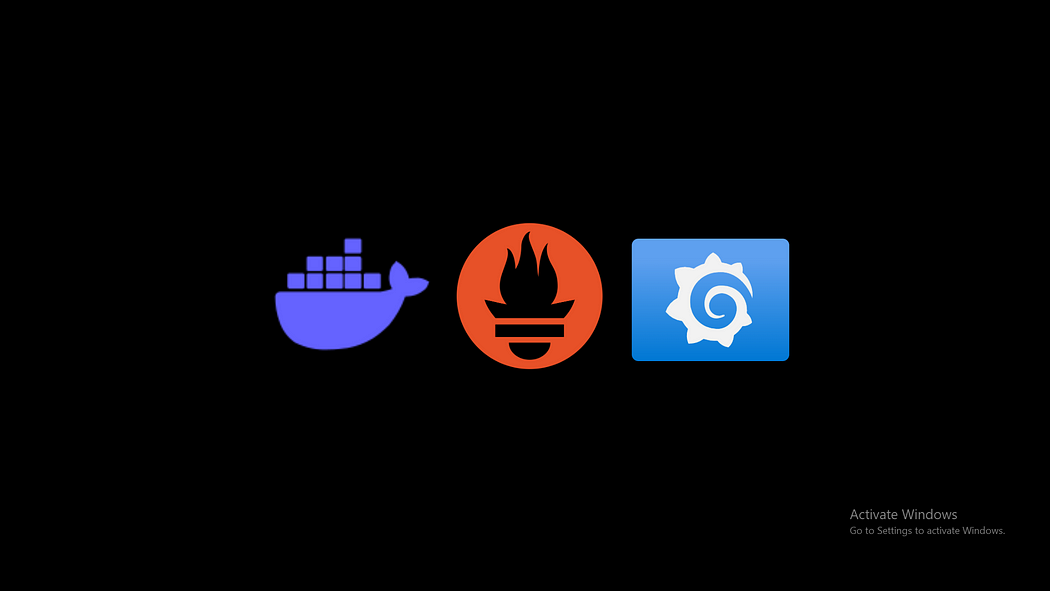
**Monitor Docker Container with Grafana & Prometheus**



1. Launch ubuntu 20/22 with below ports in AWS

9393,3000,9090

*2.* Login your machine and install docker

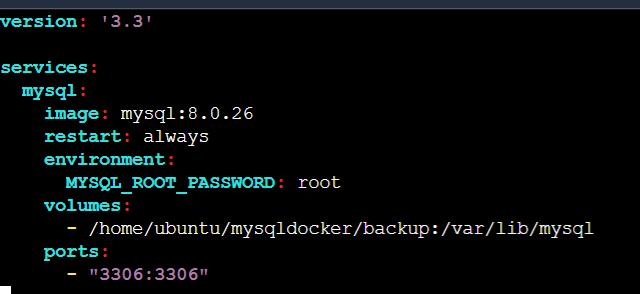
apt-get update  
apt-get install docker.io -y  
apt-get install docker-compose -y

3. To Monitor docker we need to deploy some container so then create one folder name as mysqldocker and go to that folder and inside that create backup folder to pvc

mkdir mysqldocker  
cd mysqldocker  
mkdir backup

*4.* then create a 1 docker-compose.yml file to run one docker container

version: '3.3'  
  
services:  
 mysql:  
 image: mysql:8.0.26  
 restart: always  
 environment:  
 MYSQL\_ROOT\_PASSWORD: root  
 volumes:  
 - /home/ubuntu/mysqldocker/backup:/var/lib/mysql  
 ports:  
 - "3306:3306"



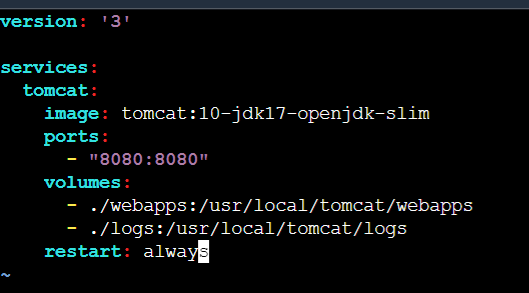
5. and then up that docker-compose file and check your mysql container are running

docker-compose up -d  
docker ps

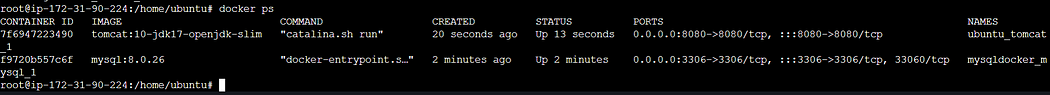


6. again go /home/ubuntu and these time again create tomcat file

version: '3'  
  
services:  
 tomcat:  
 image: tomcat:10-jdk17-openjdk-slim  
 ports:  
 - "8080:8080"  
 volumes:  
 - ./webapps:/usr/local/tomcat/webapps  
 - ./logs:/usr/local/tomcat/logs  
 restart: always



