OpenShift Admin Lab

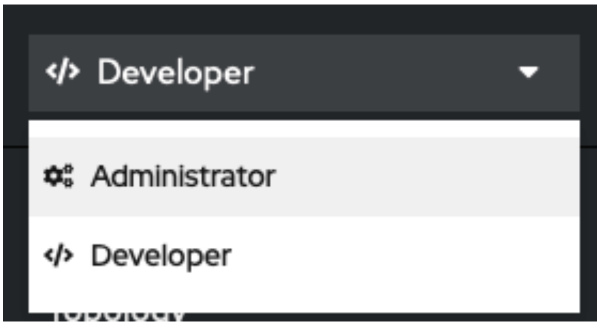
Duration: 30 minutes

Introduction

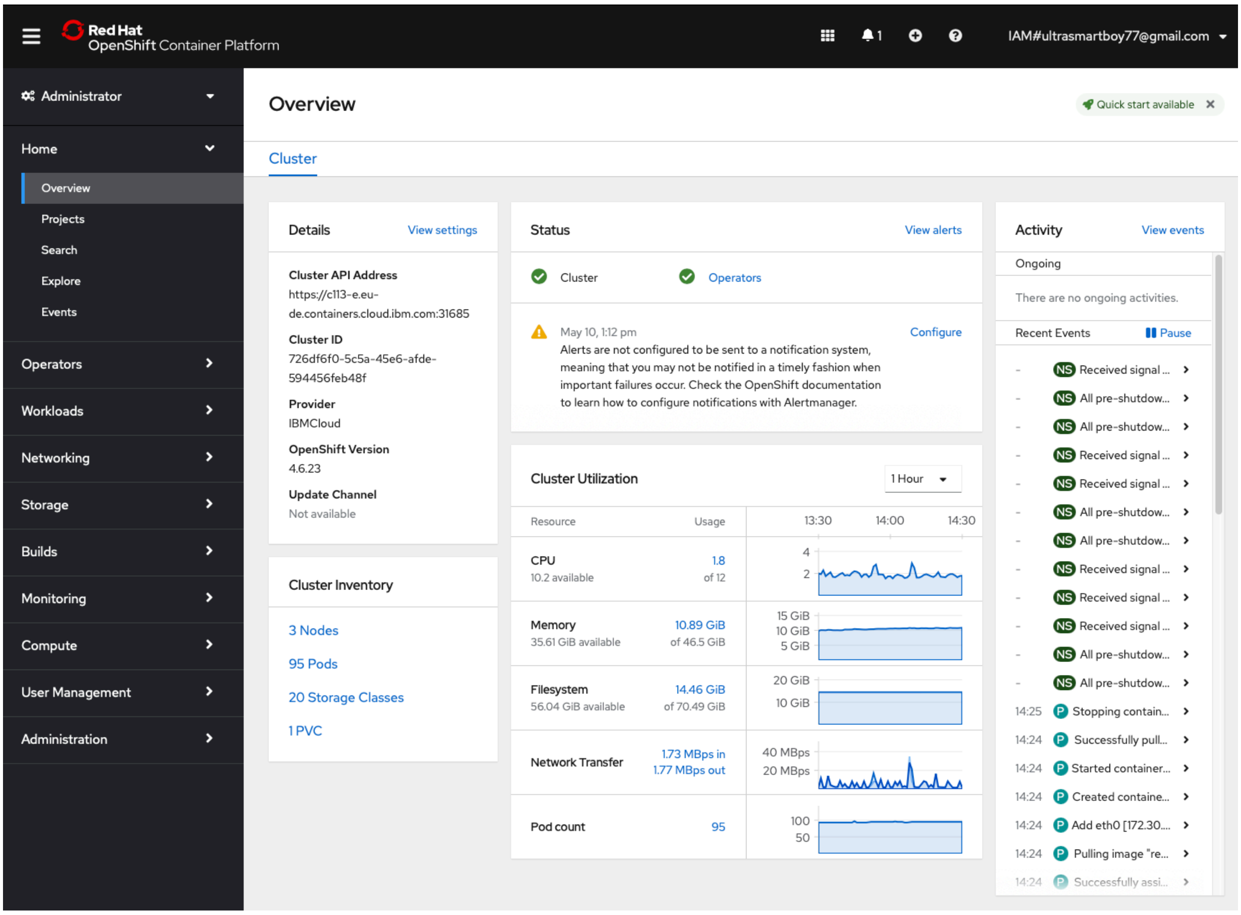
This lab is the continuation of the first OpenShift Lab. During this new lab, we are going to concentrate on the cluster administrator work like managing the pods, nodes, versions, CRDs, monitoring, scaling ...

Task #8 - OpenShift Administrator Console

If you are still on the **Developer** side, move to the **Administrator**.



You will a screen like this one:



Details

Cluster API Address
https://c113-e.eu-de.containers.cloud.ibm.com:31685

Cluster ID
726df6f0-5c5a-45e6-afde-594456feb48f

Provider
IBMCloud

OpenShift Version
4.6.23

Update Channel
Not available

Cluster Inventory

3 Nodes
95 Pods
20 Storage Classes
1 PVC

Status

✓ Cluster ✓ Operators

May 10, 1:12 pm
Alerts are not configured to be sent to a notification system, meaning that you may not be notified in a timely fashion when important failures occur. Check the OpenShift documentation to learn how to configure notifications with Alertmanager.

Cluster Utilization

Resource	Usage
CPU 10.2 available	1.8 of 12
Memory 35.61 GiB available	10.89 GiB of 46.5 GiB
Filesystem 56.04 GiB available	14.46 GiB of 70.49 GiB
Network Transfer	1.73 MBps in 1.77 MBps out
Pod count	95

Activity

Ongoing

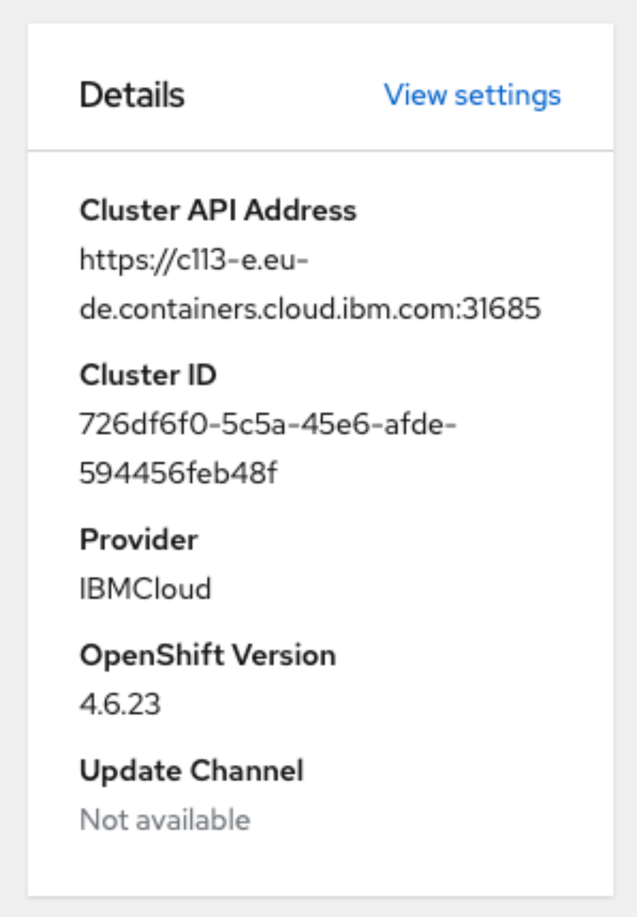
There are no ongoing activities.

Recent Events

-	NS	Received signal ...
-	NS	All pre-shutdow...
-	NS	All pre-shutdow...
-	NS	Received signal ...
-	NS	Received signal ...
-	NS	All pre-shutdow...
-	NS	All pre-shutdow...
-	NS	Received signal ...
-	NS	Received signal ...
-	NS	Received signal ...
-	NS	All pre-shutdow...
-	NS	All pre-shutdow...
14:25	P	Stopping contain...
14:24	P	Successfully pull...
14:24	P	Started container...
14:24	P	Created containe...
14:24	P	Add eth0 [172.30...
14:24	P	Pulling image "re...
14:24	P	Successfully assi...

The Dashboard (Overview) is very interesting and very popular for troubleshooting.

Notice the following cards : first the **Details** where you can check the OpenShift **version** and if you can update to a newer version.


Details

View settings

Cluster API Address

https://c113-e.eu-de.containers.cloud.ibm.com:31685

Cluster ID

726df6f0-5c5a-45e6-afde-594456feb48f

Provider

IBMCloud

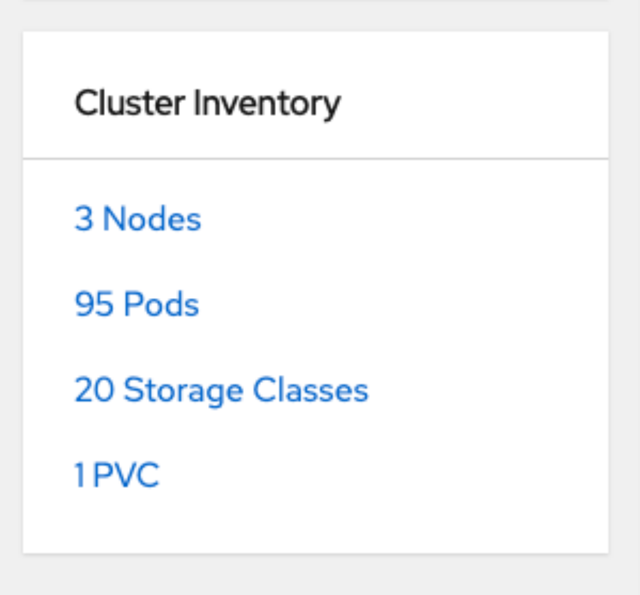
OpenShift Version

4.6.23

Update Channel

Not available

Then the **Cluster Inventory** : This is were you can check the different resources (like nodes or pods and storage). If you have a red point near Pods, then you will be quickly drill down to the **failling** or suspended pod(s).


Cluster Inventory

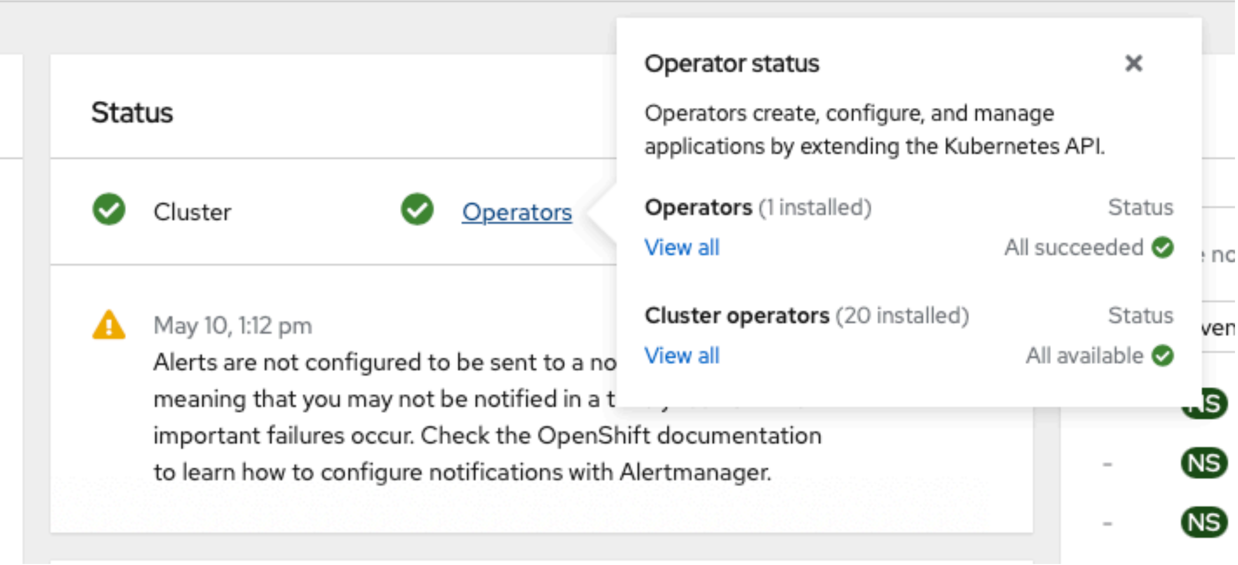
3 Nodes

95 Pods

20 Storage Classes

1 PVC

In the Status, you will see if the different components (like operators) in the cluster are healthy or not:



Status

✓ Cluster ✓ Operators

⚠ May 10, 1:12 pm
Alerts are not configured to be sent to a no
meaning that you may not be notified in a t
important failures occur. Check the OpenShift documentation
to learn how to configure notifications with Alertmanager.

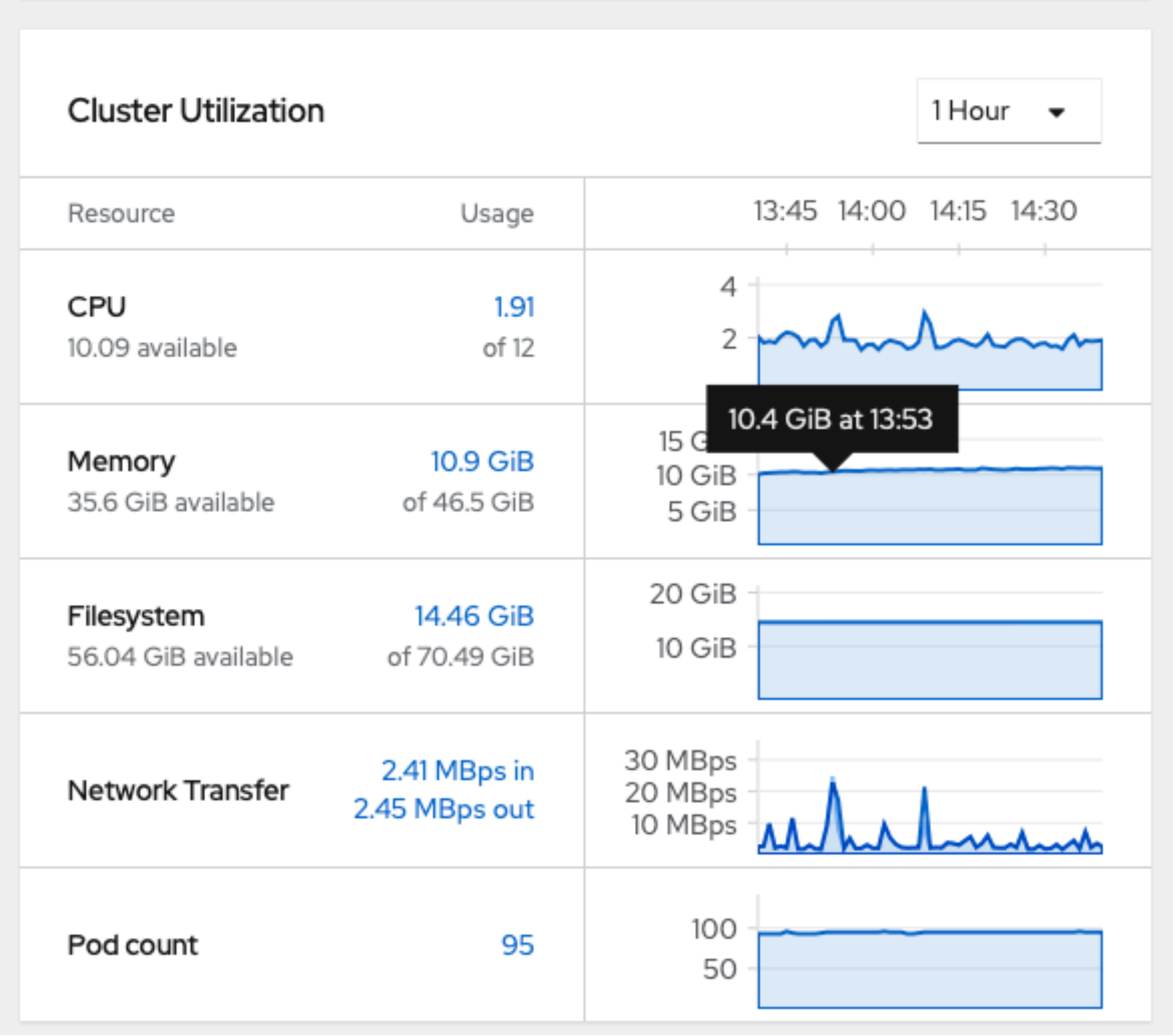
Operator status ×

Operators create, configure, and manage applications by extending the Kubernetes API.

Operators (1 installed)	Status
View all	All succeeded ✓

Cluster operators (20 installed)	Status
View all	All available ✓

The **Cluster Utilisation** is very important to see how your Pods are consumming the infrastructure resources like CPU, RAM, storage, bandwidth ...

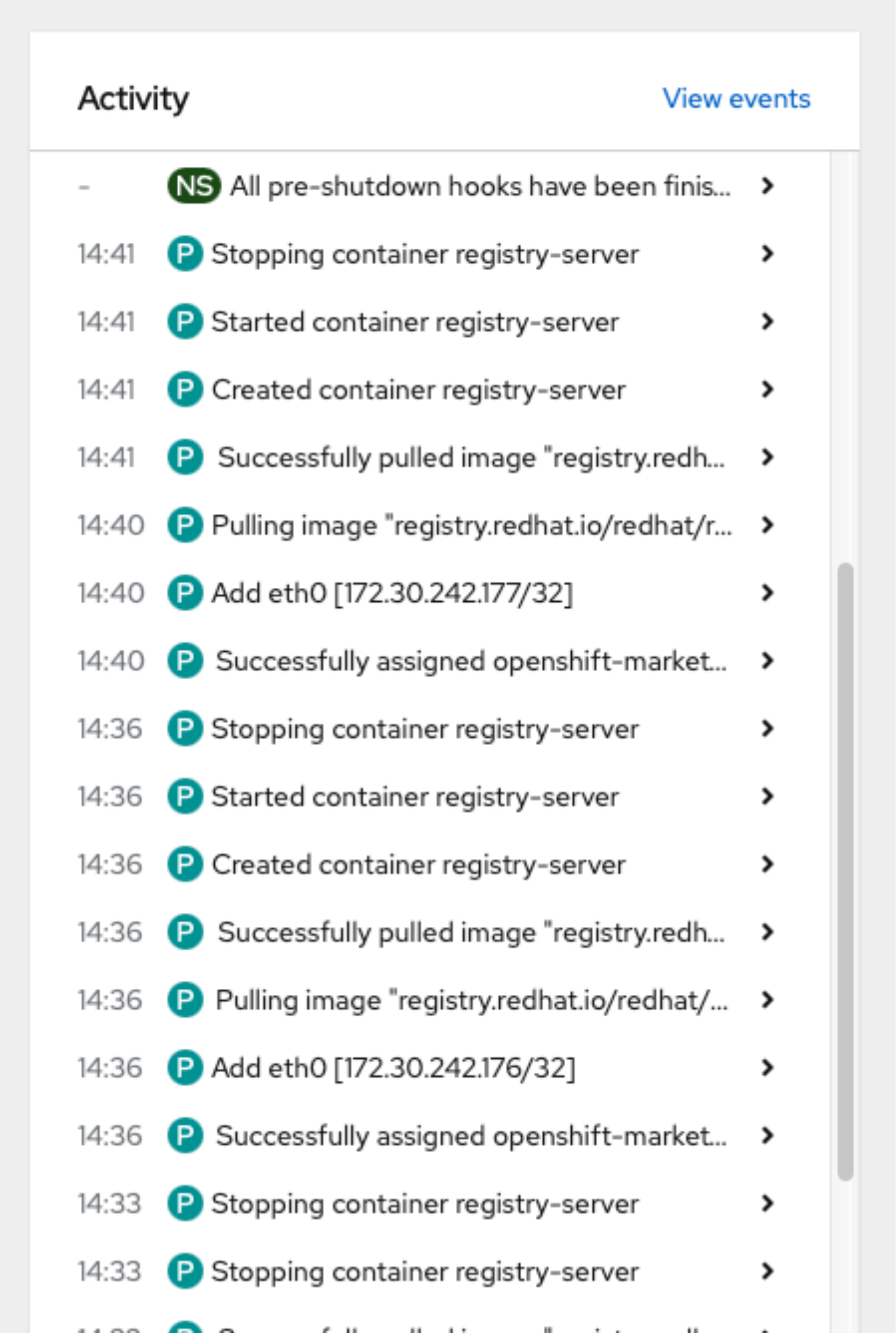


Cluster Utilization

1 Hour ▾

Resource	Usage
CPU 10.09 available	1.91 of 12
Memory 35.6 GiB available	10.9 GiB of 46.5 GiB
Filesystem 56.04 GiB available	14.46 GiB of 70.49 GiB
Network Transfer	2.41 MBps in 2.45 MBps out
Pod count	95

Finally, one most important componant in Kubernetes : the **event** messages ! All Kubernetes events are shown in one place and this very helpfull is you want to analyse and quickly identify issues, error or impacting problems.



