B.M.S. COLLEGE OF ENGINEERING
Autonomous Institute, Affiliated to VTU
Estd. 1946

DEPARTMENT OF CSE
CTY Project Work In collaboration with HPE

Project Title	Open source monitoring and observability stack on Kubernetes		
Student Team	Name: Pooja Srinivasan USN:1BM18CS069 SEM:UG-6th SEM	Name: Anusree Manoj K USN:1BM18CS017 SEM:UG-6th SEM	
	Name: Shikha N USN:1BM18CS149 SEM:UG-6th SEM	Name: Niha USN:1BM18CS060 SEM:UG-6th SEM	
Faculty Mentor	Dr. Nandhini V Associate Professor	**HPE Mentors**	Divakar Padiyar Sonu Sudhakaran
Review for the Period	18-05-2021	20-06-2021	
Task Given	1. Use Grafana and Prometheus stack on your cluster to observe metrics of your cluster. 2. Add a custom Grafana dashboard. 3. Have multiple clusters one using MicroK8S & another using Minikube, and enable Prometheus stack on each of them. And deploy Grafana to monitor both the clusters. 4. Installation of K3s and creating a dashboard using Grafana for a K3s cluster.		
Difficulties Faced	None		
Libraries Used	None		
Github Link for the code:	None		
Code:	None		

What is Prometheus?	Prometheus collects metrics from configured targets at given intervals, evaluates rule expressions, displays the results, and can trigger alerts when specific conditions are observed. Prometheus gathers metrics from the Kubernetes endpoints discussed in the previous sections. Prometheus is closely associated with Alertmanager. Describing the deployment steps of Prometheus is outside the scope of this document. However, you should be aware of which of the few deployment layouts is at hand. The use case we expect to have is a number of MicroK8s clusters all sending metrics to a central Prometheus installation. A few ways to achieve this layout are: ● Scrape remote k8s clusters: Run the prometheus node-exporter and the prometheus adapter for kubernetes metrics APIs (or any other exporter) to gather information from each MicroK8s cluster to a central Prometheus installation. ● Remote Prometheus as Grafana data sources: Run the entire Prometheus on each cluster and have a central Grafana that would view each Prometheus as a different data source. In this case the Prometheus service needs to be exposed and be reachable outside the K8s cluster. ● Federation: With federation you can consolidate selected metrics from multiple k8s clusters.
What is Grafana?	**Grafana** is a multi-platform open source analytics and interactive visualization web application. It provides charts, graphs, and alerts for the web when connected to supported data sources. A licensed Grafana Enterprise version with additional capabilities is also available as a self-hosted installation or an account on the Grafana Labs cloud service. It is expandable through a plug-in system. End users can create complex monitoring dashboards using interactive query builders. Grafana is divided into a front end and back end, written in TypeScript and Go, respectively. As a visualization tool, Grafana is a popular component in monitoring stacks, often used in combination with time series databases such as InfluxDB, Prometheus and Graphite; monitoring platforms such as Sensu, Icinga, Checkmk, Zabbix, Netdata, and PRTG; SIEMs such as Elasticsearch and Splunk; and other data sources.

Deployment of prometheus and grafana in microK8s

[1]

Prometheus and grafana are added to microK8s cluster with the help of built-in add-on.

- Enable Prometheus add-on to monitor metrics on MicroK8s Cluster.
- Enabling built-in Prometheus add-on on primary Node.

```
hpe@hpe-VirtualBox:~$ microk8s enable prometheus dashboard dns
Addon dns is already enabled.
Fetching kube-prometheus version v0.7.0.
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   143  100   143    0     0    240      0 --:--:-- --:--:-- --:--:--   240
100  287k    0  287k    0     0   198k      0 --:--:--  0:00:01 --:--:--  198k
kube-prometheus-0.7.0/
kube-prometheus-0.7.0/.github/
kube-prometheus-0.7.0/.github/ISSUE_TEMPLATE/
kube-prometheus-0.7.0/.github/ISSUE_TEMPLATE/bug.md
kube-prometheus-0.7.0/.github/ISSUE_TEMPLATE/feature.md
kube-prometheus-0.7.0/.github/ISSUE_TEMPLATE/support.md
kube-prometheus-0.7.0/.github/workflows/
kube-prometheus-0.7.0/.github/workflows/ci.yaml
kube-prometheus-0.7.0/.gitignore
kube-prometheus-0.7.0/DCO
kube-prometheus-0.7.0/LICENSE
kube-prometheus-0.7.0/Makefile
kube-prometheus-0.7.0/NOTICE
kube-prometheus-0.7.0/OWNERS
kube-prometheus-0.7.0/README.md
kube-prometheus-0.7.0/build.sh
kube-prometheus-0.7.0/code-of-conduct.md
kube-prometheus-0.7.0/docs/
kube-prometheus-0.7.0/docs/EKS-cni-support.md
kube-prometheus-0.7.0/docs/GKE-cadvisor-support.md
kube-prometheus-0.7.0/docs/community-support.md
kube-prometheus-0.7.0/docs/developing-prometheus-rules-and-grafana-dashboards.m
```

- Monitoring the number services and pods running to check whether prometheus and grafana are running or not:

```
hpe@hpe-VirtualBox:~$ microk8s kubectl get pod -n monitoring
NAME                                   READY   STATUS              RESTARTS   A
GE
prometheus-operator-7649c7454f-d9dqr   0/2     ContainerCreating   0          3
m7s
node-exporter-vmwcg                    0/2     ContainerCreating   0          2
m45s
kube-state-metrics-78dc55b74b-jrbfp    0/3     ContainerCreating   0          2
m47s
grafana-6b8df57c5b-z5qcj               0/1     ContainerCreating   0          2
m49s
prometheus-adapter-69b8496df6-pf56c    0/1     ContainerCreating   0          2
m23s
alertmanager-main-0                    0/2     ContainerCreating   0          1
07s
prometheus-k8s-0                       0/2     ContainerCreating   0          1
```

```
hpe@hpe-VirtualBox:~$ microk8s kubectl get services -n monitoring
NAME                    TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)
                  AGE
prometheus-operator     ClusterIP   None             <none>        8443/TCP
                  3m5s
alertmanager-main       ClusterIP   10.152.183.149   <none>        9093/TCP
                  2m51s
grafana                 ClusterIP   10.152.183.245   <none>        3000/TCP
                  2m24s
kube-state-metrics      ClusterIP   None             <none>        8443/TCP,944
3/TCP             2m22s
node-exporter           ClusterIP   None             <none>        9100/TCP
                  2m21s
prometheus-adapter      ClusterIP   10.152.183.37    <none>        443/TCP
                  2m5s
alertmanager-operated   ClusterIP   None             <none>        9093/TCP,909
4/TCP,9094/UDP    85s
prometheus-operated     ClusterIP   None             <none>        9090/TCP
                  84s
prometheus-k8s          ClusterIP   10.152.183.135   <none>        9090/TCP
                  79s
```

- Prometheus port-forwarding to enable external access:

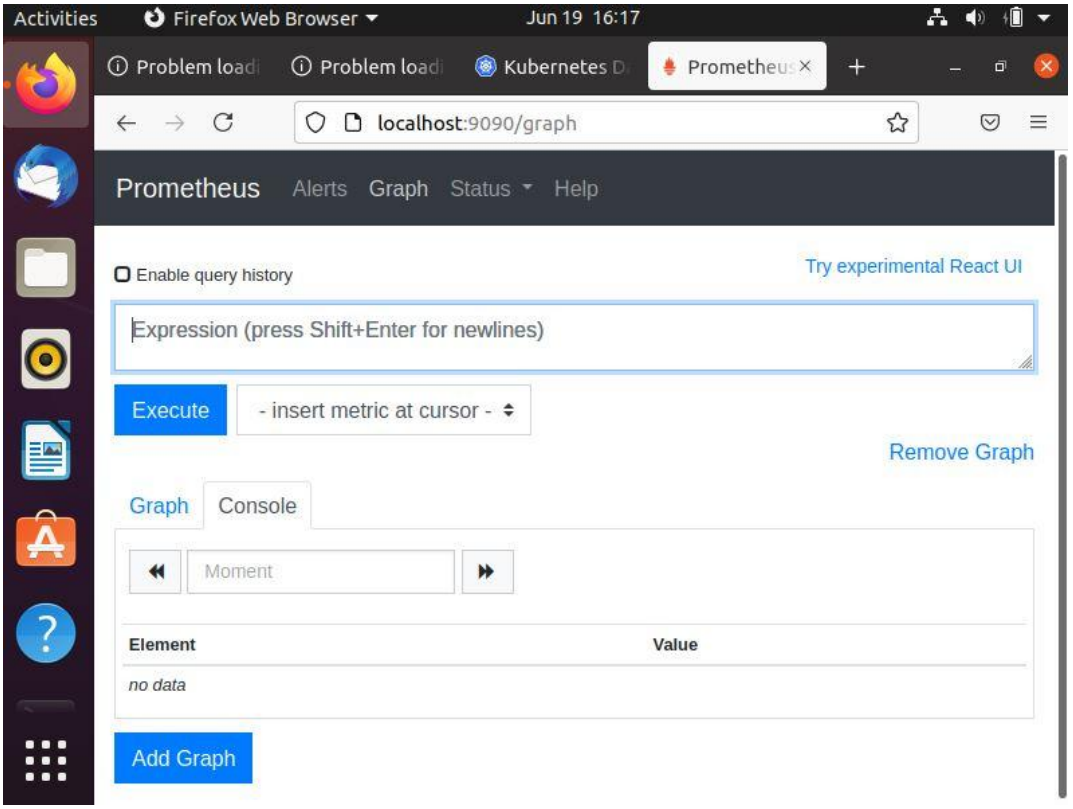
```
hpe@hpe-VirtualBox:~$ microk8s kubectl port-forward -n monitoring services/prom
etheus-k8s --address 0.0.0.0 9090:9090
Forwarding from 0.0.0.0:9090 -> 9090
Handling connection for 9090
```

- Grafana port-forwarding to enable external access:

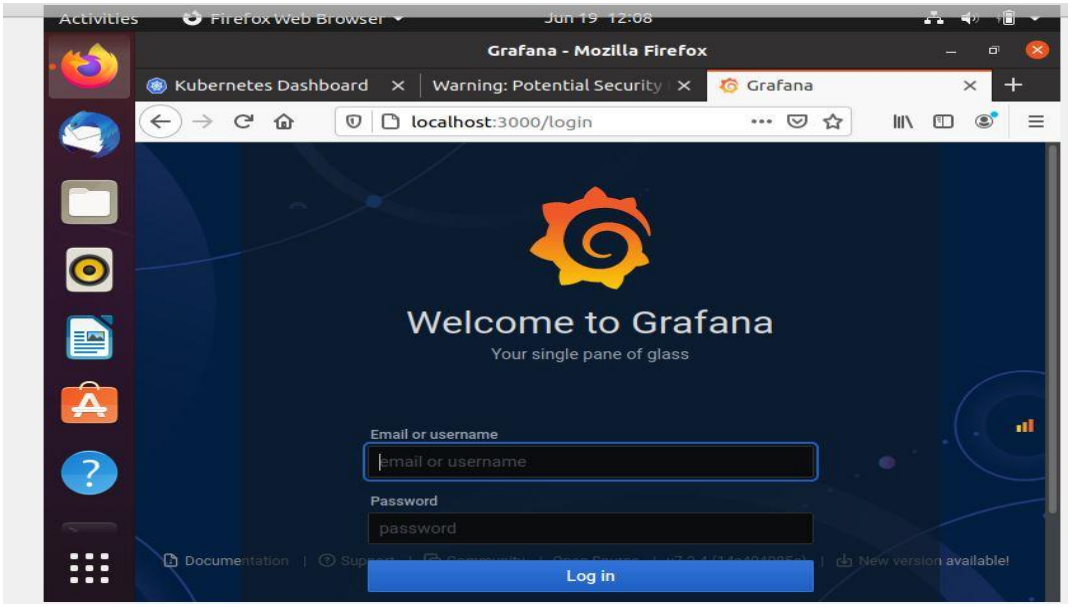
```
hpe@hpe-VirtualBox:~$ microk8s kubectl port-forward -n monitoring services/graf
ana --address 0.0.0.0 3000:3000
Forwarding from 0.0.0.0:3000 -> 3000
Handling connection for 3000
Handling connection for 3000
```

