***Prometheus & Grafana for Server Metric Monitoring***

**Prometheus**

Prometheus is an open-source monitoring and alerting toolkit designed for reliability and scalability. it collects and stores time-series data by scraping metrics from configured endpoints at specified intervals. Prometheus is widely used for infrastructure monitoring, particularly for cloud-native environments, because of its pull-based architecture and powerful querying capabilities.

**Grafana**

Grafana is an open-source analytics and visualization platform that allows users to create dynamic, interactive dashboards from various data sources, including Prometheus. It enables real-time monitoring, alerting, and in-depth analysis of metrics through customizable panels and graphs.

**Metrics Monitoring in Prometheus & Grafana**

**Prometheus Metrics Collection:**

- Prometheus scrapes metrics from applications, servers, and exporters (like Node Exporter for system metrics).
- Stores metrics in a time-series database.
- Can trigger alerts based on defined conditions via alert manager.

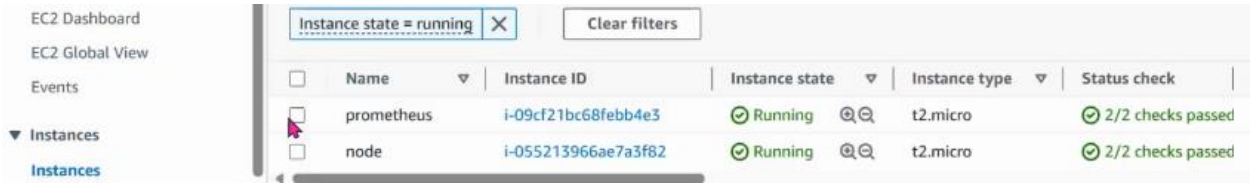
**Grafana Visualization:**

- Connects to Prometheus as a data source.
- Uses dashboards and panels to display real-time metrics.
- Supports alerting and notifications based on threshold values.

Common Use Cases for Server Monitoring:

**✓ CPU Usage Monitoring**
**✓ Memory & Disk Usage Tracking**
**✓ Network Traffic Analysis**
**✓ Process & Service Health Checks**
**✓ Custom Application Metrics**

Step: 1 -> We have to create two virtual server one is for prometheus and another one is for node exporter on the AWS.




Step: 2 -> Login as Ubuntu linux by using prometheus virtual server IP address and SSH port 22 and switch to root user.

```
login as: ubuntu
```

```
ubuntu@ip-172-31-45-164:~$ sudo -i
```

We will install the prometheus in /opt path by using wget command

```
root@ip-172-31-45-164:~# cd /opt
```

```
root@ip-172-31-45-164:/opt# wget https://github.com/prometheus/prometheus/releases/download/v2.45.0/prometheus-2.4
5.0.linux-amd64.tar.gz
```

We will unzip the prometheus file by using the below command

```
:/opt# tar -xvzf prometheus-2.45.0.linux-amd64.tar.gz
```

```
/opt/prometheus-2.45.0.linux-amd64# ll
```

```
./
../
LICENSE
NOTICE
console_libraries/
consoles/
prometheus*
prometheus.yml
promtool*
```

