Task: Configure Prometheus With Grafana For Monitoring Of Multiple Client Nodes.

◆ Understanding Prometheus and Grafana

Prometheus and **Grafana** are two widely-used tools in the tech industry for monitoring and visualization.

- **Prometheus**: It is an open-source monitoring and alerting toolkit designed for reliability and scalability. It collects metrics from configured endpoints at specified intervals, stores them, and allows for complex querying and alerting.]

- **Grafana**: This is an open-source platform for monitoring and observability that provides powerful visualizations and dashboards. It can integrate with various data sources, including Prometheus, to create dynamic and interactive graphs and dashboards.

◆ Why Configuring Prometheus with Grafana is Important

- **Real-Time Monitoring:** The tech industry often relies on real-time data to ensure systems are functioning optimally. Prometheus collects and stores time-series data, while Grafana visualizes this data, allowing for immediate insights into system performance.
- **Scalability and Flexibility:** In modern tech environments, applications and infrastructure are often complex and distributed across multiple nodes. Prometheus is designed to scale and handle large volumes of metrics, making it suitable for monitoring multiple client nodes. Grafana can visualize data from these numerous sources in a consolidated and user-friendly manner.
- **Alerting and Incident Management:** Prometheus supports alerting based on defined metrics. This capability helps in proactively managing issues before they become critical. Grafana's integration allows for better visibility into these alerts, improving response times and reducing downtime.
- **Data-Driven Decisions:** Accurate monitoring data helps in making informed decisions about system improvements, resource allocation, and performance tuning. Grafana's visualizations provide clear and actionable insights.

◆ Impact of Using Prometheus and Grafana

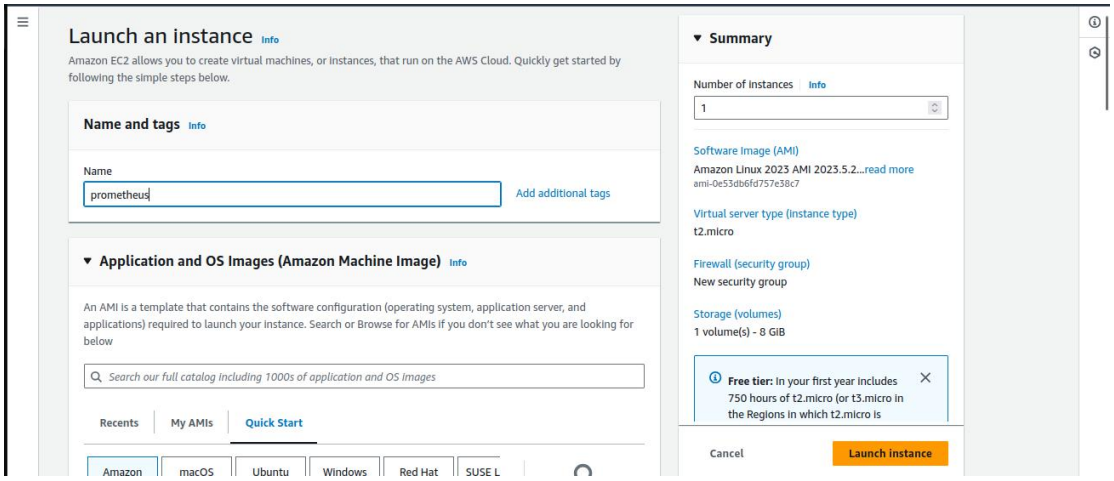
- **Enhanced Visibility:** Using Prometheus and Grafana together provides comprehensive visibility into the health and performance of your systems. It transforms raw metrics into meaningful insights through visualizations.
- **Improved Troubleshooting:** When issues arise, Grafana's dashboards can help quickly identify the root cause by correlating different metrics. This speeds up troubleshooting and reduces mean time to resolution (MTTR).
- **Proactive Monitoring:** With alerts configured in Prometheus and visualized in Grafana, you can proactively monitor your systems and address potential issues before they impact end-users.
- **Customizable Dashboards:** Grafana allows you to create highly customizable dashboards, which means you can tailor the views to specific needs and stakeholders, providing relevant information in an easily digestible format.

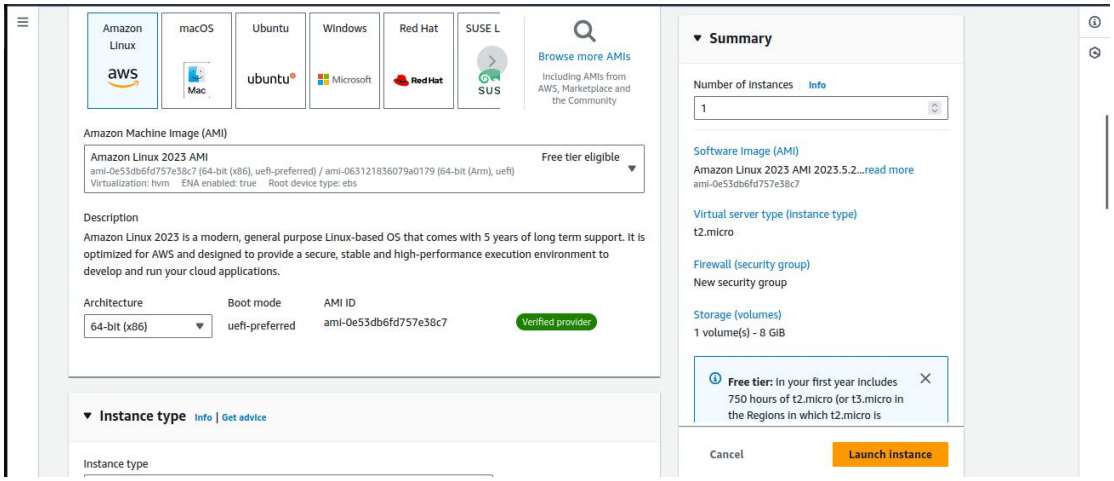
◆ <u>How It Will Help in Your Project</u>

- **Consolidated Monitoring:** For a project involving multiple client nodes, Prometheus will collect data from each node and store it efficiently. Grafana will then pull this data and present it in a unified dashboard, allowing you to monitor all nodes from a single interface.
- **Performance Metrics:** By configuring Prometheus to collect various performance metrics (CPU usage, memory consumption, network activity), and visualizing them in Grafana, you can track the performance of each client node, identify bottlenecks, and optimize performance.
- **Customized Alerts:** Setting up alerts based on your specific metrics ensures you are notified about critical issues promptly. This can be configured in Prometheus and displayed in Grafana, so you can manage and respond to alerts effectively.
- **Documentation and Reporting:** Grafana's dashboards can serve as a documentation tool, providing visual evidence of system performance and reliability over time. This can be useful for reporting progress to stakeholders or for post-incident reviews.

* Let's Explore actually how we can use Prometheus & Grafana

1] Launch an EC2 instance which is used as prometheus server.





* Launch & Connnect ec2 then used below steps for configure prometheus server.

2] First we need to create system user which haven't home & shell directory, use below command for that.

- **sudo useradd --no-create-home --shell /bin/false prometheus**

* Creating a dedicated user for Prometheus enhances security by isolating the service, applying the principle of least privilege, and preventing unauthorized access or system modifications. The `--no-create-home` and `--shell /bin/false` options further limit potential risks by not creating a home directory or allowing shell access.

3] Create the directories in which we will be storing our configuration files and libraries with this commands:

- **sudo mkdir /etc/prometheus**
- **sudo mkdir /var/lib/prometheus**

4] Set the ownership of the /var/lib/prometheus directory with below command:

- **sudo chown prometheus:prometheus /var/lib/prometheus**
- **sudo chown prometheus:prometheus /etc/prometheus**

5] You need to inside /tmp in which we have to download prometheus with below command:

- **cd /tmp/**

* Download prometheus using wget using below command, if you want download latest prometheus then visit prometheus download page

wgethttps://github.com/prometheus/prometheus/releases/download/v2.31.1/prometheus-2.31.1.linux-amd64.tar.gz

6] Extract the files using tar :

- **tar -xvf prometheus-2.31.1.linux-amd64.tar.gz**

7] You need to inside prometheus-2.31.1.linux-amd64 so run the below command

- **cd prometheus-2.31.1.linux-amd64 (version might be change)**

8] Move the configuration file and set the owner to the prometheus user:

- **sudo mv console* /etc/prometheus**
- **sudo mv prometheus.yml /etc/prometheus**
- **sudo chown -R prometheus:prometheus /etc/prometheus**

9] Move the binaries and set the owner:

- **sudo mv prometheus /usr/local/bin/**
- **sudo chown prometheus:prometheus /usr/local/bin/prometheus**

10] Create the prometheus service file using below command:

- **vim /etc/systemd/system/prometheus.service**

* Add the below line in this /etc/systemd/system/prometheus.service

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

[Service]
User=prometheus
Group=prometheus
Type=simple
ExecStart=/usr/local/bin/prometheus \
    --config.file /etc/prometheus/prometheus.yml \
    --storage.tsdb.path /var/lib/prometheus/ \
    --web.console.templates=/etc/prometheus/consoles \
    --web.console.libraries=/etc/prometheus/console_libraries

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

11] Reload systemd using below command:

- **sudo systemctl daemon-reload**

12] Enable Prometheus service using below commands:

