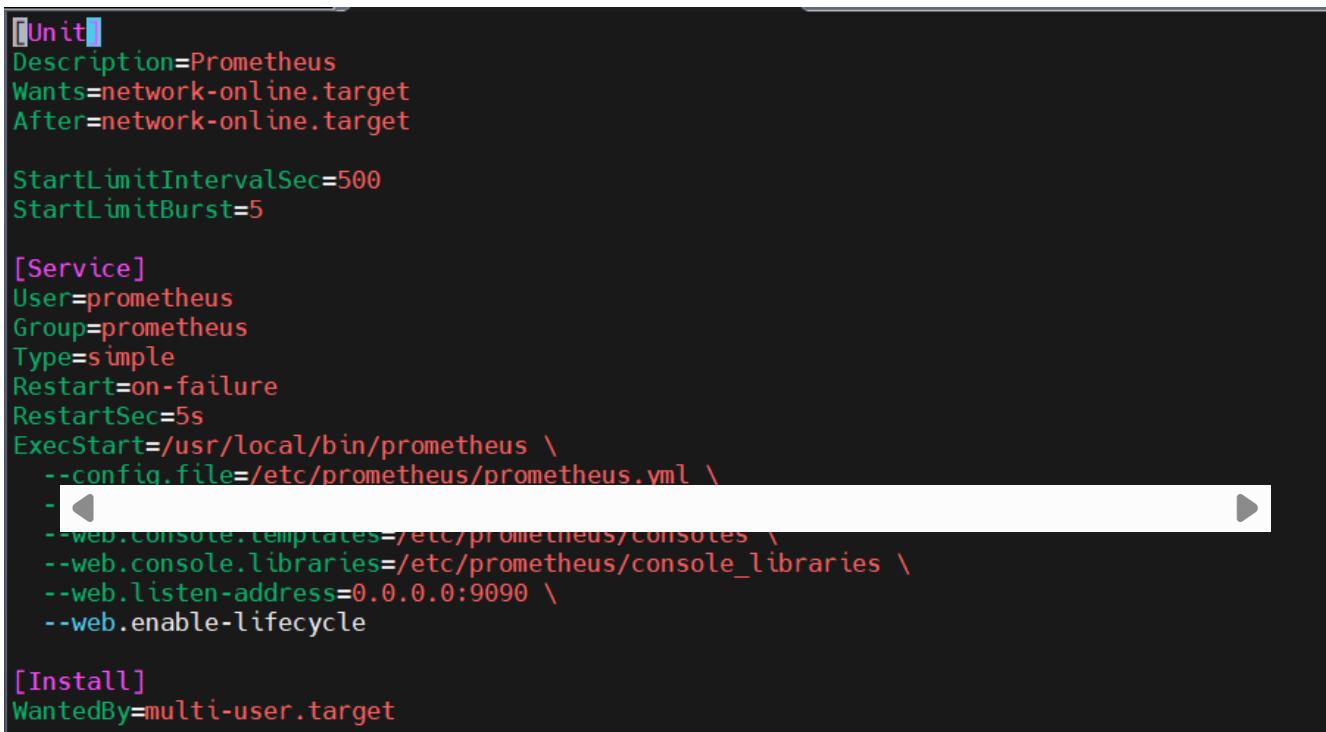
```
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ sudo vim /etc/systemd/system/prometheus.service
```

## Prometheus.service

```
[Unit]  
Description=Prometheus  
Wants=network-online.target  
After=network-online.target  
StartLimitIntervalSec=500  
StartLimitBurst=5  
[Service]  
User=prometheus  
Group=prometheus  
Type=simple  
Restart=on-failure  
RestartSec=5s  
ExecStart=/usr/local/bin/prometheus \
```

```
--config.file=/etc/prometheus/prometheus.yml \
--storage.tsdb.path=/data \
--web.console.templates=/etc/prometheus/consoles \
--web.console.libraries=/etc/prometheus/console_libraries \
--web.listen-address=0.0.0.0:9090 \
--web.enable-lifecycle

[Unit]
WantedBy=multi-user.target
```



```
[Unit]
Description=Prometheus
Wants=network-online.target
After=network-online.target

StartLimitIntervalSec=500
StartLimitBurst=5

[Service]
User=prometheus
Group=prometheus
Type=simple
Restart=on-failure
RestartSec=5s
ExecStart=/usr/local/bin/prometheus \
--config.file=/etc/prometheus/prometheus.yml \
--storage.tsdb.path=/data \
--web.console.templates=/etc/prometheus/consoles \
--web.console.libraries=/etc/prometheus/console_libraries \
--web.listen-address=0.0.0.0:9090 \
--web.enable-lifecycle

[Install]
WantedBy=multi-user.target
```

Let's go over a few of the most important options related to Systemd and Prometheus. **Restart** – Configures whether the service shall be restarted when the service process exits, is killed, or a timeout is reached.

**RestartSec** – Configures the time to sleep before restarting a service.

**User** and **Group** – Are Linux user and a group to start a Prometheus process.

**-config.file=/etc/prometheus/prometheus.yml** – Path to the main Prometheus configuration file.

**-storage.tsdb.path=/data** – Location to store Prometheus data.

**-web.listen-address=0.0.0.0:9090** – Configure to listen on all network interfaces. In some situations, you may have a proxy such as nginx to redirect requests to Prometheus. In that case, you would configure Prometheus to listen only on **localhost**.

**-web.enable-lifecycle** – Allows to manage Prometheus, for example, to reload configuration without restarting the service.

To automatically start the Prometheus after reboot, run enable.

`sudo systemctl enable prometheus`



```
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ sudo systemctl enable prometheus
Created symlink /etc/systemd/system/multi-user.target.wants/prometheus.service → /etc/systemd/system/prometheus.service.
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$
```

Then just start the Prometheus.

`sudo systemctl start prometheus`



```
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ sudo systemctl start prometheus
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$
```

To check the status of Prometheus run the following command:

`sudo systemctl status prometheus`



```
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ sudo systemctl status prometheus
● prometheus.service - Prometheus
   Loaded: loaded (/etc/systemd/system/prometheus.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2023-10-06 09:00:39 UTC; 13s ago
     Main PID: 1941 (prometheus)
       Tasks: 6 (limit: 1141)
      Memory: 16.2M
        CPU: 64ms
      CGroup: /system.slice/prometheus.service
              └─1941 /usr/local/bin/prometheus --config.file=/etc/prometheus/prometheus.yml --storage.tsdb.path=/data --web.console.templates=/etc/prometheus/consoles --web.console.lit

Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.798Z caller=tls_config.go:274 level=info components=web msg="Listening on " address=[::]:9090
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.798Z caller=tls_config.go:277 level=info components=web msg="TLS is disabled." http2=false address=[::]:9090
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.798Z caller=head.go:760 level=info component=tscdb msg="WAL segment loaded" segment=0 maxSegment=0
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.798Z caller=head.go:767 level=info component=tscdb msg="WAL replay completed" checkpoint_replay_duration=42.055s
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.881Z caller=main.go:1045 level=info fs_type=EXT4 SUPER_MAGIC
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.881Z caller=main.go:1048 level=info msg="TSDB started"
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.881Z caller=main.go:1229 level=info msg="Loading configuration file" filename=/etc/prometheus/prometheus.yml
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.885Z caller=main.go:1266 level=info msg="Completed loading of configuration file" filename=/etc/prometheus/pr...
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.885Z caller=main.go:1009 level=info msg="Server is ready to receive web requests."
Oct 06 09:00:39 ip-172-31-38-156 prometheus[1941]: ts=2023-10-06T09:00:39.885Z caller=manager.go:1089 level=info component="rule manager" msg="Starting rule manager...."
[lines 1-20/28 (END)]
```

Suppose you encounter any issues with Prometheus or are unable to start it. The easiest way to find the problem is to use the journalctl command and search for errors.



```
journalctl -u prometheus -f --no-pager
```



Now we can try to access it via the browser. I'm going to be using the IP address of the Ubuntu server. You need to append port 9090 to the IP.



```
<public-ip:9090>
```



The screenshot shows the Prometheus web interface at <http://43.205.117.84:9090>. The top navigation bar includes links for Gmail, LinkedIn, Docker — A Beginner's Guide, Cloud Quest, 100+ Essential Commands, #90DaysOfDevOps, Sharma's Blog!, DevOps Project 04, gsbarure/netflix-ci-cd, and Gajanan. The main page has a dark theme with a header for 'Prometheus' and tabs for 'Alerts', 'Graph', 'Status', and 'Help'. Below the header are checkboxes for 'Use local time', 'Enable query history', 'Enable autocomplete' (which is checked), 'Enable highlighting' (which is checked), and 'Enable linter'. A search bar contains the placeholder 'Expression (press Shift+Enter for newlines)'. To the right of the search bar are 'Table' and 'Graph' buttons, and an 'Execute' button. A panel below the search bar displays the message 'No data queried yet'. At the bottom left is a blue 'Add Panel' button, and at the bottom right is a 'Remove Panel' link.

If you go to targets, you should see only one – Prometheus target. It scrapes itself every 15 seconds by default.

## Install Node Exporter on Ubuntu 22.04

Next, we're going to set up and configure Node Exporter to collect Linux system metrics like CPU load and disk I/O. Node Exporter will expose these as Prometheus-style metrics. Since the installation process is very similar, I'm not going to cover as deep as Prometheus.

First, let's create a system user for Node Exporter by running the following command:

```
sudo useradd \
--system \
--no-create-home \
--shell /bin/false node_exporter
```

```
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ sudo useradd \
--system \
--no-create-home \
--shell /bin/false node_exporter
ubuntu@ip-172-31-38-156:~$
```

You can [download Node Exporter](#) from the same page.

Use the wget command to download the binary.

```
wget https://github.com/prometheus/node_exporter/releases/download/v1.6.1/node_exporter-1.6.1.linux-amd64.tar.gz
```

```
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ wget https://github.com/prometheus/node_exporter/releases/download/v1.6.1/node_exporter-1.6.1.linux-amd64.tar.gz
--2023-10-06 09:03:19--  https://github.com/prometheus/node_exporter/releases/download/v1.6.1/node_exporter-1.6.1.linux-amd64.tar.gz
Resolving github.com (github.com)... 20.207.73.82
Connecting to github.com (github.com)|20.207.73.82|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/9524057/5509b569-5c34-471e-8598-c05c0733bb7f?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWNYAX4C5VH5JN%2F20231006%2Fus-east-1%2Faws4_request&X-Amz-Date=20231006T090319Z&X-Amz-Expires=300&X-Amz-Signature=e01c47ad7d3d29de369b2c68bc38d0f043ef28c4a9ba/346b781b5e5e3403&X-Amz-SignedHeaders=host&actor_id=0&key_id=9524057&response-content-disposition=attachment%3Bfilename%3Dnode_exporter-1.6.1.linux-amd64.tar.gz&response-content-type=application%2Foctet-stream [following]
--2023-10-06 09:03:19--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/9524057/5509b569-5c34-471e-8598-c05c0733bb7f?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWNYAX4C5VH5JN%2F20231006%2Fus-east-1%2Faws4_request&X-Amz-Date=20231006T090319Z&X-Amz-Expires=300&X-Amz-Signature=e01c47ad7d3d29de369b2c68bc38d0f043ef28c4a90a73460781b5e3e3403&X-Amz-SignedHeaders=host&actor_id=0&key_id=9524057&response-content-disposition=attachment%3Bfilename%3Dnode_exporter-1.6.1.linux-amd64.tar.gz&response-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.111.133, 185.199.108.133, 185.199.109.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 10368103 (9.9M) [application/octet-stream]
Saving to: 'node_exporter-1.6.1.linux-amd64.tar.gz'

node_exporter-1.6.1.linux-amd64.tar.gz      100%[=====]  9.89M  --.-KB/s   in 0.07s

2023-10-06 09:03:20 (135 MB/s) - 'node_exporter-1.6.1.linux-amd64.tar.gz' saved [10368103/10368103]
ubuntu@ip-172-31-38-156:~$
```

Extract the node exporter from the archive.

```
tar -xvf node_exporter-1.6.1.linux-amd64.tar.gz
```



```
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ ls  
node_exporter-1.6.1.linux-amd64.tar.gz  prometheus-2.47.1.linux-amd64  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ tar -xvf node_exporter-1.6.1.linux-amd64.tar.gz  
node_exporter-1.6.1.linux-amd64/  
node_exporter-1.6.1.linux-amd64/NOTICE  
node_exporter-1.6.1.linux-amd64/node_exporter  
node_exporter-1.6.1.linux-amd64/LICENSE  
ubuntu@ip-172-31-38-156:~$ █
```

Move binary to the /usr/local/bin.



```
sudo mv \  
  node_exporter-1.6.1.linux-amd64/node_exporter \  
  /usr/local/bin/
```



```
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ sudo mv \  
  node_exporter-1.6.1.linux-amd64/node_exporter \  
  /usr/local/bin/  
ubuntu@ip-172-31-38-156:~$ █
```

Clean up, and delete node\_exporter archive and a folder.



```
rm -rf node_exporter*
```



```
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ ls  
node_exporter-1.6.1.linux-amd64 node_exporter-1.6.1.linux-amd64.tar.gz prometheus-2.47.1.linux-amd64  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ rm -rf node_exporter*  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ ls  
prometheus-2.47.1.linux-amd64  
ubuntu@ip-172-31-38-156:~$
```

Verify that you can run the binary.



```
node_exporter --version
```



```
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ node_exporter --version  
node_exporter, version 1.6.1 (branch: HEAD, revision: 4a1b77600c1873a8233f3ffb55afcedbb63b8d84)  
  build user:      root@586879db11e5  
  build date:    20230717-12:10:52  
  go version:     go1.20.6  
  platform:       linux/amd64  
  tags:           netgo osusergo static_build  
ubuntu@ip-172-31-38-156:~$ █
```

Node Exporter has a lot of plugins that we can enable. If you run Node Exporter help you will get all the options.



```
node_exporter --help
```



-collector.logind We're going to enable the login controller, just for the demo.

