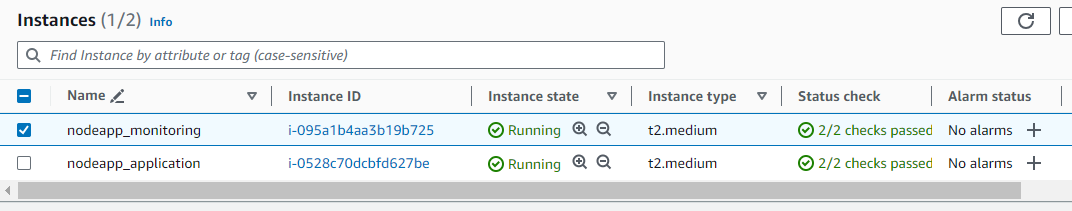
**Configuring, Provisioning, and Monitoring Node.js Application - Capstone Project**

1. GIT Project Used for deployment

<https://github.com/amithtg-199/vue-crud.git>

1. Create a 3 Tyre Infra with help of terraform and create 2 EC2s one for Application and other for Monitoring



1. Install Docker and Minikube & kubectl on Application server.

Docker Installation: <https://docs.docker.com/engine/install/ubuntu/>

Minukube Installation (Linux): <https://minikube.sigs.k8s.io/docs/start/>

Kubectl installation: <https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/>

1. Create Docker file and deployment file for k8s

**Docker file:**

*FROM node:15.6.0 as build-stage*

*RUN mkdir -p /home/node/app/node\_modules*

*WORKDIR /home/node/app*

*COPY package\*.json ./*

*RUN npm install*

*COPY . .*

*RUN npm run load-template simple-crud*

*RUN npm run build*

*FROM nginx:stable-alpine as production-stage*

*COPY --from=build-stage /home/node/app/dist /usr/share/nginx/html*

*EXPOSE 80*

*CMD ["nginx", "-g", "daemon off;"]*

**deployment.yaml:**

Note: Deployment file consists of below services and deployments

1. App deployment with Rolling update
2. Nodeapp service for NodePort
3. HorizontalPodAutoscaler in case if load increases beyond threshold
4. Ingress resource created

*apiVersion: apps/v1*

*kind: Deployment*

*metadata:*

*name: node-deployment*

*spec:*

*replicas: 3*

*strategy:*

*type: RollingUpdate*

*rollingUpdate:*

*maxSurge: 2*

*maxUnavailable: 1*

*selector:*

*matchLabels:*

*name: node-pod*

*template:*

*metadata:*

*labels:*

*name: node-pod*

*spec:*

*containers:*

*- name: node-pod*

*image: amithtg/node\_crud\_app*

*ports:*

*- containerPort: 80*

*resources:*

*requests:*

*cpu: .2*

*memory: 1Gi*

*limits:*

*cpu: .5*

*memory: 2Gi*

*---*

*apiVersion: v1*

*kind: Service*

*metadata:*

*name: node-service*

*labels:*

*name: node-service*

*spec:*

*type: NodePort*

*ports:*

*- port: 80*

*targetPort: 80*

*nodePort: 30004*

*selector:*

*name: node-pod*

*---*

*apiVersion: autoscaling/v1*

*kind: HorizontalPodAutoscaler*

*metadata:*

*name: node-app-hpa*

*spec:*

*scaleTargetRef:*

*apiVersion: apps/v1*

*kind: Deployment*

*name: node-deployment*

*minReplicas: 3*

*maxReplicas: 10*

*targetCPUUtilizationPercentage: 50*

*---*

*apiVersion: networking.k8s.io/v1*

*kind: Ingress*

*metadata:*

*name: node-ingress*

*annotations:*

*nginx.ingress.kubernetes.io/rewrite-target: /$1*

*spec:*

*rules:*

*- host: www.app.com*

*http:*

*paths:*

*- path: /*

*pathType: Prefix*

*backend:*

*service:*

*name: node-service*

*port:*

*number: 80*

1. Build Docker image and upload to docker hub

*git clone* [*https://github.com/amithtg-199/vue-crud.git*](https://github.com/amithtg-199/vue-crud.git)

*cd vue-crud*

*docker build -t amithtg/node\_crud\_app .*

**Upload the image to docker hub**

*docker images* 🡪 obtain the docker name:tag

Run:

