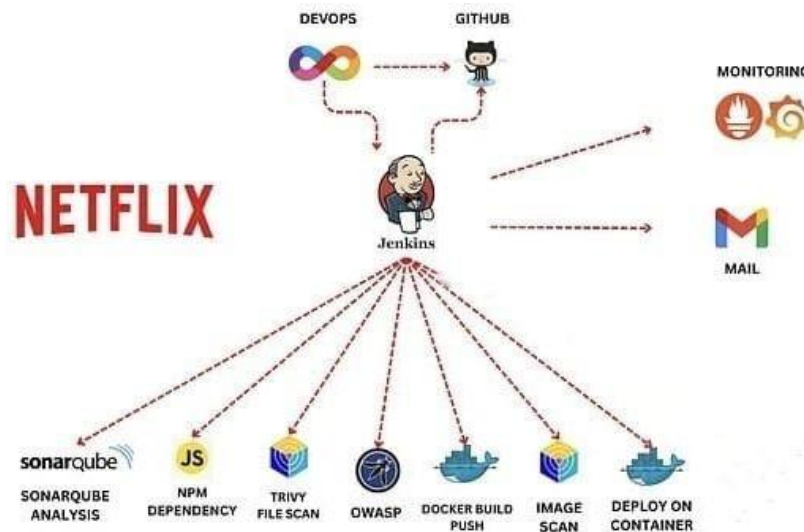


Netflix Clone CI-CD



Project created by : Janhavi Bagul

Link : https://github.com/JanhaviBagul315/Netflix_clone

Project Overview

I will be deploying a Netflix clone. I will be using Jenkins as a CICD tool and deploying our application on a Docker container and I will monitor the Jenkins using Grafana, Prometheus and Node exporter.

Project 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
- Step 9 — Install OWASP Dependency Check Plugins
- Step 10 — Docker Image Build and Push
- Step 11 — Deploy the image using Docker
- Step 12 — Access the Netflix app on the Browser.
- Step 13 — Terminate the AWS EC2 Instances.

Step 1 : Launch ubuntu instance t2.large

Instances (1) Info

↻

Connect

Instance state ▾

Actions ▾

Launch instances ▾

🔍 Find Instance by attribute or tag (case-sensitive)

< 1 > ⚙

<input type="checkbox"/>	Name ✎ ▾	Instance ID	Instance state ▾	Instance type ▾	Status check	Alarm status	Availability Zone
<input type="checkbox"/>	Jenkins	i-0b94fe1eebaff9531	🟢 Running ⚙ 🔍	t2.large	🟢 2/2 checks passed	No alarms +	us-east-1e

Step 2 : Login to the Instance

```
ubuntu@ip-172-31-49-232: ~
login as: ubuntu
Authenticating with public key "Linuxkey"
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.0-1012-aws x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:       https://ubuntu.com/advantage

System information as of Sun Nov 19 08:02:26 UTC 2023

System load:  0.0087890625      Processes:            113
Usage of /:   5.4% of 28.89GB   Users logged in:     0
Memory usage: 2%               IPv4 address for eth0: 172.31.49.232
Swap usage:   0%
```

Step 3 : Create one shell script file, bcz of we need to install Jenkins on this server

```
ubuntu@ip-172-31-49-232: ~
ubuntu@ip-172-31-49-232:~$ sudo vi jenkins.sh
```

Step 4 : Add the Jenkins download steps in the script file

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

sudo apt-get update -y
sudo apt-get install jenkins -y
sudo systemctl start jenkins
sudo systemctl status jenkins
```

```
ubuntu@ip-172-31-49-232: ~
#!/bin/bash
sudo apt update -y
#sudo apt upgrade -y
wget -O - https://packages.adoptium.net/artifactory/api/gpg/key/public | tee /etc/apt/keyrings/adoptium.asc
echo "deb [signed-by=/etc/apt/keyrings/adoptium.asc] https://packages.adoptium.net/artifactory/deb $(awk -F= '/^VERSION_CODENAME/{print$2}' /etc/os-release) main" | tee
/etc/apt/sources.list.d/adoptium.list
sudo apt update -y
sudo apt install temurin-17-jdk -y
/usr/bin/java --version
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \
/usr/share/keyrings/jenkins-keyring.asc > /dev/null
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
/etc/apt/sources.list.d/jenkins.list > /dev/null

sudo apt-get update -y
sudo apt-get install jenkins -y
sudo systemctl start jenkins
```

Step 5 : Give Permission to the file : (**sudo chmod 777 jenkins.sh**)

```
ubuntu@ip-172-31-49-232:~$ sudo chmod 777 jenkins.sh
```

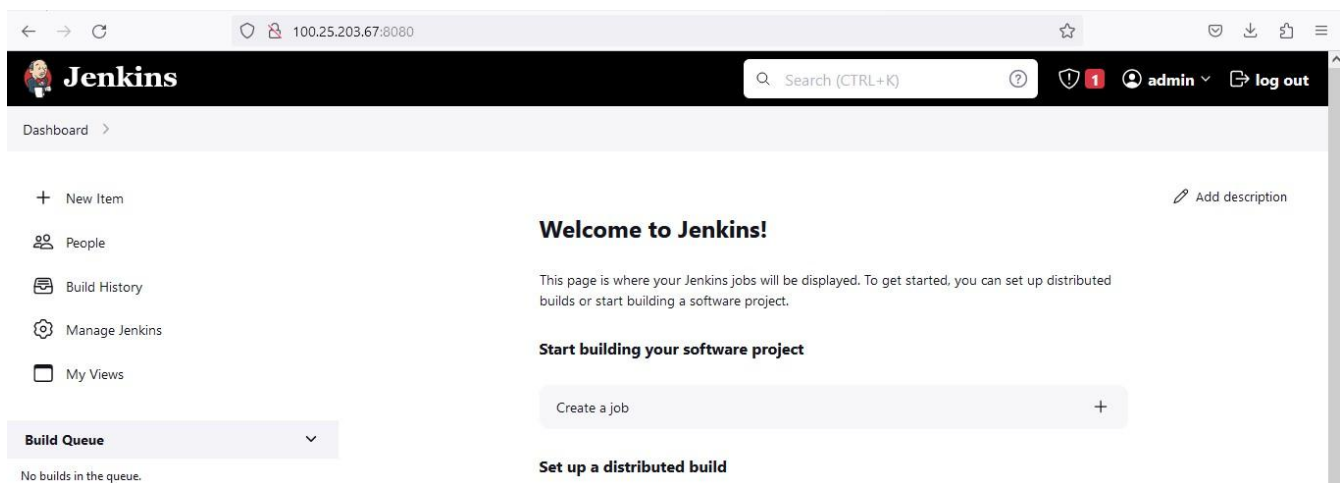
Step 6 : Then run the script file (**./ Jenkins.sh**)

