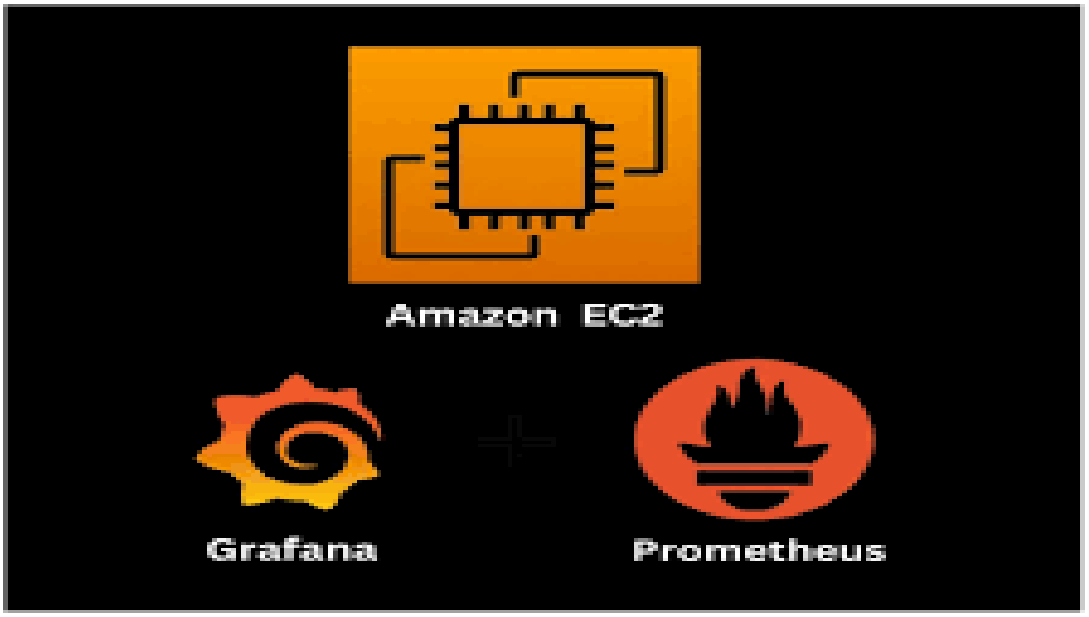
cprime

# Prometheus, Node Exporter, and Grafana Setup on AWS EC2




**NAME : T MANOJ**

**EMPID : LYAKE2KHS**

# Prometheus and Grafana Setup for AWS EC2

## Introduction

In modern cloud environments, monitoring is crucial for ensuring system reliability, performance, and availability. Prometheus, an open-source monitoring and alerting toolkit, is widely used for collecting system metrics in real-time. It utilizes a pull-based mechanism to scrape metrics from various endpoints. Node Exporter, a lightweight exporter, helps gather system-level metrics from Linux machines. Grafana, a visualization tool, transforms collected data into meaningful dashboards for analysis.

This document provides a step-by-step guide on setting up Prometheus, Node Exporter, and Grafana on AWS EC2 instances. The setup enables real-time system monitoring, visualization, and alerting for efficient infrastructure management.

## Overview

This setup involves three main components:

1. **Prometheus** - The core monitoring system that collects and stores metrics.
2. **Node Exporter** - A data exporter that allows Prometheus to collect machine-level metrics.
3. **Grafana** - A visualization tool for creating interactive monitoring dashboards.

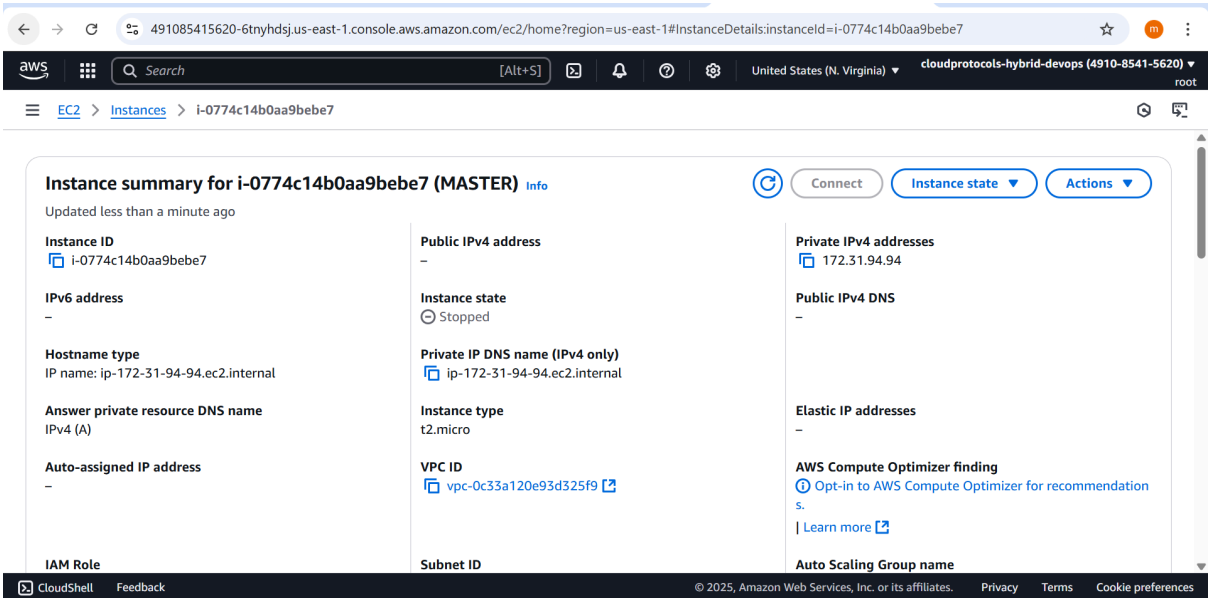
The following steps will guide you through the installation and configuration of Prometheus, Node Exporter, and Grafana:

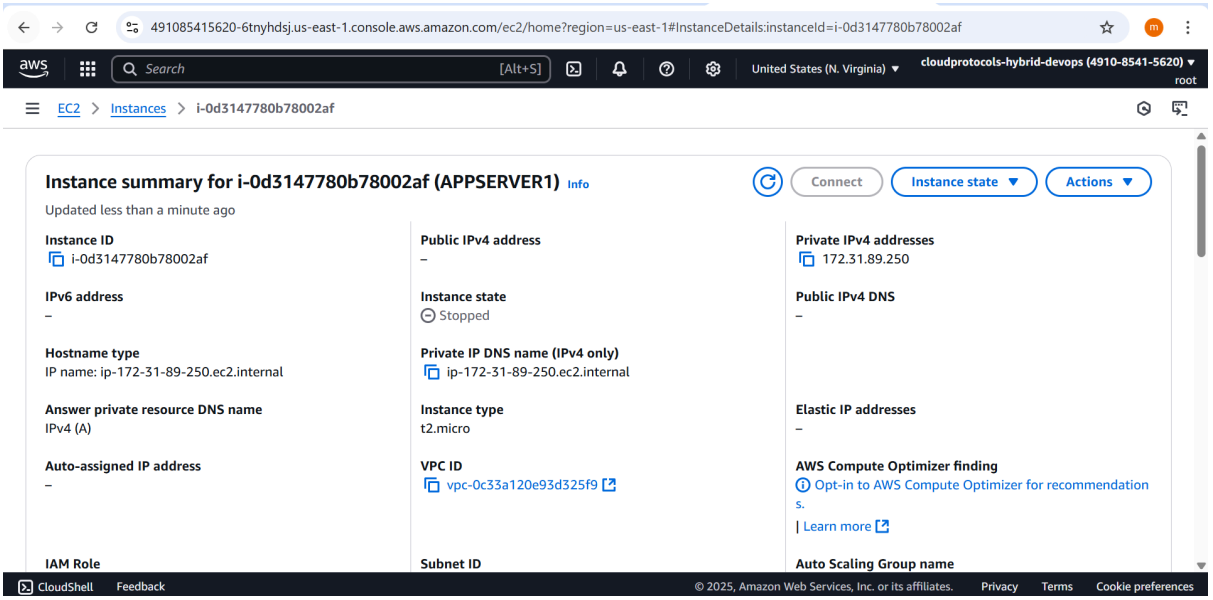
1. Launch an EC2 Instance for Prometheus - Install and configure Prometheus to collect metrics.
2. Install Node Exporter on Agent EC2 - Deploy Node Exporter on separate instances to gather system metrics.
3. Connect Prometheus to Node Exporter - Configure Prometheus to pull metrics from Node Exporter.
4. Install Grafana on Prometheus EC2 Instance - Deploy Grafana for visualization.
5. Configure Prometheus as Data Source in Grafana - Connect Prometheus to Grafana for dashboard integration.
6. Import Grafana Dashboard - Load pre-built dashboards for easier monitoring.
7. Setup Alerts in Grafana - Configure alerting mechanisms for proactive monitoring.

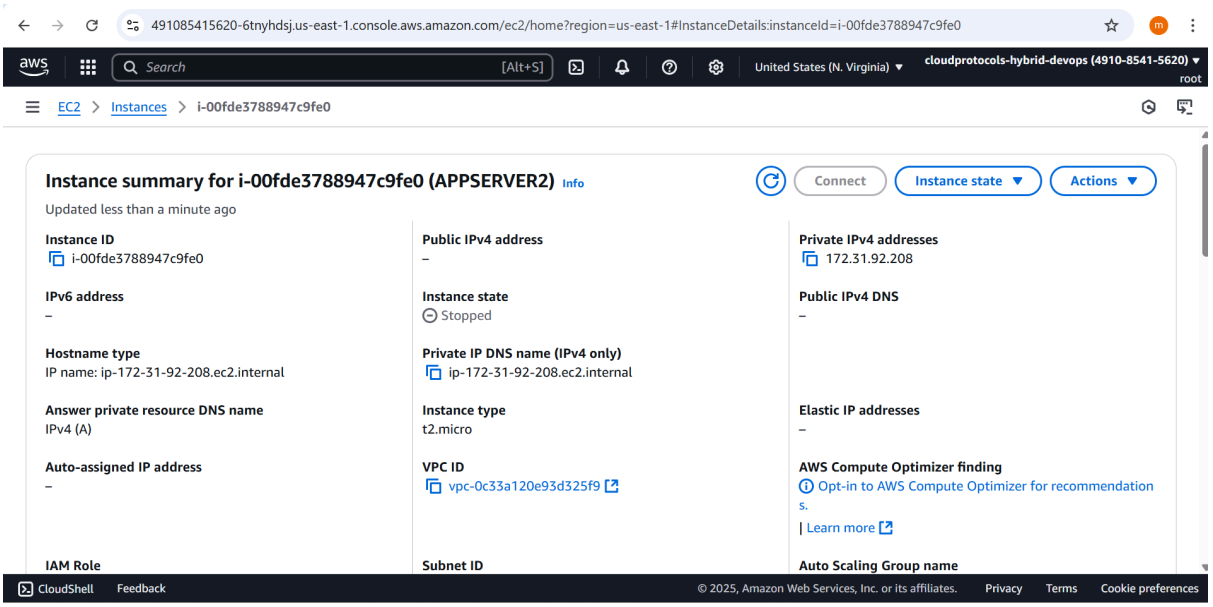
# 1. Launch EC2 Instance for Prometheus Server

## 1.1. Create EC2 Instance

1. Login to AWS Management Console
2. Navigate to EC2 Dashboard
3. Click "Launch Instance"
4. Select "Amazon Linux 2023" AMI
5. Choose instance type:
    - t2.micro for testing environments
6. Add storage (default 8GB is sufficient for initial setup)
7. Add tags:
    - Key: Name
    - Value: Prometheus-Server
8. Configure Security Group:
    - Create new security group: "prometheus-sg"
    - Add the following inbound rules:
        - SSH (TCP/22) from your IP
        - Custom TCP (9090) from your IP for Prometheus
        - Custom TCP (3000) from your IP for Grafana
        - Custom TCP (9093) from your IP for Alertmanager (optional)
9. Review and Launch
10. Create a new key pair or select an existing one
11. Launch instance