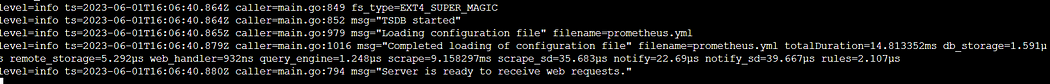
7. and again follow step number 5. and these time you will see 2 container running

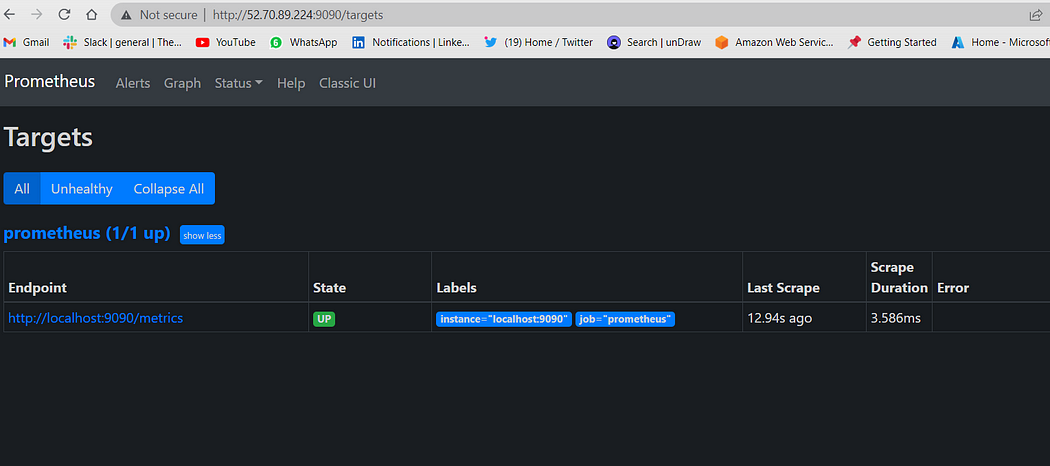


8. Now we install Prometheus and grafana first will see installation of Prometheus and then grafana

##################Prometheus############################  
#Download the latest version of Prometheus using the following command:  
wget https://github.com/prometheus/prometheus/releases/download/v2.30.0/prometheus-2.30.0.linux-amd64.tar.gz  
#Extract the downloaded archive:  
tar xvfz prometheus-2.30.0.linux-amd64.tar.gz  
#Move into the extracted directory:  
cd prometheus-2.30.0.linux-amd64  
#Start Prometheus as a background service:  
./prometheus --config.file=prometheus.yml &