```
ubuntu@ip-172-31-49-232:~$ ./jenkins.sh
```

Step 7 : Copy the server ip with port no.8080 and paste in google u get the Jenkins page



Step 8 : Jenkins Installed successfully



Step 9 : Then Install docker on the same server

Commands

```
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
```

Step 10 : Then run the sonar in container (**docker run -d --name sonar -p 9000:9000 sonarqube:lts-community**)

```
ubuntu@ip-172-31-49-232: ~
ubuntu@ip-172-31-49-232:~$ docker run -d --name sonar -p 9000:9000 sonarqube:lts-community
```

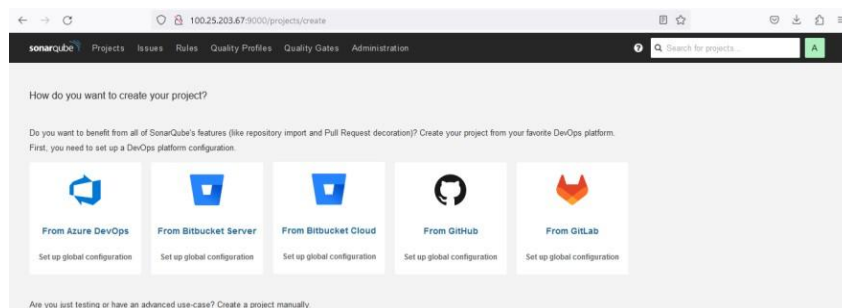
Step 11 : Sonar Container created successfully

```
-----
ubuntu@ip-172-31-49-232:~$ docker ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED        STATUS        PORTS                               NAMES
1c1a6e94cbf2   sonarqube:lts-community "/opt/sonarqube/dock_   About a minute Up About a minute 0.0.0.0:9000->9000/tcp, :::9000->9000/tcp   sonar
ubuntu@ip-172-31-49-232:~$
```

Step 12 : Copy the server ip with port no 9000 u get the sonar page (username and pwd= admin)



Step 13 : Sonar created successfully



Step 14 : Create token in sonar (path – Administration/security/update token/create token)

Tokens of Administrator

Generate Tokens

Name Expires in 30 days

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

Copy `squ_74685f4e4fa70bdf161adc35b30206b52046122c`

Name	Type	Project	Last use	Created	Expiration	
sonar-token	User		Never	November 19, 2023	December 19, 2023	<input type="button" value="Revoke"/>

Step 15 : Create webhook in sonar (path – Configuration/webhooks/create webhooks, name = Jenkins and URL = Copy and paste the Jenkins URL and add /sonarqube-webhook/)

Create Webhook

All fields marked with * are required

Name *

URL *

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

Secret

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

Administration

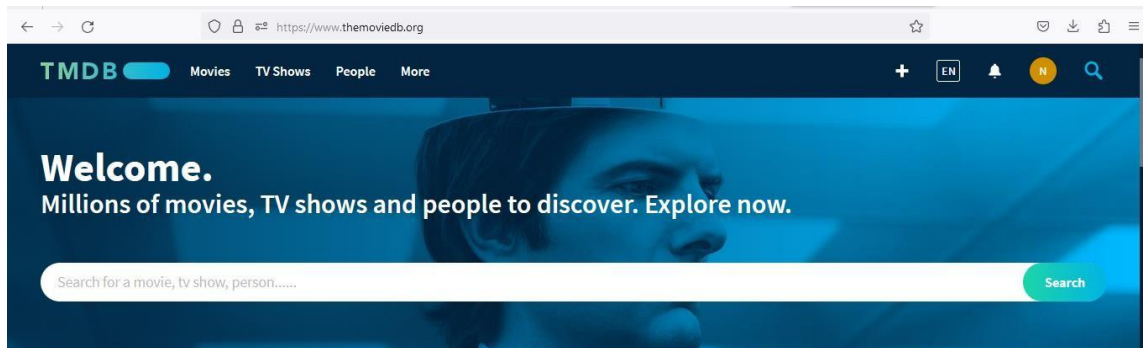
[Configuration](#) [Security](#) [Projects](#) [System](#) [Marketplace](#)

Webhooks

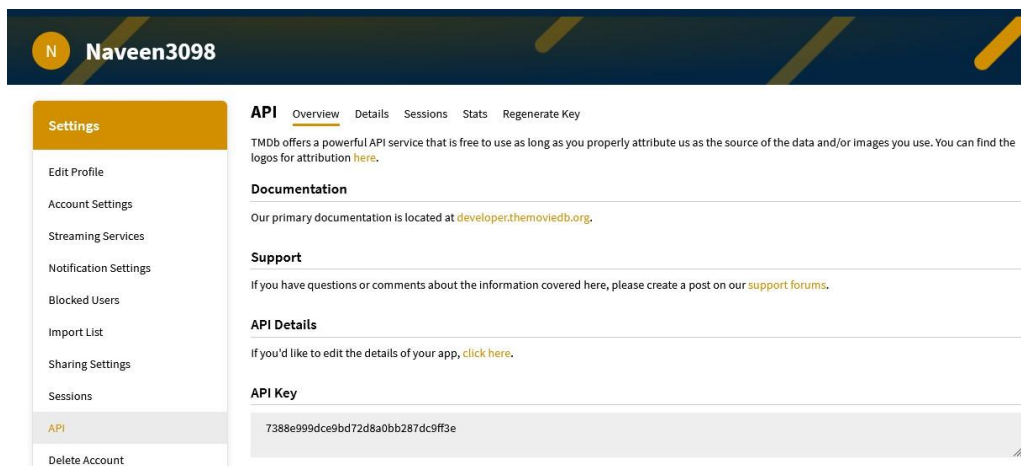
Webhooks are used to notify external services when a project analysis is done. An HTTP POST request including a JSON payload is sent to each of the provided URLs. Learn more in the [Webhooks documentation](#).

Name	URL	Has secret?	Last delivery	Actions
jenkins	http://100.25.203.67:8080/sonarqube-webhook/	No	Never	

Step 16 : Go to google and search TMDB and click on first link



Step 17 : Create API (Path – click profile icon/setting/API/create API)



Step 18 : Create one more new server name of Prometheus/grafana

Instances (1/2) Info								Launch instances	
Find Instance by attribute or tag (case-sensitive)								< 1 >	
	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone		
<input type="checkbox"/>	Prometheus-G...	i-0ec85d00187ce92ad	Running	t2.medium	2/2 checks passed	No alarms	us-east-1e		

Step 19 : Instance created successfully