*docker login*

*username:* 🡪Input docker hub username

*password:* 🡪Input docker hub password

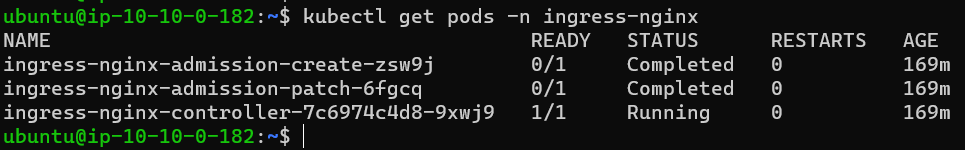
Push the docker image to docker hub

*docker push image\_name:tag*

1. Install NGINX-Ingress-Controller

*minikube addons enable ingress* 🡪 enabling Ingress controller

*kubectl get pods -n ingress-nginx* 🡪 verify if its running

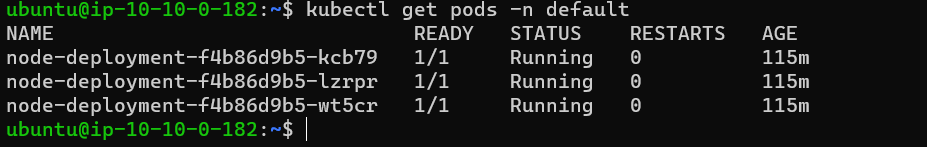


1. Create K8s resource for the Node.js CRUD application

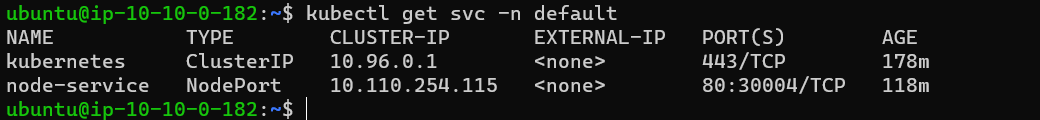
*kubectl create -f deployment.yaml*

**check the created resources:**

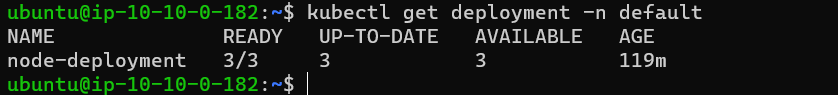
*kubectl get pods -n default*



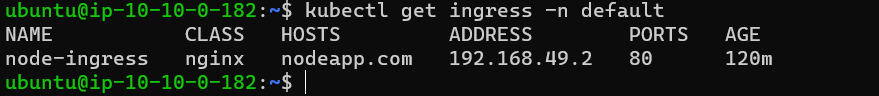
*kubectl get svc -n default*



*kubectl get deployment -n default*



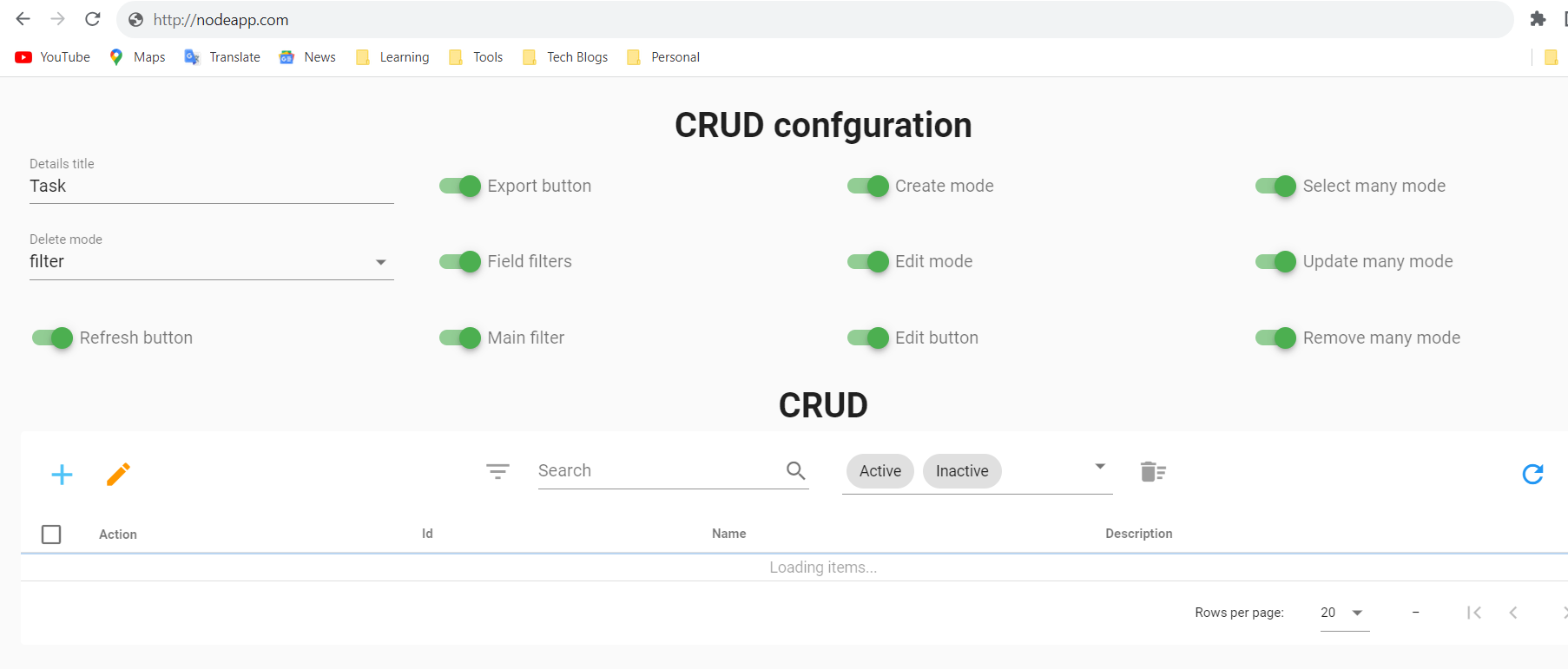
*kubectl get ingress -n default*



1. As we are not using any External LB we need to enable port forwarding:

*nohup kubectl port-forward --address 0.0.0.0 services/pyapp-service 30004:80 > /dev/null 2>&1 &*

1. Access the Application from your workstation



1. Configuring Prometheus and Grafana
2. Install Node exporter on Application and Monitoring server:

*sudo apt-get install -y prometheus-node-exporter*

1. Install Prometheus and Grafana on Monitoring node

Install Prometheus:

*wget https://github.com/prometheus/prometheus/releases/download/v2.29.1/prometheus-2.29.1.linux-amd64.tar.gz*

*tar -xvf prometheus-2.29.1.linux-amd64.tar.gz*

Install Grafana:

*wget -q -O - https://packages.grafana.com/gpg.key | gpg --dearmor | sudo tee /usr/share/keyrings/grafana.gpg > /dev/null*

*echo "deb [signed-by=/usr/share/keyrings/grafana.gpg] https://packages.grafana.com/oss/deb stable main" | sudo tee -a /etc/apt/sources.list.d/grafana.list*

