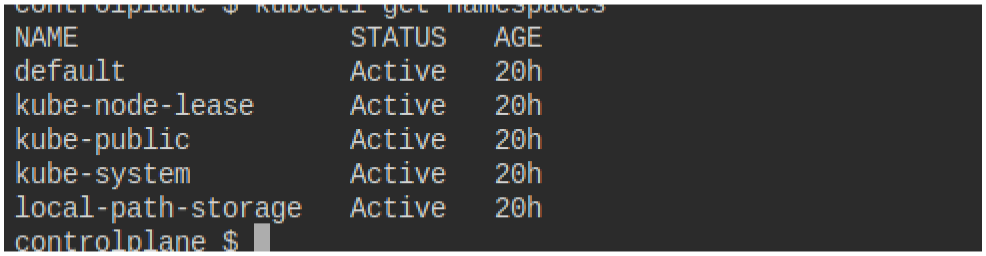
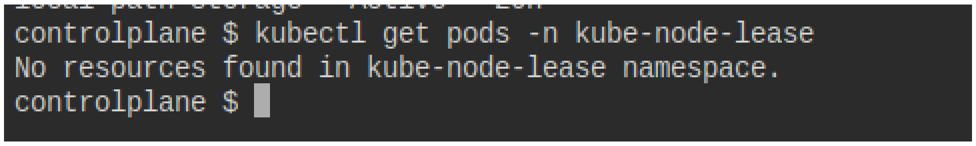
1- get namespaces of the system



```
controlplane $ kubectl get namespaces
NAME                 STATUS   AGE
default              Active   20h
kube-node-lease      Active   20h
kube-public          Active   20h
kube-system          Active   20h
local-path-storage   Active   20h
controlplane $
```

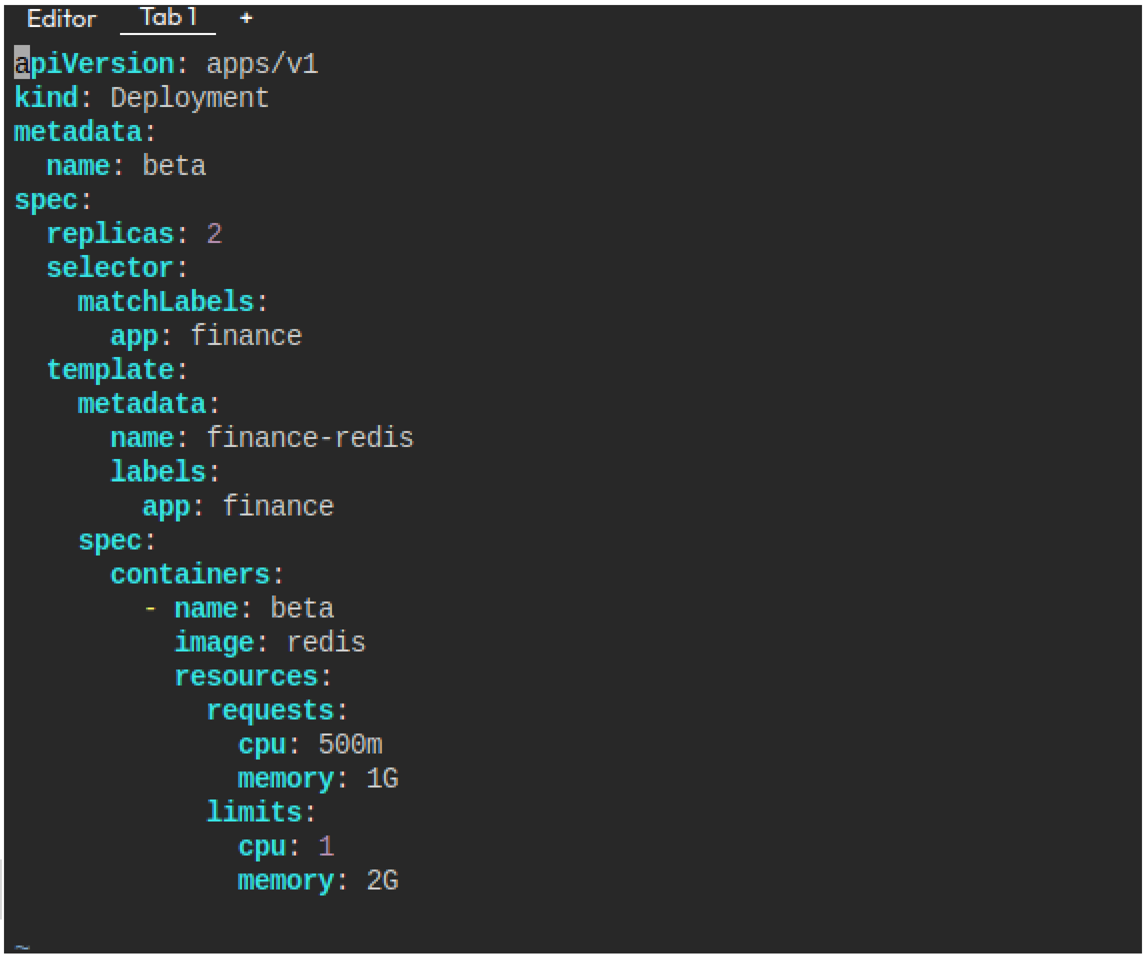
2- get amount of pods in the kube-system namespace