9. After proper installation and run Prometheus you see following output on cli and on browser (public ip:9090)

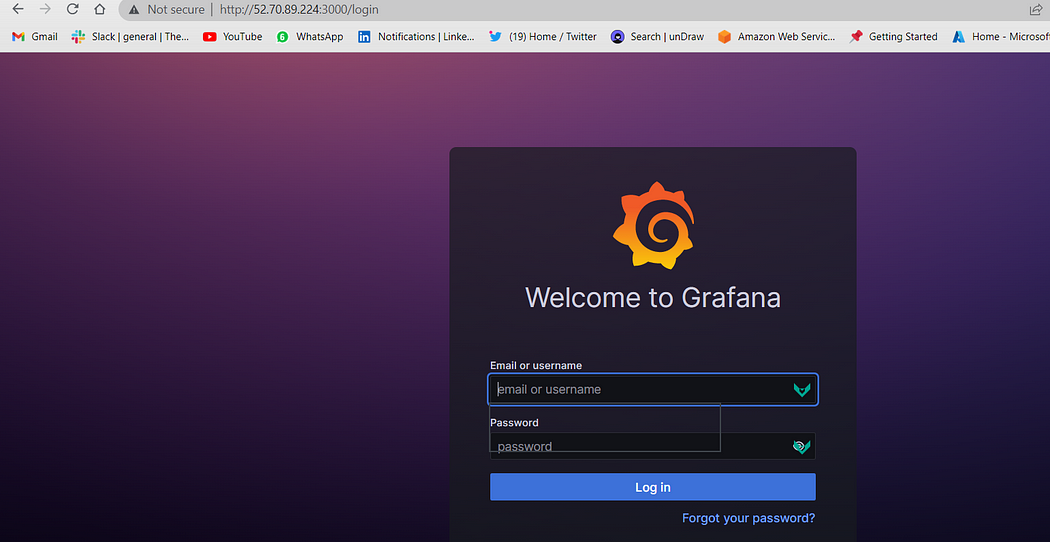




#####################Grafana############################\  
#Import the GPG key used by the Grafana package:  
wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -  
#Add the Grafana repository to the APT sources:  
sudo add-apt-repository "deb https://packages.grafana.com/oss/deb stable main"  
#Update the package lists:  
sudo apt update  
#Install Grafana:  
sudo apt install grafana  
#Start the Grafana service:  
sudo systemctl start grafana-server  
#Enable the Grafana service to start on system boot:  
sudo systemctl enable grafana-server

10. After Proper Installation of Grafana you will see following output on cli and browser (public ip:3000)



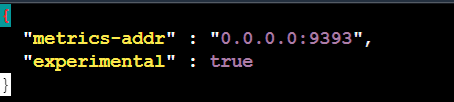


11. You can login grafana with default “User= admin”, “Password=admin”

12. Now modify daemon.json file to gather metrics of docker (there is no by default daemon.json you need to create it)

cd /etc/docker  
vi daemon.json

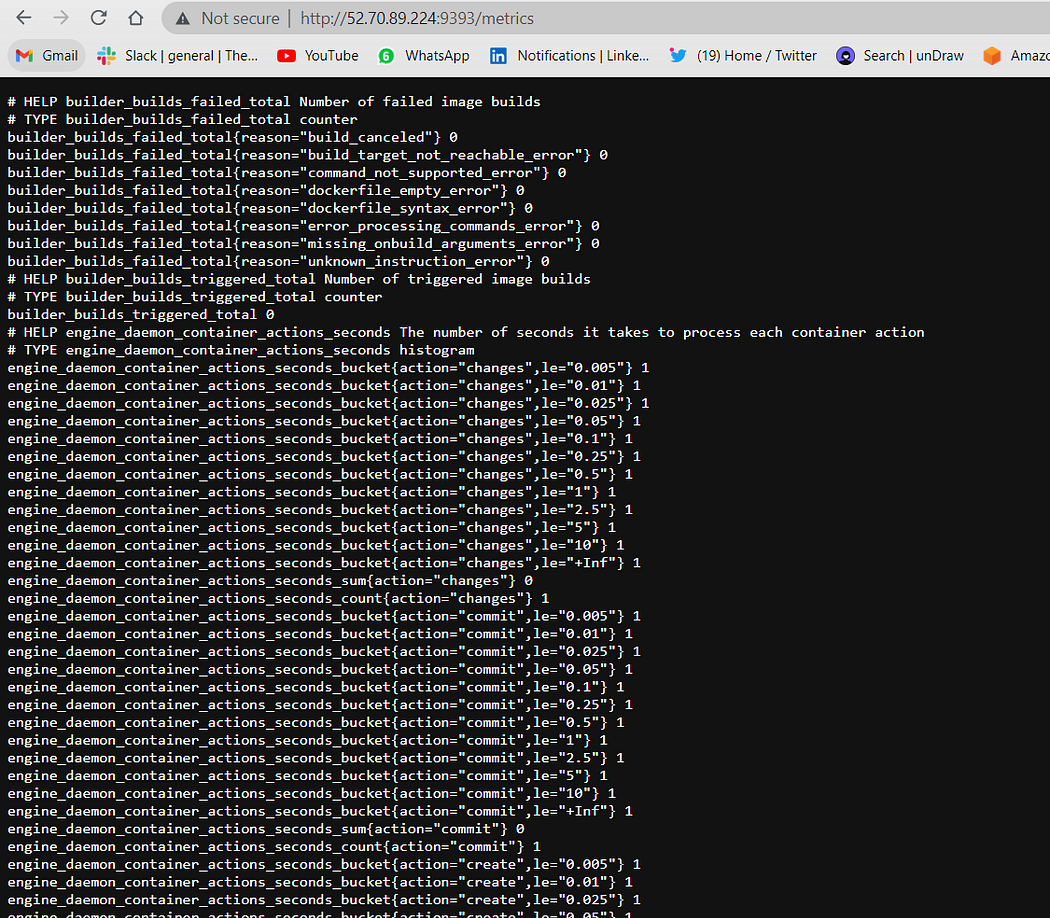
#these is daemon.json file content  
{  
 "metrics-addr" : "0.0.0.0:9393",  
 "experimental" : true  
}