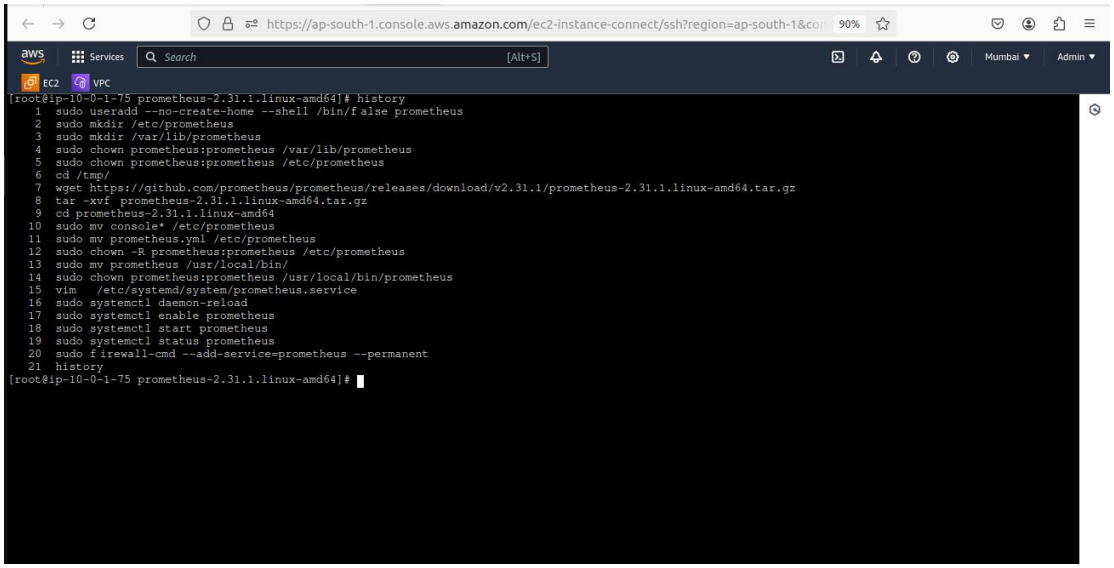
- **sudo systemctl enable prometheus**

13] Start Prometheus service using below commands:

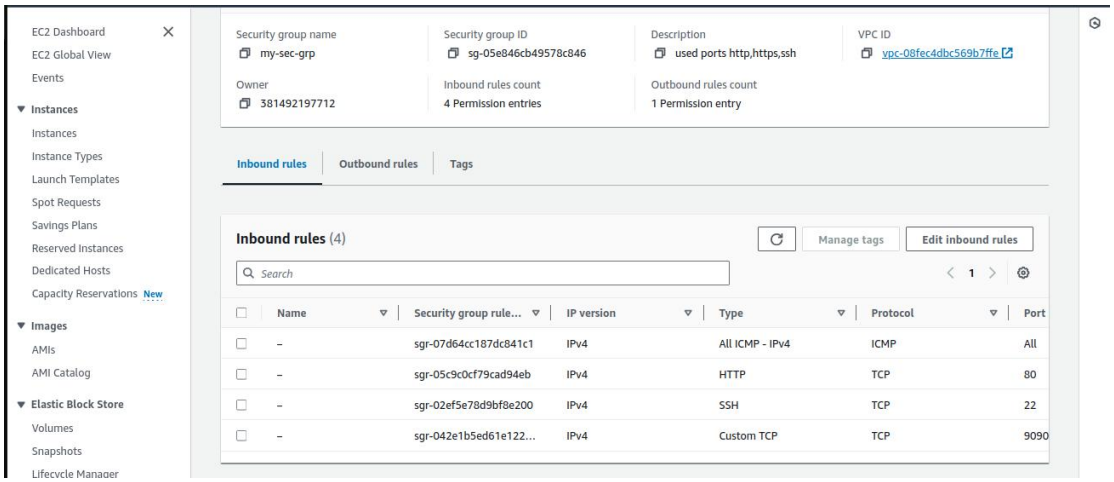
- **sudo systemctl start prometheus**

14] Check Status of Prometheus service using below commands:

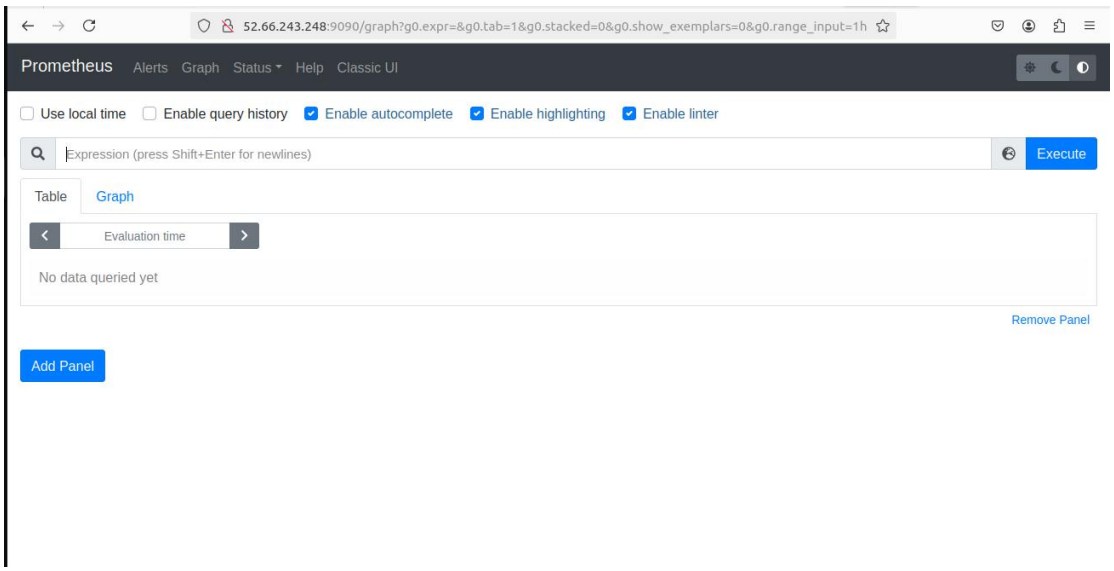
- **sudo systemctl status prometheus**



15] We need to open port no. 9090 which is allow us to connect prometheus server.



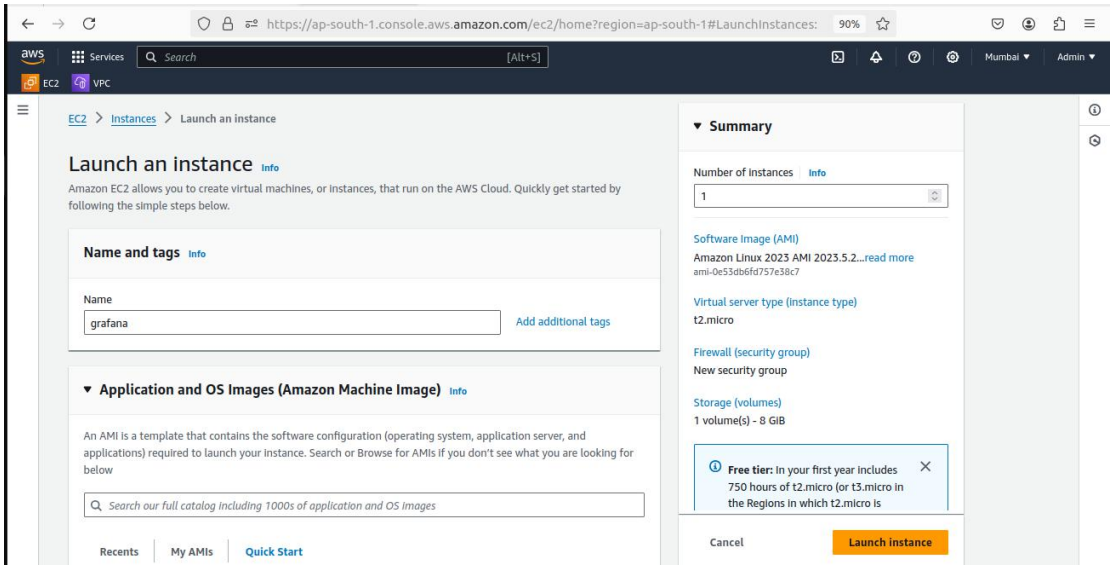
16] Go to the browser copy & paste instance public ip with :9090 (ex-52.66.243.248:9090).

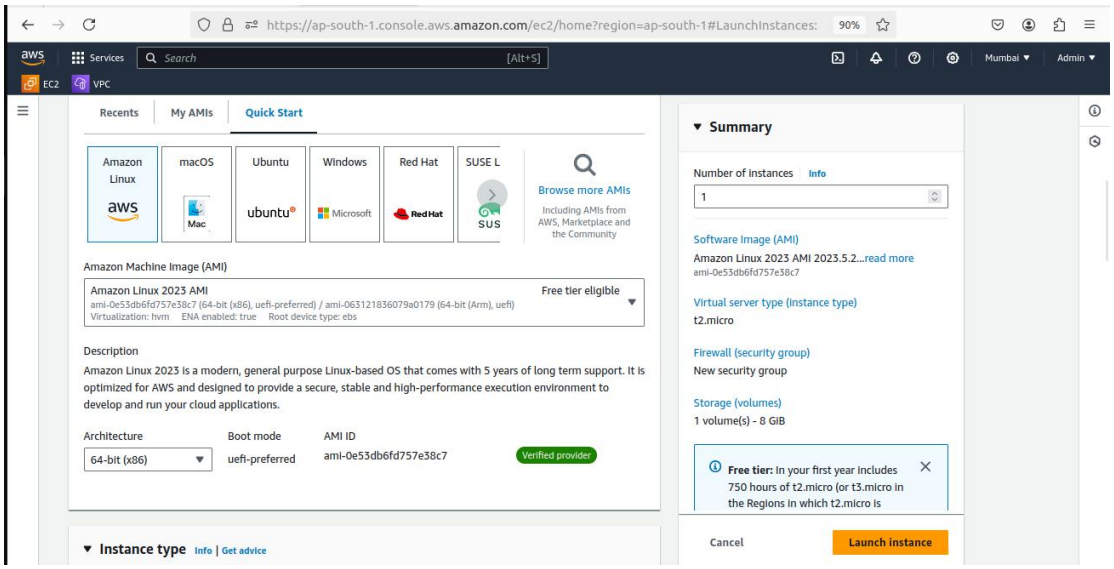


*****!!! Congartulations !!!*****

You Have Successfully Configure Prometheus Server, Now we have to configure Grafana Server.

1] Launch an EC2 instance which is used as Grafana server.





* Launch & Connnect ec2 then used below steps for configure prometheus server.