To open the Prometheus.yml file

```
vi prometheus.yml
```

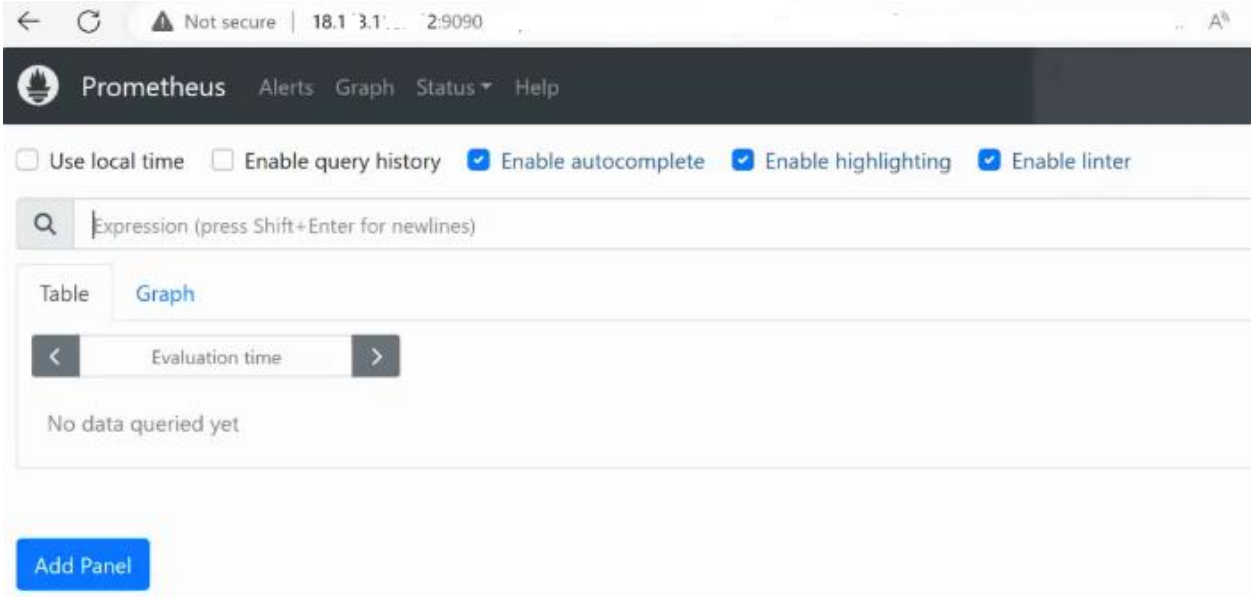
```
# my global config
global:
  scrape_interval: 15s # Set the scrape interval to every 15 seconds. Default is every 1 minute.
  evaluation_interval: 15s # Evaluate rules every 15 seconds. The default is every 1 minute.
  # scrape_timeout is set to the global default (10s).

# 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"]
```

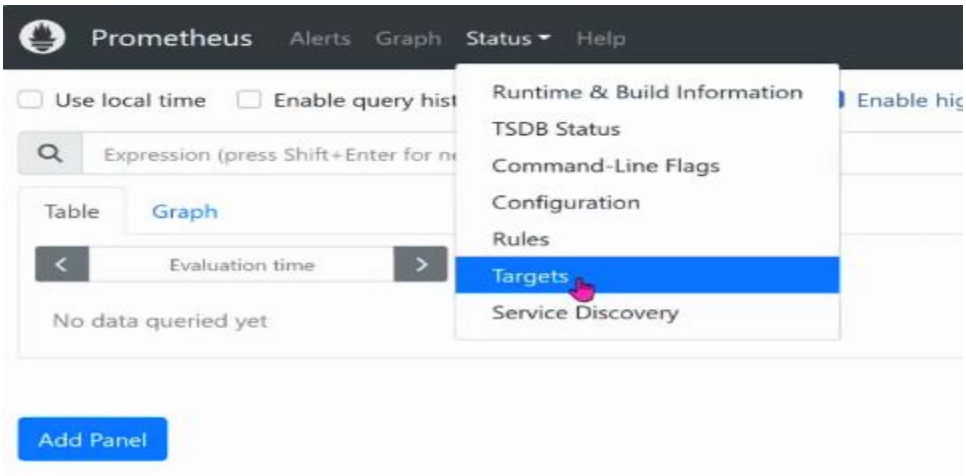
Now we will run the Prometheus by using the below command

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

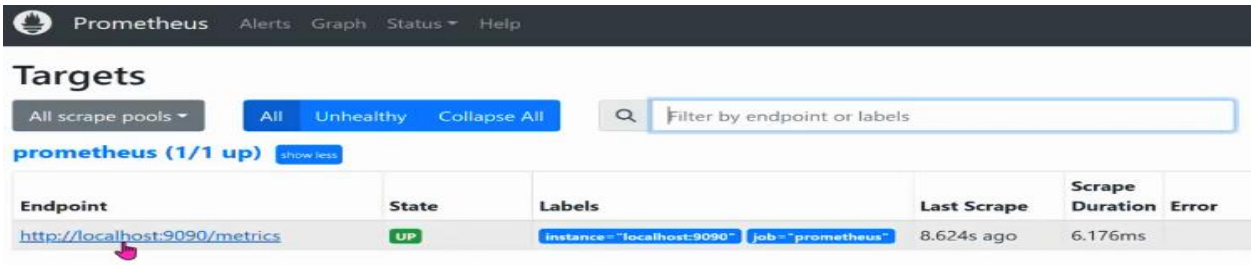
Prometheus run on the port 9090

We will login Prometheus webpage by using “Prometheus IP address:9090”




One local host is connected in target




Target is monitoring local host’s metrics.