13. Now restart the docker

systemctl restart docker

14. and now hot public ip with port 9393 you will get docker metrics make sure you use following syntax “Public IP:9393/metrics”



15. If you want stop Prometheus (Not required but additional information)(Optional Step*)*

ps aux | grep prometheus  
kill -SIGTERM <PID>

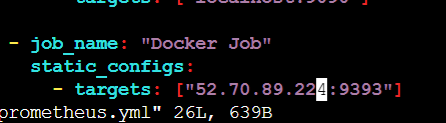
16. Now modify the configuration in promethus.yml where you install Prometheus

cd /home/ubuntu/prometheus-2.30.0.linux-amd64  
vi prometheus.yml

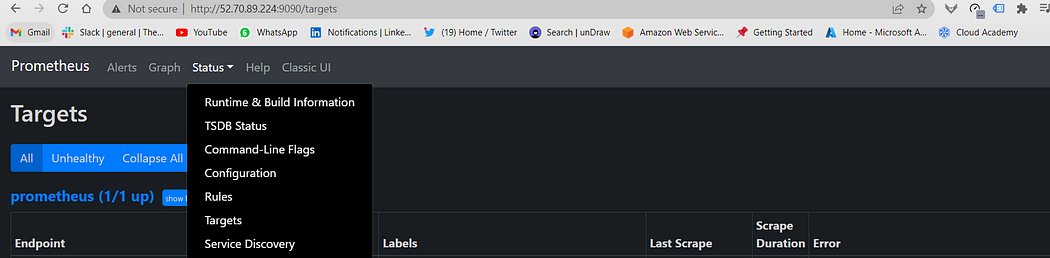
17. Delete default file with “dd” option and paste it following file and save it

# my global config  
global:  
 scrape\_interval: 15s  
 evaluation\_interval: 15s  
  
# 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: Prometheus itself.  
scrape\_configs:  
 - job\_name: "prometheus"  
 static\_configs:  
 - targets: ["localhost:9090"]  
  
 - job\_name: "Docker Job"  
 static\_configs:  
 - targets: ["Public IP:9393"]

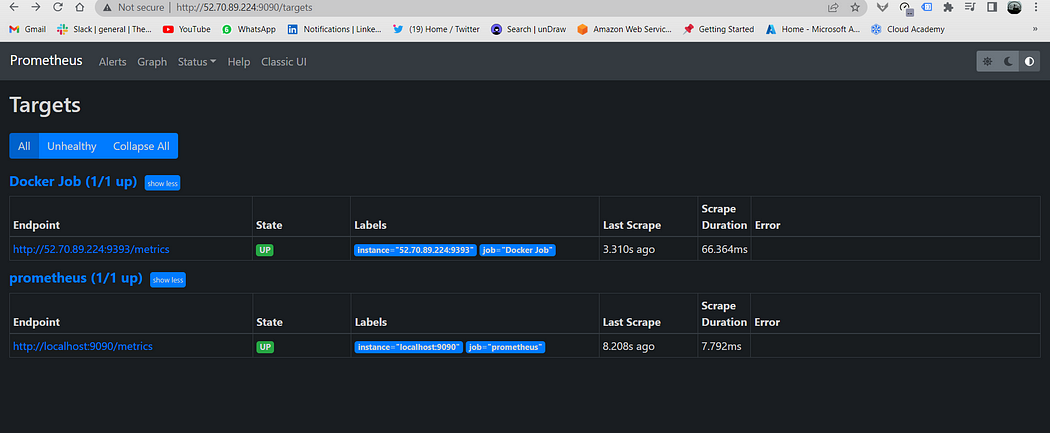
**Note: Make sure you replace the public ip**



18. now go again in Prometheus and check whether target resource is up or not for that go to “Prometheus > Status > targets*”*