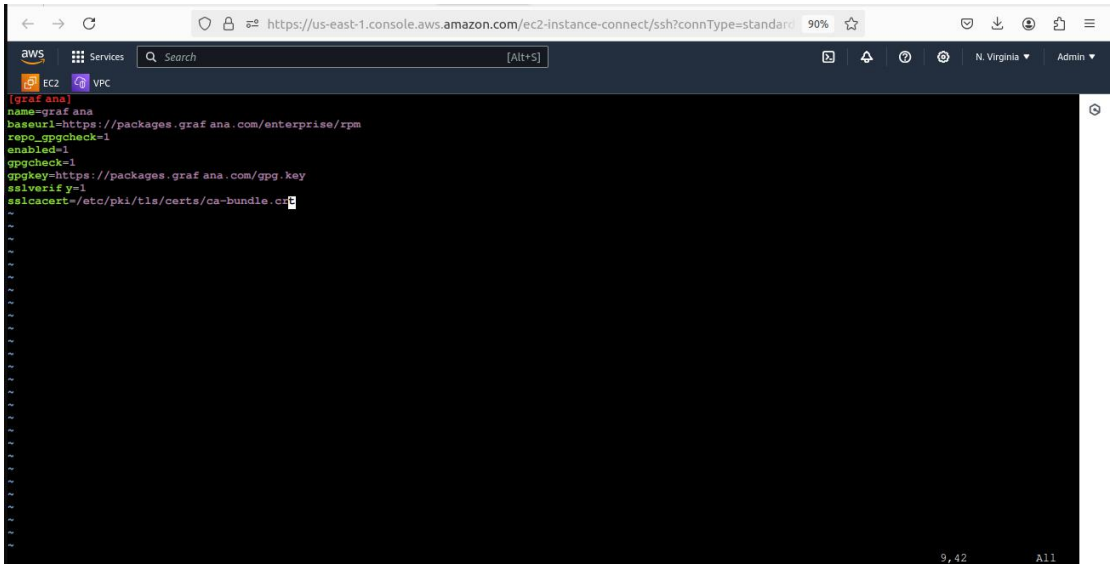
2] Add the Grafana YUM repo using vim command:

- sudo vim /etc/yum.repos.d/grafana.repo

* A YUM repository (repo) is a collection of packages and metadata used by the YUM package manager on RPM-based Linux distributions (like CentOS, RHEL, and Fedora) to install, update, and manage software. The repo configuration file, like /etc/yum.repos.d/grafana.repo, specifies where YUM can find these packages for specific software, such as Grafana in this case.

[grafana]
name=grafana

baseurl=https://packages.grafana.com/enterprise/rpm
repo_gpgcheck=1
enabled=1
gpgcheck=1
gpgkey=https://packages.grafana.com/gpg.key
sslverify=1
sslcacert=/etc/pki/tls/certs/ca-bundle.crt



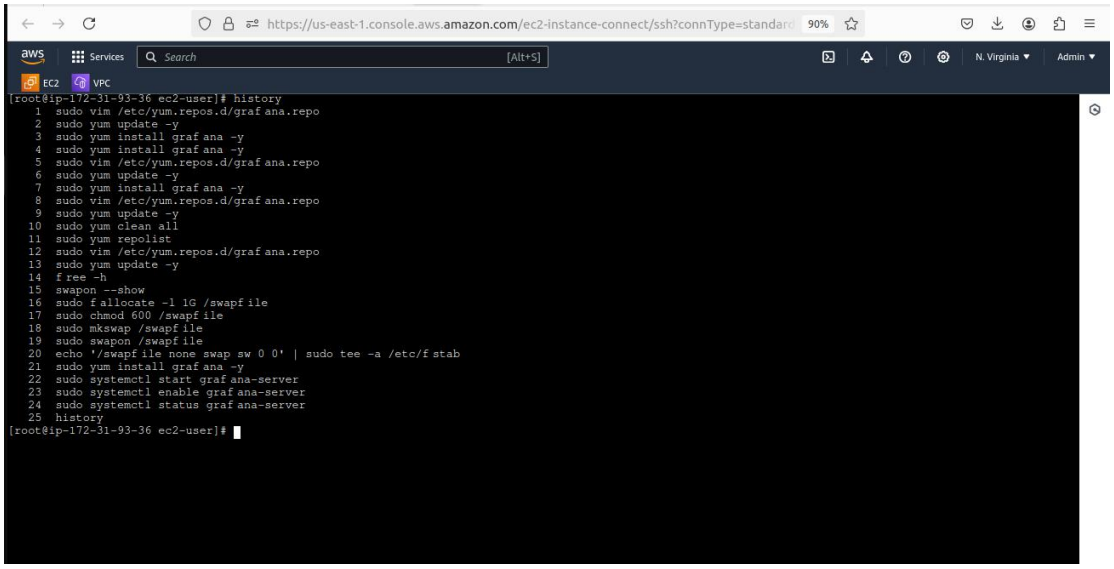
3] update the system packages to take effect
- **sudo yum update -y**

4] Now lets Install Grafana using below command
- **sudo yum install grafana -y**

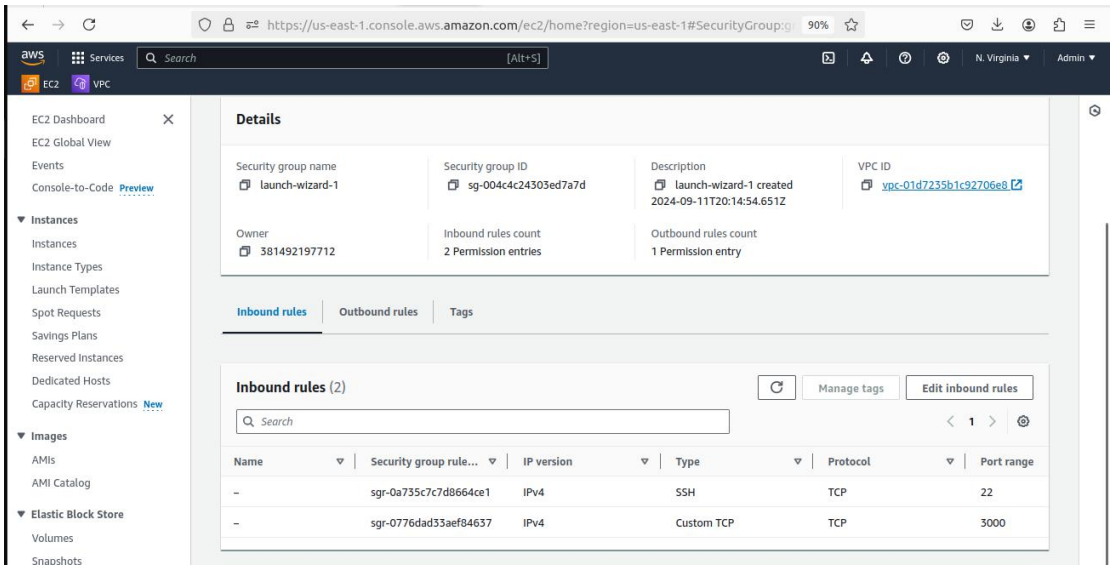
* You might be facing issue with memory issue for encounter that error follow below commands:
  - **free-h**
  - **swapon --show**
  - **sudo fallocate -l 1G /swapfile**
  - **sudo chmod 600 /swapfile**
  - **sudo mkswap /swapfile**
  - **sudo swapon /swapfile**
  - **echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab**

5] Now start and enable the Grafana service using below command:
- **sudo systemctl start grafana-server**
- **sudo systemctl enable grafana-server**

6] Verify the Grafana Service Status using below command:
- **sudo systemctl status grafana-server**

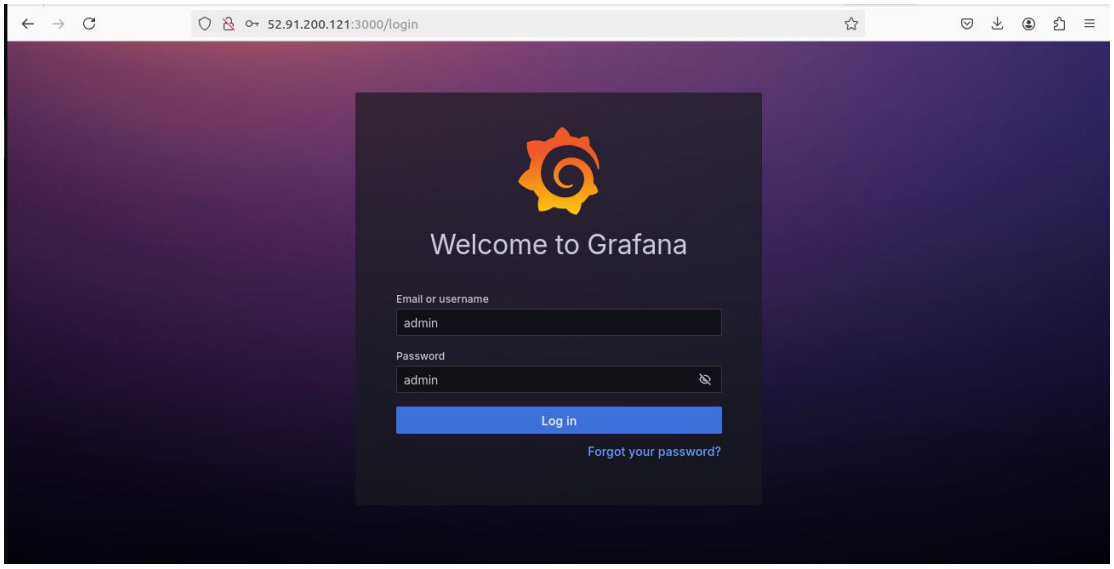


7] We need to open port no. 3000 which is allow us to connect Grafana server.

