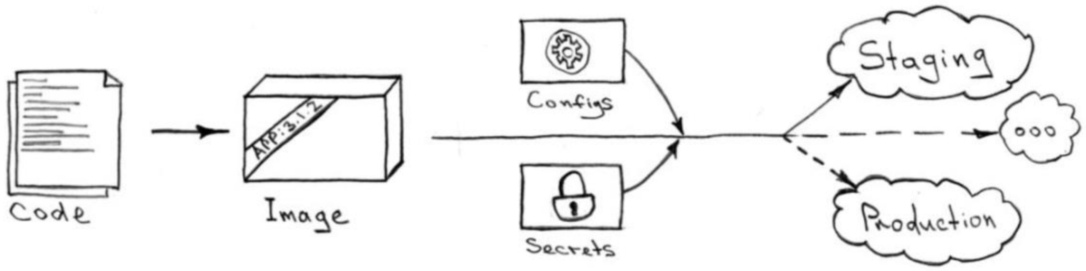
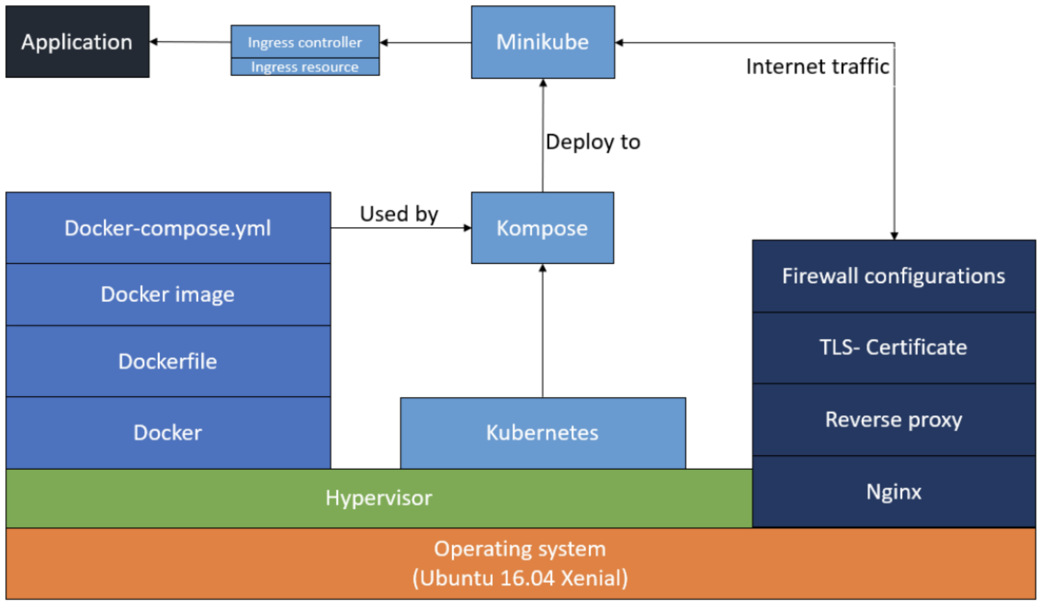
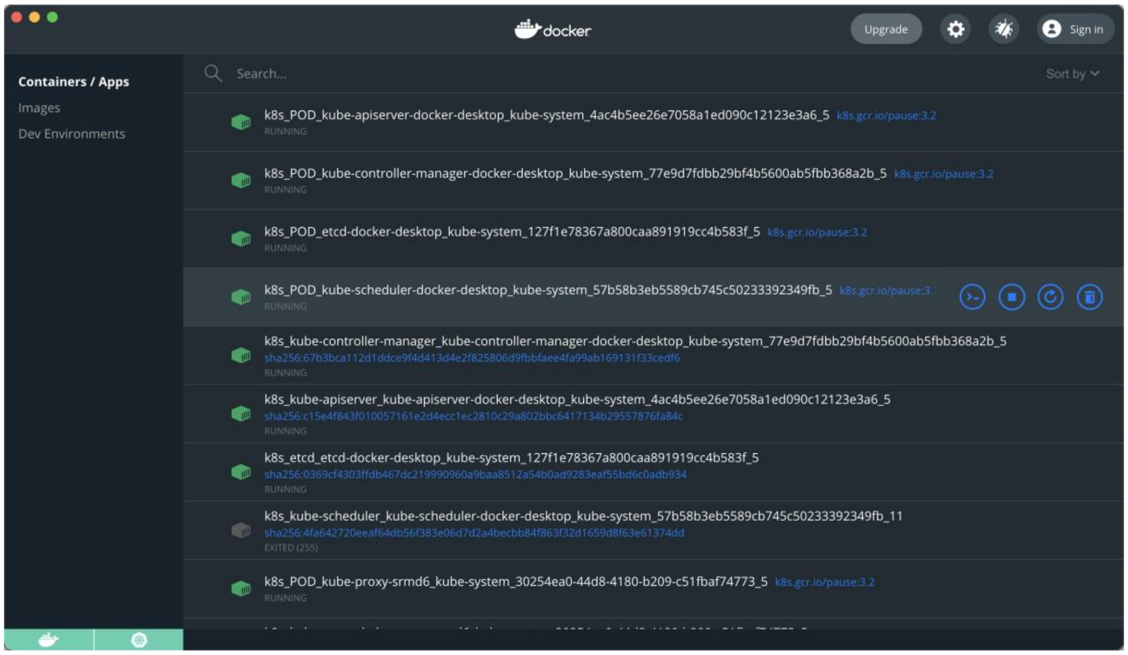