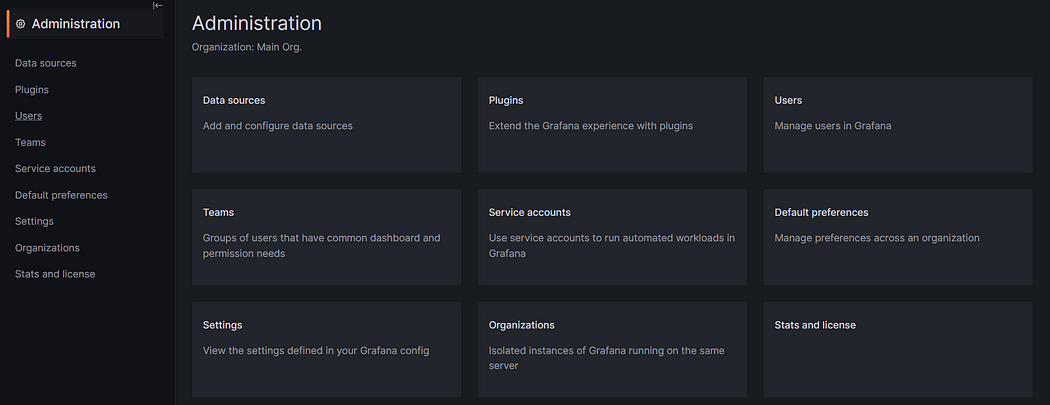


19. If you wont see output then follow step no 15. and again start Prometheus then you will see following output



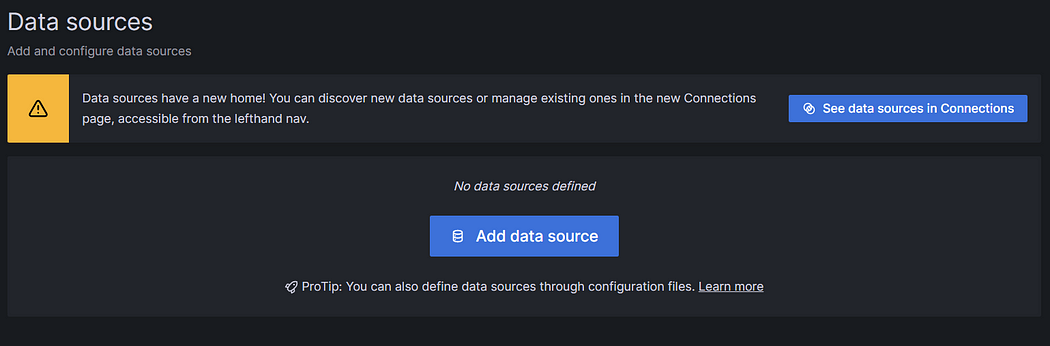
20. then go to grafana login with credentials

21. then click on “Administration => Data Source => Prometheus => and add Prometheus url”