*sudo apt update*

*sudo apt install -y grafana*

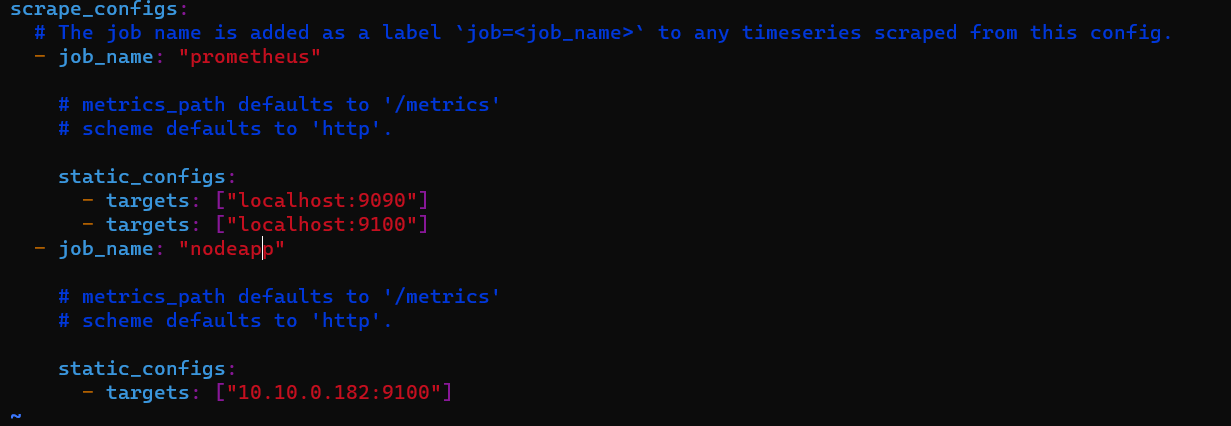
*sudo systemctl start grafana-server*

1. Configure Prometheus config file and add targets for data scraping

*cd prometheus-2.29.1.linux-amd64/*

*vi prometheus.yml*

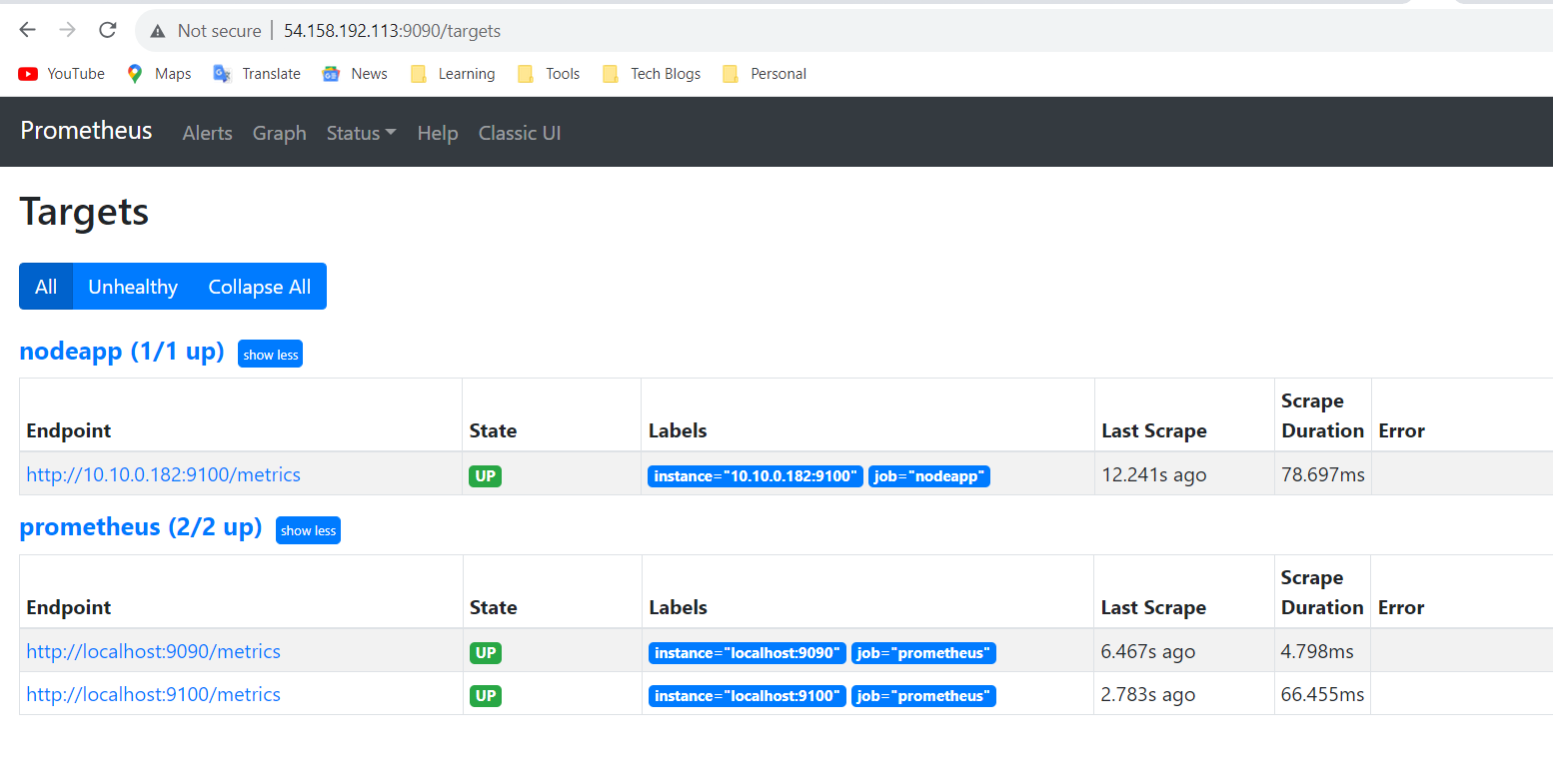
Configure as below add the Private IPs of EC2 machines, node-exporter runs on 9100 and HTTP Server runs on 9090