## 1.2. Connect to EC2 Instance

1. Open terminal (Linux/Mac) or SSH client (Windows)

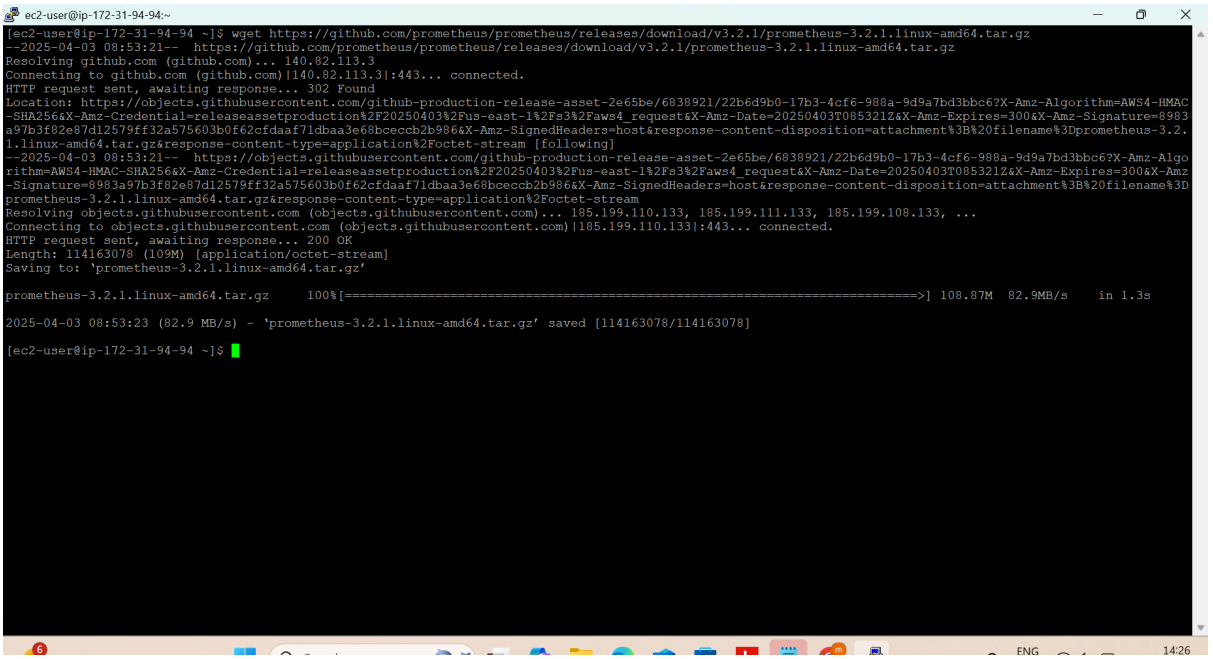
```
Use SSH to connect:
 ssh -i your-key.pem ec2-user@your-instance-public-ip
```

```
Update the system:
sudo yum update -y
```

## Step 2: Install Prometheus

Download Prometheus:

```
wgethttps://github.com/prometheus/prometheus/releases/download/v2.41.0/prometheus-2.41.0.linux-amd64.tar.gz
```




Extract the archive:

```
tar -xvzf prometheus-2.41.0.linux-amd64.tar.gz
```


Move into the Prometheus directory:

```
cd prometheus-2.41.0.linux-amd64
```

Move Prometheus binaries to `/usr/local/bin/`:

```
sudo cp prometheus /usr/local/bin/
```

```
sudo cp promtool /usr/local/bin/
```

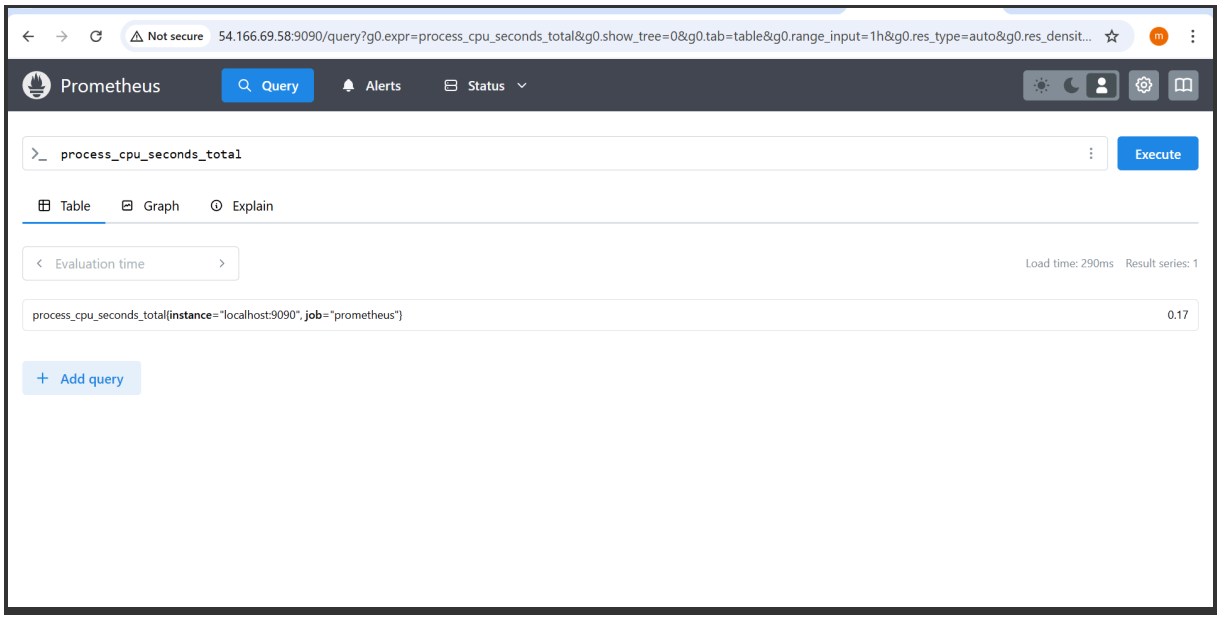
```
[ec2-user@ip-172-31-94-94 prometheus-3.2.1.linux-amd64]$ ll
total 285504
-rw-r--r--. 1 ec2-user ec2-user     11357 Feb 26 09:37 LICENSE
-rw-r--r--. 1 ec2-user ec2-user      3773 Feb 26 09:37 NOTICE
-rwxr-xr-x. 1 ec2-user ec2-user 150306663 Feb 26 09:21 prometheus
-rw-r--r--. 1 ec2-user ec2-user       934 Feb 26 09:37 prometheus.yml
-rwxr-xr-x. 1 ec2-user ec2-user 142027433 Feb 26 09:21 promtool
[ec2-user@ip-172-31-94-94 prometheus-3.2.1.linux-amd64]$
```

Start Prometheus in the background:

```
prometheus --config.file=/etc/prometheus/prometheus.yml &
```

1. Access Prometheus dashboard:

- Open browser and visit: `http://<Prometheus_EC2_IP>:9090`




# 2. Install Node Exporter on Agent EC2

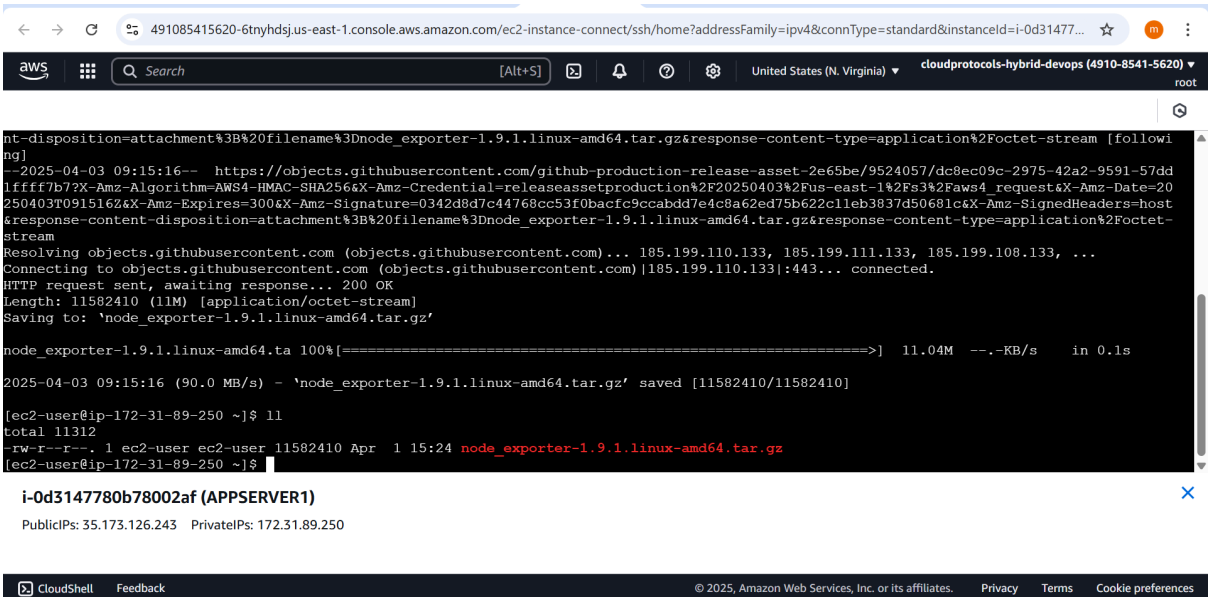
## Step 1: Launch a New EC2 Instance

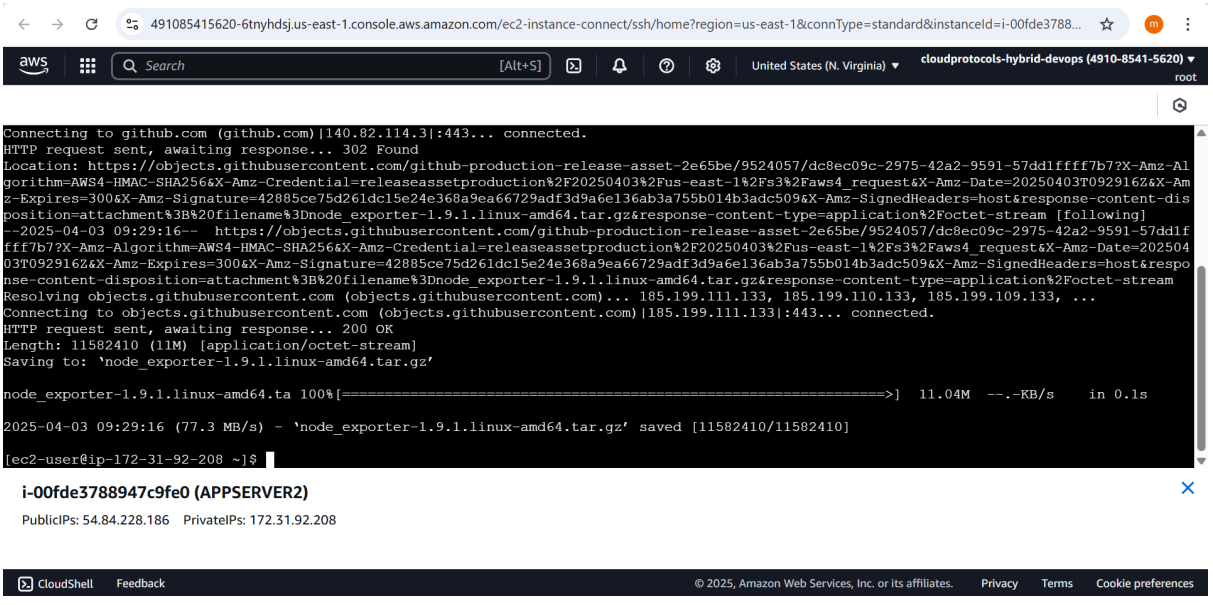
1. Go to AWS EC2 dashboard and launch another **Amazon Linux 2** instance.
2. Configure security group to allow **Port 9100** (for Node Exporter).
3. Launch the instance and connect via SSH.

