Task - 2

Prometheus and Grafana

Prometheus:

Prometheus is an open-source systems monitoring and alerting toolkit originally built at SoundCloud. Here in Prometheus, we can monitor the Infrastructure of the server.

Prometheus collects and stores its metrics as time series data, i.e. metrics information is stored with the timestamp at which it was recorded, alongside optional key-value pairs called labels.

Grafana:

Grafana is an open-source visualization and analytics software. No matter where your data is stored, it can be queried, visualized, and explored. In plain English, it provides you with tools to turn your time-series database (TSDB) data into beautiful graphs and visualizations.
Here we can add different data sources and we can customize the dashboard as per our requirement.

Prometheus setup steps:

Step:1

1. Launch 2 instances one is Prometheus another one is Node Exporter.
2. Login to the Prometheus VM using Putty.
3. Standard things to do...
    i. Create user
    ii. Update VM
4. wget https://github.com/prometheus/prometheus/releases/download/v2.42.0/prometheus-2.42.0.linux-amd64.tar.gz (Prometheus file to download)
5. **tar -xvzf Prometheus-2.42.0.linux-amd64.tar.gz (to unzip the file)**
6. cd prometheus-2.42.0.Linux-amd64
7. vi prometheus.yml (here we can do some configuration like time scrap, targets, etc...)
8. ./prometheus (to start prometheus)
9. Prometheus will be running in port number 9090 and in the background, the instance has to be active, if the instance stops Prometheus will also stop. For this, we can make a duplicate file to run the instance in the background but it will not be recommended so instead of this I am gonna write a Prometheus as a service.
10. To run Prometheus as a service, First am pasting all the Prometheus files into **user/local/bin**, there I am going to write a yml file and going to run it as a service.

11. **cp -r . /usr/local/bin/Prometheus** (here we are creating a Prometheus directory to keep all files in the same folder.)
12. **vi /etc/systemd/system/prometheus.service** (here we need to attach the file below)

```
[Unit]
Description=Prometheus Service #any as per wish
After=network.target

[Service]
Type=simple
ExecStart=/usr/local/bin/prometheus/prometheus
--config.file=/usr/local/bin/prometheus/prometheus.yml

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

13. **systemctl daemon-reload** (used to reload the systemd manager configuration.)
14. **service prometheus start** (to start prometheus service)
15. **systemctl enable prometheus** (to enable prometheus service)
16. **service Prometheus status** (to check the status of the Prometheus)

Good to Go, We had configured the complete prometheus setup. Prometheus is running perfectly.

Snips as per the steps followed above:

```
● prometheus.service - Prometheus Service
     Loaded: loaded (/etc/systemd/system/prometheus.service; disabled; vendor preset: enabled)
     Active: active (running) since Tue 2023-07-25 08:23:11 UTC; 6h ago
   Main PID: 24308 (prometheus)
      Tasks: 8 (limit: 1141)
     Memory: 50.9M
        CPU: 17.904s
     CGroup: /system.slice/prometheus.service
             └─24308 /usr/local/bin/prometheus/prometheus --web.page-title=Prometheus-Demo --config.file=/usr/local/bin/prometheus/pr>
```

Prometheus is running perfectly.

Step:2 (Node Exporter)

1. Login to node exporter VM using putty.
2. Standard things to do…
    i. Create user
    ii. Update VM
3. Wget https://github.com/prometheus/node_exporter/releases/download/v1.5.0/node_exporter-1.5.0.linux-amd64.tar.gz
4. ./node_exporter (to start node_exporter)
5. Node_exporter will be running in port number 9100 and in the background, the instance has to be active, if the instance stops node_exporter will also stop. For this, we can make a duplicate file to run the instance in the background but it will not be recommended so instead of this I am gonna write a node_exporter as a service.
6. To run node_exporter as a service, First am copying all the node_exporter files into **user/local/bin**, there I am going to write a yml file and going to run it as a service.
7. **cp -r . /usr/local/bin/node** (here we are creating a node directory to keep all files in the same folder.)
8. **vi /etc/systemd/system/node_exporter.service** (here we need to attach the file below)

```
[Unit]
Description=Node_exporter Service #any as per wish
After=network.target

[Service]
Type=simple
ExecStart=/usr/local/bin/node_exporter/node_exporter
--config.file=/usr/local/bin/node_exporter/node_exporter.yml

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

9. **systemctl daemon-reload** (used to reload the systemd manager configuration.)
10. **service node_exporter start** (to start node_exporter service)
11. **systemctl enable node_exporter** (to enable node_exporter service)
12. **service node_exporter status** (to check the status of the node_exporter)

Now, node_exporter is also ready.