Now we will login Ubuntu linux by using node exporter virtual server IP address and install node exporter.

```
ubuntu@ip-172-31-39-30:~$ sudo -i
root@ip-172-31-39-30:~# cd /opt
```

```
root@ip-172-31-39-30:/opt# wget https://github.com/prometheus/node_exporter/releases/download/v1.6.1/node_exporter
-1.6.1.linux-amd64.tar.gz
```

We will unzip node exporter file by using below command

```
# tar -xvzf node_exporter-1.6.1.linux-amd64.tar.gz
```

```
# cd node_exporter-1.6.1.linux-amd64/
```

```
# ll
```

```
./
../
LICENSE
NOTICE
node_exporter*
```

We will start the node exporter by using the below command & Node Exporter port number – 9100

```
# ./node_exporter &
```

Now we have successfully installed node exporter in the slave machine.

Prometheus will monitor theirs slave machine metrics

Open the Prometheus.yml file from the Promethus master node

```
root@ip-172-31-45-164:/opt/prometheus-2.45.0.linux-amd64# vi prometheus.yml
```

We will add slave machine IP address:9100 node exporter port on the targets and save & quit (!wq). If we have to monitor n number of slave machines metric, we just add the slave machines IP addresses:9100 node exporter port one by one on the targets.

```
# 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: "node"

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

    static_configs:
      - targets: ["13.222.13.2.5:9100"]
~
```

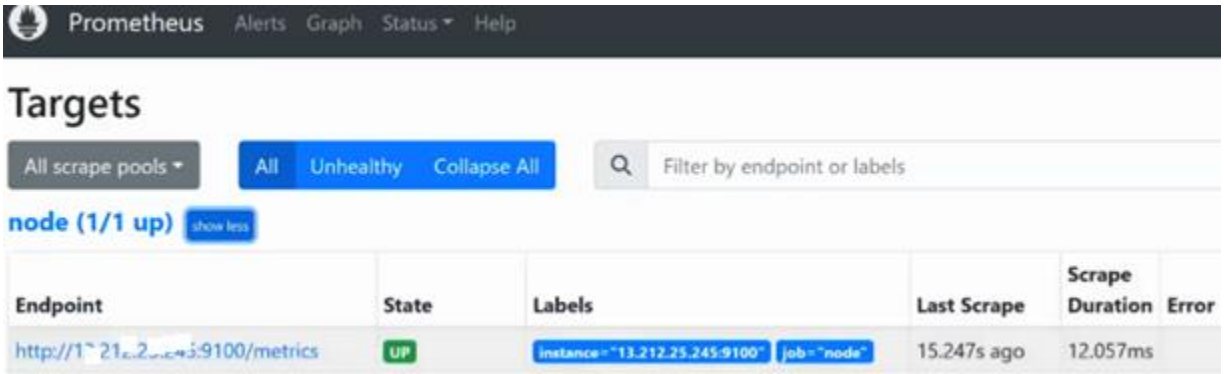
To check & stop the processor ID under prometheus by using the below command before restart

```
# ps -ef| grep prometheus
```

```
root         1440    1372  0 05:15 pts/1    00:00:00 ./prometheus --config.file=prometheus.yml
root         1489    1372  0 05:26 pts/1    00:00:00 grep --color=auto prometheus
root@ip-172-31-45-164:/opt/prometheus-2.45.0.linux-amd64# kill -9 1440
root@ip-172-31-45-164:/opt/prometheus-2.45.0.linux-amd64# ps -ef| grep prometheus
root         1493    1372  0 05:27 pts/1    00:00:00 grep --color=auto prometheus
[1]+  Killed                  ./prometheus --config.file=prometheus.yml
```

We will restart the Prometheus

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




Node has been successfully added on the Prometheus

We will install the Grafana tool on the master machine for the visualization dashboard setup

```
wget https://dl.grafana.com/oss/release/grafana-10.1.1.linux-amd64.tar.gz
```

```
tar -zxvf grafana-10.1.1.linux-amd64.tar.gz
```

```
:# ll
:# ll
```

```
./
../
grafana-10.1.1/
grafana-10.1.1.linux-amd64.tar.gz
prometheus-2.45.0.linux-amd64/
prometheus-2.45.0.linux-amd64.tar.gz
```

```
:# cd grafana-10.1.1/
```

```
# ll
```

```
Dockerfile
LICENSE
NOTICE.md
README.md
VERSION
bin/
conf/
docs/
npm-artifacts/
packaging/
plugins-bundled/
public/
storybook/
```

```
# cd bin
```

```
# ll
```