## Step 2: Install Node Exporter

Download Node Exporter:

```
wgethttps://github.com/prometheus/node_exporter/releases/download/v1.5.0/node_exporter-1.5.0.linux-amd64.tar.gz
```

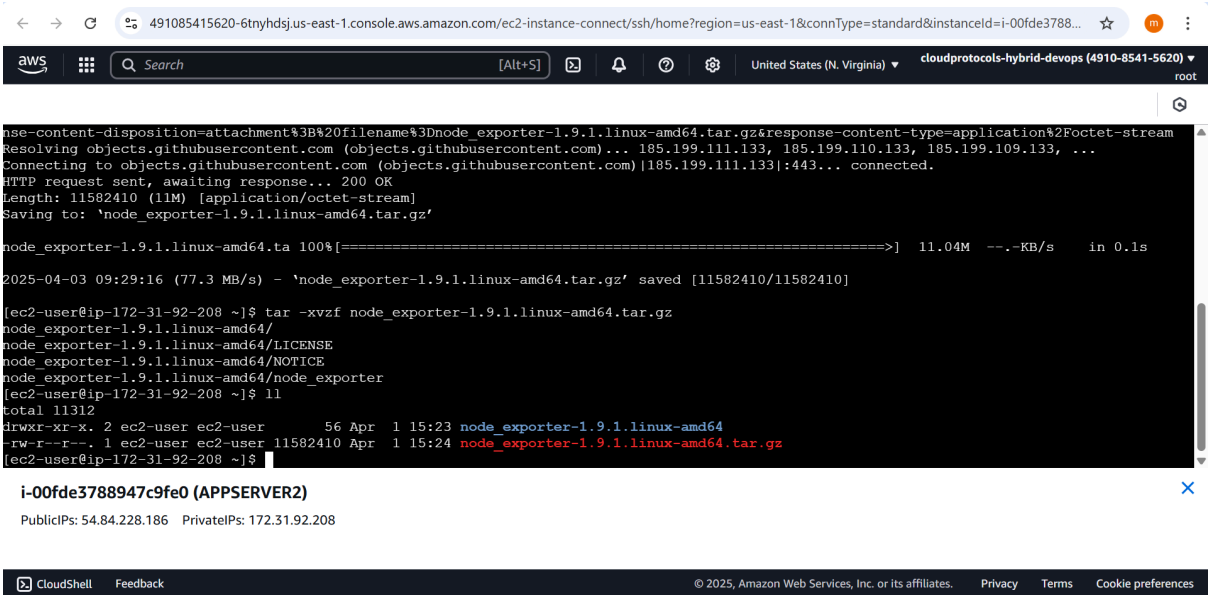



Extract the archive:

```
tar -xvzf node_exporter-1.5.0.linux-amd64.tar.gz
```

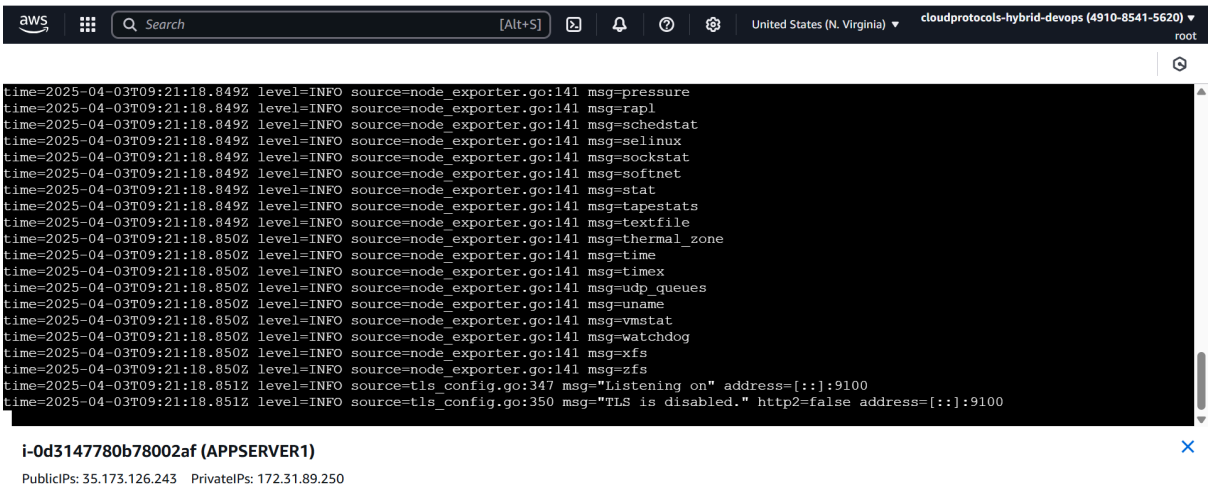
```
[ec2-user@ip-172-31-89-250 ~]$ tar -xvzf node_exporter-1.9.1.linux-amd64.tar.gz
node_exporter-1.9.1.linux-amd64/
node_exporter-1.9.1.linux-amd64/LICENSE
node_exporter-1.9.1.linux-amd64/NOTICE
node_exporter-1.9.1.linux-amd64/node_exporter
[ec2-user@ip-172-31-89-250 ~]$
```

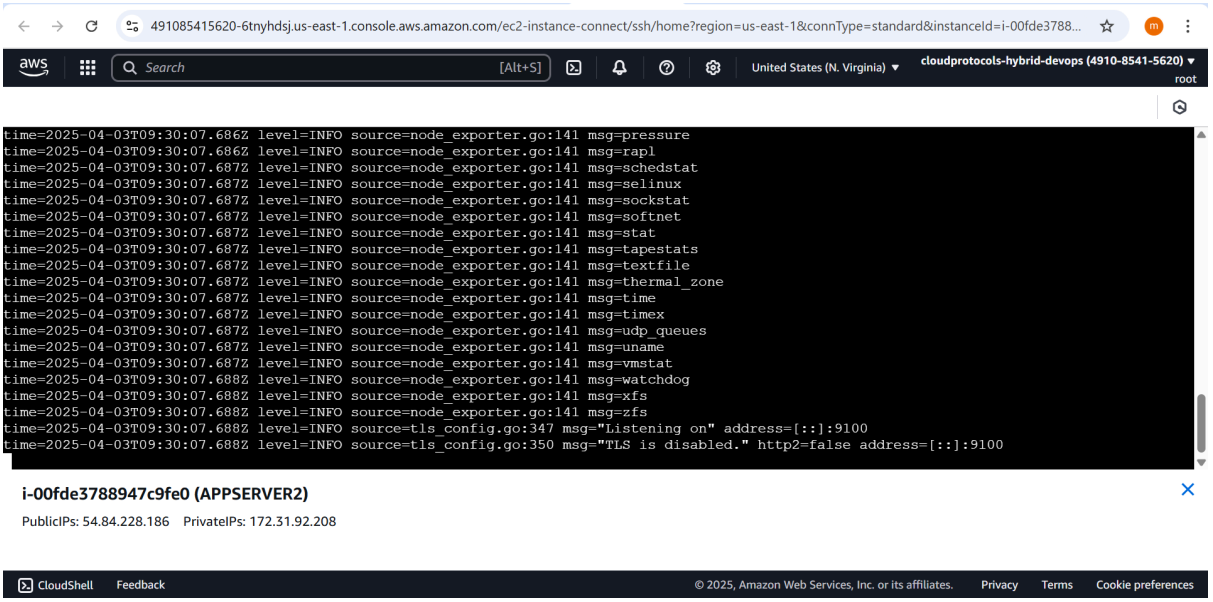



Start Node Exporter in the background:

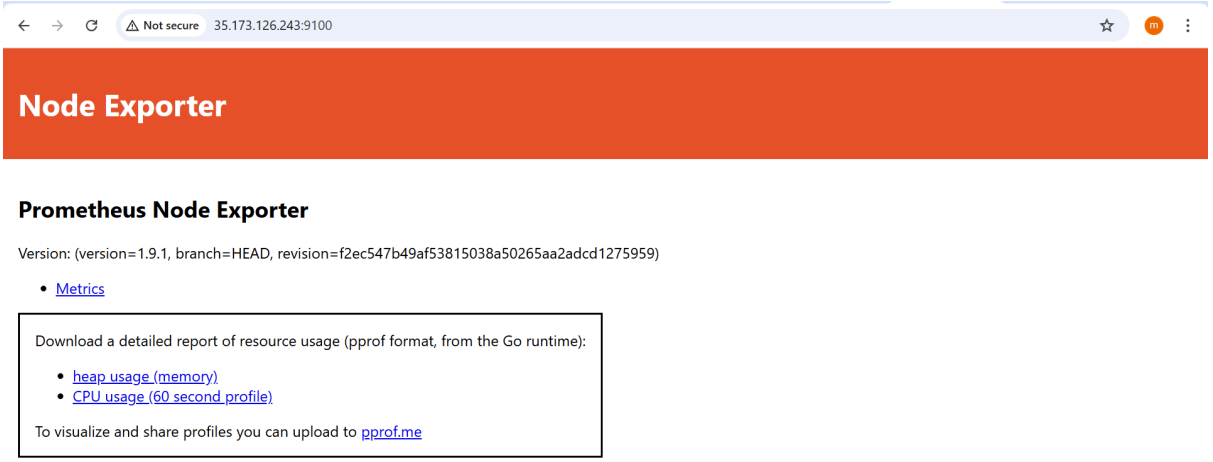
```
./node_exporter &
```

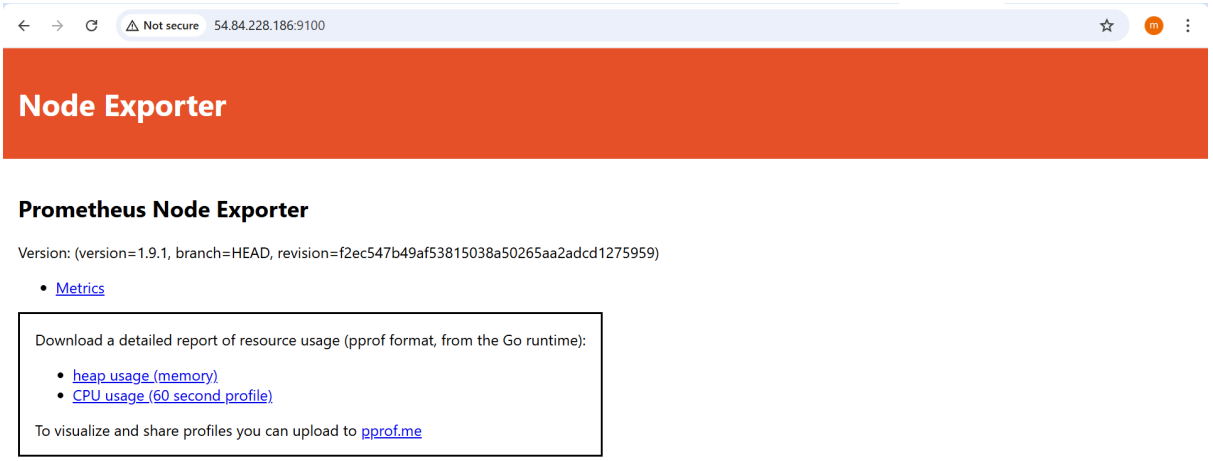
```
[ec2-user@ip-172-31-89-250 node_exporter-1.9.1.linux-amd64]$ ll
total 21700
-rw-r--r--. 1 ec2-user ec2-user 11357 Apr 1 15:23 LICENSE
-rw-r--r--. 1 ec2-user ec2-user 463 Apr 1 15:23 NOTICE
-rwxr-xr-x. 1 ec2-user ec2-user 22204245 Apr 1 15:19 node_exporter
[ec2-user@ip-172-31-89-250 node_exporter-1.9.1.linux-amd64]$
```






