

PROMETHEUS (CM)

1) What is Prometheus?

- A) Prometheus is an open-source monitoring and alerting toolkit originally developed by SoundCloud in 2012 and later donated to the Cloud Native Computing Foundation (CNCF) in 2016. It is widely used in the field of cloud-native and containerized environments for monitoring and managing the health and performance of various systems and applications.

Key features of Prometheus include:

Time Series Database: Prometheus stores all the collected monitoring data as time-series, allowing it to track and analyse metrics over time.

Metrics Collection: Prometheus pulls metrics data from various sources, such as applications, services, and system components, through an HTTP-based pull model. These sources need to expose their metrics in a specific format called the Prometheus exposition format.

Query Language: Prometheus provides a powerful query language called PromQL (Prometheus Query Language) that allows users to perform complex queries and aggregations on the collected metrics.

Alerting: Prometheus has a built-in alerting mechanism that allows users to define alerting rules based on specific conditions and thresholds. When a rule is triggered, it can send alerts to various channels like email, Slack, or other alerting systems.

Service Discovery: Prometheus supports service discovery, making it easier to monitor dynamic systems like container orchestration platforms (e.g., Kubernetes) where instances of applications and services can change rapidly.

Grafana Integration: Prometheus is often used in combination with Grafana, a popular open-source visualization tool. Grafana can connect to Prometheus and create interactive, visually appealing dashboards to display monitoring data.

Prometheus has become an essential part of many cloud-native deployments due to its flexibility, scalability, and ability to handle high-dimensional data. It plays a crucial role in helping developers and operators gain insights into the performance and health of their systems, troubleshoot issues, and ensure reliability in modern, distributed infrastructure environments.

2) What is prometheus exporter?

- A) A Prometheus Exporter is a part of software that allows it to fetch statistics from another, non-prometheus system. It converts those statistics into prometheus metrics, using a client library. You can start a web server which exposes a /metrics URL, and can see that URL display the system metrics.

3) How To Install & configure prometheus, node_exporter and grafana on Ubuntu 20.04/22.04?

A) For standalone installation of prometheus, node_exporter and grafana on UBUNTU 20.04/22.04 follow the following steps

To properly configure Prometheus, you must follow the below mentioned steps.

STEP-1: UPDATE SYSTEM PACKAGES

sudo apt update && sudo apt upgrade

STEP-2: CREATE PROMETHEUS USER AND GROUP

First, create the “prometheus” system group using the following command

sudo groupadd --system prometheus

Next, create the “prometheus” system user using the following command

sudo useradd -s /sbin/nologin --system -g prometheus prometheus

STEP 3: CREATE DIRECTORIES FOR PROMETHEUS

sudo mkdir /var/lib/prometheus

Next, create the primary configuration files directory for Prometheus using the following command. All Prometheus related data will be stored in this folder.

for i in rules rules.d files_sd; do sudo mkdir -p /etc/prometheus/\${i}; done

STEP 4: DOWNLOAD PROMETHEUS

First check whether curl is available or not in UBUNTU, as follows

curl --version

If curl is available, the following message shown in screenshot will appear.



```
ubuntu@ip-10-0-2-178:~$ curl --version
curl 7.68.0 (x86_64-pc-linux-gnu) libcurl/7.68.0 OpenSSL/1.1.1f zlib/1.2.11 brotli/1.0.7 libidn2/2.1.40.0 librtmp/2.3
Release-Date: 2020-01-08
Protocols: dict file ftp ftps gopher http https imap imaps ldap ldaps pop3 pop3s rtmp rtsp
Features: AsynchDNS brotli GSS-API HTTP2 HTTPS-proxy IDN IPv6 Kerberos Largefile libz NTLM
ubuntu@ip-10-0-2-178:~$
```

To view curl version

After that, run the following commands for downloading the Prometheus.

mkdir -p /tmp/prometheus

cd /tmp/prometheus

wget https://raw.githubusercontent.com/mannem302/download/main/prom.sh

chmod 755 prom.sh

sh prom.sh

After executing the earlier commands, Prometheus will get downloaded in that folder.

STEP 5: EXTRACT PROMETHEUS

Once you have downloaded the latest version of Prometheus for your Ubuntu system, you can extract the software using the following command.

tar xvfz prometheus-*.*.linux-amd64.tar.gz

After executing the above command, the following messages shown in screen will appear.

```
ubuntu@ip-10-0-2-178:/tmp/prometheus$ tar xvfz prometheus-*.*.linux-amd64.tar.gz
prometheus-2.46.0.linux-amd64/
prometheus-2.46.0.linux-amd64/console_libraries/
prometheus-2.46.0.linux-amd64/console_libraries/prom.lib
prometheus-2.46.0.linux-amd64/console_libraries/menu.lib
prometheus-2.46.0.linux-amd64/NOTICE
prometheus-2.46.0.linux-amd64/promtool
prometheus-2.46.0.linux-amd64/prometheus.yml
prometheus-2.46.0.linux-amd64/LICENSE
prometheus-2.46.0.linux-amd64/prometheus
prometheus-2.46.0.linux-amd64/consoles/
prometheus-2.46.0.linux-amd64/consoles/node-disk.html
prometheus-2.46.0.linux-amd64/consoles/node-cpu.html
prometheus-2.46.0.linux-amd64/consoles/prometheus.html
prometheus-2.46.0.linux-amd64/consoles/prometheus-overview.html
prometheus-2.46.0.linux-amd64/consoles/node.html
prometheus-2.46.0.linux-amd64/consoles/index.html.example
prometheus-2.46.0.linux-amd64/consoles/node-overview.html
ubuntu@ip-10-0-2-178:/tmp/prometheus$
```

Extracting the files

Then, enter into the extracted folder by using the following command

cd prometheus*/

After entering into that folder, run the following commands for configuring the Prometheus.

Next, move the binary files to the **/usr/local/bin/** directory using the following command.

sudo mv prometheus promtool /usr/local/bin/

To verify the installation of Prometheus, you can use the following commands.

prometheus --version

promtool --version

After executing the above commands, the following message will get displayed.

```
ubuntu@ip-10-0-2-178:/tmp/prometheus/prometheus-2.46.0.linux-amd64$ prometheus --version
prometheus, version 2.46.0 (branch: HEAD, revision: cbb69e51423565ec40146e74141f2d0b3b71b4f0)
  build user:    root@42454fc0f41e
  build date:    20230725-12:31:24
  go version:    go1.20.6
  platform:     linux/amd64
  tags:         netgo,builtinassets,stringlabels
ubuntu@ip-10-0-2-178:/tmp/prometheus/prometheus-2.46.0.linux-amd64$ promtool --version
promtool, version 2.46.0 (branch: HEAD, revision: cbb69e51423565ec40146e74141f2d0b3b71b4f0)
  build user:    root@42454fc0f41e
  build date:    20230725-12:31:24
  go version:    go1.20.6
  platform:     linux/amd64
  tags:         netgo,builtinassets,stringlabels
ubuntu@ip-10-0-2-178:/tmp/prometheus/prometheus-2.46.0.linux-amd64$
```

To check versions

Then, move the Prometheus configuration template to the `/etc/prometheus/` directory using the following command

```
sudo mv prometheus.yml /etc/prometheus/prometheus.yml
```

Finally, move the `consoles` and `console_libraries` directories to the `/etc/prometheus/` directory using the following command.

```
sudo mv consoles/ console_libraries/ /etc/prometheus/  
cd $HOME
```

STEP 6: CREATE SYSTEMD SERVICE

By default, Prometheus does not come with a systemd service, which makes it difficult to manage and control the software.

To, create the systemd service file using the following command.

```
sudo vi /etc/systemd/system/prometheus.service
```

After creating `prometheus.service` file, you have to paste the following content in it and save it.

[Unit]

Description=Prometheus

Documentation=<https://prometheus.io/docs/introduction/overview/>

Wants=network-online.target

After=network-online.target

[Service]

Type=simple

User=prometheus

Group=prometheus

ExecReload=/bin/kill -HUP \$MAINPID

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 \

--web.listen-address=0.0.0.0:9090 \

--web.external-url=

SyslogIdentifier=prometheus

Restart=always

[Install]

WantedBy=multi-user.target

Next, change the directory permissions for the Prometheus user and group using the following commands, one by one.

```
for i in rules rules.d files_sd; do sudo chown -R prometheus:prometheus /etc/prometheus/${i}; done
```

```
for i in rules rules.d files_sd; do sudo chmod -R 775 /etc/prometheus/${i}; done
```

```
sudo chown -R prometheus:prometheus /var/lib/prometheus/
```

After changing the directory permissions, reload the systemd daemon using the following command.

```
sudo systemctl daemon-reload
```

Finally, to start the Prometheus systemd service use the following command.

```
sudo systemctl start prometheus
```

After starting the prometheus service, use the following command to view the status.

```
sudo systemctl status prometheus
```