```
ubuntu@ip-172-31-51-59: ~  
login as: ubuntu  
Authenticating with public key "Linuxkey"  
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.0-1012-aws x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:       https://ubuntu.com/advantage  
  
System information as of Sun Nov 19 09:17:05 UTC 2023  
  
System load:  0.00146484375   Processes:            106  
Usage of /:   5.4% of 28.89GB   Users logged in:     0  
Memory usage: 6%              IPv4 address for eth0: 172.31.51.59  
Swap usage:   0%
```

Step 20 : To create a system user or system account, run the following command:

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

Step 21 : Download Prometheus :

(wget <https://github.com/prometheus/prometheus/releases/download/v2.47.1/prometheus-2.47.1.linux-amd64.tar.gz>)

```
ubuntu@ip-172-31-51-59: ~  
ubuntu@ip-172-31-51-59:~$ wget https://github.com/prometheus/prometheus/releases/download/v2.47.1/prometheus-2.47.1.linux-amd64.tar.gz
```

Step 22 : Untar the Prometheus file (**tar -xvf prometheus-2.47.1.linux-amd64.tar.gz**)

```
ubuntu@ip-172-31-51-59:~$ tar -xvf prometheus-2.47.1.linux-amd64.tar.gz
```

Step 23 : Create one Prometheus directory under etc directory (**sudo mkdir -p /data /etc/prometheus**)

```
ubuntu@ip-172-31-51-59: ~  
ubuntu@ip-172-31-51-59:~$ sudo mkdir -p /data /etc/prometheus  
ubuntu@ip-172-31-51-59:~$
```


Step 24 : Enter into the Prometheus directory (**cd prometheus-2.47.1.linux-amd64/**)

```
ubuntu@ip-172-31-51-59:~$ cd prometheus-2.47.1.linux-amd64/
ubuntu@ip-172-31-51-59:~/prometheus-2.47.1.linux-amd64$ ls
LICENSE NOTICE console Libraries consoles prometheus prometheus.yml promtool
ubuntu@ip-172-31-51-59:~/prometheus-2.47.1.linux-amd64$
```

Step 25 : Move the Prometheus & promtool file to /usr/local/bin

Move the console & console libraries and prometheus.yml file to etc/Prometheus

(**sudo mv prometheus promtool /usr/local/bin/**

sudo mv consoles/ console_libraries/ /etc/prometheus/

sudo mv prometheus.yml /etc/prometheus/prometheus.yml)

```
ubuntu@ip-172-31-51-59:~/prometheus-2.47.1.linux-amd64$ sudo mv prometheus promtool /usr/local/bin/
ubuntu@ip-172-31-51-59:~/prometheus-2.47.1.linux-amd64$ sudo mv consoles/ console_libraries/ /etc/prometheus/
ubuntu@ip-172-31-51-59:~/prometheus-2.47.1.linux-amd64$ sudo mv prometheus.yml /etc/prometheus/prometheus.yml
ubuntu@ip-172-31-51-59:~/prometheus-2.47.1.linux-amd64$
```

Step 26 : 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-172-31-51-59:~/prometheus-2.47.1.linux-amd64$ sudo chown -R prometheus:prometheus /etc/prometheus/ /data/
ubuntu@ip-172-31-51-59:~/prometheus-2.47.1.linux-amd64$
```

Step 27 : 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-172-31-51-59:~/prometheus-2.47.1.linux-amd64$ cd ..
ubuntu@ip-172-31-51-59:~$ ls
prometheus-2.47.1.linux-amd64 prometheus-2.47.1.linux-amd64.tar.gz
ubuntu@ip-172-31-51-59:~$ rm -rf prometheus-2.47.1.linux-amd64 prometheus-2.47.1.linux-amd64.tar.gz
```

Step 28 : 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**)

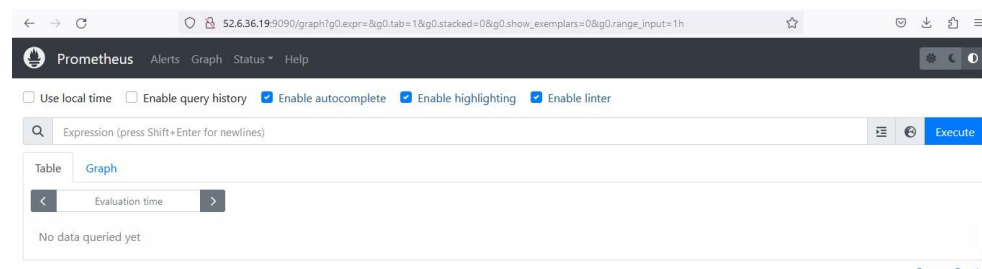
```
ubuntu@ip-172-31-51-59:~$ sudo vim /etc/systemd/system/prometheus.service
```

<pre>ubuntu@ip-172-31-51-59: ~ [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</pre>	<pre>[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</pre>
---	---

Step 30 : Start the Prometheus service (**sudo systemctl enable Prometheus**)

```
ubuntu@ip-172-31-51-59:~$ sudo systemctl enable prometheus  
Created symlink /etc/systemd/system/multi-user.target.wants/prometheus.service -> /etc/systemd/system/prometheus.service.  
ubuntu@ip-172-31-51-59:~$ sudo systemctl start prometheus  
ubuntu@ip-172-31-51-59:~$
```

Step 31 : Copy the Prometheus server ip with port no.9090 u get Prometheus page



Step 32 : To create a system user or system account, run the following command:

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

Code

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

Step 33 : Download node exporter (wget

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

```
ubuntu@ip-172-31-51-59:~$ wget https://github.com/prometheus/node_exporter/releases/download/v1.6.1/node_exporter-1.6.1.linux-amd64.tar.gz
```

Step 34 : Untar the node exporter file (tar -xvf node_exporter-1.6.1.linux-amd64.tar.gz)

```
ubuntu@ip-172-31-51-59: ~  
ubuntu@ip-172-31-51-59:~$ ls  
node_exporter-1.6.1.linux-amd64.tar.gz  
ubuntu@ip-172-31-51-59:~$ tar -xvf node_exporter-1.6.1.linux-amd64.tar.gz
```

Step 35 : Move the node exporter file

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

```
ubuntu@ip-172-31-51-59:~$ ls  
node_exporter-1.6.1.linux-amd64 node_exporter-1.6.1.linux-amd64.tar.gz  
ubuntu@ip-172-31-51-59:~$ sudo mv \  
  node_exporter-1.6.1.linux-amd64/node_exporter \  
  /usr/local/bin/
```

Step 36 : After moving the node exporter file remove the tar file (`rm -rf node_exporter*`)