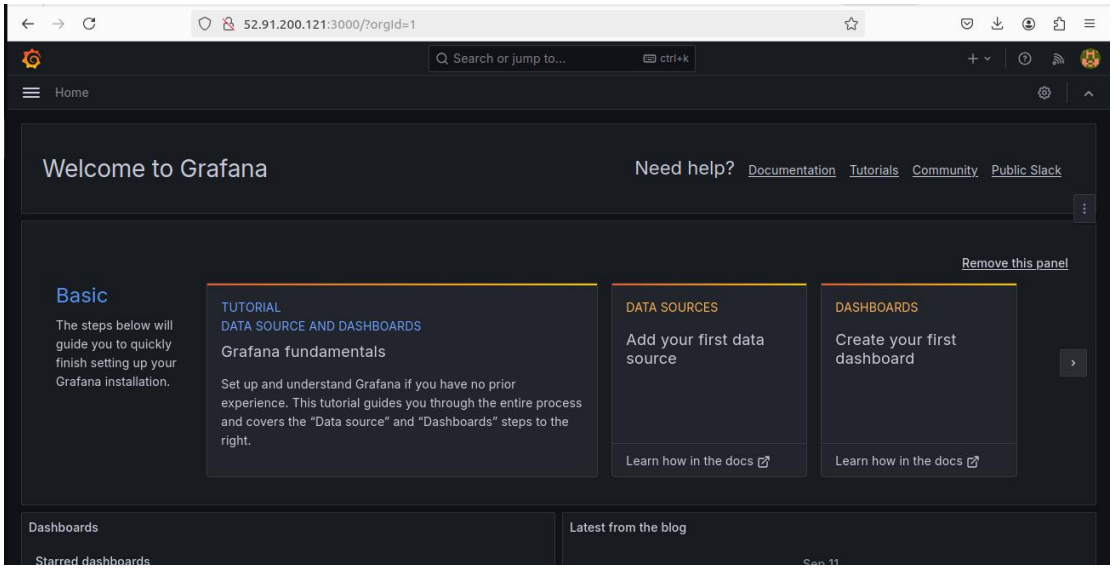


8] Now open browser copy and paste instance public ip with :3000 (ex-52.91.200.121:3000)

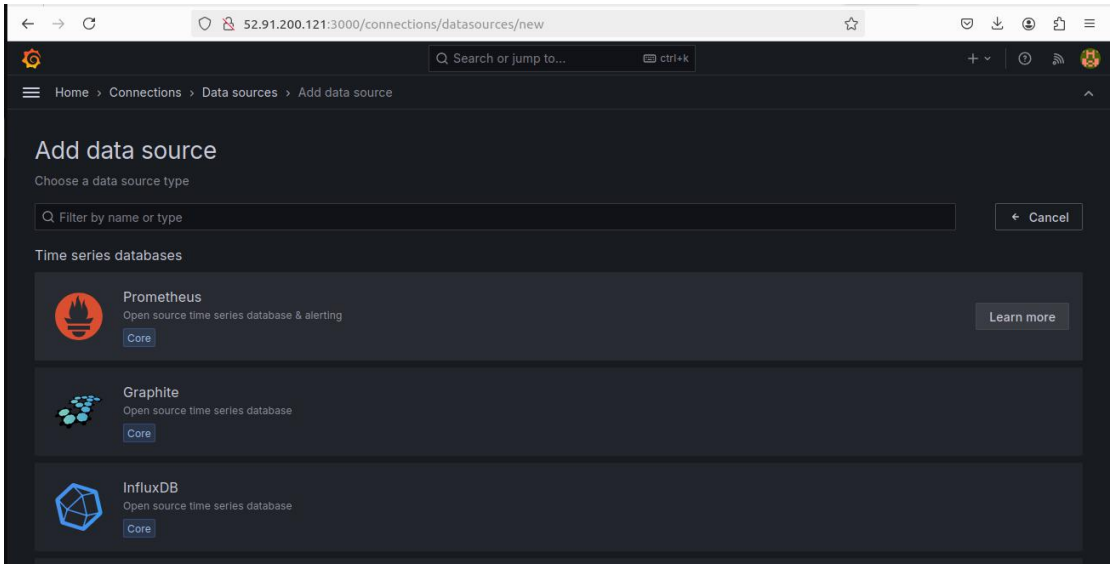
* Default username and password both are admin after login you can set the password as per your choice.



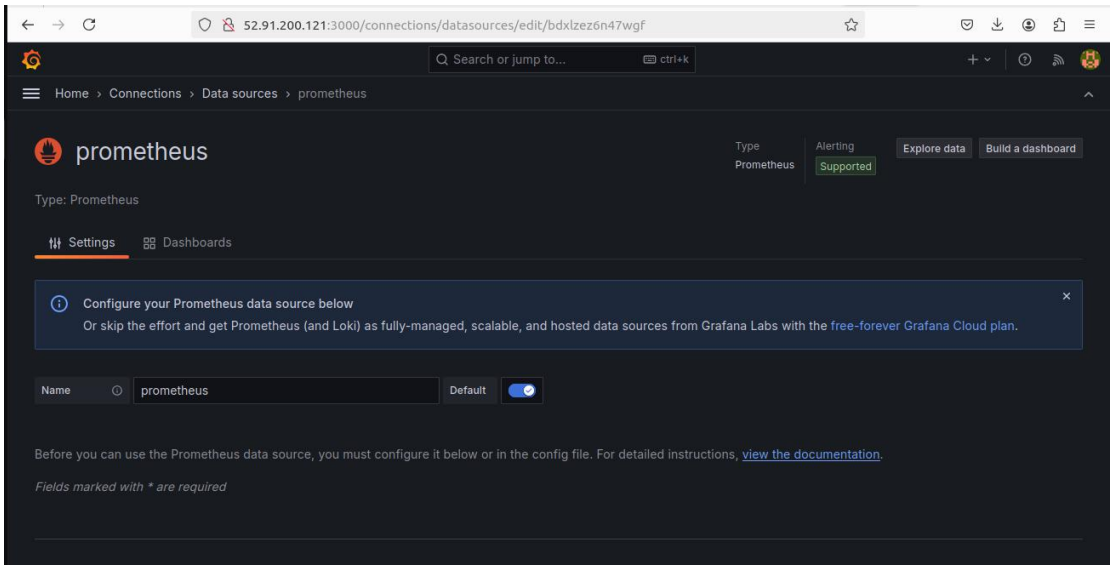
* Configure Prometheus as Grafana DataSource
  Go to Home > Click on Data Sources