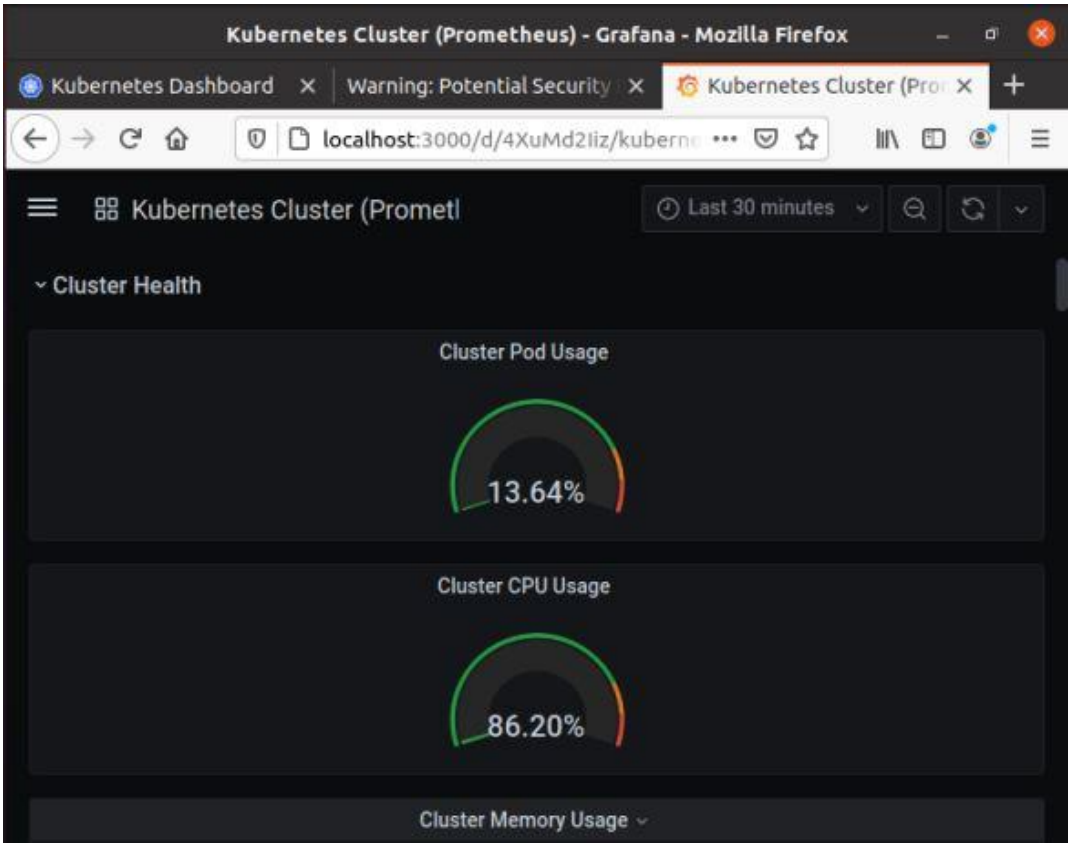
Connection between the prometheus and grafana for monitoring and observing is done via adding data source that is prometheus and importing the dashboard from Grafana.com using 6417. In the dashboard configuration select the Prometheus Datasource. And hence we can visualize using grafana.

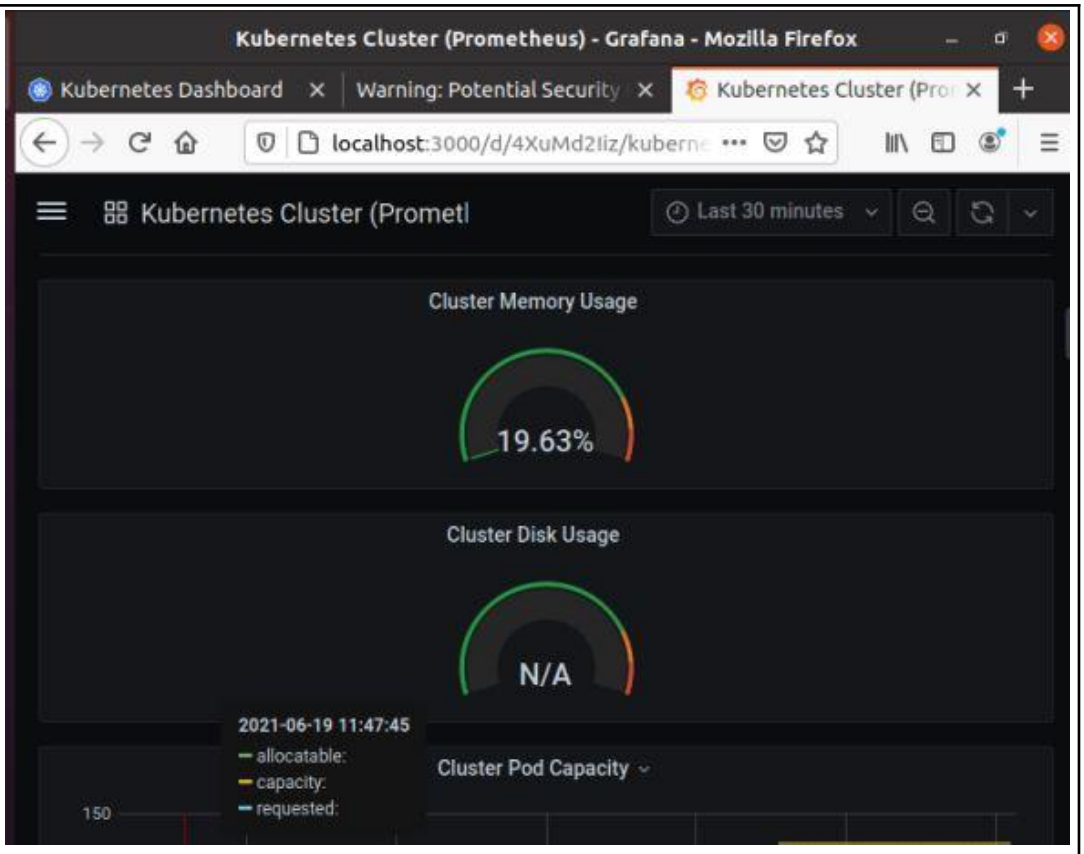
- Prometheus dashboard:



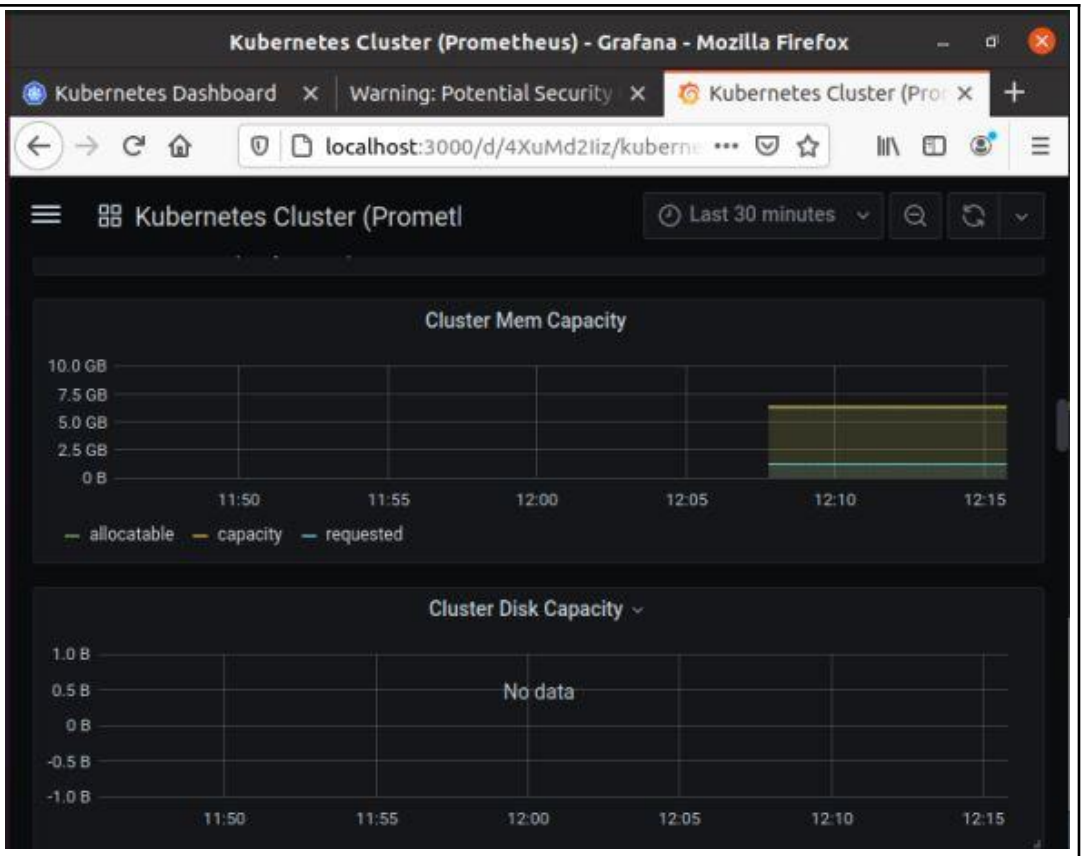
- Grafana dashboard: Visualization of cluster
  It shows the metrics for cluster, nodes, pods, jobs and deployments running on it.