Next, create a similar systemd unit file.



```
sudo vim /etc/systemd/system/node_exporter.service
```



```
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ sudo vim /etc/systemd/system/node_exporter.service
```

## node\_exporter.service



```
[Unit]  
Description=Node Exporter  
Wants=network-online.target  
After=network-online.target  
StartLimitIntervalSec=500  
StartLimitBurst=5  
[Service]  
User=node_exporter  
Group=node_exporter  
Type=simple  
Restart=on-failure  
RestartSec=5s  
ExecStart=/usr/local/bin/node_exporter \  
--collector.logind  
[Install]  
WantedBy=multi-user.target
```



```
[Unit]  
Description=Node Exporter  
Wants=network-online.target  
After=network-online.target  
StartLimitIntervalSec=500  
StartLimitBurst=5  
[Service]  
User=node_exporter  
Group=node_exporter  
Type=simple  
Restart=on-failure  
RestartSec=5s  
ExecStart=/usr/local/bin/node_exporter \  
--collector.logind  
[Install]  
WantedBy=multi-user.target
```

Replace Prometheus user and group to node\_exporter, and update the ExecStart command.

To automatically start the Node Exporter after reboot, enable the service.



`sudo systemctl enable node_exporter`



Then start the Node Exporter.



`sudo systemctl start node_exporter`



```
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ sudo systemctl enable node_exporter
Created symlink /etc/systemd/system/multi-user.target.wants/node_exporter.service → /etc/systemd/system/node_exporter.service.
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ sudo systemctl start node_exporter
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$
```

Check the status of Node Exporter with the following command:



`sudo systemctl status node_exporter`



```
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ sudo systemctl status node_exporter
● node_exporter.service - Node Exporter
   Loaded: loaded (/etc/systemd/system/node_exporter.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2023-10-06 09:07:39 UTC; 8s ago
     Main PID: 2030 (node exporter)
        Tasks: 3 (limit: 1141)
       Memory: 2.0M
          CPU: 9ms
        CGroup: /system.slice/node_exporter.service
                  └─2030 /usr/local/bin/node_exporter --collector.logind

Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=thermal_zone
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=time
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=timex
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=udp_queues
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=uname
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=vmstat
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=xfs
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.415Z caller=tls_config.go:274 level=info msg="listening on" address=[::]:9100
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.415Z caller=tls_config.go:277 level=info msg="TLS is disabled." http2=false address=[::]:9100
ubuntu@ip-172-31-38-156:~$
```

If you have any issues, check logs with journalctl



```
journalctl -u node_exporter -f --no-pager
```



At this point, we have only a single target in our Prometheus. There are many different service discovery mechanisms built into Prometheus. For example, Prometheus can dynamically discover targets in AWS, GCP, and other clouds based on the labels. In the following tutorials, I'll give you a few examples of deploying Prometheus in a cloud-specific environment. For this tutorial, let's keep it simple and keep adding static targets. Also, I have a lesson on how to deploy and manage Prometheus in the Kubernetes cluster.

To create a static target, you need to add job\_name with static\_configs.



```
sudo vim /etc/prometheus/prometheus.yml
```



```
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ sudo vim /etc/prometheus/prometheus.yml  
ubuntu@ip-172-31-38-156:~$ █
```

## prometheus.yml



```
- job_name: node_export  
  static_configs:  
    - targets: ["localhost:9100"]
```



```
# my global config
global:
  scrape_interval: 15s # Set the scrape interval to every 15 seconds. Default is every 1 minute.
  evaluation_interval: 15s # Evaluate rules every 15 seconds. The default is every 1 minute.
  # scrape_timeout is set to the global default (10s).

# Alertmanager configuration
alerting:
  alertmanagers:
    - static_configs:
      - targets:
          # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global 'evaluation_interval'.
rule_files:
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scraped from this config.
  - job_name: "prometheus"

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ["localhost:9090"]

  - job_name: node_export
    static_configs:
      - targets: ["localhost:9100"]
```

~

By default, Node Exporter will be exposed on port 9100.

Since we enabled lifecycle management via API calls, we can reload the Prometheus config without restarting the service and causing downtime.

Before, restarting check if the config is valid.



`promtool check config /etc/prometheus/prometheus.yml`



```
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ promtool check config /etc/prometheus/prometheus.yml
Checking /etc/prometheus/prometheus.yml
SUCCESS: /etc/prometheus/prometheus.yml is valid prometheus config file syntax

ubuntu@ip-172-31-38-156:~$
```

Then, you can use a POST request to reload the config.



```
curl -X POST http://localhost:9090/-/reload
```



```
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ curl -X POST http://localhost:9090/-/reload  
ubuntu@ip-172-31-38-156:~$ █
```

Check the targets section



<http://<ip>:9090/targets>



Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:9100/metrics	UP	instance="localhost:9100" job="node_export"	1.953s ago	25.413ms	

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:9090/metrics	UP	instance="localhost:9090" job="prometheus"	2.748s ago	6.655ms	

## Install Grafana on Ubuntu 22.04

To visualize metrics we can use Grafana. There are many different data sources that Grafana supports, one of them is Prometheus.

First, let's make sure that all the dependencies are installed.



```
sudo apt-get install -y apt-transport-https software-properties-common
```



```
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ sudo apt-get install -y apt-transport-https software-properties-common
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  python3-software-properties
The following NEW packages will be installed:
  apt-transport-https
The following packages will be upgraded:
  python3-software-properties software-properties-common
2 upgraded, 1 newly installed, 0 to remove and 127 not upgraded.
Need to get 44.4 kB of archives.
After this operation, 169 kB of additional disk space will be used.
Get:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 apt-transport-https all 2.4.10 [1510 B]
Get:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 software-properties-common all 0.99.22.7 [14.1 kB]
Get:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 python3-software-properties all 0.99.22.7 [28.8 kB]
Fetched 44.4 kB in 0s (2002 kB/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 64295 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_2.4.10_all.deb ...
Unpacking apt-transport-https (2.4.10) ...
Preparing to unpack .../software-properties-common_0.99.22.7_all.deb ...
Unpacking software-properties-common (0.99.22.7) over (0.99.22.6) ...
Preparing to unpack .../python3-software-properties_0.99.22.7_all.deb ...
Unpacking python3-software-properties (0.99.22.7) over (0.99.22.6) ...
Setting up apt-transport-https (2.4.10) ...
Setting up python3-software-properties (0.99.22.7) ...
Setting up software-properties-common (0.99.22.7) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for dbus (1.12.20-2ubuntu4.1) ...
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.
```

Next, add the GPG key.



```
wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -
```



```
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$
```

Add this repository for stable releases.



```
echo "deb https://packages.grafana.com/oss/deb stable main" | sudo tee -a /etc/apt/sources.list.d/grafana.list
```



```
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ echo "deb https://packages.grafana.com/oss/deb stable main" | sudo tee -a /etc/apt/sources.list.d/grafana.list
deb https://packages.grafana.com/oss/deb stable main
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ 
```

After you add the repository, update and install Garafana.

```
sudo apt-get update
```



```
sudo apt-get -y install grafana
```



```
ubuntu@ip-172-31-38-156:~$ sudo apt-get -y install grafana
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  fontconfig-config fonts-dejavu-core libfontconfig1 musl
The following NEW packages will be installed:
  fontconfig-config fonts-dejavu-core grafana libfontconfig1 musl
0 upgraded, 5 newly installed, 0 to remove and 127 not upgraded.
Need to get 104 MB of archives.
After this operation, 379 MB of additional disk space will be used.
Get:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 fontconfig-config all 2.37-2build1 [1041 kB]
Get:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 libfontconfig1 amd64 2.13.1-4.2ubuntu5 [29.1 kB]
Get:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 libfontconfig1 amd64 2.13.1-4.2ubuntu5 [131 kB]
Get:4 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 musl amd64 1.2.2-4 [407 kB]
Get:5 https://packages.grafana.com/oss/deb stable/main amd64 grafana amd64 10.1.4 [102 MB]
17% [5 grafana 0 B/102 MB 0%]
```

To automatically start the Grafana after reboot, enable the service.



```
sudo systemctl enable grafana-server
```



Then start the Grafana.



```
sudo systemctl start grafana-server
```



```
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ sudo systemctl enable grafana-server
Synchronizing state of grafana-server.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable grafana-server
Created symlink /etc/systemd/system/multi-user.target.wants/grafana-server.service → /lib/systemd/system/grafana-server.service.
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ sudo systemctl start grafana-server
ubuntu@ip-172-31-38-156:~$
```

To check the status of Grafana, run the following command:

```
sudo systemctl status grafana-server
```



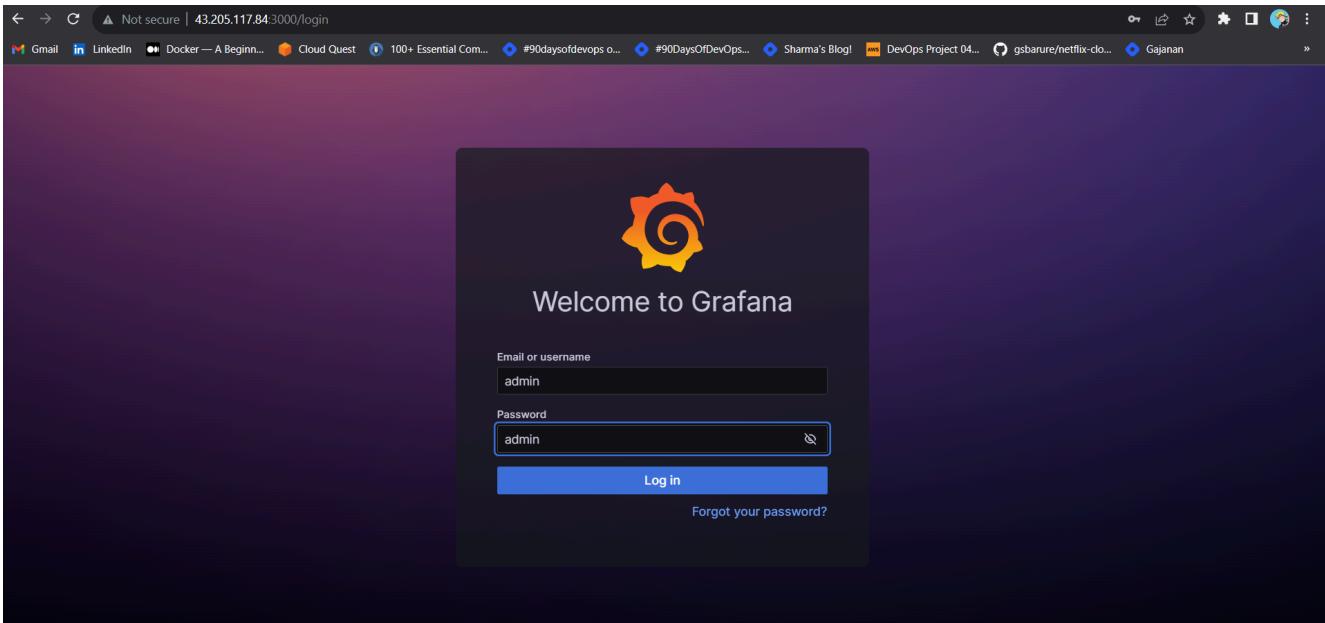
```
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ sudo systemctl status grafana-server
● grafana-server.service - Grafana instance
  Loaded: loaded (/lib/systemd/system/grafana-server.service; enabled; vendor preset: enabled)
  Active: active (running) since Fri 2023-10-06 09:14:01 UTC; 7s ago
    Docs: http://docs.grafana.org
 Main PID: 3263 (grafana)
   Tasks: 5 (limit: 1141)
     Memory: 175.9M
        CPU: 4.330s
      CGroup: /system.slice/grafana-server.service
             └─3263 /usr/share/grafana/bin/grafana server --config=/etc/grafana/grafana.ini --pidfile=/run/grafana/grafana-server.pid --packaging=deb cfg=default.paths.logs=/var/log/grafana

Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=migrator t=2023-10-06T09:14:06.750963027Z level=info msg="Executing migration" id="Add unique index for folder.title and folder.description"
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=migrator t=2023-10-06T09:14:06.757199161Z level=info msg="migrations completed" performed=497 skipped=0 duration=4.148747644s
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=sqldb t=2023-10-06T09:14:06.774714527Z level=info msg="Created default admin" user=admin
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=sqldb t=2023-10-06T09:14:06.775163061Z level=info msg="Created default organization"
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=secrets t=2023-10-06T09:14:06.782069114Z level=info msg="Envelope encryption state" enabled=true currentprovider=secretKey.v1
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=plugin.loader t=2023-10-06T09:14:06.883193807Z level=warn msg="Plugin missing module.js" pluginId=input warning="Missing module.js"
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=plugin.loader t=2023-10-06T09:14:06.883523142Z level=info msg="Plugin registered" pluginId=input
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=local.finder t=2023-10-06T09:14:06.883678317Z level=warn msg="Skipping finding plugins as directory does not exist" path=/var/lib/grafana/plugins
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=query_data t=2023-10-06T09:14:06.890278741Z level=info msg="Query Service initialization"
Oct 06 09:14:06 ip-172-31-38-156 grafana[3263]: logger=live.push.http t=2023-10-06T09:14:06.897367677Z level=info msg="Live Push Gateway initialization"
[lines 1-21/21 (END)]
```