Add node_exporter as a target file into the prometheus

Why node_exporter?

Node_exporter is not a must but it helps easily to find and manage the metrics.

1. System Metrics Collection
2. Prometheus Compatibility
3. Scalability and Efficiency
4. Exposes Metrics via HTTP

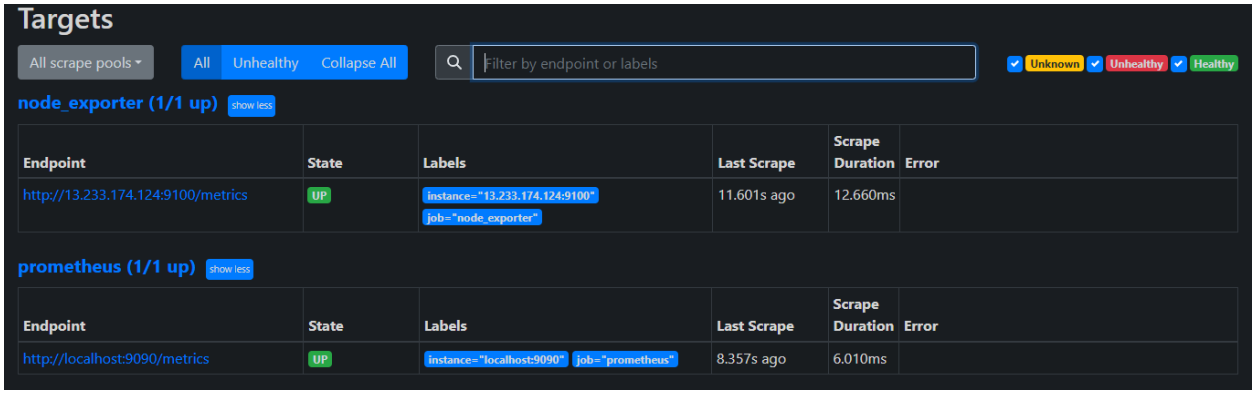
1. Once done with the node_exporter setup, **we need to make node_exporter communicate with the prometheus server.**
2. Open Prometheus VM and go to the path where prometheus binary files are available and there will **prometheus.yml** file.
3. **vi prometheus.yml**

```
prometheus
root@ip-172-31-41-95:/usr/local/bin# cd prometheus/
root@ip-172-31-41-95:/usr/local/bin/prometheus# ls
LICENSE  NOTICE  console_libraries  consoles  data  prometheus  prometheus.yml  promtool
root@ip-172-31-41-95:/usr/local/bin/prometheus#
```

4. In the yml file, we need to add a new job name as node_exporter.

```
  - job_name: "prometheus"

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

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

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

    static_configs:
      - targets: ["13.233.174.124:9100"]
```

5. Now can just save the file.
6. Map into the prometheus web page and check the target file, there the node_exporter is added as a target file and prometheus is monitoring the self server by default.



Node_exporter added to target file

Prometheus Graph

Command Line Flags → Exact path to configure

vi /etc/systemd/system/prometheus.service

Configuration:

```
global:
  scrape_interval: 15s
  scrape_timeout: 10s
  evaluation_interval: 15s
alerting:
  alertmanagers:
  - follow_redirects: true
    enable_http2: true
    scheme: http
    timeout: 10s
    api_version: v2
    static_configs:
    - targets: []
scrape_configs:
- job_name: prometheus
  honor_timestamps: true
  scrape_interval: 15s
  scrape_timeout: 10s
  metrics_path: /metrics
  scheme: http
  follow_redirects: true
  enable_http2: true
  static_configs:
  - targets:
    - localhost:9090
- job_name: node_exporter
  honor_timestamps: true
  scrape_interval: 15s
  scrape_timeout: 10s
  metrics_path: /metrics
  scheme: http
  follow_redirects: true
  enable_http2: true
  static_configs:
  - targets:
    - 13.233.174.124:9100
```

Grafana

Why Grafana?