After executing the above command, the following screen will appear.

```
ubuntu@ip-10-0-2-178:~$ sudo systemctl status prometheus
● prometheus.service - Prometheus
   Loaded: loaded (/etc/systemd/system/prometheus.service; disabled; vendor preset: enabled)
   Active: active (running) since Wed 2023-08-02 16:38:33 UTC; 14s ago
     Docs: https://prometheus.io/docs/introduction/overview/
  Main PID: 3189 (prometheus)
    Tasks: 5 (limit: 1126)
   Memory: 70.2M
   CGroup: /system.slice/prometheus.service
           └─3189 /usr/local/bin/prometheus --config.file=/etc/prometheus/prometheus.yml --storage.tsdb.path=/var/lib/prometheus --we

Aug 02 16:38:34 ip-10-0-2-178 prometheus[3189]: ts=2023-08-02T16:38:34.384Z caller=head.go:676 level=info component=tsdb msg="On-disk m
Aug 02 16:38:34 ip-10-0-2-178 prometheus[3189]: ts=2023-08-02T16:38:34.384Z caller=head.go:684 level=info component=tsdb msg="Replaying
Aug 02 16:38:34 ip-10-0-2-178 prometheus[3189]: ts=2023-08-02T16:38:34.384Z caller=head.go:755 level=info component=tsdb msg="WAL segme
Aug 02 16:38:34 ip-10-0-2-178 prometheus[3189]: ts=2023-08-02T16:38:34.385Z caller=head.go:792 level=info component=tsdb msg="WAL repla
Aug 02 16:38:34 ip-10-0-2-178 prometheus[3189]: ts=2023-08-02T16:38:34.386Z caller=main.go:1047 level=info fs type=EXT4 SUPER MAGIC
Aug 02 16:38:34 ip-10-0-2-178 prometheus[3189]: ts=2023-08-02T16:38:34.386Z caller=main.go:1050 level=info msg="TSDB started"
Aug 02 16:38:34 ip-10-0-2-178 prometheus[3189]: ts=2023-08-02T16:38:34.387Z caller=main.go:1231 level=info msg="Loading configuration f
Aug 02 16:38:34 ip-10-0-2-178 prometheus[3189]: ts=2023-08-02T16:38:34.392Z caller=main.go:1268 level=info msg="Completed loading of co
Aug 02 16:38:34 ip-10-0-2-178 prometheus[3189]: ts=2023-08-02T16:38:34.393Z caller=main.go:1011 level=info msg="Server is ready to rece
Aug 02 16:38:34 ip-10-0-2-178 prometheus[3189]: ts=2023-08-02T16:38:34.394Z caller=manager.go:1009 level=info component="rule manager"
lines 1-20/20 (END)
```

To start the Prometheus service automatically, when system startup, use the following command.

```
sudo systemctl enable prometheus
```

The following message will appear.

```
ubuntu@ip-10-0-2-178:~$ sudo systemctl enable prometheus
Created symlink /etc/systemd/system/multi-user.target.wants/prometheus.service → /etc/systemd/system/prometheus.service.
ubuntu@ip-10-0-2-178:~$
```

STEP 7: CONFIGURE SECURITY GROUPS/ UFW FIREWALL

Option-1: If you are using UBUNTU in AWS cloud, you have to edit inbound rules for the ports 9090 and 9093 in the security groups.

Note:- Don't do the following option2 in AWS cloud.

Option-2: If you are using UBUNTU in local machine, you have to allow the ports and enable the UFW status. By using the following commands.

sudo ufw status (It displays the UFW status)

sudo ufw allow 9090/tcp

sudo ufw allow 9093/tcp

After executing the above commands, you have to enable the UFW by using the following commands.

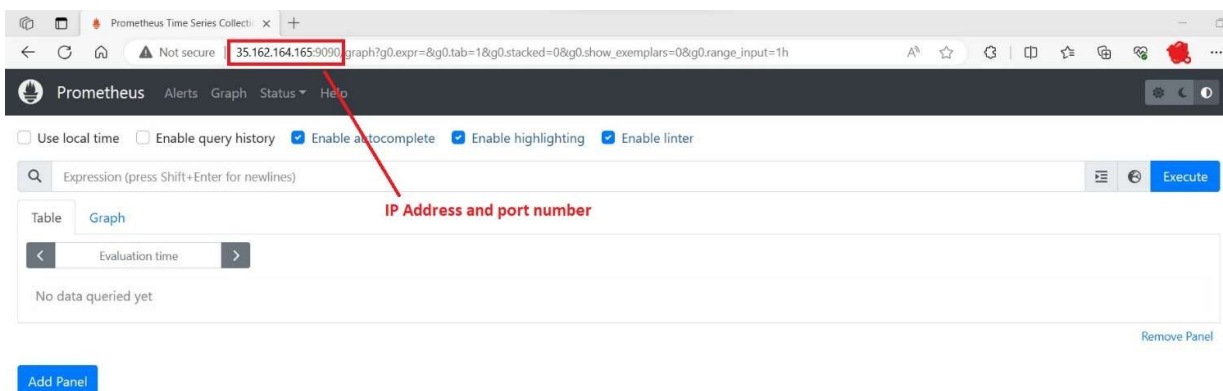
sudo ufw enable (Only Installing in local VM's we have to do this)

STEP 8: ACCESS PROMETHEUS WEB UI

After completion of above steps, to view prometheus web page use the following URL in browser.

http://YOUR_IP:9090/

The following page will appear.



If you are seeing this page, prometheus setup was completed successfully.

Node Exporter Installation

Now we have to install node_exporter for monitoring linux system and visualize the data, Grafana need to be installed and integrated with node_exporter as follows.

To download node_exporter, first create a directory using following command

```
mkdir node_exporter
```

```
cd node_exporter
```

Then, we have to execute the following commands for downloading the node_exporter.

```
wget https://raw.githubusercontent.com/mannem302/download/main/node.sh
```

```
sudo chmod 755 node.sh
```

```
sh node.sh
```

After completion of downloading the file, extract that folder by using the following commands.

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

After extracting the file, rename the folder by using the following commands

```
mv node_exporter-*.linux-amd64 node_exporter
```

By, default node_exporter was not starting automatically. So, to set it we have to create a systemd service file, by using the following commands.

```
sudo vi /etc/systemd/system/node_exporter.service
```

paste the following content in that service file

```
[Unit]
```

```
Description=Prometheus Node Exporter
```

```
Documentation=https://github.com/prometheus/node_exporter
```

```
After=network-online.target
```

```
[Service]
```

```
User=root
```

```
EnvironmentFile=/etc/default/node_exporter
```

```
ExecStart=/opt/node_exporter/node_exporter $OPTIONS
```

```
Restart=on-failure
```

```
RestartSec=5
```

```
[Install]
```

```
WantedBy=multi-user.target
```


After saving the file, you have to run the following command.

sudo systemctl daemon-reload

After that, create the options file by using the following commands

sudo vi /etc/default/node_exporter

We have to paste the following content into that file.

OPTIONS=""

To start the service at system startup. We have to execute the following commands.

systemctl enable node_exporter

After executing the above command, the following message will appear.

```
ubuntu@ip-10-0-2-178:~$ sudo systemctl enable node_exporter
Created symlink /etc/systemd/system/multi-user.target.wants/node_exporter.service → /etc/systemd/system/node_exporter.service.
```

To start the node_exporter service, execute the following commands.

systemctl start node_exporter

To view the status, execute the following command.

systemctl status node_exporter

After executing the above command, the following message will appear.

```
ubuntu@ip-10-0-2-178:~$ sudo systemctl start node_exporter
ubuntu@ip-10-0-2-178:~$ sudo systemctl status node_exporter
● node_exporter.service - Prometheus Node Exporter
   Loaded: loaded (/etc/systemd/system/node_exporter.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2023-08-03 04:46:12 UTC; 11s ago
     Docs: https://github.com/prometheus/node_exporter
    Main PID: 1534 (node_exporter)
      Tasks: 3 (limit: 1126)
     Memory: 2.0M
    CGroup: /system.slice/node_exporter.service
            └─1534 /opt/node_exporter/node_exporter

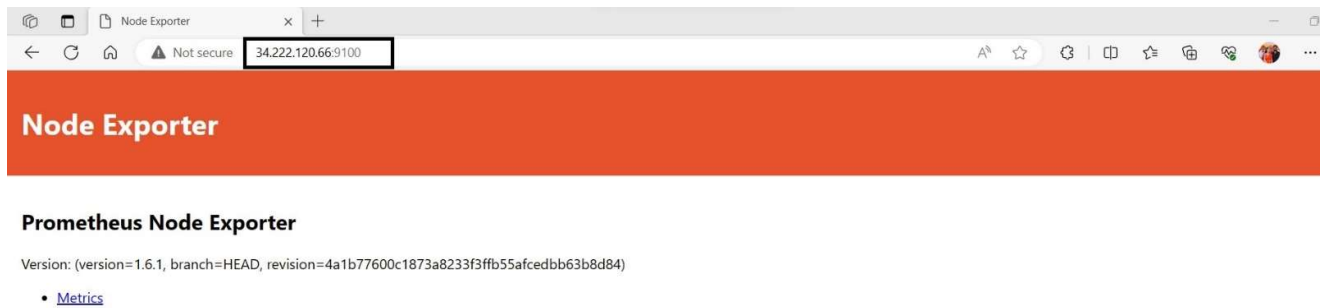
The service is in active state

Aug 03 04:46:12 ip-10-0-2-178 node_exporter[1534]: ts=2023-08-03T04:46:12.861Z caller=node_exporter.go:117 level=info collector=the
Aug 03 04:46:12 ip-10-0-2-178 node_exporter[1534]: ts=2023-08-03T04:46:12.861Z caller=node_exporter.go:117 level=info collector=tim
Aug 03 04:46:12 ip-10-0-2-178 node_exporter[1534]: ts=2023-08-03T04:46:12.861Z caller=node_exporter.go:117 level=info collector=tim
Aug 03 04:46:12 ip-10-0-2-178 node_exporter[1534]: ts=2023-08-03T04:46:12.861Z caller=node_exporter.go:117 level=info collector=udg
Aug 03 04:46:12 ip-10-0-2-178 node_exporter[1534]: ts=2023-08-03T04:46:12.861Z caller=node_exporter.go:117 level=info collector=una
Aug 03 04:46:12 ip-10-0-2-178 node_exporter[1534]: ts=2023-08-03T04:46:12.861Z caller=node_exporter.go:117 level=info collector=vms
Aug 03 04:46:12 ip-10-0-2-178 node_exporter[1534]: ts=2023-08-03T04:46:12.861Z caller=node_exporter.go:117 level=info collector=xfs
Aug 03 04:46:12 ip-10-0-2-178 node_exporter[1534]: ts=2023-08-03T04:46:12.862Z caller=node_exporter.go:117 level=info collector=zfs
Aug 03 04:46:12 ip-10-0-2-178 node_exporter[1534]: ts=2023-08-03T04:46:12.862Z caller=tls_config.go:274 level=info msg="Listening c
Aug 03 04:46:12 ip-10-0-2-178 node_exporter[1534]: ts=2023-08-03T04:46:12.862Z caller=tls_config.go:277 level=info msg="TLS is disa
lines 1-20/20 (END)
```

To check the node_exporter web UI, execute the following URL in your browser,
node_exporter port number is 9100.

http://Your_IP:9100/

After entering that URL in browser, the following page will appear.