1. Verify Node Exporter:
    - Open browser and visit: `http://<Agent_EC2_IP>:9100`





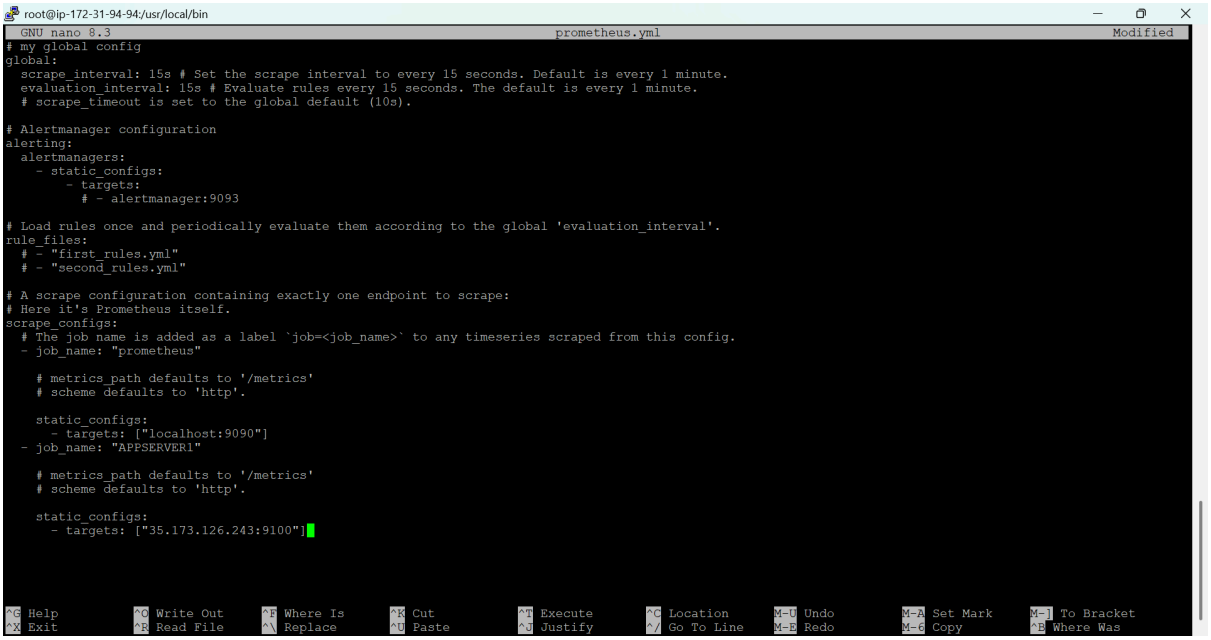

# 3. Connect Prometheus to Node Exporter

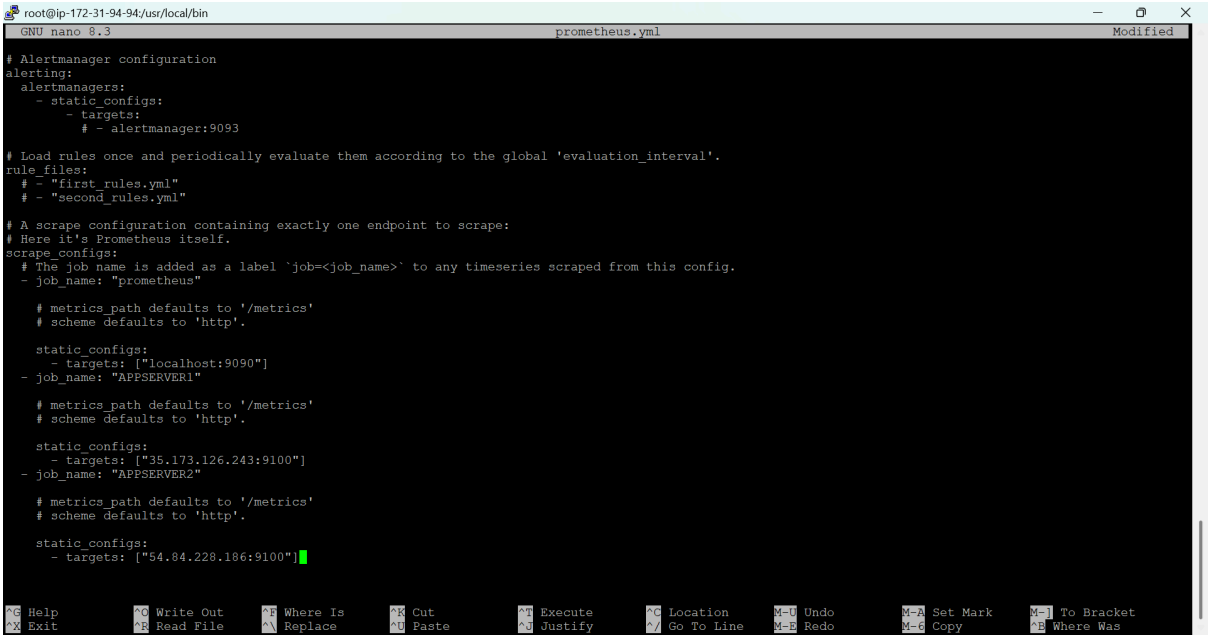
## Step 1: Edit Prometheus Configuration

1. SSH into the Prometheus EC2 instance.

Edit Prometheus configuration file:

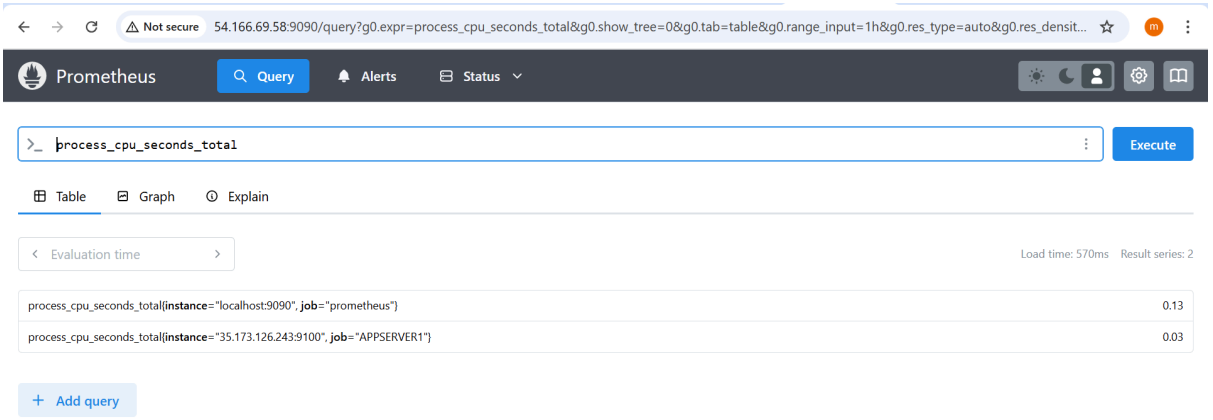
```
 sudo vi /etc/prometheus/prometheus.yml
Add Node Exporter target:

 scrape_configs:
  - job_name: 'node_exporter'
    static_configs:
      - targets: ['<AGENT_IP>:9100']
```

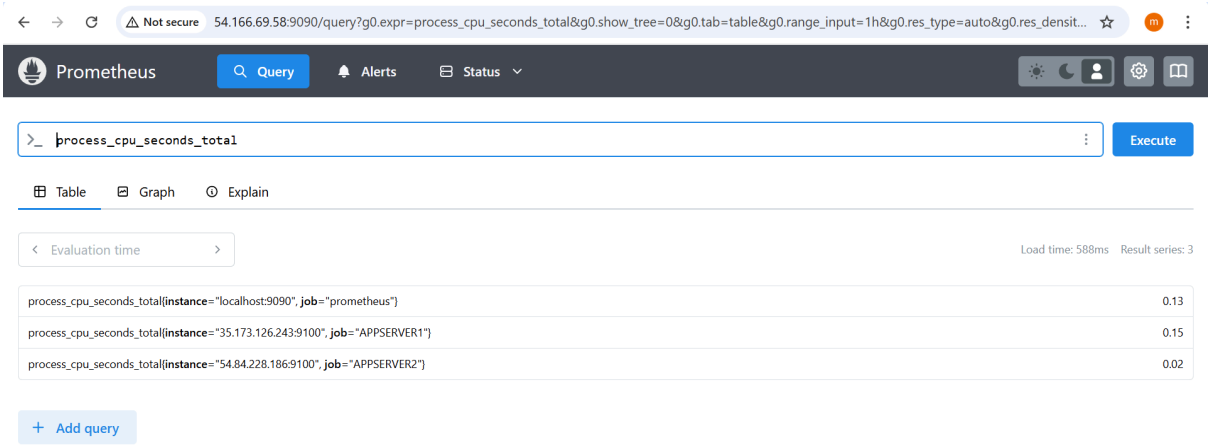


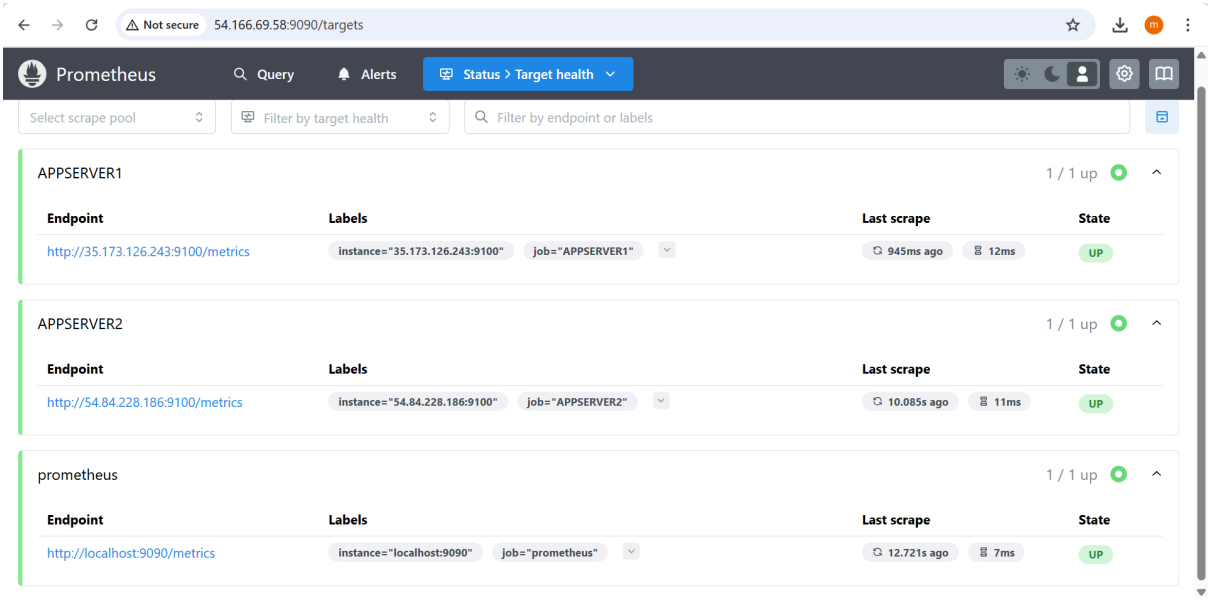




```
prometheus --config.file=/etc/prometheus/prometheus.yml &
```

2. Check the output on the Prometheus dashboard:
   `http://<Prometheus_EC2_IP>:9090`


Not secure
54.166.69.58:9090/query?g0.expr=process_cpu_seconds_total&g0.show_tree=0&g0.tab=table&g0.range_input=1h&g0.res_type=auto&g0.res_densit...
Prometheus
Query
Alerts
Status
process_cpu_seconds_total
Execute
Table
Graph
Explain
Evaluation time
Load time: 570ms
Result series: 2
process_cpu_seconds_total{instance="localhost:9090", job="prometheus"}
0.13
process_cpu_seconds_total{instance="35.173.126.243:9100", job="APPSERVER1"}
0.03
Add query


Not secure
54.166.69.58:9090/query?g0.expr=process_cpu_seconds_total&g0.show_tree=0&g0.tab=table&g0.range_input=1h&g0.res_type=auto&g0.res_densit...
Prometheus
Query
Alerts
Status
process_cpu_seconds_total
Execute
Table
Graph
Explain
Evaluation time
Load time: 588ms
Result series: 3
process_cpu_seconds_total{instance="localhost:9090", job="prometheus"}
0.13
process_cpu_seconds_total{instance="35.173.126.243:9100", job="APPSERVER1"}
0.15
process_cpu_seconds_total{instance="54.84.228.186:9100", job="APPSERVER2"}
0.02
Add query

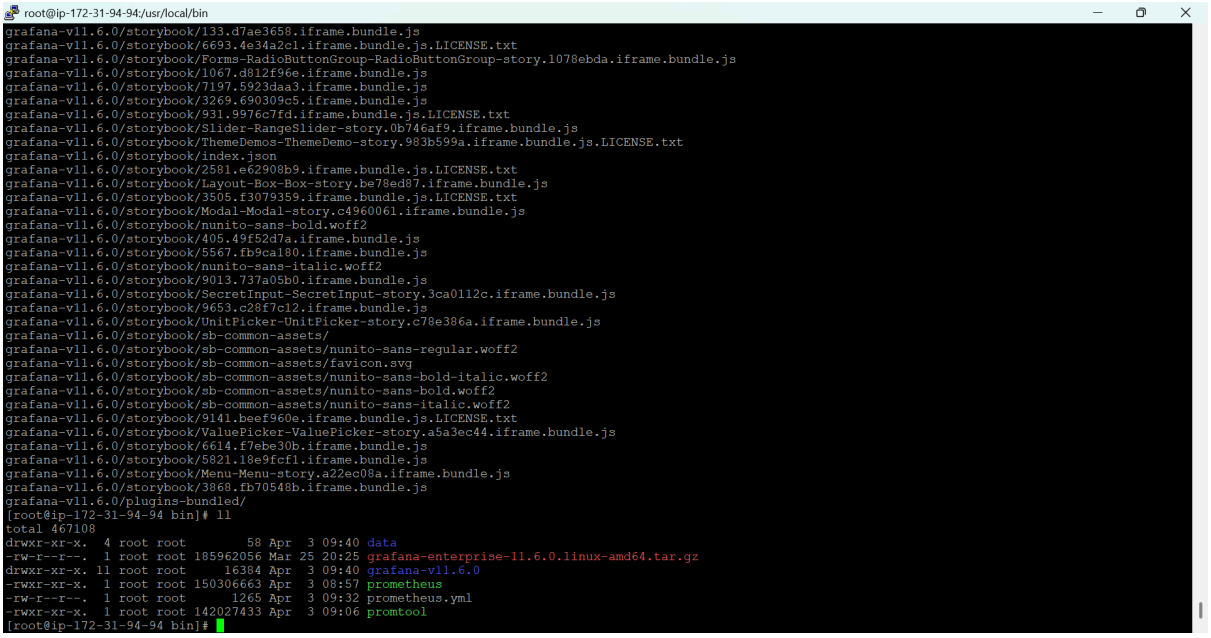



# 4. Install Grafana on Prometheus EC2 Instance

## Step 1: Install Grafana

Download Grafana:

```
 Wget
https://dl.grafana.com/oss/release/grafana-10.0.0.linux-amd64.tar.gz
```