Grafana is an open-source visualization and analytics software. No matter where your data is stored, it can be queried, visualized, and explored. In plain English, it provides you with tools to turn your time-series database (TSDB) data into beautiful graphs and visualizations.

Grafana is better suited for applications that require continuous real-time monitoring metrics like CPU load, memory, etc.

Steps to setup:

1. wget https://dl.grafana.com/enterprise/release/grafana-enterprise-10.0.2.linux-amd64.tar.gz
2. tar -zxvf grafana-enterprise-10.0.2.linux-amd64.tar.gz

Using this need to install the grafana in to the prometheus server

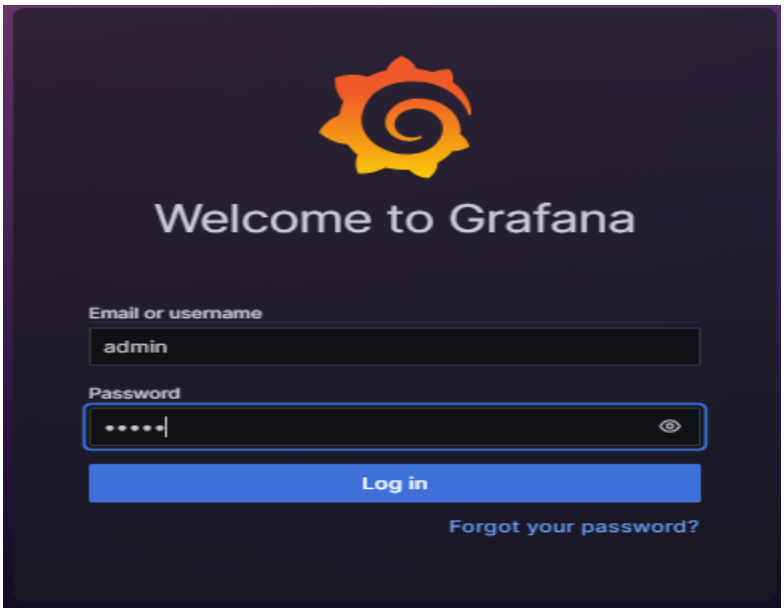
3. sudo systemctl start grafana-server
4. sudo systemctl stop grafana-server
5. sudo systemctl status grafana-server
6. sudo systemctl enable grafana-server

Use above command to install, start, stop, enable, status the grafana server.

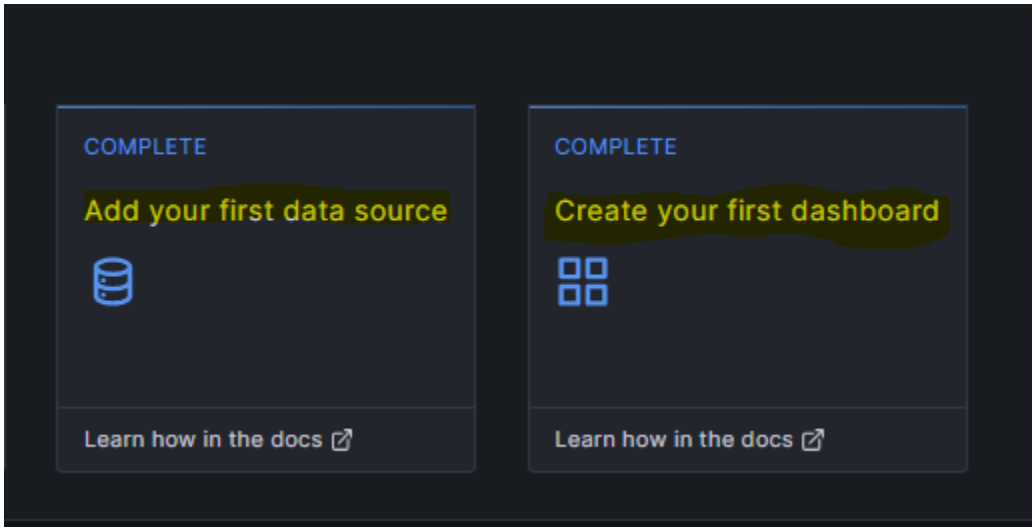
7. http://13.233.33.223:3000/ grafana will work in port 3000

```
INFO [07-26|05:23:24] Completed cleanup jobs                    logger=cleanup duration=1.622756ms
INFO [07-26|05:23:24] Update check succeeded                    logger=grafana.update.checker duration=10.766765ms
INFO [07-26|05:23:24] Update check succeeded                    logger=plugins.update.checker duration=284.354943ms
INFO [07-26|05:33:24] Completed cleanup jobs                    logger=cleanup duration=1.666494ms
INFO [07-26|05:33:24] Update check succeeded                    logger=grafana.update.checker duration=10.302865ms
INFO [07-26|05:33:24] Update check succeeded                    logger=plugins.update.checker duration=285.927297ms
INFO [07-26|05:43:24] Completed cleanup jobs                    logger=cleanup duration=1.83989ms
INFO [07-26|05:43:24] Validated license token                   logger=licensing appURL=http://localhost:3000/ source=disk status=NotFo
und
WARN [07-26|05:43:24] failed to load or validate token          logger=licensing.renewal err="license token file not found: /root/grafa
na/grafana-10.0.2/data/license.jwt"
INFO [07-26|05:43:24] Update check succeeded                    logger=grafana.update.checker duration=13.113683ms
INFO [07-26|05:43:24] Update check succeeded                    logger=plugins.update.checker duration=288.245552ms
```