```
ubuntu@ip-172-31-51-59: ~  
ubuntu@ip-172-31-51-59:~$ rm -rf node_exporter*  
ubuntu@ip-172-31-51-59:~$
```

Step 37 : 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/node_exporter.service`)

```
ubuntu@ip-172-31-51-59:~$ sudo vim /etc/systemd/system/node_exporter.service
```

Step 38 : Add script in the file

<pre>ubuntu@ip-172-31-51-59: ~ [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</pre>	<pre>[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</pre>
--	---

Step 39 : Start the node exporter

```
service ( sudo systemctl enable  
node_exporter sudo systemctl start  
node_exporter )
```

```
ubuntu@ip-172-31-51-59:~$ 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-172-31-51-59:~$ sudo systemctl start node_exporter  
ubuntu@ip-172-31-51-59:~$
```

Step 40 : Add the node exporter job in Prometheus.yml

```
file ( sudo vim /etc/prometheus/prometheus.yml )
```

```
ubuntu@ip-172-31-51-59:~$ sudo vim /etc/prometheus/prometheus.yml
```

Step 41 : Node exporter job with port no.9100, copy this code and paste in Prometheus.yml file

```
- job_name: "Node-Exporter"  
  
  # metrics_path defaults to '/metrics'  
  # scheme defaults to 'http'.  
  
  static_configs:  
    - targets: ["52.6.36.19:9100"]
```

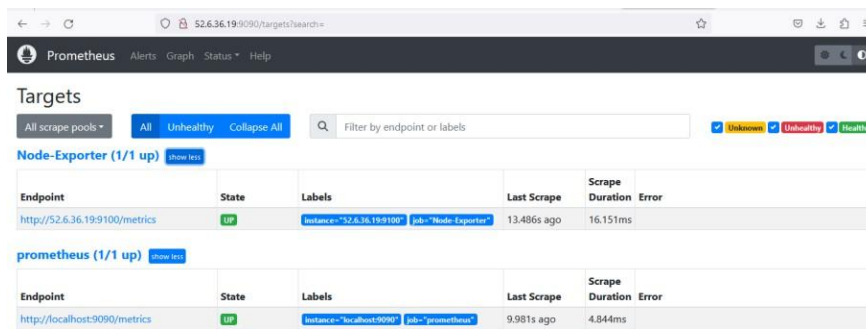
```
- job_name: node_export  
  
static_configs:  
  
  - targets: ["localhost:9100"]
```

Step 42 : Start and reload the service

```
( promtool check config  
/etc/prometheus/prometheus.yml ) ( curl -X POST  
http://localhost:9090/-/reload )
```

```
ubuntu@ip-172-31-51-59:~$ 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-172-31-51-59:~$ curl -X POST http://localhost:9090/-/reload  
ubuntu@ip-172-31-51-59:~$
```

Step 43 : Go to Prometheus/target u saw the node exporter job



The screenshot shows the Prometheus web interface. The 'Targets' tab is selected, displaying a list of scrape targets. There are two targets listed: 'Node-Exporter (1/1 up)' and 'prometheus (1/1 up)'. Each target has a table with columns for Endpoint, State, Labels, Last Scrape, Scrape Duration, and Error.

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://52.6.36.19:9100/metrics	UP	instance="52.6.36.19:9100" job="Node-Exporter"	13.486s ago	16.151ms	

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:9090/metrics	UP	instance="localhost:9090" job="prometheus"	9.981s ago	4.844ms	

Step 44 : Download grafana

(sudo apt-get install -y apt-transport-https software-properties-common)

```
ubuntu@ip-172-31-51-59:~$ sudo apt-get install -y apt-transport-https software-properties-common
```

Step 45 : Download gpg key (wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -)

```
ubuntu@ip-172-31-51-59:~$ 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-172-31-51-59:~$
```

Step 46 : 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-172-31-51-59:~$ 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-172-31-51-59:~$
```

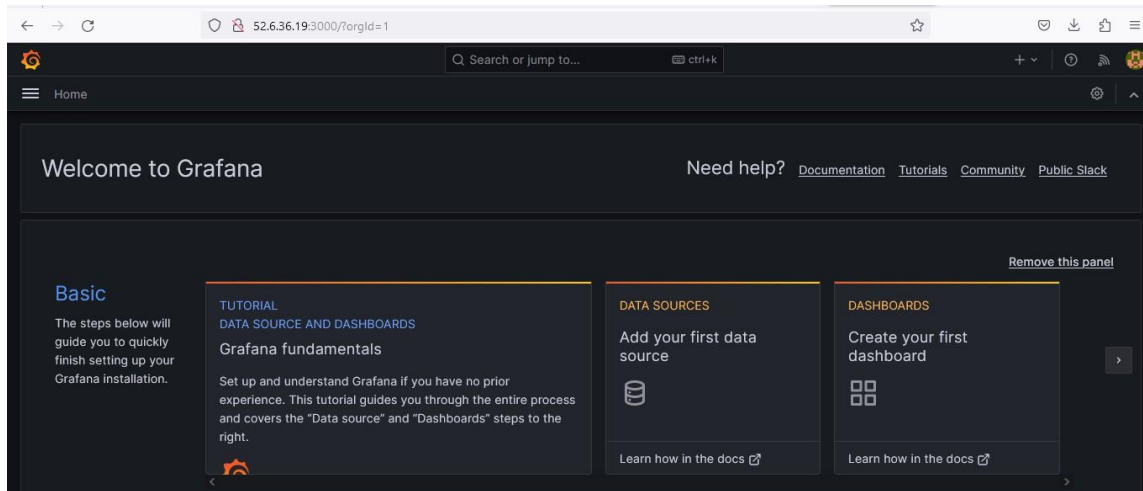
Step 47 : Download Grafana by the below command

```
sudo apt-get update
```

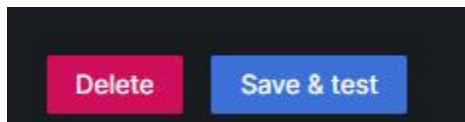
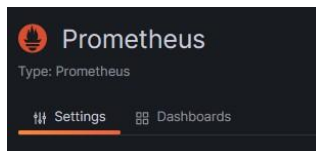
```
sudo apt-get -y install grafana
```