Extract the archive:

```
tar -xvzf grafana-10.0.0.linux-amd64.tar.gz
```

```
[root@ip-172-31-94-94 bin]# cd grafana-v11.6.0/
[root@ip-172-31-94-94 grafana-v11.6.0]# ll
total 112
-rw-r--r--.  1 root root  6536 Mar 21 14:00 Dockerfile
-rw-r--r--.  1 root root 12155 Mar 21 14:02 LICENSE
-rw-r--r--.  1 root root   105 Mar 21 14:00 NOTICE.md
-rw-r--r--.  1 root root  3269 Mar 21 14:00 README.md
-rw-r--r--.  1 root root     8 Mar 21 14:02 VERSION
drwxr-xr-x.  2 root root    62 Mar 21 14:06 bin
drwxr-xr-x.  3 root root   107 Mar 21 14:00 conf
drwxr-xr-x.  3 root root    21 Apr  3 09:40 docs
drwxr-xr-x.  2 root root 16384 Mar 21 14:05 npm-artifacts
drwxr-xr-x.  6 root root    58 Apr  3 09:40 packaging
drwxr-xr-x.  2 root root     6 Mar 21 14:03 plugins-bundled
drwxr-xr-x. 17 root root 16384 Mar 21 14:07 public
drwxr-xr-x.  8 root root 32768 Mar 21 14:05 storybook
drwxr-xr-x.  2 root root    26 Apr  3 09:40 tools
[root@ip-172-31-94-94 grafana-v11.6.0]#
```

Move into the Grafana directory:

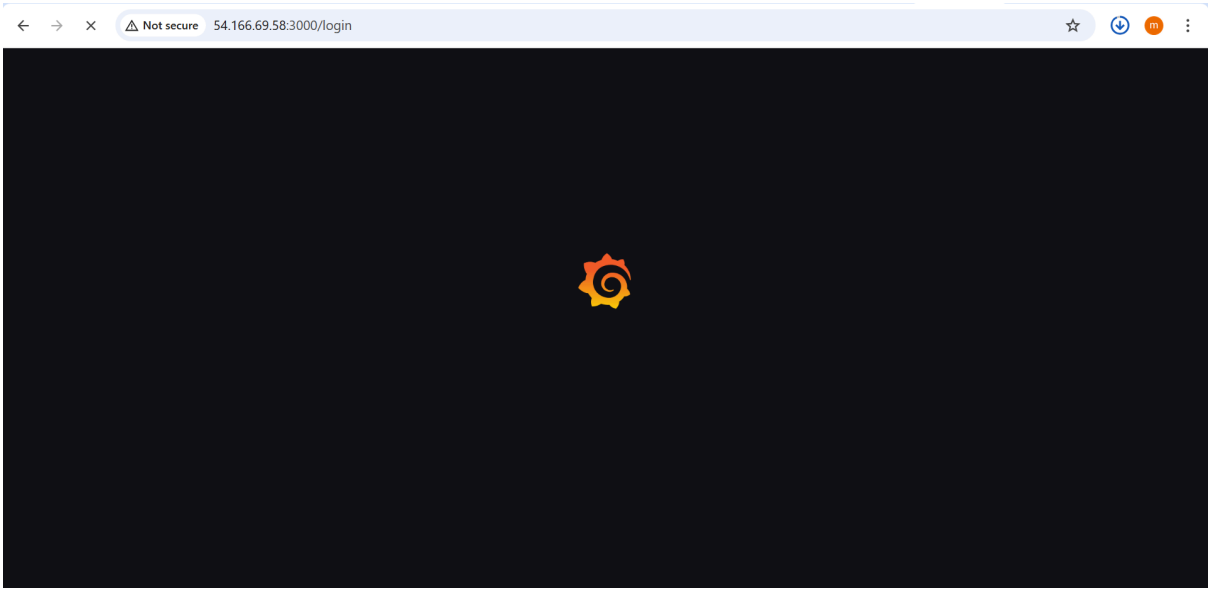
```
cd grafana-11.0.0
```

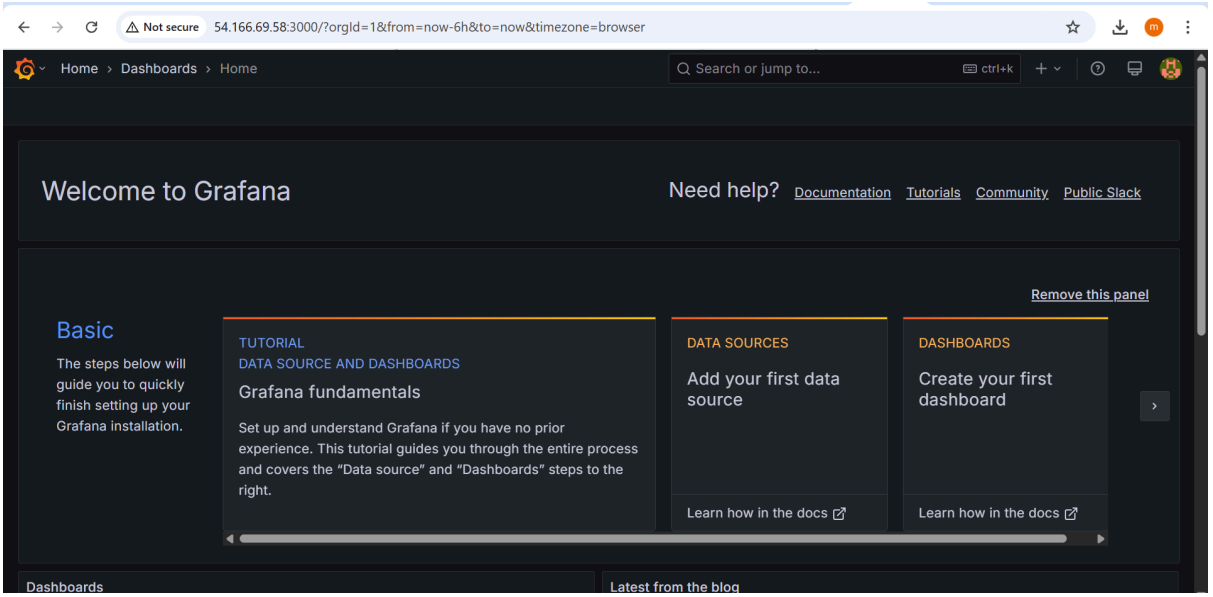
Start Grafana in the background:

```
./bin/grafana-server &
```

```
root@ip-172-31-94-94:/usr/local/bin/grafana-v11.6.0/bin
INFO [04-03|09:48:40] Applying new configuration to Alertmanager logger=ngalert.notifier.alertmanager org=1 configHash=d2c56faca6af2a5772ff4253222f7386
INFO [04-03|09:48:40] Using simple database alert instance store logger=ngalert
INFO [04-03|09:48:40] Using sync state persister             logger=ngalert.state.manager.persist
INFO [04-03|09:48:40] Running in alternative execution of Error/NoData mode logger=ngalert.state.manager
INFO [04-03|09:48:40] registering usage stat providers       logger=infra.usagestats.collector usageStatsProvidersLen=2
INFO [04-03|09:48:40] HTTP Server Listen                     logger=http.server address=[::]:3000 protocol=http subUrl= socket=
INFO [04-03|09:48:40] Warming state cache for startup        logger=ngalert.state.manager
ERROR[04-03|09:48:40] Failed to get renderer plugin sources  logger=renderer.manager error="failed to open plugins path"
INFO [04-03|09:48:40] Storage starting                       logger=grafanaStorageLogger
INFO [04-03|09:48:40] Installing plugin                      logger=plugin.backgroundinstaller pluginId=grafana-pyroscope-app version=
INFO [04-03|09:48:40] Starting MultiOrg Alertmanager         logger=ngalert.multiorg.alertmanager
INFO [04-03|09:48:40] State cache has been initialized       logger=ngalert.state.manager states=0 duration=66.026468ms
INFO [04-03|09:48:40] Starting scheduler                     logger=ngalert.scheduler tickInterval=10s maxAttempts=3
INFO [04-03|09:48:40] starting                               logger=ticker first_tick=2025-04-03T09:48:50Z
INFO [04-03|09:48:40] starting to provision alerting         logger=provisioning.alerting
INFO [04-03|09:48:40] finished to provision alerting         logger=provisioning.alerting
INFO [04-03|09:48:40] starting to provision dashboards       logger=provisioning.dashboard
INFO [04-03|09:48:40] finished to provision dashboards       logger=provisioning.dashboard
INFO [04-03|09:48:41] Patterns update finished               logger=plugin.angulardetectorsprovider.dynamic duration=309.184356ms
INFO [04-03|09:48:41] Update check succeeded                 logger=grafana.update.checker duration=422.438318ms
INFO [04-03|09:48:41] Update check succeeded                 logger=plugins.update.checker duration=419.357673ms
INFO [04-03|09:48:41] Installing plugin                      logger=plugin.installer pluginId=grafana-pyroscope-app version=
INFO [04-03|09:48:41] Adding GroupVersion featuretoggle.grafana.app v0alpha1 to ResourceManager logger=grafana-apiserver
INFO [04-03|09:48:41] Adding GroupVersion iam.grafana.app v0alpha1 to ResourceManager logger=grafana-apiserver
INFO [04-03|09:48:41] Adding GroupVersion notifications.alerting.grafana.app v0alpha1 to ResourceManager logger=grafana-apiserver
INFO [04-03|09:48:41] Adding GroupVersion userstorage.grafana.app v0alpha1 to ResourceManager logger=grafana-apiserver
INFO [04-03|09:48:41] Adding GroupVersion playlist.grafana.app v0alpha1 to ResourceManager logger=grafana-apiserver
INFO [04-03|09:48:41] Adding GroupVersion dashboard.grafana.app v0alpha1 to ResourceManager logger=grafana-apiserver
INFO [04-03|09:48:41] Adding GroupVersion dashboard.grafana.app v1alpha1 to ResourceManager logger=grafana-apiserver
INFO [04-03|09:48:41] Adding GroupVersion dashboard.grafana.app v2alpha1 to ResourceManager logger=grafana-apiserver
INFO [04-03|09:48:41] Downloaded and extracted grafana-pyroscope-app v1.2.0 zip successfully to /usr/local/bin/grafana-v11.6.0/data/plugins/grafana-pyroscope
-app logger=installer.fs
INFO [04-03|09:48:41] app registry initialized               logger=app-registry
INFO [04-03|09:48:41] Plugin registered                      logger=plugins.registration pluginId=grafana-pyroscope-app
INFO [04-03|09:48:41] Plugin successfully installed          logger=plugin.backgroundinstaller pluginId=grafana-pyroscope-app version= duration=1.107198504
s
INFO [04-03|09:48:41] Installing plugin                      logger=plugin.backgroundinstaller pluginId=grafana-lokiexplore-app version=
INFO [04-03|09:48:42] Installing plugin                      logger=plugin.installer pluginId=grafana-lokiexplore-app version=
INFO [04-03|09:48:42] Downloaded and extracted grafana-lokiexplore-app v1.0.9 zip successfully to /usr/local/bin/grafana-v11.6.0/data/plugins/grafana-lokiexp
lore-app logger=installer.fs
INFO [04-03|09:48:42] Plugin registered                      logger=plugins.registration pluginId=grafana-lokiexplore-app
INFO [04-03|09:48:42] Plugin successfully installed          logger=plugin.backgroundinstaller pluginId=grafana-lokiexplore-app version= duration=476.57831
ms
```

1. Access Grafana dashboard:
    - Open browser and visit: `http://<Prometheus_EC2_IP>:3000`
    - Default credentials:
        - Username: `admin`
        - Password: `admin`