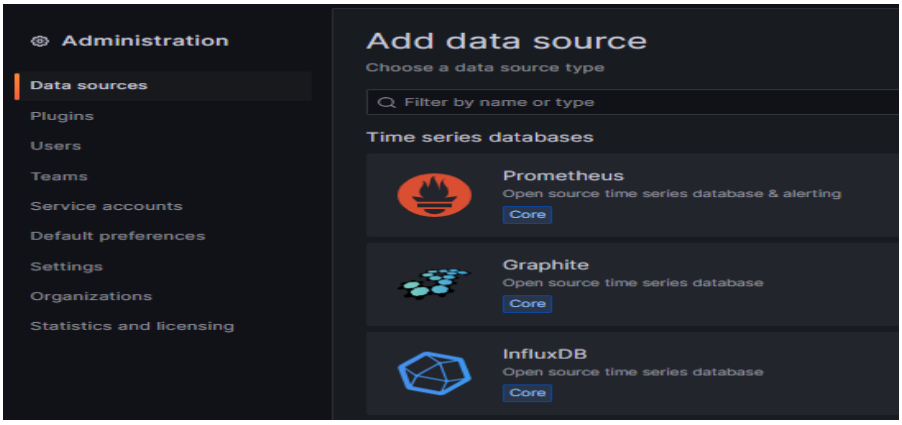
- ➔ Here you can see grafana is running perfectly and it is running in port 3000.
- ➔ Using the IP you can loging to the Grafana server.



- Login page be like.
- When we are logging in first time, need to enter admin as a username as well as password.
- Then you will find a page to create a new password as you like.

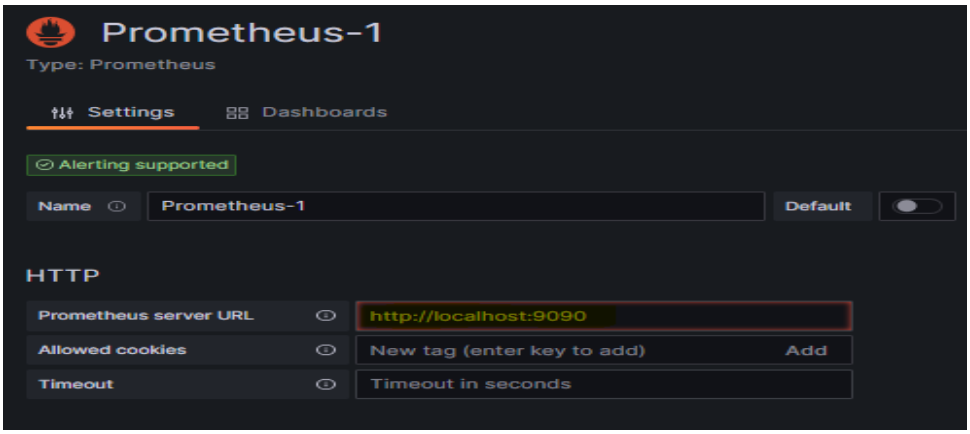


- In the home page, there will be option to add data source and to create a dash board as you like and we can import the dash board from the Grafana lab.