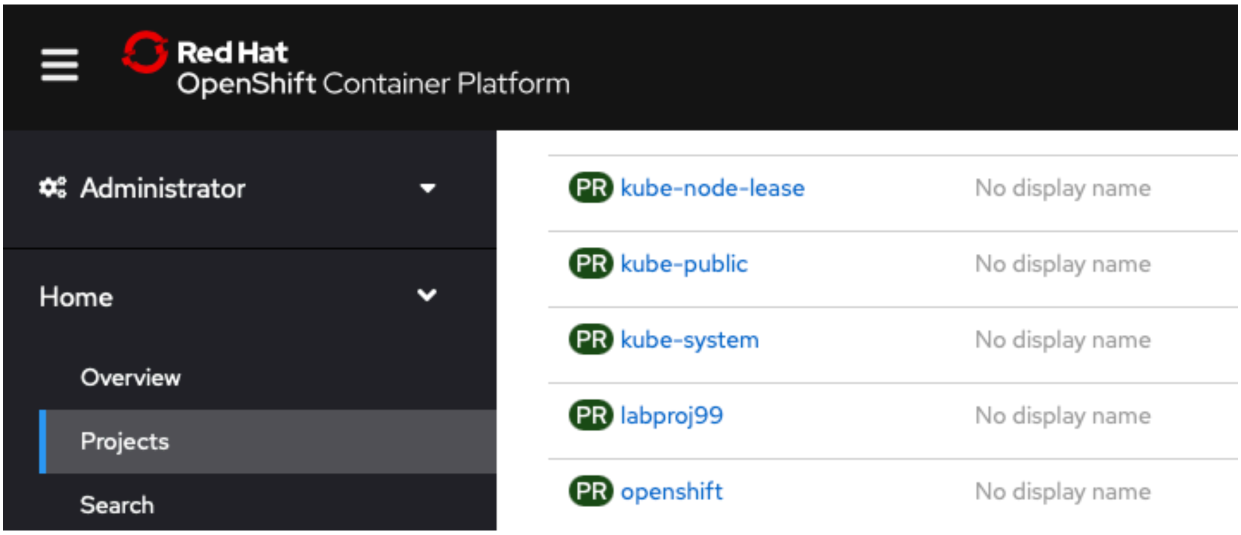
Activity View events

```
-     NS All pre-shutdown hooks have been finis...
14:41 P Stopping container registry-server
14:41 P Started container registry-server
14:41 P Created container registry-server
14:41 P Successfully pulled image "registry.redh...
14:40 P Pulling image "registry.redhat.io/redhat/r...
14:40 P Add eth0 [172.30.242.177/32]
14:40 P Successfully assigned openshift-market...
14:36 P Stopping container registry-server
14:36 P Started container registry-server
14:36 P Created container registry-server
14:36 P Successfully pulled image "registry.redh...
14:36 P Pulling image "registry.redhat.io/redhat/...
14:36 P Add eth0 [172.30.242.176/32]
14:36 P Successfully assigned openshift-market...
14:33 P Stopping container registry-server
14:33 P Stopping container registry-server
```

So to summarize, in one dashboard overview, you can quickly analyze the situation and find a solution for a specific issue.

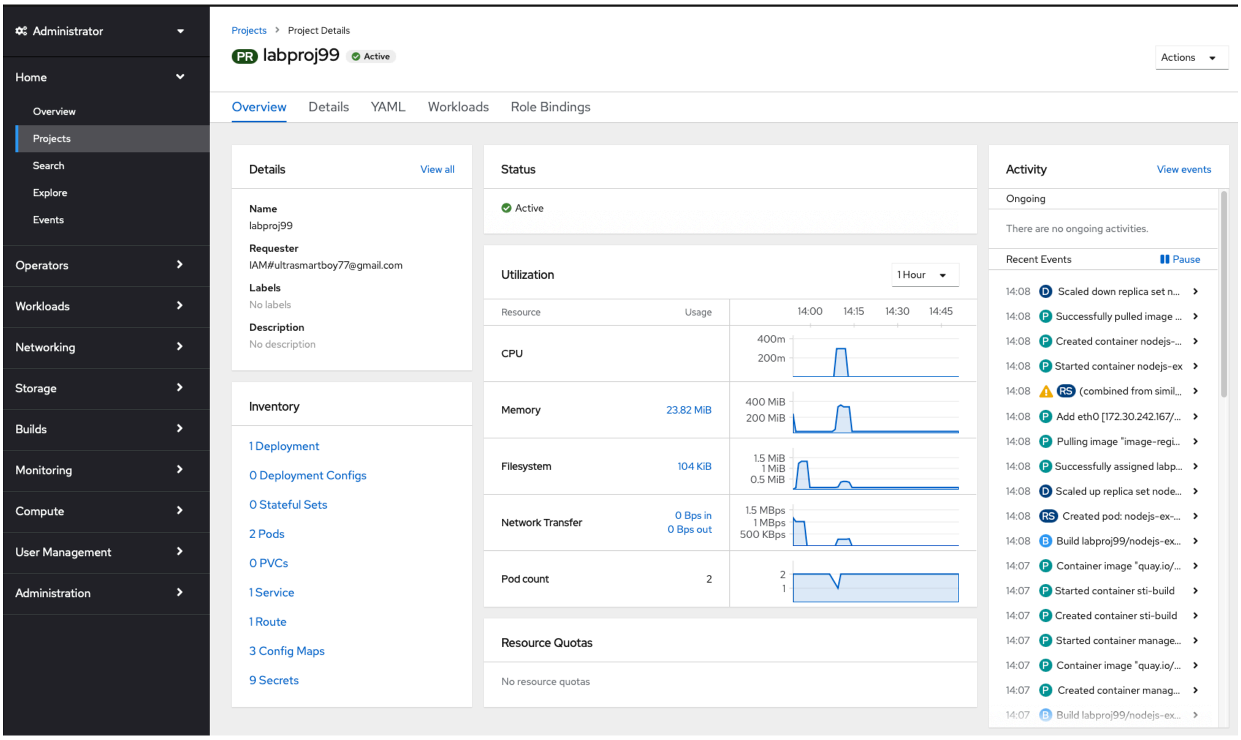
Task #9 - Projects and Workloads

Now move to the **projects** : as the cluster administrator, you can view all the projects (namespaces) in the cluster (normally the developers just see their own namespace). Click on you **labproj**



PR kube-node-lease	No display name
PR kube-public	No display name
PR kube-system	No display name
PR labproj99	No display name
PR openshift	No display name

This brings you to this overview but specifically for this **project** (labprojxx).



PR labproj99 Active

Details

Name
labproj99

Requester
IAM#ultrasmartboy77@gmail.com

Labels
No labels

Description
No description

Inventory

1 Deployment
0 Deployment Configs
0 Stateful Sets
2 Pods
0 PVCs
1 Service
1 Route
3 Config Maps
9 Secrets

Status

Active

Utilization

Resource	Usage
CPU	
Memory	23.82 MiB
Filesystem	104 KiB
Network Transfer	0 Bps in 0 Bps out
Pod count	2

Resource Quotas

No resource quotas

Activity

Ongoing

There are no ongoing activities.

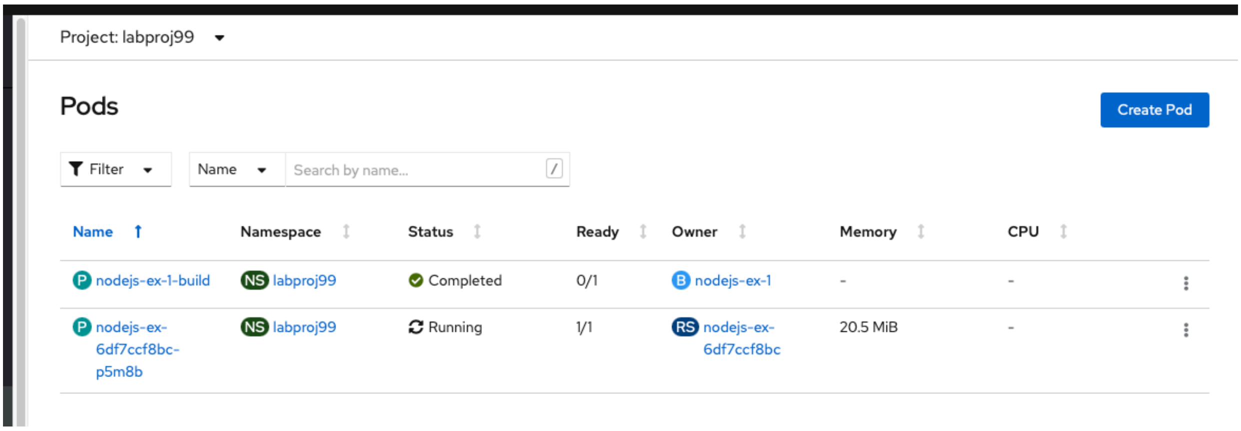
Recent Events

```
14:08  D  Scaled down replica set n...
14:08  P  Successfully pulled image ...
14:08  P  Created container nodejs-...
14:08  P  Started container nodejs-ex
14:08  ⚠ RS (combined from simil...
14:08  P  Add eth0 [172.30.242.167/...
14:08  P  Pulling image "image-regi...
14:08  P  Successfully assigned labp...
14:08  D  Scaled up replica set node...
14:08  RS Created pod: nodejs-ex-...
14:08  B  Build labproj99/nodejs-ex...
14:07  P  Container image "quay.io/...
14:07  P  Started container sti-build
14:07  P  Created container sti-build
14:07  P  Started container manage...
14:07  P  Container image "quay.io/...
14:07  P  Created container manag...
14:07  B  Build labproj99/nodejs-ex...
```

The Dashboard of your labproj is very interesting because you can see a lot of informations:

- Inventory of OpenShift objects like Pods, Deployments, Services ...
- Status (if some errors) for the project
- Utilization (CPU, Memory ...) for the project
- Activity (from the events) for the resources in that project
- Quotas

Now drill down to the Pods to look at the **Pods** in Inventory in your **labproj**:



Project: labproj99

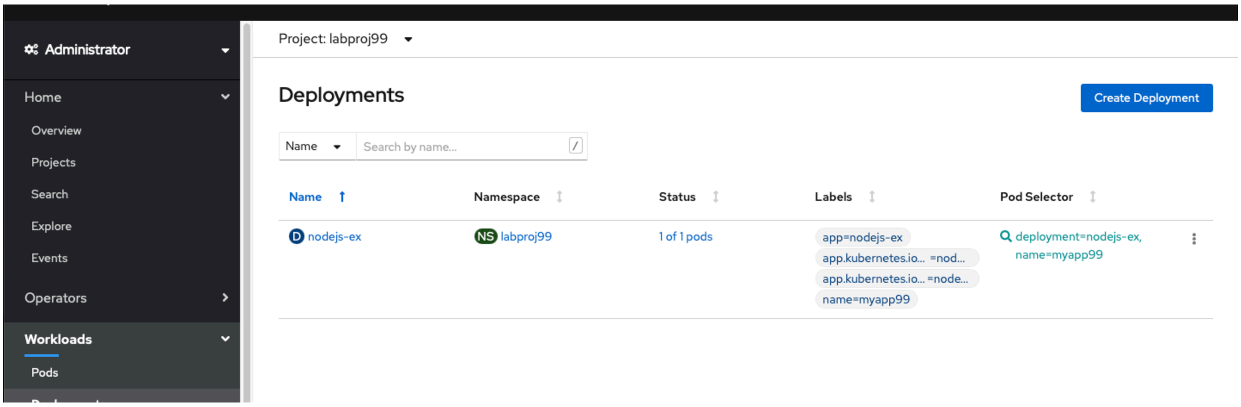
Pods

Create Pod

Filter | Name | Search by name...

Name	Namespace	Status	Ready	Owner	Memory	CPU
P nodejs-ex-1-build	NS labproj99	✓ Completed	0/1	B nodejs-ex-1	-	-
P nodejs-ex-6df7ccf8bc-p5m8b	NS labproj99	Running	1/1	RS nodejs-ex-6df7ccf8bc	20.5 MiB	-

On the left pane, click on the **Deployment**



Project: labproj99

Deployments

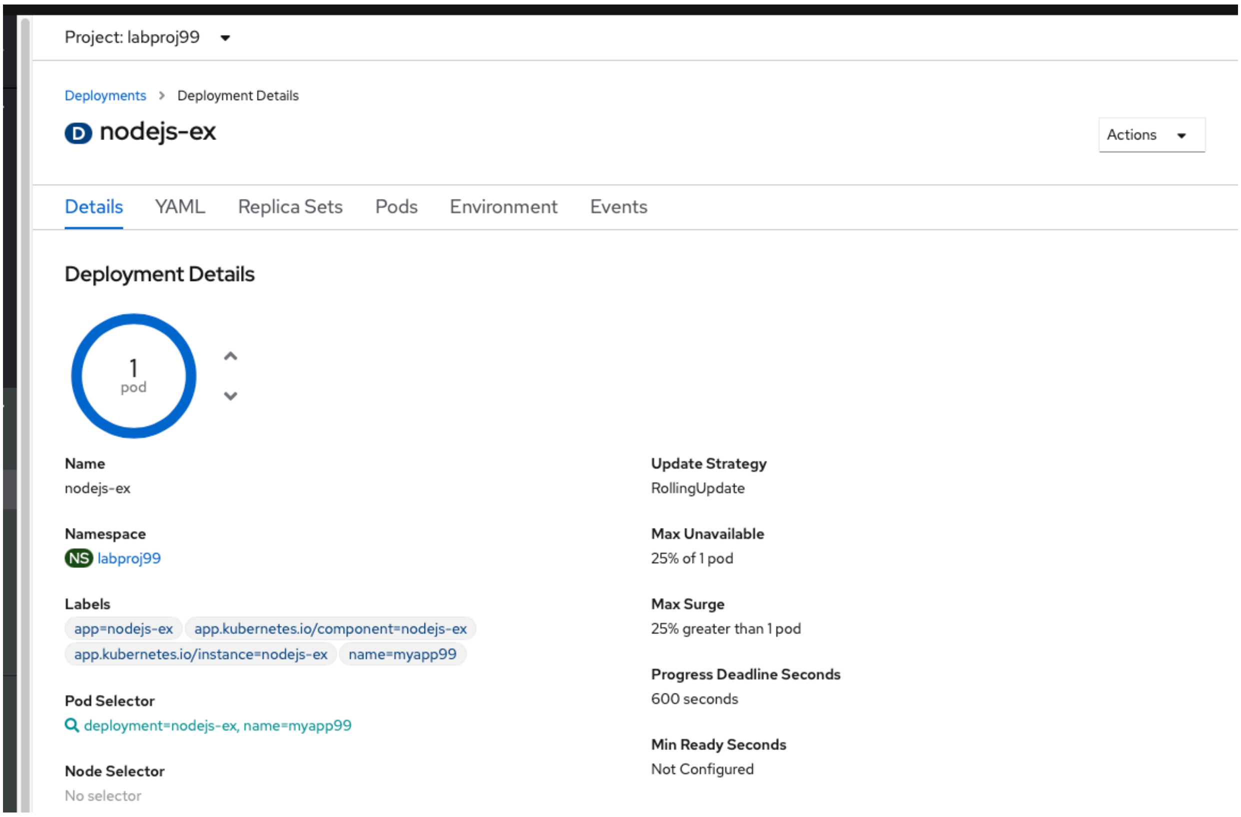
Create Deployment

Name | Search by name...

Name	Namespace	Status	Labels	Pod Selector

D nodejs-ex	NS labproj99	1 of 1 pods	app=nodejs-ex
			app.kubernetes.io... =nod...
			app.kubernetes.io... =node...
			name=myapp99
				deployment=nodejs-ex, name=myapp99

Let's click on that specific **deployment**. A deployement contains the **replicasets**, the **pods**, the **quotas** ...



Project: labproj99

Deployments > Deployment Details

D nodejs-ex

Actions

Deployment Details

Name
nodejs-ex

Namespace
NS labproj99

Labels
app=nodejs-ex app.kubernetes.io/component=nodejs-ex app.kubernetes.io/instance=nodejs-ex name=myapp99