heck



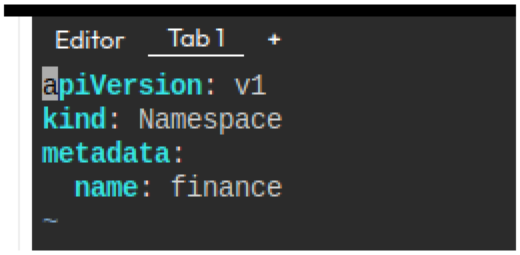
```
local-path-storage   Active   20h
controlplane $ kubectl get pods -n kube-node-lease
No resources found in kube-node-lease namespace.
controlplane $
```

3- create a deployment of 2 replicas with the name of beta



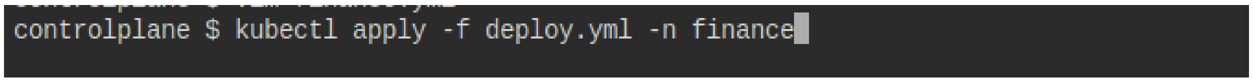
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: beta
spec:
  replicas: 2
  selector:
    matchLabels:
      app: finance
  template:
    metadata:
      name: finance-redis
      labels:
        app: finance
    spec:
      containers:
        - name: beta
          image: redis
          resources:
            requests:
              cpu: 500m
              memory: 1G
            limits:
              cpu: 1
              memory: 2G

~
```

Creating the finance namespace

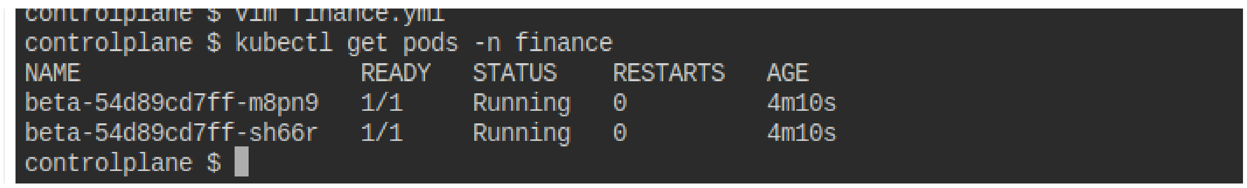


```
apiVersion: v1
kind: Namespace
metadata:
  name: finance
~
```

Apply in the finance namespace

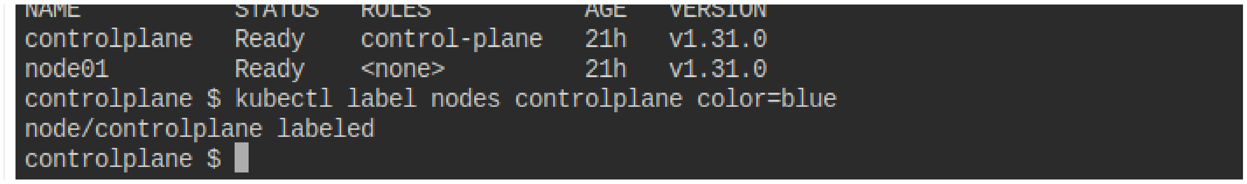