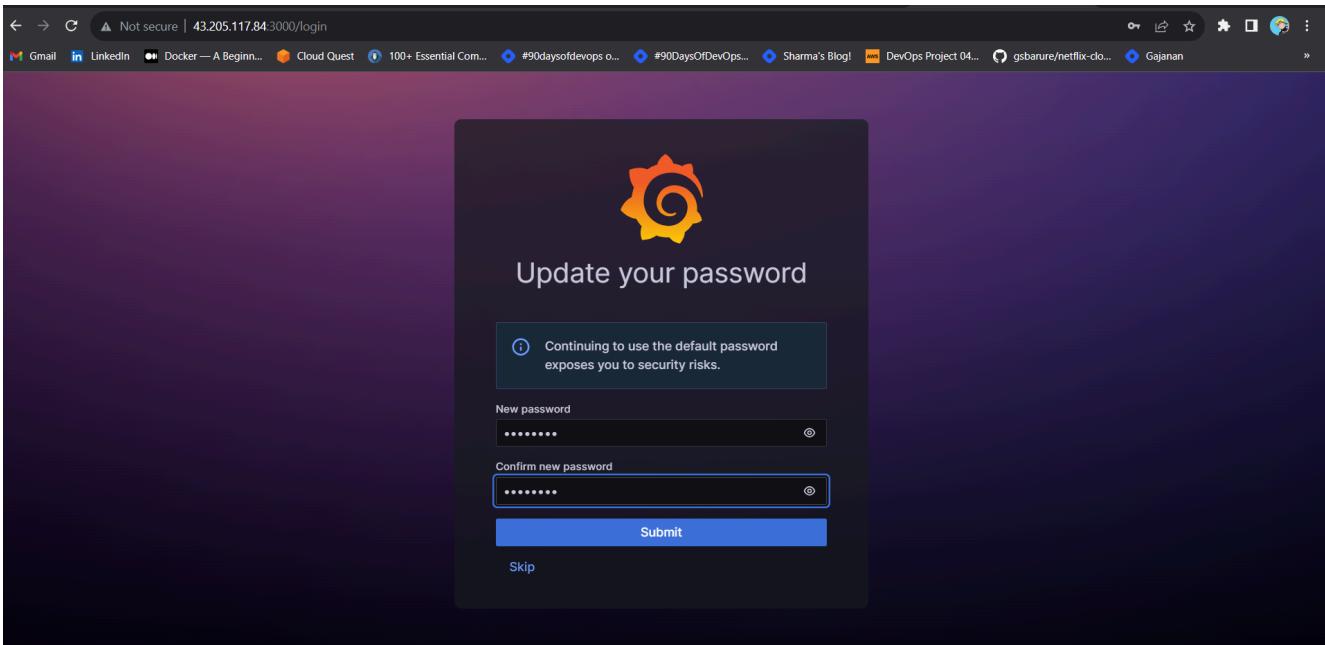
Go to <http://<ip>:3000> and log in to the Grafana using default credentials. The username is admin, and the password is admin as well.

```
username admin
password admin
```





When you log in for the first time, you get the option to change the password.



To visualize metrics, you need to add a data source first.

Welcome to Grafana

Need help? Documentation Tutorials Community Public Slack Untitled

**Basic**

The steps below will guide you to quickly finish setting up your Grafana installation.

**TUTORIAL**

DATA SOURCE AND DASHBOARDS

Grafana fundamentals

Set up and understand Grafana if you have no prior experience. This tutorial guides you through the entire process and covers the "Data source" and "Dashboards" steps to the right.

**DATA SOURCES**

Add your first data source

Learn how in the docs

**DASHBOARDS**

Create your first dashboard

Learn how in the docs

Remove this panel

Click Add data source and select Prometheus.

Not secure | 43.205.117.84:3000/connections/datasources/new

Home > Connections > Data sources > Add data source

**Add data source**

Choose a data source type

Q Filter by name or type

Time series databases

- Prometheus** Open source time series database & alerting Core Learn more
- Graphite Open source time series database Core
- InfluxDB Open source time series database Core

For the URL, enter localhost:9090 and click Save and test. You can see Data source is working.

<public-ip:9090>



Prometheus

Type: Prometheus

**Settings** **Dashboards**

Configure your Prometheus data source below  
Or skip the effort and get Prometheus (and Loki) as fully-managed, scalable, and hosted data sources from Grafana Labs with the [free-forever Grafana Cloud plan](#).

**Alerting supported**

Name: Prometheus Default:

**HTTP**

Prometheus server URL:  http://43.205.117.84:9090

Allowed cookies:  Add

Timeout:

Click on Save and Test.

Not secure | 43.205.117.84:3000/connections/datasources/edit/e03d1fbe-4435-485c-a497-1099d1684239

Home > Connections > Data sources > Prometheus

**Connections**

Add new connection **Data sources**

Disable metrics lookup

**Performance**

Prometheus type: Choose Cache level: Low

Incremental querying (beta):  Disable recording rules (beta):

**Other**

Custom query parameters: Example: max\_source\_resolution=5m&timeout  
HTTP method: POST

**Exemplars**

+ Add

**Actions**

Delete Save & test

Let's add Dashboard for a better view

The screenshot shows the Grafana interface. In the top right, there's a context menu with options like 'New dashboard', 'Import dashboard', and 'Create alert rule'. The 'Import dashboard' option is highlighted with a red box. On the left sidebar, the 'Dashboards' button is also highlighted with a red box.

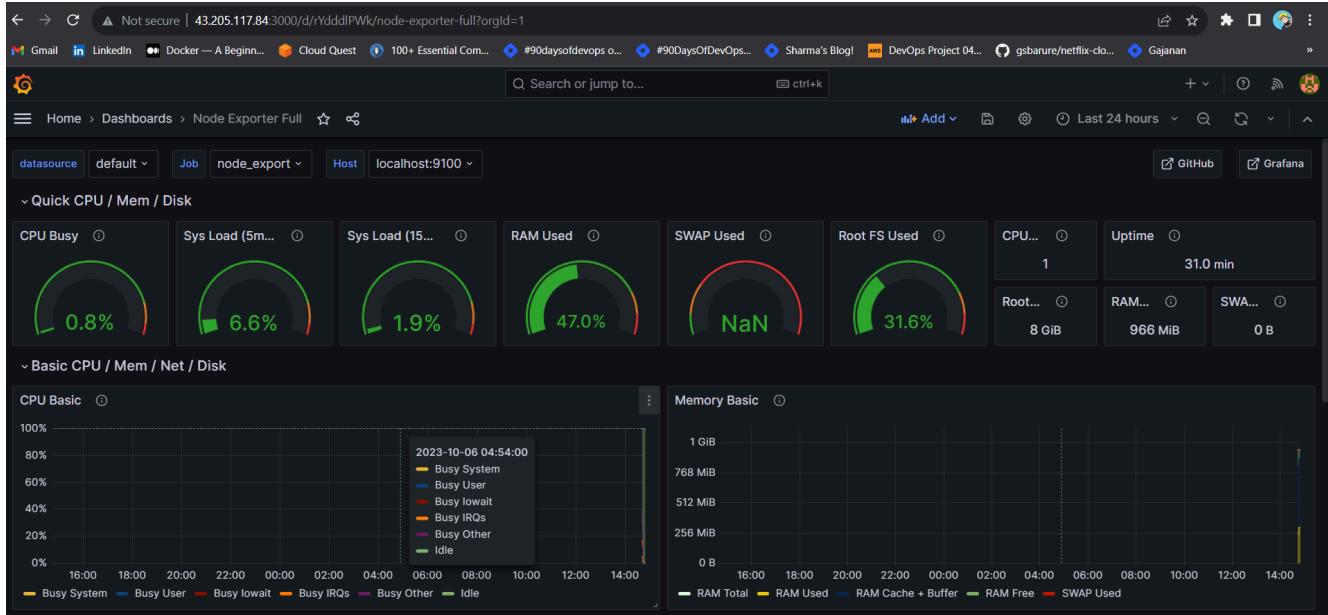
Click on Import Dashboard paste this code 1860 and click on load

This screenshot shows the 'Import dashboard' page. It has a file upload section for 'Dashboard JSON file' and a text input field labeled 'Import via grafana.com' containing the value '1860'. The 'Load' button is highlighted with a red box.

Select the Datasource and click on Import

This screenshot shows the 'Importing dashboard from Grafana.com' page. It displays details about the imported dashboard, including its publisher and update time. Below that is an 'Options' section with fields for 'Name' and 'Folder'. The 'Prometheus' folder is selected and highlighted with a red box. At the bottom, there's a 'Select a Prometheus data source' dropdown and an 'Import' button, which is also highlighted with a red box.

You will see this output



## Step 5 – Install the Prometheus Plugin and Integrate it with the Prometheus server

Let's Monitor JENKINS SYSTEM

Need Jenkins up and running machine

Goto Manage Jenkins → Plugins → Available Plugins

Search for Prometheus and install it

The screenshot shows the Jenkins Manage Jenkins > Plugins page. The "Available plugins" tab is active. A search bar contains the text "prome". A plugin card for "Prometheus metrics 2.3.3" is highlighted with a red box. The card includes the following details:

- Install button
- Name: Prometheus metrics 2.3.3
- Released: 17 days ago
- Description: Jenkins Prometheus Plugin expose an endpoint (default /prometheus) with metrics where a Prometheus Server can scrape.

Once that is done you will Prometheus is set to /Prometheus path in system configurations

Dashboard > Manage Jenkins > System >

### Prometheus

Path ?  
prometheus

Default Namespace ?  
default

Enable authentication for prometheus end-point ?

Collecting metrics period in seconds ?  
120

Count duration of successful builds ?  
 Count duration of unstable builds ?  
 Count duration of failed builds ?  
 Count duration of not-built builds ?  
 Count duration of aborted builds ?  
 Fetch the last results of builds ?

**Save** **Apply**

Nothing to change click on apply and save

To create a static target, you need to add job\_name with static\_configs. go to Prometheus server



sudo vim /etc/prometheus/prometheus.yml



```
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ sudo vim /etc/prometheus/prometheus.yml
```

Paste below code



```
- job_name: 'jenkins'  
  metrics_path: '/prometheus'  
  static_configs:  
    - targets: ['<jenkins-ip>:8080']
```



```
# my global config
global:
  scrape_interval: 15s # Set the scrape interval to every 15 seconds. Default is every 1 minute.
  evaluation_interval: 15s # Evaluate rules every 15 seconds. The default is every 1 minute.
  # scrape_timeout is set to the global default (10s).

# Alertmanager configuration
alerting:
  alertmanagers:
    - static_configs:
      - targets:
        # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global 'evaluation_interval'.
rule_files:
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scraped from this config.
  - job_name: "prometheus"

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ["localhost:9090"]

  - job_name: node_export
    static_configs:
      - targets: ["localhost:9100"]

  - job_name: jenkins
    metrics_path: '/prometheus'
    static_configs:
      - targets: ['3.111.170.92:8080']
```

Before, restarting check if the config is valid.



```
promtool check config /etc/prometheus/prometheus.yml
```



Then, you can use a POST request to reload the config.



```
curl -X POST http://localhost:9090/-/reload
```



```
ubuntu@ip-172-31-38-156:~$ promtool check config /etc/prometheus/prometheus.yml
Checking /etc/prometheus/prometheus.yml
SUCCESS: /etc/prometheus/prometheus.yml is valid prometheus config file syntax

ubuntu@ip-172-31-38-156:~$ curl -X POST http://localhost:9090/-/reload
ubuntu@ip-172-31-38-156:~$
```

Check the targets section



<http://<ip>:9090/targets>



You will see Jenkins is added to it

The screenshot shows the Prometheus Targets page at <http://43.205.117.84:9090/targets?search=#pool-jenkins>. The page displays three sections of targets:

- Targets**: Shows one target named "jenkins" (1/1 up) from the "jenkins" pool. The target is **UP**, instance is "3.111.170.92:8080", job is "jenkins", last scraped 15.932s ago, and scrape duration is 50.433ms.
- node\_export (1/1 up)**: Shows one target named "node\_export" (1/1 up) from the "node\_export" pool. The target is **UP**, instance is "localhost:9100", job is "node\_export", last scraped 15.610s ago, and scrape duration is 17.760ms.
- prometheus (1/1 up)**: Shows one target named "prometheus" (1/1 up) from the "prometheus" pool. The target is **UP**, instance is "localhost:9090", job is "prometheus", last scraped 16.405s ago, and scrape duration is 6.085ms.