Pod Selector
deployment=nodejs-ex, name=myapp99

Node Selector
No selector

Update Strategy
RollingUpdate

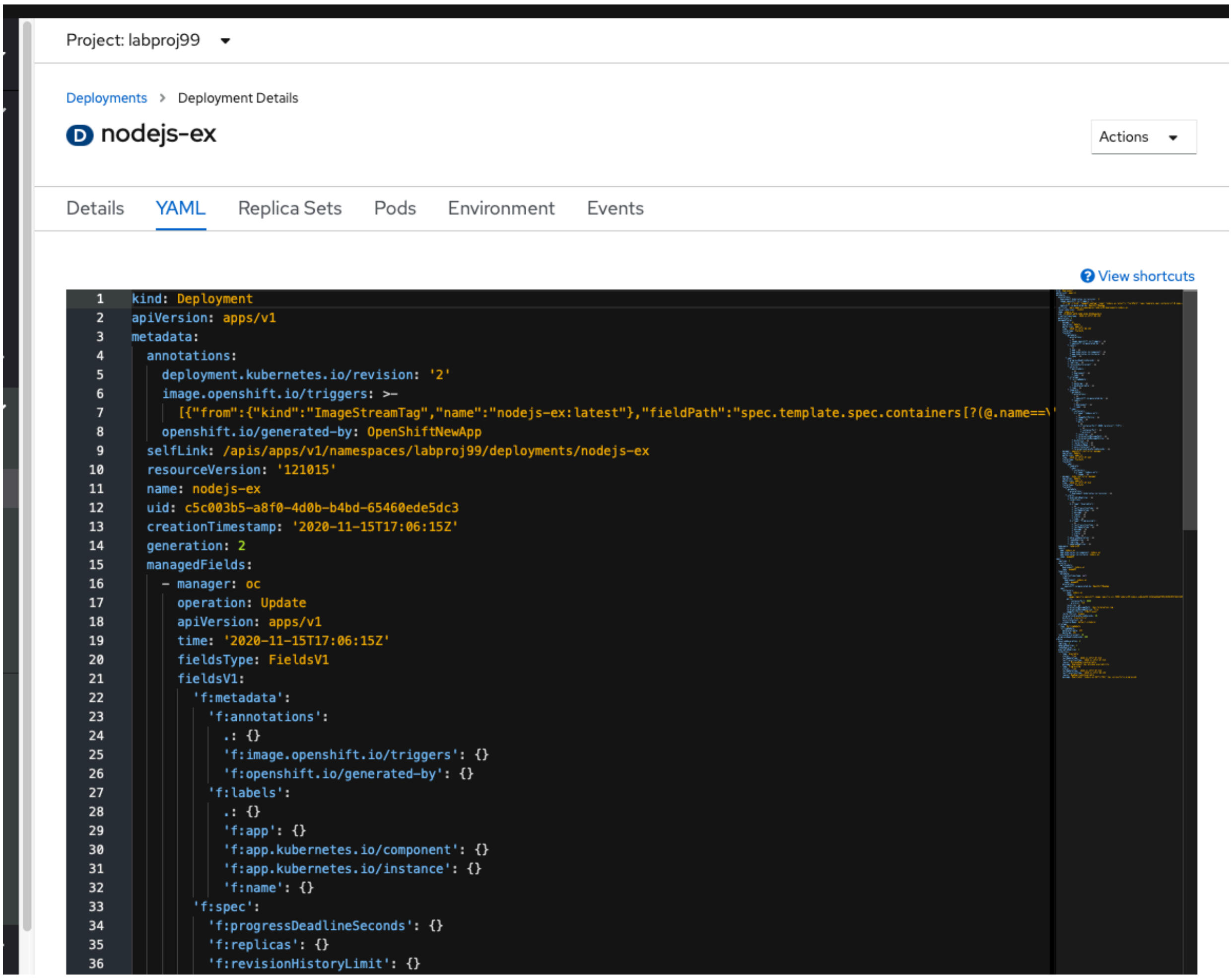
Max Unavailable
25% of 1 pod

Max Surge
25% greater than 1 pod

Progress Deadline Seconds
600 seconds

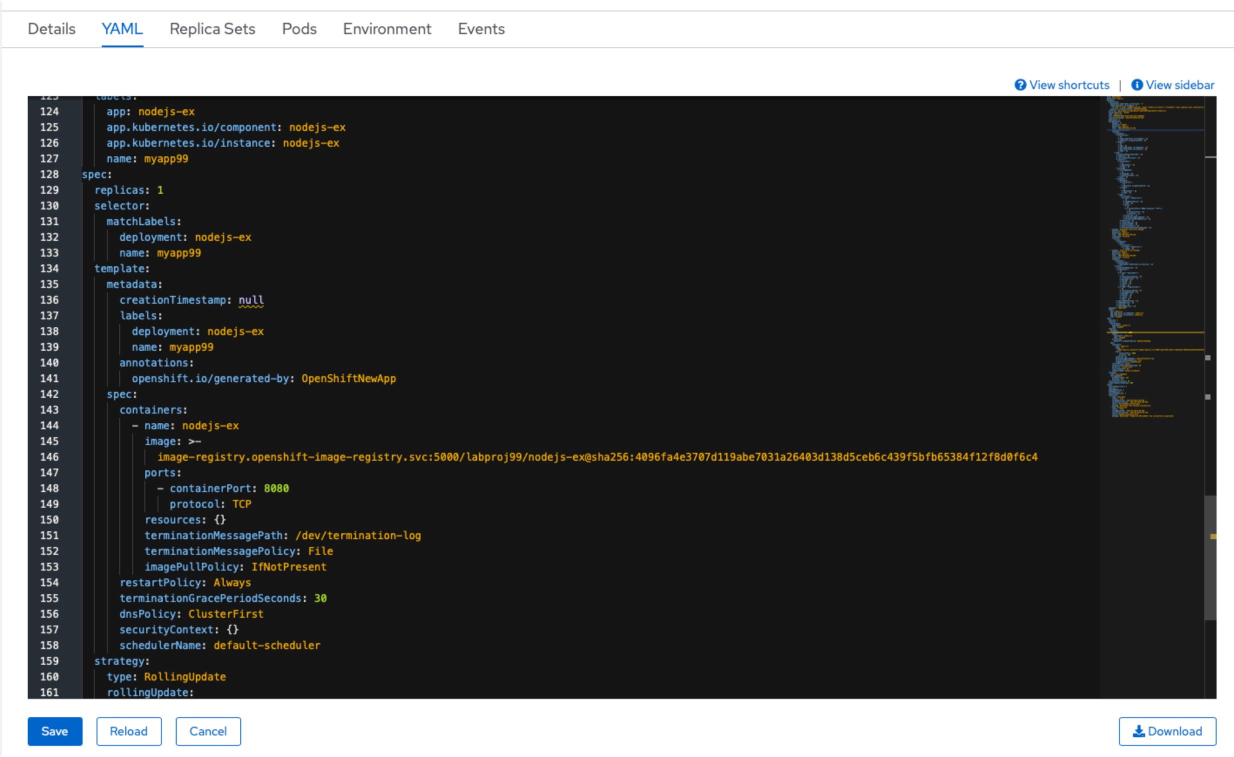
Min Ready Seconds
Not Configured

The **YAML definition** represents the definition of the deployment. In recent versions of Kubernetes, you will see a lot new rows (like the **fieldsType**). fieldsType are added automatically to help to understand how Kunetrnetes is working. Unfortunately, this doesn't add clarity to read the yaml definition.



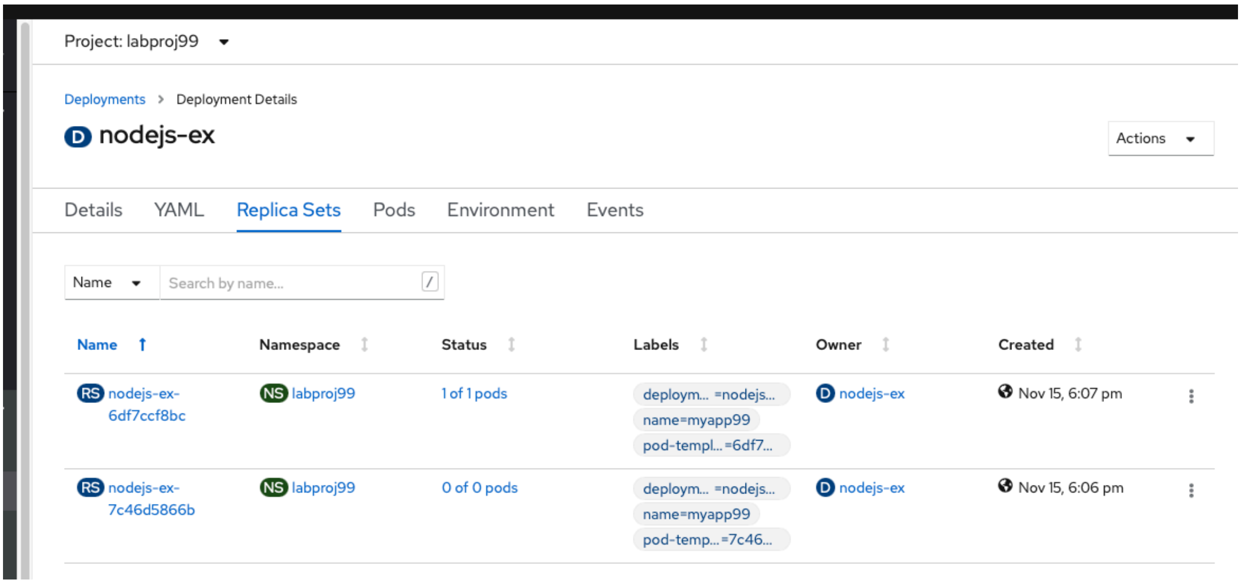
```yaml
kind: Deployment
apiVersion: apps/v1
metadata:
  annotations:
    deployment.kubernetes.io/revision: '2'
    image.openshift.io/triggers: >-
      [{"from":{"kind":"ImageStreamTag","name":"nodejs-ex:latest"},"fieldPath":"spec.template.spec.containers[?(@.name==\'
    openshift.io/generated-by: OpenShiftNewApp
  selfLink: /apis/apps/v1/namespaces/labproj99/deployments/nodejs-ex
  resourceVersion: '121015'
  name: nodejs-ex
  uid: c5c003b5-a8f0-4d0b-b4bd-65460ede5dc3
  creationTimestamp: '2020-11-15T17:06:15Z'
  generation: 2
  managedFields:
    - manager: oc
      operation: Update
      apiVersion: apps/v1
      time: '2020-11-15T17:06:15Z'
      fieldsType: FieldsV1
      fieldsV1:
        'f:metadata':
          'f:annotations':
            .: {}
            'f:image.openshift.io/triggers': {}
            'f:openshift.io/generated-by': {}
          'f:labels':
            .: {}
            'f:app': {}
            'f:app.kubernetes.io/component': {}
            'f:app.kubernetes.io/instance': {}
            'f:name': {}
        'f:spec':
          'f:progressDeadlineSeconds': {}
          'f:replicas': {}
          'f:revisionHistoryLimit': {}
```

If you browse the YAML file at the **end of the file**, you will see the POD specifications and the **image** of the container:



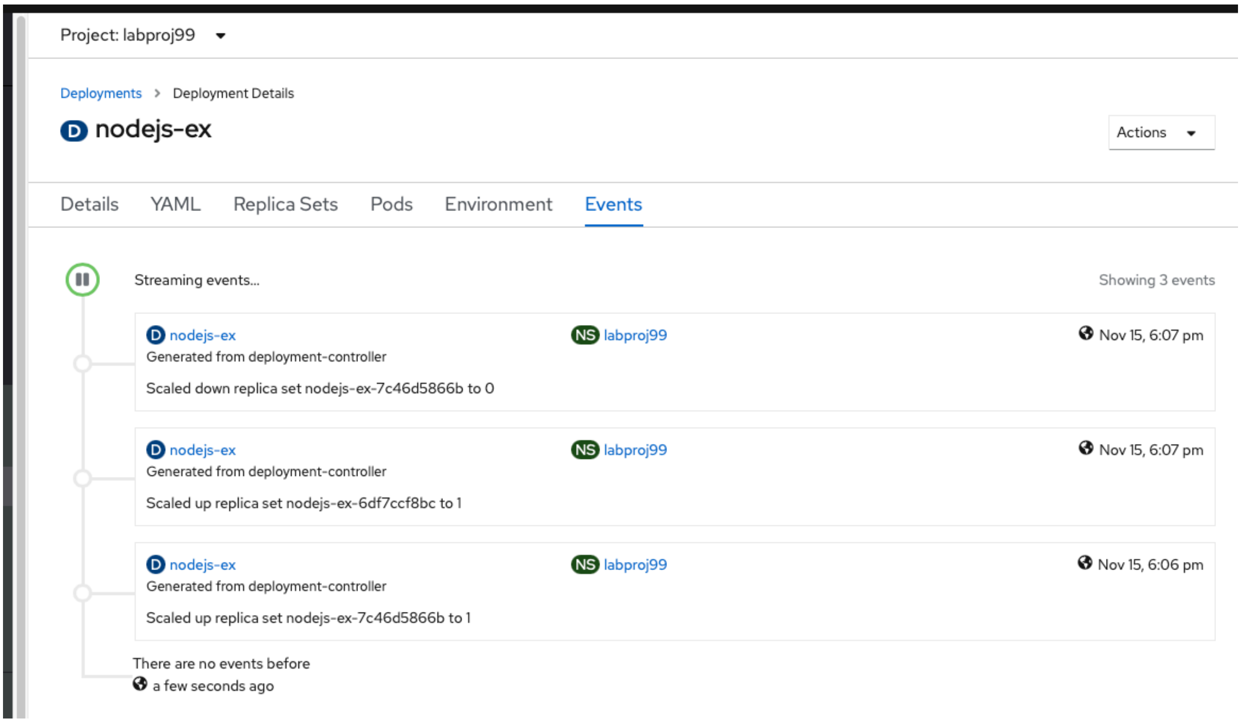
```
  labels:
    app: nodejs-ex
    app.kubernetes.io/component: nodejs-ex
    app.kubernetes.io/instance: nodejs-ex
    name: myapp99
spec:
  replicas: 1
  selector:
    matchLabels:
      deployment: nodejs-ex
      name: myapp99
  template:
    metadata:
      creationTimestamp: null
      labels:
        deployment: nodejs-ex
        name: myapp99
      annotations:
        openshift.io/generated-by: OpenShiftNewApp
    spec:
      containers:
        - name: nodejs-ex
          image: >-
            image-registry.openshift-image-registry.svc:5000/labproj99/nodejs-ex@sha256:4096fa4e3707d119abe7031a26403d138d5ceb6c439f5bfb65384f12f8d0f6c4
          ports:
            - containerPort: 8080
              protocol: TCP
          resources: {}
          terminationMessagePath: /dev/termination-log
          terminationMessagePolicy: File
          imagePullPolicy: IfNotPresent
      restartPolicy: Always
      terminationGracePeriodSeconds: 30
      dnsPolicy: ClusterFirst
      securityContext: {}
      schedulerName: default-scheduler
  strategy:
    type: RollingUpdate
    rollingUpdate:
```

The **Replica Sets** Tab will show you the number of replicas that we started (one here)



Name	Namespace	Status	Labels	Owner	Created
RS nodejs-ex-6df7ccf8bc	NS labproj99	1 of 1 pods	deploym... =nodejs... name=myapp99 pod-templ... =6df7...	D nodejs-ex	Nov 15, 6:07 pm
RS nodejs-ex-7c46d5866b	NS labproj99	0 of 0 pods	deploym... =nodejs... name=myapp99 pod-temp... =7c46...	D nodejs-ex	Nov 15, 6:06 pm

Finally the **Events** Tab (this log of events is concerning the activity around the deployment of the Pods):



Project: labproj99

Deployments > Deployment Details

D nodejs-ex

Streaming events...

Showing 3 events

D nodejs-ex NS labproj99 Nov 15, 6:07 pm
Generated from deployment-controller
Scaled down replica set nodejs-ex-7c46d5866b to 0

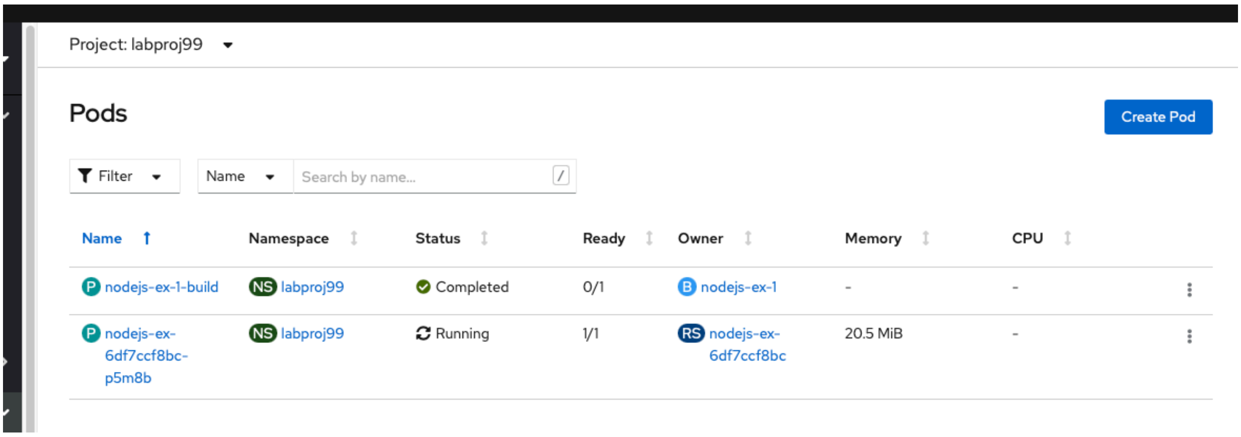
D nodejs-ex NS labproj99 Nov 15, 6:07 pm
Generated from deployment-controller
Scaled up replica set nodejs-ex-6df7ccf8bc to 1

D nodejs-ex NS labproj99 Nov 15, 6:06 pm
Generated from deployment-controller
Scaled up replica set nodejs-ex-7c46d5866b to 1

There are no events before
a few seconds ago

Task #10 - Into the PODs

Now from the deployment, click on the **Pods** tab. And you might see 2 PODs : one for the **build** (S2I) and the other concerning the running node.js **application**.

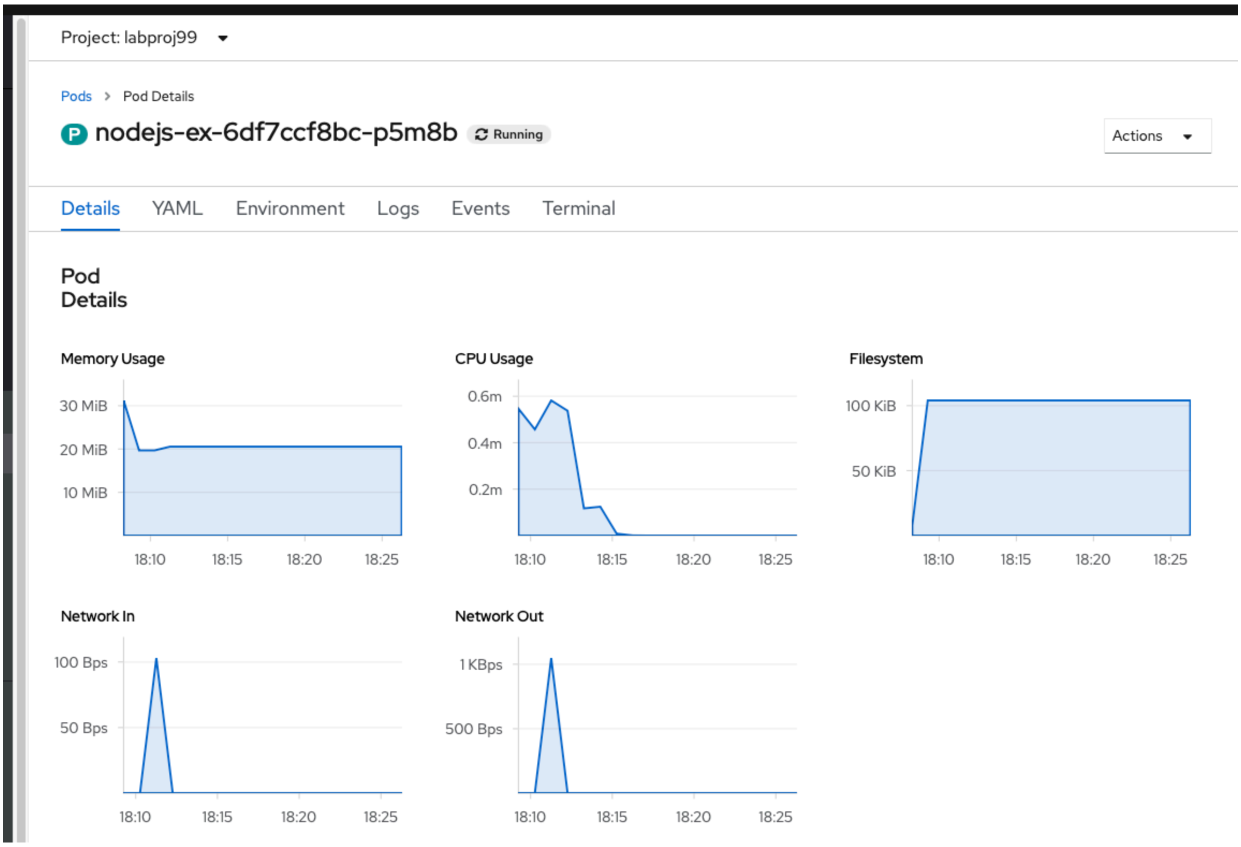


Project: labproj99

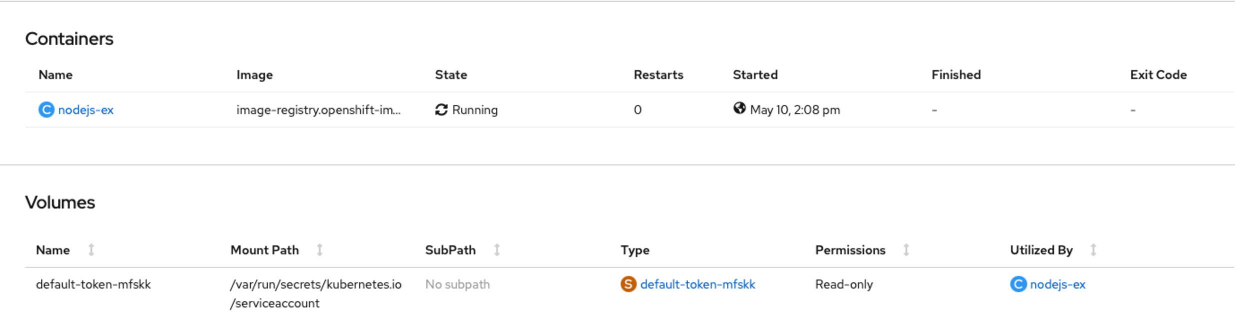
Pods

Name	Namespace	Status	Ready	Owner	Memory	CPU
P nodejs-ex-1-build	NS labproj99	Completed	0/1	B nodejs-ex-1	-	-
P nodejs-ex-6df7ccf8bc-p5m8b	NS labproj99	Running	1/1	RS nodejs-ex-6df7ccf8bc	20.5 MiB	-

You can notice that you have one pod that is running. **Click on that Pod**:



At the bottom of that page, you will see the list of **containers** and **volumes**