Code
Image
APP:3.1.2
Configs
Secrets
Staging
ooo
Production


Application
Ingress controller
Ingress resource
Minikube
Internet traffic
Deploy to
Docker-compose.yml
Used by
Kompose
Docker image
Dockerfile
Docker
Kubernetes
Firewall configurations
TLS- Certificate
Reverse proxy
Nginx
Hypervisor
Operating system
(Ubuntu 16.04 Xenial)



```
k8s_POD_kube-apiserver-docker-desktop_kube-system_4ac4b5ee26e7058a1ed090c12123e3a6_5  k8s.gcr.io/pause:3.2
RUNNING
k8s_POD_kube-controller-manager-docker-desktop_kube-system_77e9d7fdbb29bf4b5600ab5fbb368a2b_5  k8s.gcr.io/pause:3.2
RUNNING
k8s_POD_etcd-docker-desktop_kube-system_127f1e78367a800caa891919cc4b583f_5  k8s.gcr.io/pause:3.2
RUNNING
k8s_POD_kube-scheduler-docker-desktop_kube-system_57b58b3eb5589cb745c50233392349fb_5  k8s.gcr.io/pause:3..
RUNNING
k8s_kube-controller-manager_kube-controller-manager-docker-desktop_kube-system_77e9d7fdbb29bf4b5600ab5fbb368a2b_5
sha256:67b3bca112d1ddce9f4d413d4e2f825806d9fbbfaee4fa99ab169131f33cedf6
RUNNING
k8s_kube-apiserver_kube-apiserver-docker-desktop_kube-system_4ac4b5ee26e7058a1ed090c12123e3a6_5
sha256:c15e4f843f010057161e2d4ecc1ec2810c29a802bbc6417134b29557876fa84c
RUNNING
k8s_etcd_etcd-docker-desktop_kube-system_127f1e78367a800caa891919cc4b583f_5
sha256:0369cf4303ffdb467dc219990960a9baa8512a54b0ad9283eaf55bd6c0adb934
RUNNING
k8s_kube-scheduler_kube-scheduler-docker-desktop_kube-system_57b58b3eb5589cb745c50233392349fb_11
sha256:4fa642720eeaf64db56f383e06d7d2a4becbb84f863f32d1659d8f63e61374dd
EXITED (255)
k8s_POD_kube-proxy-srmd6_kube-system_30254ea0-44d8-4180-b209-c51fbaf74773_5  k8s.gcr.io/pause:3.2
RUNNING
```

a. settings

```
# firewall:
$ systemctl stop firewalld
$ systemctl disable firewalld

# selinux:
$ sed -i 's/enforcing/disabled/' /etc/selinux/config
$ setenforce 0

# swap:
$ vim /etc/fstab

# hostname and IP
$ vim /etc/sysconfig/network
HOSTNAME=k8s-master
$ vim /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.local
domain4
::1         localhost localhost.localdomain localhost6 localhost6.local
domain6
10.211.55.5     k8s-master
10.211.55.6     k8s-node01
10.211.55.7     k8s-node02

# IPv4 to iptables
$ cat > /etc/sysctl.d/k8s.conf << EOF
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
EOF
$ sysctl --system
```

b. Docker/kubeadm/kubelet

1) install Docker

```
$ yum install -y yum-utils device-mapper-persistent-data lvm2

## Add the Docker repository
$ yum-config-manager --add-repo https://mirrors.aliyun.com/docker-ce/li
nux/centos/docker-ce.repo

# Install Docker CE
$ yum update -y && yum install -y   containerd.io-1.2.13   docker-ce-19
.03.11   docker-ce-cli-19.03.11

$ mkdir -p /etc/systemd/system/docker.service.d
# Restart Docker
$ systemctl daemon-reload
$ systemctl restart docker
$ systemctl enable docker.service
$ echo "net.ipv4.ip_forward = 1" >> /etc/sysctl.conf
$ sysctl -p
```

2) install kubeadm, kubelet and kubectl

```
$ yum install -y kubelet-1.18.8 kubeadm-1.18.8 kubectl-1.18.8 --disable
excludes=kubernetes

$ systemctl enable --now kubelet
$ systemctl restart kubelet
$ yum install bash-completion -y
$ source /usr/share/bash-completion/bash_completion
```

3. deploy Kubernetes Master

```
$ kubectl completion bash > /etc/profile.d/kubectl.sh
$ git clone https://github.com/AliyunContainerService/k8s-for-docker-de
sktop.git cd k8s-for-docker-desktop
$ yum -y install git
$ git clone https://github.com/AliyunContainerService/k8s-for-docker-de
sktop.git;cd k8s-for-docker-desktop
$ git checkout v1.18.8
$ ./load_images.sh
$ docker images
$ kubeadm init --apiserver-advertise-address=10.211.55.5 --image-reposi
tory registry.aliyuncs.com/google_containers --kubernetes-version v1.18
.8 --service-cidr=10.1.0.0/16 --pod-network-cidr=10.244.0.0/16
```

4. Pod CNI install

```
$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/a70
459be0084506e4ec919aa1c114638878db11b/Documentation/kube-flannel.yml
```

5. join Kubernetes Node

```
$  kubeadm join 10.211.55.5:6443 --token j8abi1.aemvzybz4lgz1kyi --disc
overy-token-ca-cert-hash sha256:9cfdd61328f34c94d5342f892394dd55cf2b94a
68252061133b9fc3423a4ef80
```