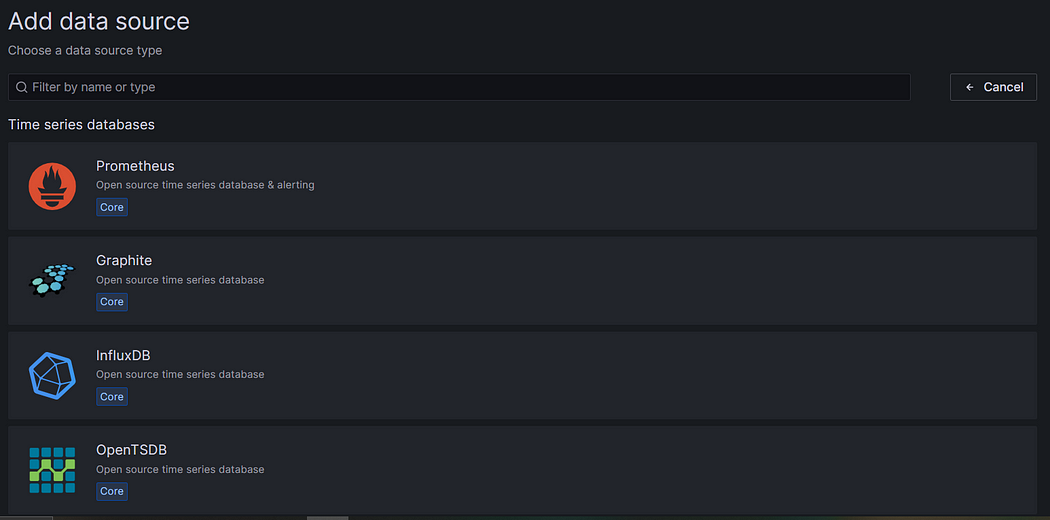
Administration

22. Then click on “Data Source”

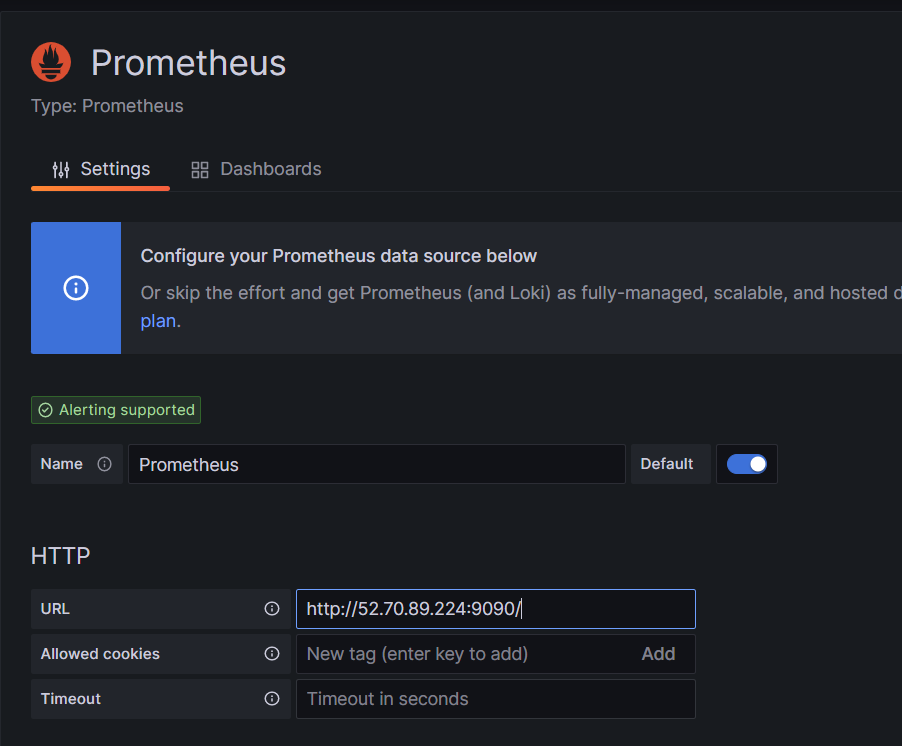


Data Source

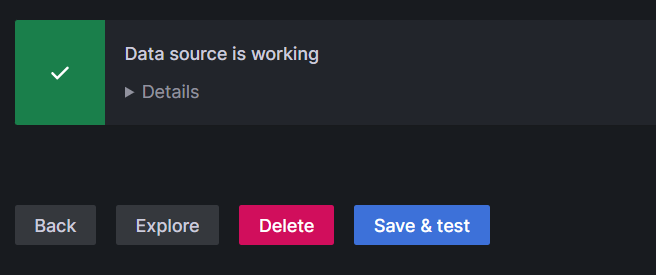
*23*. Click on Add Data Source and select “Prometheus”



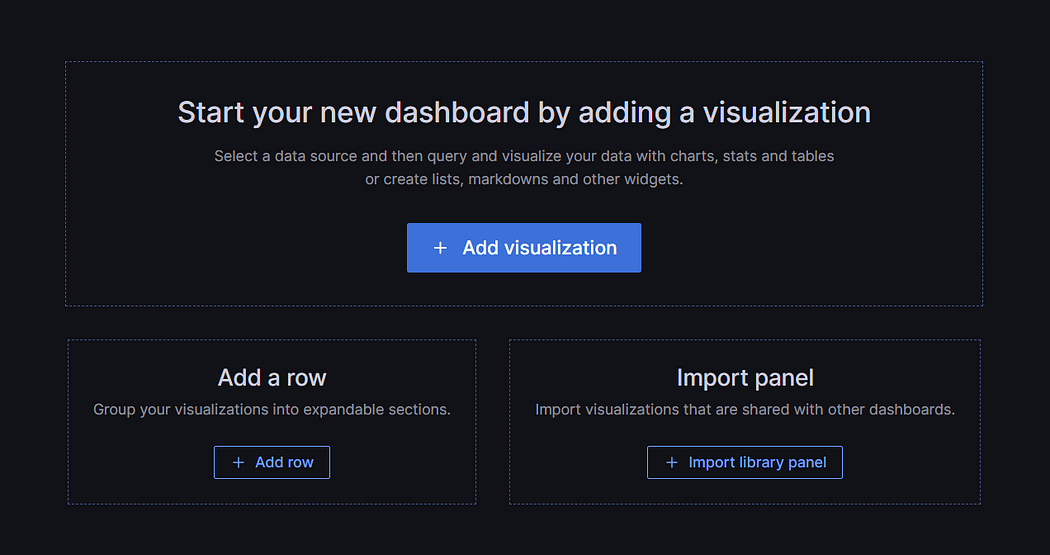
24. In setting paste Prometheus URL and keep other setting as by default and click “Save and Test”