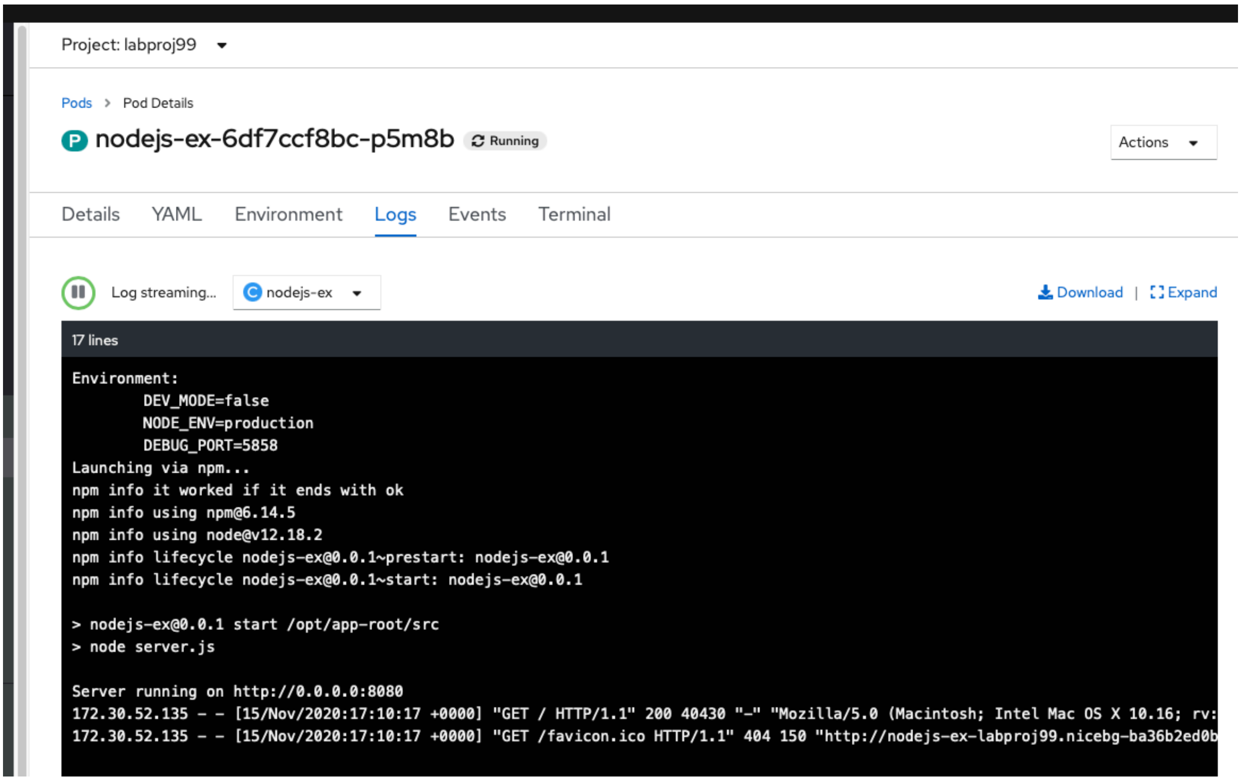
Containers

Name	Image	State	Restarts	Started	Finished	Exit Code
C nodejs-ex	image-registry.openshift-im...	Running	0	May 10, 2:08 pm	-	-

Volumes

Name	Mount Path	SubPath	Type	Permissions	Utilized By
default-token-mfskk	/var/run/secrets/kubernetes.io /serviceaccount	No subpath	S default-token-mfskk	Read-only	C nodejs-ex

Now click in the **log** (concerning the application) :

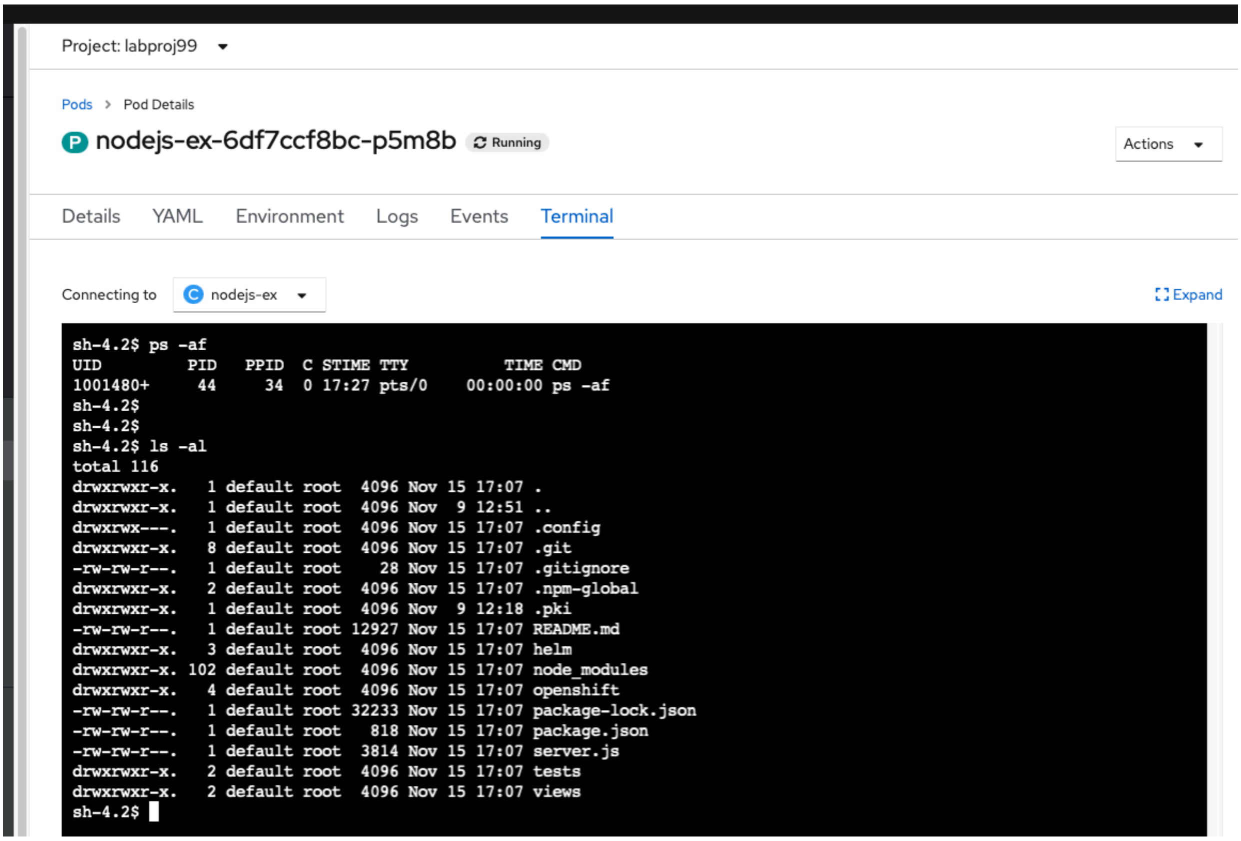


```
Environment:
        DEV_MODE=false
        NODE_ENV=production
        DEBUG_PORT=5858
Launching via npm...
npm info it worked if it ends with ok
npm info using npm@6.14.5
npm info using node@v12.18.2
npm info lifecycle nodejs-ex@0.0.1~prestart: nodejs-ex@0.0.1
npm info lifecycle nodejs-ex@0.0.1~start: nodejs-ex@0.0.1

> nodejs-ex@0.0.1 start /opt/app-root/src
> node server.js

Server running on http://0.0.0.0:8080
172.30.52.135 - - [15/Nov/2020:17:10:17 +0000] "GET / HTTP/1.1" 200 40430 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.16; rv:
172.30.52.135 - - [15/Nov/2020:17:10:17 +0000] "GET /favicon.ico HTTP/1.1" 404 150 "http://nodejs-ex-labproj99.nicebg-ba36b2ed0b
```

And then finally, click on the Terminal Tab to get access **inside the container** (try typing several linux commands like ps or ls):

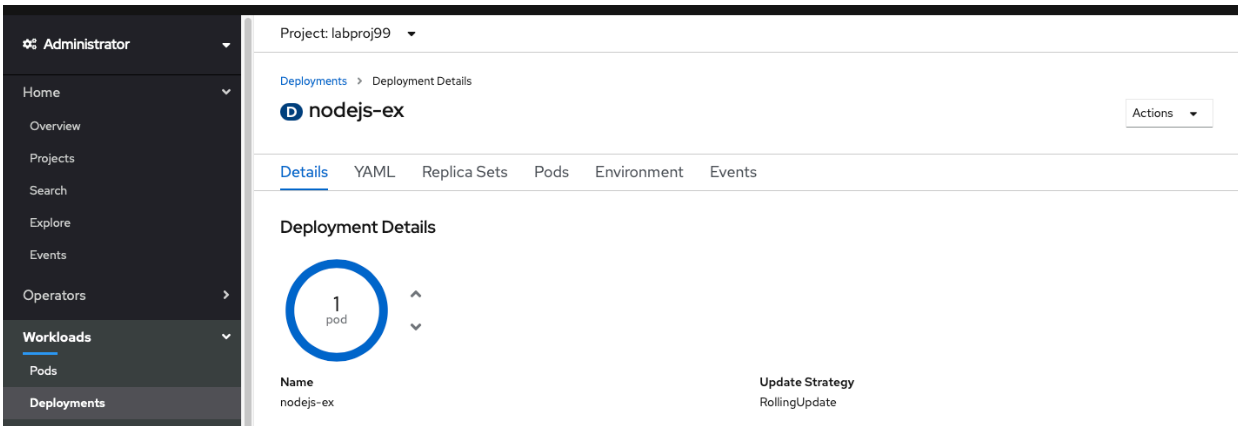


```
sh-4.2$ ps -af
UID         PID   PPID  C STIME TTY          TIME CMD
1001480+     44     34  0 17:27 pts/0    00:00:00 ps -af
sh-4.2$
sh-4.2$
sh-4.2$ ls -al
total 116
drwxrwxr-x.   1 default root  4096 Nov 15 17:07 .
drwxrwxr-x.   1 default root  4096 Nov  9 12:51 ..
drwxrwx---.   1 default root  4096 Nov 15 17:07 .config
drwxrwxr-x.   8 default root  4096 Nov 15 17:07 .git
-rw-rw-r--.   1 default root    28 Nov 15 17:07 .gitignore
drwxrwxr-x.   2 default root  4096 Nov 15 17:07 .npm-global
drwxrwxr-x.   1 default root  4096 Nov  9 12:18 .pki
-rw-rw-r--.   1 default root 12927 Nov 15 17:07 README.md
drwxrwxr-x.   3 default root  4096 Nov 15 17:07 helm
drwxrwxr-x. 102 default root  4096 Nov 15 17:07 node_modules
drwxrwxr-x.   4 default root  4096 Nov 15 17:07 openshift
-rw-rw-r--.   1 default root 32233 Nov 15 17:07 package-lock.json
-rw-rw-r--.   1 default root   818 Nov 15 17:07 package.json
-rw-rw-r--.   1 default root  3814 Nov 15 17:07 server.js
drwxrwxr-x.   2 default root  4096 Nov 15 17:07 tests
drwxrwxr-x.   2 default root  4096 Nov 15 17:07 views
sh-4.2$
```

Task #11 - Scaling your application

One important task of the cluster administrator is to manage the workload and to be able to increase the number of PODs when the number of requests is becoming more important.

To learn about scaling your application, on the left pane, click on the **Deployment** and then click on your specific deployment:

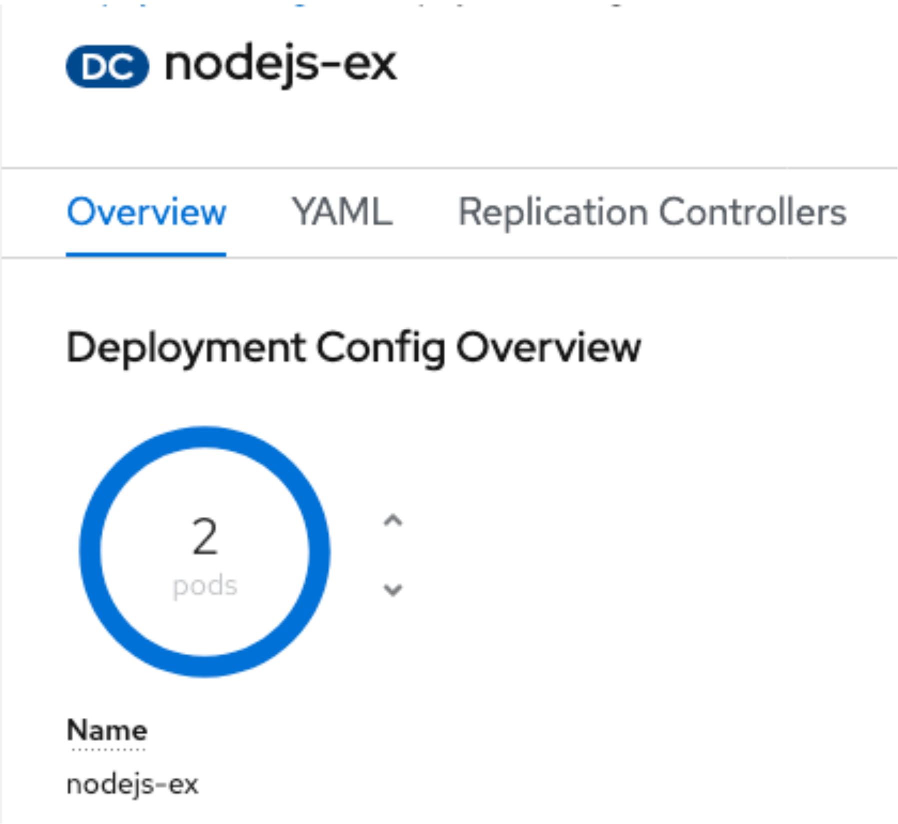


Project: labproj99

Name
nodejs-ex

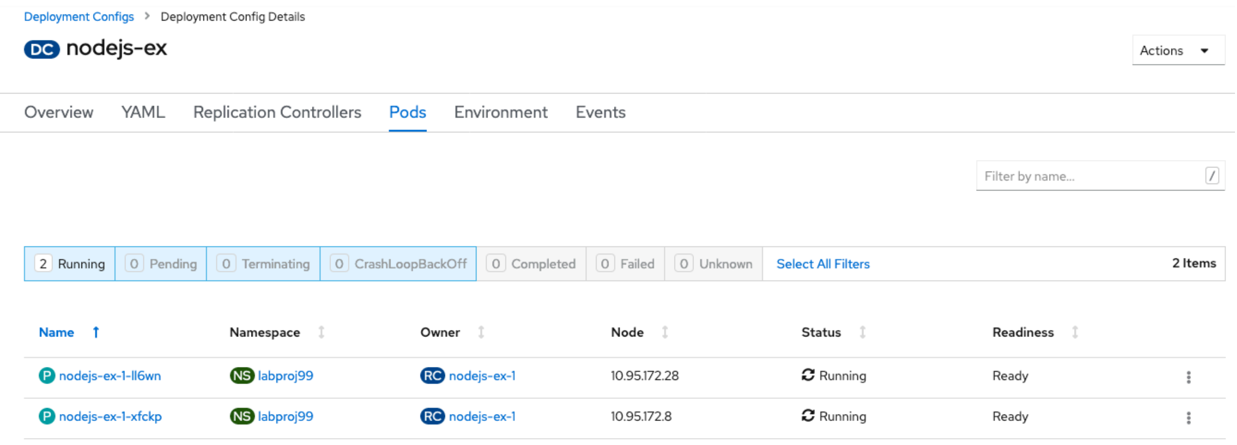
Update Strategy
RollingUpdate

Finally, increase the number of pods (**not too much**: 2 for instance)



Name
nodejs-ex

And after a while, you will see 2 active pods.



Deployment Configs > Deployment Config Details

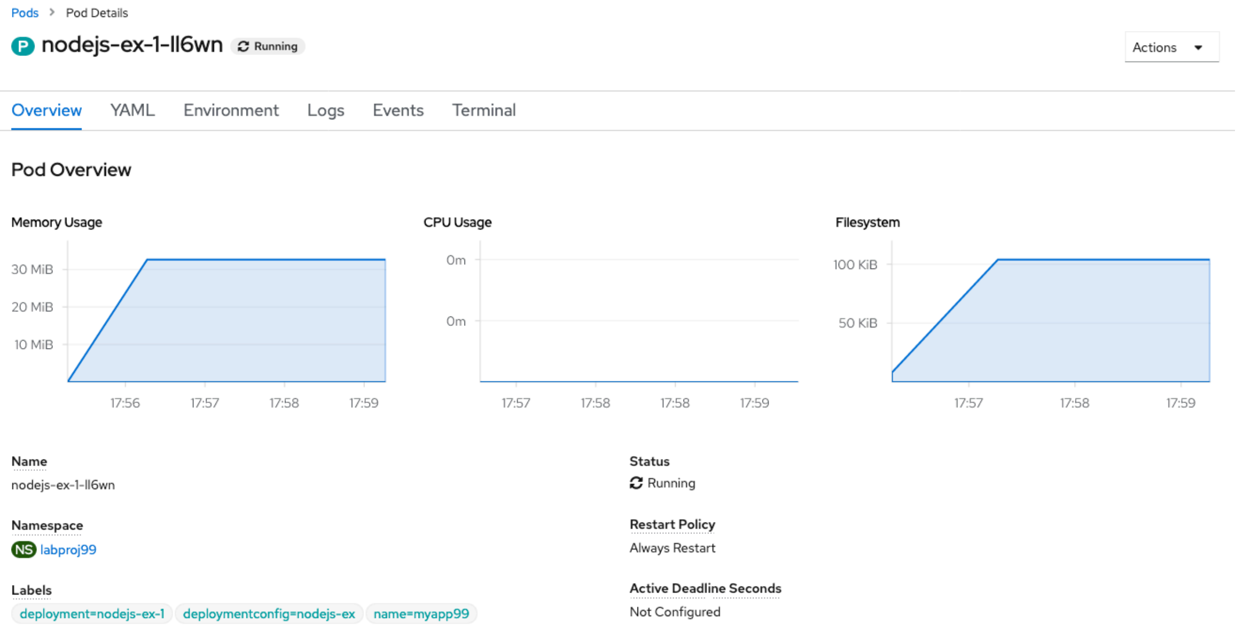
DC nodejs-ex

2 Items

Name	Namespace	Owner	Node	Status	Readiness
P nodejs-ex-1-ll6wn	NS labproj99	RC nodejs-ex-1	10.95.172.28	Running	Ready
P nodejs-ex-1-xfckp	NS labproj99	RC nodejs-ex-1	10.95.172.8	Running	Ready

This means that now 2 pods are serving the requests behind the same kubernetes service. And all the requests will go in round robin automatically on one or the other pods.