6. test Kubernetes cluster

Create a nginx pod to test.

```
$ vim nginx-deployment.yaml
apiVersion: apps/v1
kind: Deployment #type : Deployment
metadata:
  name: nginx-deployment #name of Deployment
  labels:
    app: nginx #value is the label of nginx
spec:
  replicas: 1 #create an instance
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers: #generate a container
      - name: nginx #name of the container
        image: nginx:1.7.9 #use image: nginx:1.7.9 to create the contai
ner


$ kubectl apply -f nginx-deployment.yaml
$ kubectl get pods -A
$ kubectl get deployments
$ vim nginx-service.yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx-service #Service name
  labels:      #Service label
    app: nginx
spec:
  selector:
    app: nginx #choose the app with the label: nginx Pod
  ports:
```

```
  - name: nginx-port #name of the port
    protocol: TCP     #protocal type: TCP/UDP
    port: 80         #container group in the cluster can through 80 por
t to access the Service
    nodePort: 32600   #any node, through 32600 port to access Service
    targetPort: 80 #transfer to Pod’s 80 port
  type: NodePort #Serivetype，ClusterIP/NodePort/LoaderBalancer
$ kubectl apply -f nginx-service.yaml
```

7. deploy Dashboard

```
$ kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboa
rd/v2.2.0/aio/deploy/recommended.yaml

$ kubectl apply -f kubernetes-dashboard.yaml
$ kubectl get pods,svc -n kube-system
```

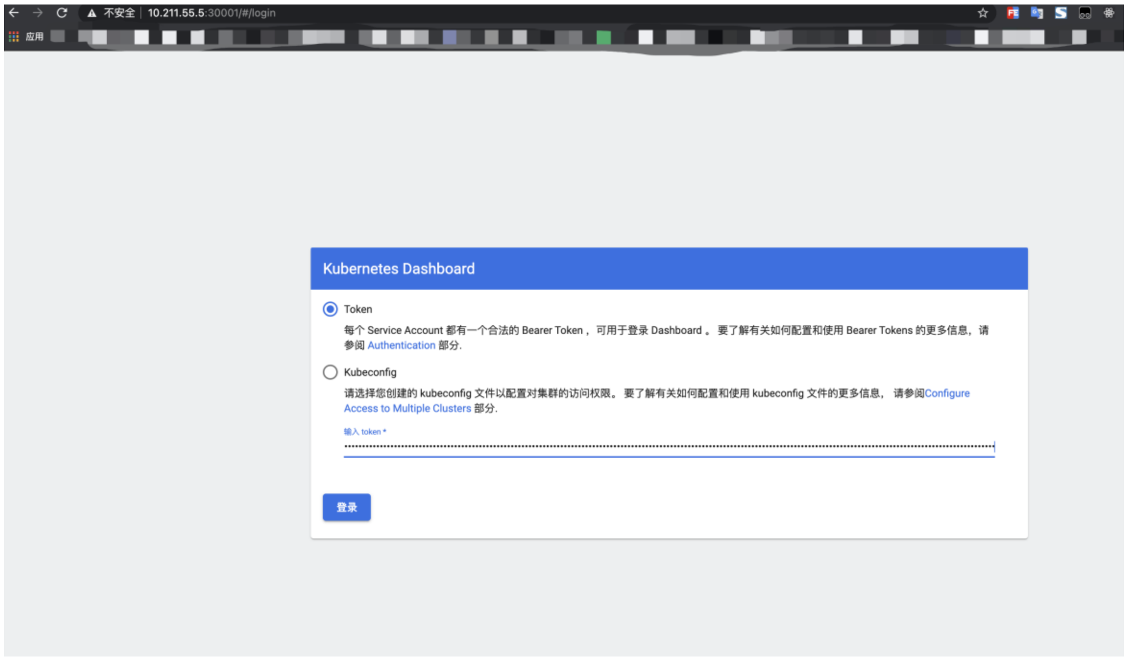
Create the service accountand bind to cluster-admin:

```
$ kubectl create serviceaccount dashboard-admin -n kube-system
$ kubectl create clusterrolebinding dashboard-admin --clusterrole=clust
er-admin --serviceaccount=kube-system:dashboard-admin
$ kubectl describe secrets -n kube-system $(kubectl -n kube-system get
secret | awk '/dashboard-admin/{print $1}')
Name:         dashboard-admin-token-7tmcs
Namespace:    kube-system
Labels:       <none>
Annotations:  kubernetes.io/service-account.name: dashboard-admin
              kubernetes.io/service-account.uid: 394e596f-d740-46bc-830
f-f803987e3180

Type:  kubernetes.io/service-account-token

Data
====
token:      eyJhbGciOiJSUzI1NiIsImtpZCI6ImxTWGtNWEE4Tm1PaFJlZ0NsMjJseV9
WZHZ0d0lwcEVob2hvcVJRQjFnV1EifQ.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2N
vdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJrdWJlLX
N5c3RlbSIsImt1YmVybmV0ZXMuaW8vc2VydmljZWFjY291bnQvc2VjcmV0Lm5hbWUiOiJkY
XNoYm9hcmQtYWRtaW4tdG9rZW4tN3RtY3MiLCJrdWJlcm5ldGVzLmlvL3NlcnZpY2VhY2Nv
dW50L3NlcnZpY2UtYWNjb3VudC5uYW1lIjoiZGFzaGJvYXJkLWFkbWluIiwia3ViZXJuZXR
lcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQudWlkIjoiMzk0ZTU5NmYtZD
c0MC00NmJjLTgzMGYtZjgwMzk4N2UzMTgwIiwic3ViIjoic3lzdGVtOnNlcnZpY2VhY2Nvd
W50Omt1YmUtc3lzdGVtOmRhc2hib2FyZC1hZG1pbiJ9.Lfu9Vh6J4eS5GgwOmkPqt2ODMUO
y1_kQUuHkOi9eXcLng8_Uz-QPni_9j37G9A31Wsu36p3SRUmwc1487gaBuNxAyQqwcclNLS
dc5TzhPr39-zubat3pwvJJu7oZNJJZjxILzpXQCk_9nmCBjj18YHFoPNY5NNSv9Hne1Jm3Q
wl-wmCPp0RnRYRtSz3sBD_E3vDBMehnCyWiNTT7Sq4KOpBuNY8ky-vSQDRbsJ5nBpAsII9N
uHersu-YlJaQEsJYrIZthPp7WTwbjeFWKbE1xUT9uh_p5PJCYSCMQ8BoGIX3E4-ZxqmX2ox
VpIqEh2HqD1xG2DTZV9lS2Ouakctv2Q
ca.crt:     1025 bytes
namespace:  11 bytes
```