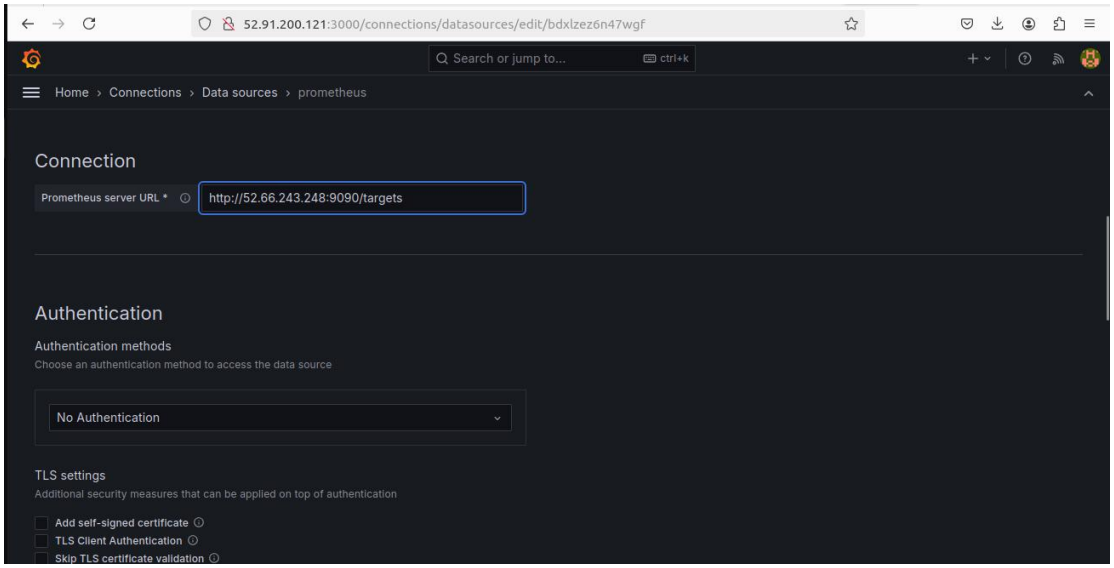
Add data source as prometheus



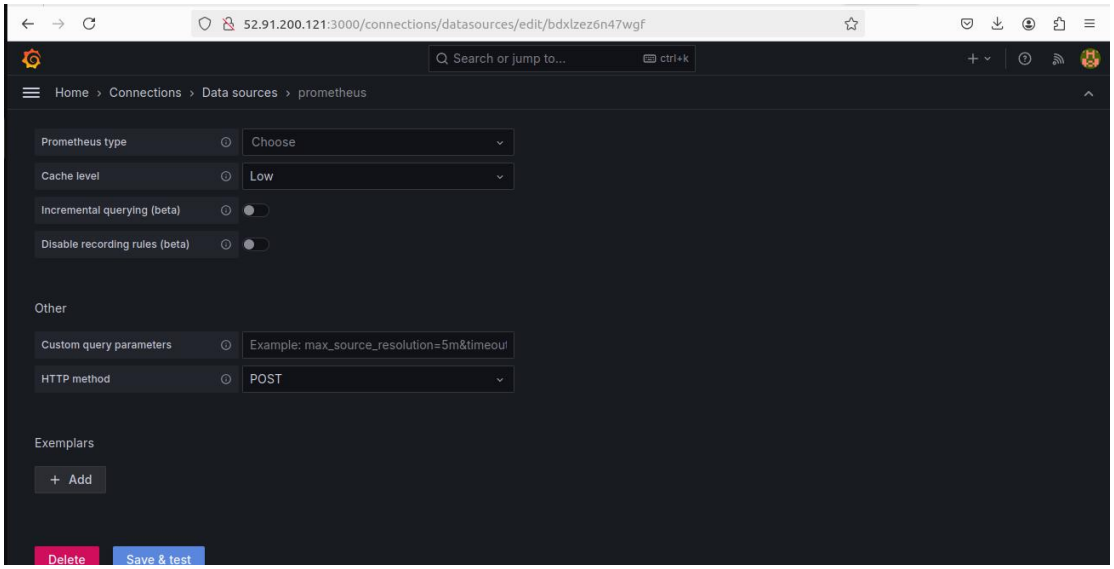
* Give name for our datasource



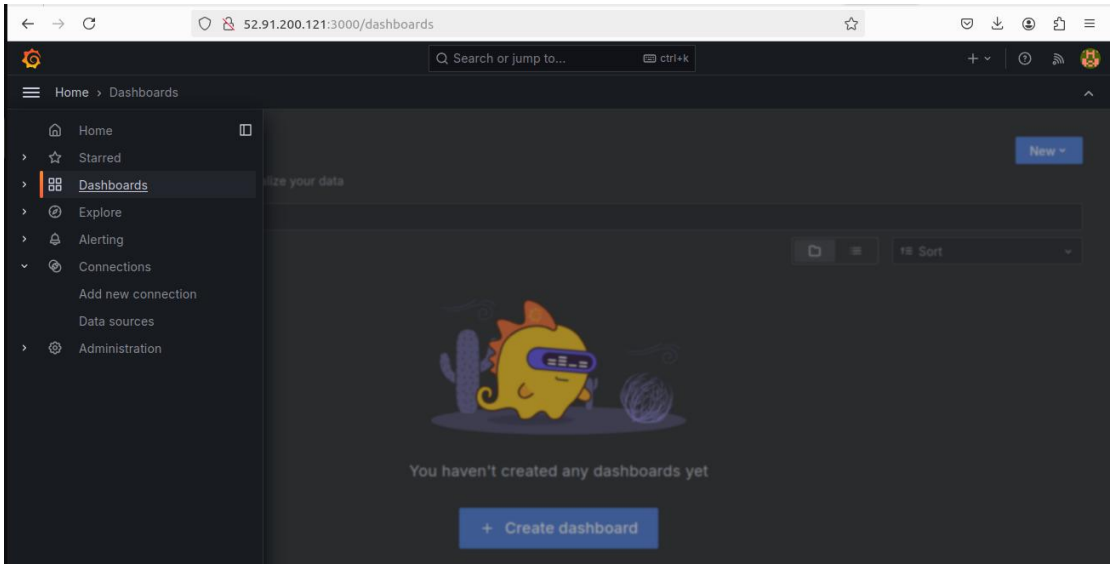
* copy and paste your prometheus server URL (ex- http://52.66.243.248:9090/)



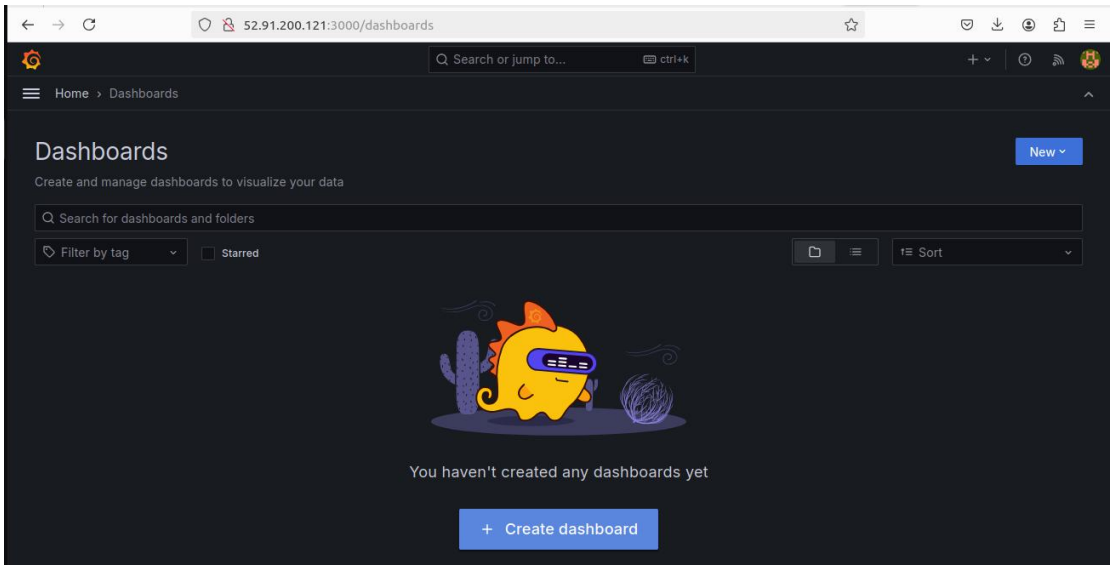
* Click save and next



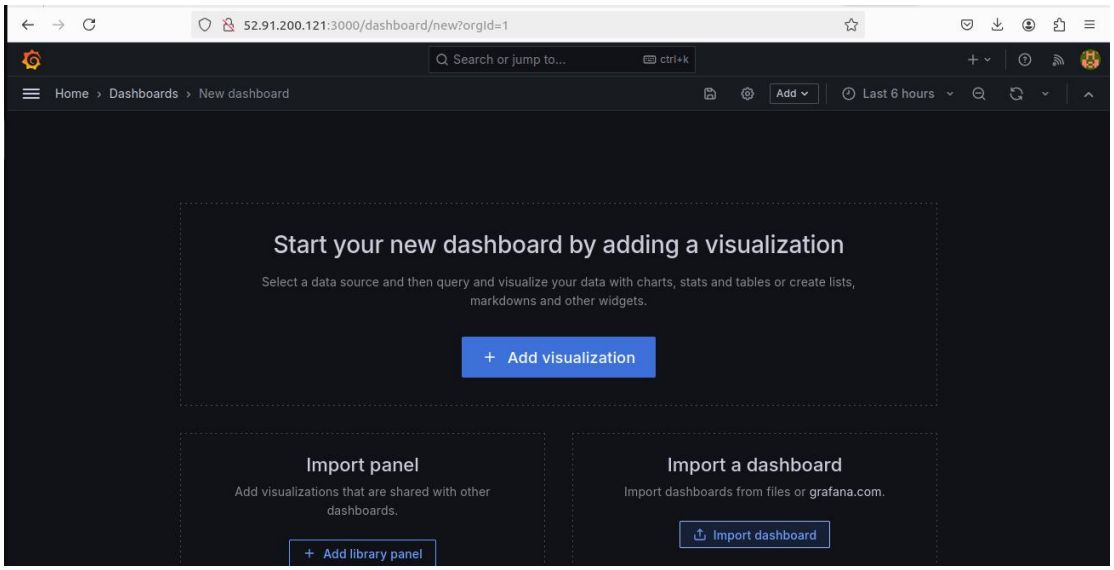
* Now time to set attractive dashboard.
  Click on 3 lines on top of left corner > click on Dashboards >