Now click on one of these new **pods** and you should see a page like this one showing new activity:



Pods > Pod Details

P nodejs-ex-1-ll6wn Running

Pod Overview

Name
nodejs-ex-1-ll6wn

Namespace
NS labproj99

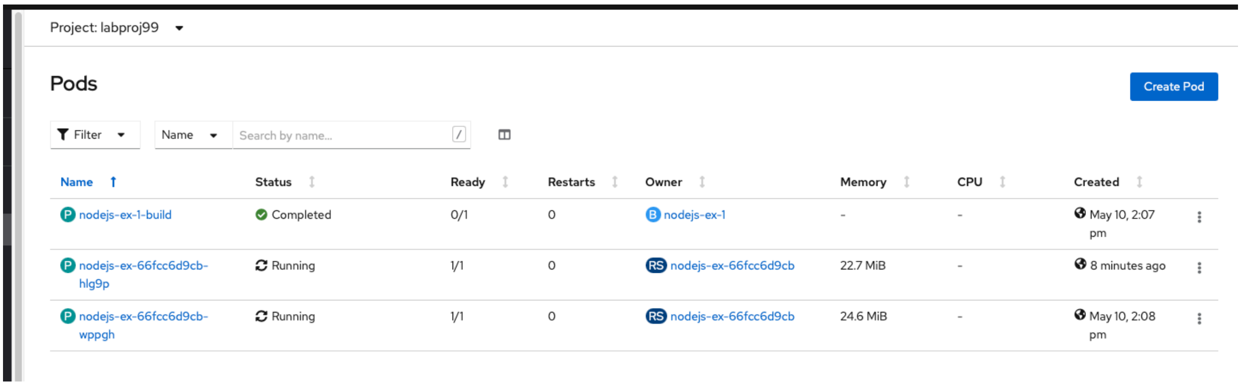
Labels
deployment=nodejs-ex-1 deploymentconfig=nodejs-ex name=myapp99

Status
Running

Restart Policy
Always Restart

Active Deadline Seconds
Not Configured

Now go back to the **list of PODs** (click on Pods on the left pane):

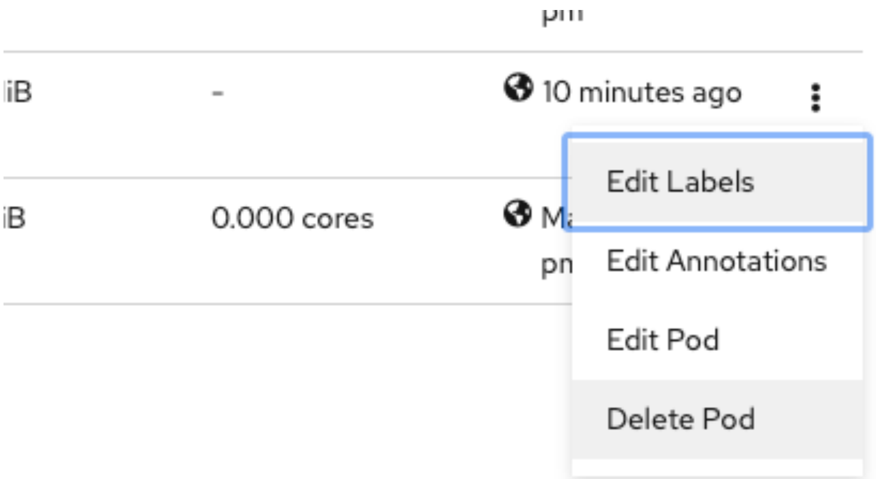


Project: labproj99

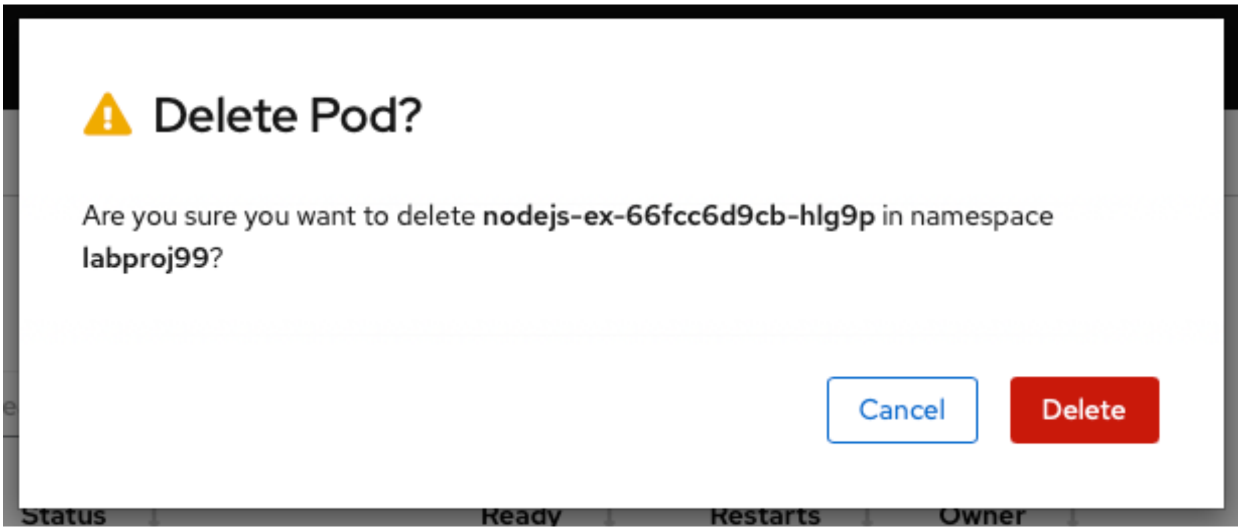
Pods

Name	Status	Ready	Restarts	Owner	Memory	CPU	Created
P nodejs-ex-1-build	Completed	0/1	0	B nodejs-ex-1	-	-	May 10, 2:07 pm
P nodejs-ex-66fcc6d9cb-hlg9p	Running	1/1	0	RS nodejs-ex-66fcc6d9cb	22.7 MiB	-	8 minutes ago
P nodejs-ex-66fcc6d9cb-wppgh	Running	1/1	0	RS nodejs-ex-66fcc6d9cb	24.6 MiB	-	May 10, 2:08 pm

Let's try a crazy experience and kill a pod ! click on the three dots at the end of the row and select `delete Pod`



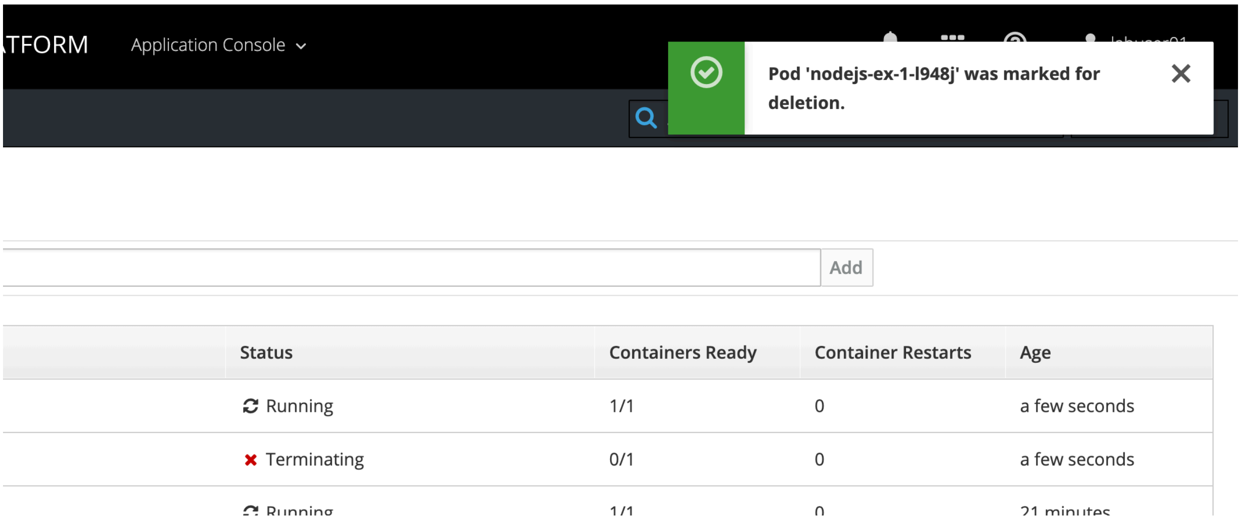
The following popup window will appear:



Delete Pod?

Are you sure you want to delete **nodejs-ex-66fcc6d9cb-hlg9p** in namespace **labproj99**?

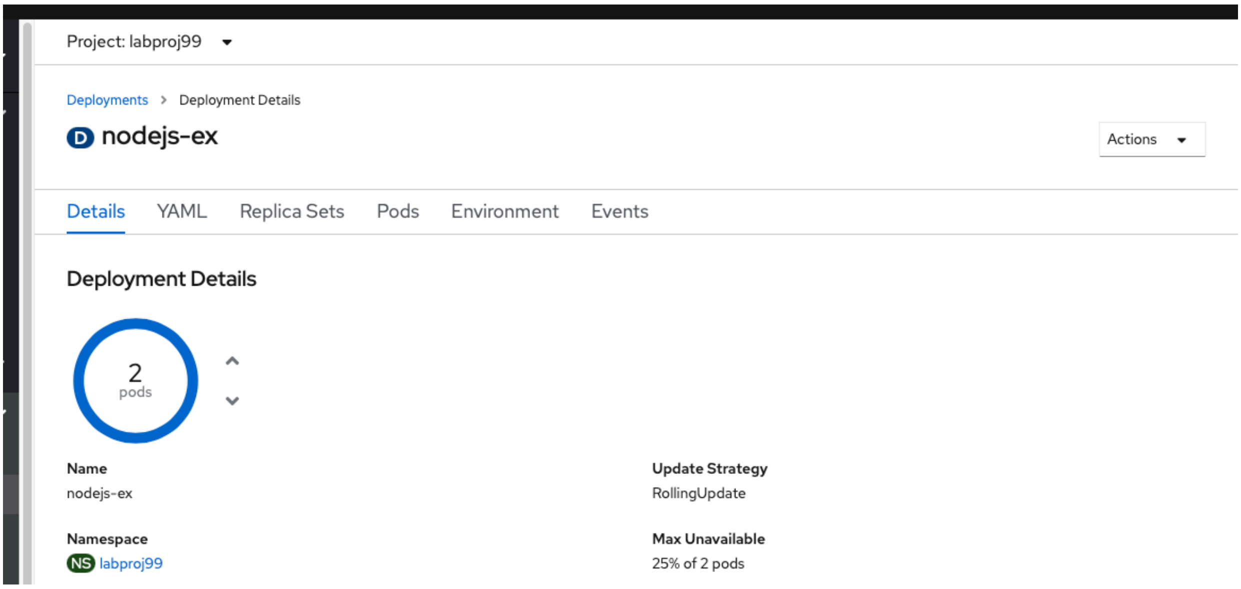
Click on `Delete` and you should see breifly your pod terminating.



Pod 'nodejs-ex-1-l948j' was marked for deletion.

Status	Containers Ready	Container Restarts	Age
Running	1/1	0	a few seconds
✖ Terminating	0/1	0	a few seconds
Running	1/1	0	21 minutes

A new pod is **automatically started** !!! Because the number of **replicaset** has been defined to **2** , even if a pod is crashing or has been deleted, it will be replace automatically by a new one.



Project: labproj99

Deployments > Deployment Details

D nodejs-ex

Actions

Details | YAML | Replica Sets | Pods | Environment | Events

Deployment Details

2 pods

Name
nodejs-ex

Namespace
NS labproj99

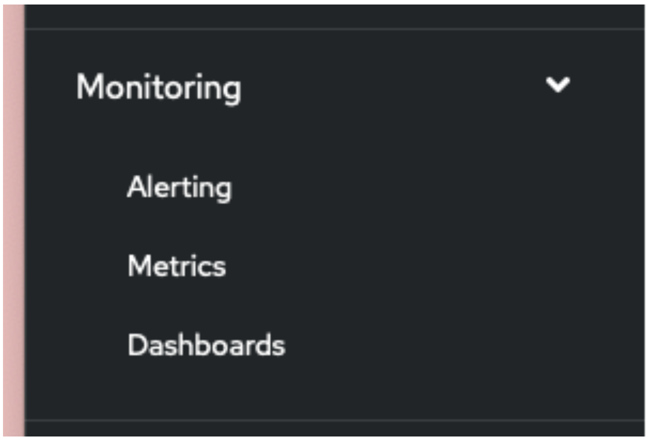
Update Strategy
RollingUpdate

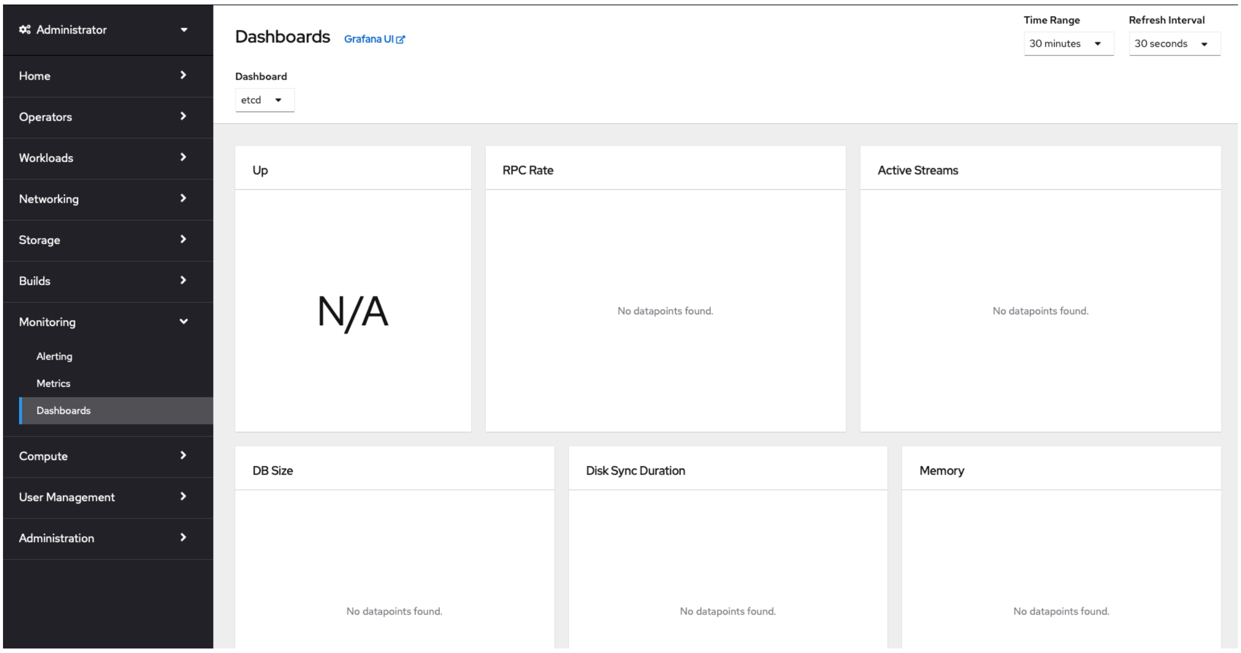
Max Unavailable
25% of 2 pods

Task #12 - Monitoring your cluster

In Openshift, out of the box, you have a Prometheus server and a Grafana server that you can access from the OpenShift web Console.

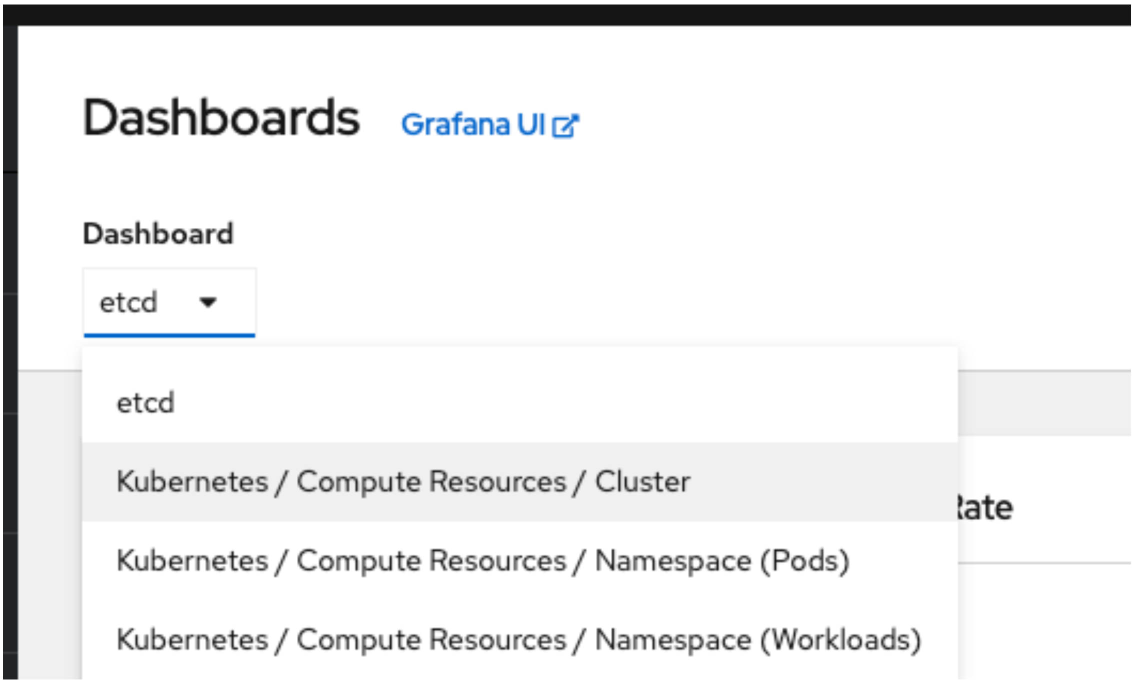
Click on the **Monitoring** section on the left pane and then click on **Dashboards**:



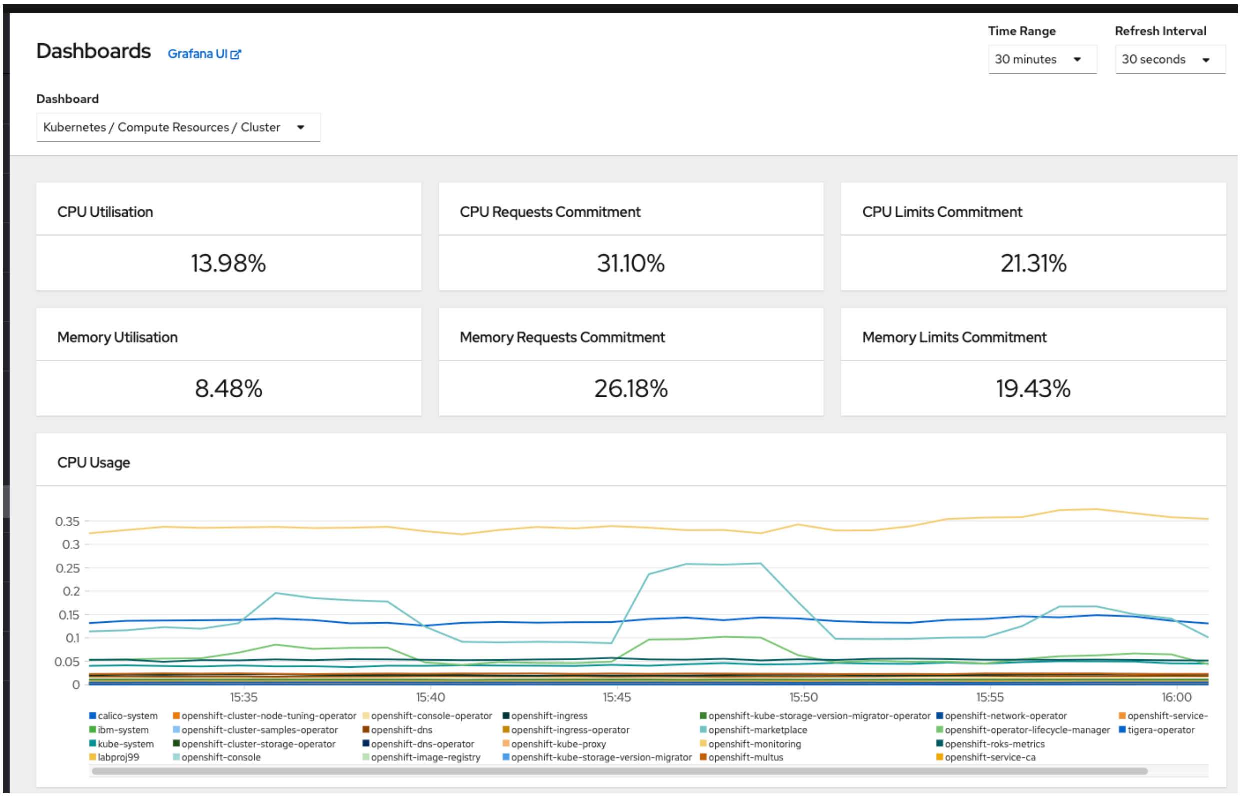


No data is collected for ETCD so the dashboard is empty.