As this page appears, you have successfully installed node_exporter.

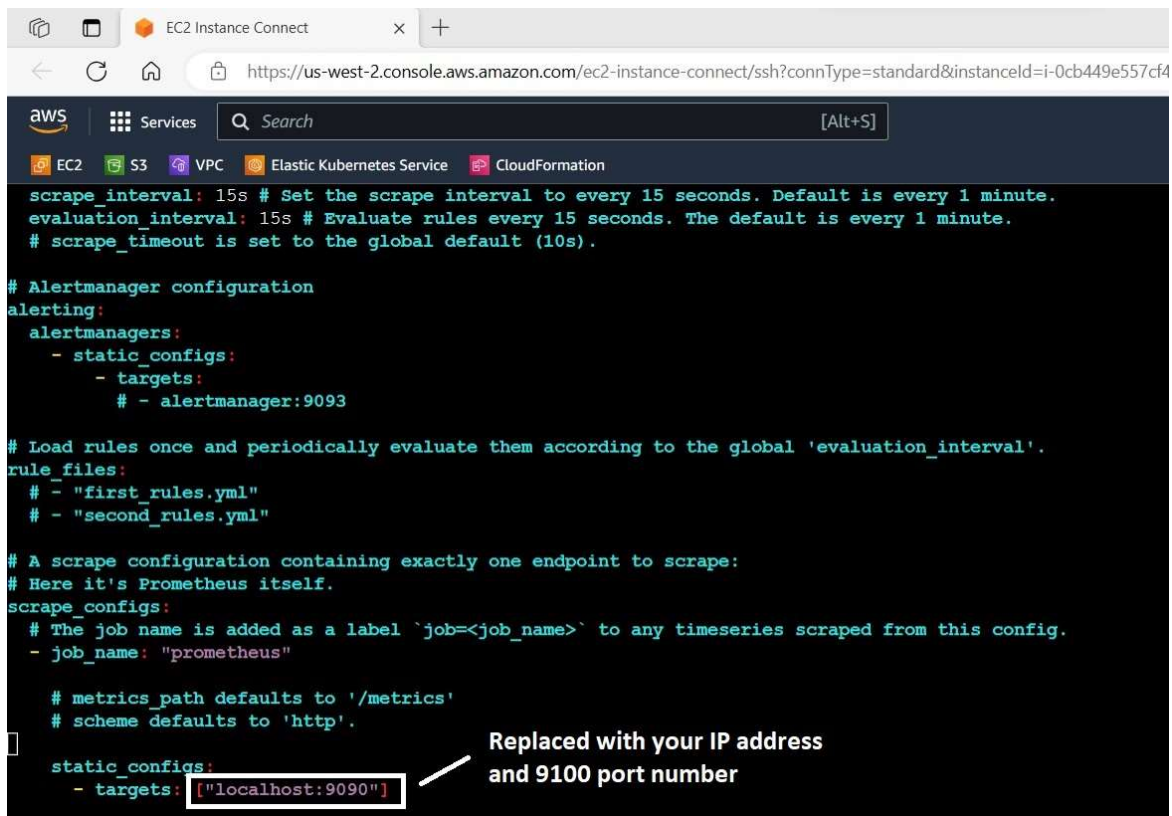
Integration of prometheus and node_exporter

For collecting linux system metrics, we need node_exporter and it have to be integrated with prometheus server. For collecting metrics.

Follow the below steps, for integrating both

vi /etc/prometheus/prometheus.yml

After opening that file, you will see the following screen.



At the bottom, you will find the following lines, there you have to edit those. In place of localhost IP address shall be given and in place of 9090 port, you have to give 9100 static_configs:

- targets: ["localhost:9090"]

After replacing the IP and port number, save the file and restart the prometheus.service by using the following commands.

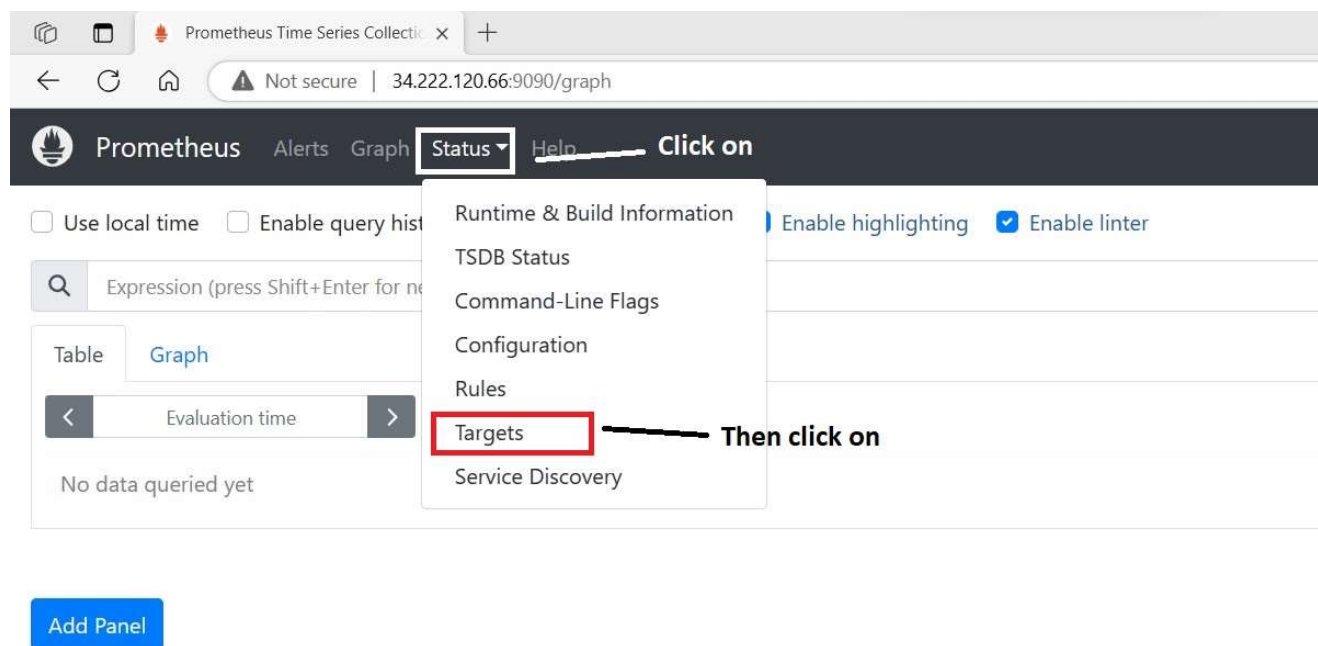
- targets: ["34.222.120.66:9100"]

sudo systemctl restart prometheus.service

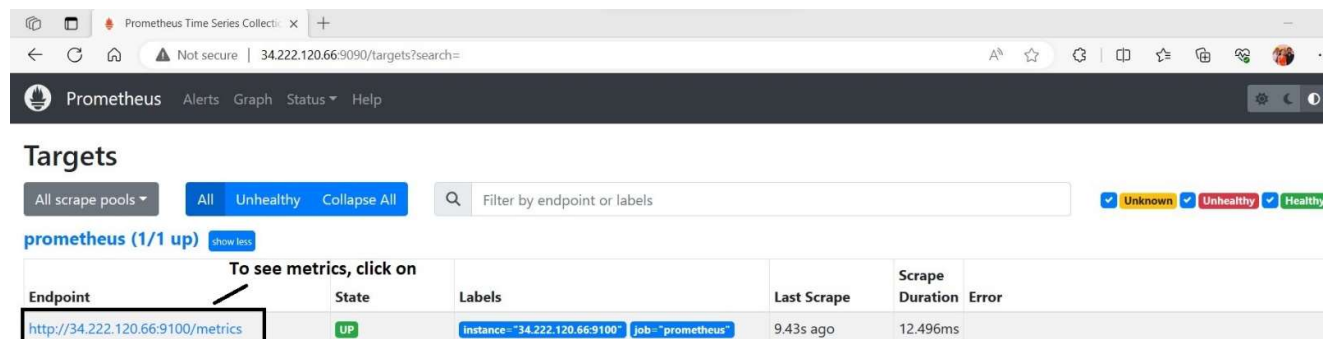
Now, prometheus and node_exporter had integrated. We can view that in web UI, as shown below.

http://Your_IP:9090

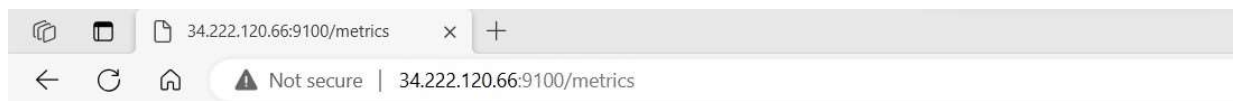
The following page will appear



After clicking on targets option, the following page will appear.



After clicking on that link, to see metrics the following page will appear.