Let's add Dashboard for a better view in Grafana

Click On Dashboard -> + symbol -> Import Dashboard

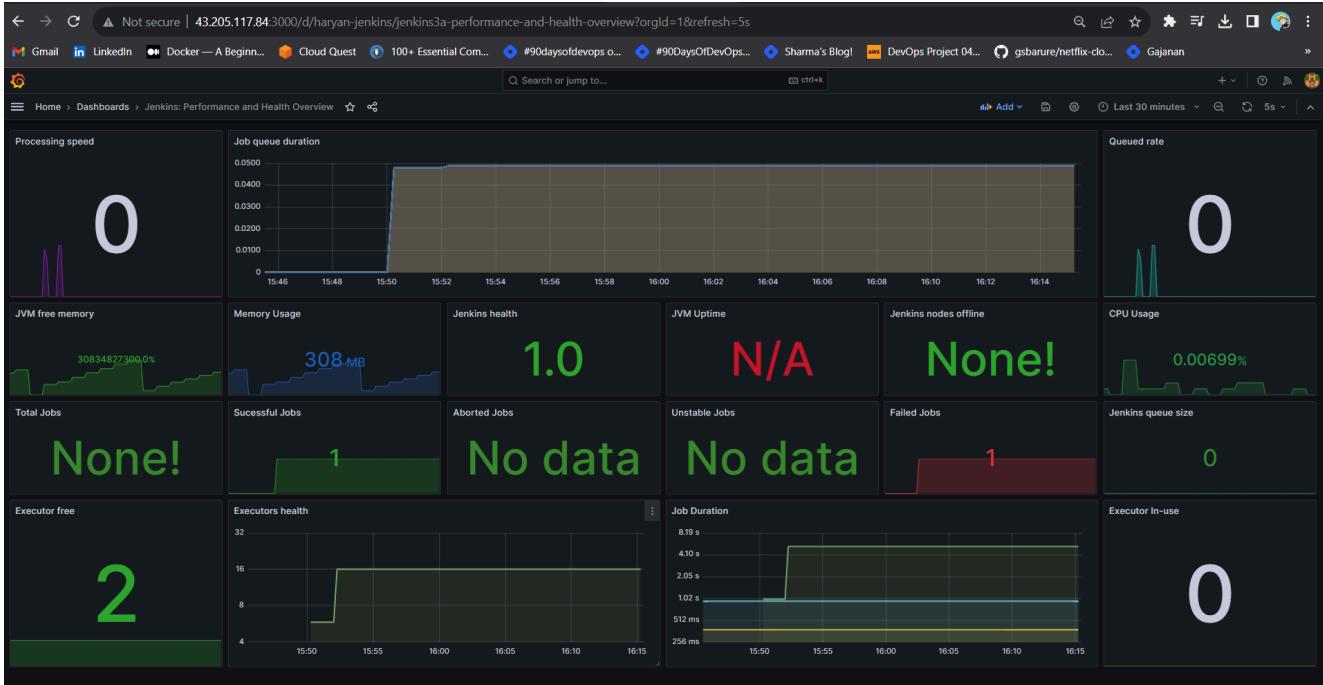
Use Id 9964 and click on load

The screenshot shows the 'Import dashboard' page in Grafana. On the left sidebar, there are links for Dashboards, Playlists, Snapshots, and Library panels. The main area has a title 'Import dashboard' and a sub-instruction 'Import dashboard from file or Grafana.com'. Below this is a dashed box labeled 'Upload dashboard JSON file' with a placeholder 'Drag and drop here or click to browse'. Underneath is a section for 'Import via grafana.com' with an input field containing '9964' and a red box around it, followed by a 'Load' button.

Select the data source and click on Import

This screenshot shows the 'Importing dashboard from Grafana.com' configuration screen. It includes fields for 'Published by' (haryan), 'Updated on' (2023-08-24 15:04:53), and 'Options' like 'Name' (Jenkins: Performance and Health Overview) and 'Folder' (General). In the 'Unique identifier (UID)' section, there's a 'Change uid' button and a dropdown menu where 'Prometheus' is selected, highlighted with a red box. At the bottom are 'Import' and 'Cancel' buttons, with the 'Import' button also highlighted with a red box.

Now you will see the Detailed overview of Jenkins



## Step 6 – Email Integration With Jenkins and Plugin Setup

Install Email Extension Plugin in Jenkins

The Jenkins Plugins page shows the following interface:

- Left sidebar:** Updates, Available plugins (selected), Installed plugins, Advanced settings, Download progress.
- Search bar:** Email Ex
- Plugin list:**
  - Email Extension Template 1.5** (Selected):  
Build Notifiers emailext  
This plugin allows administrators to create global templates for the Extended Email Publisher.  
This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.
  - Released 11 mo ago

Go to your Gmail and click on your profile

Then click on Manage Your Google Account -> click on the security tab on the left side panel you will get this page(provide mail password).



- Home
- Personal info
- Data and privacy
- Security
- People and sharing
- Payments and subscriptions
- About

[See details](#)

#### Recent security activity

App password removed	1 Sept · Telangana, India	
App password created	31 Aug · Telangana, India	
App password removed	31 Aug · Telangana, India	
<a href="#">Review security activity (4)</a>		

#### How you sign in to Google

Make sure that you can always access your Google Account by keeping this information up to date

	2-Step Verification		On since 8 Oct 2022	
--	---------------------	--	---------------------	--

2-step verification should be enabled.

Search for the app in the search bar you will get app passwords like the below image

Google Account

Search: app

3 RESULTS

- Your connections to third-party apps and services  
Security
- App passwords**  
Security
- Web & App Activity  
Data and privacy

See details

Google Account

[← App passwords](#)

App passwords let you sign in to your Google Account from apps on devices that don't support 2-Step Verification. You'll only need to enter it once so you don't need to remember it. [Learn more](#)

You don't have any app passwords.

Select the app and device for which you want to generate the app password.

Jenkins

X

**GENERATE**

Click on other and provide your name and click on Generate and copy the password

## ← App passwords

### Generated app password

Your app password for your device

bkec mhur oddp hppw

#### How to use it

Go to the settings for your Google Account in the application or device you are trying to set up. Replace your password with the 16-character password shown above. Just like your normal password, this app password grants complete access to your Google Account. You won't need to remember it, so don't write it down or share it with anyone.

#### Email

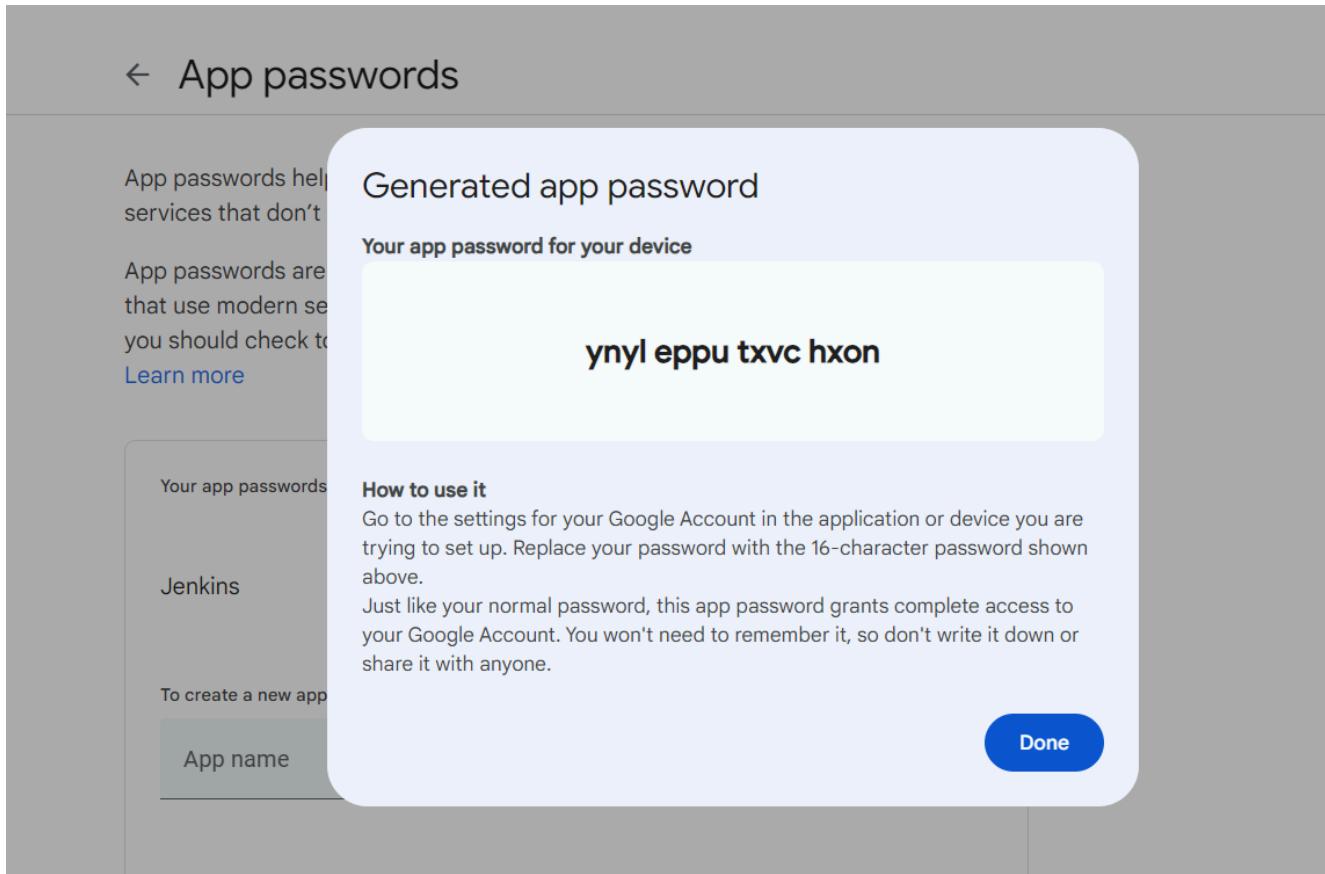
securesally@gmail.com

#### Password

••••••••••••

[DONE](#)

In the new update, you will get a password like this



Once the plugin is installed in Jenkins, click on manage Jenkins -> configure system there under the E-mail Notification section configure the details as shown in the below image



Dashboard > Manage Jenkins > System >

The screenshot shows the Jenkins System configuration page under the Global Mailer section. It includes fields for Username (postbox.aj99@gmail.com), Password (redacted), SSL/TLS settings (Use SSL checked, Use TLS unchecked), SMTP Port (465), Reply-To Address (empty), Charset (UTF-8), and a Test configuration checkbox (unchecked). Below the form is a Dependency-Track section with Save and Apply buttons.

Click on Apply and save.

Click on Manage Jenkins-> credentials and add your mail username and generated password

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

The screenshot shows the Jenkins Manage Jenkins > Credentials > Global credentials (unrestricted) page. It displays a form for creating a new credential. The fields are: Username (postbox.aj99@gmail.com), Password (redacted), Scope (Global), ID (mail), and Description (mail). The Create button is at the bottom.

This is to just verify the mail configuration

Now under the Extended E-mail Notification section configure the details as shown in the below images

Dashboard > Manage Jenkins > System > Advanced Email Configuration

**SMTP server**: smtp.gmail.com

**SMTP Port**: 465

**Credentials**: postbox.aj99@gmail.com/\*\*\*\*\* (mail)

**Add**

Use SSL

Use TLS

Use OAuth 2.0

Dashboard > Manage Jenkins > System > Advanced Email Properties

**Default user e-mail suffix**: ?

**Advanced**  Edited

**Default Content Type**: ?

HTML (text/html)

**List ID**: ?

Add 'Precedence: bulk' E-mail Header

Dashboard > Manage Jenkins > System > Additional groovy classpath

**Add**

Enable Debug Mode

Require Administrator for Template Testing

Enable watching for jobs

Allow sending to unregistered users

**Default Triggers** ^

**Default Triggers** ?

- Aborted
- Always
- Before Build
- Failure - 1st
- Failure - 2nd
- Failure - Any
- Failure - Still
- Failure - X
- Failure -> Unstable (Test Failures)
- Fixed
- Not Built

**Save** **Apply**

Click on Apply and save.



```
post {  
    always {  
        emailext attachLog: true,  
            subject: "'${currentBuild.result}'",  
            body: "Project: ${env.JOB_NAME}<br/>" +  
                "Build Number: ${env.BUILD_NUMBER}<br/>" +  
                "URL: ${env.BUILD_URL}<br/>",  
            to: 'postbox.aj99@gmail.com', #change Your mail  
            attachmentsPattern: 'trivyfs.txt,trivyimage.txt'  
    }  
}
```



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

## Step 7 – Install Plugins like JDK, Sonarqube Scanner, NodeJs, OWASP Dependency Check

### 7A – Install Plugin

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 Plugins page. On the left, there's a sidebar with links: Updates, Available plugins (which is selected), Installed plugins, Advanced settings, and Download progress. The main area is titled "Plugins" and has a search bar. It lists two plugins:

- Eclipse Temurin installer 1.5**: Provides an installer for the JDK tool that downloads the JDK from <https://adoptium.net>. Status: Released, 11 mo ago. A note says: "This plugin is up for adoption! We are looking for new maintainers. Visit our [Adopt a Plugin](#) initiative for more information."
- SonarQube Scanner 2.15**: External Site/Tool Integrations, Build Reports. Status: Released, 9 mo 19 days ago. Description: "This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality."