```
controlplane $ kubectl apply -f deploy.yml -n finance
```



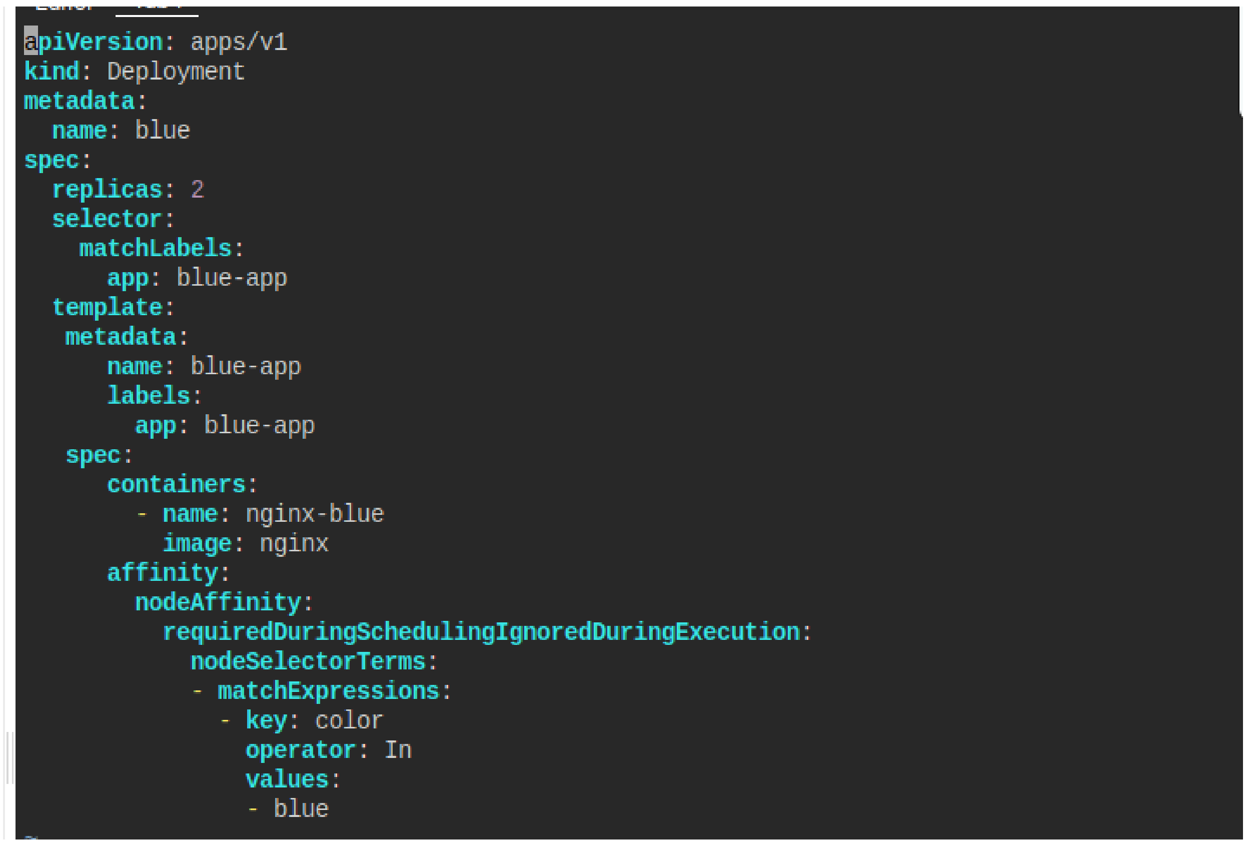
```
controlplane $ vim finance.yml
controlplane $ kubectl get pods -n finance
NAME                     READY   STATUS    RESTARTS   AGE
beta-54d89cd7ff-m8pn9    1/1     Running   0          4m10s
beta-54d89cd7ff-sh66r    1/1     Running   0          4m10s
controlplane $
```

4- assigning label blue ot master node



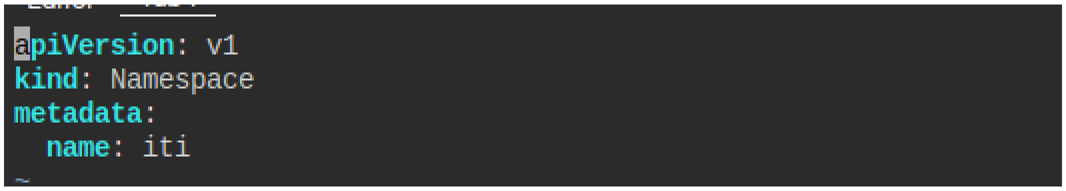
```
NAME           STATUS   ROLES           AGE   VERSION
controlplane   Ready    control-plane   21h   v1.31.0
node01         Ready    <none>          21h   v1.31.0
controlplane $ kubectl label nodes controlplane color=blue
node/controlplane labeled
controlplane $
```

5- creating deployment using node affinity on blue

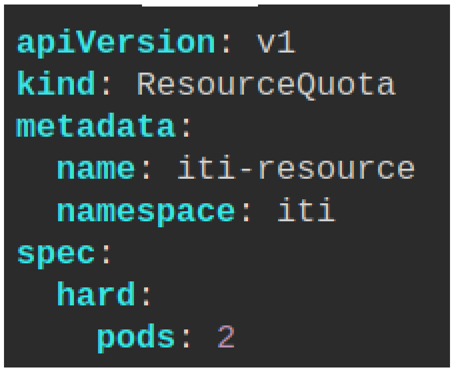