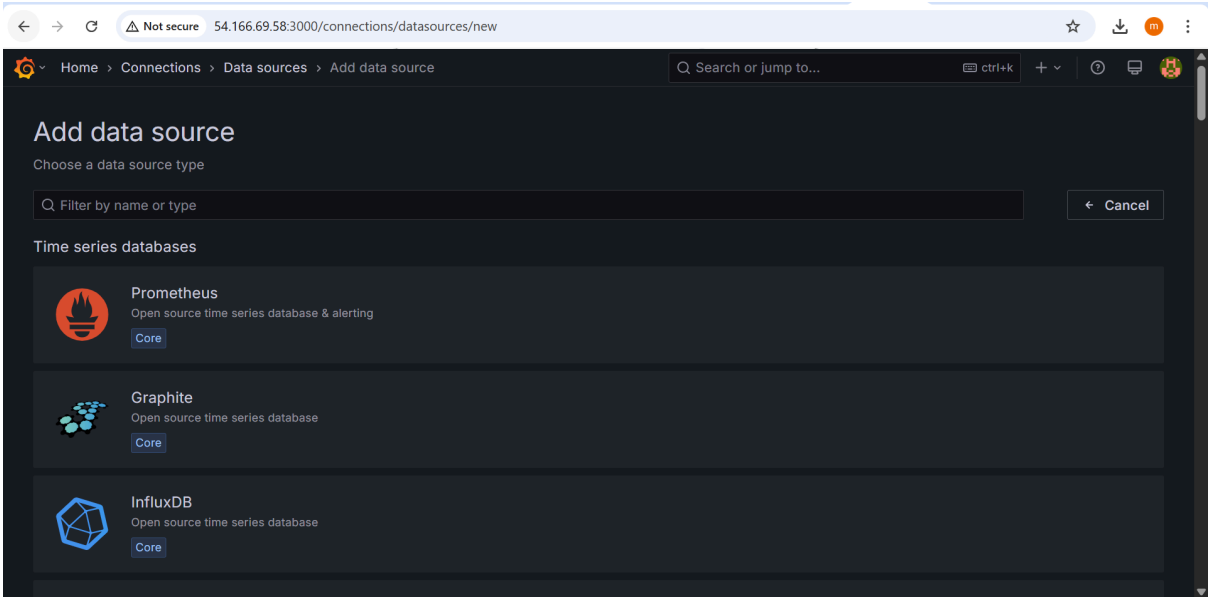


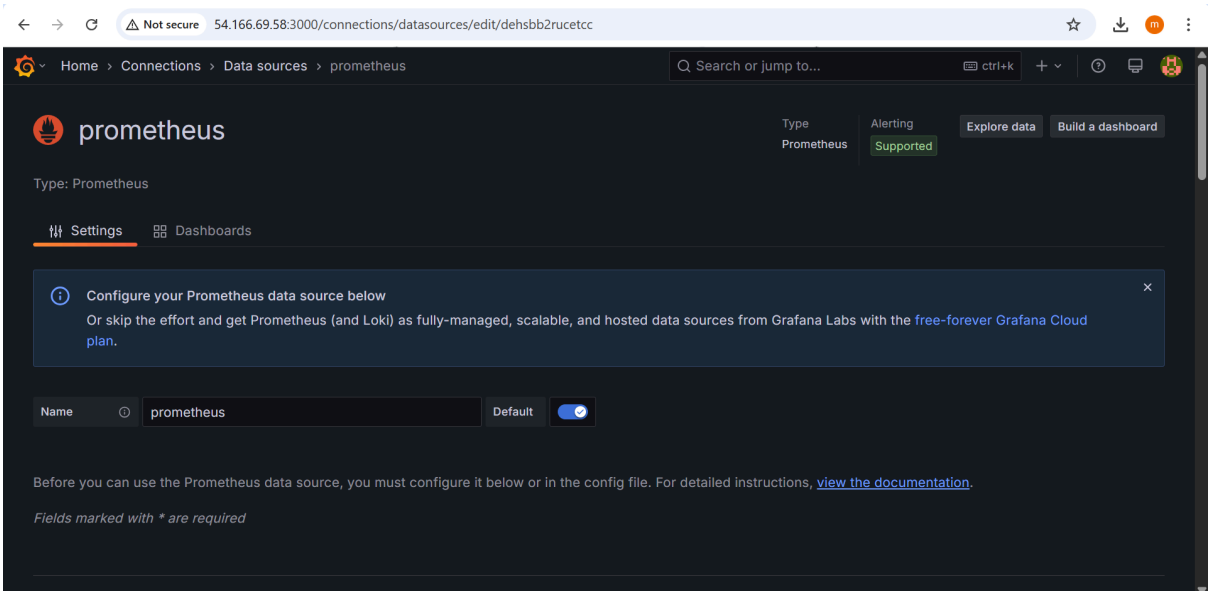



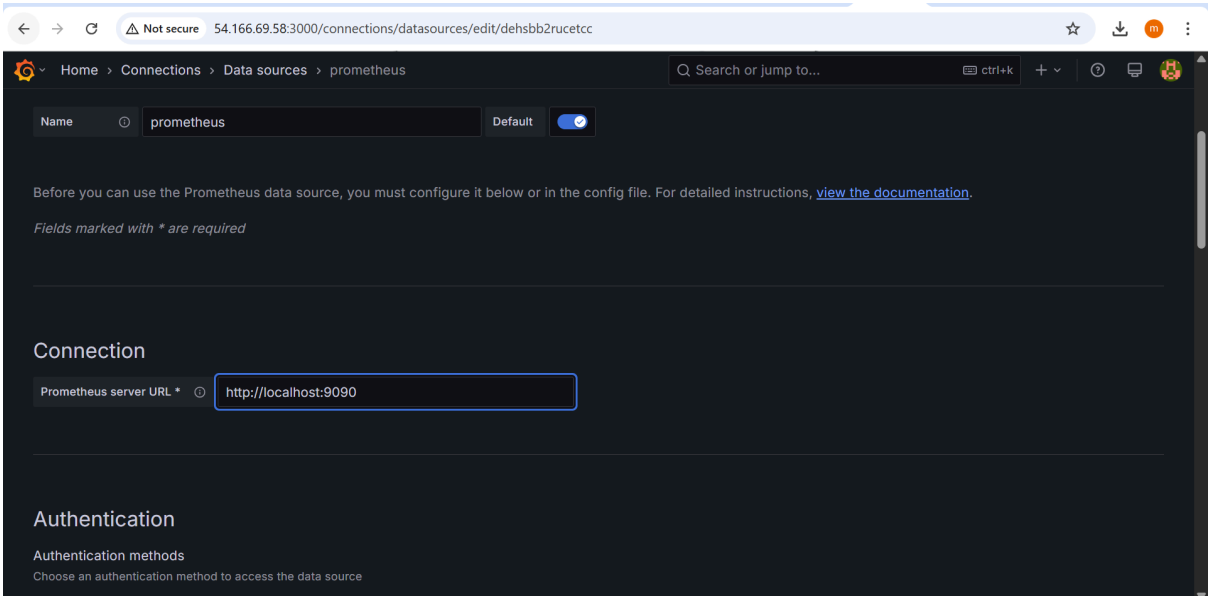

# 5. Configure Prometheus as Data Source in Grafana

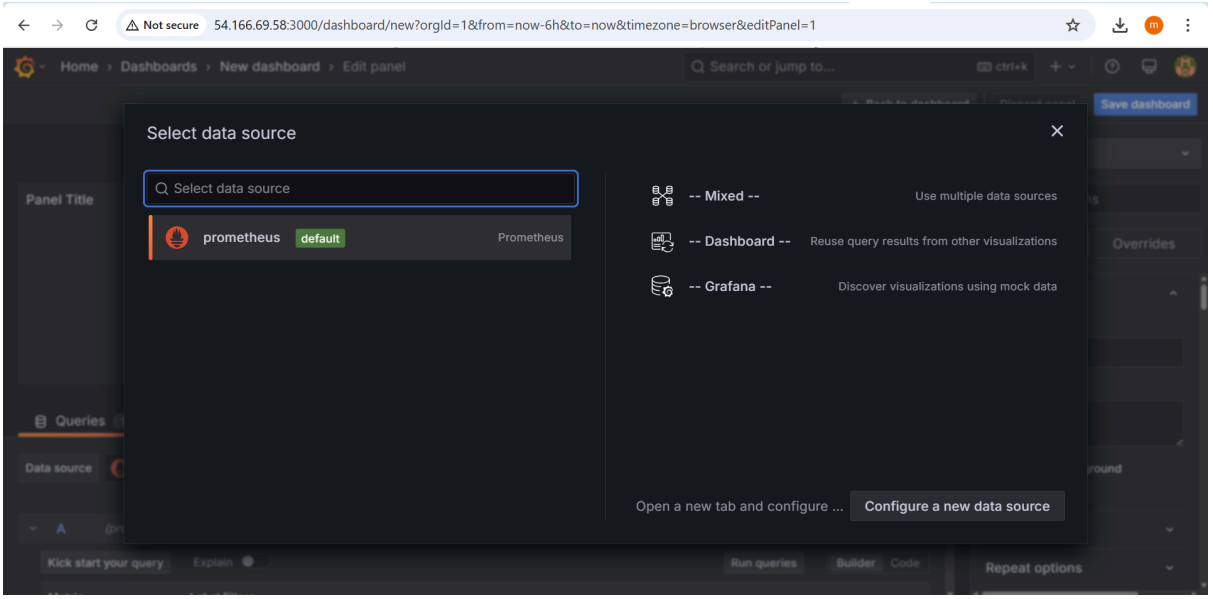
1. Log into **Grafana Dashboard**.
2. Go to **Configuration → Data Sources**.
3. Click **Add Data Source**.
4. Select **Prometheus**.
5. Enter URL: `http://localhost:9090`.