Below this, another section shows a single plugin:

- NodeJS 1.6.1**: npm. Status: Released, 1 mo 2 days ago. Description: "NodeJS Plugin executes [NodeJS](#) script as a build step."

## 7B – 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 configuration page under "JDK installations". There is a button to "Add JDK". One JDK installation is listed:

- Name**: jdk17
- Install automatically**: checked
- Install from adoptium.net**: checked
- Version**: jdk-17.0.8.1+1
- Add Installer**: button

## 7C – Create a Job

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

## Step 8 – 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

click on update Token

Create a token with a name and generate

**Tokens of Administrator**

**Generate Tokens**

Name	Expires in
Enter Token Name	30 days
<input type="button" value="Generate"/>	

**New token "Jenkins" has been created. Make sure you copy it now, you won't be able to see it again!**

**Copy** :squ\_21d162904c1c72cf8b39665f96480185c99dc2f9

Name	Type	Project	Last use	Created	Expiration
Jenkins	User		Never	September 8, 2023	October 8, 2023
					<b>Revoke</b>

**copy Token**

Goto Jenkins Dashboard → Manage Jenkins → Credentials → Add Secret Text. It should look like this

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

### New credentials

Kind: Secret text

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

Secret: POST THE TOKEN HERE

ID: Sonar-token

Description: Sonar-token

**Create**

You will see this page once you click on create

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

ID	Name	Kind	Description	
Sonar-token	sonar	Secret text	sonar	

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

**SonarQube servers**

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

**Environment variables** Enable injection of SonarQube server configuration as build environment variables

**SonarQube installations**

List of SonarQube installations

Name	<input type="text" value="sonar-server"/>	<span style="color: red;">X</span>
Server URL	Default is http://localhost:9000 <input type="text" value="http://13.232.17.191:9000"/>	
Server authentication token	SonarQube authentication token. Mandatory when anonymous access is disabled. <input type="text" value="Sonar-token"/> <button>Add</button>	
<input type="button" value="Save"/> <input type="button" value="Apply"/>		

Click on Apply and Save

**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.

**SonarQube Scanner installations**

[Add SonarQube Scanner](#)

**SonarQube Scanner****Name**

**Install automatically** ?

**Install from Maven Central****Version**

[Add Installer](#)

[Add SonarQube Scanner](#)

In the Sonarqube Dashboard add a quality gate also

## Administration-> Configuration->Webhooks

The screenshot shows the SonarQube administration interface. The top navigation bar has tabs for Gmail, YouTube, Amazon Web Servic..., Coaching on DevO..., LinkedIn, Docker — A Beginn..., Cloud Quest, 100+ Essential Com..., Advanced End-to-E..., LINUX - YouTube T..., and How to Install Jenki... The main navigation bar includes sonarqube, Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration (which is highlighted with a red box), and a search bar for projects. Below the main navigation, there's a sub-navigation for Administration with tabs for General Settings, Encryption, and Webhooks (also highlighted with a red box). A sidebar on the left lists General Settings, Encryption, and Webhooks. The main content area shows a table of users with one entry: Administrator admin, last connected < 1 hour ago, with groups sonar-administrators and sonar-users, and 1 token. A 'Create User' button is located in the top right corner of the user table.

Click on Create

The screenshot shows the SonarQube configuration page for Webhooks. The top navigation bar has tabs for sonarqube, Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration (highlighted with a blue bar), and a search bar for projects. Below the main navigation, there's a sub-navigation for Administration with tabs for Configuration, Security, Projects, System, and Marketplace. The main content area is titled 'Webhooks' and contains a brief description: 'Webhooks are used to notify external services when a project analysis is done. An HTTP POST request including a JSON payload is sent to each of the provided URLs. Learn more in the [Webhooks documentation](#)'. A large 'Create' button is prominently displayed on the right side of the page.

Add details



```
#in url section of quality gate
<http://jenkins-public-ip:8080>/sonarqube-webhook/
```



The screenshot shows the SonarQube administration interface under the 'Webhooks' section. A modal window titled 'Create Webhook' is open, prompting for a 'Name' (set to 'jenkins') and a 'URL' (set to 'http://43.204.36.242:8090/sonarqube-webhook/'). Below the URL field is a note about using an embedded database. A tooltip at the bottom left of the modal states: '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...'. On the right side of the modal, there are 'Create' and 'Cancel' buttons.

Let's go to our Pipeline and add the script in our Pipeline Script.

```
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/Aj7Ay/Netf'
            }
        }
        stage("Sonarqube Analysis"){
            steps{
                withSonarQubeEnv('sonar-server') {
                    sh ''' $SCANNER_HOME/bin/sonar-scanner -Dsonar.proje
                }
            }
        }
    }
}
```

```

        -Dsonar.projectKey=Netflix '''
    }
}

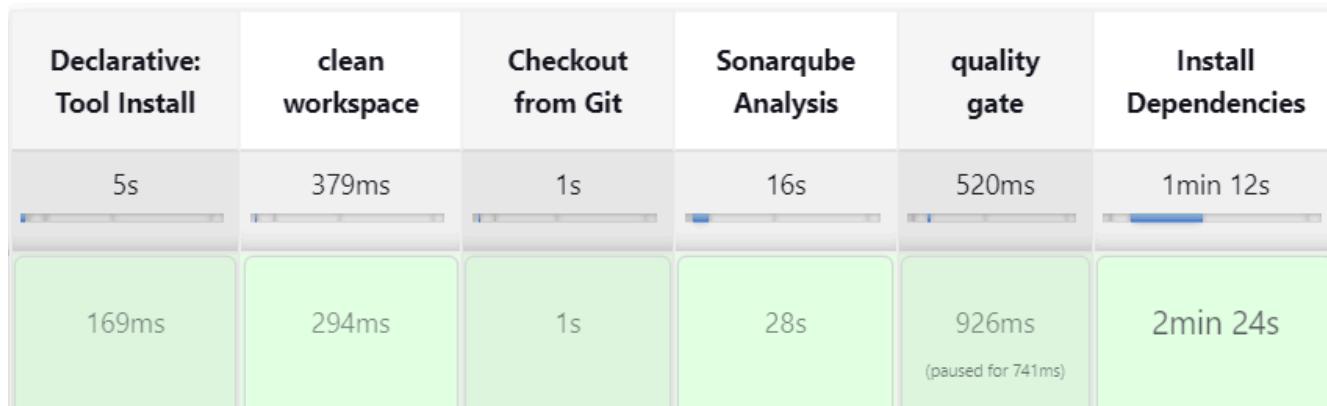
stage("quality gate"){
    steps {
        script {
            waitForQualityGate abortPipeline: false, credential:
        }
    }
}

stage('Install Dependencies') {
    steps {
        sh "npm install"
    }
}
}

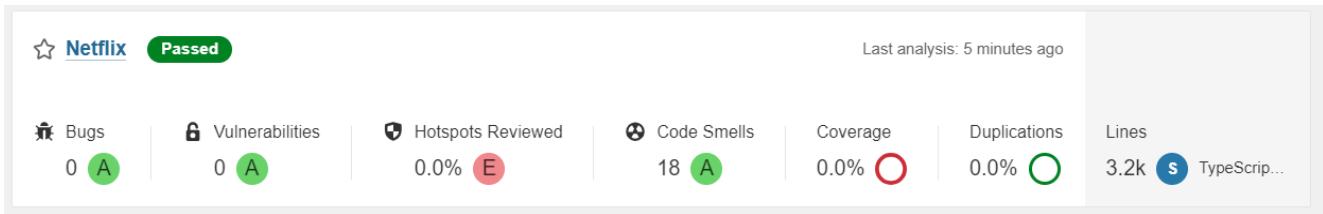
post {
    always {
        emailext attachLog: true,
        subject: "${currentBuild.result}",
        body: "Project: ${env.JOB_NAME}<br/>" +
            "Build Number: ${env.BUILD_NUMBER}<br/>" +
            "URL: ${env.BUILD_URL}<br/>",
        to: 'postbox.aj99@gmail.com',
        attachmentsPattern: 'trivyfs.txt,trivyimage.txt'
    }
}
}
}

```

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



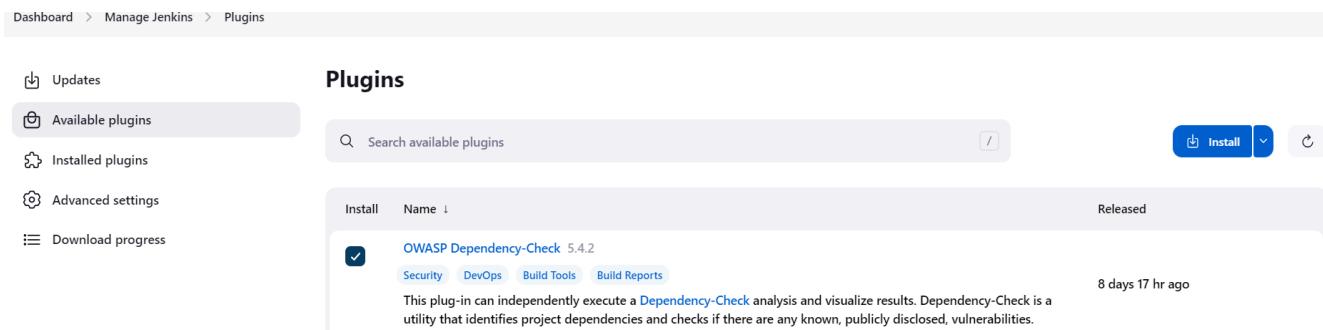
To see the report, you can go to Sonarqube Server and go to Projects.



You can see the report has been generated and the status shows as passed. You can see that there are 3.2k lines it scanned. To see a detailed report, you can go to issues.

## Step 9 – Install OWASP Dependency Check Plugins

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



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

Goto Dashboard → Manage Jenkins → Tools →

## Dependency-Check installations

[Add Dependency-Check](#)

### Dependency-Check

Name

DP-Check

 Install automatically [?](#)

#### Install from github.com

Version

dependency-check 6.5.1

[Add Installer ▾](#)

Click on Apply and Save here.

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

```
stage('OWASP FS SCAN') {  
    steps {  
        dependencyCheck additionalArguments: '--scan ./ --disab'  
        dependencyCheckPublisher pattern: '**/dependency-check-1  
    }  
}  
stage('TRIVY FS SCAN') {  
    steps {  
        sh "trivy fs . > trivyfs.txt"  
    }  
}
```

The stage view would look like this,



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

### Dependency-Check Results

SEVERITY DISTRIBUTION			
13	39	13	
			Search <input type="button" value="🔍"/>
File Name	Vulnerability	Severity	Weakness
+ ansi-html:0.0.7	[NVD] CVE-2021-23424	High	NVD-CWE-noinfo
+ ansi-regex:4.1.0	[NVD] CVE-2021-3807	High	CWE-1333
+ async:2.6.3	[NVD] CVE-2021-43138	High	CWE-1321
+ browserslist:4.14.2	[NVD] CVE-2021-23364	Medium	CWE-1333
+ css-what:3.4.2	[OSSINDEX] CVE-2022-21222	High	CWE-1333
+ decode-uri-component:0.2.0	[NVD] CVE-2022-38778	Medium	CWE-20
+ decode-uri-component:0.2.0	[NVD] CVE-2022-38900	High	CWE-20
+ ejs:2.7.4	[OSSINDEX] CVE-2022-29078	High	CWE-94
+ eventsource:1.1.0	[NVD] CVE-2022-1650	Critical	CWE-212
+ express:4.17.1	[OSSINDEX] CVE-2022-24999	High	CWE-1321

## Step 10 – Docker Image Build and Push

We need to install the Docker tool in our system, 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 without restart

The screenshot shows the Jenkins Plugins page with a search bar for "docker". The results list three plugins:

- Docker 1.5**: Version 439.va\_3cb\_0a\_6a\_fb\_29, released 3 days 15 hr ago. This plugin integrates Jenkins with Docker.
- Docker Commons**: Version 439.va\_3cb\_0a\_6a\_fb\_29, released 1 mo 29 days ago. Provides the common shared functionality for various Docker-related plugins.
- Docker Pipeline**: Version 572.v950f58993843, released 27 days ago. Build and use Docker containers from pipelines.
- Docker API**: Version 3.3.1-79.v20b\_53427e041, released 3 mo 4 days ago. This plugin provides docker-java API for other plugins.

Now, goto Dashboard → Manage Jenkins → Tools →

The screenshot shows the Jenkins Tools configuration page under "Docker installations". A new configuration is being added with the following details:

- Name**: docker
- Install automatically**: checked
- Download from docker.com**
- Docker version**: latest
- Add Installer**: dropdown menu

Add DockerHub Username and Password under Global Credentials

Kind

Username with password

Scope ?

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

Username ?

sevenajay

 Treat username as secret ?

Password ?

.....

ID ?

docker

Description ?