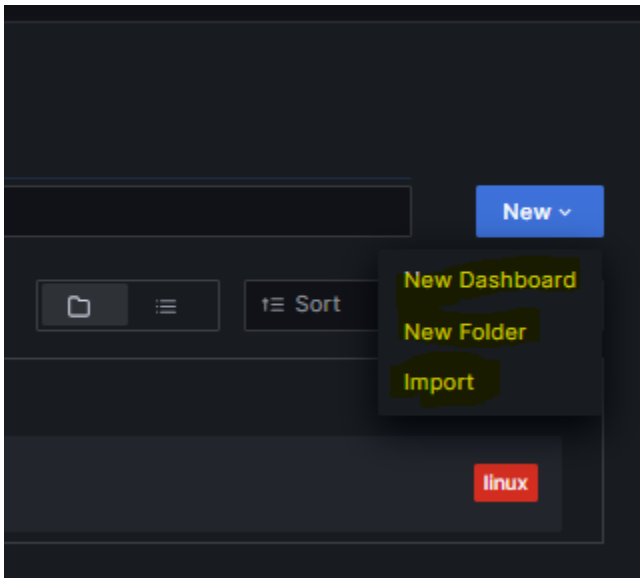
➔ Data source can be choosed as per the need.

➔ For server monitoring, I will go with prometheus.



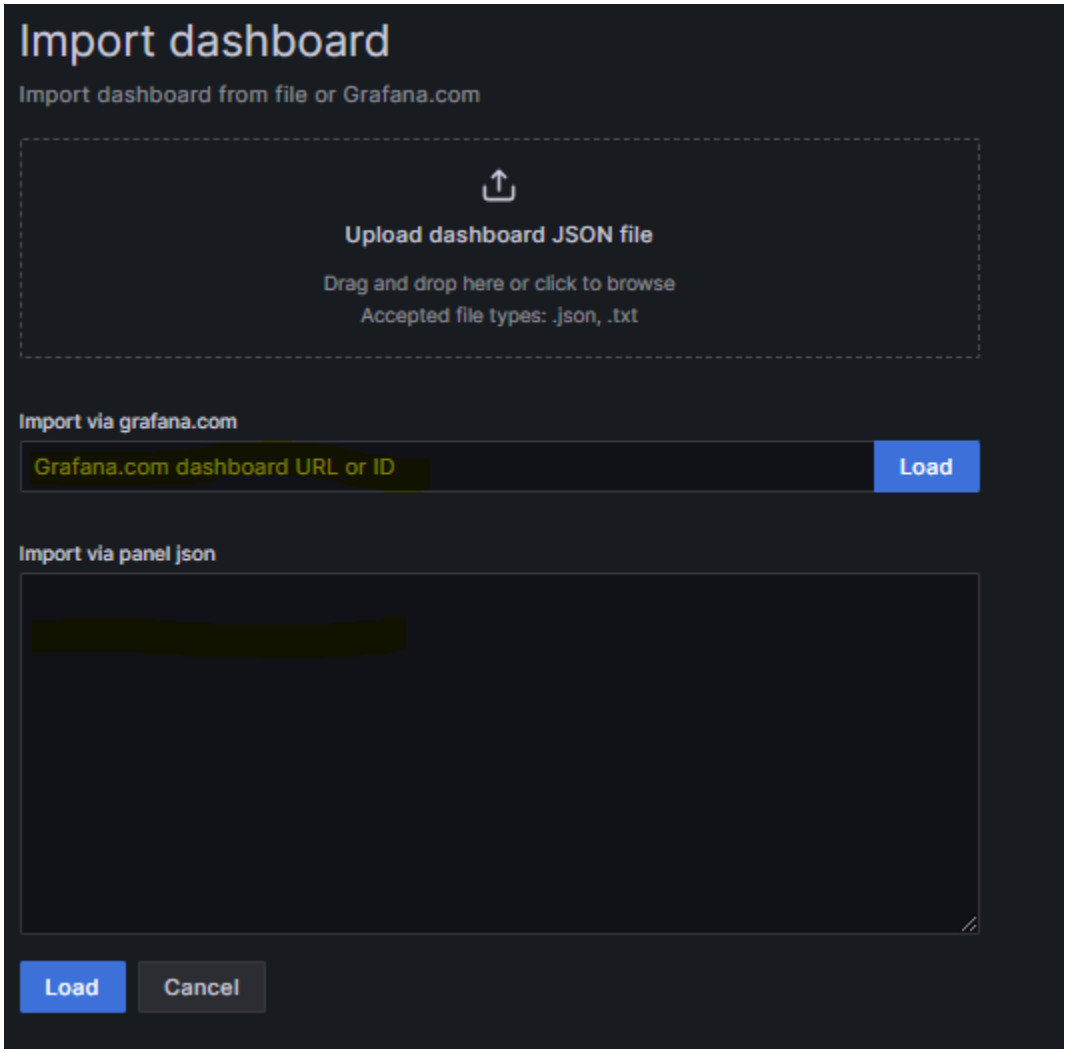
➔ Here need to give URL of the prometheus server.