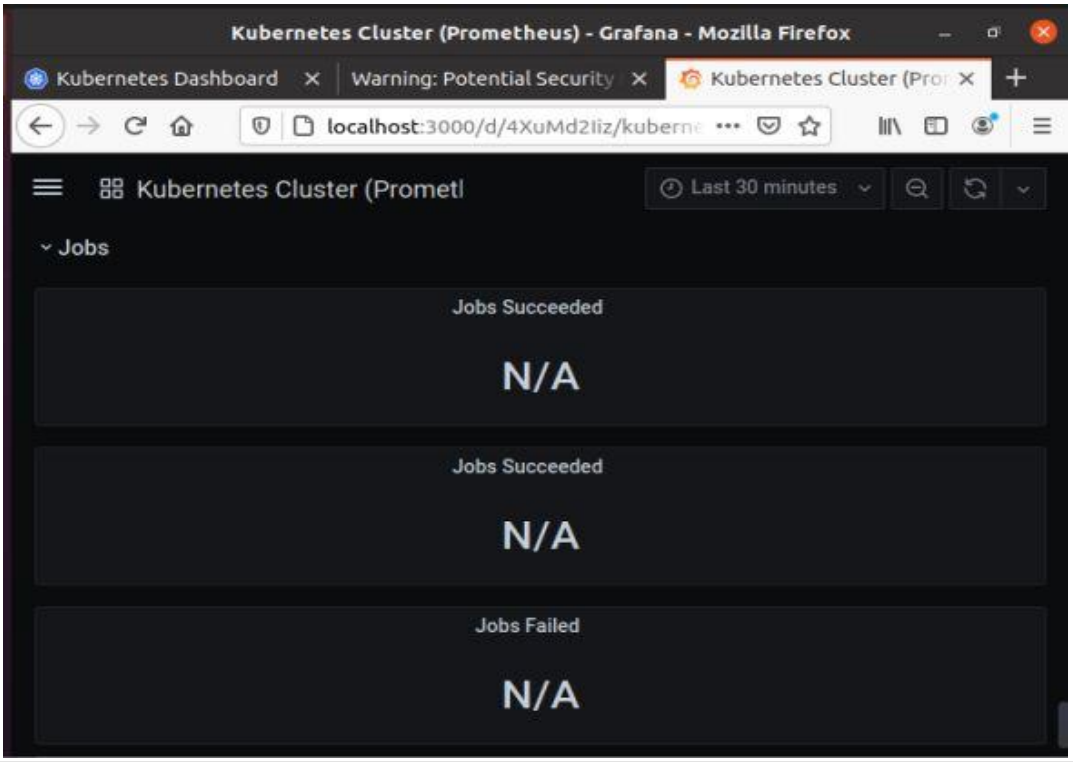


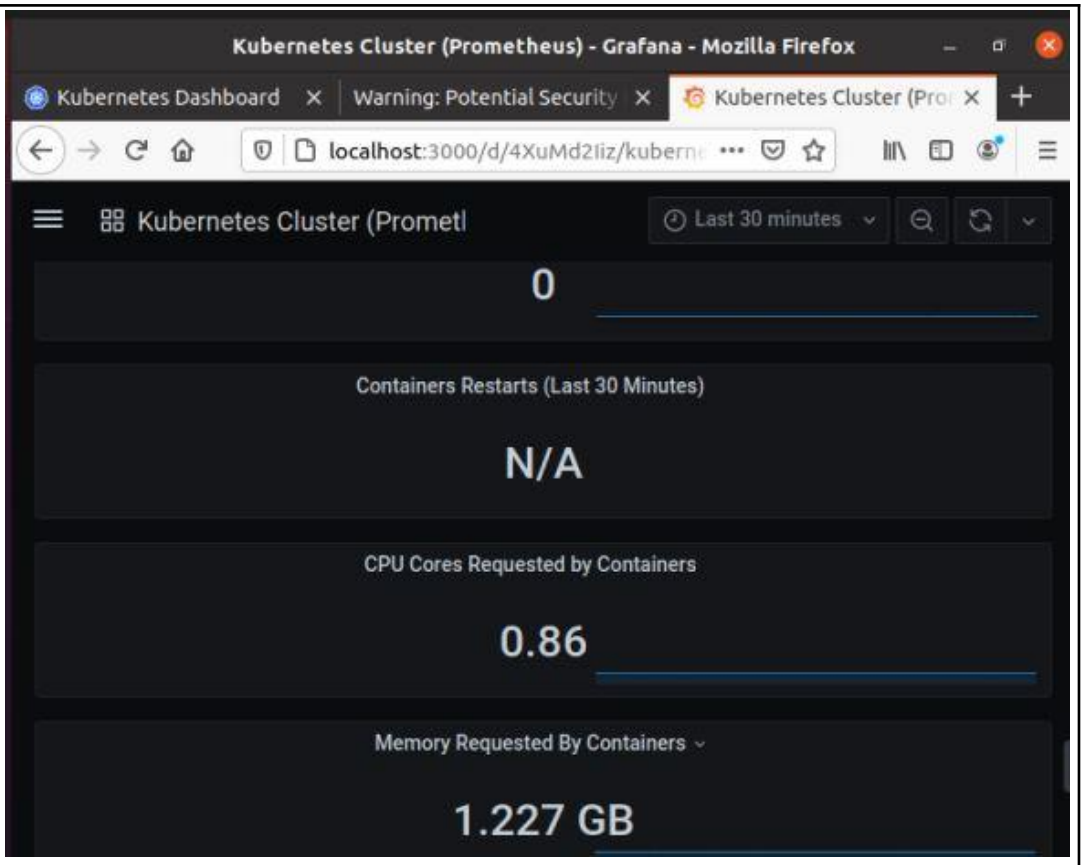


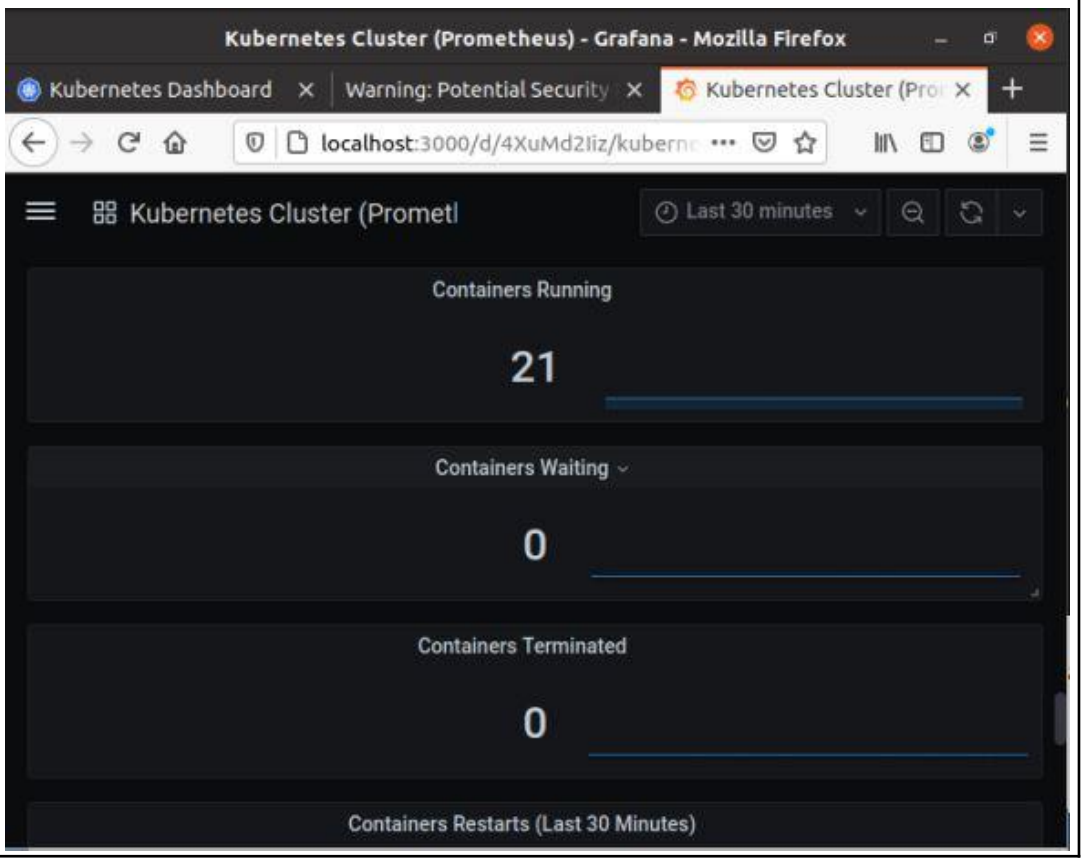

Kubernetes Cluster (Prometheus) - Grafana - Mozilla Firefox
Kubernetes Dashboard
Warning: Potential Security
Kubernetes Cluster (Pro
localhost:3000/d/4XuMd2Iiz/kuberne
Kubernetes Cluster (Prometl
Last 30 minutes
Cluster Memory Usage
19.63%
Cluster Disk Usage
N/A
2021-06-19 11:47:45
allocatable:
capacity:
requested:
Cluster Pod Capacity
150

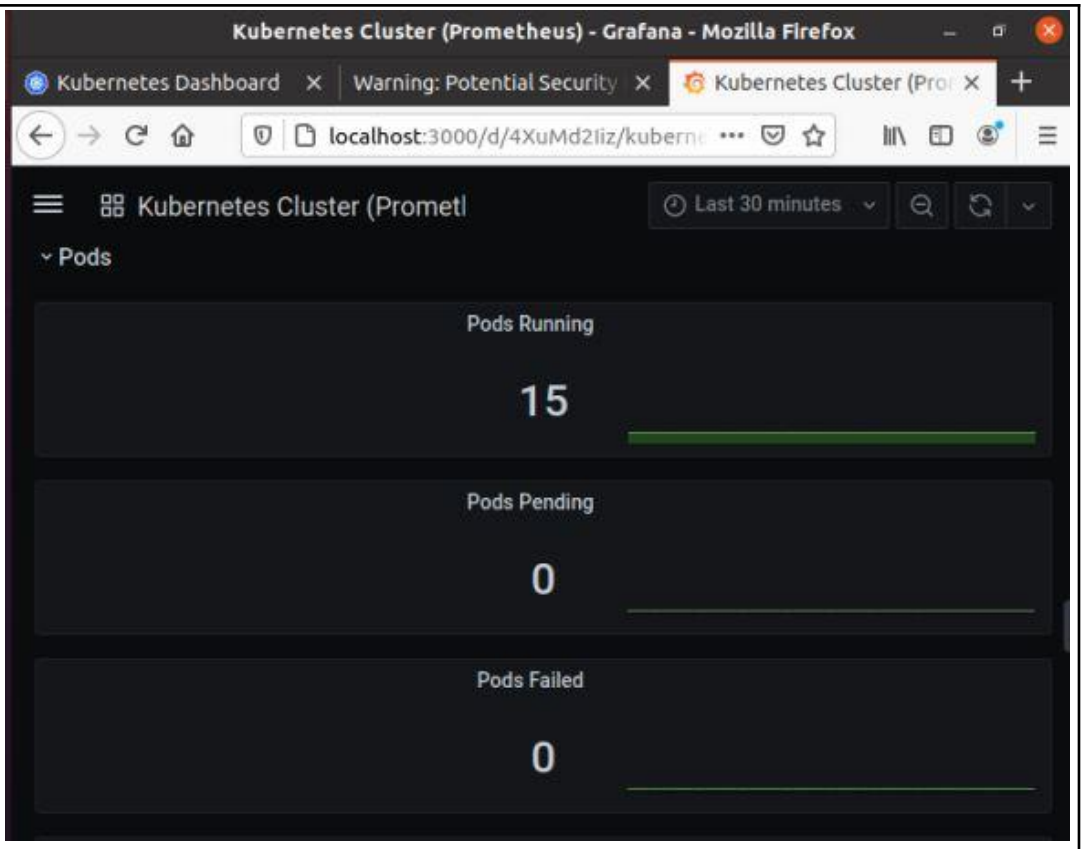

Cluster Pod Capacity
150
100
50
0
pods
11:50
11:55
12:00
12:05
12:10
12:15
allocatable
capacity
requested
Cluster CPU Capacity
1.05
1.00
0.95
0.90
0.85
cores
11:50
11:55
12:00
12:05
12:10
12:15
allocatable
capacity
requested

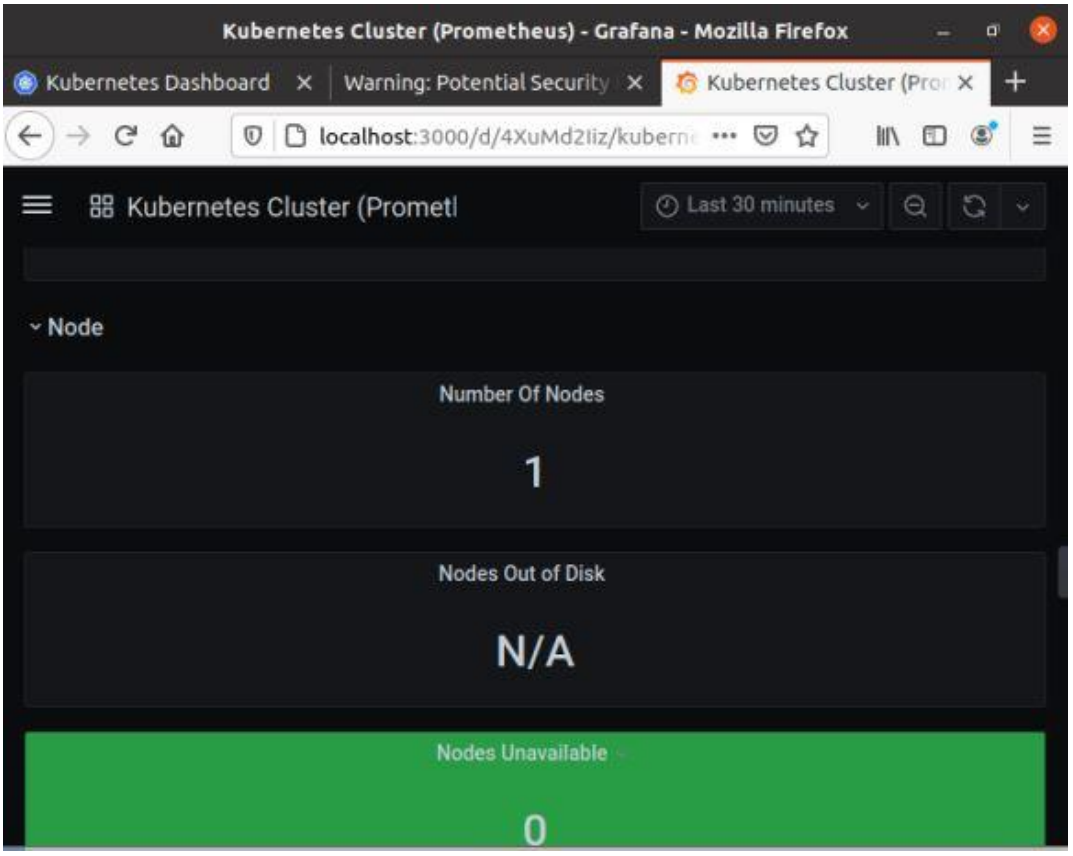

Kubernetes Cluster (Prometheus) - Grafana - Mozilla Firefox
Kubernetes Dashboard
Warning: Potential Security
Kubernetes Cluster (Pro
localhost:3000/d/4XuMd2Iiz/kuberne
Kubernetes Cluster (Prometl
Last 30 minutes
Cluster Mem Capacity
10.0 GB
7.5 GB
5.0 GB
2.5 GB
0 B
11:50
11:55
12:00
12:05
12:10
12:15
allocatable
capacity
requested
Cluster Disk Capacity
1.0 B
0.5 B
0 B
-0.5 B
-1.0 B
No data