6. Click **Save & Test**.



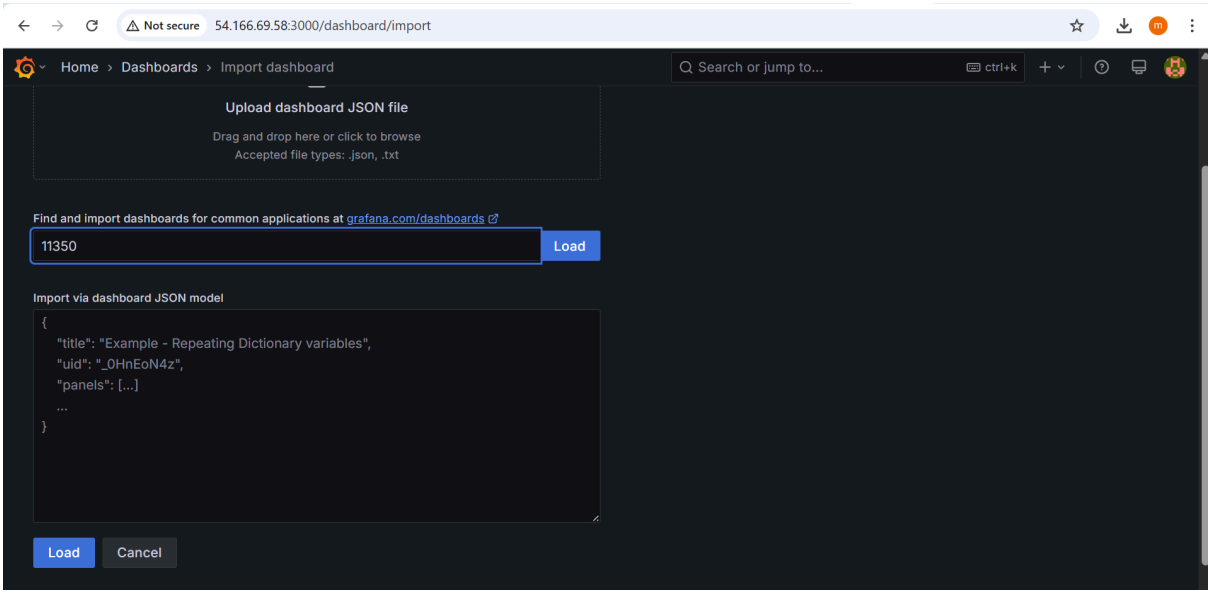





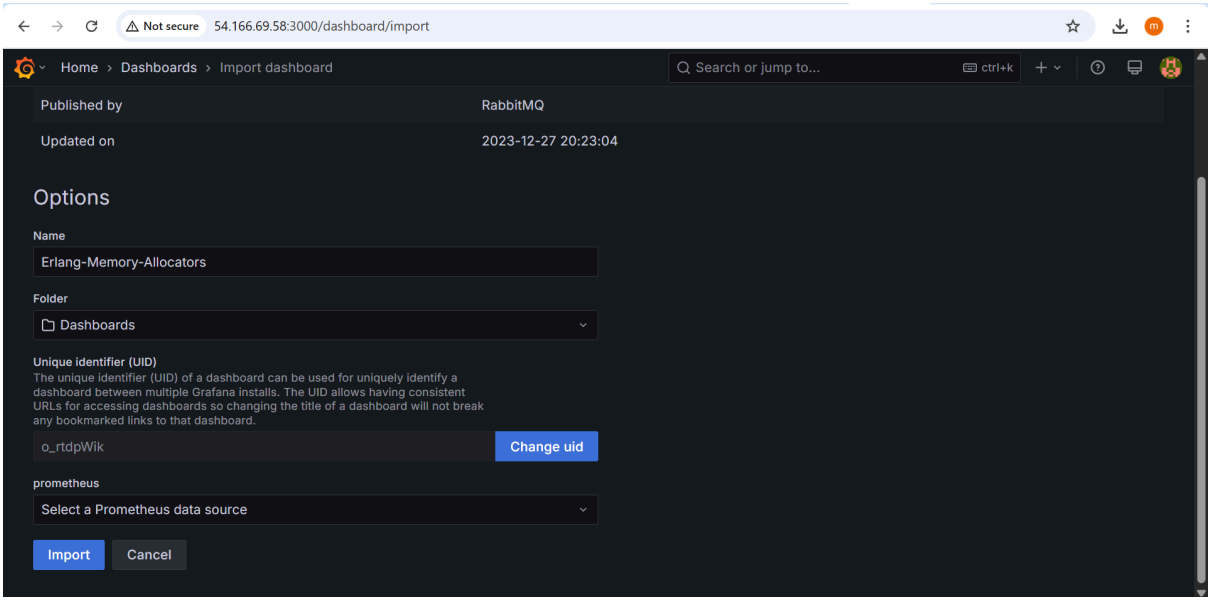



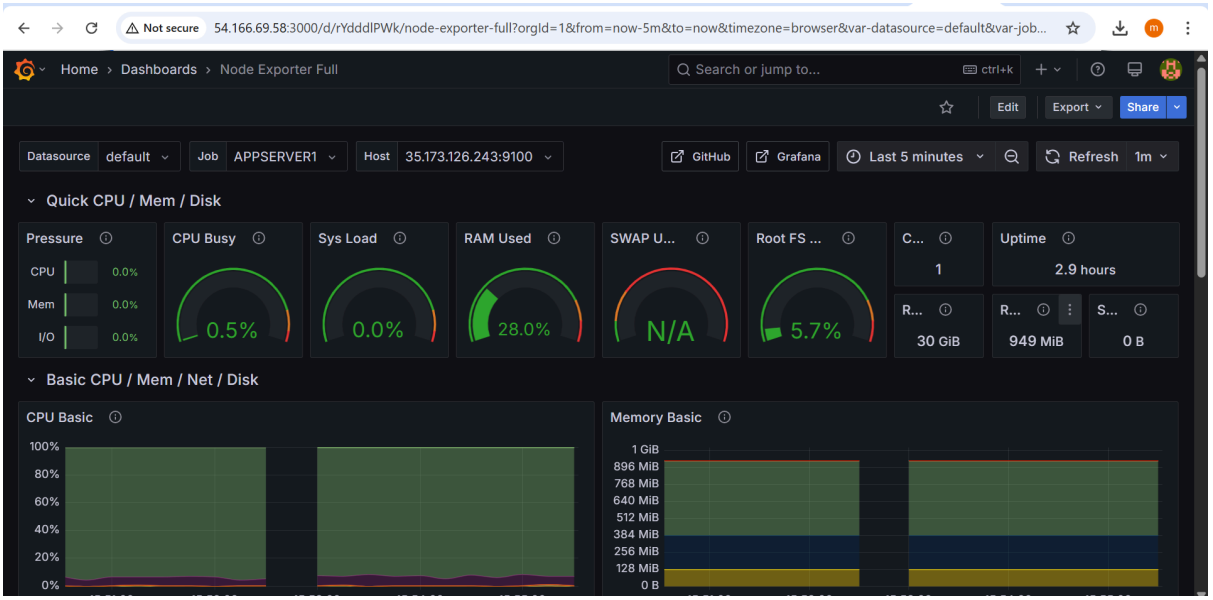

# 6. Import Grafana Dashboard

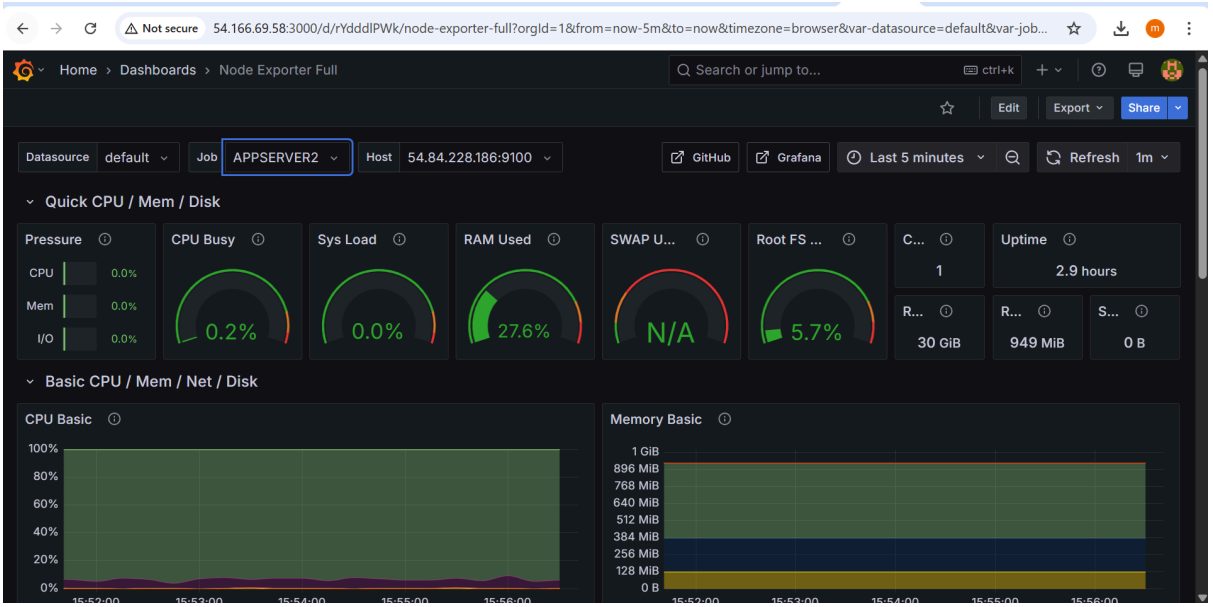
1. Go to **Grafana Dashboard → Import**.
2. Enter Dashboard ID: `11350` (Node Exporter Dashboard).