Use token to login


不安全 | 10.211.55.5:30001/#/login

[redacted]

Kubernetes Dashboard

[✓ Token]

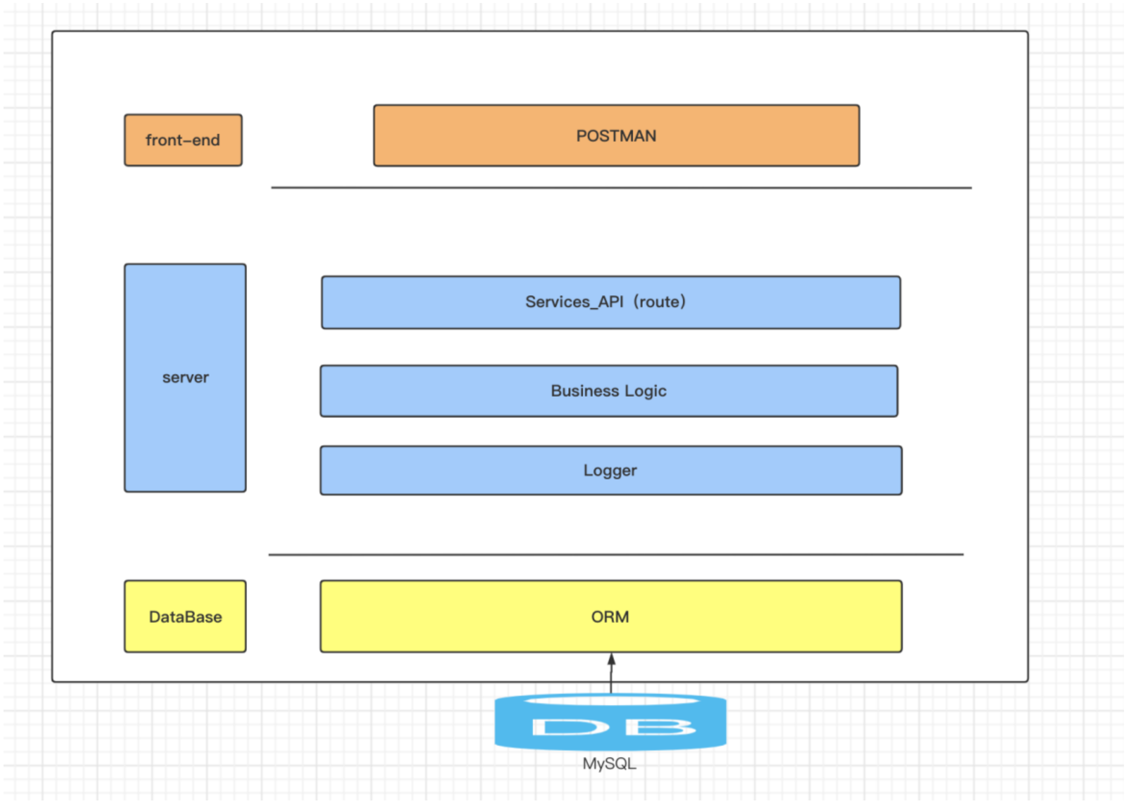
每个 Service Account 都有一个合法的 Bearer Token， 可用于登录 Dashboard 。 要了解有关如何配置和使用 Bearer Tokens 的更多信息，请参阅 Authentication 部分.