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

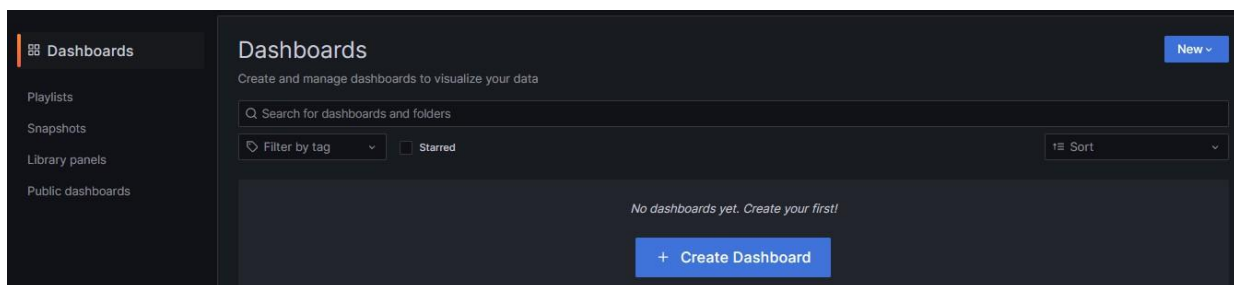
Step 48 : Copy Ip with port no.3000 and paste in chrome u get grafana page after u get the page click on data sources



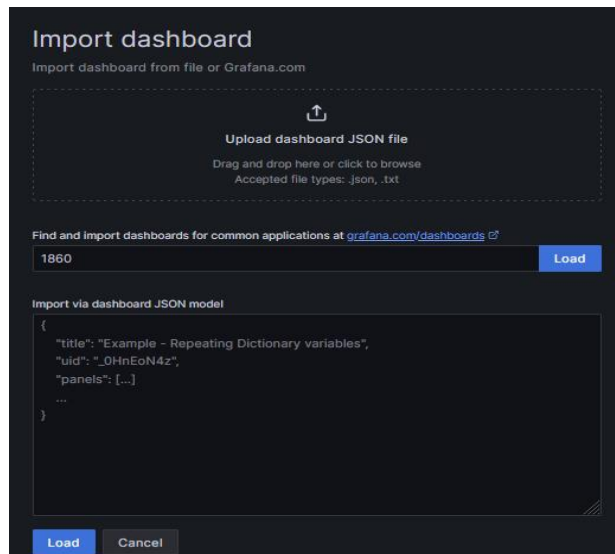
Step 49 : And Click on Prometheus add the Prometheus URL and click on save test



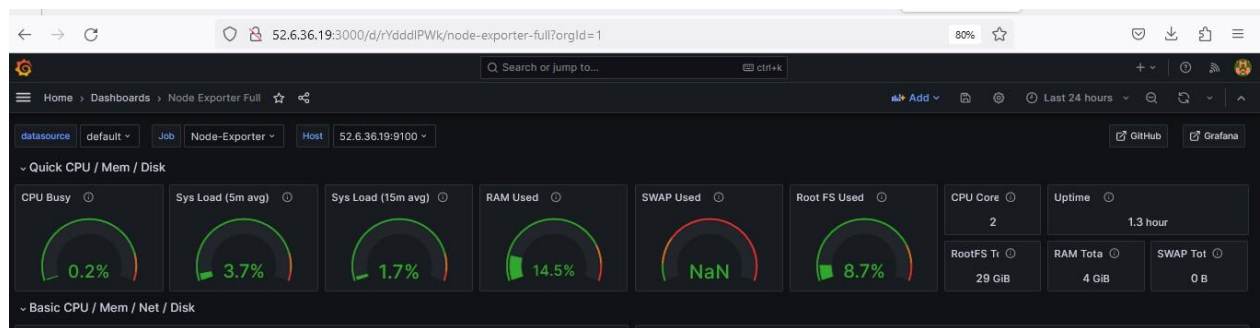
Step 50 : Click on new and click on Import dashboard



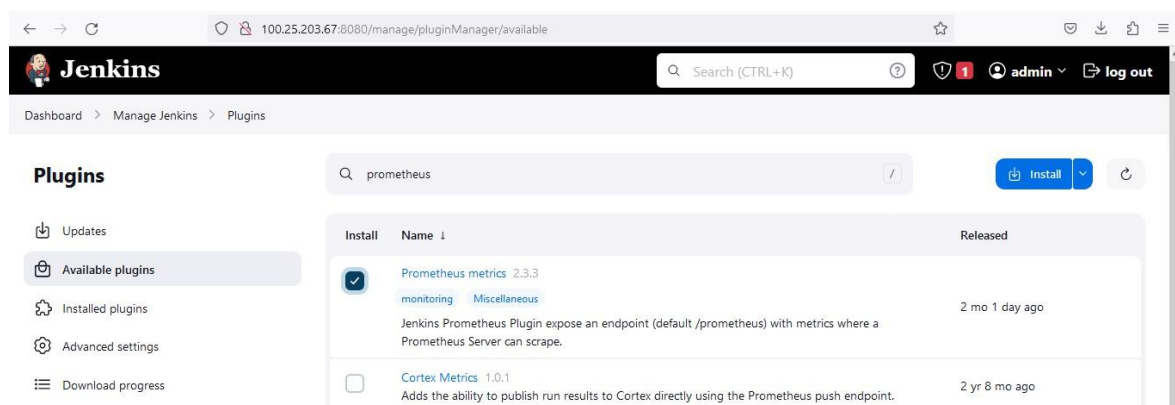
Step 51 : Enter no.1860 and click on load



Step 52 : U get monitoring page of node exporter



Step 53 : Go to Jenkins download Prometheus plugin



Step 54 : Add the Jenkins job in Prometheus.yaml (sudo vim /etc/prometheus/prometheus.yml)

