

## Create a Namespace & ClusterRole

First we need to create namespace for our monitoring using the following command, after we start the minikube

```
minikube start
```

create new namespace using

```
kubectl create namespace monitoring
```

To start the dashboard for kubernetes cluster run **minikube dashboard** and navigate to **Cluster / Namespaces** and we can see that the namespace that we create.

The screenshot shows the Kubernetes dashboard interface. On the left is a sidebar menu with categories: Service, Config and Storage, Cluster, and Settings. The 'Namespaces' option under the 'Cluster' category is selected and highlighted in blue. The main content area displays details for the 'monitoring' namespace. It includes a 'Metadata' section with a table showing the namespace's name, creation time (May 4, 2023), age (55 minutes ago), and UID. Below this is a 'Resource information' section showing the status as 'Active'. There are also sections for 'Resource Quotas' and 'Resource Limits', both of which display 'There is nothing to display here' with the note 'No resources found.' At the bottom, the 'Events' section shows 'Items: 0'.

Name	Created	Age	UID
monitoring	May 4, 2023	55 minutes ago	61bd398b-4be1-4db1-a9f3-7e0b017832f6

Labels: kubernetes.io/metadata.name: monitoring

Status: Active

Resource Quotas: There is nothing to display here. No resources found.

Resource Limits: There is nothing to display here. No resources found.

Events: Items: 0

PROF

To verify that namespace **monitoring** is created in command line type:

```
kubectl get namespaces
```

```
Terminal
ds@ds-HP-ProBook-440-G6:~$ kubectl get namespaces
NAME                STATUS    AGE
default             Active   18m
kube-node-lease     Active   18m
kube-public         Active   18m
kube-system         Active   18m
kubernetes-dashboard Active   16m
monitoring          Active   5m43s
ds@ds-HP-ProBook-440-G6:~$
```

Now lets create new file caled **clusterRole.yaml**

Note: In the role, given below, you can see that we have added ***get, list, and watch*** permissions to nodes, services endpoints, pods, and ingresses. The role binding is bound to the monitoring namespace. If you have any use case to retrieve metrics from any other object, you need to add that in this cluster role.

**clusterRole.yaml**

```
! clusterRole.yaml x
monitoring > ! clusterRole.yaml > [ ] subjects > {} 0 > namespace
io.k8s.api.rbac.v1.ClusterRoleBinding (v1@clusterrolebinding.json) | io.k8s.api.rbac.v1.ClusterRole (v1@clusterrole.json)
1  apiVersion: rbac.authorization.k8s.io/v1
2  kind: ClusterRole
3  metadata:
4    name: prometheus
5  rules:
6  - apiGroups: ["" ]
7    resources:
8      - nodes
9      - nodes/proxy
10     - services
11     - endpoints
12     - pods
13     verbs: ["get", "list", "watch"]
14  - apiGroups:
15     - extensions
16     resources:
17       - ingresses
18     verbs: ["get", "list", "watch"]
19  - nonResourceURLs: ["/metrics"]
20     verbs: ["get"]
21  ---
22  apiVersion: rbac.authorization.k8s.io/v1
23  kind: ClusterRoleBinding
24  metadata:
25    name: prometheus
26  roleRef:
27    apiGroup: rbac.authorization.k8s.io
28    kind: ClusterRole
29    name: prometheus
30  subjects:
31  - kind: ServiceAccount
32    name: default
33    namespace: monitoring
```

run the following command to create the role

PROF

```
kubectl create -f clusterRole.yaml
```

```
Terminal
ds@ds-HP-ProBook-440-G6:~/Documents/monitoring$ kubectl create -f clusterRole.yaml
clusterrole.rbac.authorization.k8s.io/prometheus created
clusterrolebinding.rbac.authorization.k8s.io/prometheus created
ds@ds-HP-ProBook-440-G6:~/Documents/monitoring$
```

## Create a Config Map To Externalize Prometheus Configurations

All configurations for Prometheus are part of prometheus.yaml file and all the alert rules for Alertmanager are configured in prometheus.rules.

***prometheus.yaml:*** This is the main Prometheus configuration which holds all the scrape configs, service discovery details, storage locations, data retention configs, etc)

***prometheus.rules:*** This file contains all the Prometheus alerting rules

Lets create new file config-map.yaml with the followin content [config-map.yaml](#)