Kubeconfig

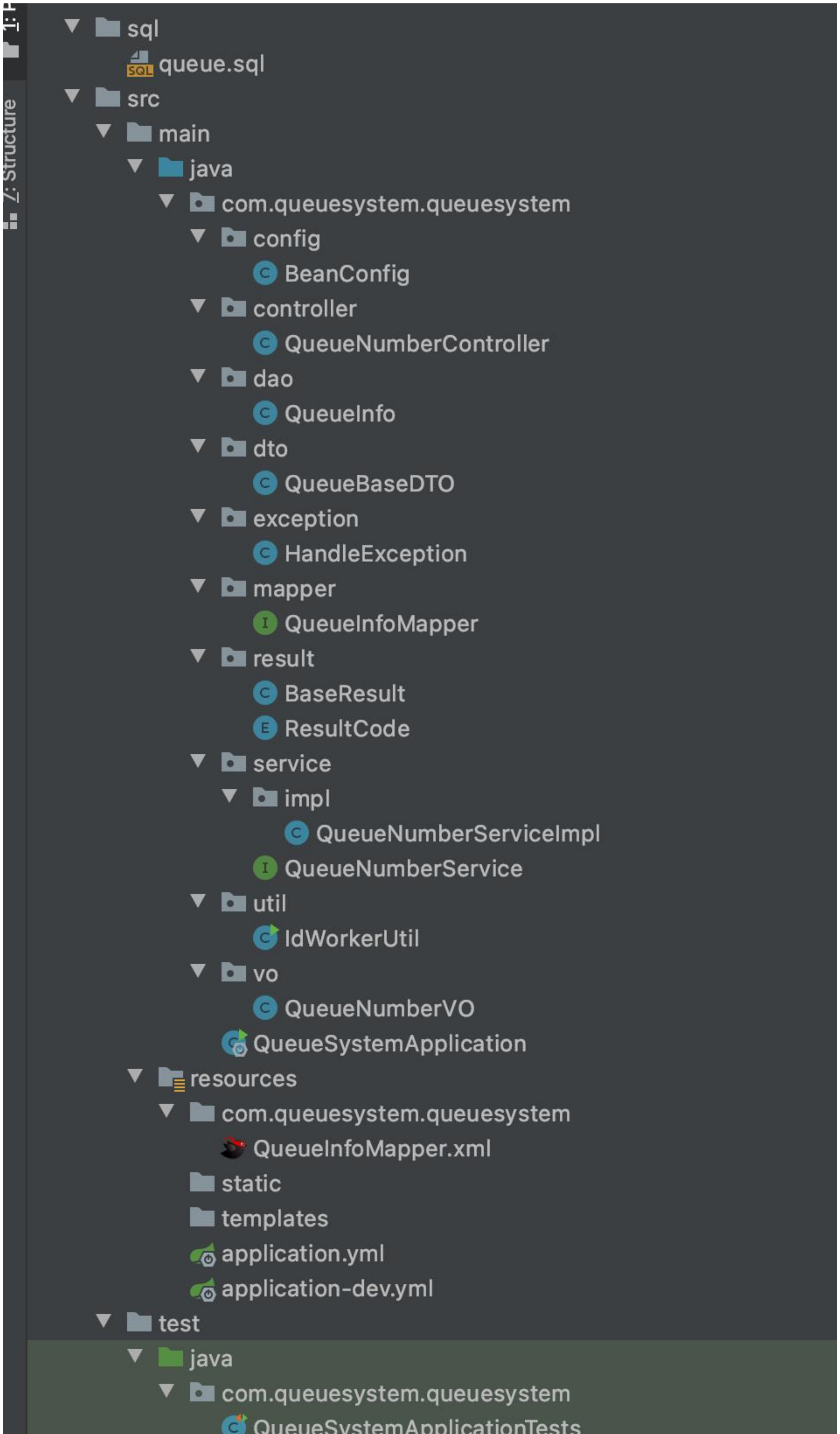
请选择您创建的 kubeconfig 文件以配置对集群的访问权限。 要了解有关如何配置和使用 kubeconfig 文件的更多信息， 请参阅Configure Access to Multiple Clusters 部分.