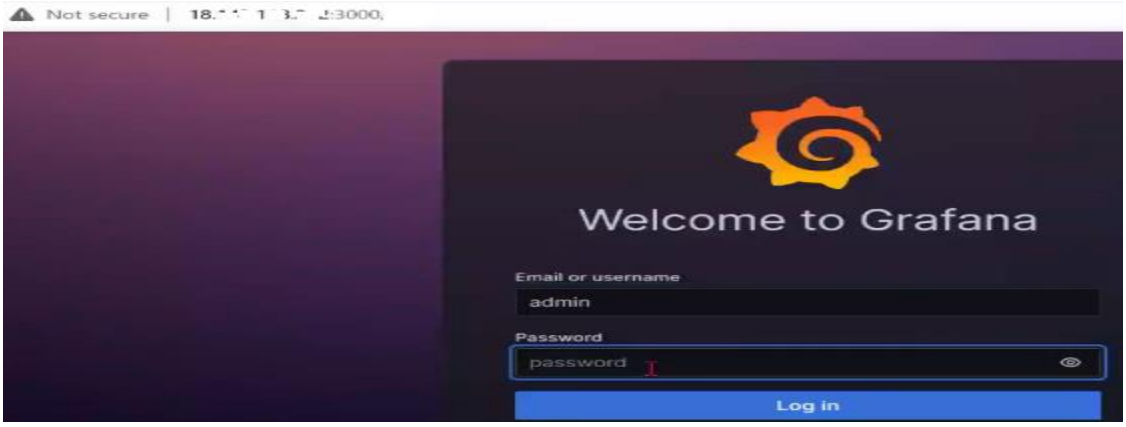
```
grafana*
grafana-cli*
grafana-server*
```