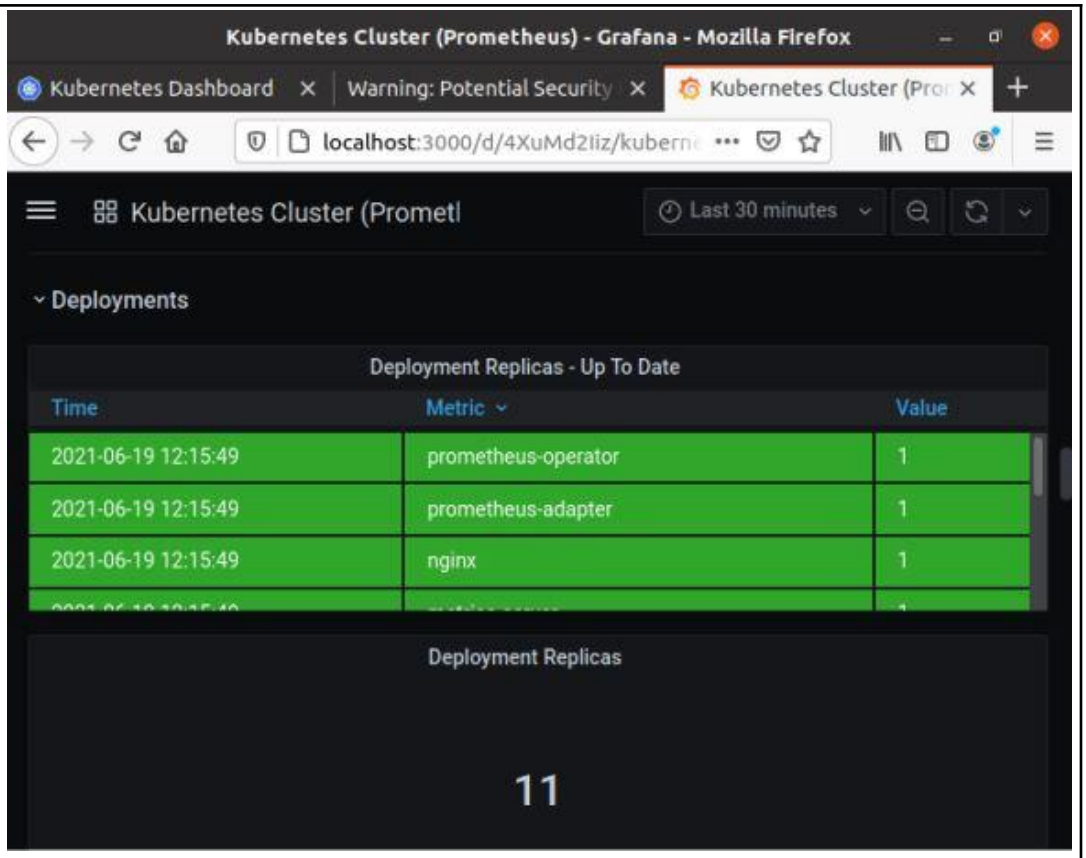

Kubernetes Cluster (Prometheus) - Grafana - Mozilla Firefox
Kubernetes Dashboard
Warning: Potential Security
Kubernetes Cluster (Pro
localhost:3000/d/4XuMd2Iiz/kuberne
Kubernetes Cluster (Prometl
Last 30 minutes
Jobs
Jobs Succeeded
N/A
Jobs Succeeded
N/A
Jobs Failed
N/A

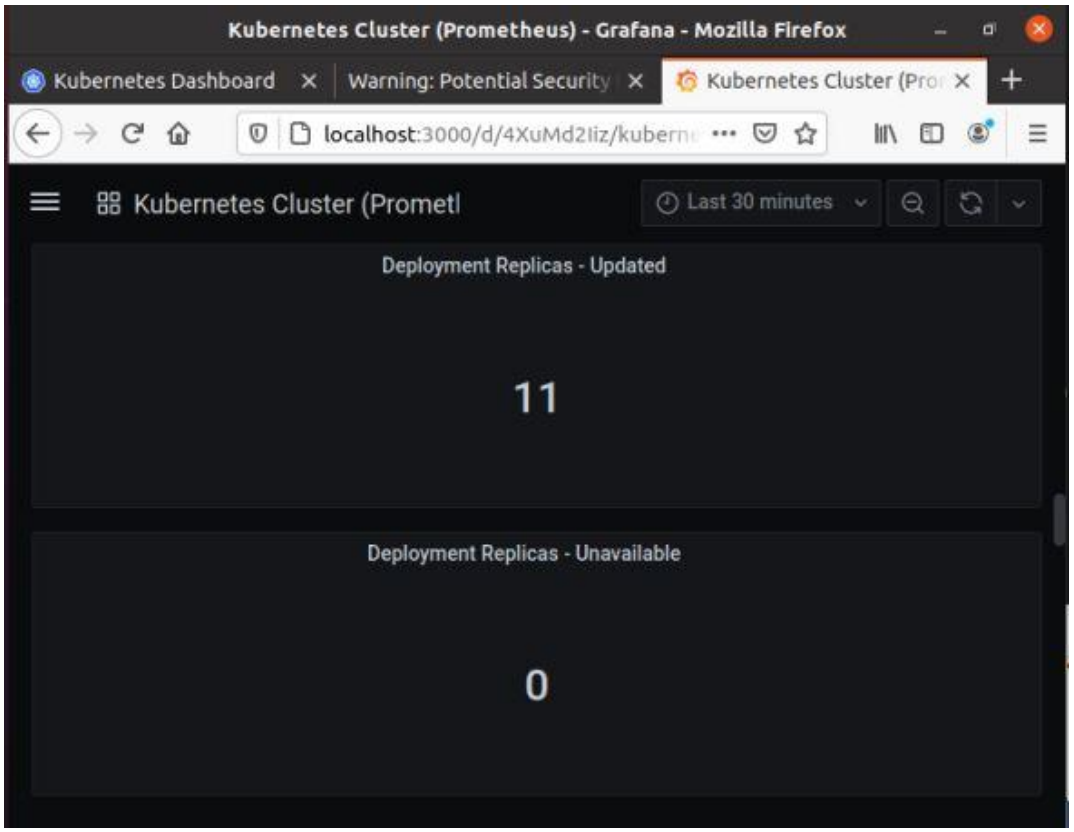

Kubernetes Cluster (Prometheus) - Grafana - Mozilla Firefox
Kubernetes Dashboard
Warning: Potential Security
Kubernetes Cluster (Pro
localhost:3000/d/4XuMd2Iiz/kubern
Kubernetes Cluster (Prometl
Last 30 minutes
0
Containers Restarts (Last 30 Minutes)
N/A
CPU Cores Requested by Containers
0.86
Memory Requested By Containers
1.227 GB


Kubernetes Cluster (Prometheus) - Grafana - Mozilla Firefox
Kubernetes Dashboard
Warning: Potential Security
Kubernetes Cluster (Pro
localhost:3000/d/4XuMd2Iiz/kubern
Kubernetes Cluster (Prometl
Last 30 minutes
Containers Running
21
Containers Waiting
0
Containers Terminated
0
Containers Restarts (Last 30 Minutes)


Kubernetes Cluster (Prometheus) - Grafana - Mozilla Firefox
Kubernetes Dashboard
Warning: Potential Security
Kubernetes Cluster (Pro
localhost:3000/d/4XuMd2Iiz/kubern
Kubernetes Cluster (Prometl
Last 30 minutes
Pods
Pods Running
15
Pods Pending
0
Pods Failed
0


Kubernetes Cluster (Prometheus) - Grafana - Mozilla Firefox
Kubernetes Dashboard
Warning: Potential Security
Kubernetes Cluster (Pro
localhost:3000/d/4XuMd2Iiz/kubern
Kubernetes Cluster (Prometl
Last 30 minutes
Node
Number Of Nodes
1
Nodes Out of Disk
N/A
Nodes Unavailable
0


Kubernetes Cluster (Prometheus) - Grafana - Mozilla Firefox
Kubernetes Dashboard
Warning: Potential Security
Kubernetes Cluster (Pro
localhost:3000/d/4XuMd2Iiz/kuberne
Kubernetes Cluster (Prometl
Last 30 minutes
Deployments
Deployment Replicas - Up To Date
Time
Metric
Value
2021-06-19 12:15:49
prometheus-operator
1
2021-06-19 12:15:49
prometheus-adapter
1
2021-06-19 12:15:49
nginx
1
Deployment Replicas
11


Kubernetes Cluster (Prometheus) - Grafana - Mozilla Firefox
Kubernetes Dashboard
Warning: Potential Security
Kubernetes Cluster (Pro
localhost:3000/d/4XuMd2Iiz/kuberne
Kubernetes Cluster (Prometl
Last 30 minutes
Deployment Replicas - Updated
11
Deployment Replicas - Unavailable
0

Deploying Prometheus and Grafana into the Minikube cluster using Helm charts [3]

Prometheus will help us monitor our Kubernetes Cluster and other resources running on it. Grafana will help us visualize metrics recorded by Prometheus and display them in fancy dashboards.