输入 token *

The framework

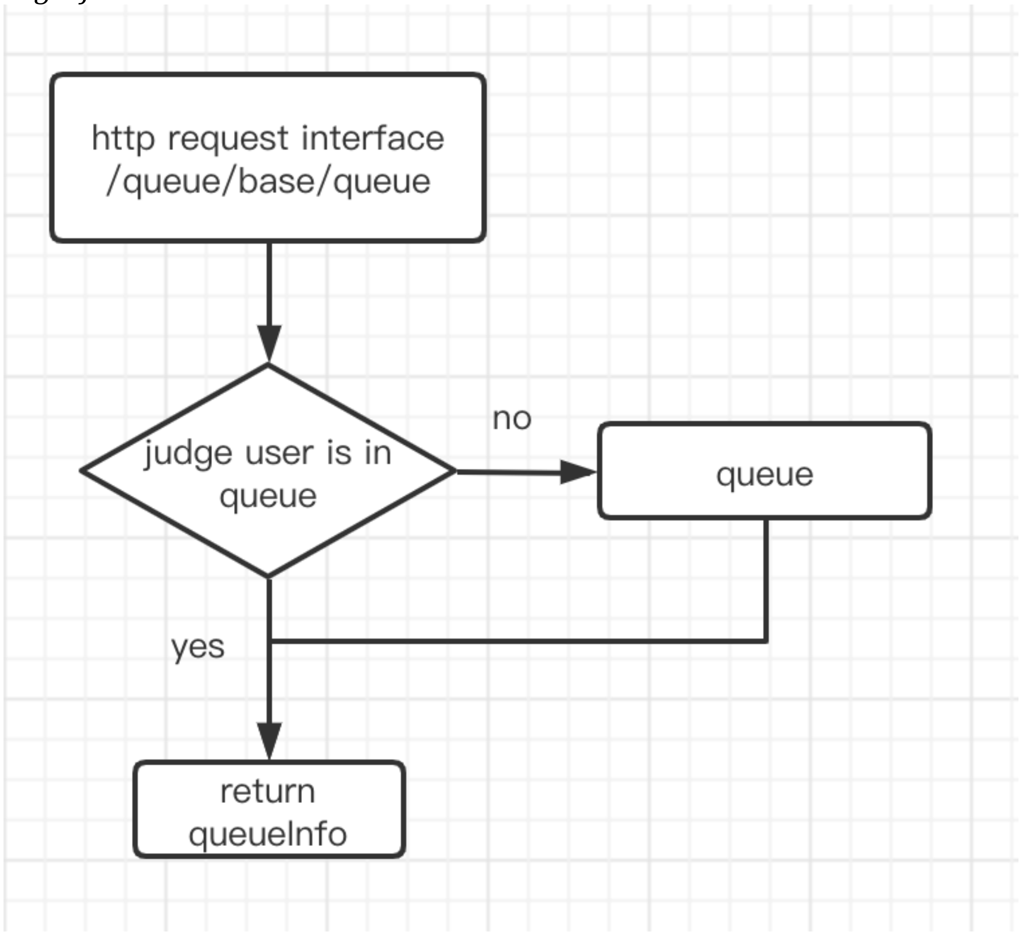


Application structure



```
▼ sql
    queue.sql
▼ src
  ▼ main
    ▼ java
      ▼ com.queuesystem.queuesystem
        ▼ config
            BeanConfig
        ▼ controller
            QueueNumberController
        ▼ dao
            QueueInfo
        ▼ dto
            QueueBaseDTO
        ▼ exception
            HandleException
        ▼ mapper
            QueueInfoMapper
        ▼ result
            BaseResult
            ResultCode
        ▼ service
          ▼ impl
              QueueNumberServiceImpl
            QueueNumberService
        ▼ util
            IdWorkerUtil
        ▼ vo
            QueueNumberVO
          QueueSystemApplication
    ▼ resources
      ▼ com.queuesystem.queuesystem
          QueueInfoMapper.xml
        static
        templates
        application.yml
        application-dev.yml
  ▼ test
    ▼ java
      ▼ com.queuesystem.queuesystem
          QueueSystemApplicationTests
```

Logic flow chart



Use Dockerfile to construct the image:

```
FROM java:8

LABEL version="1.0"
LABEL user="HaoXu"

RUN  mkdir -p  /home/work/data/www/queue-system
WORKDIR /home/work/data/www/queue-system
COPY ./target/*.jar ./app.jar

CMD ["java", "-jar", "app.jar", "--spring.profiles.active=dev"]

EXPOSE 8080
```

To ensure we have the image:

```
[root@k8s-master ~]# docker images
REPOSITORY                                                      TAG
               IMAGE ID            CREATED                  SIZE
queue                                                           v1
               a0eaf0743686        Less than a second ago   667MB
calico/node                                                     v3.18
.1             50b52cdadbcf        4 weeks ago              172MB
calico/pod2daemon-flexvol                                       v3.18
.1             3994c62982cc        4 weeks ago              21.7MB
calico/cni                                                      v3.18
.1             21fdaa2fccee        4 weeks ago              131MB
k8s.gcr.io/metrics-server/metrics-server                        v0.4.
2              17c225a562d9        2 months ago             60.5MB
k8s.gcr.io/kube-proxy                                           v1.18
.8             0fb7201f92d0        8 months ago             117MB
registry.aliyuncs.com/google_containers/kube-proxy              v1.18
.8             0fb7201f92d0        8 months ago             117MB
k8s.gcr.io/kube-controller-manager                              v1.18
.8             6a979351fe5e        8 months ago             162MB
registry.aliyuncs.com/google_containers/kube-controller-manager   v1.18
.8             6a979351fe5e        8 months ago             162MB
k8s.gcr.io/kube-apiserver                                       v1.18
.8             92d040a0dca7        8 months ago             173MB
registry.aliyuncs.com/google_containers/kube-apiserver          v1.18
.8             92d040a0dca7        8 months ago             173MB
k8s.gcr.io/kube-scheduler                                       v1.18
.8             6f7135fb47e0        8 months ago             95.3MB
registry.aliyuncs.com/google_containers/kube-scheduler          v1.18
.8             6f7135fb47e0        8 months ago             95.3MB
k8s.gcr.io/pause                                                3.2
               80d28bedfe5d        14 months ago            683kB
registry.aliyuncs.com/google_containers/pause                   3.2
               80d28bedfe5d        14 months ago            683kB
k8s.gcr.io/coredns                                              1.6.7
               67da37a9a360        14 months ago            43.8MB
registry.aliyuncs.com/google_containers/coredns                 1.6.7
               67da37a9a360        14 months ago            43.8MB
k8s.gcr.io/etcd                                                 3.4.3
-0             303ce5db0e90        17 months ago            288MB
registry.aliyuncs.com/google_containers/etcd                    3.4.3
-0             303ce5db0e90        17 months ago            288MB
quay.io/kubernetes-ingress-controller/nginx-ingress-controller   0.26.
1              29024c9c6e70        18 months ago            483MB
mysql
```