The screenshot shows a web browser window with the address bar displaying '34.222.120.66:9100/metrics'. The page content is a list of Go metrics and their values, including garbage collection statistics, goroutine counts, and memory usage details.

```
# HELP go_gc_duration_seconds A summary of the pause duration of garbage collection cycles.
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 2.0271e-05
go_gc_duration_seconds{quantile="0.25"} 3.6181e-05
go_gc_duration_seconds{quantile="0.5"} 3.8277e-05
go_gc_duration_seconds{quantile="0.75"} 4.2722e-05
go_gc_duration_seconds{quantile="1"} 9.602e-05
go_gc_duration_seconds_sum 0.008686372
go_gc_duration_seconds_count 200
# HELP go_goroutines Number of goroutines that currently exist.
# TYPE go_goroutines gauge
go_goroutines 8
# HELP go_info Information about the Go environment.
# TYPE go_info gauge
go_info{version="go1.20.6"} 1
# HELP go_memstats_alloc_bytes Number of bytes allocated and still in use.
# TYPE go_memstats_alloc_bytes gauge
go_memstats_alloc_bytes 3.056448e+06
# HELP go_memstats_alloc_bytes_total Total number of bytes allocated, even if freed.
# TYPE go_memstats_alloc_bytes_total counter
go_memstats_alloc_bytes_total 1.6940212e+08
# HELP go_memstats_buck_hash_sys_bytes Number of bytes used by the profiling bucket hash table.
# TYPE go_memstats_buck_hash_sys_bytes gauge
go_memstats_buck_hash_sys_bytes 1.489784e+06
# HELP go_memstats_frees_total Total number of frees.
# TYPE go_memstats_frees_total counter
go_memstats_frees_total 2.05451e+06
# HELP go_memstats_gc_sys_bytes Number of bytes used for garbage collection system metadata.
# TYPE go_memstats_gc_sys_bytes gauge
go_memstats_gc_sys_bytes 8.431056e+06
# HELP go_memstats_heap_alloc_bytes Number of heap bytes allocated and still in use.
# TYPE go_memstats_heap_alloc_bytes gauge
go_memstats_heap_alloc_bytes 3.056448e+06
# HELP go_memstats_heap_idle_bytes Number of heap bytes waiting to be used.
# TYPE go_memstats_heap_idle_bytes gauge
go_memstats_heap_idle_bytes 3.710976e+06
# HELP go_memstats_heap_inuse_bytes Number of heap bytes that are in use.
# TYPE go_memstats_heap_inuse_bytes gauge
go_memstats_heap_inuse_bytes 4.284416e+06
# HELP go_memstats_heap_objects Number of allocated objects.
# TYPE go_memstats_heap_objects gauge
go_memstats_heap_objects 24179
# HELP go_memstats_heap_released_bytes Number of heap bytes released to OS.
# TYPE go_memstats_heap_released_bytes gauge
go_memstats_heap_released_bytes 3.145728e+06
# HELP go_memstats_heap_sys_bytes Number of heap bytes obtained from system.
# TYPE go_memstats_heap_sys_bytes gauge
go_memstats_heap_sys_bytes 7.995392e+06
```

You can see the logs and metrics, but we cannot understand those metrics. To visualize the data received from node_exporter. We have to install Grafana and it shall be integrated.

Grafana Installation

To install the grafana, you have to execute the following commands in UBUNTU.

```
sudo apt-get install -y libfontconfig1
```

```
wget https://dl.grafana.com/enterprise/release/grafana-enterprise_10.0.3_amd64.deb
```

```
sudo dpkg -i grafana-enterprise_10.0.3_amd64.deb
```

After executing the above commands, the following message screen will appear.

```
Last login: Thu Aug 3 09:39:15 2023 from 18.237.140.164
ubuntu@ip-10-0-2-178:~$ sudo dpkg -i grafana-enterprise_10.0.3_amd64.deb
Selecting previously unselected package grafana-enterprise.
(Reading database ... 90825 files and directories currently installed.)
Preparing to unpack grafana-enterprise_10.0.3_amd64.deb ...
Unpacking grafana-enterprise (10.0.3) ...
Setting up grafana-enterprise (10.0.3) ...
Adding system user `grafana' (UID 114) ...
Adding new user `grafana' (UID 114) with group `grafana' ...
Not creating home directory `/usr/share/grafana'.
### NOT starting on installation, please execute the following statements to configure grafana to start automatically using systemd
sudo /bin/systemctl daemon-reload
sudo /bin/systemctl enable grafana-server — To enable the service automatically, when system startup
### You can start grafana-server by executing
sudo /bin/systemctl start grafana-server — To start the grafana service
Processing triggers for systemd (245.4-4ubuntu3.22) ...
ubuntu@ip-10-0-2-178:~$
```

To enable the grafana server start automatically, when system start-up, we have to execute the following commands.

sudo systemctl enable grafana-server

To start the grafana server, execute the following command.

sudo systemctl start grafana-server

To check the grafana server status, execute the following command.

sudo systemctl status grafana-server

After executing the above command, the following message will appear.

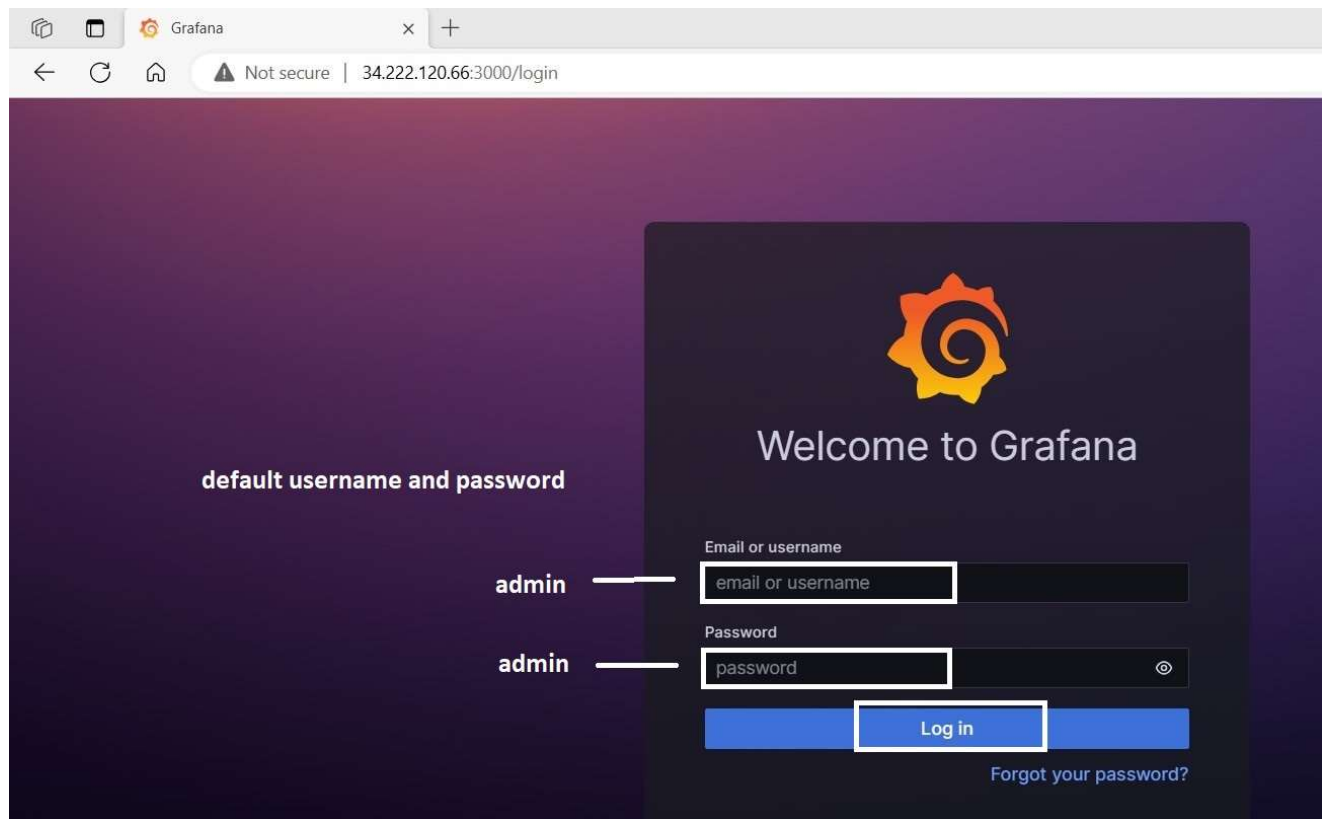
```
ubuntu@ip-10-0-2-178:~$ sudo systemctl status grafana-server.service
● grafana-server.service - Grafana instance
   Loaded: loaded (/lib/systemd/system/grafana-server.service; disabled; vendor preset: enabled)
   Active: active (running) since Thu 2023-08-03 10:48:10 UTC; 21s ago
     Docs: http://docs.grafana.org
   Main PID: 6107 (grafana)
    Tasks: 7 (limit: 1126)
   Memory: 148.1M
   CGroup: /system.slice/grafana-server.service
           └─6107 /usr/share/grafana/bin/grafana server --config=/etc/grafana/grafana.ini --pidfile=/run/grafana/grafana-serv

Aug 03 10:48:17 ip-10-0-2-178 grafana[6107]: logger=http.server t=2023-08-03T10:48:17.610211343Z level=info msg="HTTP Server Li
Aug 03 10:48:17 ip-10-0-2-178 grafana[6107]: logger=ngalert.state.manager t=2023-08-03T10:48:17.612206827Z level=info msg="Warm
Aug 03 10:48:17 ip-10-0-2-178 grafana[6107]: logger=ngalert.state.manager t=2023-08-03T10:48:17.613164735Z level=info msg="Stat
Aug 03 10:48:17 ip-10-0-2-178 grafana[6107]: logger=ticker t=2023-08-03T10:48:17.614636035Z level=info msg="starting first_tick=
Aug 03 10:48:17 ip-10-0-2-178 grafana[6107]: logger=grafanaStorageLogger t=2023-08-03T10:48:17.638578982Z level=info msg="stora
Aug 03 10:48:17 ip-10-0-2-178 grafana[6107]: logger=caching.service t=2023-08-03T10:48:17.656298196Z level=warn msg="Caching se
Aug 03 10:48:17 ip-10-0-2-178 grafana[6107]: logger=report t=2023-08-03T10:48:17.656584991Z level=warn msg="Scheduling and send
Aug 03 10:48:17 ip-10-0-2-178 grafana[6107]: logger=ngalert.multiorg.alertmanager t=2023-08-03T10:48:17.669248361Z level=info m
Aug 03 10:48:17 ip-10-0-2-178 grafana[6107]: logger=grafana.update.checker t=2023-08-03T10:48:17.80858557Z level=info msg="Upda
Aug 03 10:48:17 ip-10-0-2-178 grafana[6107]: logger=plugins.update.checker t=2023-08-03T10:48:17.850800491Z level=info msg="Upd
lines 1-20/20 (END)
```

As, it shows the server in active state. We have to open the following URL in browser, to view web UI. Grafana server port number is 3000.