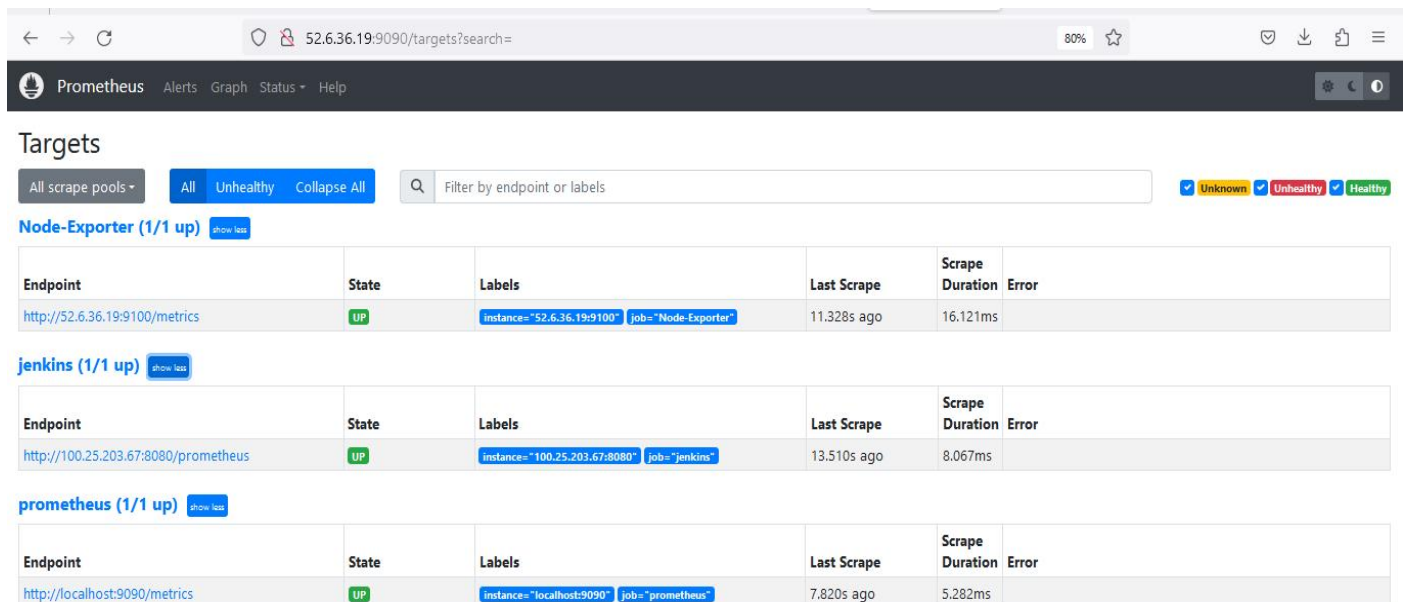
```
ubuntu@ip-172-31-51-59:~$ sudo vim /etc/prometheus/prometheus.yml
```

Step 55 : Add this Jenkins job in Prometheus.yml

```
- job_name: 'jenkins'
  metrics_path: '/prometheus'
  static_configs:
    - targets: ['100.25.203.67:8080']
```

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

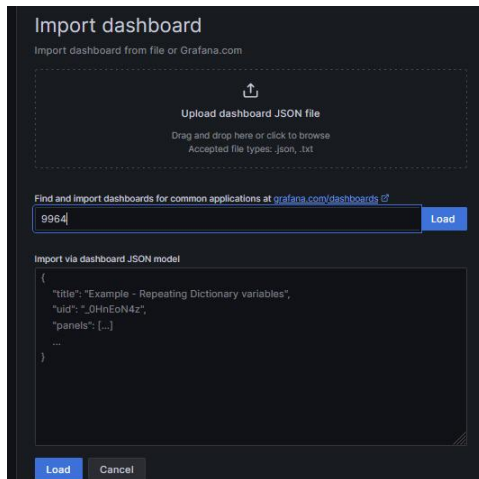
Step 56 : Go to Prometheus/Status/target u get Jenkins info



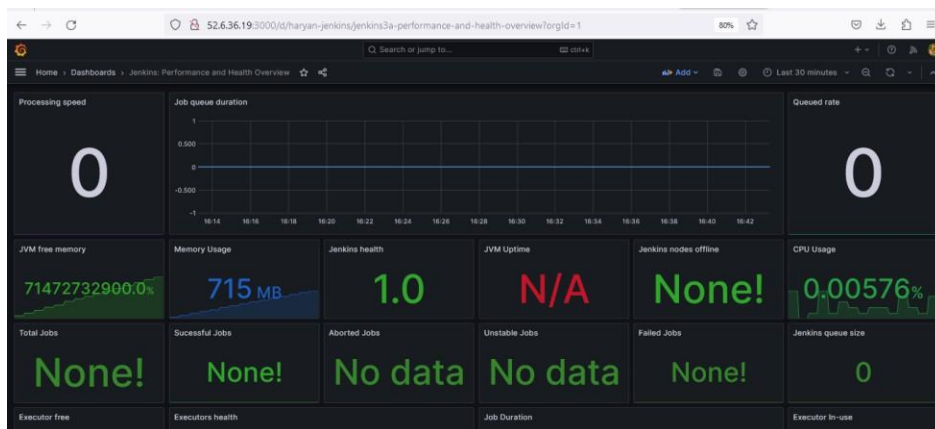
The screenshot shows the Prometheus web interface at the 'Targets' page. The URL bar shows '52.6.36.19:9090/targets?search='. The page has a navigation bar with 'Prometheus', 'Alerts', 'Graph', 'Status', and 'Help'. Below the navigation bar, there are filters for 'All scrape pools', 'All', 'Unhealthy', and 'Collapse All'. A search bar is present with the text 'Filter by endpoint or labels'. To the right, there are status indicators: 'Unknown', 'Unhealthy', and 'Healthy'. The main content area lists three scrape pools:

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
Node-Exporter (1/1 up) show less					
http://52.6.36.19:9100/metrics	UP	instance="52.6.36.19:9100" job="Node-Exporter"	11.328s ago	16.121ms	
jenkins (1/1 up) show less					
http://100.25.203.67:8080/prometheus	UP	instance="100.25.203.67:8080" job="jenkins"	13.510s ago	8.067ms	
prometheus (1/1 up) show less					
http://localhost:9090/metrics	UP	instance="localhost:9090" job="prometheus"	7.820s ago	5.282ms	

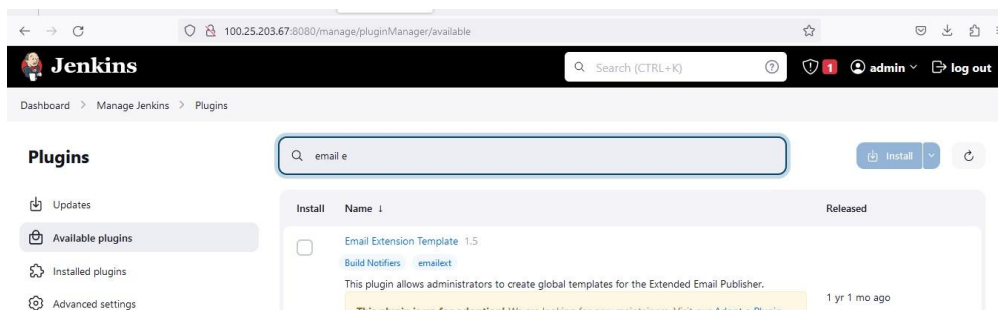
Step 57 : Click on Import dashboard and enter no 9964 and click load