Select a different dashboard in the list: Select the Cluster dashboard:


Dashboards
Grafana UI
Dashboard
etcd
etcd
Kubernetes / Compute Resources / Cluster
Kubernetes / Compute Resources / Namespace (Pods)
Kubernetes / Compute Resources / Namespace (Workloads)
ate

Then you will see this page with global metrics (CPU, RAM, Network, and commitments). Browse the page:



Dashboards Grafana UI

Time Range: 30 minutes

Refresh Interval: 30 seconds

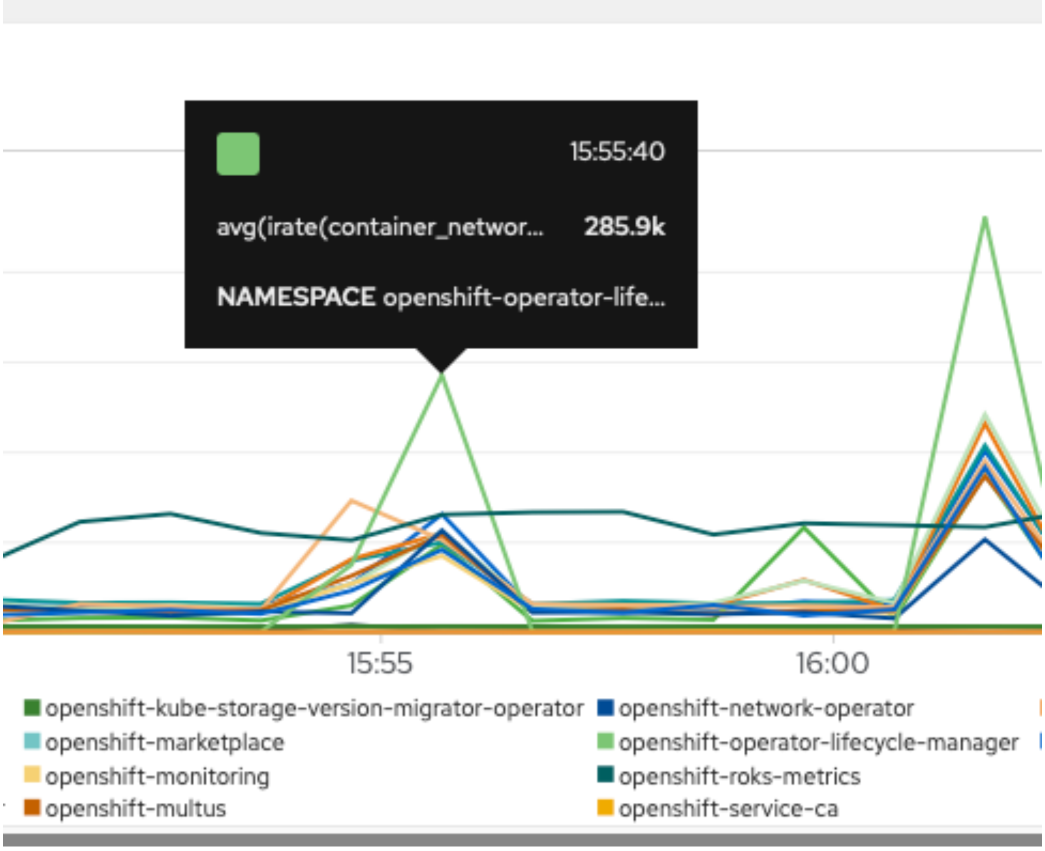
Dashboard: Kubernetes / Compute Resources / Cluster

CPU Utilisation	CPU Requests Commitment	CPU Limits Commitment
13.98%	31.10%	21.31%

Memory Utilisation	Memory Requests Commitment	Memory Limits Commitment
8.48%	26.18%	19.43%

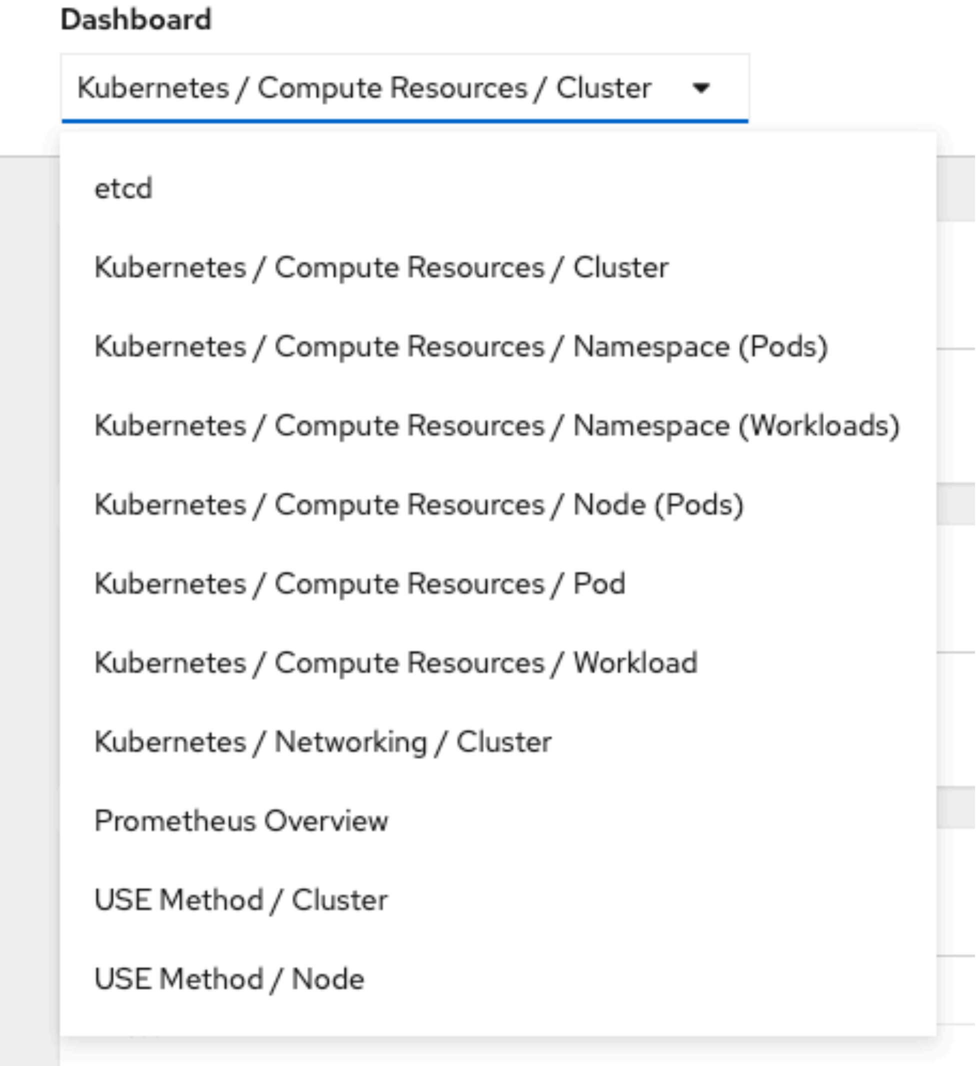
CPU Usage

Then move your cursor to a graph to show specific metrics:



15:55:40
avg(irate(container_networ... 285.9k
NAMESPACE openshift-operator-life...
15:55
16:00
openshift-kube-storage-version-migrator-operator
openshift-marketplace
openshift-monitoring
openshift-multus
openshift-network-operator
openshift-operator-lifecycle-manager
openshift-roks-metrics
openshift-service-ca

You can explore some other dashboards from the list:

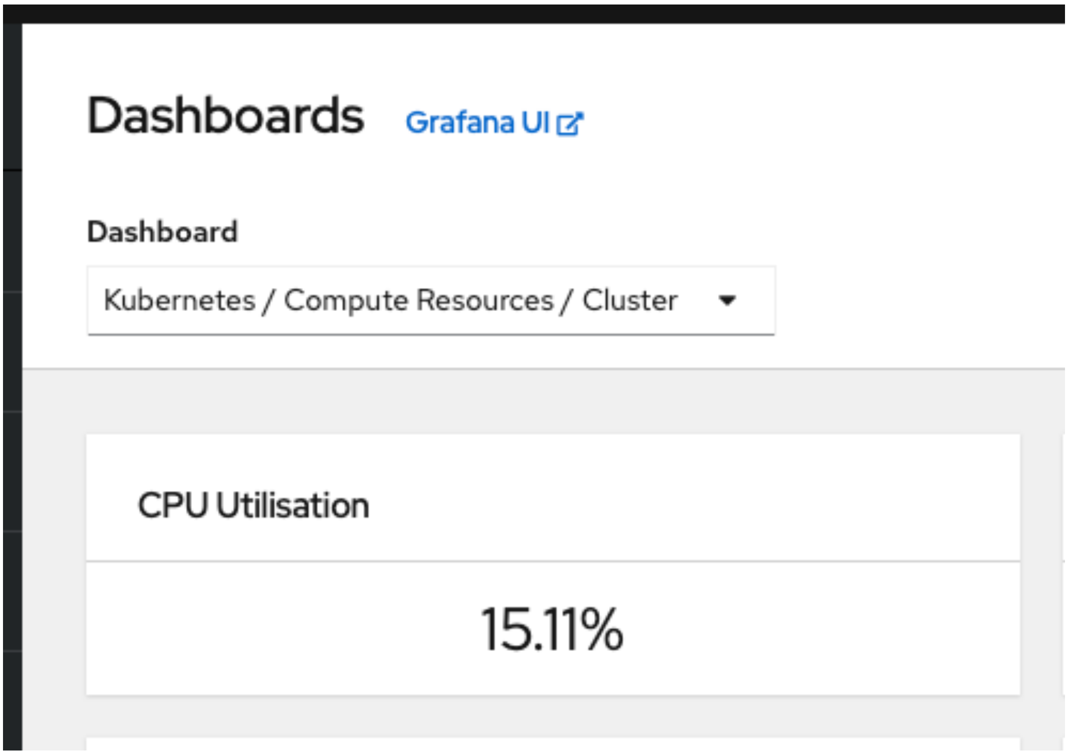


Dashboard

Kubernetes / Compute Resources / Cluster

etcd
Kubernetes / Compute Resources / Cluster
Kubernetes / Compute Resources / Namespace (Pods)
Kubernetes / Compute Resources / Namespace (Workloads)
Kubernetes / Compute Resources / Node (Pods)
Kubernetes / Compute Resources / Pod
Kubernetes / Compute Resources / Workload
Kubernetes / Networking / Cluster
Prometheus Overview
USE Method / Cluster
USE Method / Node

You can also go to **Grafana UI**:



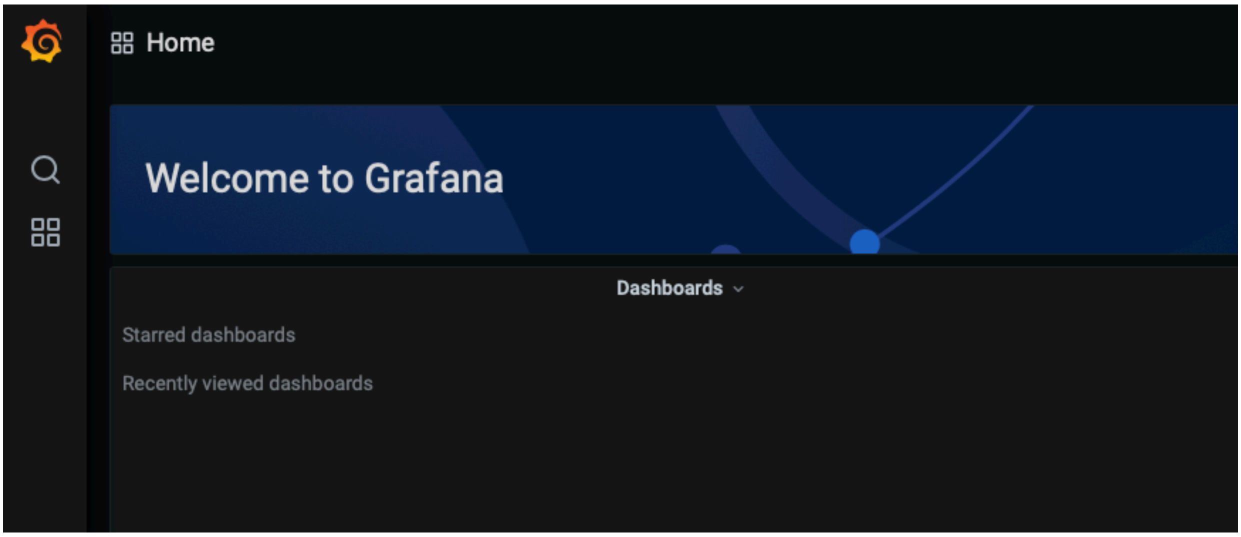
Dashboards

Dashboard

Kubernetes / Compute Resources / Cluster

CPU Utilisation

15.11%



Home

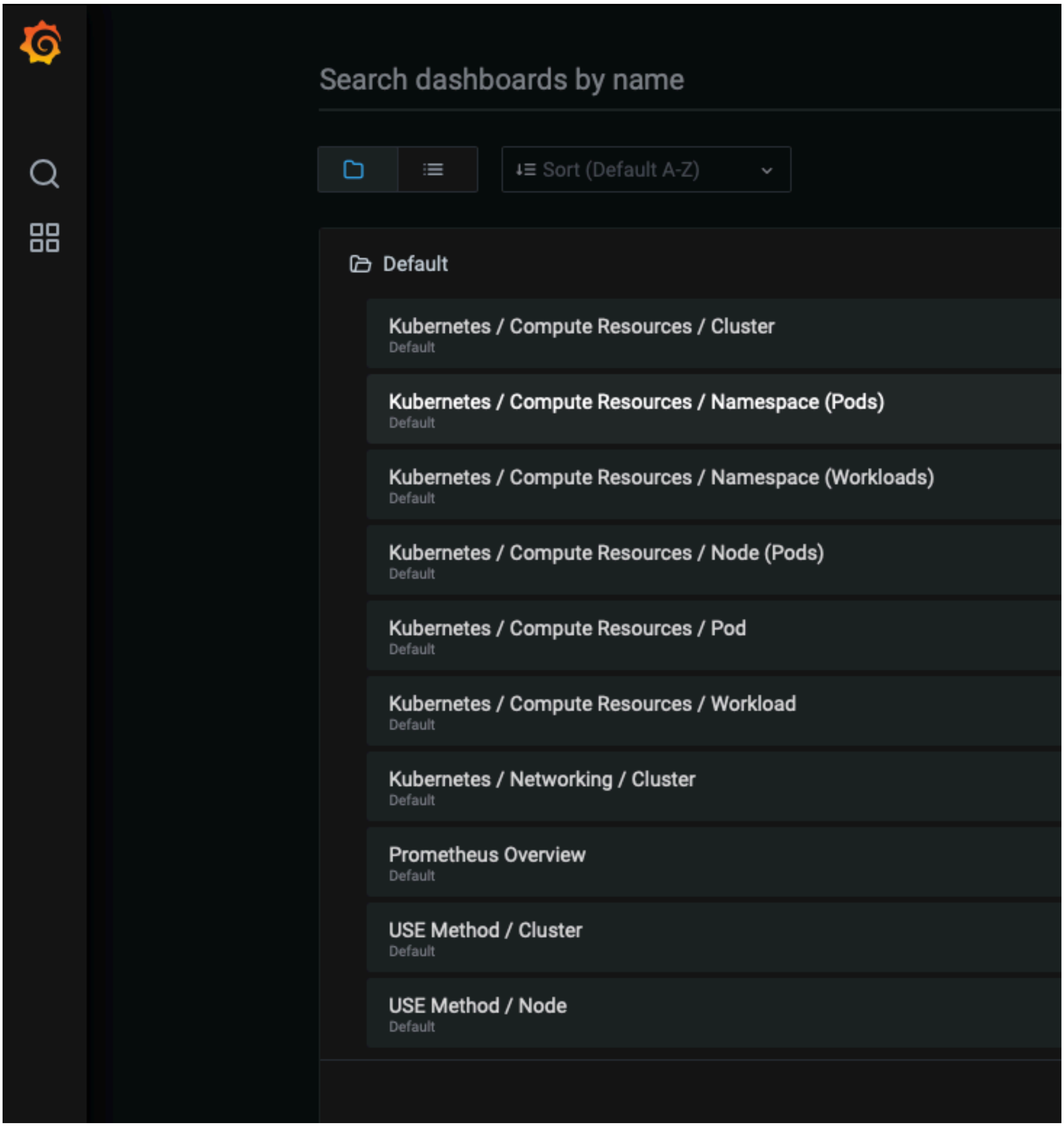
Welcome to Grafana

Dashboards

Starred dashboards

Recently viewed dashboards

Click **Home > Default**

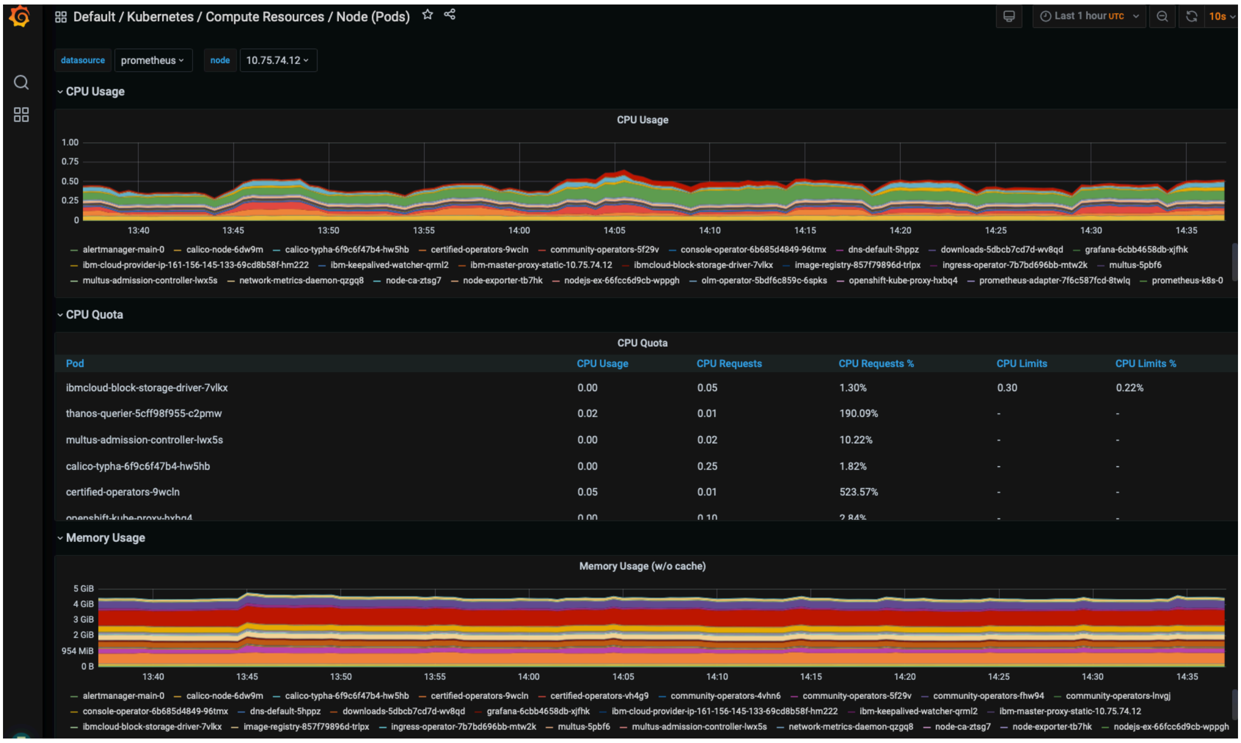


Search dashboards by name

Default

- Kubernetes / Compute Resources / Cluster
  Default
- Kubernetes / Compute Resources / Namespace (Pods)
  Default
- Kubernetes / Compute Resources / Namespace (Workloads)
  Default
- Kubernetes / Compute Resources / Node (Pods)
  Default
- Kubernetes / Compute Resources / Pod
  Default
- Kubernetes / Compute Resources / Workload
  Default
- Kubernetes / Networking / Cluster
  Default
- Prometheus Overview
  Default
- USE Method / Cluster
  Default
- USE Method / Node
  Default

Pick one graph in the list:

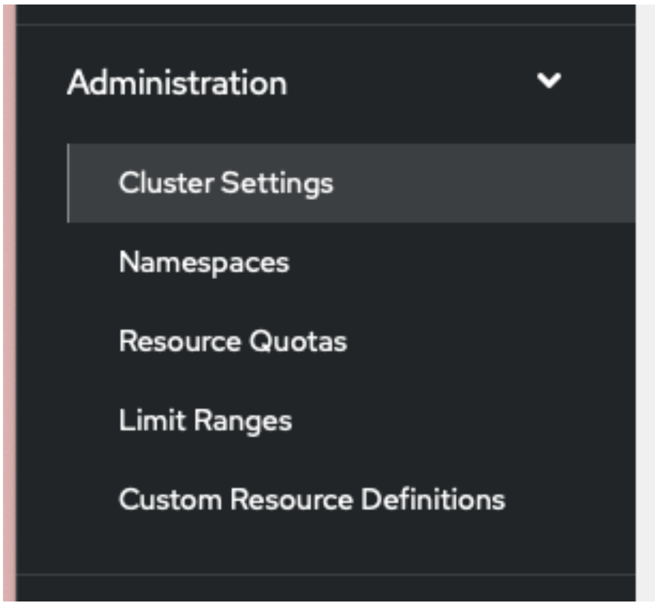

Default / Kubernetes / Compute Resources / Node (Pods)
Last 1 hour UTC
10s
datasource
prometheus
node
10.75.74.12
CPU Usage
CPU Usage
1.00
0.75
0.50
0.25
0
13:40
13:45
13:50
13:55
14:00
14:05
14:10
14:15
14:20
14:25
14:30
14:35
alertmanager-main-0
calico-node-6dw9m
calico-typha-6f9c6f47b4-hw5hb
certified-operators-9wcln
community-operators-5f29v
console-operator-6b685d4849-96tmx
dns-default-5hppz
downloads-5dbcb7cd7d-wv8qd
grafana-6cbb4658db-xjfhk
ibm-cloud-provider-ip-161-156-145-133-69cd8b58f-hm222
ibm-keepalived-watcher-qrml2
ibm-master-proxy-static-10.75.74.12
ibmcloud-block-storage-driver-7vlkx
image-registry-857f79896d-trlpx
ingress-operator-7b7bd696bb-mtw2k
multus-5pbf6
multus-admission-controller-lwx5s
network-metrics-daemon-qzgq8
node-ca-ztsg7
node-exporter-tb7hk
nodejs-ex-66fcc6d9cb-wppgh
olm-operator-5bdf6c859c-6spks
openshift-kube-proxy-hxbq4
prometheus-adapter-7f6c587fcd-8twlq
prometheus-k8s-0
CPU Quota
CPU Quota

Pod	CPU Usage	CPU Requests	CPU Requests %	CPU Limits	CPU Limits %
ibmcloud-block-storage-driver-7vlkx	0.00	0.05	1.30%	0.30	0.22%
thanos-querier-5cff98f955-c2pmw	0.02	0.01	190.09%	-	-
multus-admission-controller-lwx5s	0.00	0.02	10.22%	-	-
calico-typha-6f9c6f47b4-hw5hb	0.00	0.25	1.82%	-	-
certified-operators-9wcln	0.05	0.01	523.57%	-	-
openshift-kube-proxy-hxbq4	0.00	0.10	2.84%	-	-

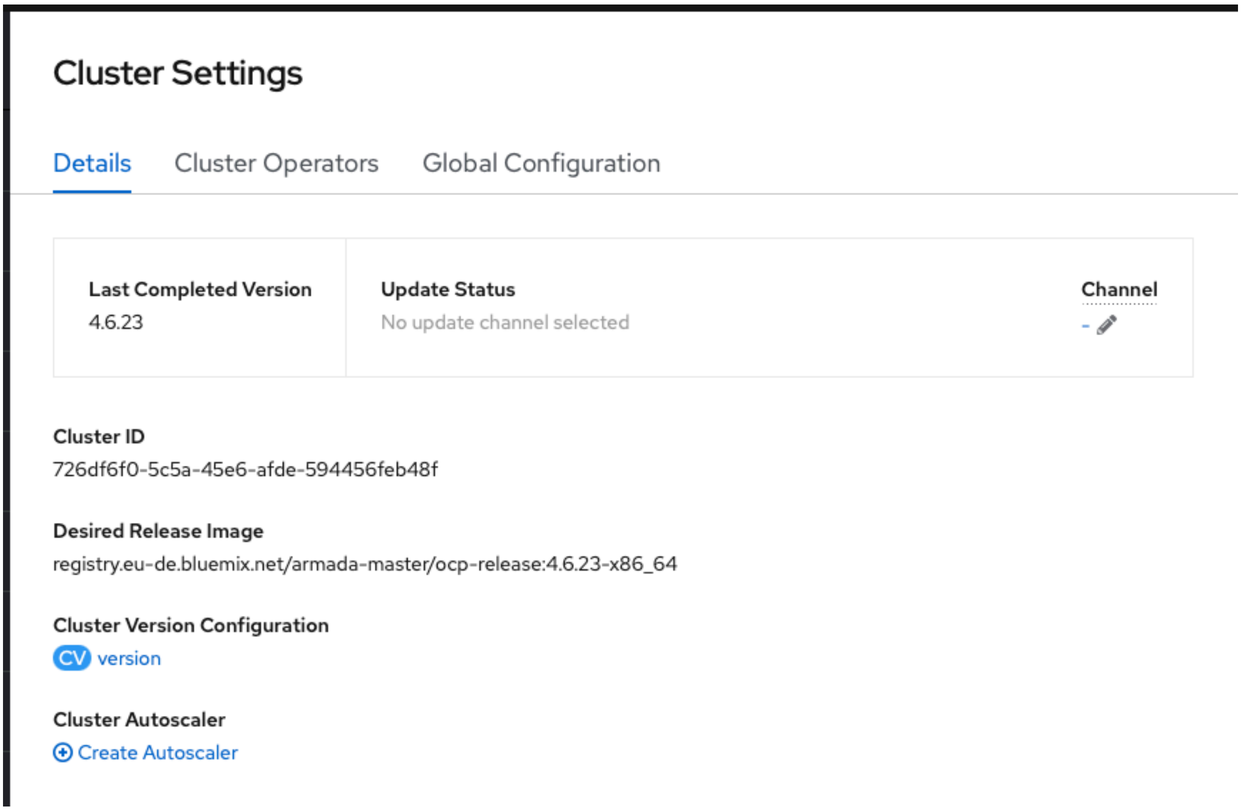
Memory Usage
Memory Usage (w/o cache)
5 GiB
4 GiB
3 GiB
2 GiB
954 MiB
0 B
13:40
13:45
13:50
13:55
14:00
14:05
14:10
14:15
14:20
14:25
14:30
14:35
alertmanager-main-0
calico-node-6dw9m
calico-typha-6f9c6f47b4-hw5hb
certified-operators-9wcln
certified-operators-vh4g9
community-operators-4vhn6
community-operators-5f29v
community-operators-fhw94
community-operators-lnvgj
console-operator-6b685d4849-96tmx
dns-default-5hppz
downloads-5dbcb7cd7d-wv8qd
grafana-6cbb4658db-xjfhk
ibm-cloud-provider-ip-161-156-145-133-69cd8b58f-hm222
ibm-keepalived-watcher-qrml2
ibm-master-proxy-static-10.75.74.12
ibmcloud-block-storage-driver-7vlkx
image-registry-857f79896d-trlpx
ingress-operator-7b7bd696bb-mtw2k
multus-5pbf6
multus-admission-controller-lwx5s
network-metrics-daemon-qzgq8
node-ca-ztsg7
node-exporter-tb7hk
nodejs-ex-66fcc6d9cb-wppgh

Task #13 - Cluster Administration

From the left pane, click on **Cluster Settings**:



In this section, you can upgrade the cluster version to a newer one (no update at the moment):



Cluster Settings

Last Completed Version
4.6.23

Update Status
No update channel selected

Channel
-

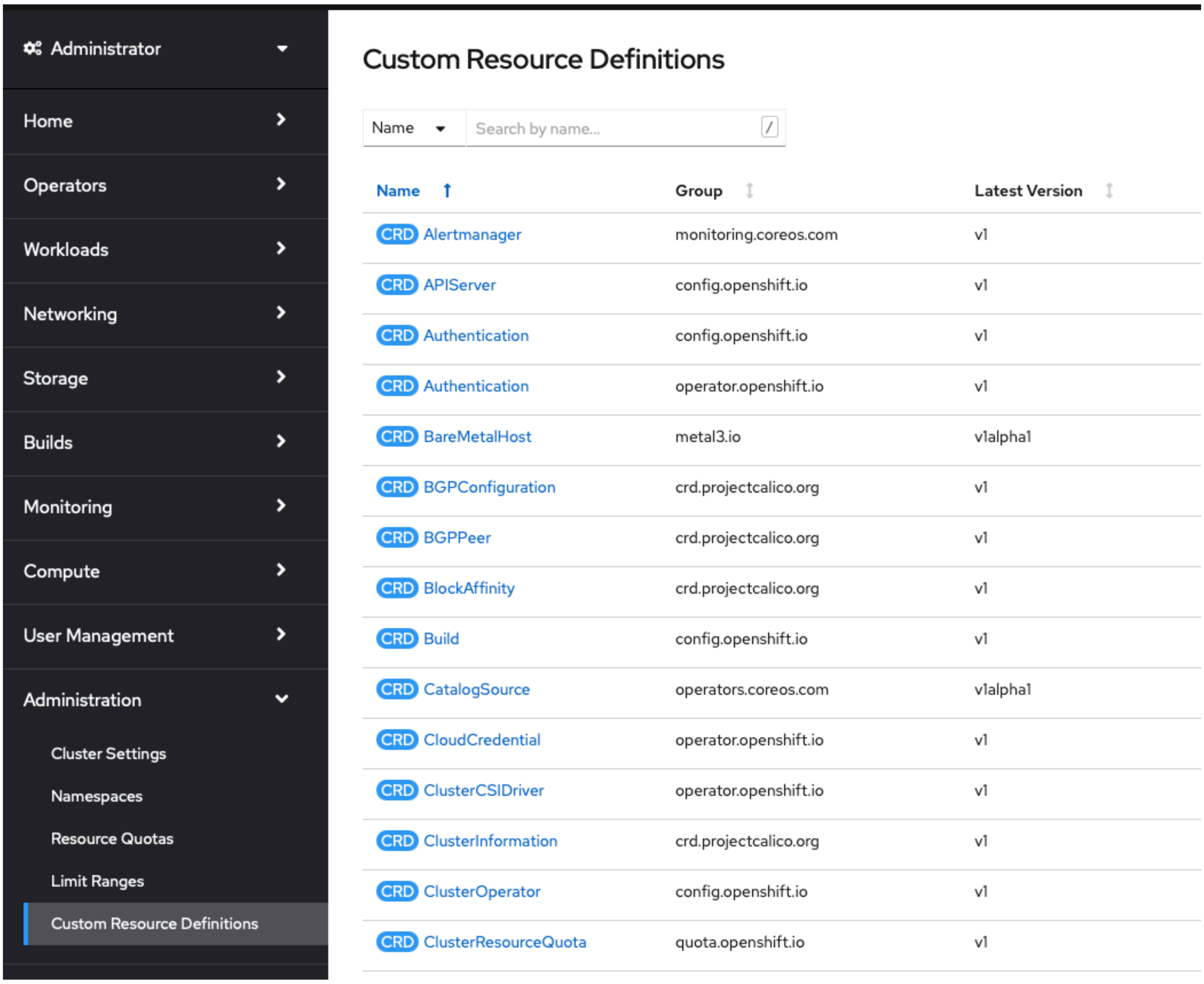
Cluster ID
726df6f0-5c5a-45e6-afde-594456feb48f

Desired Release Image
registry.eu-de.bluemix.net/armada-master/ocp-release:4.6.23-x86_64

Cluster Version Configuration
CV version

Cluster Autoscaler
Create Autoscaler

To look at the different **Custom Resource Definitions** (CRD) :

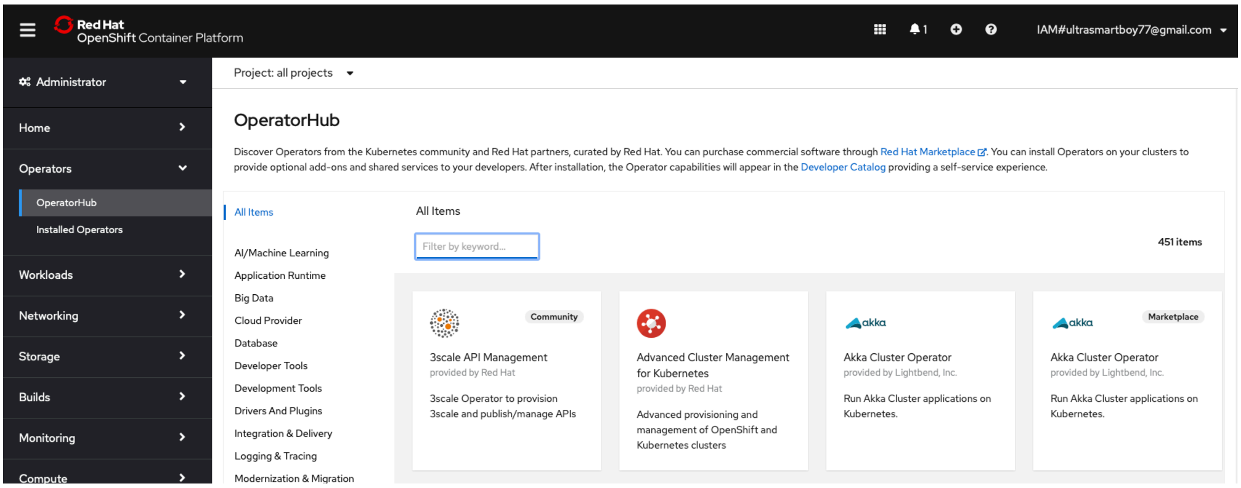


Custom Resource Definitions

Name	Group	Latest Version
CRD Alertmanager	monitoring.coreos.com	v1
CRD APIServer	config.openshift.io	v1
CRD Authentication	config.openshift.io	v1
CRD Authentication	operator.openshift.io	v1
CRD BareMetalHost	metal3.io	v1alpha1
CRD BGPConfiguration	crd.projectcalico.org	v1
CRD BGPPeer	crd.projectcalico.org	v1
CRD BlockAffinity	crd.projectcalico.org	v1
CRD Build	config.openshift.io	v1
CRD CatalogSource	operators.coreos.com	v1alpha1
CRD CloudCredential	operator.openshift.io	v1
CRD ClusterCSIDriver	operator.openshift.io	v1
CRD ClusterInformation	crd.projectcalico.org	v1
CRD ClusterOperator	config.openshift.io	v1
CRD ClusterResourceQuota	quota.openshift.io	v1

CRDs are generaly used in **Operators**.

Go to **OperatorHub** to list all possible Operator that you can install on your cluster:



Red Hat
OpenShift Container Platform

IAM#ultrasmartboy77@gmail.com

Administrator

Home

Operators

OperatorHub

Installed Operators

Workloads

Networking

Storage

Builds

Monitoring

Compute

Project: all projects

OperatorHub

Discover Operators from the Kubernetes community and Red Hat partners, curated by Red Hat. You can purchase commercial software through Red Hat Marketplace. You can install Operators on your clusters to provide optional add-ons and shared services to your developers. After installation, the Operator capabilities will appear in the Developer Catalog providing a self-service experience.

All Items

AI/Machine Learning
Application Runtime
Big Data
Cloud Provider
Database
Developer Tools
Development Tools
Drivers And Plugins
Integration & Delivery
Logging & Tracing
Modernization & Migration

All Items

Filter by keyword...

451 items

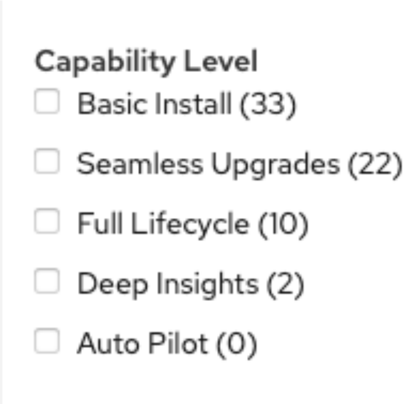
Community
3scale API Management
provided by Red Hat
3scale Operator to provision 3scale and publish/manage APIs

Advanced Cluster Management for Kubernetes
provided by Red Hat
Advanced provisioning and management of OpenShift and Kubernetes clusters

Akka Cluster Operator
provided by Lightbend, Inc.
Run Akka Cluster applications on Kubernetes.

Marketplace
Akka Cluster Operator
provided by Lightbend, Inc.
Run Akka Cluster applications on Kubernetes.

Most of the solutions are installed by using operators. Each operator corresponds to a specific solution or application. Operators propose **5 levels of capabilities** that you can on the left side of this page :



Capability Level

□ Basic Install (33)

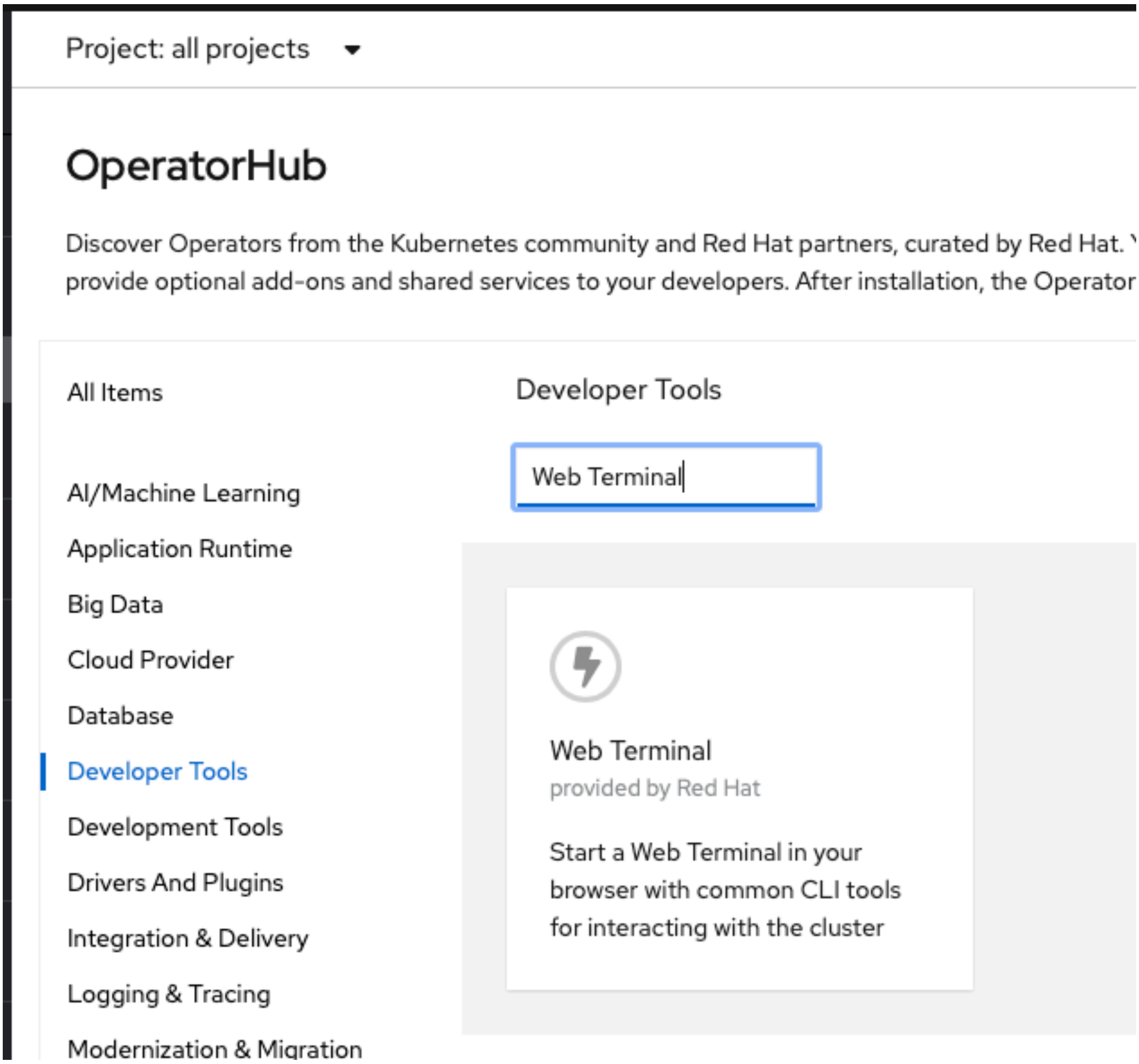
□ Seamless Upgrades (22)

□ Full Lifecycle (10)

□ Deep Insights (2)

□ Auto Pilot (0)

Find the `Web Terminal` operator :