Now we will start the Grafana server by using the below command

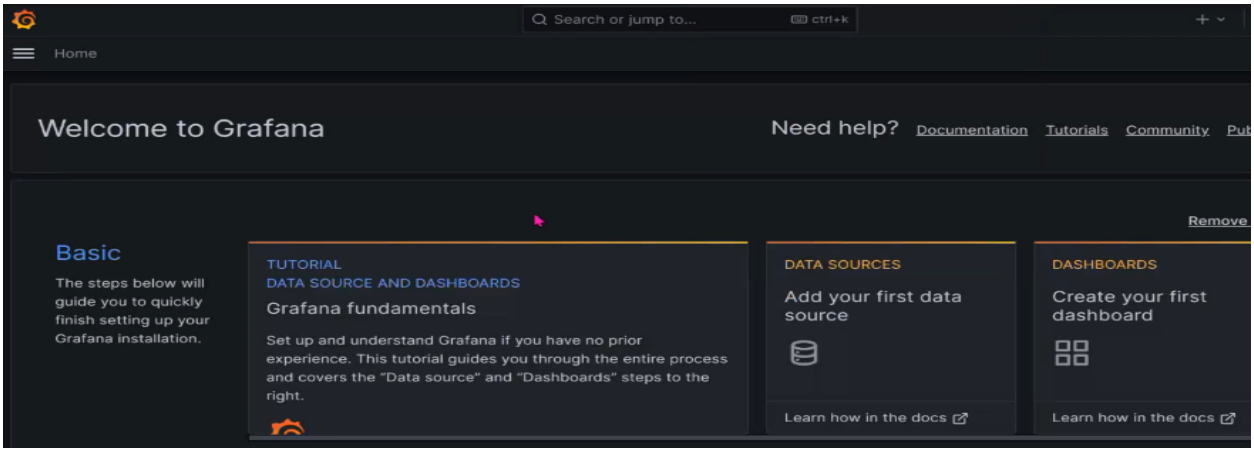
```
./grafana-server &
```

Grafana runs on the port 3000

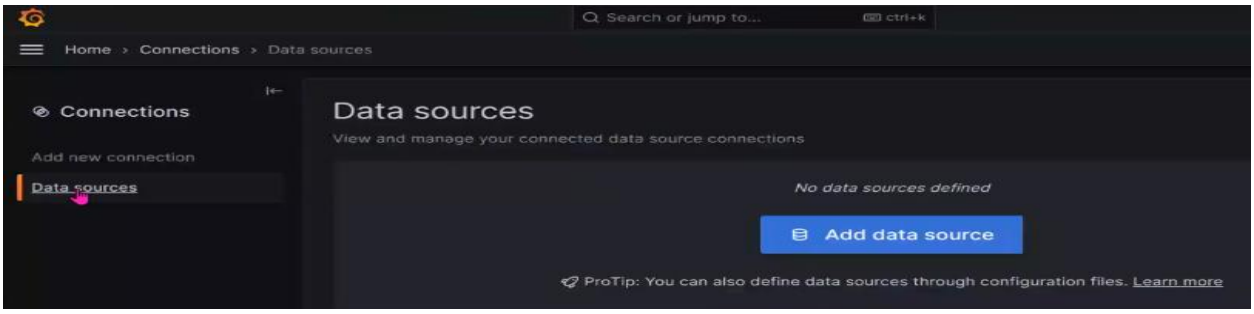
We will login grafana webpage by using “Master Machine IP address: 3000” with default username and password – “admin”



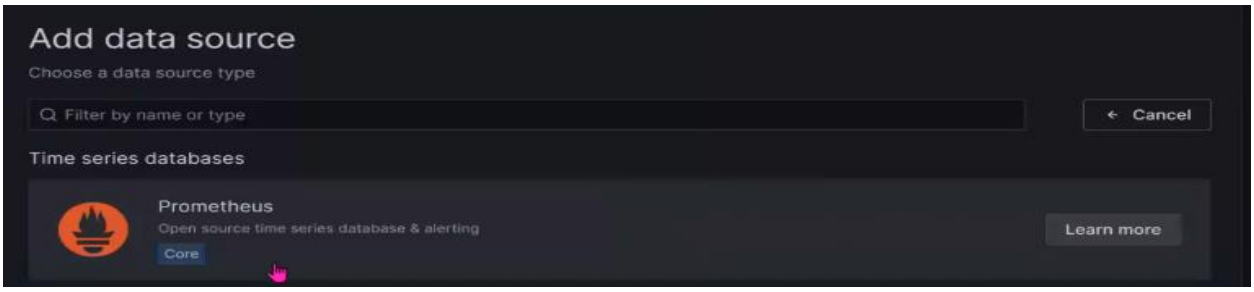

Set new password



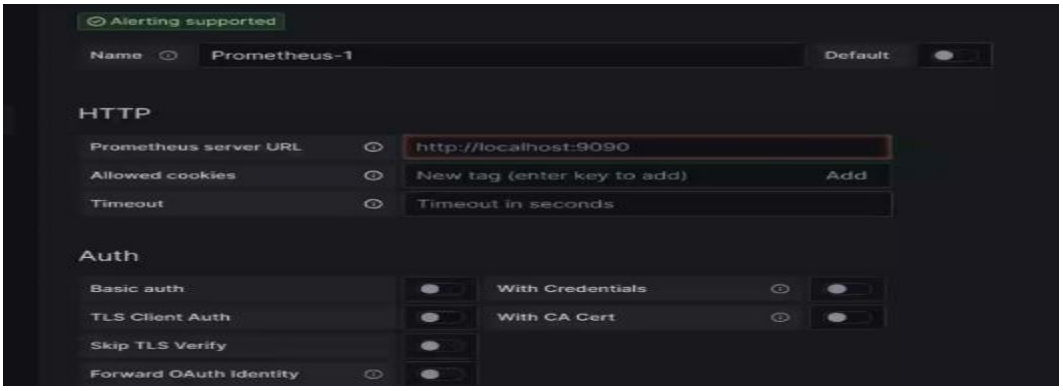

Click on -> Home -> Connection -> Data source -> Add Data Source



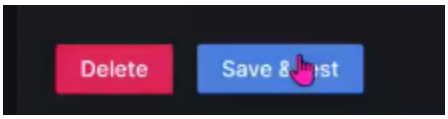

Click on "Prometheus"