➔ Finally need to save and test the data source

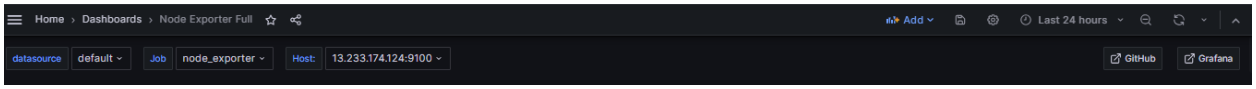


➔ Here I gone through Import option. Dashboard will be import from the Graana Lab.

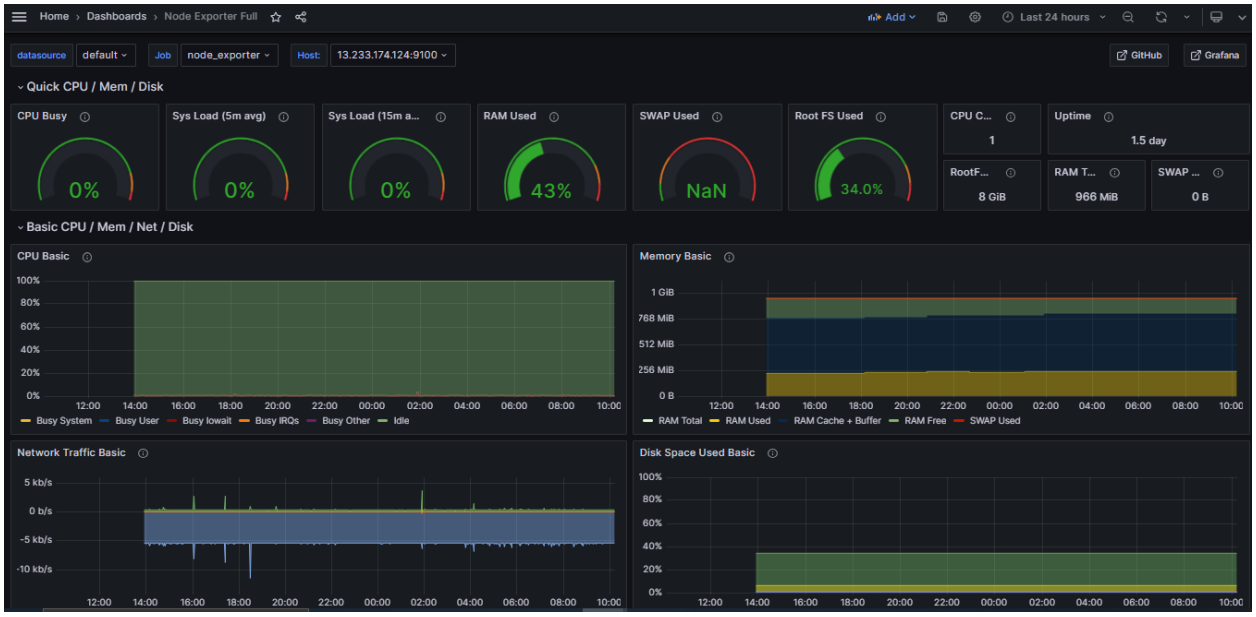
- https://grafana.com/grafana/dashboards/ using this link we can Import the dashboard as we want.
- There will be 2 Import dashboard templates one is Dashboard ID and the other is JSON.



- Here you can see, Import via JSON or using Dashboard ID.



- Here we can select the datasource, Job, Host, time range, time interval, dash board setting and many.



Final Dashboad be like and here we can monitor our server.