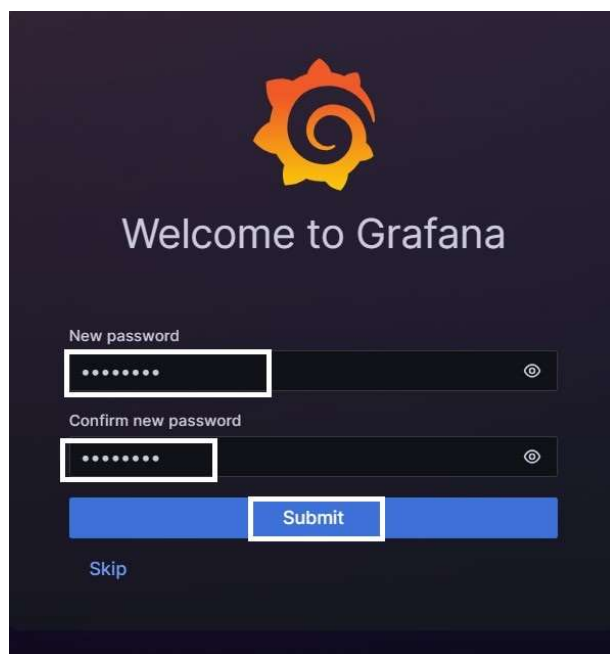
http://your_ip:3000/

The following page will display.



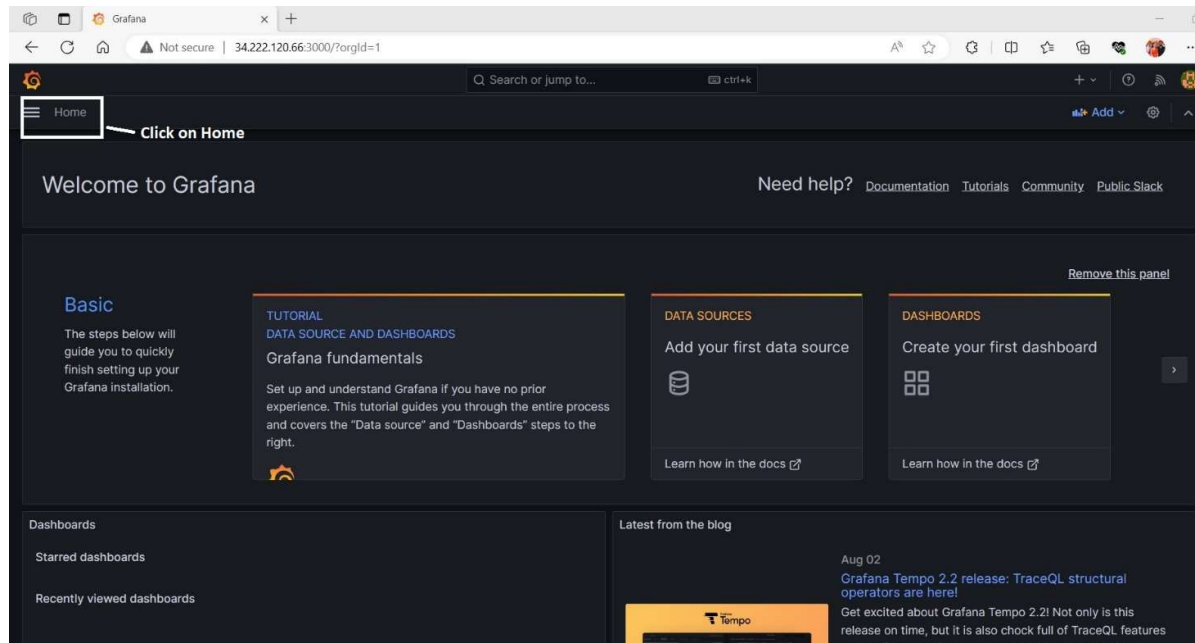
You have to enter username- **admin** and password- **admin**, after entering those click on 'log in' button.

After clicking on that log in button. The following page will appear.

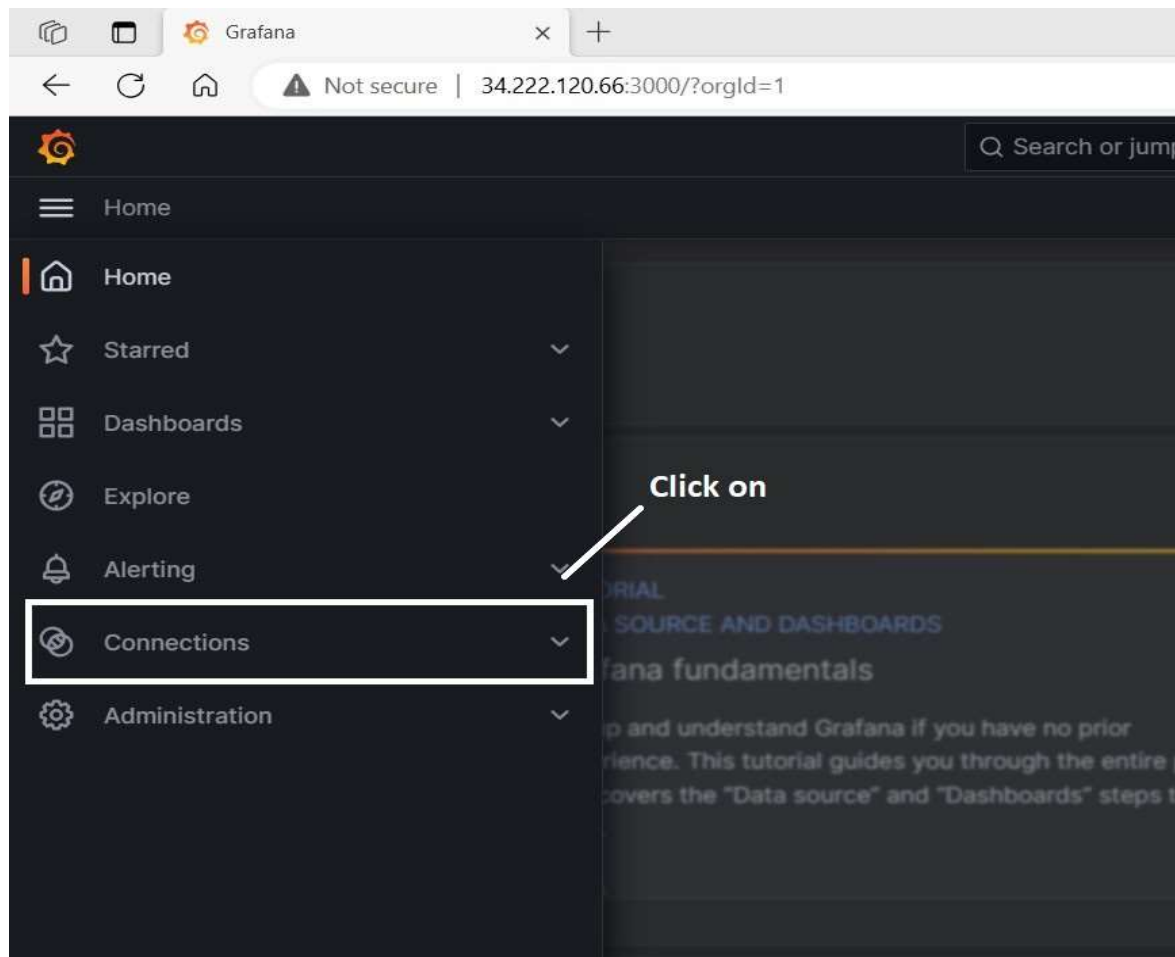


We have to set new password for grafana server and click on submit button.