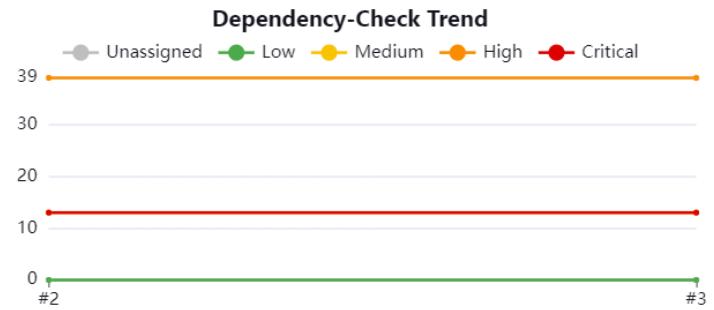
docker

**Create****Add this stage to Pipeline Script**

```
stage("Docker Build & Push"){
    steps{
        script{
            withDockerRegistry(credentialsId: 'docker', toolName:
                sh "docker build --build-arg TMDB_V3_API_KEY=Aj7z...
                sh "docker tag netflix sevenajay/netflix:latest '...
                sh "docker push sevenajay/netflix:latest "
            }
        }
    }
}
stage("TRIVY"){
    steps{
        sh "trivy image sevenajay/netflix:latest > trivyimage.t...
    }
}
```



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



Declarative: Tool Install	clean workspace	Checkout from Git	Sonarqube Analysis	quality gate	Install Dependencies	OWASP FS SCAN	TRIVY FS SCAN	Docker Build & Push	TRIVY
3s	366ms	1s	19s	451ms	1min 20s	2min 1s	16s	3min 9s	4s
154ms	341ms	1s	25s	315ms	1min 36s	2min 31s	23s	3min 9s	4s

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

sevenajay / netflix

Description

This repository does not have a description

Docker commands

To push a new tag to this repository:

[Public View](#)

`docker push sevenajay/netflix:tagname`

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

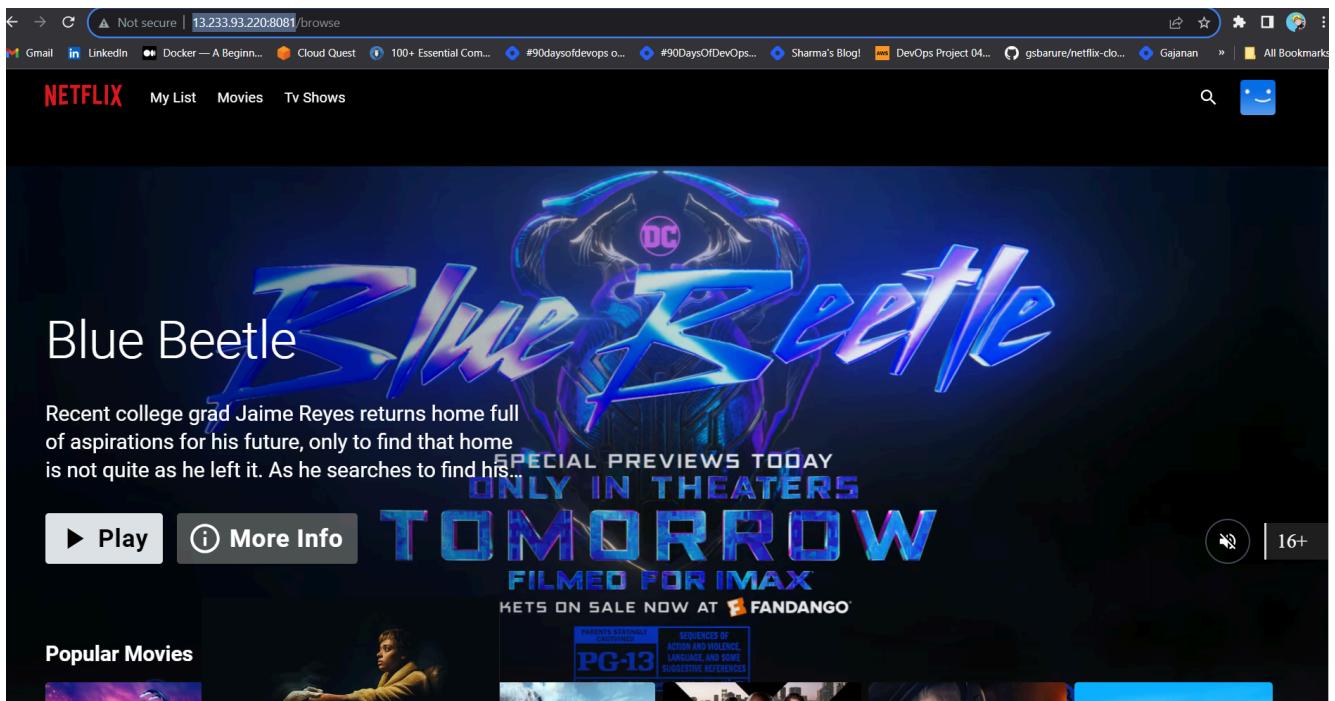
```
stage('Deploy to container'){
    steps{
        sh 'docker run -d --name netflix -p 8081:80 sevenajay/netflix'
    }
}
```

stage view



<Jenkins-public-ip:8081>

You will get this output



## Step 11 – Kuberentes Setup

Connect your machines to Putty or Mobaxtreme

**Take-Two Ubuntu 20.04 instances one for k8s master and the other one for worker.**

Install Kubectl on Jenkins machine also.

**Kubectl is to be installed on Jenkins also**

```
sudo apt update
sudo apt install curl
curl -LO https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/releas
sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
kubectl version --client
```



## Part 1 ----Master Node----



```
sudo hostnamectl set-hostname K8s-Master
```



## -----Worker Node-----



```
sudo hostnamectl set-hostname K8s-Worker
```



## Part 2 -----Both Master & Node -----



```
sudo apt-get update
sudo apt-get install -y docker.io
sudo usermod -aG docker Ubuntu
newgrp docker
sudo chmod 777 /var/run/docker.sock
```



## Part 3 ----- Master -----



```
sudo apt-get update
sudo apt-get install -y apt-transport-https ca-certificates curl gpg
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.29/deb/Release.key | sudo gpg --dearmor - > /etc/apt/keyrings/kubernetes-apt-keyring.gpg
echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.29/deb/ /' > /etc/apt/sources.list.d/kubernetes.list
sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
```



## -----Worker Node-----



```
sudo kubeadm init --pod-network-cidr=10.244.0.0/16
# in case you're in root exit from it and run below commands
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/
```

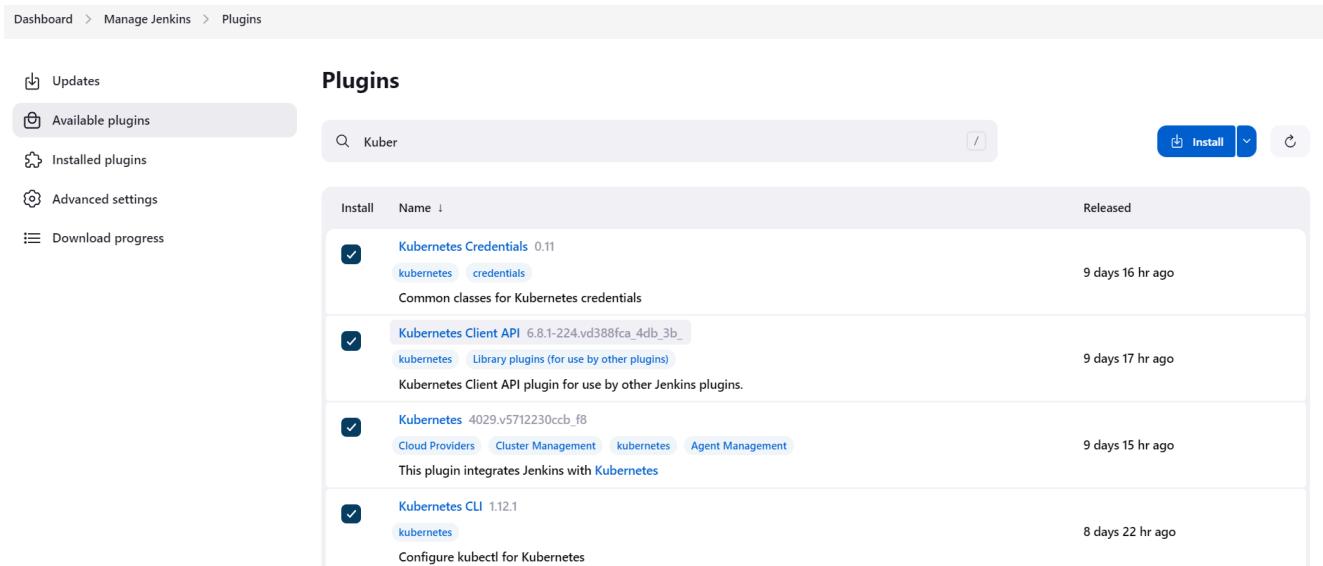


Copy the config file to Jenkins master or the local file manager and save it

copy it and save it in documents or another folder save it as secret-file.txt

Note: create a secret-file.txt in your file explorer save the config in it and use this at the kubernetes credential section.

Install Kubernetes Plugin, Once it's installed successfully



goto manage Jenkins -> manage credentials -> Click on Jenkins global -> add credentials

### New credentials

Kind

Secret file

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

File  
Choose File Secret File.txt

ID ?  
k8s

Description ?  
k8s

**Create**

## Install Node\_exporter on both master and worker

Let's add Node\_exporter on Master and Worker to monitor the metrics

First, let's create a system user for Node Exporter by running the following command:

```
sudo useradd \
--system \
--no-create-home \
--shell /bin/false node_exporter
```

```
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ 
ubuntu@ip-172-31-38-156:~$ sudo useradd \
--system \
--no-create-home \
--shell /bin/false node_exporter
ubuntu@ip-172-31-38-156:~$
```

You can [download Node Exporter](#) from the same page.

Use the wget command to download the binary.

```
wget https://github.com/prometheus/node_exporter/releases/download/v1.6.1/node_exporter-1.6.1.linux-amd64.tar.gz
```

```
ubuntu@ip-172-31-38-156:~$ wget https://github.com/prometheus/node_exporter/releases/download/v1.6.1/node_exporter-1.6.1.linux-amd64.tar.gz
--2023-10-06 09:03:19-- https://github.com/prometheus/node_exporter/releases/download/v1.6.1/node_exporter-1.6.1.linux-amd64.tar.gz
Resolving github.com (github.com)... 20.207.73.82
Connecting to github.com (github.com)|20.207.73.82|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/9524057/5509b569-5c34-471e-8598-c05c0733bb7f?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWNYAX4CSVEH53A%2F20231006%2Fus-east-1%2Faws4_request&&X-Amz-Date=20231006T090319Z&X-Amz-Expires=300&X-Amz-Signature=e11c47ad7d30d29de36062c68bc38d50f1433ef28c496a3403&&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=9524057&response-content-disposition=attachment%3Bfilename%3Dnode_exporter-1.6.1.linux-amd64.tar.gz&response-content-type=application%2Foctet-stream [following]
--2023-10-06 09:03:19-- https://objects.githubusercontent.com/github-production-release-asset-2e65be/9524057/5509b569-5c34-471e-8598-c05c0733bb7f?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWNYAX4CSVEH53A%2F20231006%2Fus-east-1%2Faws4_request&&X-Amz-Date=20231006T090319Z&X-Amz-Expires=300&X-Amz-Signature=e11c47ad7d30d29de36062c68bc38d50f1433ef28c496a0a73460781b5ege3403&&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=9524057&response-content-disposition=attachment%3Bfilename%3Dnode_exporter-1.6.1.linux-amd64.tar.gz&response-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.111.133, 185.199.108.133, 185.199.109.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 10368103 (9.9M) [application/octet-stream]
Saving to: 'node_exporter-1.6.1.linux-amd64.tar.gz'

node_exporter-1.6.1.linux-amd64.tar.gz      100%[=====]  9.89M  --.-KB/s   in  0.07s

2023-10-06 09:03:20 (135 MB/s) - 'node_exporter-1.6.1.linux-amd64.tar.gz' saved [10368103/10368103]

ubuntu@ip-172-31-38-156:~$
```

Extract the node exporter from the archive.

```
tar -xvf node_exporter-1.6.1.linux-amd64.tar.gz
```

```
ubuntu@ip-172-31-38-156:~$ ls
node_exporter-1.6.1.linux-amd64.tar.gz  prometheus-2.47.1.linux-amd64
ubuntu@ip-172-31-38-156:~$ tar -xvf node_exporter-1.6.1.linux-amd64.tar.gz
node_exporter-1.6.1.linux-amd64/
node_exporter-1.6.1.linux-amd64/NOTICE
node_exporter-1.6.1.linux-amd64/node_exporter
node_exporter-1.6.1.linux-amd64/LICENSE
ubuntu@ip-172-31-38-156:~$
```

Move binary to the /usr/local/bin.

```
sudo mv \
    node_exporter-1.6.1.linux-amd64/node_exporter \
    /usr/local/bin/
```

```
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ sudo mv \  
  node_exporter-1.6.1.linux-amd64/node_exporter \  
  /usr/local/bin/  
ubuntu@ip-172-31-38-156:~$ █
```

Clean up, and delete node\_exporter archive and a folder.



```
rm -rf node_exporter*
```



```
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ ls  
node_exporter-1.6.1.linux-amd64  node_exporter-1.6.1.linux-amd64.tar.gz  prometheus-2.47.1.linux-amd64  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ rm -rf node_exporter*  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ ls  
prometheus-2.47.1.linux-amd64  
ubuntu@ip-172-31-38-156:~$
```

Verify that you can run the binary.



```
node_exporter --version
```



```
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ node_exporter --version  
node_exporter, version 1.6.1 (branch: HEAD, revision: 4a1b77600c1873a8233f3ffb55afcedbb63b8d84)  
  build user:      root@0586879db11e5  
  build date:    20230717-12:10:52  
  go version:    go1.20.6  
  platform:      linux/amd64  
  tags:          netgo osusergo static_build  
ubuntu@ip-172-31-38-156:~$ █
```

Node Exporter has a lot of plugins that we can enable. If you run Node Exporter help you will get all the options.



```
node_exporter --help
```



-collector.logind We're going to enable the login controller, just for the demo.