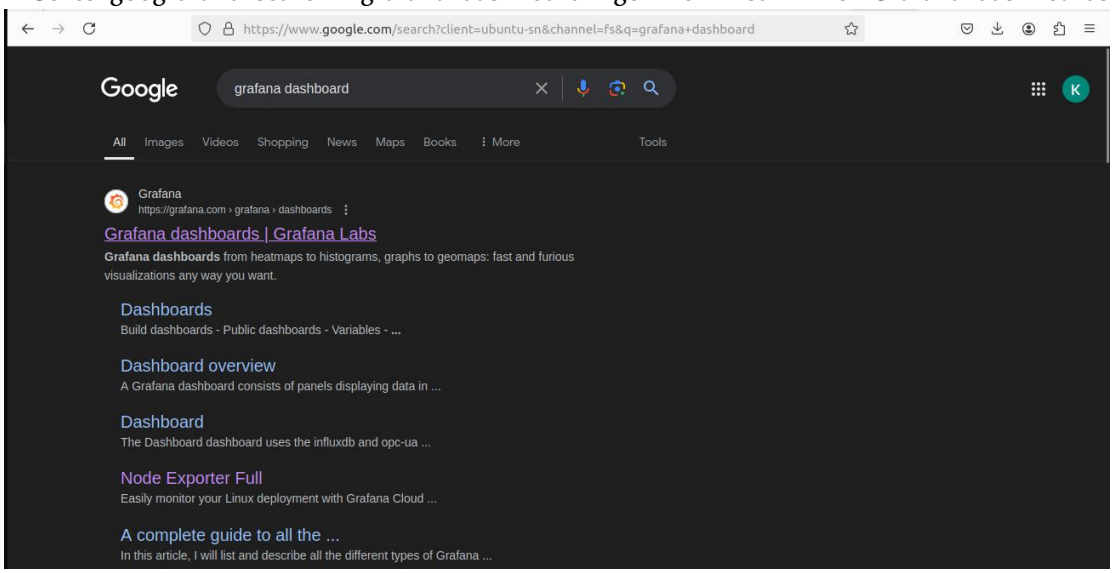
* Click on + Create dashboard



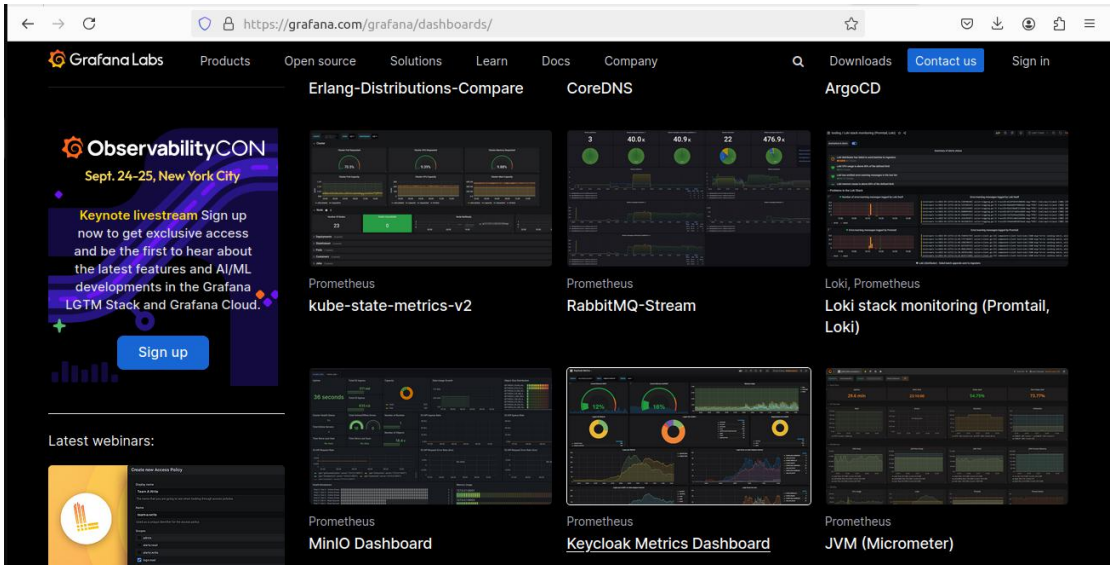
* Click on Import dashboard



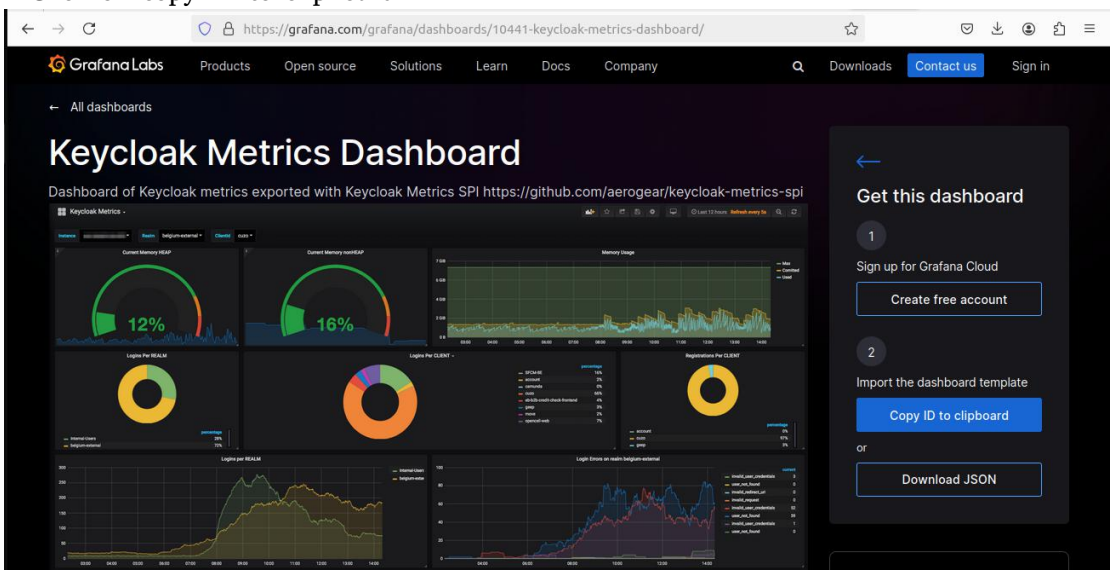
* Go to google and search “grafana dashboard” go with first link of Grafana dashboards



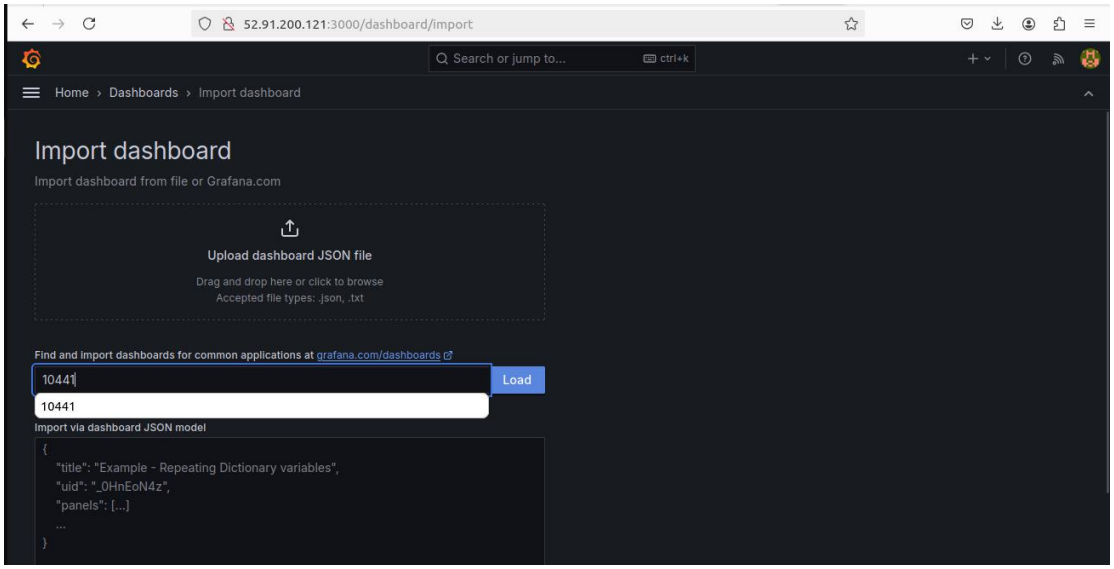
* Select dashboard with your choice and click it on



* Click on copy ID to clipboard



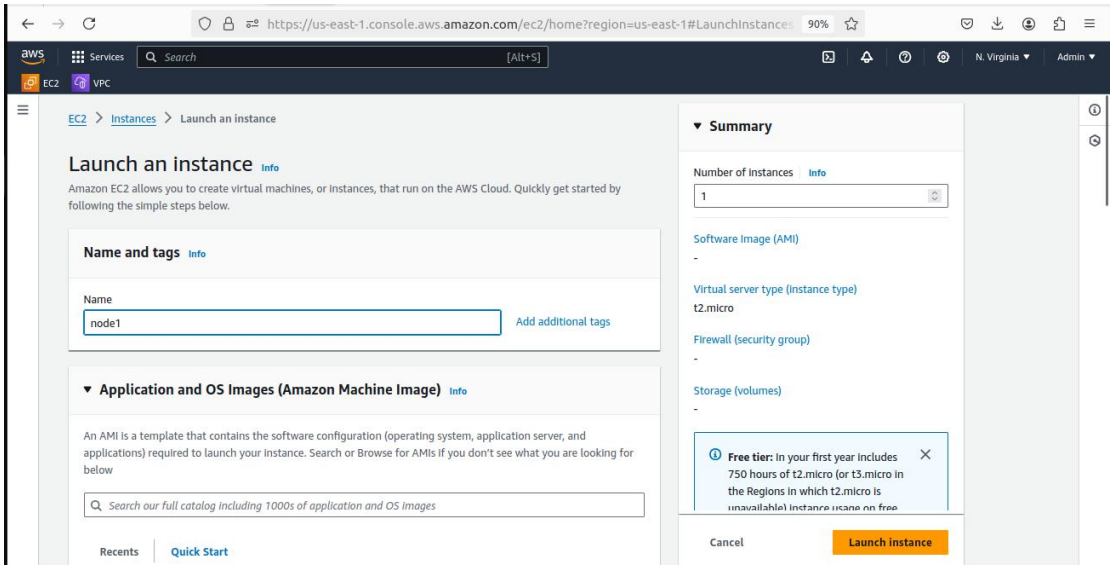
* Now go to grafana server tab and paste dashboard id and click on Load



*****!!! Congratualations !!!*****

You have successfully configure Grafana server and Dashboard of Grafana server. Now time to configure node and verify dashboard data.

1] Launch an EC2 instance which have to configure node



* Launch and connect the instance.

1] You need to inside /tmp in which we have to download prometheus with below command:

- **cd /tmp**

* To install Node Exporter first navigate to Prometheus official download page (https://prometheus.io/download/), Scroll down and you will get node_exporter section and then select Linux OS for amd64.
Now right click on node exporter and copy link address

- **sudo wget** https://github.com/prometheus/node_exporter/releases/download/v1.7.0/node_exporter-1.7.0.linux-amd64.tar.gz

2] Unzip the downloaded the file using below command

- **sudo tar -xvzf node_exporter-*.*-amd64.tar.gz**

3] Move the binary file of node exporter to /usr/local/bin location.

- **sudo mv node_exporter-*.*-amd64/node_exporter /usr/local/bin/**

4] Create a node_exporter user to run the node exporter service.

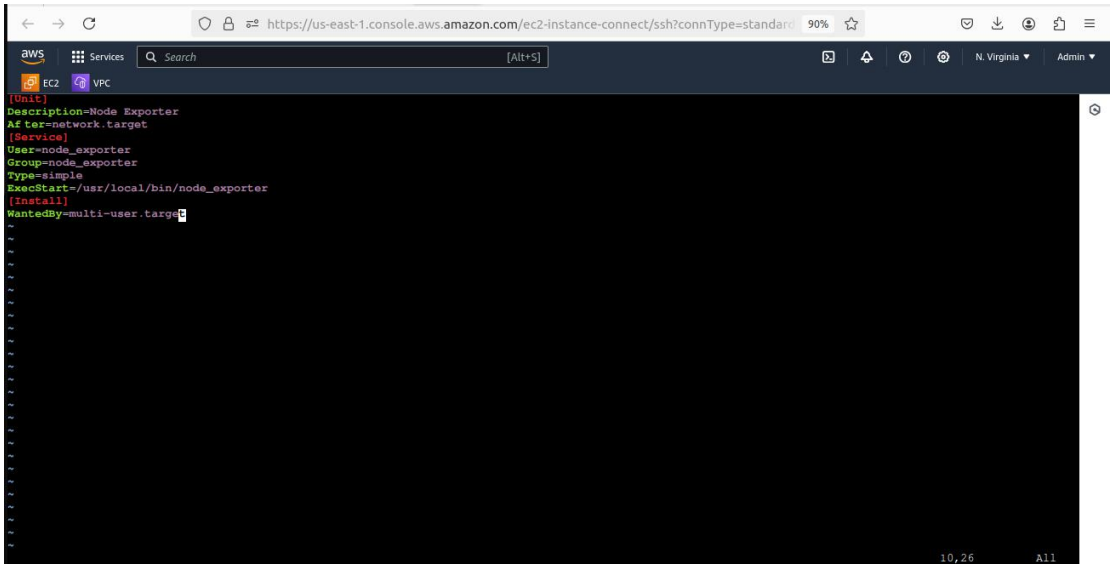
- **sudo useradd -rs /bin/false node_exporter**