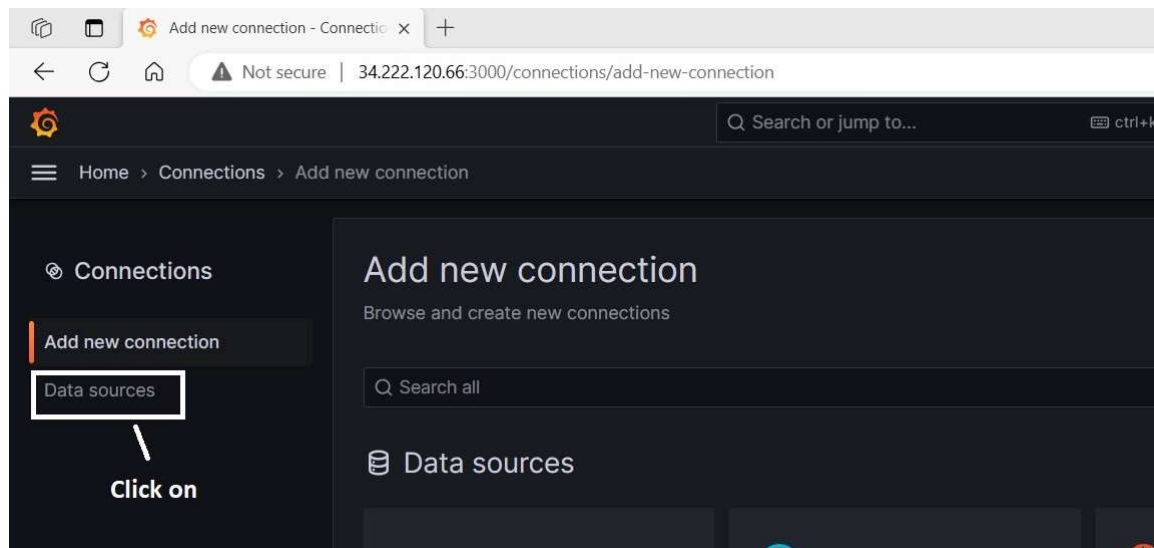
After clicking on submit button, the following page will appear.



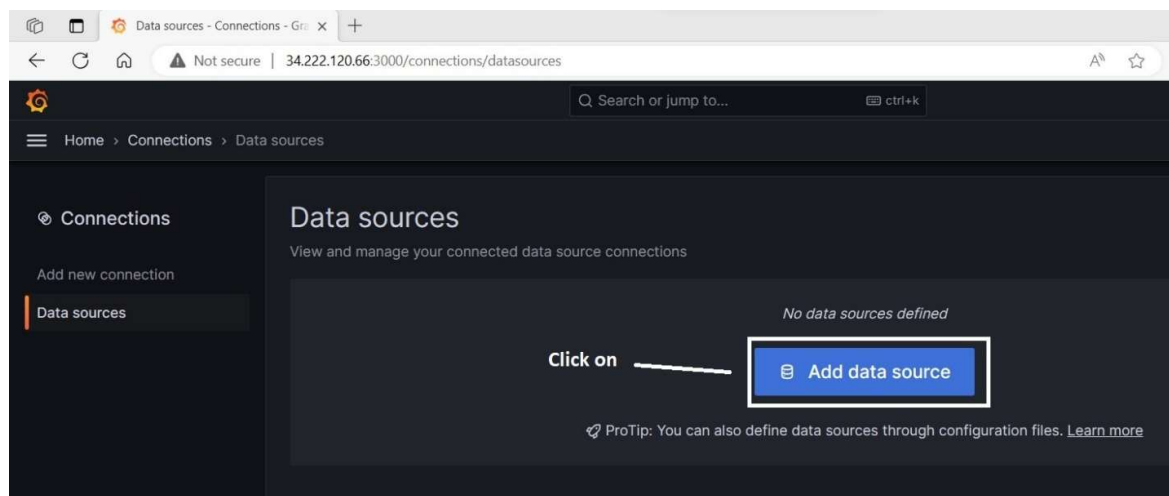
After clicking on home button, the following options screen will display.



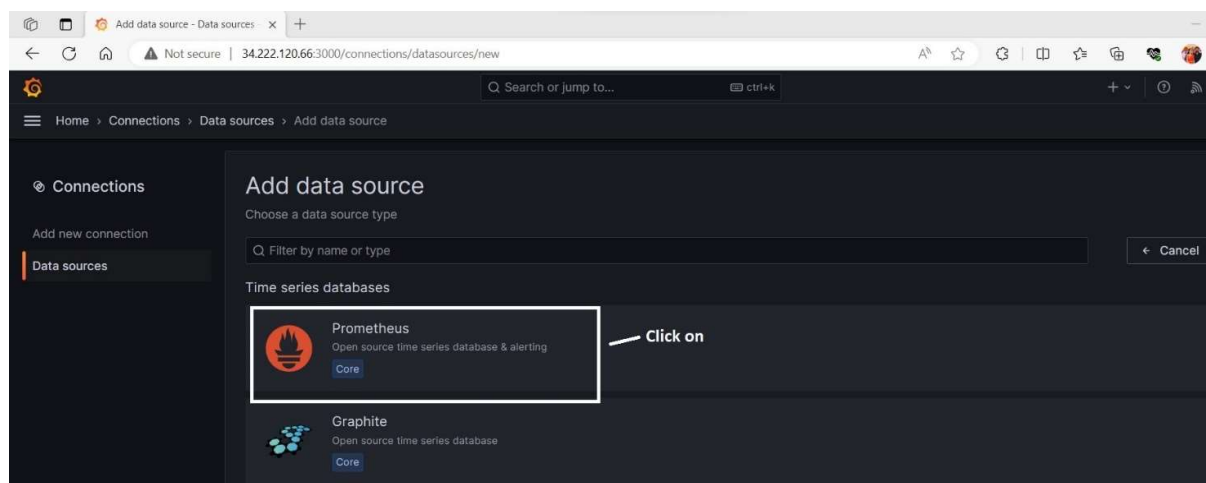
After clicking on connections button, the following page will display.



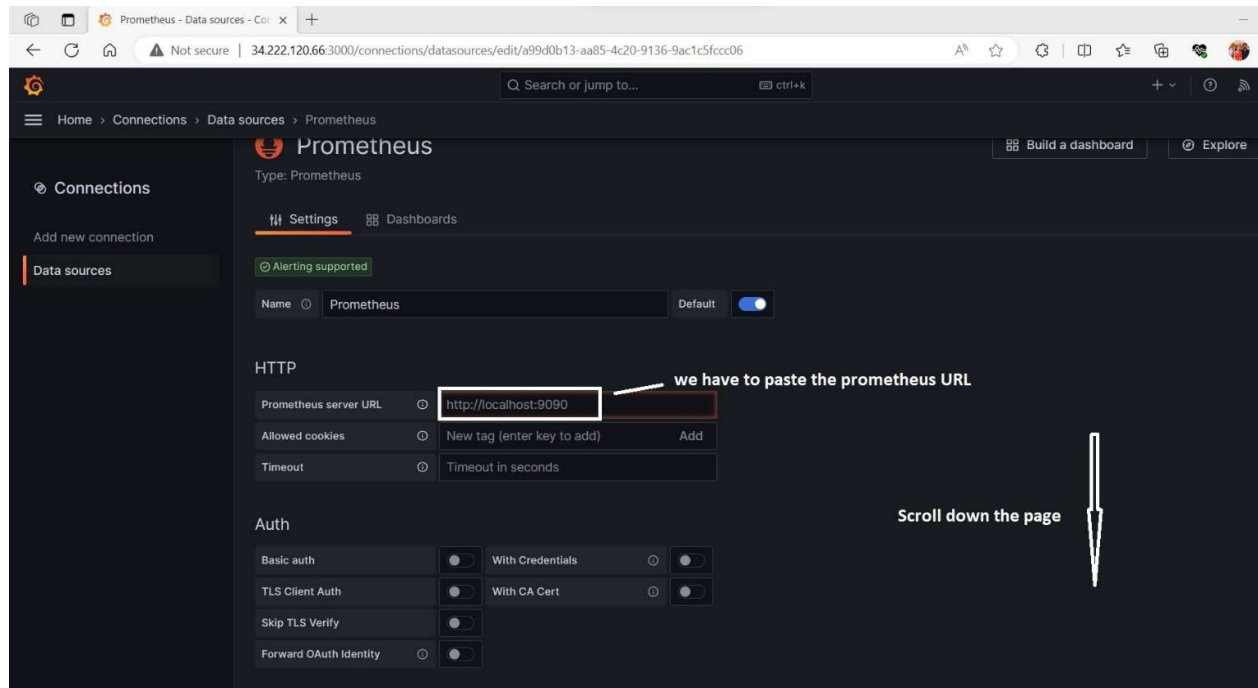
After clicking on 'data sources' button, the following page will be appeared.