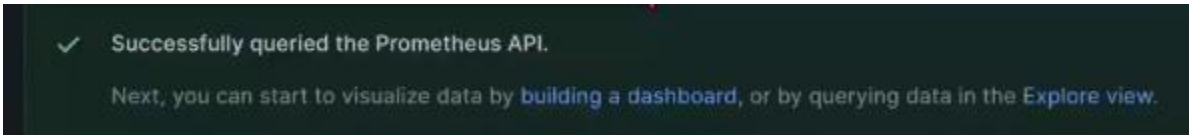



Give the Prometheus URL

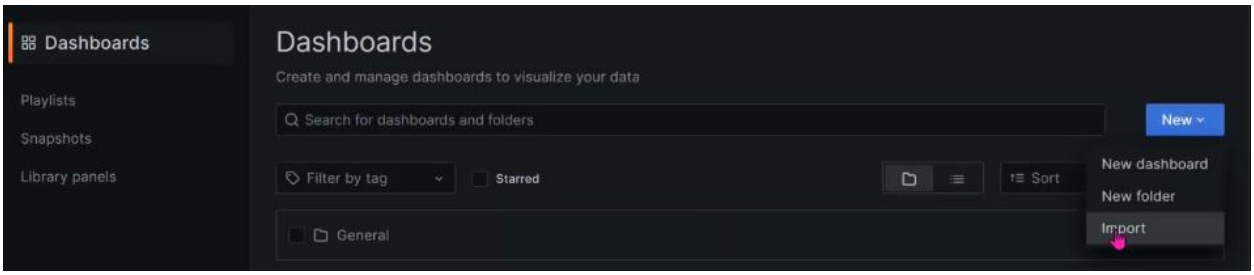


Click on “Test & Save”