Next, create a similar systemd unit file.



```
sudo vim /etc/systemd/system/node_exporter.service
```



```
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ sudo vim /etc/systemd/system/node_exporter.service
```

## node\_exporter.service



```
[Unit]  
Description=Node Exporter  
Wants=network-online.target  
After=network-online.target  
StartLimitIntervalSec=500  
StartLimitBurst=5  
[Service]  
User=node_exporter  
Group=node_exporter  
Type=simple  
Restart=on-failure  
RestartSec=5s  
ExecStart=/usr/local/bin/node_exporter \  
--collector.logind  
[Install]  
WantedBy=multi-user.target
```



```
[Unit]
Description=Node Exporter
Wants=network-online.target
After=network-online.target

StartLimitIntervalSec=500
StartLimitBurst=5

[Service]
User=node_exporter
Group=node_exporter
Type=simple
Restart=on-failure
RestartSec=5s
ExecStart=/usr/local/bin/node_exporter \
--collector.logind

[Install]
WantedBy=multi-user.target
```

Replace Prometheus user and group to node\_exporter, and update the ExecStart command.

To automatically start the Node Exporter after reboot, enable the service.

`sudo systemctl enable node_exporter`



Then start the Node Exporter.

`sudo systemctl start node_exporter`



```
ubuntu@ip-172-31-38-156:~$ sudo systemctl enable node_exporter
Created symlink /etc/systemd/system/multi-user.target.wants/node_exporter.service → /etc/systemd/system/node_exporter.service.
ubuntu@ip-172-31-38-156:~$ sudo systemctl start node_exporter
ubuntu@ip-172-31-38-156:~$
ubuntu@ip-172-31-38-156:~$
```

Check the status of Node Exporter with the following command:



```
sudo systemctl status node_exporter
```

```
ubuntu@tp-172-31-38-156:~$ sudo systemctl status node_exporter
● node_exporter.service - Node Exporter
   Loaded: loaded (/etc/systemd/system/node_exporter.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2023-10-06 09:07:39 UTC; 8s ago
       Main PID: 2030 (node exporter)
         Tasks: 3 (limit: 1141)
        Memory: 2.0M
          CPU: 9ms
        CGroup: /system.slice/node_exporter.service
                └─2030 /usr/local/bin/node_exporter --collector.logind

Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=thermal_zone
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=time
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=tunex
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=udp_queues
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=uname
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=vmstat
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=xfs
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.414Z caller=node_exporter.go:117 level=info collector=zfs
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.415Z caller=tls_config.go:274 level=info msg="listening on" address=[::]:9100
Oct 06 09:07:39 ip-172-31-38-156 node_exporter[2030]: ts=2023-10-06T09:07:39.415Z caller=tls_config.go:277 level=info msg="TLS is disabled." http2=false address=[::]:9100
ubuntu@tp-172-31-38-156:~$ █
```

If you have any issues, check logs with journalctl

```
journalctl -u node_exporter -f --no-pager
```

At this point, we have only a single target in our Prometheus. There are many different service discovery mechanisms built into Prometheus. For example, Prometheus can dynamically discover targets in AWS, GCP, and other clouds based on the labels. In the following tutorials, I'll give you a few examples of deploying Prometheus in a cloud-specific environment. For this tutorial, let's keep it simple and keep adding static targets. Also, I have a lesson on how to deploy and manage Prometheus in the Kubernetes cluster.

To create a static target, you need to add job\_name with static\_configs. Go to Prometheus server

```
sudo vim /etc/prometheus/prometheus.yml
```

```
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ sudo vim /etc/prometheus/prometheus.yml  
ubuntu@ip-172-31-38-156:~$
```

## prometheus.yml



```
- job_name: node_export_masterk8s
  static_configs:
    - targets: ["<master-ip>:9100"]
- job_name: node_export_workerk8s
  static_configs:
    - targets: ["<worker-ip>:9100"]
```



By default, Node Exporter will be exposed on port 9100.

```
alerting:
  alertmanagers:
    - static_configs:
      - targets:
        # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global 'evaluation_interval'.
rule_files:
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scraped from this config.
  - job_name: "prometheus"
    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ["localhost:9090"]

  - job_name: node_export
    static_configs:
      - targets: ["localhost:9100"]

  - job_name: 'jenkins'
    metrics_path: '/prometheus'
    static_configs:
      - targets: ['13.234.114.77:8080']

  - job_name: node_export_workerk8s
    static_configs:
      - targets: ["65.0.45.118:9100"]

  - job_name: node_export_master_k8s
    static_configs:
      - targets: ["43.204.100.15:9100"]
```

Since we enabled lifecycle management via API calls, we can reload the Prometheus config without restarting the service and causing downtime.

Before, restarting check if the config is valid.



```
promtool check config /etc/prometheus/prometheus.yml
```



```
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ promtool check config /etc/prometheus/prometheus.yml  
Checking /etc/prometheus/prometheus.yml  
SUCCESS: /etc/prometheus/prometheus.yml is valid prometheus config file syntax  
ubuntu@ip-172-31-38-156:~$
```

Then, you can use a POST request to reload the config.



```
curl -X POST http://localhost:9090/-/reload
```



```
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$  
ubuntu@ip-172-31-38-156:~$ curl -X POST http://localhost:9090/-/reload  
ubuntu@ip-172-31-38-156:~$
```

Check the targets section



```
http://<ip>:9090/targets
```



node_export (1/1 up) <a href="#">show less</a>					
Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
<a href="#">http://localhost:9100/metrics</a>	<span>UP</span>	instance="localhost:9100" job="node_export"	10.123s ago	17.500ms	

node_export_master_k8s (1/1 up) <a href="#">show less</a>					
Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
<a href="#">http://43.204.100.15:9100/metrics</a>	<span>UP</span>	instance="43.204.100.15:9100" job="node_export_master_k8s"	8.758s ago	20.950ms	

node_export_workerk8s (1/1 up) <a href="#">show less</a>					
Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
<a href="#">http://65.0.45.118:9100/metrics</a>	<span>UP</span>	instance="65.0.45.118:9100" job="node_export_workerk8s"	11.906s ago	24.987ms	

prometheus (1/1 up) <a href="#">show less</a>					
Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
<a href="#">http://localhost:9090/metrics</a>	<span>UP</span>	instance="localhost:9090" job="prometheus"	7.629s ago	5.791ms	

final step to deploy on the Kubernetes cluster

```
stage('Deploy to kubernets'){
    steps{
        script{
            dir('Kubernetes') {
                withKubeConfig(caCertificate: '', clusterName:
                    sh 'kubectl apply -f deployment.yml'
                    sh 'kubectl apply -f service.yml'
                }
            }
        }
    }
}
```

stage view

Declarative: Tool Install	clean workspace	Checkout from Git	Sonarqube Analysis	quality gate	Install Dependencies	OWASP FS SCAN	TRIVY FS SCAN	Docker Build & Push	TRIVY	Deploy to container	Deploy to kubernets
132ms	264ms	1s	25s	295ms	1min 49s	2min 38s	23s	1min 51s	1min 35s	1s	2s
133ms	261ms	1s	25s	284ms	1min 51s	2min 46s	23s	1min 23s	1min 52s	1s	1s

In the Kubernetes cluster(master) give this command



```
kubectl get all
kubectl get svc
```



```
ubuntu@ip-172-31-40-131:~$ kubectl get all
NAME                                         READY   STATUS    RESTARTS   AGE
pod/petshop-768578655f-kzcd9   1/1     Running   0          43s

NAME           TYPE      CLUSTER-IP      EXTERNAL-IP   PORT(S)      AGE
service/kubernetes  ClusterIP  10.96.0.1      <none>        443/TCP   58m
service/petshop   LoadBalancer  10.104.122.152  <pending>    80:30699/TCP  21m

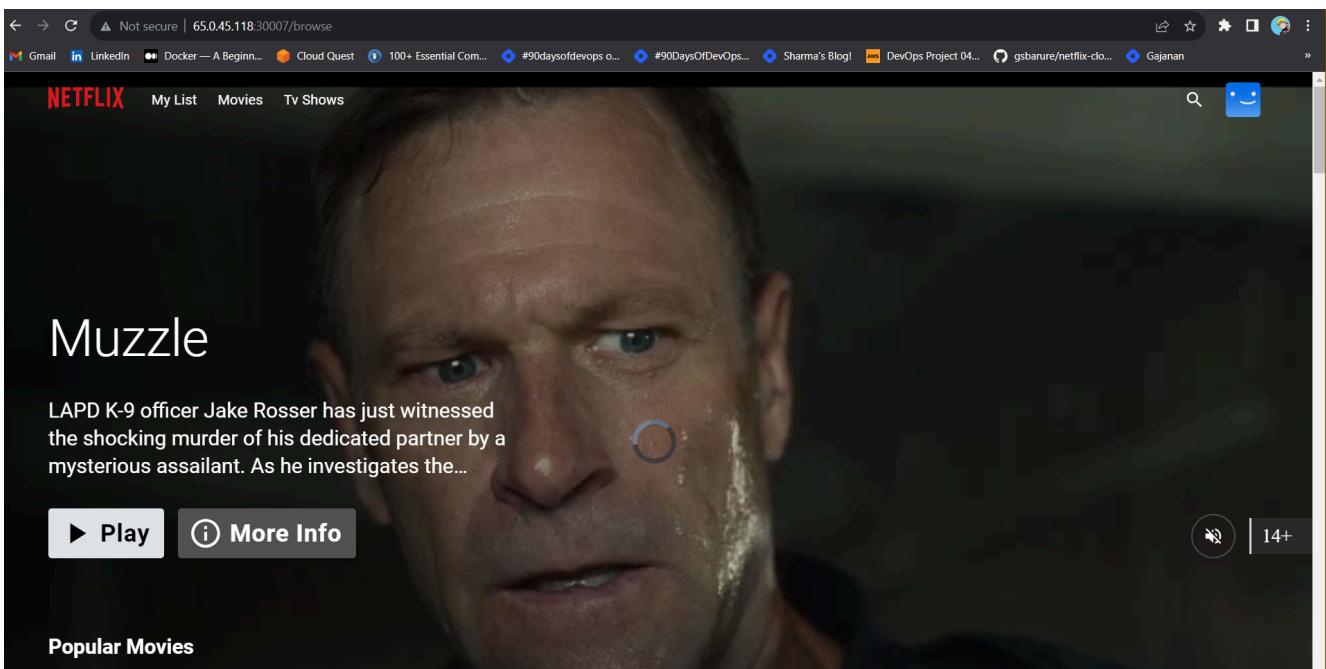
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/petshop  1/1       1           1          43s

NAME           DESIRED  CURRENT  READY   AGE
replicaset.apps/petshop-768578655f  1         1         1          43s
ubuntu@ip-172-31-40-131:~$ █
```