Project: all projects

OperatorHub

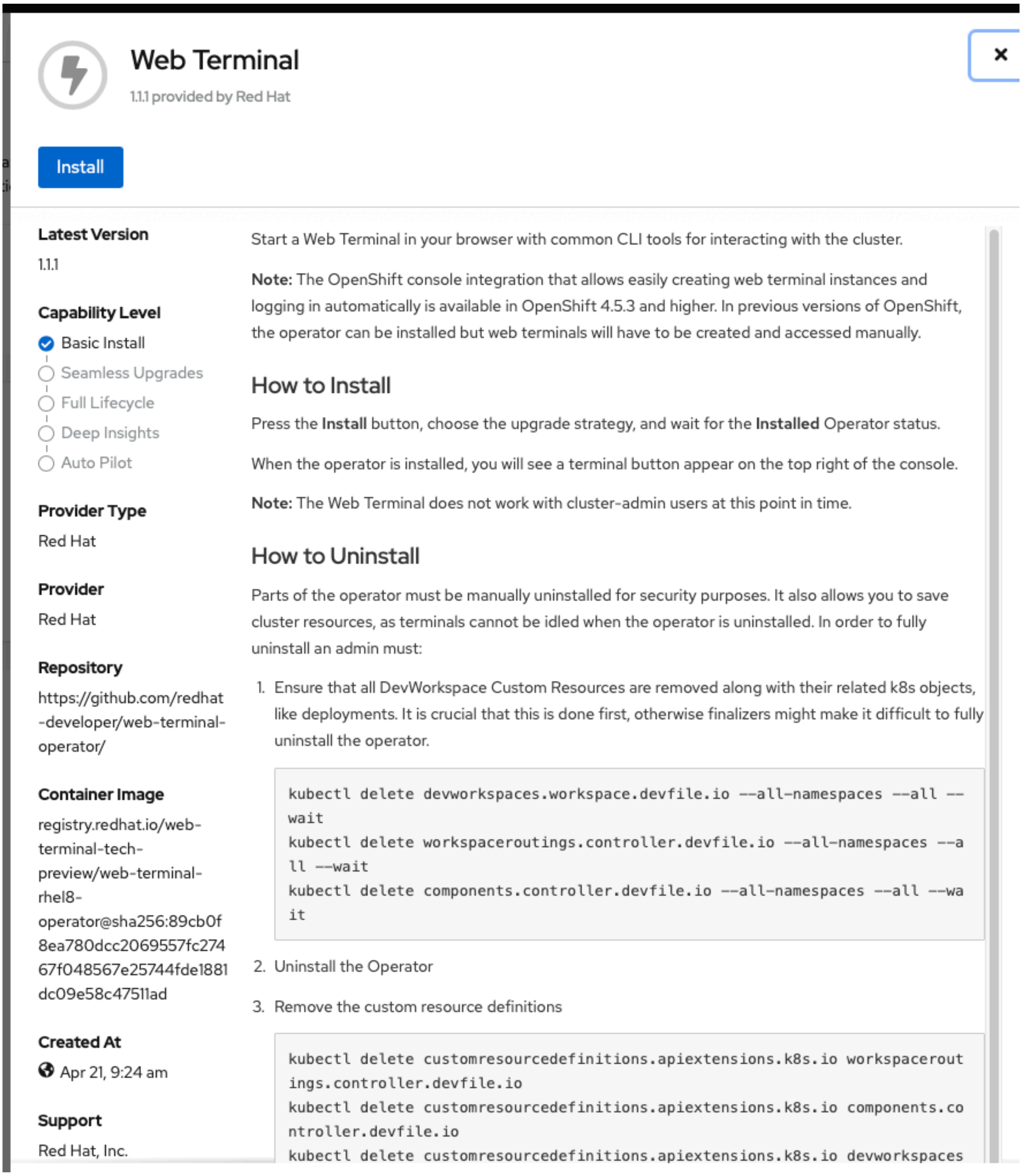
Discover Operators from the Kubernetes community and Red Hat partners, curated by Red Hat. Y provide optional add-ons and shared services to your developers. After installation, the Operator

Web Terminal

Web Terminal
provided by Red Hat

Start a Web Terminal in your browser with common CLI tools for interacting with the cluster

Click on the Web Terminal tile to see the details of the installation :



Web Terminal

1.1.1 provided by Red Hat

Latest Version

1.1.1

Capability Level

● Basic Install
○ Seamless Upgrades
○ Full Lifecycle
○ Deep Insights
○ Auto Pilot

Provider Type

Red Hat

Provider

Red Hat

Repository

https://github.com/redhat-developer/web-terminal-operator/

Container Image

registry.redhat.io/web-terminal-tech-preview/web-terminal-rhel8-operator@sha256:89cb0f8ea780dcc2069557fc27467f048567e25744fde1881dc09e58c47511ad

Created At

Apr 21, 9:24 am

Support

Red Hat, Inc.

Start a Web Terminal in your browser with common CLI tools for interacting with the cluster.

Note: The OpenShift console integration that allows easily creating web terminal instances and logging in automatically is available in OpenShift 4.5.3 and higher. In previous versions of OpenShift, the operator can be installed but web terminals will have to be created and accessed manually.

How to Install

Press the **Install** button, choose the upgrade strategy, and wait for the **Installed** Operator status.

When the operator is installed, you will see a terminal button appear on the top right of the console.

Note: The Web Terminal does not work with cluster-admin users at this point in time.

How to Uninstall

Parts of the operator must be manually uninstalled for security purposes. It also allows you to save cluster resources, as terminals cannot be idled when the operator is uninstalled. In order to fully uninstall an admin must:

1. Ensure that all DevWorkspace Custom Resources are removed along with their related k8s objects, like deployments. It is crucial that this is done first, otherwise finalizers might make it difficult to fully uninstall the operator.

```
kubectl delete devworkspaces.workspace.devfile.io --all-namespaces --all --
wait
kubectl delete workspaceroutings.controller.devfile.io --all-namespaces --a
ll --wait
kubectl delete components.controller.devfile.io --all-namespaces --all --wa
it
```

2. Uninstall the Operator
3. Remove the custom resource definitions

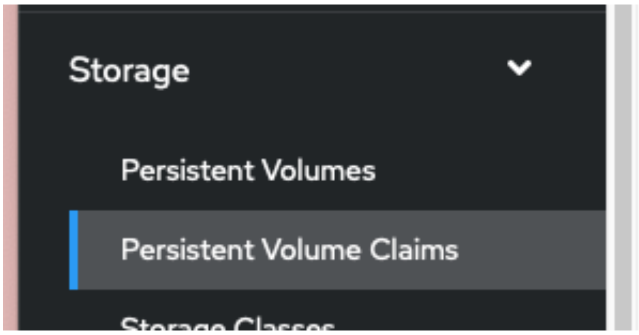
```
kubectl delete customresourcedefinitions.apiextensions.k8s.io workspacerout
ings.controller.devfile.io
kubectl delete customresourcedefinitions.apiextensions.k8s.io components.co
ntroller.devfile.io
kubectl delete customresourcedefinitions.apiextensions.k8s.io devworkspaces
```

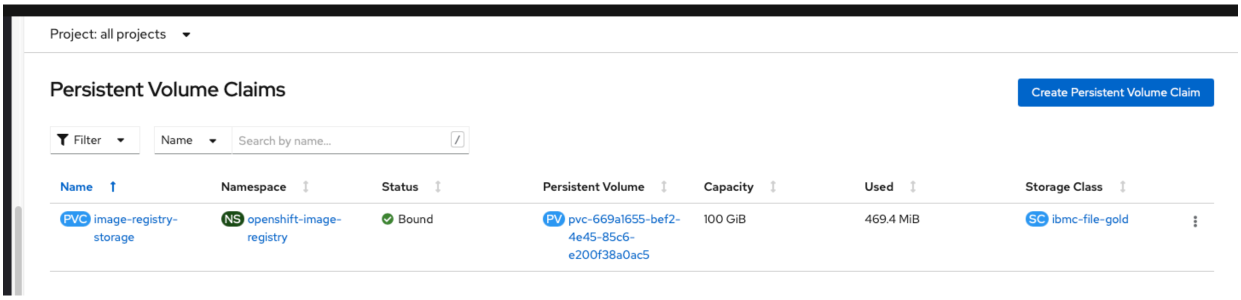
Do not install ...

Close the window and go back to the OpenShift Console

Task #14 - Storage and Networking

On the left pane, **Click on Storage > Persistent Volume Claim**

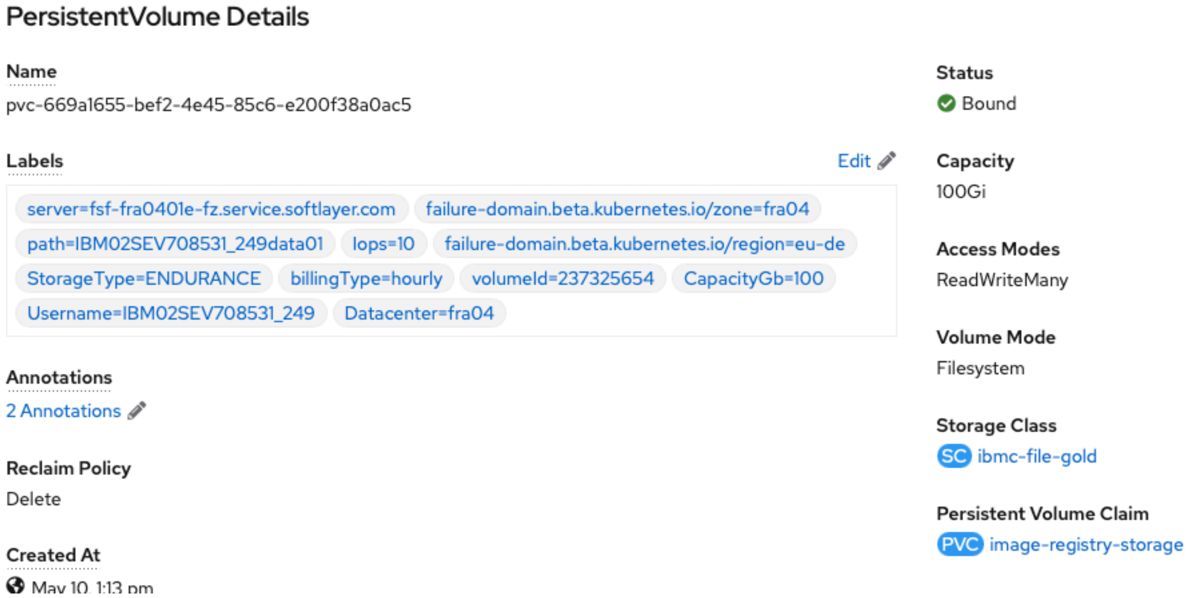




Persistent Volume Claims

Name	Namespace	Status	Persistent Volume	Capacity	Used	Storage Class
PVC image-registry-storage	NS openshift-image-registry	✓ Bound	PV pvc-669a1655-bef2-4e45-85c6-e200f38a0ac5	100 GiB	469.4 MiB	SC ibmc-file-gold

You should see the **internal OpenShift registry**. Then click on the **Persistent Volume**.



PersistentVolume Details

Name
pvc-669a1655-bef2-4e45-85c6-e200f38a0ac5

Labels

server=fsf-fra0401e-fz.service.softlayer.com
failure-domain.beta.kubernetes.io/zone=fra04
path=IBM02SEV708531_249data01
Iops=10
failure-domain.beta.kubernetes.io/region=eu-de
StorageType=ENDURANCE
billingType=hourly
volumeId=237325654
CapacityGb=100
Username=IBM02SEV708531_249
Datacenter=fra04

Annotations
2 Annotations

Reclaim Policy
Delete

Created At
May 10, 1:13 pm

Status
✓ Bound

Capacity
100Gi

Access Modes
ReadWriteMany

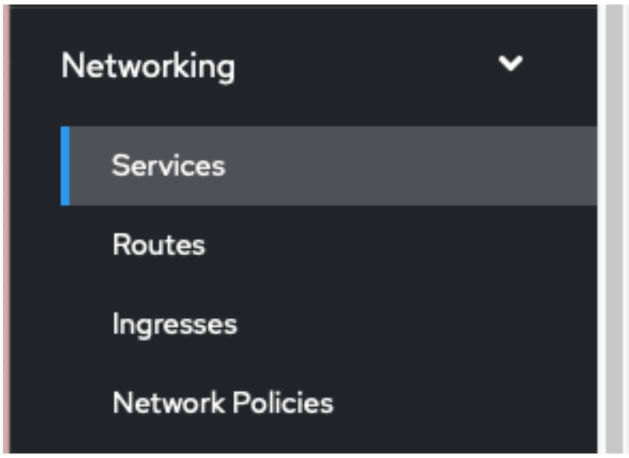
Volume Mode
Filesystem

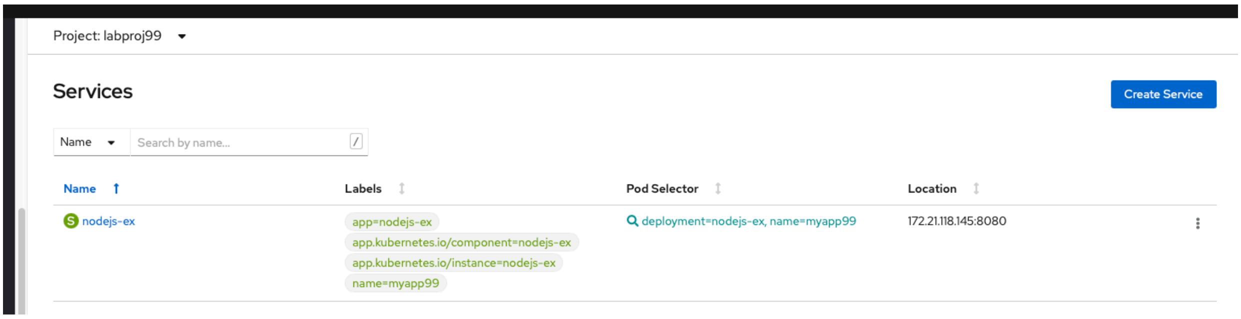
Storage Class
SC ibmc-file-gold

Persistent Volume Claim
PVC image-registry-storage

You can see the storage size by default for the internal registry : **100 Go**. This storage has been booked using a storage class (**ibmc-file-gold**). The status is `bound`.

Now move to the **Networking** section and look at you service in the **labproj**



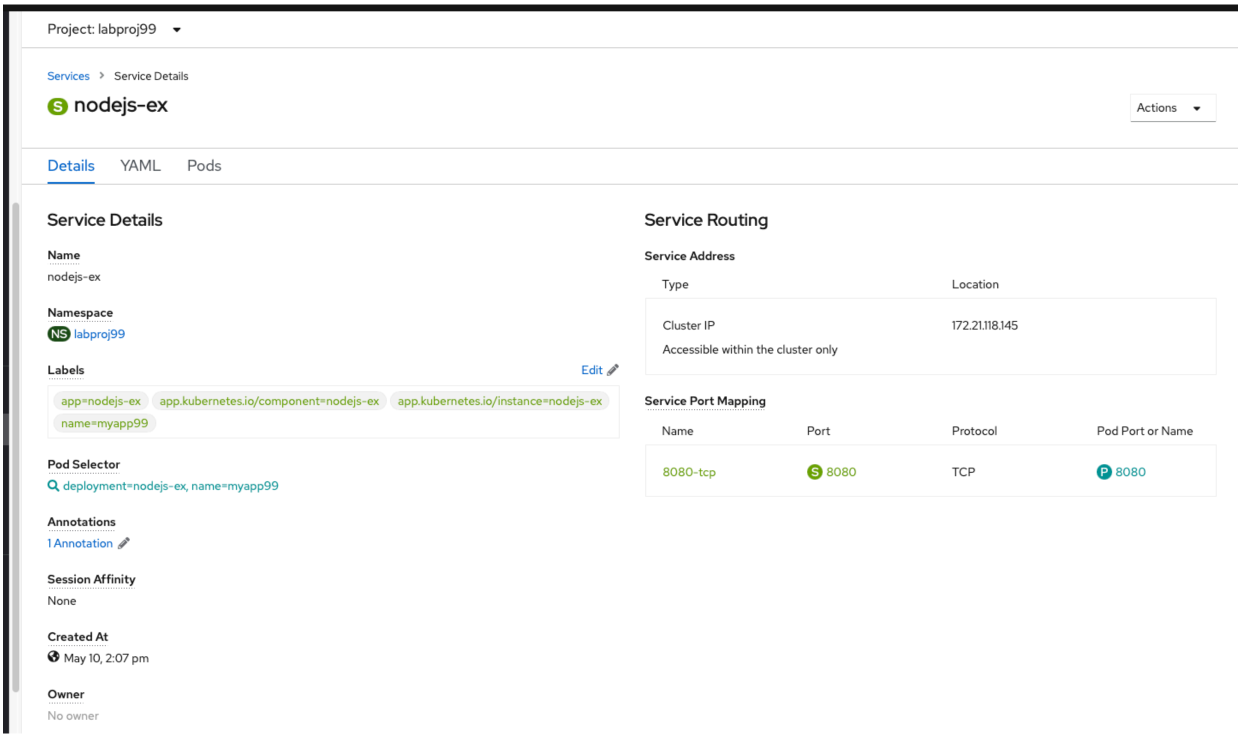


Services

Name	Labels	Pod Selector	Location

S nodejs-ex	app=nodejs-ex
	app.kubernetes.io/component=nodejs-ex
	app.kubernetes.io/instance=nodejs-ex
	name=myapp99
		deployment=nodejs-ex, name=myapp99	172.21.118.145:8080

Drill down to the **service name:**



S nodejs-ex

Service Details

Name
nodejs-ex

Namespace
NS labproj99

Labels
app=nodejs-ex app.kubernetes.io/component=nodejs-ex app.kubernetes.io/instance=nodejs-ex name=myapp99

Pod Selector
deployment=nodejs-ex, name=myapp99

Annotations
1 Annotation

Session Affinity
None

Created At
May 10, 2:07 pm

Owner
No owner

Service Routing

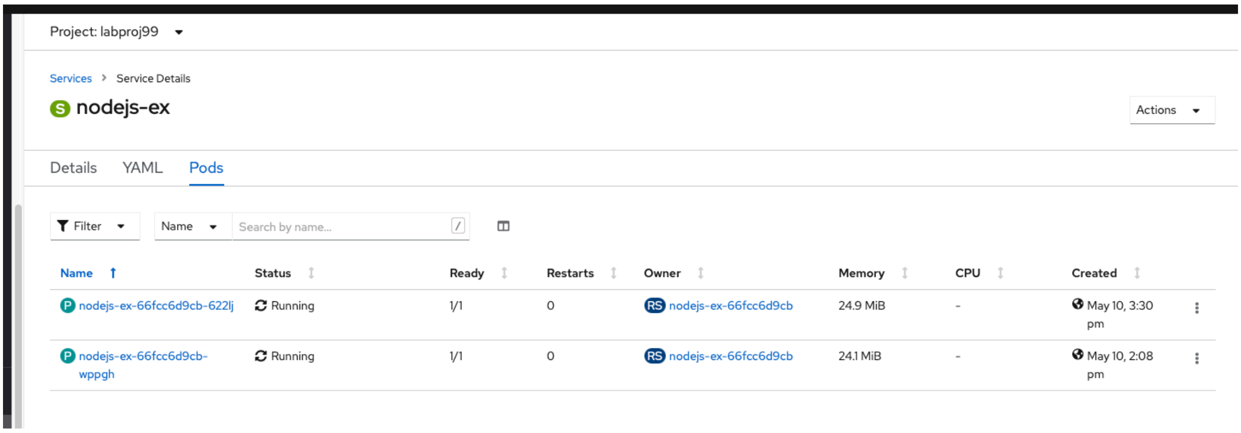
Service Address

Type	Location
Cluster IP Accessible within the cluster only	172.21.118.145

Service Port Mapping

Name	Port	Protocol	Pod Port or Name
8080-tcp	S 8080	TCP	P 8080

Drill down to the **PODs** associated to this service:



Name	Status	Ready	Restarts	Owner	Memory	CPU	Created
nodejs-ex-66fcc6d9cb-622lj	Running	1/1	0	RS nodejs-ex-66fcc6d9cb	24.9 MiB	-	May 10, 3:30 pm
nodejs-ex-66fcc6d9cb-wppgh	Running	1/1	0	RS nodejs-ex-66fcc6d9cb	24.1 MiB	-	May 10, 2:08 pm

Conclusion

Congrats !!! You successfully used the console to manage your cluster !!

You noticed the following details:

- the console facilitates the administrator work
- it is easy to change the cluster to a new version
- it is easy to drill down to troubleshoot any issue by looking at the event or at the logs
- you can even go inside a container easily
- the monitoring tool is everywhere in the console and you can also go in Grafana.
- Operators facilitate the installation of a lot of different opensource software
- Everything can be done thru the console

End of Lab