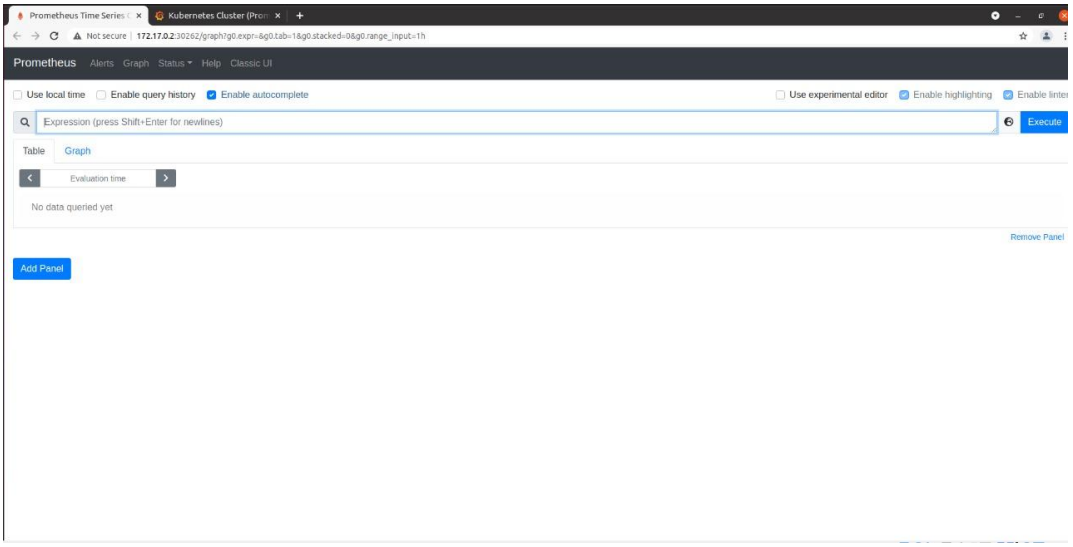
Installing Prometheus:

- Adding helm repo and installing charts

```
anusree@anusree-VirtualBox:~$ minikube start --vm-driver=docker
😄  minikube v1.7.2 on Ubuntu 20.04 (vbox/amd64)
✨  Using the docker (experimental) driver based on user configuration
🔥  Creating Kubernetes in docker container with (CPUs=2), Memory=2000MB (1987MB available) ...
🐳  Preparing Kubernetes v1.17.2 on Docker 19.03.2 ...
    ▪ kubeadm.pod-network-cidr=10.244.0.0/16
🚀  Launching Kubernetes ...
🌟  Enabling addons: default-storageclass, storage-provisioner
⌛  Waiting for cluster to come online ...
🏄  Done! kubectl is now configured to use "minikube"
⚠️  /usr/bin/kubectl is version 1.21.1, and is incompatible with Kubernetes 1.17.2. You will need to update /usr/bin/kubectl or use 'minikube kubectl' to connect with this cluster
anusree@anusree-VirtualBox:~$ helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
"prometheus-community" has been added to your repositories
anusree@anusree-VirtualBox:~$ helm install prometheus prometheus-community/prometheus
NAME: prometheus
LAST DEPLOYED: Fri Jun 18 11:06:38 2021
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
The Prometheus server can be accessed via port 80 on the following DNS name from within your cluster:
prometheus-server.default.svc.cluster.local


Get the Prometheus server URL by running these commands in the same shell:
  export POD_NAME=$(kubectl get pods --namespace default -l "app=prometheus,component=server" -o jsonpath="{.items[0].metadata.name}")
  kubectl --namespace default port-forward $POD_NAME 9090


The Prometheus alertmanager can be accessed via port 80 on the following DNS name from within your cluster:
prometheus-alertmanager.default.svc.cluster.local


Get the Alertmanager URL by running these commands in the same shell:
  export POD_NAME=$(kubectl get pods --namespace default -l "app=prometheus,component=alertmanager" -o jsonpath="{.items[0].metadata.name}")
  kubectl --namespace default port-forward $POD_NAME 9093
#################################################################################
######   WARNING: Pod Security Policy has been moved to a global property.  #####
######            use .Values.podSecurityPolicy.enabled with pod-based      #####
######            annotations                                               #####
######            (e.g. .Values.nodeExporter.podSecurityPolicy.annotations) #####
#################################################################################


The Prometheus PushGateway can be accessed via port 9091 on the following DNS name from within your cluster:
prometheus-pushgateway.default.svc.cluster.local
```

```
Get the Alertmanager URL by running these commands in the same shell:
  export POD_NAME=$(kubectl get pods --namespace default -l "app=prometheus,component=alertmanager" -o jsonpath="{.items[0].metadata.name}")
  kubectl --namespace default port-forward $POD_NAME 9093
#################################################################################
######   WARNING: Pod Security Policy has been moved to a global property.  #####
######            use .Values.podSecurityPolicy.enabled with pod-based      #####
######            annotations                                               #####
######            (e.g. .Values.nodeExporter.podSecurityPolicy.annotations) #####
#################################################################################


The Prometheus PushGateway can be accessed via port 9091 on the following DNS name from within your cluster:
prometheus-pushgateway.default.svc.cluster.local


Get the PushGateway URL by running these commands in the same shell:
  export POD_NAME=$(kubectl get pods --namespace default -l "app=prometheus,component=pushgateway" -o jsonpath="{.items[0].metadata.name}")
  kubectl --namespace default port-forward $POD_NAME 9091

For more information on running Prometheus, visit:
https://prometheus.io/
anusree@anusree-VirtualBox:~$ kubectl expose service prometheus-server --type=NodePort --target-port=9090 --name=prometheus-server-np
service/prometheus-server-np exposed
```

- Prometheus web interface



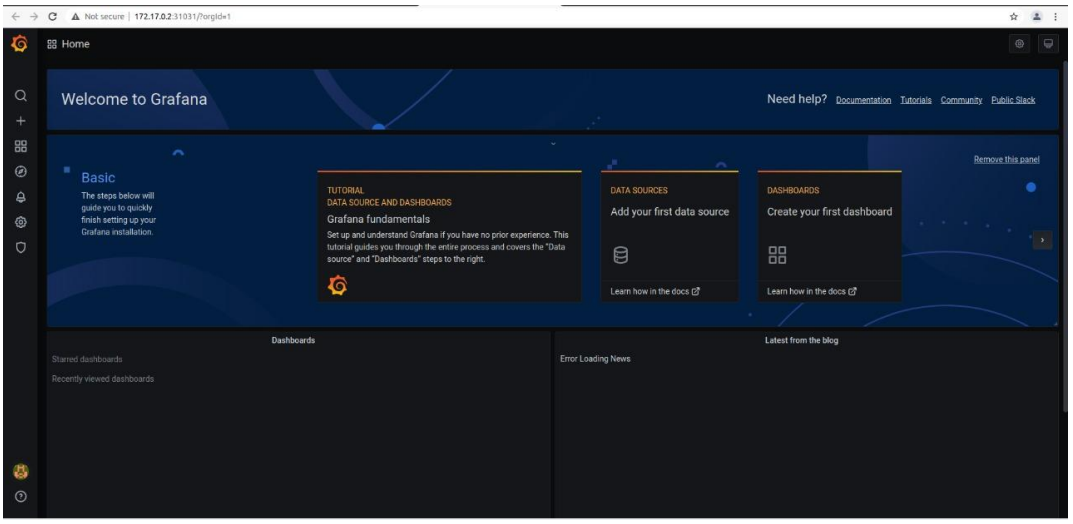
Installing Grafana:

- Adding helm repo and installing charts

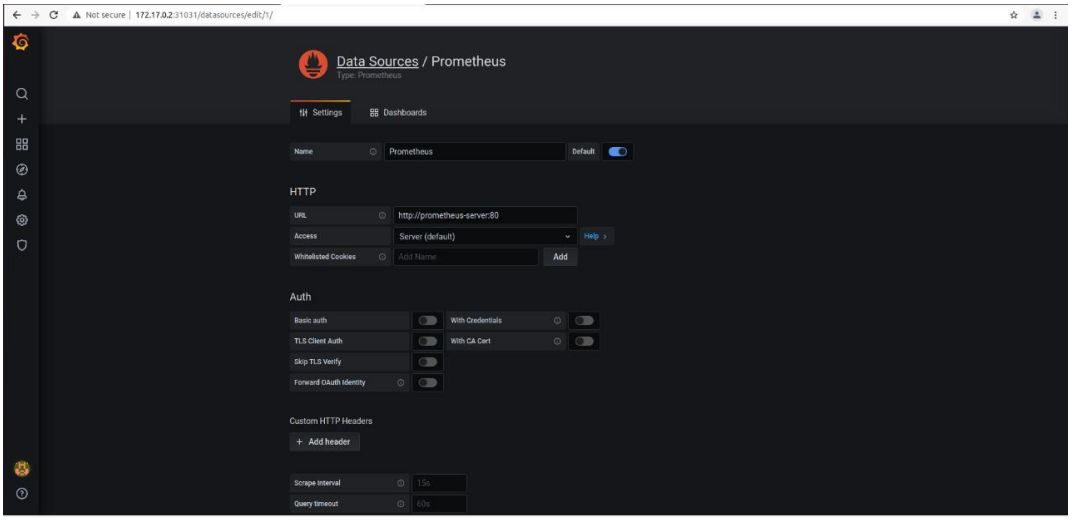
```
^Canusree@anusree-VirtualBox:~$ minikube service prometheus-server-np
|-----------|----------------------|-------------|------------------------|
| NAMESPACE |         NAME         | TARGET PORT |          URL           |
|-----------|----------------------|-------------|------------------------|
| default   | prometheus-server-np |             | http://172.17.0.2:32616 |
|-----------|----------------------|-------------|------------------------|
🎉  Opening service default/prometheus-server-np in default browser...
anusree@anusree-VirtualBox:~$
anusree@anusree-VirtualBox:~$ 1
^Canusree@anusree-VirtualBox:~$ helm repo add grafana https://grafana.github.io/helm-charts
"grafana" has been added to your repositories
anusree@anusree-VirtualBox:~$ helm install grafana stable/grafana
WARNING: This chart is deprecated
NAME: grafana
LAST DEPLOYED: Fri Jun 18 11:28:04 2021
NAMESPACE: default
STATUS: deployed
REVISION: 1
NOTES:
*******************
****DEPRECATED*****
*******************
* The chart is deprecated. Future development has been moved to https://github.com/grafana/helm2-grafana

1. Get your 'admin' user password by running:

   kubectl get secret --namespace default grafana -o jsonpath="{.data.admin-password}" | base64 --decode ; echo

2. The Grafana server can be accessed via port 80 on the following DNS name from within your cluster:

   grafana.default.svc.cluster.local

   Get the Grafana URL to visit by running these commands in the same shell:

     export POD_NAME=$(kubectl get pods --namespace default -l "app.kubernetes.io/name=grafana,app.kubernetes.io/instance=grafana" -o jsonpath="{.items[0].metadata.name}")
     kubectl --namespace default port-forward $POD_NAME 3000

3. Login with the password from step 1 and the username: admin
#################################################################################
######   WARNING: Persistence is disabled!!! You will lose your data when   #####
######            the Grafana pod is terminated.                             #####
#################################################################################
anusree@anusree-VirtualBox:~$ kubectl expose service grafana --type=NodePort --target-port=3000 --name=grafana-np
service/grafana-np exposed
anusree@anusree-VirtualBox:~$ kubectl get secret --namespace default grafana -o jsonpath="{.data.admin-password}" | base64 --decode ; echo
fzgqXrHcEYVrJAEC4vpvDIE8042fHEpVz1mUr66Z
```

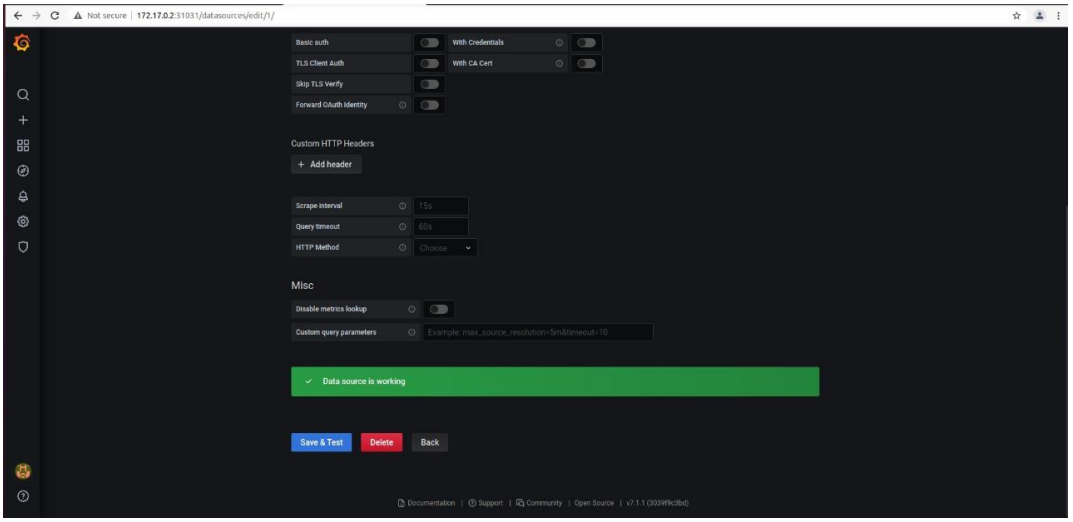
```
anusree@anusree-VirtualBox:~$ minikube service grafana-np
|-----------|------------|-------------|------------------------|
| NAMESPACE |    NAME    | TARGET PORT |          URL           |
|-----------|------------|-------------|------------------------|
| default   | grafana-np |             | http://172.17.0.2:31031 |
|-----------|------------|-------------|------------------------|
🎉  Opening service default/grafana-np in default browser...
```