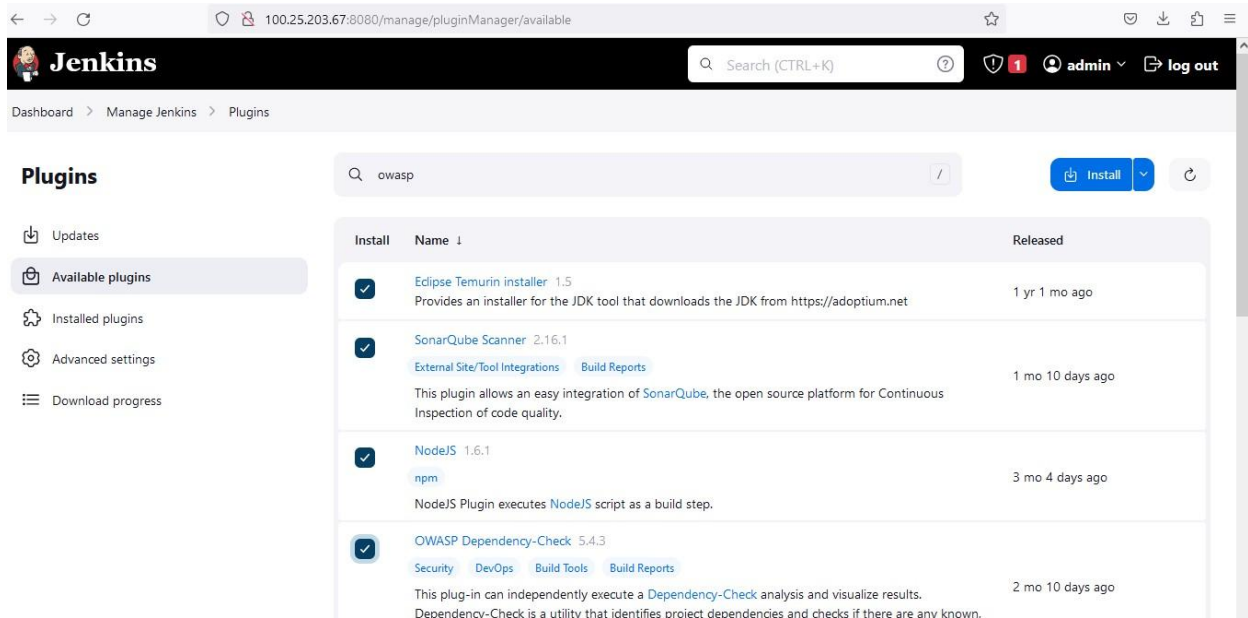
Step 58 : Now you will see the Detailed overview of Jenkins



Step 59 : Download the email plugin in jenkins



Step 61 : Download plugins like Eclipse Temurin Installer, sonarqube scanner, Node js, OWASP Dependency check



The screenshot shows the Jenkins web interface at the URL `100.25.203.67:8080/manage/pluginManager/available`. The page title is "Jenkins" and the user is logged in as "admin". The breadcrumb navigation shows "Dashboard > Manage Jenkins > Plugins".

The "Plugins" section is active, and a search for "owasp" has been performed. The search results show four plugins, all of which are already installed (indicated by a checkmark in the "Install" column).

Install	Name	Released
<input checked="" type="checkbox"/>	Eclipse Temurin installer 1.5 Provides an installer for the JDK tool that downloads the JDK from https://adoptium.net	1 yr 1 mo ago
<input checked="" type="checkbox"/>	SonarQube Scanner 2.16.1 External Site/Tool Integrations Build Reports This plugin allows an easy integration of SonarQube , the open source platform for Continuous Inspection of code quality.	1 mo 10 days ago
<input checked="" type="checkbox"/>	NodeJS 1.6.1 npm NodeJS Plugin executes NodeJS script as a build step.	3 mo 4 days ago
<input checked="" type="checkbox"/>	OWASP Dependency-Check 5.4.3 Security DevOps Build Tools Build Reports This plug-in can independently execute a Dependency-Check analysis and visualize results. Dependency-Check is a utility that identifies project dependencies and checks if there are any known.	2 mo 10 days ago

Step 62 : Download docker base all plugin (Total 5 Plugin)

The screenshot shows the Jenkins 'Plugins' page. The search bar contains 'docker'. The left sidebar shows 'Available plugins' is selected. The main table lists the following plugins:

Install	Name	Released
<input type="checkbox"/>	Docker 1.5 Cloud Providers Cluster Management docker This plugin integrates Jenkins with Docker	2 mo 15 days ago
<input type="checkbox"/>	Docker Commons 439.va_3cb_0a_6a_fb_29 Library plugins (for use by other plugins) docker Provides the common shared functionality for various Docker-related plugins.	4 mo 11 days ago
<input type="checkbox"/>	Docker Pipeline 572.v950f58993843 pipeline DevOps Deployment docker Build and use Docker containers from pipelines.	3 mo 9 days ago
<input type="checkbox"/>	Docker API 3.3.1-79.v20b_53427e041 Library plugins (for use by other plugins) docker	

Step 63 : Add the sonar server info in Jenkins tools page

The screenshot shows the 'Tools' page in Jenkins, specifically the 'SonarQube Scanner' section. The 'Name' field is set to 'sonar-scanner'. The 'Install automatically' checkbox is checked. Under the 'Install from Maven Central' section, the 'Version' dropdown is set to 'SonarQube Scanner 5.0.1.3006'.

Step 64 : Add the jdk info in Jenkins tools page

The screenshot shows the 'Tools' page in Jenkins, specifically the 'JDK' section. The 'Name' field is set to 'jdk17'. The 'Install automatically' checkbox is checked. Under the 'Install from adoptium.net' section, the 'Version' dropdown is set to 'jdk-17.0.8.1+1'. There is an 'Add Installer' button at the bottom.

Step 65 : Add the nodejs info in Jenkins tools page

Dashboard > Manage Jenkins > Tools

NodeJS

Name

node16

☒ Install automatically ?

Install from nodejs.org

Version

NodeJS 16.2.0

For the underlying architecture, if available, force the installation of the 32bit package. Otherwise the build will fail

☐ Force 32bit architecture

Step 66 : Add the dependency info in Jenkins tools page

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

Step 67 : Add the docker info in Jenkins tools page

Dashboard > Manage Jenkins > Tools

Docker installations

Add Docker

Docker

Name

Docker

☒ Install automatically ?

Download from docker.com

Docker version ?

latest

Add Installer

Step 68 : Go to Jenkins/system and integrate sonarqube with jenkins

The screenshot shows the Jenkins 'System' configuration page. At the top, there's a breadcrumb trail: 'Dashboard > Manage Jenkins > System'. Below this, a note states: 'If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.' There is an unchecked checkbox labeled 'Environment variables'. Under the heading 'SonarQube installations', it says 'List of SonarQube installations'. A form is visible with the following fields: 'Name' (containing 'sonar-server'), 'Server URL' (with a default value 'http://localhost:9000' and a custom value 'http://100.25.303.67:9000'), and 'Server authentication token' (with a default value 'sonar-server' and a custom value 'sonar-server'). There are '+ Add' and '- Remove' buttons for the list. At the bottom, there are 'Save' and 'Apply' buttons.