1. Start Prometheus:

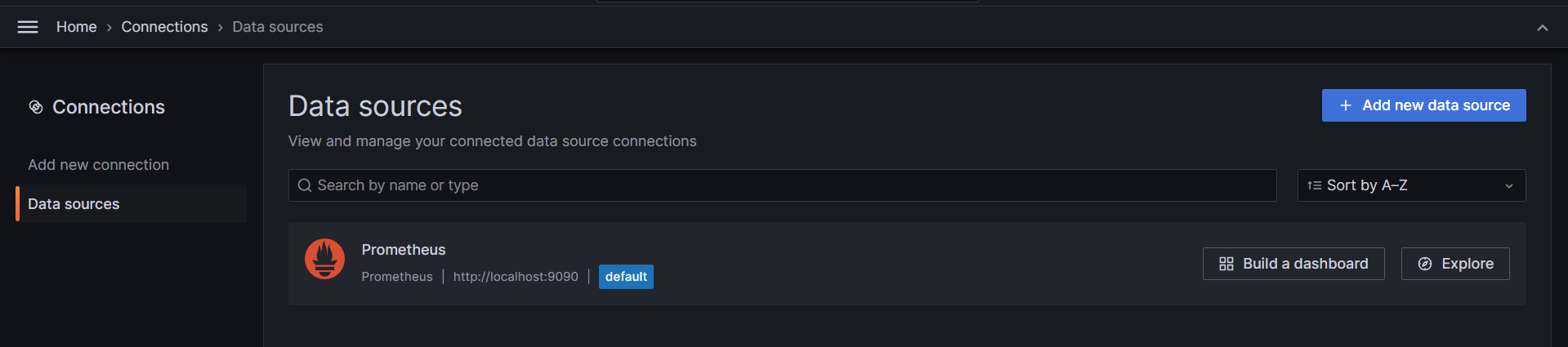
*./prometheus &*

1. Login to Prometheus Ui and check the target status

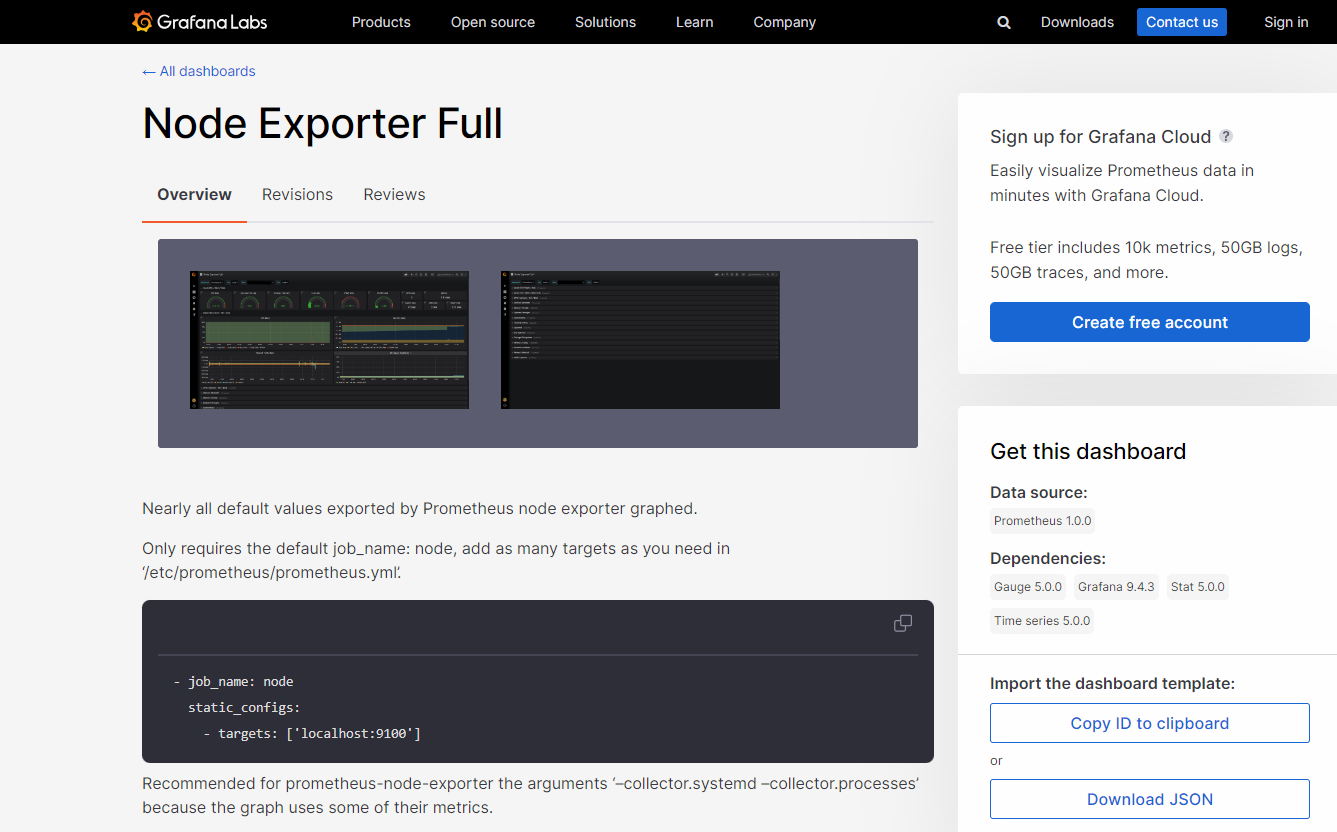


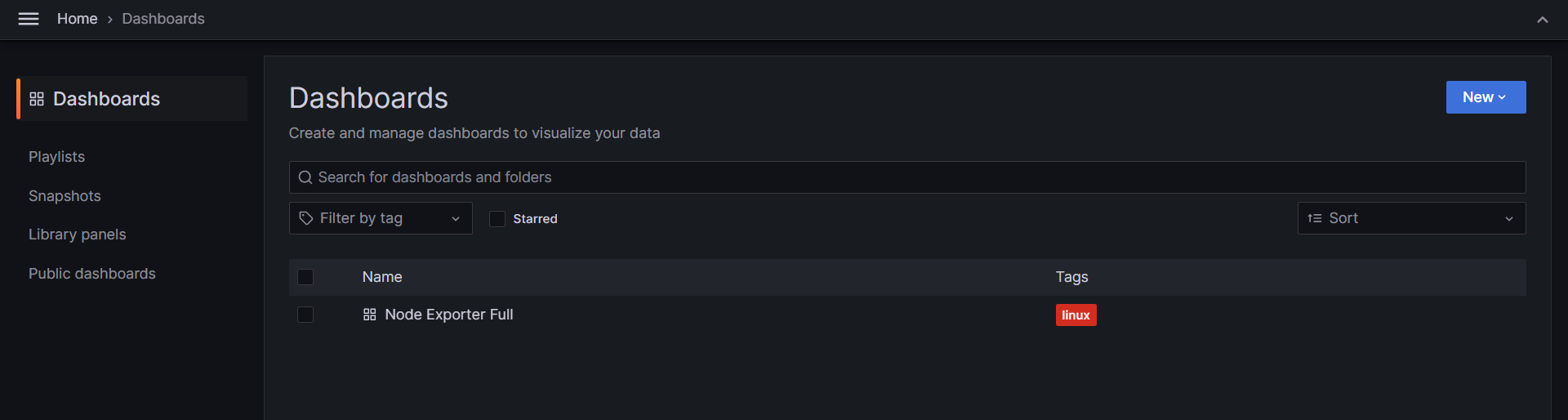
1. Configure Grafana Dash board

Login to Grafana UI and add Prometheus data source



Now Create DashBoard import the node-exporter ID from Grafana.com and import the same.





1. Dashboard looks something as below