- Grafana Dashboard



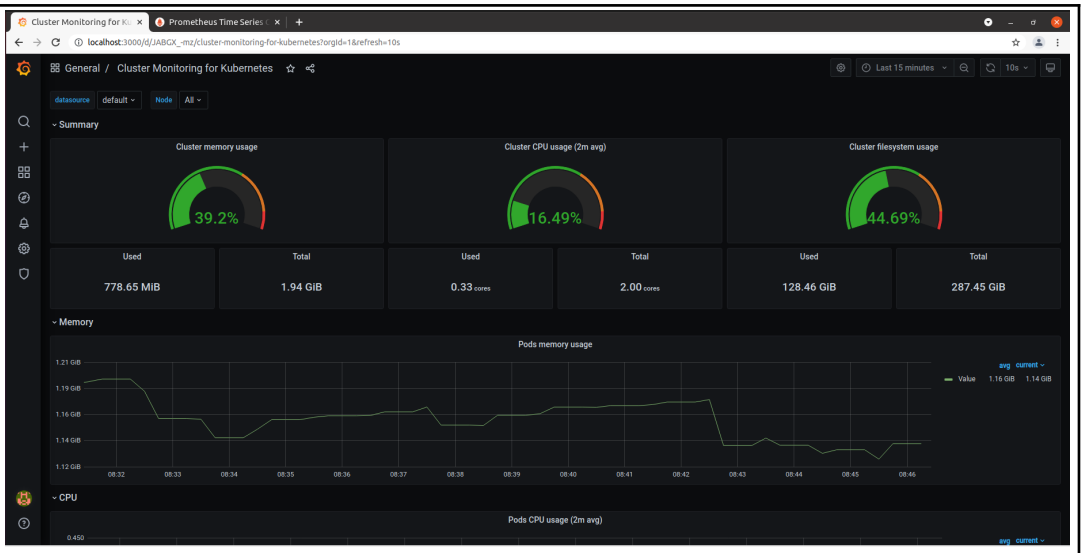
- Adding datasource (Prometheus):



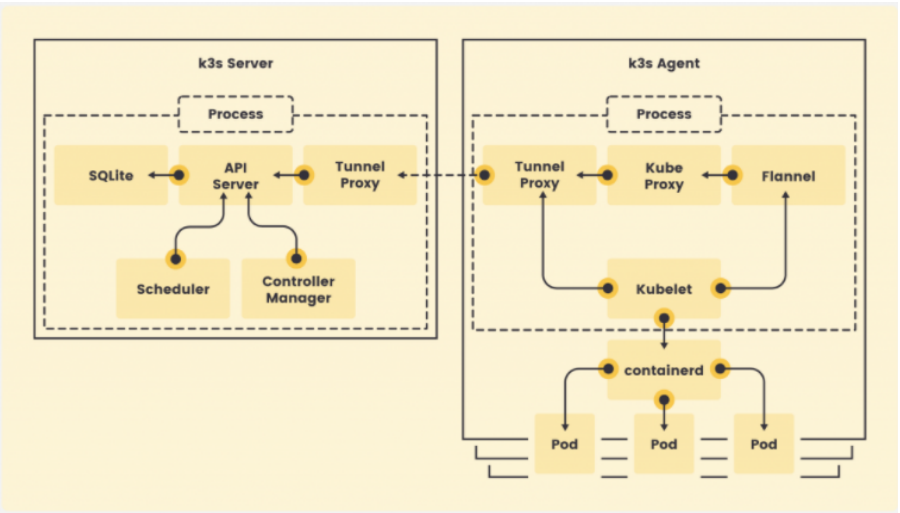


- Import dashboard from Grafana.com
  Create (+) → Import section to Import via grafana.com and set 10000 into the id field and click Load.

  In the dashboard configuration select the Prometheus Datasource created in the earlier step.


Cluster Monitoring for Ku
Prometheus Time Series C
localhost:3000/d/JABGX_-mz/cluster-monitoring-for-kubernetes?orgId=1&refresh=10s
General / Cluster Monitoring for Kubernetes
Last 15 minutes
10s
datasource
default
Node
All
Summary
Cluster memory usage
39.2%
Cluster CPU usage (2m avg)
16.49%
Cluster filesystem usage
44.69%
Used
778.65 MiB
Total
1.94 GiB
Used
0.33 cores
Total
2.00 cores
Used
128.46 GiB
Total
287.45 GiB
Memory
Pods memory usage
avg current
Value 1.16 GiB 1.14 GiB
CPU
Pods CPU usage (2m avg)


General / Cluster Monitoring for Kubernetes
Last 15 minutes
10s
CPU
Pods CPU usage (2m avg)
cores
avg current
Value 0.336 0.345
Network I/O
Network I/O pressure
Pods network I/O (2m avg)
avg current
-> 77.06 kB/s 55.76 kB/s
<- -44.63 kB/s -34.82 kB/s

What is K3s? [6]	It is lightweight Kubernetes developed by Rancher Labs and is suitable for creating development or staging clusters as it is easy to install and is half the memory size of K8s and is great for: Edge , IoT , CI Development,ARM,etc. Some of the features of K3s are: ● K3s is Packaged as a single binary along with all kubernetes control plane components allowing it to automate and manage complex cluster operations like distributing certificates. ● Provides Lightweight storage backend based on sqlite3 as the default storage mechanism. ● Simple but powerful batteries-included features such as: a local storage provider, a service load balancer, a Helm controller, and the Traefik ingress controller are provided.  In a k3s cluster, a node that runs the control plane components along with the kubelet is called a server, while a node that only runs the kubelet is called an agent. Both the server and agent have the container runtime and a kube proxy equivalent that manages the tunneling and network traffic across the cluster.

Installing K3s single node cluster

[7][8][9]

The Following shows the installation of single node k3s cluster using the script: curl -sfL https://get.k3s.io | sh -
As shown there are 7 pods running on the kube-system namespace by default.

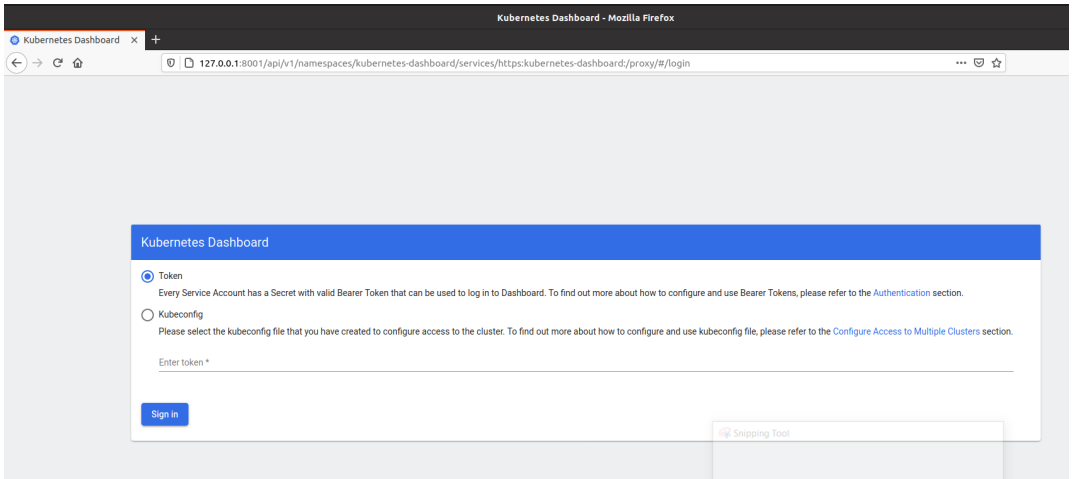
```
kube@kube1: ~
kube@kube1:~$ curl -sfL https://get.k3s.io | INSTALL_K3S_EXEC="--write-kubeconfig-mode 644" sh -
[INFO]  Finding release for channel stable
[INFO]  Using v1.21.1+k3s1 as release
[INFO]  Downloading hash https://github.com/k3s-io/k3s/releases/download/v1.21.1+k3s1/sha256sum-amd64.txt
[INFO]  Downloading binary https://github.com/k3s-io/k3s/releases/download/v1.21.1+k3s1/k3s
[INFO]  Verifying binary download
[INFO]  Installing k3s to /usr/local/bin/k3s
[INFO]  Creating /usr/local/bin/kubectl symlink to k3s
[INFO]  Creating /usr/local/bin/crictl symlink to k3s
[INFO]  Creating /usr/local/bin/ctr symlink to k3s
[INFO]  Creating killall script /usr/local/bin/k3s-killall.sh
[INFO]  Creating uninstall script /usr/local/bin/k3s-uninstall.sh
[INFO]  env: Creating environment file /etc/systemd/system/k3s.service.env
[INFO]  systemd: Creating service file /etc/systemd/system/k3s.service
[INFO]  systemd: Enabling k3s unit
Created symlink /etc/systemd/system/multi-user.target.wants/k3s.service → /etc/systemd/system/k3s.service.
[INFO]  systemd: Starting k3s
kube@kube1:~$ kubectl version
Client Version: version.Info{Major:"1", Minor:"21", GitVersion:"v1.21.1+k3s1", GitCommit:"75dba57f9b1de3ec0403b148c52c
348e1dee2a5e", GitTreeState:"clean", BuildDate:"2021-05-21T16:12:29Z", GoVersion:"go1.16.4", Compiler:"gc", Platform:"
linux/amd64"}
Server Version: version.Info{Major:"1", Minor:"21", GitVersion:"v1.21.1+k3s1", GitCommit:"75dba57f9b1de3ec0403b148c52c
348e1dee2a5e", GitTreeState:"clean", BuildDate:"2021-05-21T16:12:29Z", GoVersion:"go1.16.4", Compiler:"gc", Platform:"
linux/amd64"}
kube@kube1:~$ k3s --version
k3s version v1.21.1+k3s1 (75dba57f)
go version go1.16.4
kube@kube1:~$ kubectl get node
NAME    STATUS   ROLES                  AGE   VERSION
kube1   Ready    control-plane,master   20s   v1.21.1+k3s1
kube@kube1:~$ kubectl get pods --all-namespaces
NAMESPACE     NAME                                      READY   STATUS      RESTARTS   AGE
kube-system   metrics-server-86cbb8457f-nfcrc           1/1     Running     0          6m32s
kube-system   local-path-provisioner-5ff76fc89d-72kqq   1/1     Running     0          6m32s
kube-system   coredns-7448499f4d-jgph2                  1/1     Running     0          6m32s
kube-system   helm-install-traefik-crd-nxphh            0/1     Completed   0          6m32s
kube-system   helm-install-traefik-xmvt4                0/1     Completed   1          6m32s
kube-system   svclb-traefik-fs8xr                       2/2     Running     0          4m51s
kube-system   traefik-97b44b794-v8lq8                   1/1     Running     0          4m52s
kube@kube1:~$
```