Step 69 : Then we need to install trivy, so we first of all create trivy shell script file

```
ubuntu@ip-172-31-49-232:~$ vi trivy.sh
```

Step 70 : Add the trivy downloaded command in this file

```
sudo apt-get install wget apt-transport-https gnupg lsb-release -y

wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | gpg --dearmor | sudo tee
/usr/share/keyrings/trivy.gpg > /dev/null

echo "deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasecurity.github.io/trivy-repo/deb $(lsb_release -
sc) main" | sudo tee -a /etc/apt/sources.list.d/trivy.list

sudo apt-get update

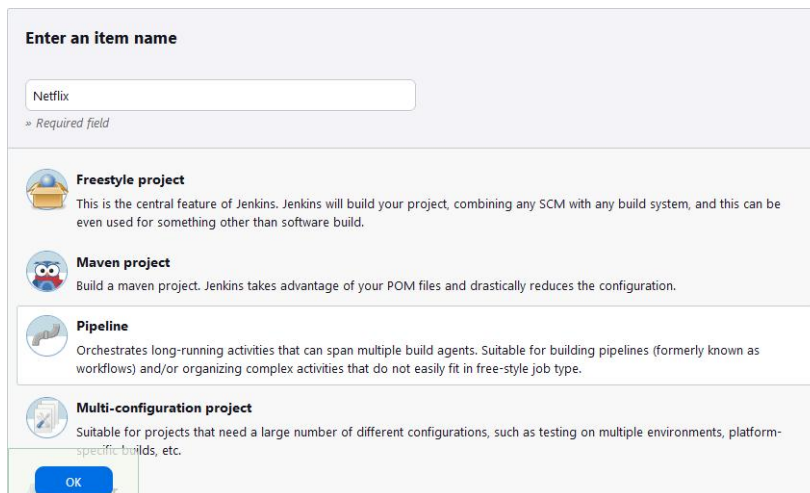
sudo apt-get install trivy -y
```

The screenshot shows a terminal window with the following commands and output:
ubuntu@ip-172-31-49-232: ~
sudo apt-get install wget apt-transport-https gnupg lsb-release -y
wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | gpg --dearmor | sudo tee /usr/share/keyrings/trivy.gpg > /dev/null
echo "deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasecurity.github.io/trivy-repo/deb \$(lsb_release -sc) main" | sudo tee -a /etc/apt/sources.list.d/trivy.l
ist
sudo apt-get update
sudo apt-get install trivy -y

Step 71 : Give permission to the file and run the script file (`sudo chmod 777 trivy.sh & ./trivy.sh`)

```
ubuntu@ip-172-31-49-232:~$ sudo chmod 777 trivy.sh
ubuntu@ip-172-31-49-232:~$ ./trivy.sh
```

Step 70 : After all this step create one pipeline name of Netflix



The image shows the 'Enter an item name' dialog box in Jenkins. The text 'Netflix' is entered into the input field. Below the input field, there is a link that says '» Required field'. Below the input field, there are four project type options, each with an icon and a description:

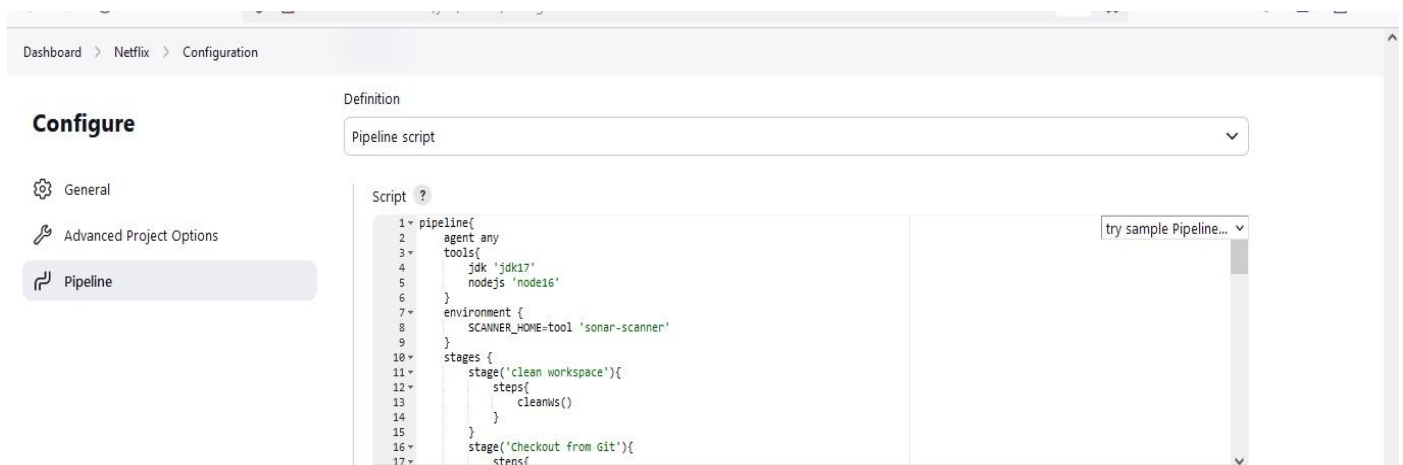
- Freestyle project**: This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
- Maven project**: Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.
- Pipeline**: Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- Multi-configuration project**: Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

At the bottom left, there is a blue 'OK' button.

Step 71 : Add the script in the pipeline

(Pipeline script stored in github :

https://github.com/JanhaviBagul315/Netflix_clone/blob/main/Pipeline%20Script)



The image is a screenshot of the Jenkins web interface showing the configuration for a pipeline named 'Netflix'. The breadcrumb navigation at the top reads 'Dashboard > Netflix > Configuration'. On the left, under the 'Configure' heading, there are three tabs: 'General', 'Advanced Project Options', and 'Pipeline'. The 'Pipeline' tab is selected and highlighted. The main area shows the 'Definition' dropdown set to 'Pipeline script'. Below this, there is a 'Script' section with a text area containing a Jenkins pipeline script. The script is as follows:

```
1 pipeline{
2   agent any
3   tools{
4     jdk 'jdk17'
5     nodejs 'node16'
6   }
7   environment {
8     SCANNER_HOME=tool 'sonar-scanner'
9   }
10  stages {
11    stage('clean workspace'){
12      steps{
13        cleanws()
14      }
15    }
16    stage('Checkout from Git'){
17      steps{
```

On the right side of the script text area, there is a button that says 'try sample Pipeline...'.

Step 72 : Run the pipeline



Step 73. Pipeline done and all stages are completed successfully

Stage View



Step 73 : Copy the Ip with port no u get the Netflix page successfully