*25.* After click on “Save and Test” you will see below output

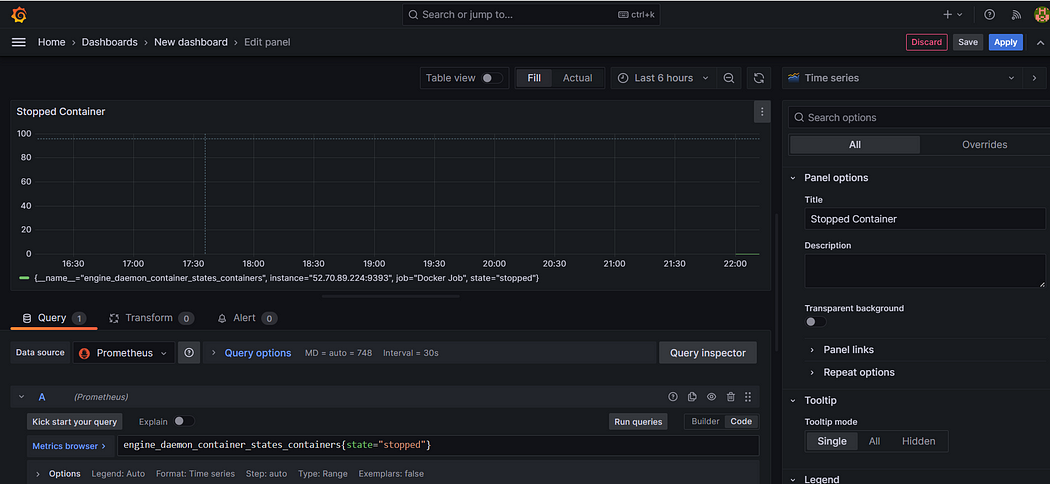


*26. Now go to dashboard and in right hand corner click on “+” icon =>New Dashboard => Add visualisation*



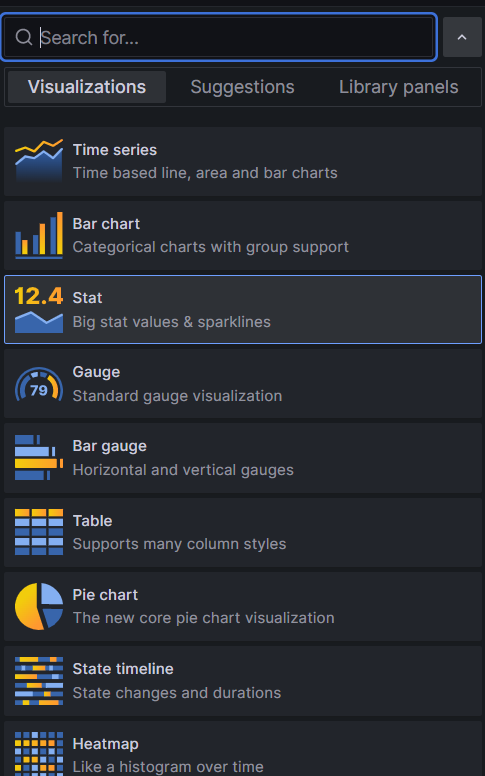
27. In the Metrics browser section add below query. The below query will fetch the number of stopped containers*.*

engine\_daemon\_container\_states\_containers{state="stopped"}



28. then click on apply and save and again click on “+” icon

29. If you want to see data in Stat then click on “Time Series” then you will see multiple visualization, I am selecting “Stat” option



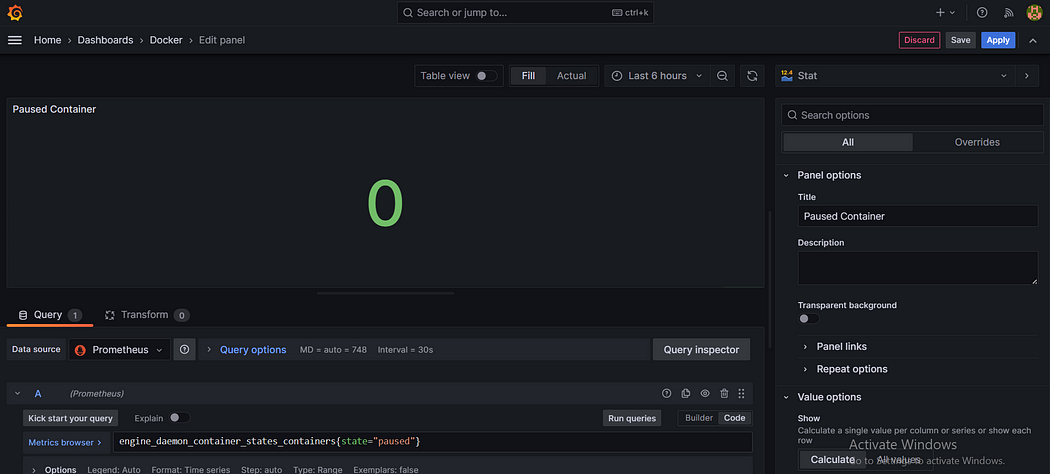
30. Now Follow same process for Paused container,