Kubenetes manage the app

Edit queue-deployment.yaml, and Service: service.yaml then

use :`kubectl create -f queue-deployment.yaml` to do the deployment

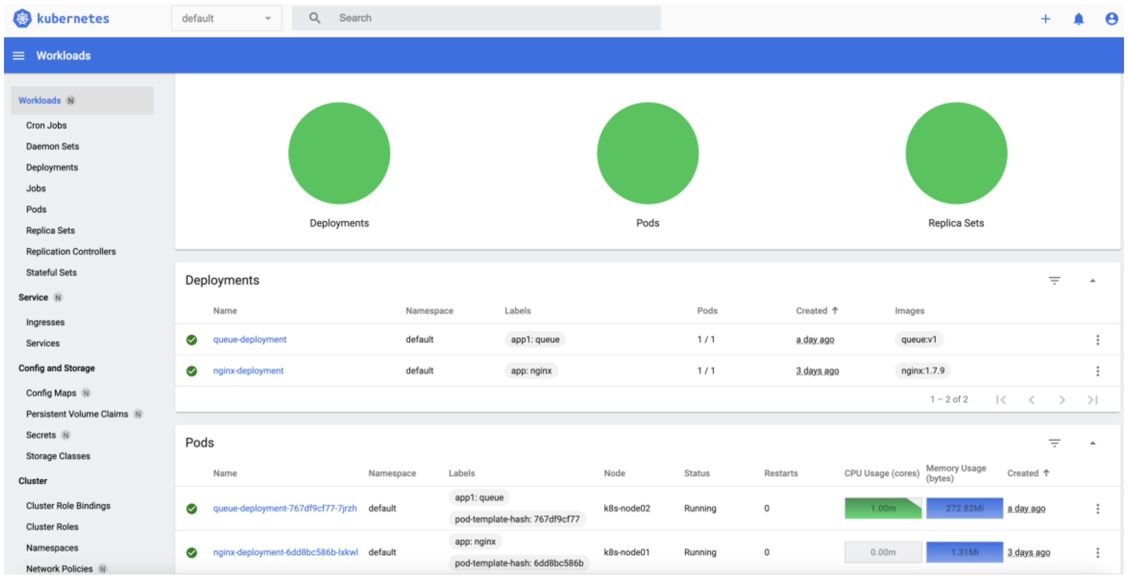
use `kubectl top pod --all-namespaces` to see the status:

```
NAMESPACE              NAME                                         CPU
(cores)   MEMORY(bytes)
default                nginx-deployment-6dd8bc586b-lxkwl            0m
          1Mi
default                queue-deployment-767df9cf77-7jrzh            2m
          428Mi
kube-system            calico-kube-controllers-65d7476764-7xjsg     1m
          13Mi
kube-system            calico-node-86l5n                            28m
          71Mi
kube-system            calico-node-dnnc8                            28m
          73Mi
kube-system            calico-node-nztzn                            28m
          59Mi
kube-system            coredns-7ff77c879f-6cthf                     2m
          9Mi
kube-system            coredns-7ff77c879f-jwn4l                     3m
          8Mi
kube-system            etcd-k8s-master                              15m
          129Mi
kube-system            kube-apiserver-k8s-master                    34m
          311Mi
kube-system            kube-controller-manager-k8s-master           14m
          40Mi
kube-system            kube-proxy-d2jnp                             1m
          18Mi
kube-system            kube-proxy-fvrnf                             1m
          12Mi
kube-system            kube-proxy-nfvlz                             1m
          18Mi
kube-system            kube-scheduler-k8s-master                    3m
          12Mi
kube-system            metrics-server-5855ddf686-qhlb6              3m
          15Mi
kubernetes-dashboard   dashboard-metrics-scraper-78f5d9f487-4pj9f   1m
          9Mi
kubernetes-dashboard   kubernetes-dashboard-577bd97bc-vng45         1m
          24Mi
```

For a better view the CPU(cores) and MEMORY(bytes)， in k8s dashboard, we deploy metrics to show them.

Demonstration of dashboard

Init_status:



Deployments

Name	Namespace	Labels	Pods	Created ↑	Images
queue-deployment	default	app1: queue	1 / 1	a day ago	queue:v1
nginx-deployment	default	app: nginx	1 / 1	3 days ago	nginx:1.7.9

1 – 2 of 2

Pods

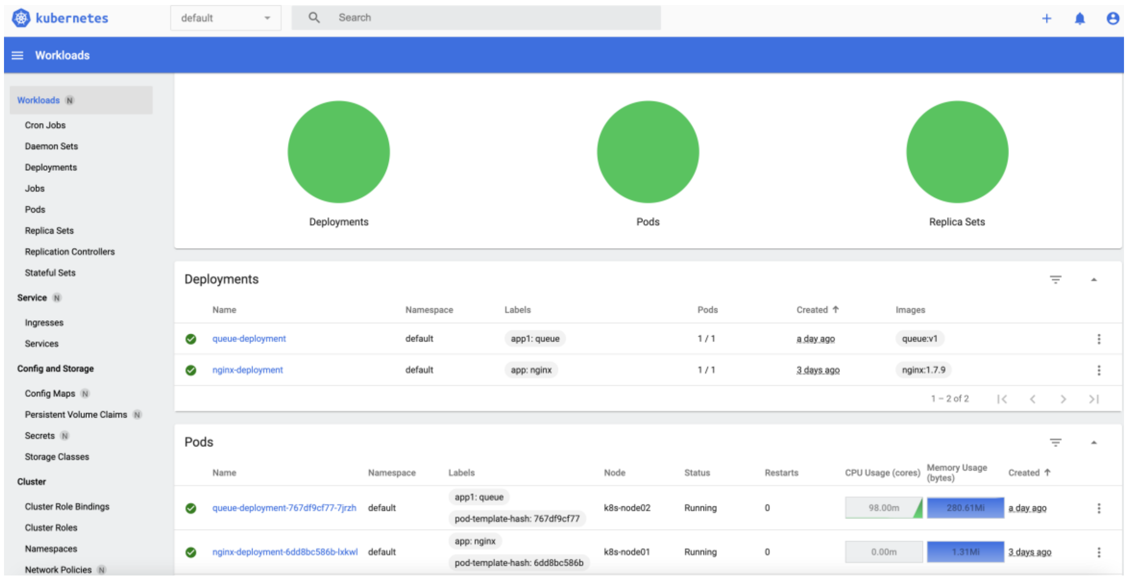
Name	Namespace	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created ↑
queue-deployment-767df9cf77-7jrzh	default	app1: queue pod-template-hash: 767df9cf77	k8s-node02	Running	0	1.00m	272.82Mi	a day ago
nginx-deployment-6dd8bc586b-lxkwl	default	app: nginx pod-template-hash: 6dd8bc586b	k8s-node01	Running	0	0.00m	1.31Mi	3 days ago

To do stress test: stress.sh

```
#!/bin/bash
while (true)
do
  curl --location --request POST '10.211.55.5:32750/queue/base/queue' -
-header 'Content-Type:application/json' --data-raw '{"firstName": "hao"
,"lastName": "xu", "userId": "6771772"}'

done
```

Run the stress.sh:



Deployments

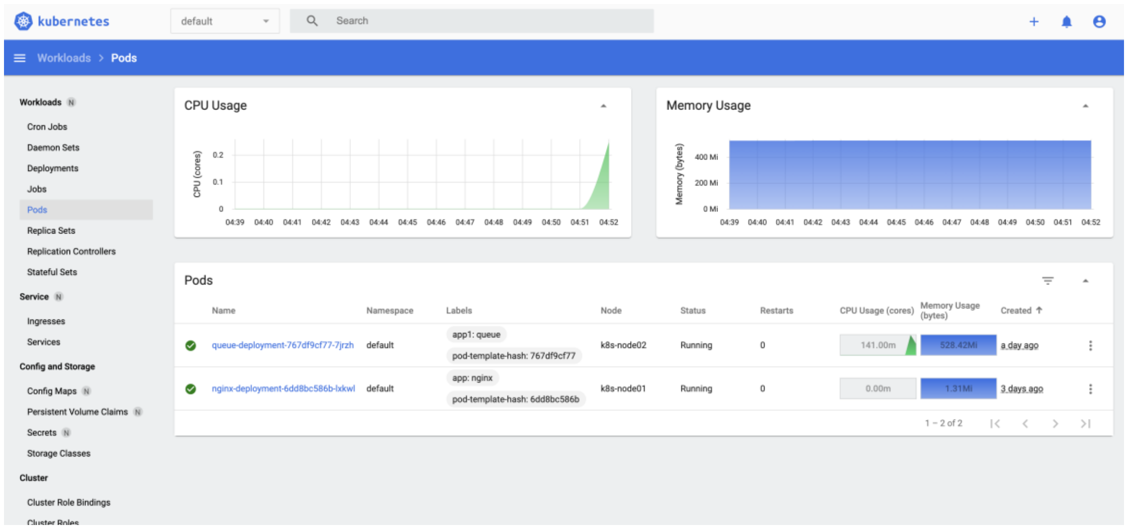
Name	Namespace	Labels	Pods	Created ↑	Images
✓ queue-deployment	default	app1: queue	1 / 1	a day ago	queue:v1
✓ nginx-deployment	default	app: nginx	1 / 1	3 days ago	nginx:1.7.9

1 – 2 of 2

Pods

Name	Namespace	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created ↑
✓ queue-deployment-767df9cf77-7jrzh	default	app1: queue pod-template-hash: 767df9cf77	k8s-node02	Running	0	98.00m	280.61Mi	a day ago
✓ nginx-deployment-6dd8bc586b-lxkwl	default	app: nginx pod-template-hash: 6dd8bc586b	k8s-node01	Running	0	0.00m	1.31Mi	3 days ago

Then,

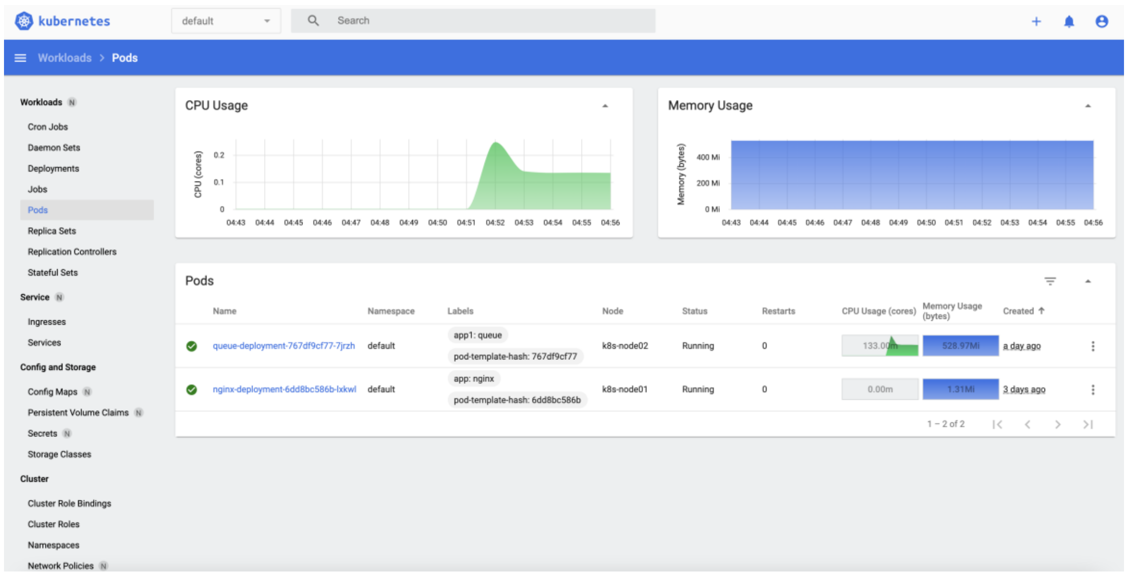


Pods

Name	Namespace	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created ↑
✓ queue-deployment-767df9cf77-7jrzh	default	app1: queue pod-template-hash: 767df9cf77	k8s-node02	Running	0	141.00m	528.42Mi	a day ago
✓ nginx-deployment-6dd8bc586b-lxkwl	default	app: nginx pod-template-hash: 6dd8bc586b	k8s-node01	Running	0	0.00m	1.31Mi	3 days ago

1 – 2 of 2

At last



	Name	Namespace	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created ↑
✓	queue-deployment-767df9cf77-7jrzh	default	app1: queue pod-template-hash: 767df9cf77	k8s-node02	Running	0	133.00m	528.97Mi	a day ago
✓	nginx-deployment-6dd8bc586b-lxkwl	default	app: nginx pod-template-hash: 6dd8bc586b	k8s-node01	Running	0	0.00m	1.31Mi	3 days ago