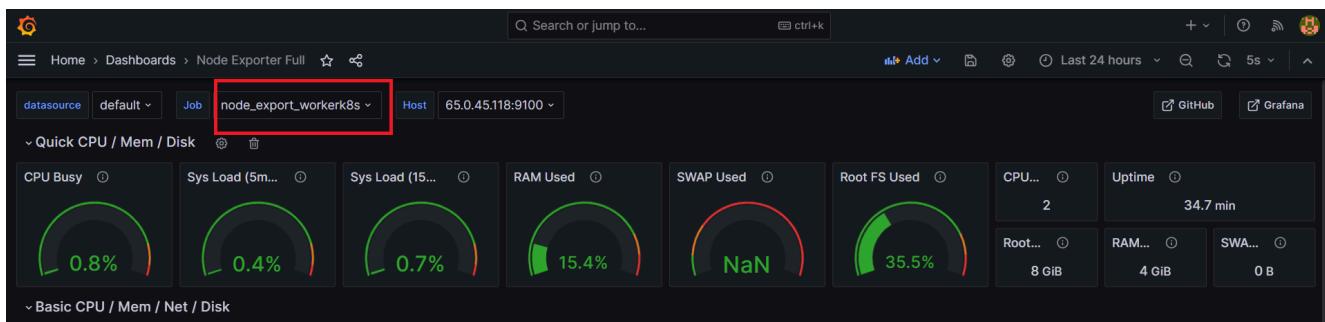
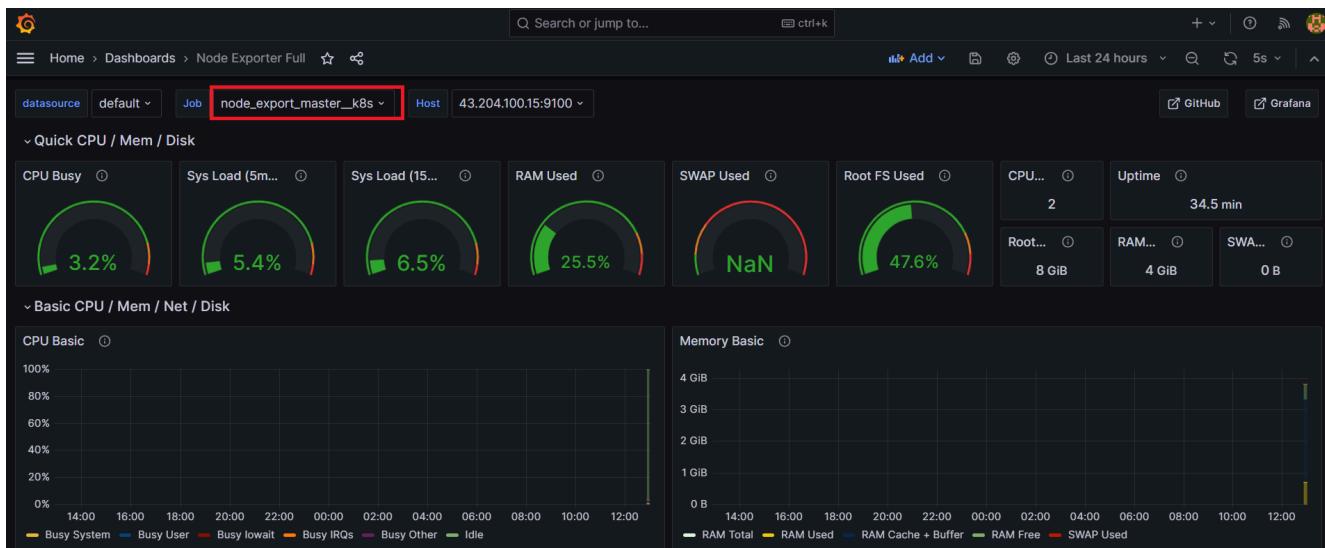
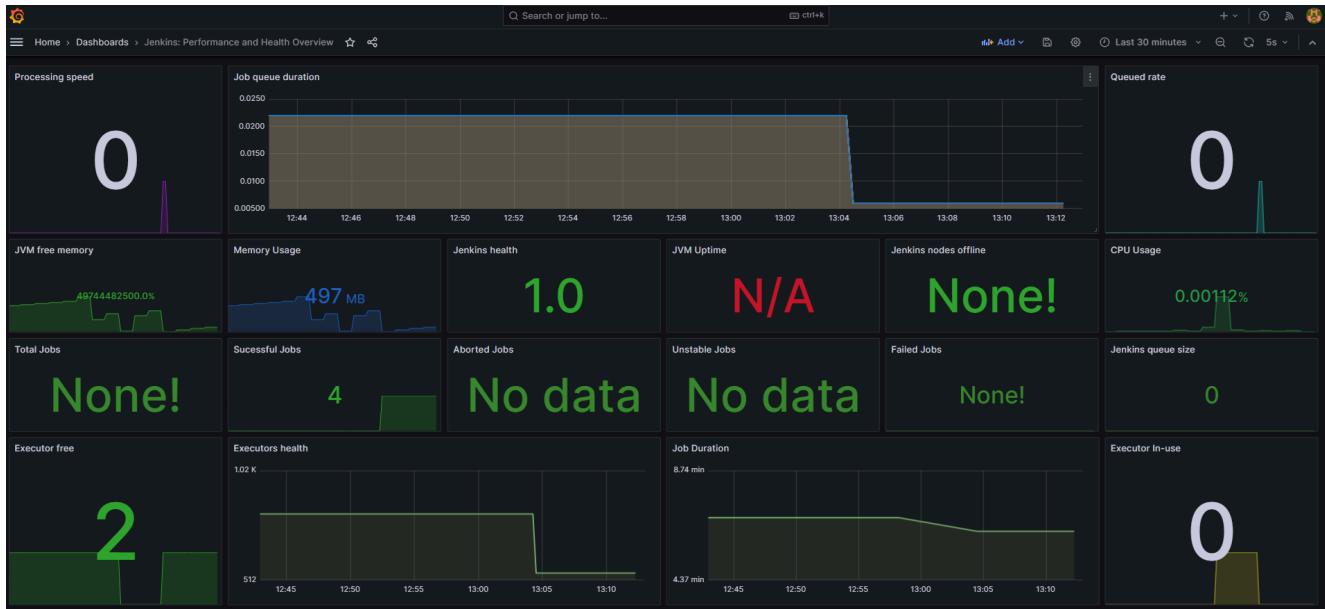
## STEP 12: Access from a Web browser with

<public-ip-of-slave:service port>

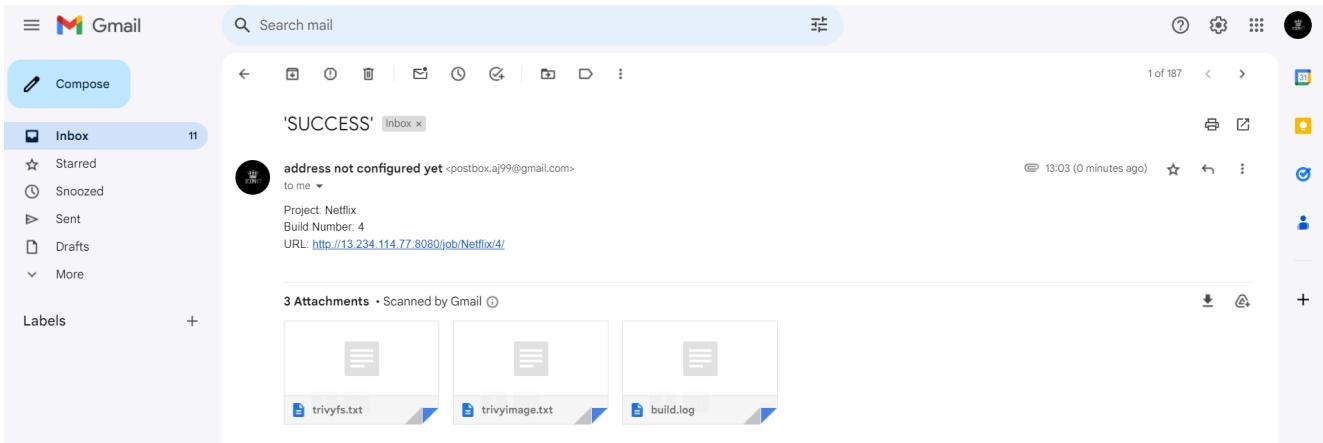
output:



## Monitoring



## Mail



## Step 13: Terminate instances.

### Complete 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/Aj7Ay/Netf'
            }
        }
        stage("Sonarqube Analysis"){
            steps{
                withSonarQubeEnv('sonar-server') {
                    sh ''' $SCANNER_HOME/bin/sonar-scanner -Dsonar.proje
                    -Dsonar.projectKey=Netflix '''
                }
            }
        }
    }
}
```

```
        }
    }
    stage("quality gate"){
        steps {
            script {
                waitForQualityGate abortPipeline: false, credentialsId: ''
            }
        }
    }
    stage('Install Dependencies') {
        steps {
            sh "npm install"
        }
    }
    stage('OWASP FS SCAN') {
        steps {
            dependencyCheck additionalArguments: '--scan ./ --disable-checks=missing-dependencies', 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', url: 'https://index.docker.io/v1/'){
                    sh "docker build --build-arg TMDB_V3_API_KEY=AJ7A8B9C0D1E2F3G4H5I6J7K8L9M0N1O2P3Q4R5S6T7U8V9W0X1Y2Z3"
                    sh "docker tag netflix sevenajay/netflix:latest"
                    sh "docker push sevenajay/netflix:latest"
                }
            }
        }
    }
    stage("TRIVY"){
        steps{
            sh "trivy image sevenajay/netflix:latest > trivyimage.txt"
        }
    }
    stage('Deploy to container'){
        steps{
            sh 'docker run -d --name netflix -p 8081:80 sevenajay/netflix'
        }
    }
}
```

```
        }
        stage('Deploy to kubernets'){
            steps{
                script{
                    dir('Kubernetes') {
                        withKubeConfig(caCertificate: '', clusterName:
                            sh 'kubectl apply -f deployment.yml'
                            sh 'kubectl apply -f service.yml'
                        )
                    }
                }
            }
        }

    }
post {
    always {
        emailext attachLog: true,
        subject: "'${currentBuild.result}'",
        body: "Project: ${env.JOB_NAME}<br/>" +
            "Build Number: ${env.BUILD_NUMBER}<br/>" +
            "URL: ${env.BUILD_URL}<br/>",
        to: 'postbox.aj99@gmail.com',
        attachmentsPattern: 'trivyfs.txt,trivyimage.txt'
    }
}
}
```



Ajay Kumar Yegireddi is a DevSecOps Engineer and System Administrator, with a passion for sharing real-world DevSecOps projects and tasks. **Mr. Cloud Book**, provides hands-on tutorials and practical insights to help others master DevSecOps tools and workflows. Content is designed to bridge the gap between development, security, and operations, making complex concepts easy to understand for both beginners and professionals.

# Comments

21 responses to “Netflix Clone CI/CD with Monitoring | Jenkins | Docker| Kubernetes| Monitoring | DevSecOps”



**Sampath**

10 January 2024

Hi,

This was great. I learned a lot here. Keep up the good work.

I found that you were missing the Trivy installation in above. So I used the code below for the installation.

```
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  
apt-get install trivy  
trivy --version
```

Regards,  
Sampath

[Reply](#)



**mrcloudbook.com**

11 January 2024

I will see thaanks for Update

[Reply](#)



**mrcloudbook.com**

11 January 2024

The code is already in blog bro may be you missed it

[Reply](#)

---



**Parag Vade**

18 November 2024

You are correct Sampath. Thank you very much for your comment.  
Was stuck at this.

And thank you so much Ajay.  
Your efforts for sharing knowledge are exceptional.

[Reply](#)

---



**snehalk**

14 March 2024

Thank you for sharing such a good project.

[Reply](#)

---



**Ravi**

29 March 2024

Thank you so much for the End-to-End Pipeline with detailed presentation. One of the best in recent times.

[Reply](#)

---



**charan**

14 April 2024

blue bottle image and muzzle image coming your netflix same thing i done that image not getting how to that image coming tell me bro

[Reply](#)

---

**mrcloudbook.com**

14 April 2024

I didnt get your point brother

[Reply](#)**Rajat**

21 April 2024

Wonderful project I have saw ever. Thanks

[Reply](#)**sharafat**

7 May 2024

First of all thanks for sharing such a wonderfull project. My questions is why pipeline always stuck at sonar quality gate. Can we make it faster. it runs fine without quality gate. but when i add that step pipeline stuck for long time.

Thanks

[Reply](#)**Johnprxy**

22 August 2024

it happened to me too, but when i use the same version of sonar used in the project, it worked and it was faster

[Reply](#)**sarvadnya**

26 May 2024

Hey Ajay, thank you for this wonderful project, your documentation was also on point, I am a newbie to Grafana, why while importing the dashboard, why we using number code, what is it?

[Reply](#)

---



**mrcloudbook.com**

11 June 2024

Number code is predefined Dashboard and already existing one and you can also create one

[Reply](#)

---



**Stephen**

28 May 2024

Thank Ajay for this project

[Reply](#)

---



**Licznik Prqdu**

7 June 2024

Your article is a great mix of thorough research and personal perspective. Really enjoyed it!

[Reply](#)

---



**jagadeesh**

8 July 2024

Hi Ajay,

First of all, thank you for this wonderful project and document. I have a small doubt or request—could you please create a Robot shopping cart project? I'm asking

because you explain each point clearly and provide excellent documentation. It would be great to see a project like this from you.

Thank you!

[Reply](#)

---



**Harsh Gupta**

25 December 2024

TMDB website is opening.

error :

This site can't be reached

<http://www.themoviedb.org> took too long to respond.

[Reply](#)

---



**mrcloudbook.com**

29 December 2024

Use VPN

[Reply](#)

---



**Robin Rawat**

27 December 2024

amazing content brother

[Reply](#)

---



**Robin Rawat**

28 December 2024

kluster is not deploying with this cmd error coming

[Reply](#)

mrcloudbook.com

29 December 2024

Use 24.04 setup

[Reply](#)

## Leave a Reply

Your email address will not be published. Required fields are marked \*

Comment \*

Name \*

Email \*

Website

Save my name, email, and website in this browser for the next time I comment.



I'm not a robot

reCAPTCHA  
[Privacy](#) - [Terms](#)

[Post Comment](#)

Uncategorized

**How to Automate Incident Response : How Q Developer Helped Me Automate a Daily Pain Point**

22 July 2025

AI

**How to Run Docker Model Runner on Ubuntu 24.04**

11 July 2025

AI, DevOps

**How to Install docker-ai on Ubuntu 24.04**

15 June 2025

## Upskill with Ajay: DevSecOps Mastery

Join Mr Cloud book to master DevSecOps through real-world projects. Learn CI/CD, security integration, automation, and more, gaining hands-on skills for industry-level challenges.



## Important Links

[Privacy Policy](#)[Terms & Conditions](#)[Contact](#)

## Resources

[Blog](#)[YouTube Channel](#)