31. For that click on Add => Visualization



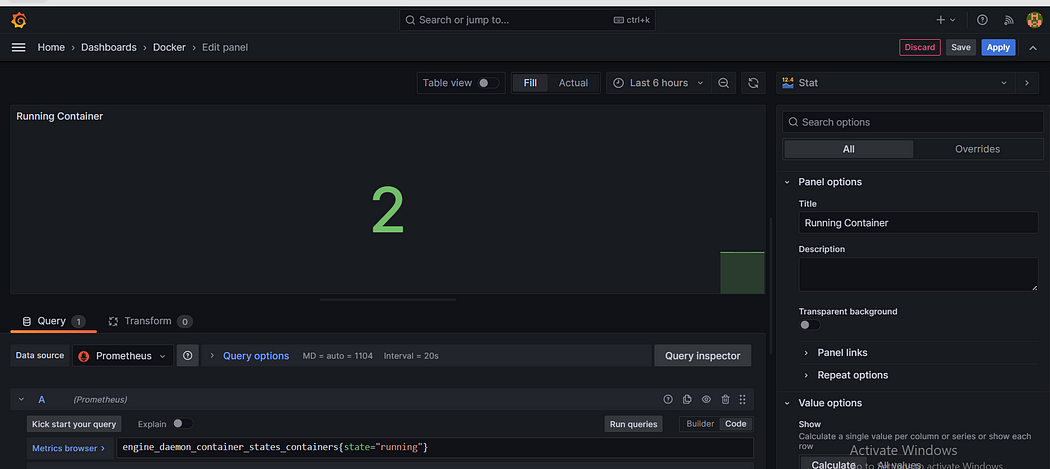
32 . In the Metrics browser section add below query. The below query will fetch the number of paused containers*.*

engine\_daemon\_container\_states\_containers{state="paused"}

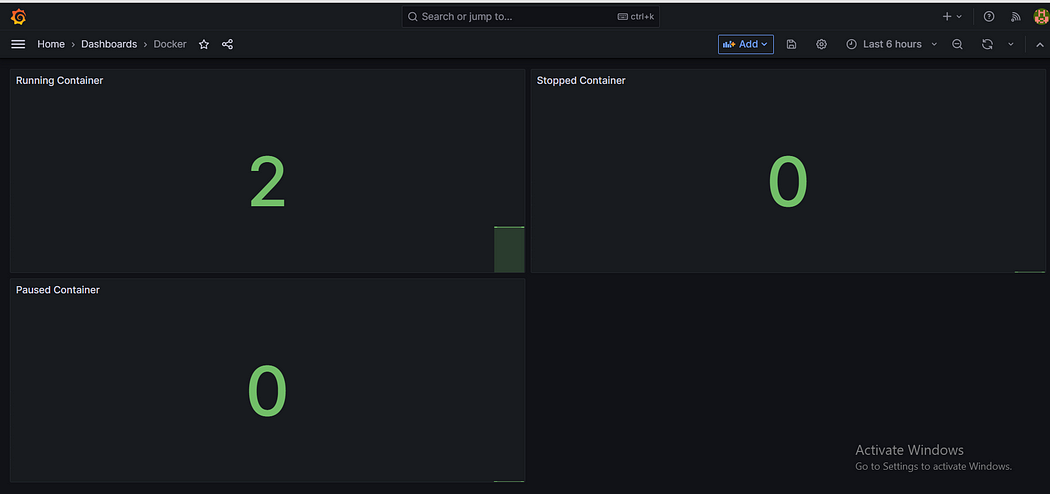


33. Again repeat step 31. In the Metrics browser section add below query. The below query will fetch the number of running containers

engine\_daemon\_container\_states\_containers{state="running"}



34. Now as you see final dashboard as follow to see final dashboard click on “Apply” button give name to Dashboard you will see such below



35. As of now my 2 container are running so we see running container 2, if I stop one then you see following output