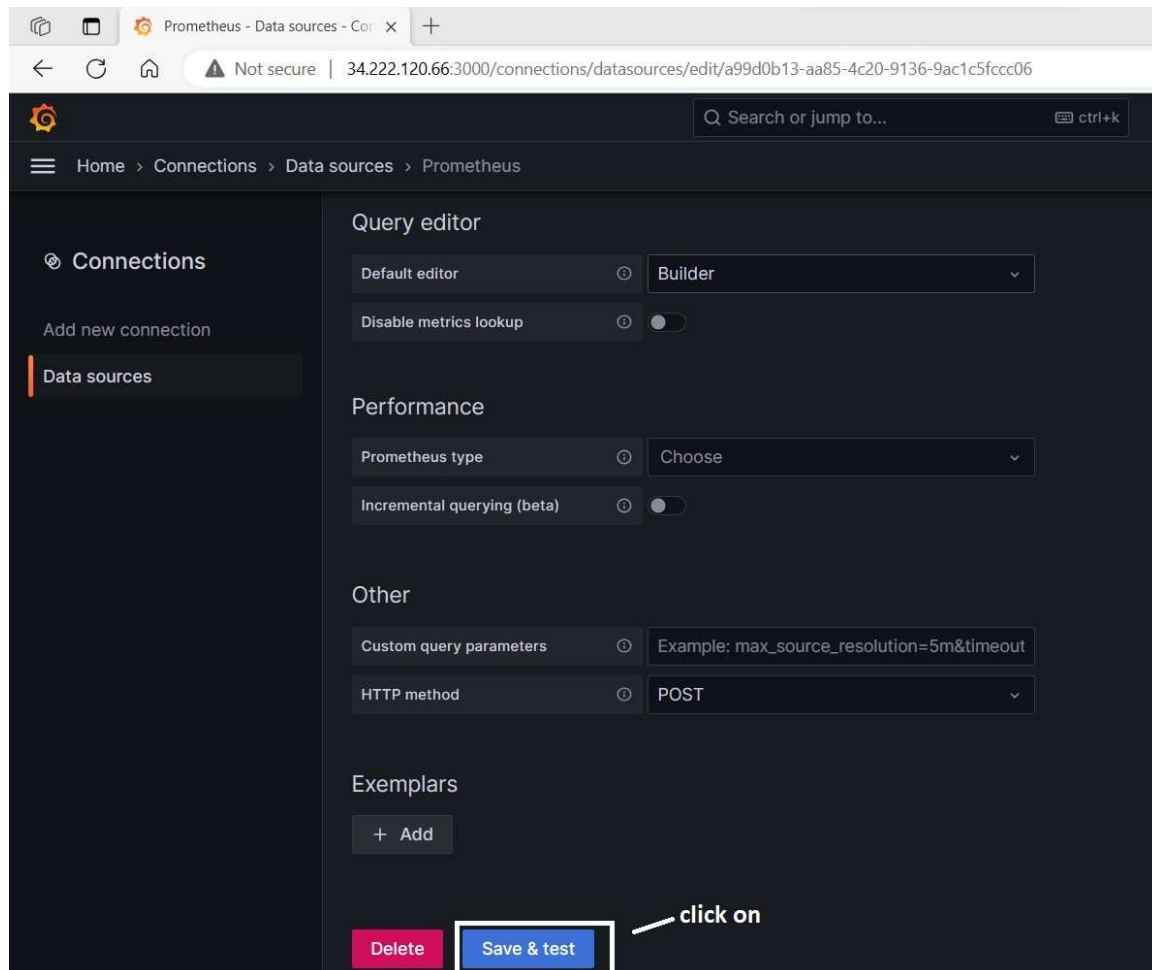
After clicking on 'Add data source' button, the following page will be appeared.



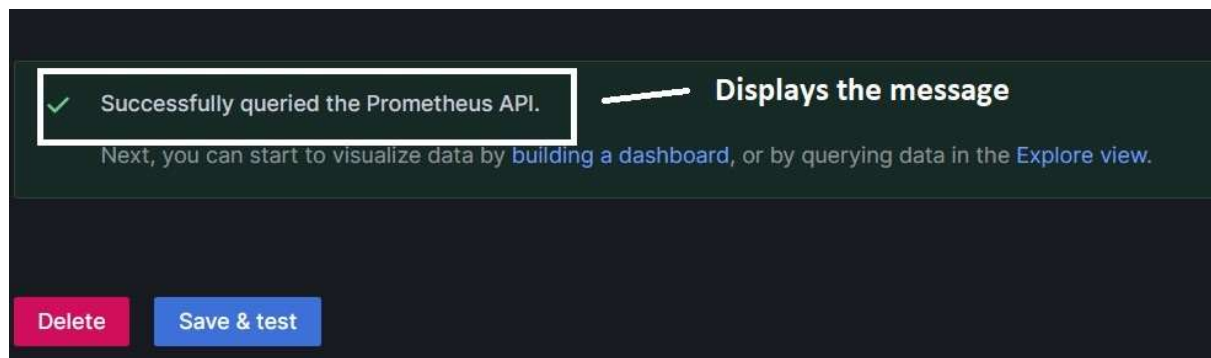
After clicking on 'prometheus' button, the following page will display.



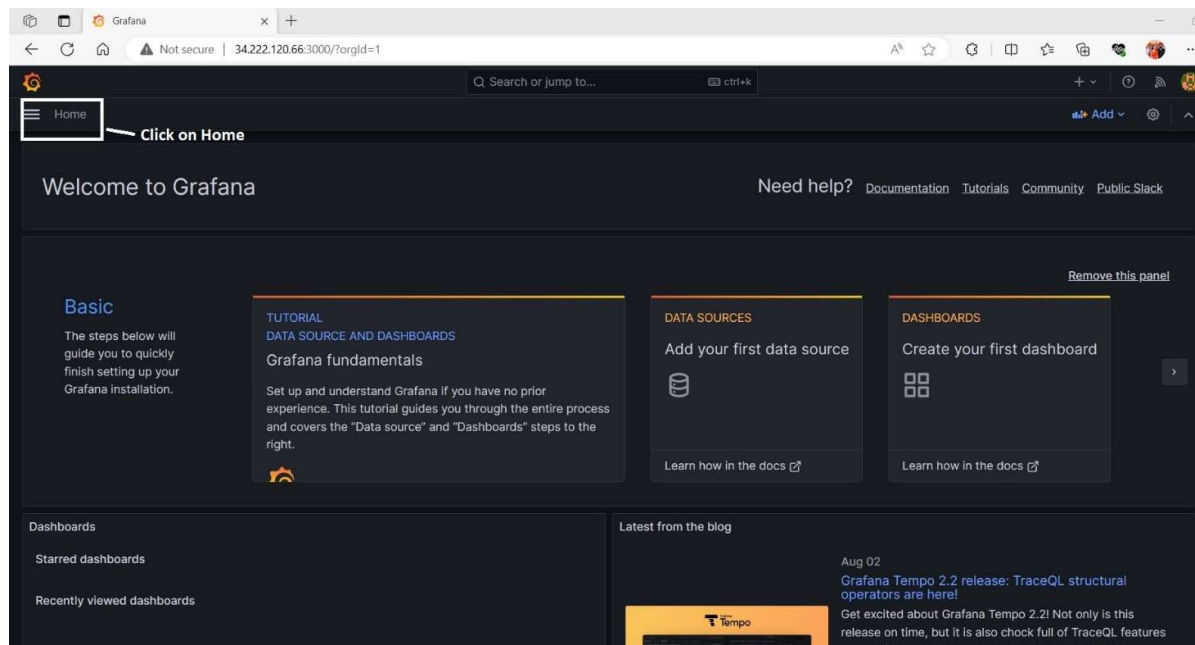
After scrolling down the above page, the following page will appear.



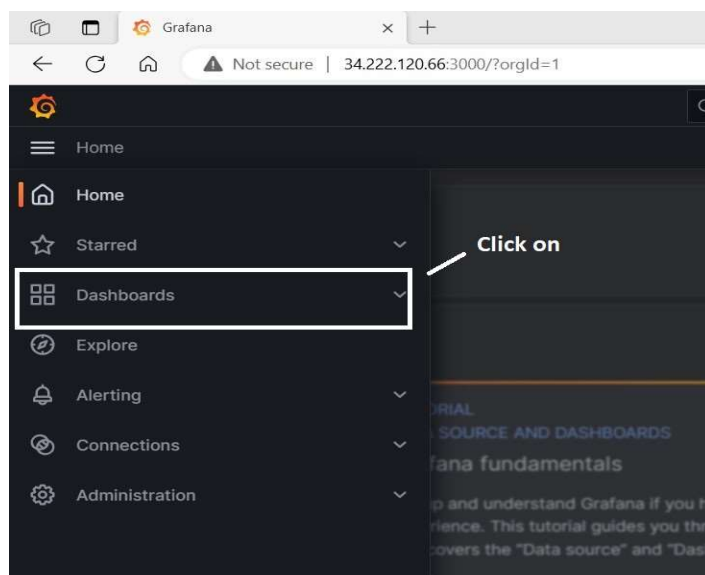
After clicking on 'save&test' button the following message will appear.



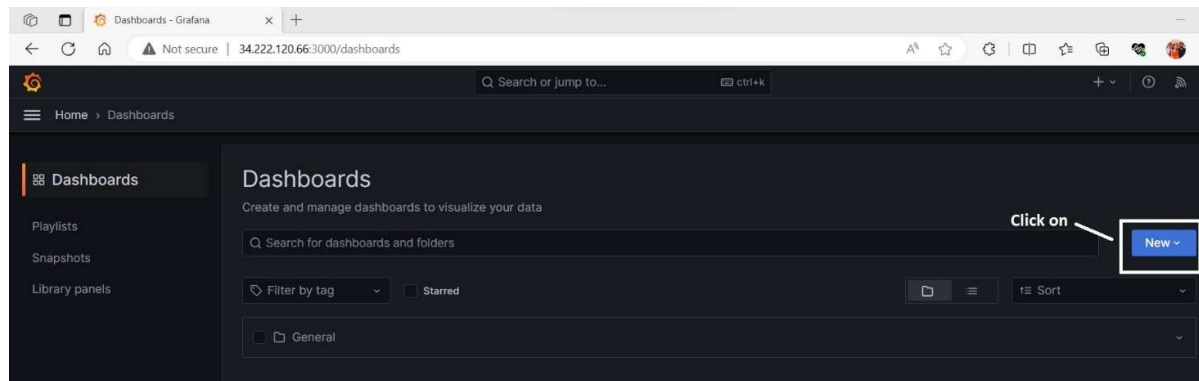
After that again click on home button, the following page will appear.