5] Create a node_exporter service file in the /etc/systemd/system directory

- **sudo vim /etc/systemd/system/node_exporter.service**

```
[Unit]
Description=Node Exporter
After=network.target
[Service]
User=node_exporter
Group=node_exporter
```

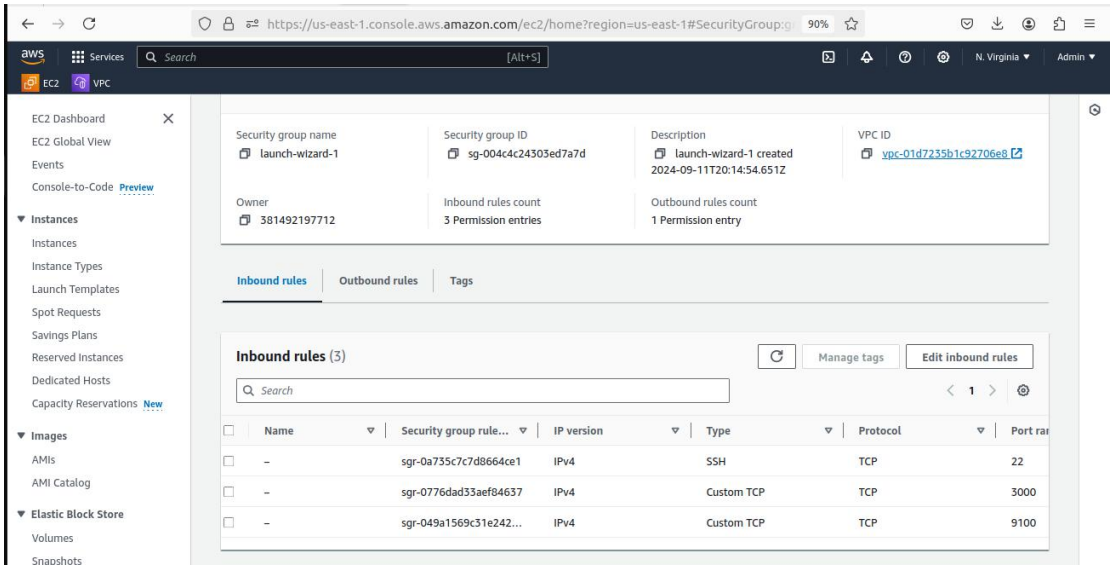
Type=simple
ExecStart=/usr/local/bin/node_exporter
[Install]
WantedBy=multi-user.target



6] Now lets start and enable the node_exporter service using below commands

- **sudo systemctl daemon-reload**
- **sudo systemctl enable node_exporter**
- **sudo systemctl start node_exporter**

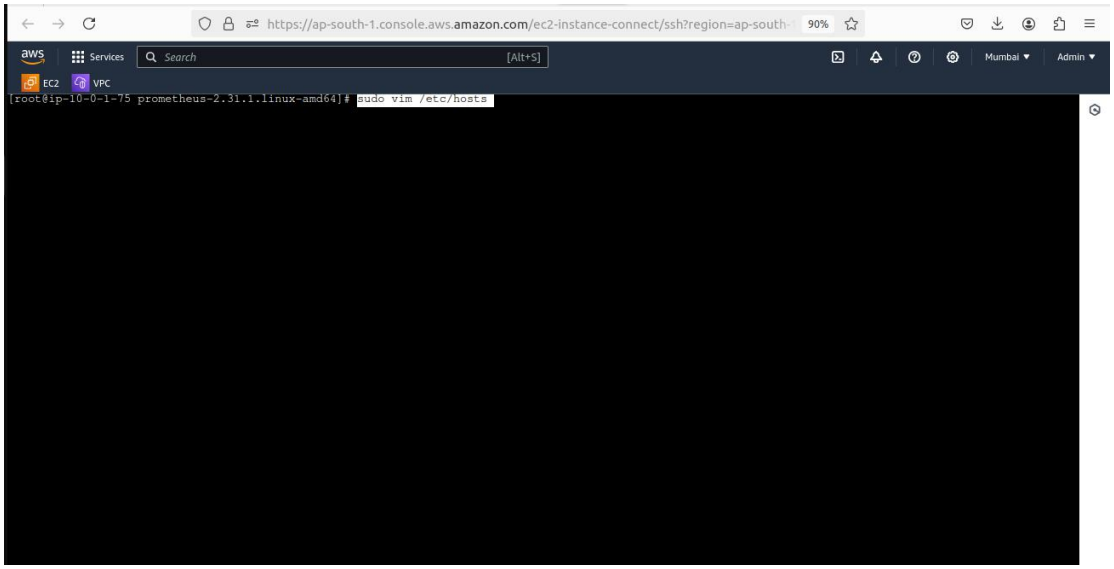
7] Security Groups Configured on EC2 Instances open port no. 9100

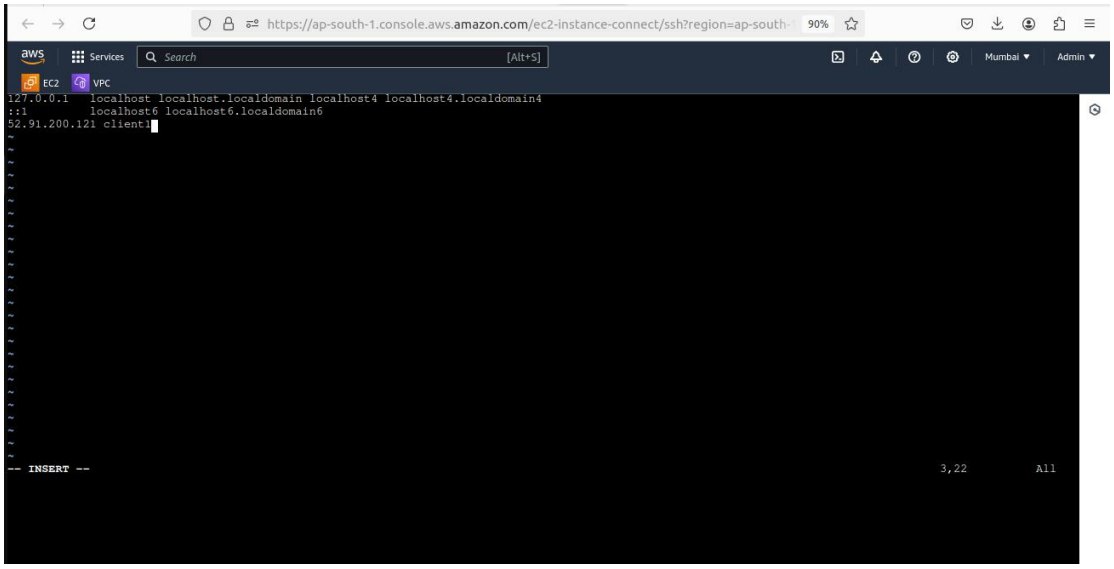


9] Now go to Prometheus Server & register new IP of client to /etc/hosts

- **sudo /etc/hosts**

(ex- 52.91.200.121 client1)





10] Add new config at prometheus server at /etc/prometheus/prometheus.yml

- **sudo vim /etc/prometheus/prometheus.yml**

(- targets: ['CLIENTHOSTNAME:NODE_EXPORTERPORT']## example
- targets: ['client1:9100'])

```
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"]
      - targets: ['client1:9100']
-- INSERT --                                                                    30,7          Bot
```

11] Now restart the Prometheus Service

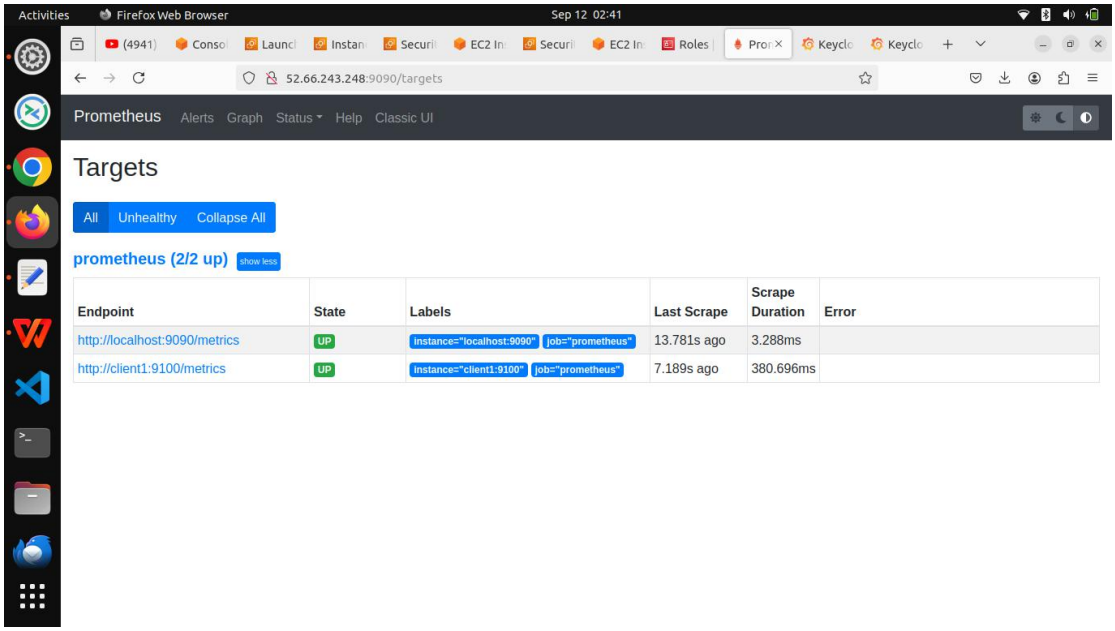
- **sudo systemctl restart prometheus**

```
   22  sudo vim/etc/hosts
   23  vim /etc/hosts
   24  sudo vim /etc/prometheus/prometheus.yml
   25  sudo systemctl restart prometheus
   26  history
[root@ip-10-0-1-75 prometheus-2.31.1.linux-amd64]#
```