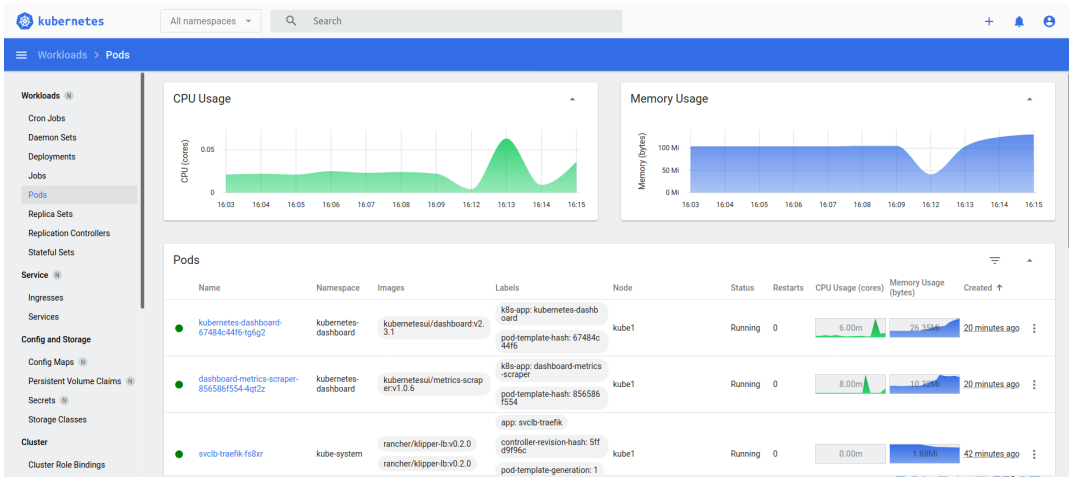
Deploying kubernetes dashboard:

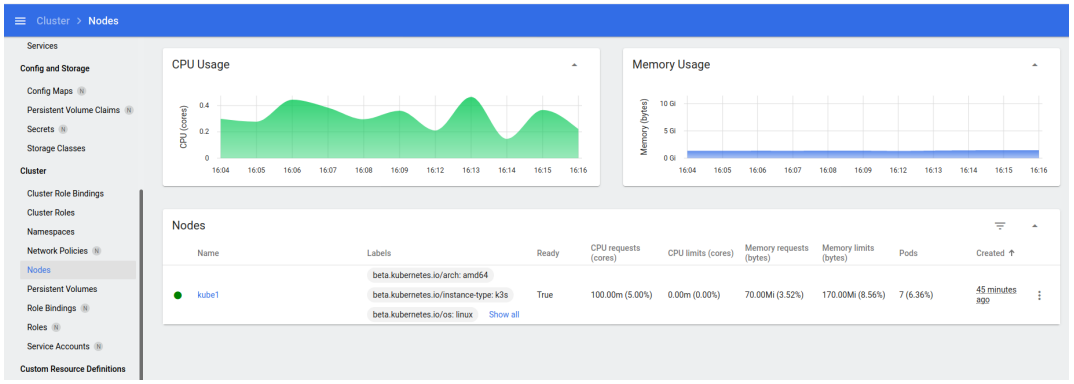
```
kube@kube1:~$ GITHUB_URL=https://github.com/kubernetes/dashboard/releases
kube@kube1:~$ VERSION_KUBE_DASHBOARD=$(curl -w '%{url_effective}' -I -L -s -S ${GITHUB_URL}/latest -o /dev/null |
 sed -e 's|.*/||')
kube@kube1:~$  kubectl create -f https://raw.githubusercontent.com/kubernetes/dashboard/${VERSION_KUBE_DASHBOARD}
/aio/deploy/recommended.yaml
namespace/kubernetes-dashboard created
serviceaccount/kubernetes-dashboard created
service/kubernetes-dashboard created
secret/kubernetes-dashboard-certs created
secret/kubernetes-dashboard-csrf created
secret/kubernetes-dashboard-key-holder created
configmap/kubernetes-dashboard-settings created
role.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrole.rbac.authorization.k8s.io/kubernetes-dashboard created
rolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
deployment.apps/kubernetes-dashboard created
service/dashboard-metrics-scraper created
deployment.apps/dashboard-metrics-scraper created
kube@kube1:~$ kubectl proxy
Starting to serve on 127.0.0.1:8001
^C
kube@kube1:~$ cat > dashboard.admin-user.yml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: admin-user
  namespace: kubernetes-dashboard
^C
kube@kube1:~$ cat > dashboard.admin-user-role.yml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: admin-user
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- kind: ServiceAccount
  name: admin-user
  namespace: kubernetes-dashboard
^C
kube@kube1:~$ kubectl create -f dashboard.admin-user.yml -f dashboard.admin-user-role.yml
```

```
kube@kube1:~$ kubectl create -f dashboard.admin-user.yml -f dashboard.admin-user-role.yml
serviceaccount/admin-user created
clusterrolebinding.rbac.authorization.k8s.io/admin-user created
kube@kube1:~$ kubectl -n kubernetes-dashboard describe secret admin-user-token | grep '^token'
token:      eyJhbGciOiJSUzI1NiIsImtpZCI6ImYtdTBzOVBJS2JPTHpUaUE3aVU1T2xSZHMzRW5ET1NpM2oydS1zRlJJV00ifQ.eyJpc3MiOi
JrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJrdWJlcm5ldGVzLWRhc2h
ib2FyZCIsImt1YmVybmV0ZXMuaW8vc2VydmljZWFjY291bnQvc2VjcmV0Lm5hbWUiOiJhZG1pbi11c2VyLXRva2VuLWJ3amg2Iiwia3ViZXJuZXRl
cy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQubmFtZSI6ImFkbWluLXVzZXIiLCJrdWJlcm5ldGVzLmlvL3NlcnZpY2VhY2NvdW50L
3NlcnZpY2UtYWNjb3VudC51aWQiOiIyY2I0ZjE1Mi1hMWY0LTQ2MWQtODM3Ny1kZWI4ODc4NmZkNmYiLCJzdWIiOiJzeXN0ZW06c2VydmljZWFjY2
91bnQ6a3ViZXJuZXRlcy1kYXNoYm9hcmQ6YWRtaW4tdXNlciJ9.UeS8rgm2D1ZBXKddSkvUG5lAgmmKhC1pFEsfmhtv3Rqtx3ZJAOeuG1Ur5LzJWt
5uwE1n4rBN11NIegXnUACBkjg2aONssLOSqKkCsUpdvVB1I2fck8HfW2jUZ7wjEdS83j8fDE9aSaBssWPPD5lVQaApb_amfkirjxI7evXu322JvCr
p__uZk2fTDjGIwVWkxOMmQHbdcBHe_Ec0OZfw5P5yx0m9H6oHPWOC9hrP-Tyr1JCr3QNWqfqbtInT-PspS6yVdFmW0bdMCTY1xxMCvptGHvNVJV_r
BnVXBBWy-Zh8DBkyEmFmc88sp0MPJDxWXnGfXsiZynYharqXZE9fAQ
kube@kube1:~$ kubectl proxy
Starting to serve on 127.0.0.1:8001
```

The kubernetes dashboard can be accessed on http://127.0.0.1:8001 after entering the valid token.







The above images show the analytics such as CPU and Memory Usage of Pods and Nodes in the k3s cluster.

Deploying Prometheus and Grafana into the K3s cluster using Helm charts[10]

After the installation of the latest helm version,The Bitnami helm chart was added to the helm repo.

```
kube@kube1:~$ helm repo list
NAME                    URL
prometheus-community    https://prometheus-community.github.io/helm-charts
stable                  https://charts.helm.sh/stable
kube@kube1:~$ helm repo add bitnami https://charts.bitnami.com/bitnami
"bitnami" has been added to your repositories
kube@kube1:~$ helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "prometheus-community" chart repository
...Successfully got an update from the "bitnami" chart repository
...Successfully got an update from the "stable" chart repository
Update Complete. ⎈Happy Helming!⎈
```

Installing Prometheus:

A namespace was created in the cluster using the command "$ kubectl create ns monitoring" and a kube-prometheus chart was installed in the namespace.

```
kube@kube1:~$ helm install prometheus bitnami/kube-prometheus -n monitoring
```

```
NAME: prometheus
LAST DEPLOYED: Sun Jun 20 17:00:54 2021
NAMESPACE: monitoring
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
** Please be patient while the chart is being deployed **

Watch the Prometheus Operator Deployment status using the command:

    kubectl get deploy -w --namespace monitoring -l app.kubernetes.io/name=kube-prometheus-operator,app.kubernetes.io/instance=prometheus

Watch the Prometheus StatefulSet status using the command:

    kubectl get sts -w --namespace monitoring -l app.kubernetes.io/name=kube-prometheus-prometheus,app.kubernetes.io/instance=prometheus

Prometheus can be accessed via port "9090" on the following DNS name from within your cluster:

    prometheus-kube-prometheus-prometheus.monitoring.svc.cluster.local

To access Prometheus from outside the cluster execute the following commands:

    echo "Prometheus URL: http://127.0.0.1:9090/"
    kubectl port-forward --namespace monitoring svc/prometheus-kube-prometheus-prometheus 9090:9090

Watch the Alertmanager StatefulSet status using the command:

    kubectl get sts -w --namespace monitoring -l app.kubernetes.io/name=kube-prometheus-alertmanager,app.kubernetes.io/instance=prometheus

Alertmanager can be accessed via port "9093" on the following DNS name from within your cluster:

    prometheus-kube-prometheus-alertmanager.monitoring.svc.cluster.local

To access Alertmanager from outside the cluster execute the following commands:

    echo "Alertmanager URL: http://127.0.0.1:9093/"
    kubectl port-forward --namespace monitoring svc/prometheus-kube-prometheus-alertmanager 9093:9093
```

The following new pods were created in the namespace:

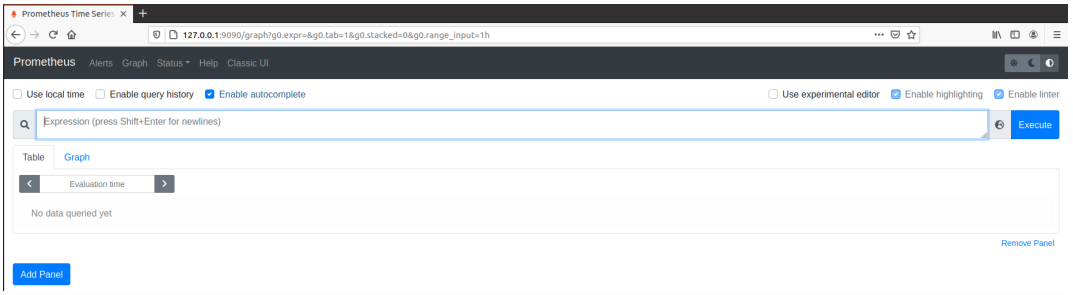
```
kube@kube1:~$ kubectl --namespace monitoring get pods
NAME                                                     READY   STATUS      RESTARTS   AGE
prometheus-prometheus-oper-admission-patch-zjwz8         0/1     Completed   1          52m
prometheus-kube-state-metrics-655d4894b6-vxgh9           1/1     Running     0          2m10s
prometheus-node-exporter-jtg7j                           1/1     Running     0          2m10s
prometheus-kube-prometheus-operator-6f786bb97c-pfcrz     1/1     Running     0          2m10s
alertmanager-prometheus-kube-prometheus-alertmanager-0   2/2     Running     0          83s
prometheus-prometheus-kube-prometheus-prometheus-0       2/2     Running     1          83s
kube@kube1:~$
```

After running the following command,prometheus time series monitoring

dashboard can be accessed on http://127.0.0.1:9090

```
kube@kube1:~$ kubectl port-forward --namespace monitoring svc/prometheus-kube-prometheus-prometheus 9090:9090
Forwarding from 127.0.0.1:9090 -> 9090
Forwarding from [::1]:9090 -> 9090
Handling connection for 9090
```

The web interface:



Installing Grafana:

```
kube@kube1:~$ helm install grafana bitnami/grafana --namespace monitoring
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /etc/rancher/k3s/k3s.yaml
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /etc/rancher/k3s/k3s.yaml
NAME: grafana
LAST DEPLOYED: Sun Jun 20 18:36:36 2021
NAMESPACE: monitoring
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
** Please be patient while the chart is being deployed **

1. Get the application URL by running these commands:
    echo "Browse to http://127.0.0.1:8080"
    kubectl port-forward svc/grafana 8080:3000 &

2. Get the admin credentials:

    echo "User: admin"
    echo "Password: $(kubectl get secret grafana-admin --namespace monitoring -o jsonpath="{.data.GF_SECURITY_ADMIN_PASSWORD}"
 | base64 --decode)"
kube@kube1:~$
```