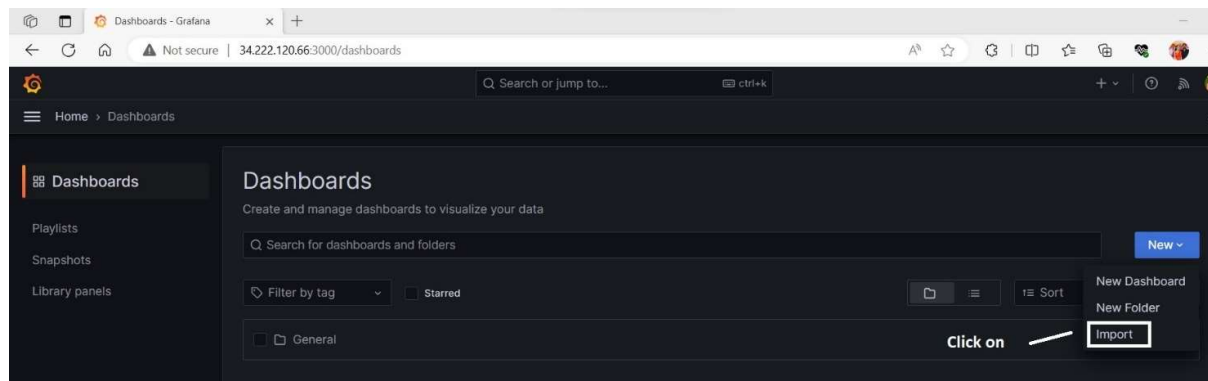
After that, clicking on home button. The following page will display.



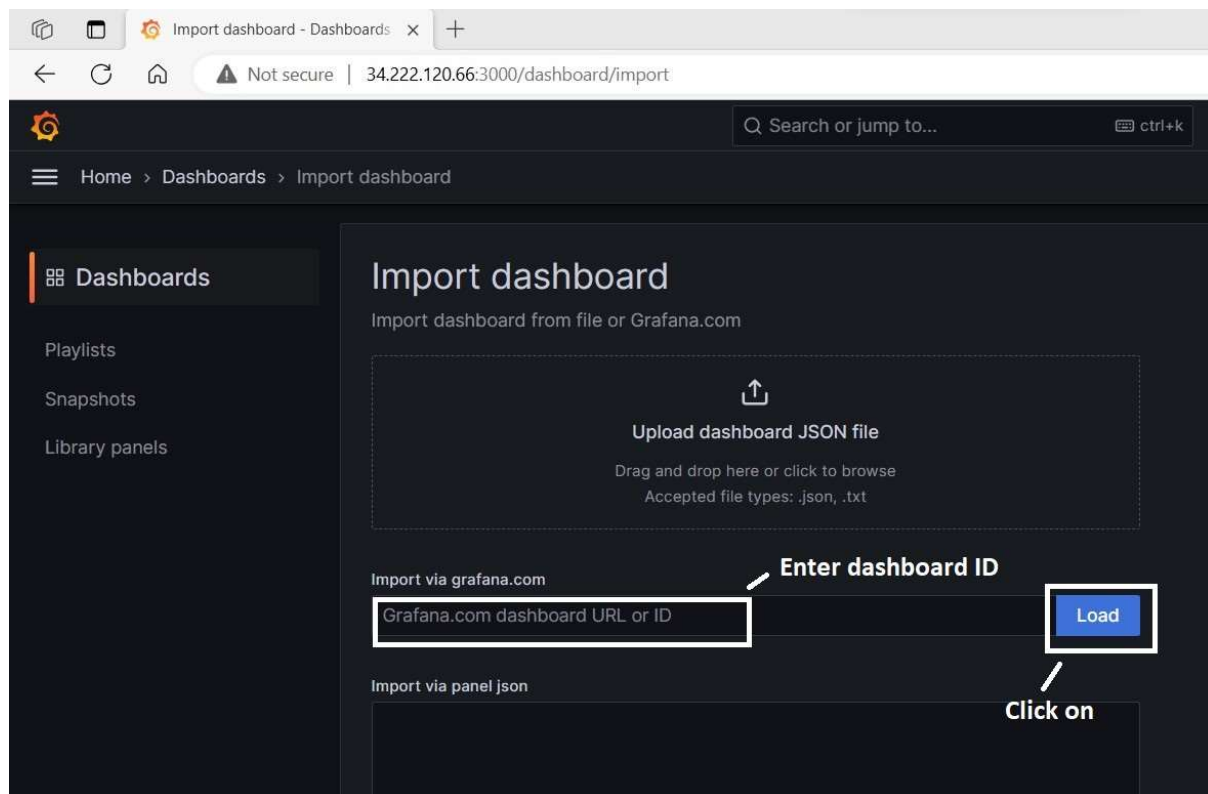
After clicking on '**dashboards**' button, the following page will appear.



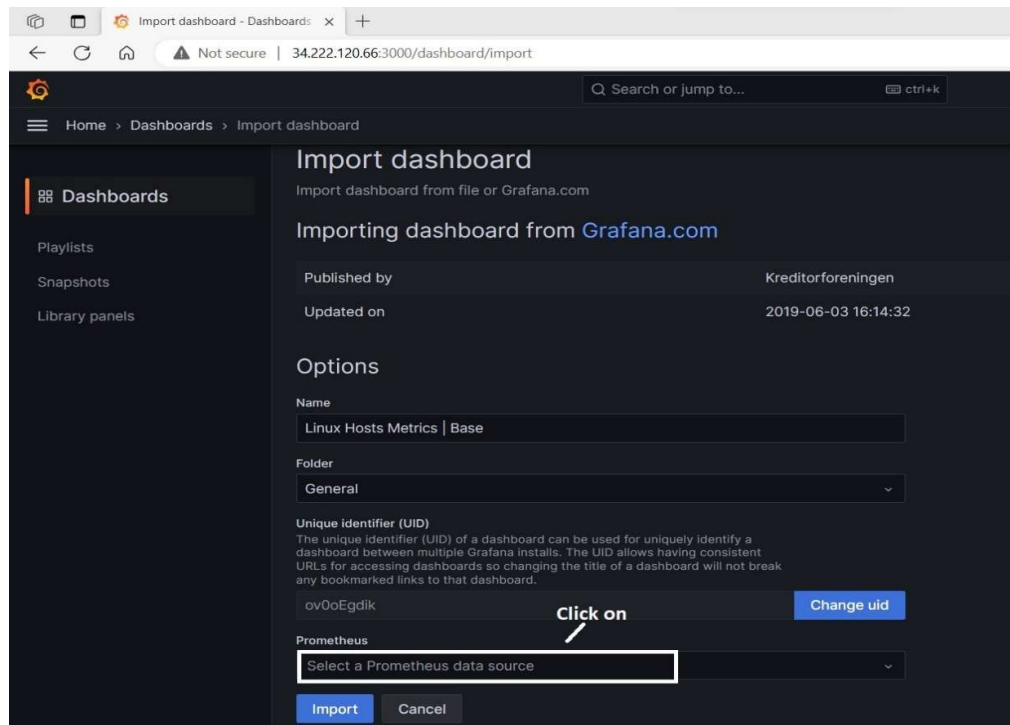
After clicking on, new dropdown menu. The following page will display.



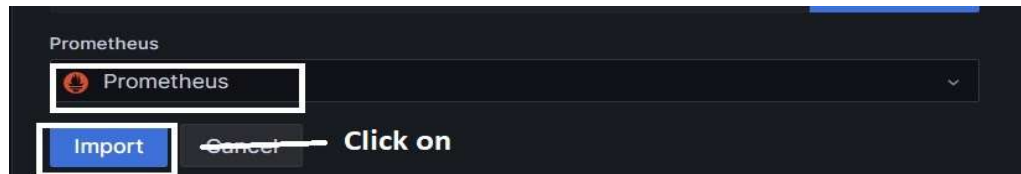
After clicking on '**import**' button, the following page will appear. (**Dashboard id-10180**)



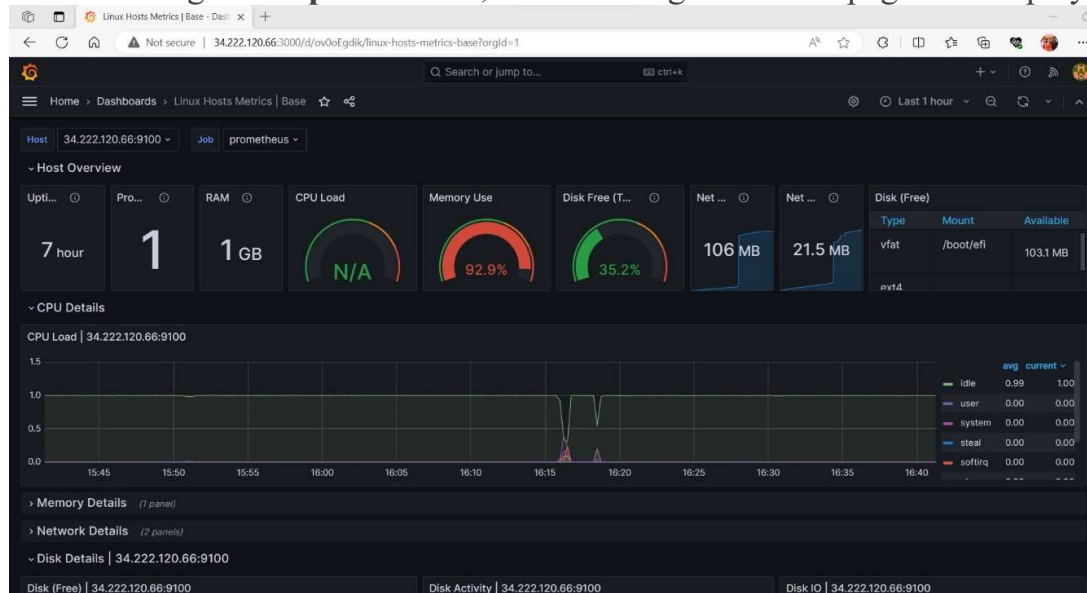
After clicking on 'Load' button, the following page will get display.



After clicking on that, the following options will get display.



After clicking on 'import' button, the following dashboard page will display.



Finally, standalone installation and configuration of prometheus, node_exporter and grafana was successfully completed.