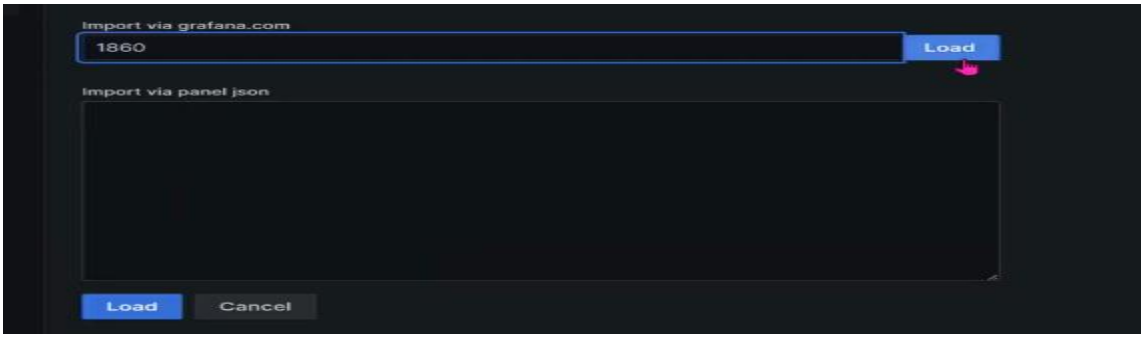


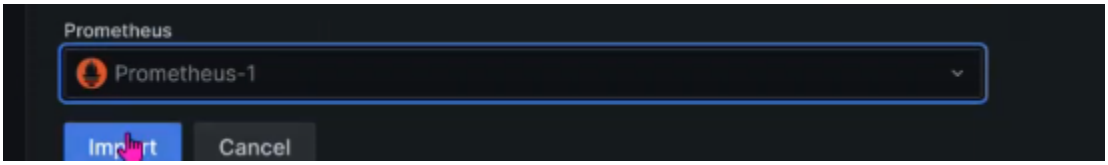




Click on Home -> Dashboards -> New -> Import

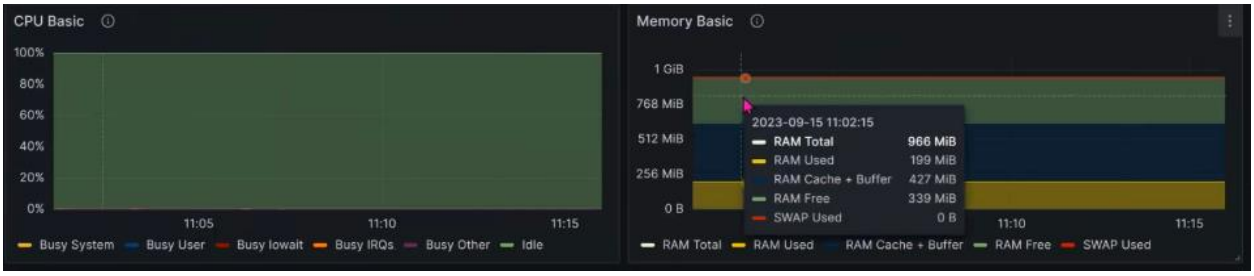


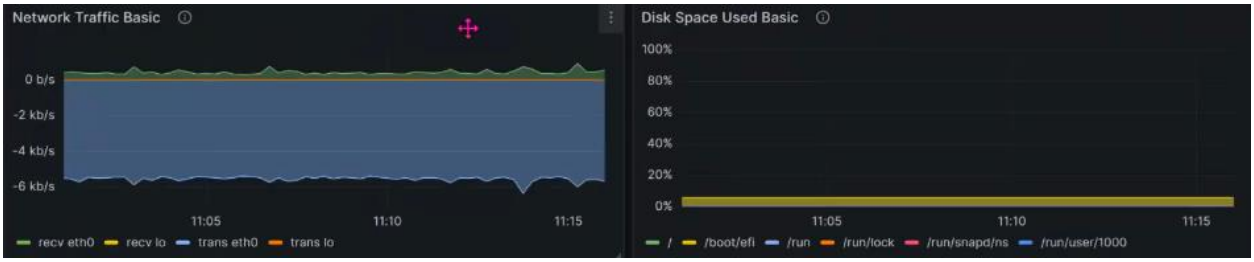

Enter the graph ID 1860 then click on “Load” and “Import”

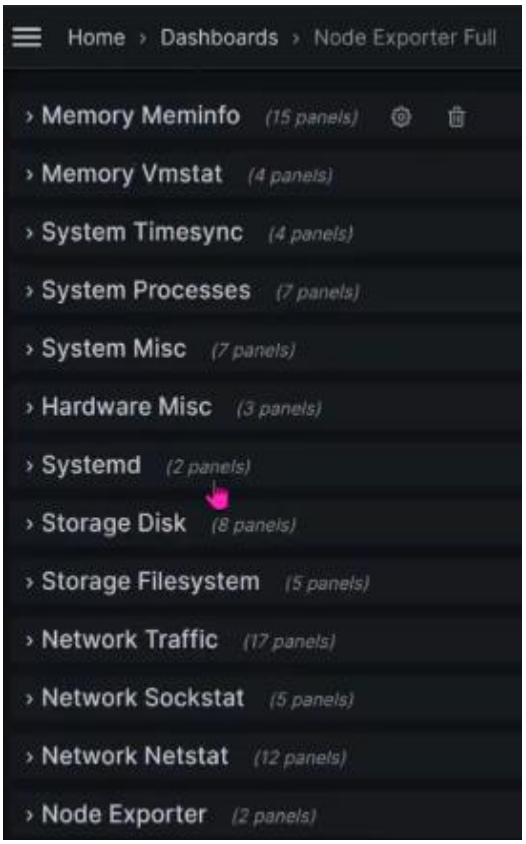



Now the Grafana dashboard has been successfully created.


CPU Basic
100%
80%
60%
40%
20%
0%
11:05
11:10
11:15
Busy System
Busy User
Busy Iowait
Busy IRQs
Busy Other
Idle
Memory Basic
1 GiB
768 MiB
512 MiB
256 MiB
0 B
2023-09-15 11:02:15
RAM Total 966 MiB
RAM Used 199 MiB
RAM Cache + Buffer 427 MiB
RAM Free 339 MiB
SWAP Used 0 B
11:10
11:15
RAM Total
RAM Used
RAM Cache + Buffer
RAM Free
SWAP Used


Network Traffic Basic
0 b/s
-2 kb/s
-4 kb/s
-6 kb/s
11:05
11:10
11:15
recv eth0
recv lo
trans eth0
trans lo
Disk Space Used Basic
100%
80%
60%
40%
20%
0%
11:05
11:10
11:15
/
/boot/efi
/run
/run/lock
/run/snapd/ns
/run/user/1000


Home › Dashboards › Node Exporter Full
› Memory Meminfo (15 panels)
› Memory Vmstat (4 panels)
› System Timesync (4 panels)
› System Processes (7 panels)
› System Misc (7 panels)
› Hardware Misc (3 panels)
› Systemd (2 panels)
› Storage Disk (8 panels)
› Storage Filesystem (5 panels)
› Network Traffic (17 panels)
› Network Sockstat (5 panels)
› Network Netstat (12 panels)
› Node Exporter (2 panels)