3. Select **Prometheus** as the Data Source.
4. Click **Import**.



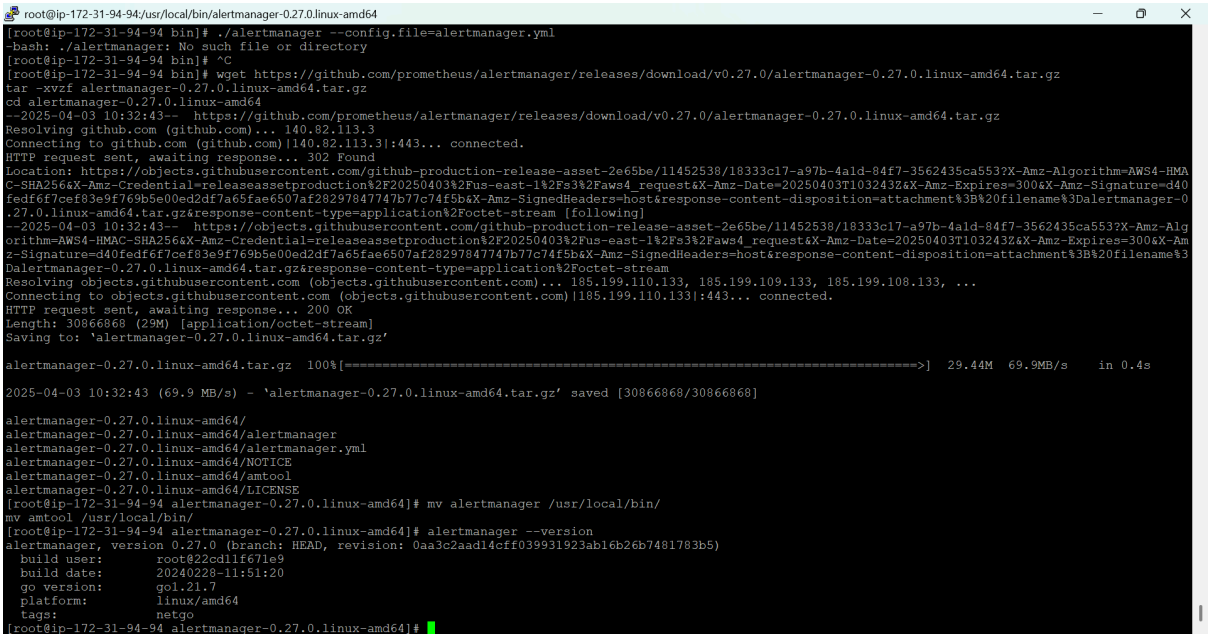


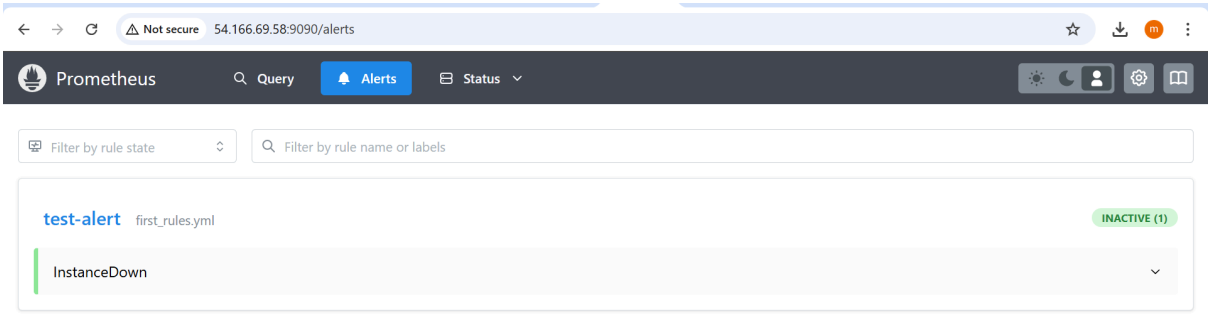

# 7. Setup Alerts in PROMETHEUS

1.create an alert yaml file

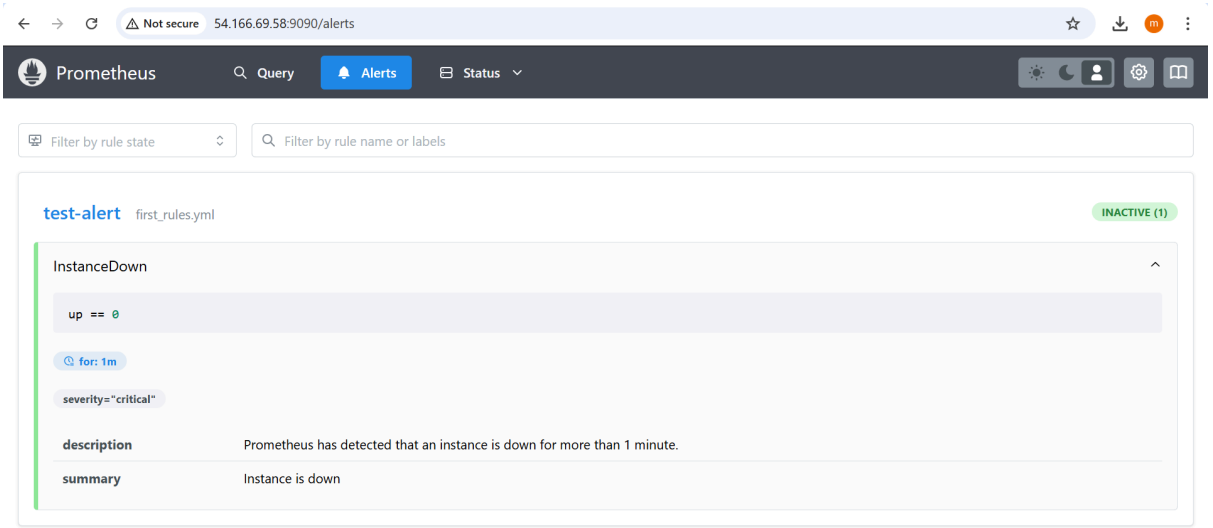

2.edit the config file and add the created yml file




3.check the output of alert in prometheus dashboard



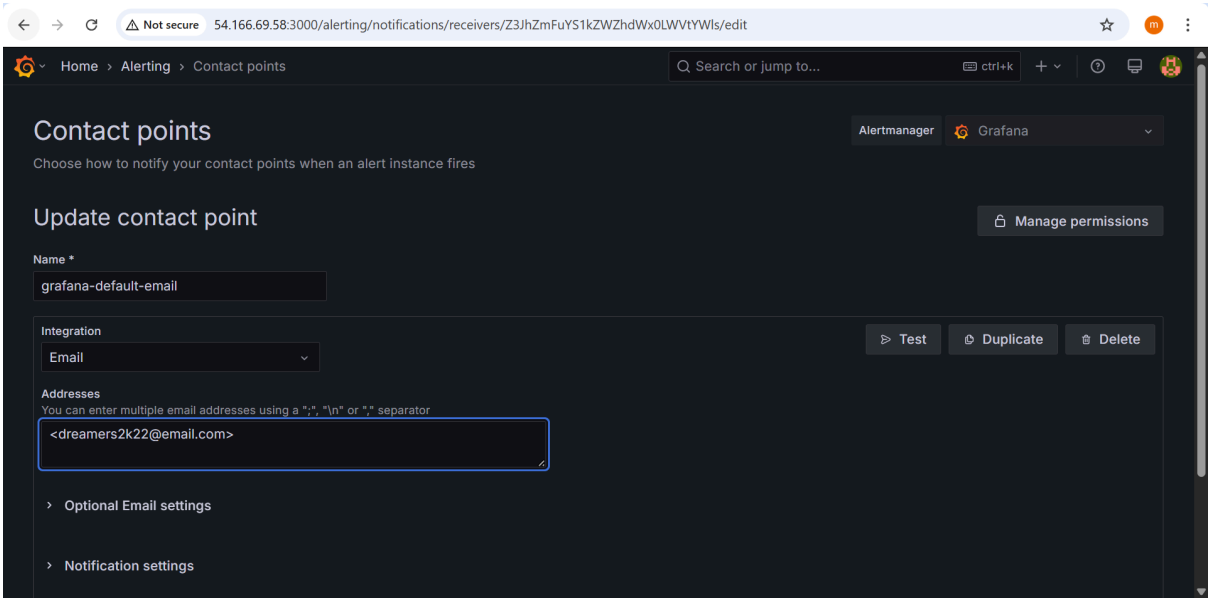

4.yml file contains of alert details of System.



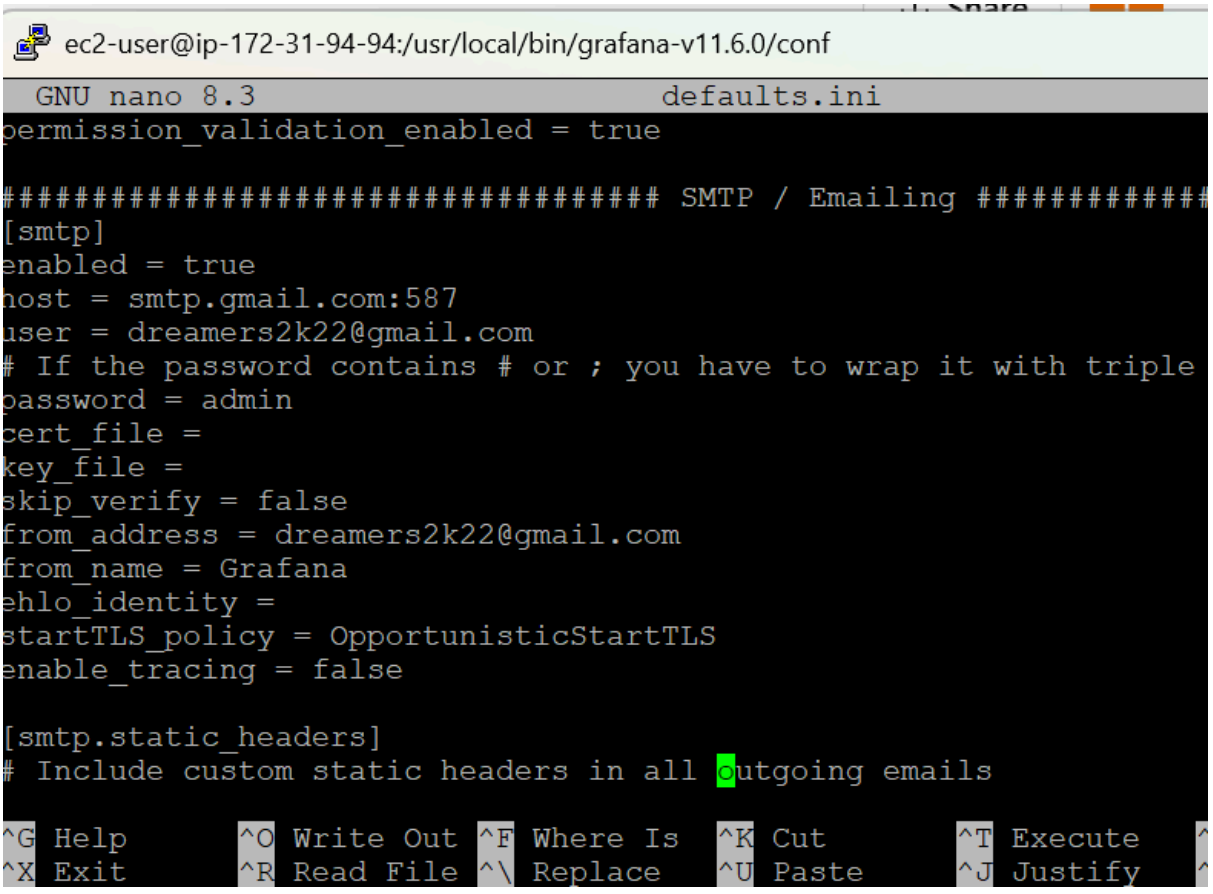

---

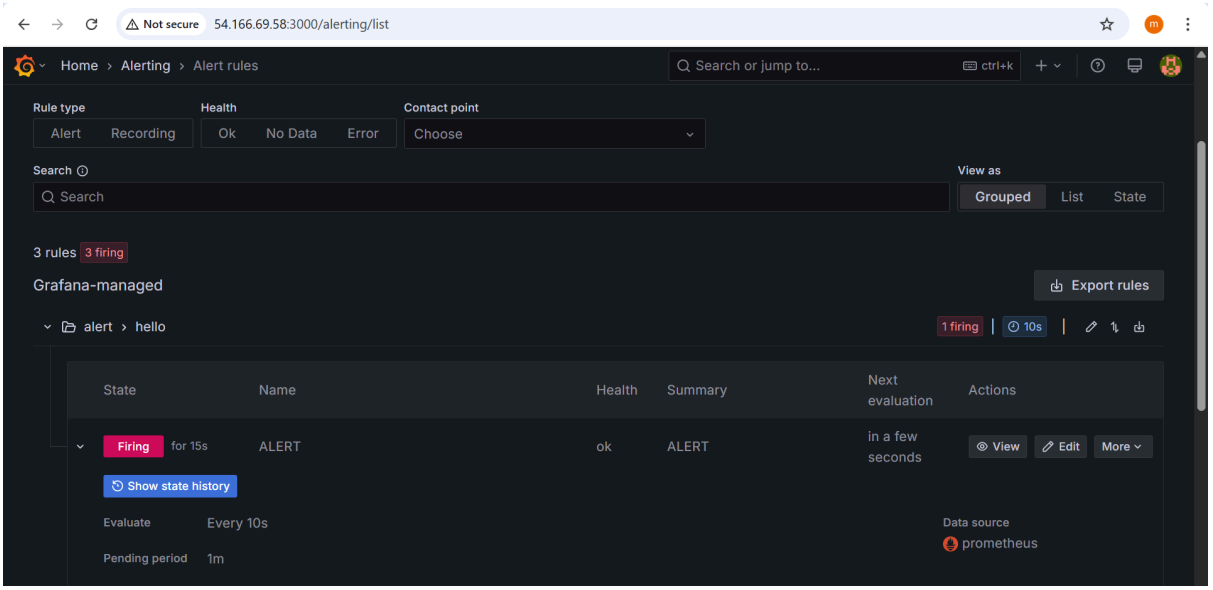
# 8. Setup Alerts in Grafana

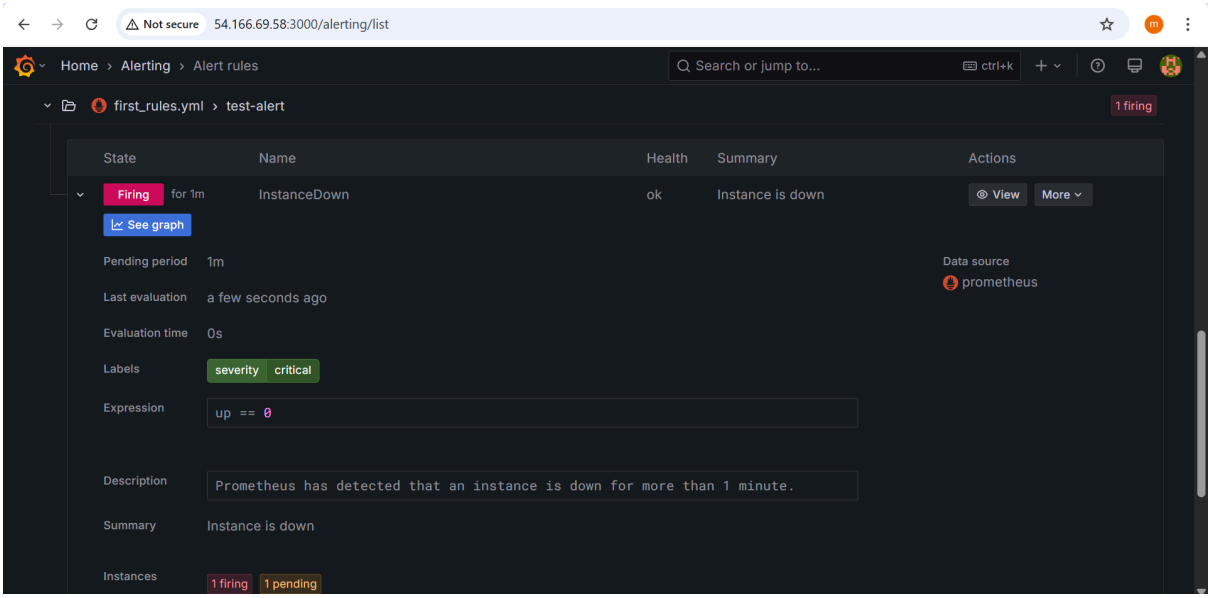
1. Go to **Alerting → Contact Points**.
2. Add Contact Point (Email, Slack, etc.).
3. Configure **Notification Rules**.
4. Enable and Test Alerts


Not secure
54.166.69.58:3000/alerting/notifications/receivers/Z3JhZmFuYS1kZWZhdWx0LWVtYWls/edit
Home › Alerting › Contact points
Search or jump to...
ctrl+k
Contact points
Choose how to notify your contact points when an alert instance fires
Alertmanager
Grafana
Update contact point
Manage permissions
Name *
grafana-default-email
Integration
Email
Test
Duplicate
Delete
Addresses
You can enter multiple email addresses using a ";", "\n" or "," separator
<dreamers2k22@email.com>
Optional Email settings
Notification settings

.


ec2-user@ip-172-31-94-94:/usr/local/bin/grafana-v11.6.0/conf
GNU nano 8.3 defaults.ini
permission_validation_enabled = true
################################## SMTP / Emailing ##########
[smtp]
enabled = true
host = smtp.gmail.com:587
user = dreamers2k22@gmail.com
# If the password contains # or ; you have to wrap it with triple
password = admin
cert_file =
key_file =
skip_verify = false
from_address = dreamers2k22@gmail.com
from_name = Grafana
ehlo_identity =
startTLS_policy = OpportunisticStartTLS
enable_tracing = false
[smtp.static_headers]
# Include custom static headers in all outgoing emails
^G Help ^O Write Out ^F Where Is ^K Cut ^T Execute
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify




# Conclusion

Prometheus serves as the core monitoring system, collecting real-time system metrics using a pull-based mechanism. Grafana acts as the visualization layer, providing interactive dashboards and alerts. Together, they ensure efficient monitoring and proactive issue detection in your AWS environment.