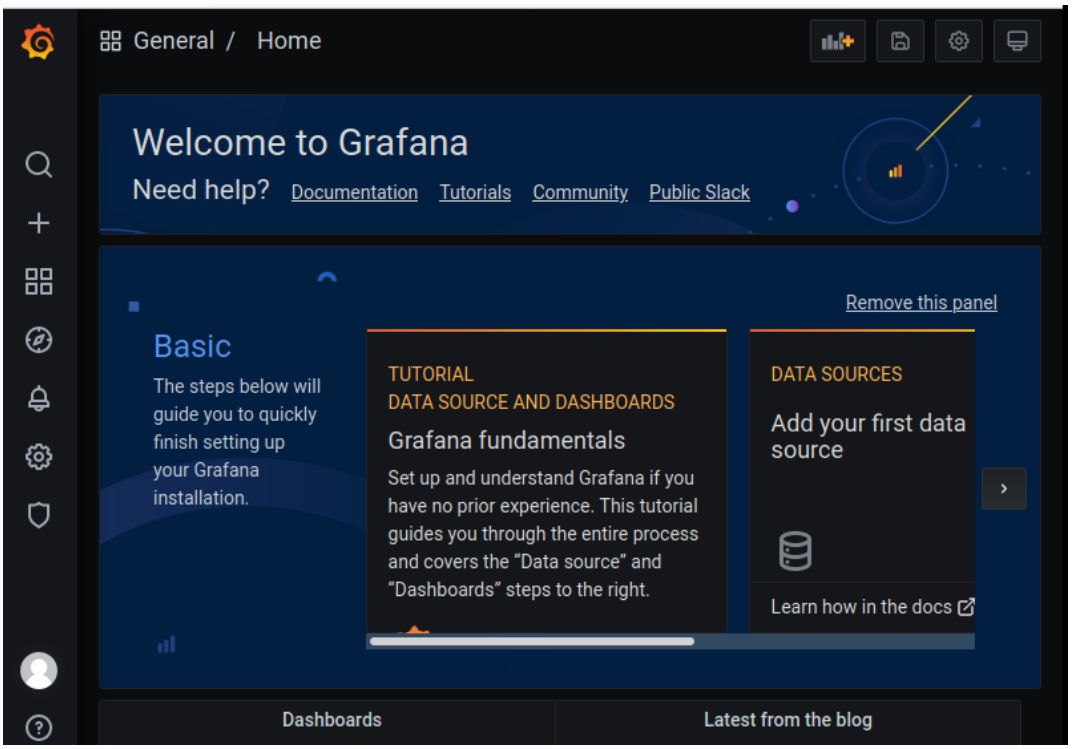
An extra grafana pod was deployed in the namespace of the cluster:

```
kube@kube1:~$ kubectl --namespace monitoring get pods
NAME                                                     READY   STATUS      RESTARTS   AGE
prometheus-prometheus-oper-admission-patch-zjwz8         0/1     Completed   1          159m
prometheus-kube-state-metrics-655d4894b6-vxgh9           1/1     Running     10         108m
prometheus-node-exporter-jtg7j                           1/1     Running     4          108m
prometheus-kube-prometheus-operator-6f786bb97c-pfcrz     1/1     Running     4          108m
alertmanager-prometheus-kube-prometheus-alertmanager-0   2/2     Running     2          107m
prometheus-prometheus-kube-prometheus-prometheus-0       2/2     Running     7          107m
grafana-5c64b4bff9-zsjrk                                 1/1     Running     0          12m
kube@kube1:~$
```

Running the following command will make grafana dashboard accessible on http://127.0.0.1/8080

```
kube@kube1:~$ kubectl port-forward svc/grafana 8080:3000 --namespace monitoring
Forwarding from 127.0.0.1:8080 -> 3000
Forwarding from [::1]:8080 -> 3000
```

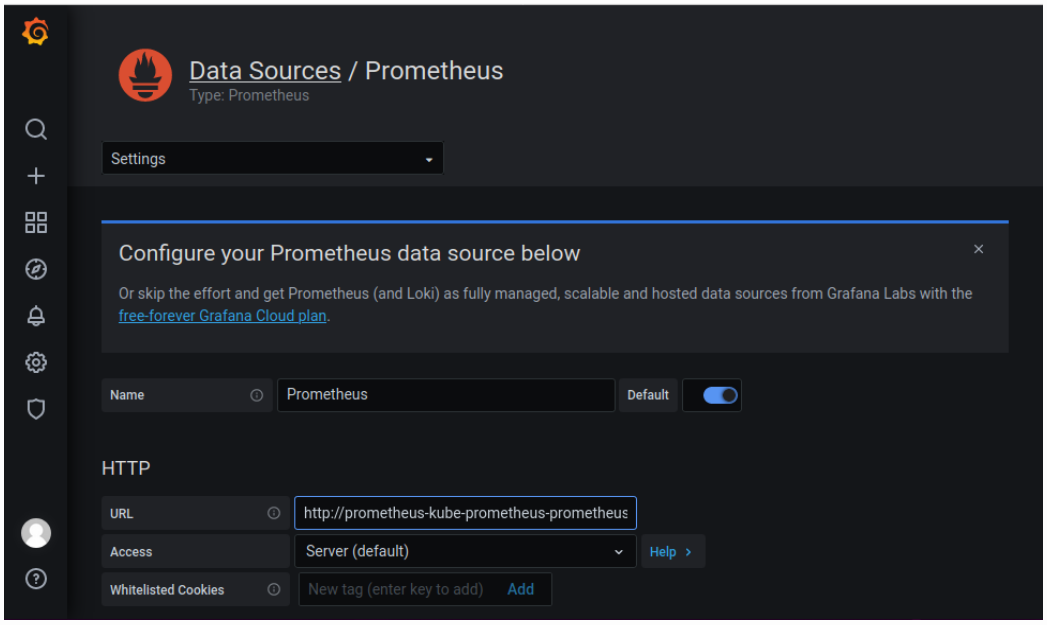
After entering the valid username and password the following home page opens.

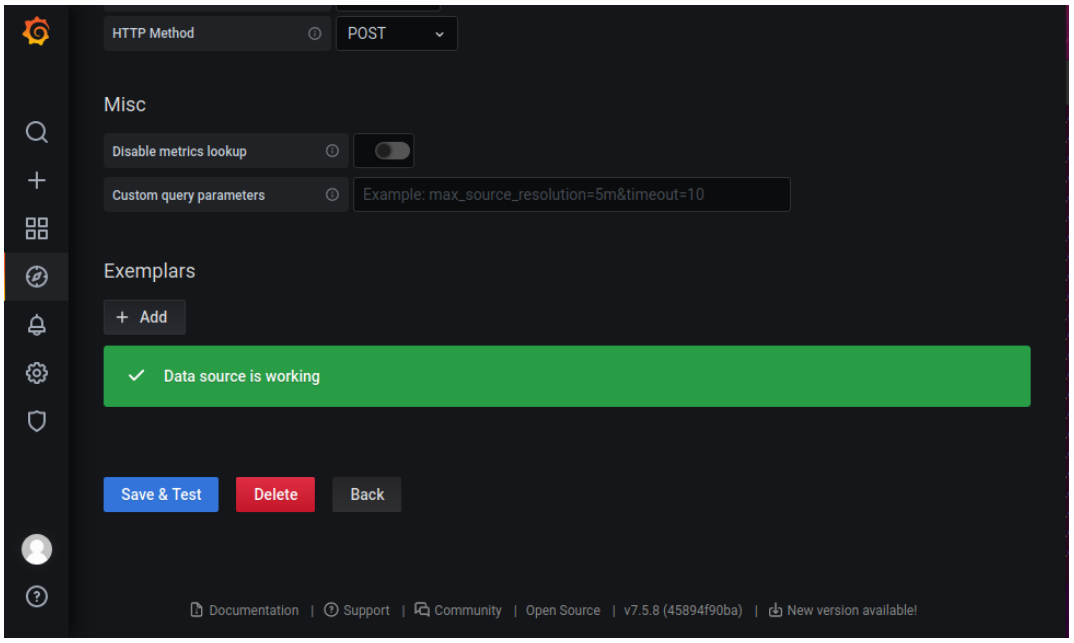


Clicking on the gear icon in the side menu of the dashboard->select data sources->Add data source -> select "Prometheus" from the list of data sources.

In the configuration screen for the new data source to be created, since both Grafana and Prometheus are running on the same cluster, they can be connected using the internal DNS to Kubernetes by providing the service name that Prometheus is connected to. Hence in the URL field this link can be typed: http://prometheus-kube-prometheus-prometheus.monitoring.svc.cluster.local:9090

Later clicking on save and test will get the data source running which can be used in the dashboard.

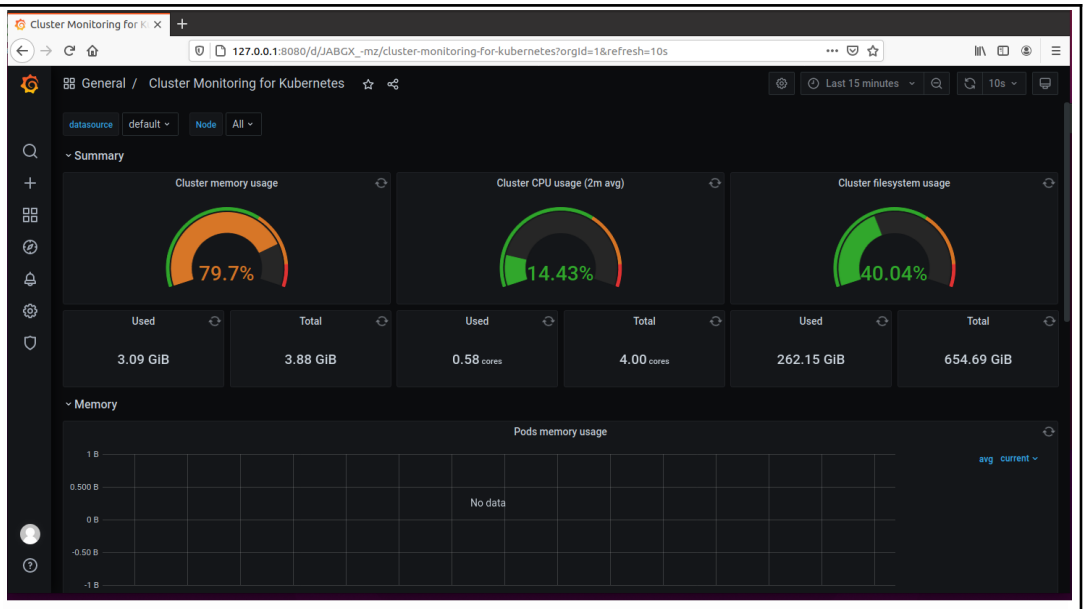




To import a dashboard using URL ID or JSON,mouse over the “Dashboards” section on the left-hand side of the Grafana screen (the icon is four squares) and choose “Manage.” On the top right of the dashboard management screen, click “Import”.

Under “Import via grafana.com”,enter “10000,” matching the ID of the existing dashboard.

Finally after choosing the newly created Prometheus data source and clicking on “Import”,the following dashboard was displayed.



References

1. https://www.server-world.info/en/note?os=Ubuntu_20.04&p=microk8s&f=8
2. https://microk8s.io/docs
3. https://blog.marcnuri.com/prometheus-grafana-setup-minikube/
4. https://webme.ie/how-to-run-minikube-on-a-virtualbox-vm/
5. https://grafana.com/grafana/dashboards/6417
6. https://thenewstack.io/how-rancher-labs-k3s-makes-it-easy-to-run-kubernetes-at-the-edge/
7. https://www.youtube.com/watch?v=2LNxGVS81mE
8. https://rancher.com/docs/k3s/latest/en/installation/kube-dashboard/
9. https://rancher.com/docs/k3s/latest/en/quick-start/
10. https://tanzu.vmware.com/developer/guides/kubernetes/observability-prometheus-grafana-p1/