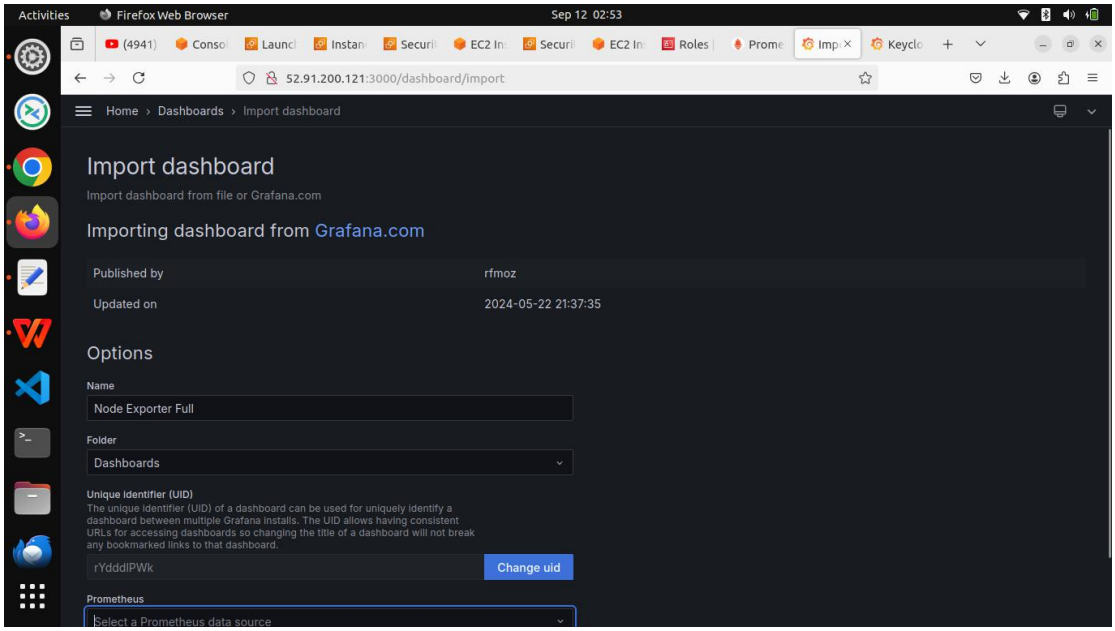
12] URL in your web browser to check weather our target is successfully scraped by Prometheus or not.

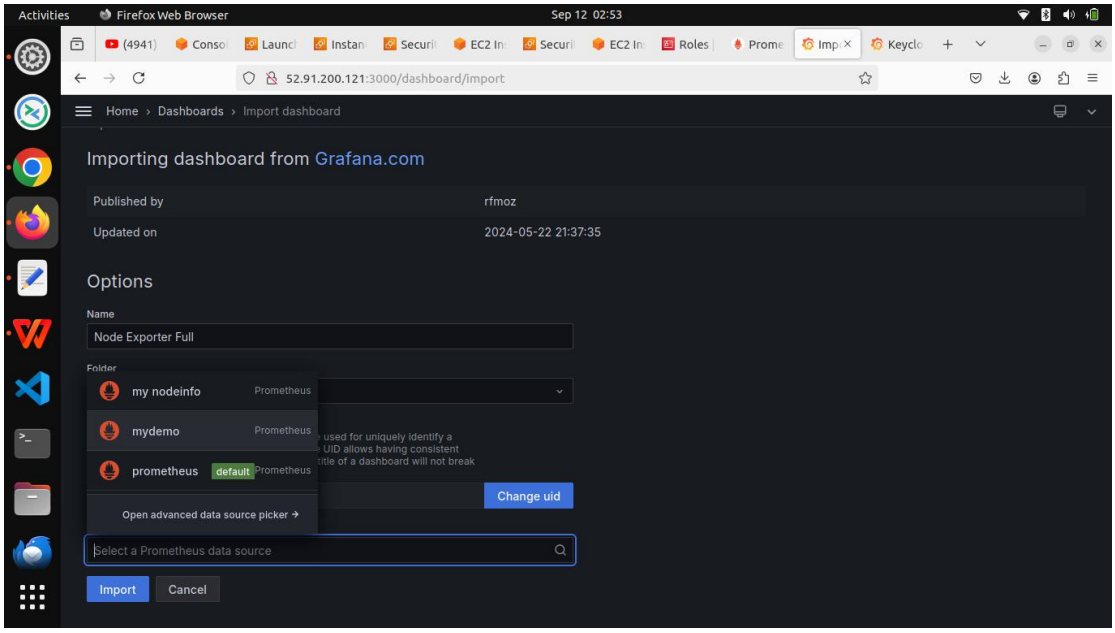
https://localhost:9100/targets

http://localhost:9090/targets

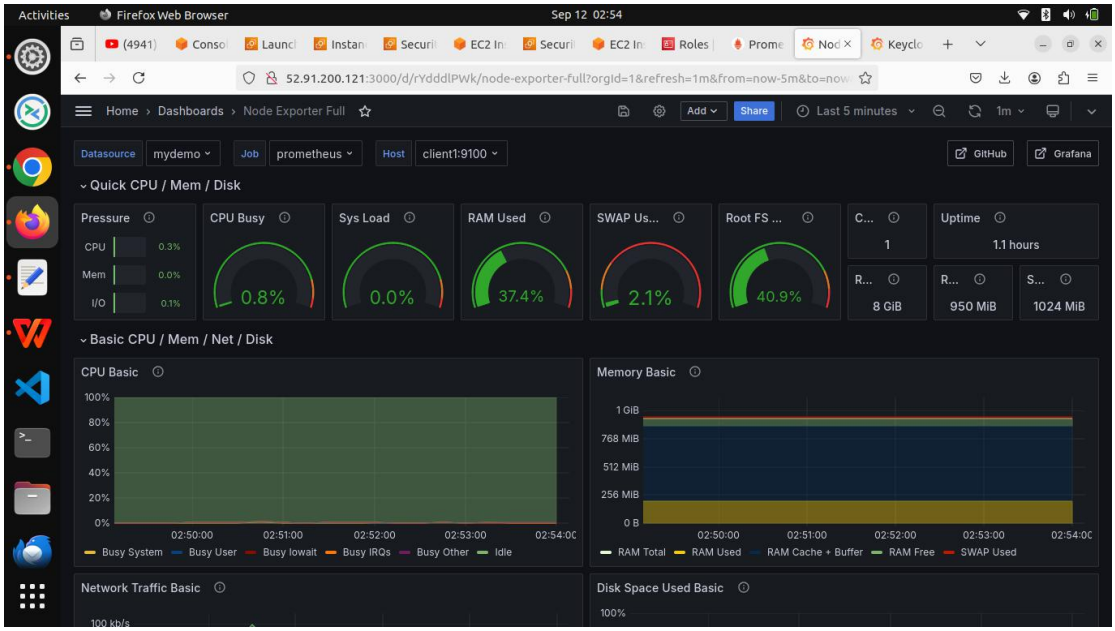


* Now create one dashboard as per above steps give name, folder and select data source and import.

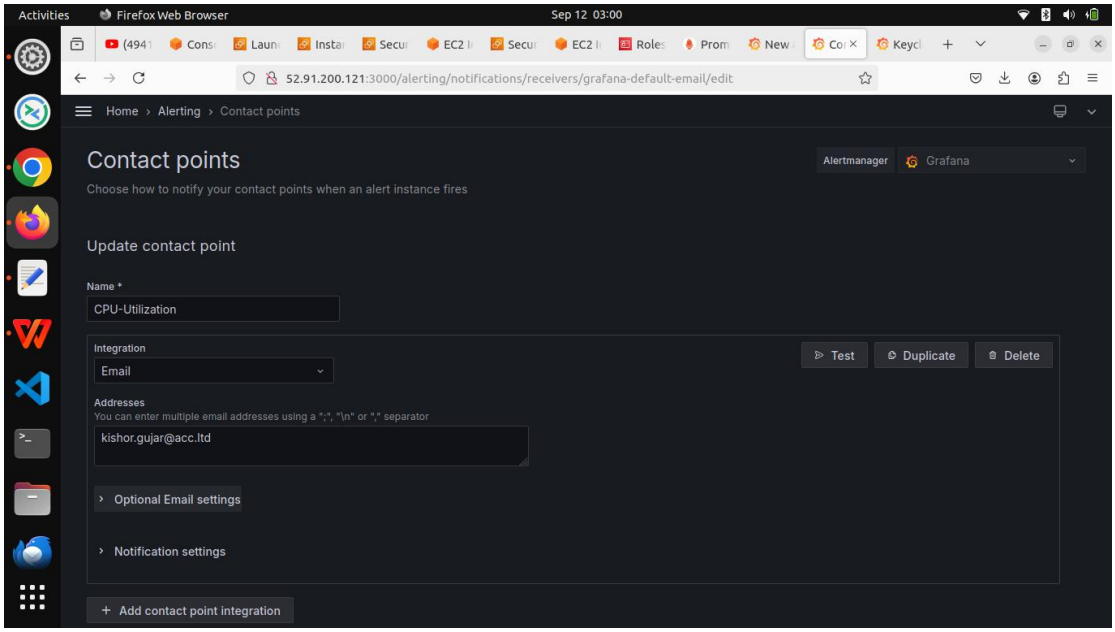




* Now go to Dashboards select dashboard which have create and now you can see our attractive dashboard with different metrics.



* Also you can set-up alerting for specific metrics with alerting feature of Grafana, you can set contact point as per your requirement.



* **We Succefully Create attractive dashboard with the help of Prometheus and Grafana**

Author: Kishor Gujar
Mail: kishorgujar2107@gmail.com