```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: blue
spec:
  replicas: 2
  selector:
    matchLabels:
      app: blue-app
  template:
   metadata:
      name: blue-app
      labels:
        app: blue-app
   spec:
      containers:
        - name: nginx-blue
          image: nginx
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
              - key: color
                operator: In
                values:
                - blue
```

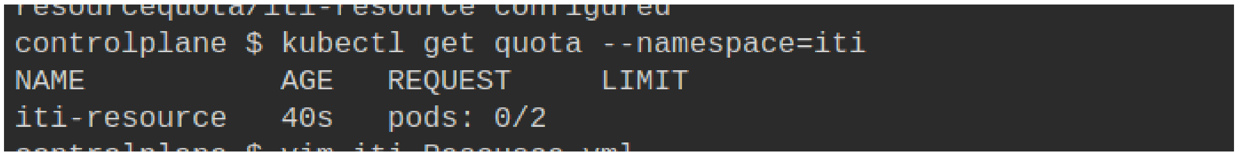
6-creating iti namespace



```yaml
apiVersion: v1
kind: Namespace
metadata:
  name: iti
```

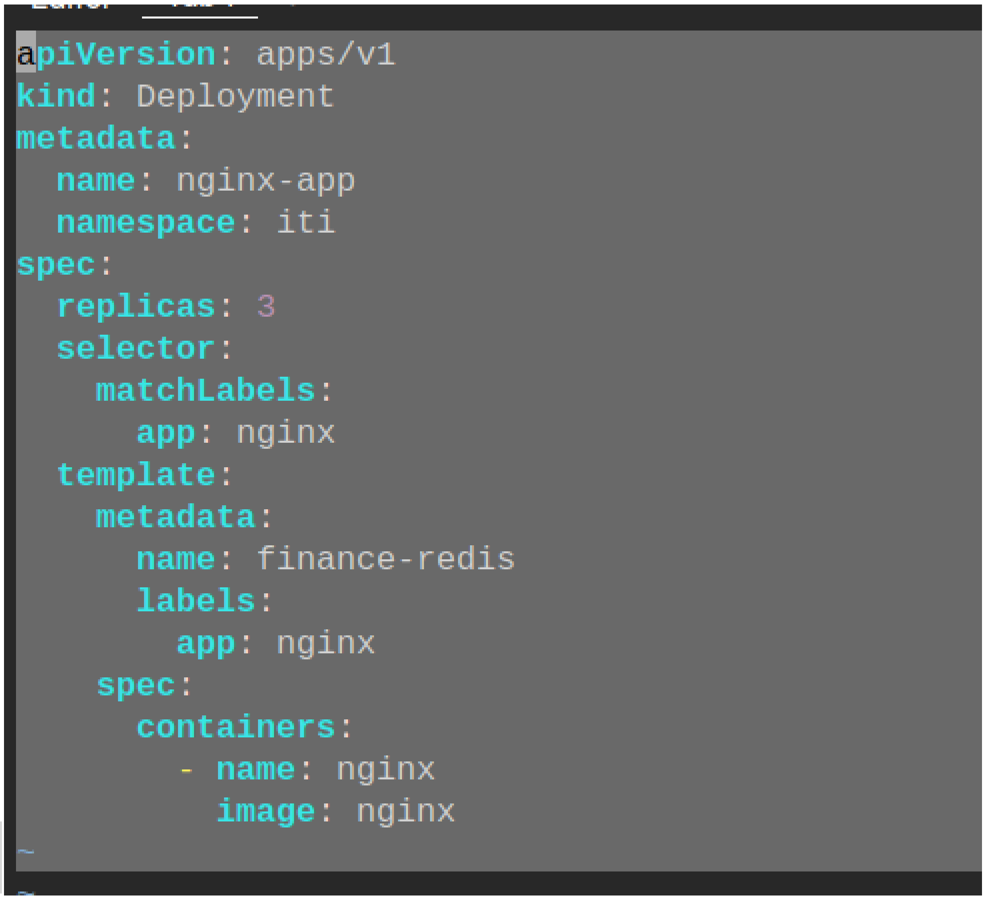


```yaml
apiVersion: v1
kind: ResourceQuota
metadata:
  name: iti-resource
  namespace: iti
spec:
  hard:
    pods: 2
```



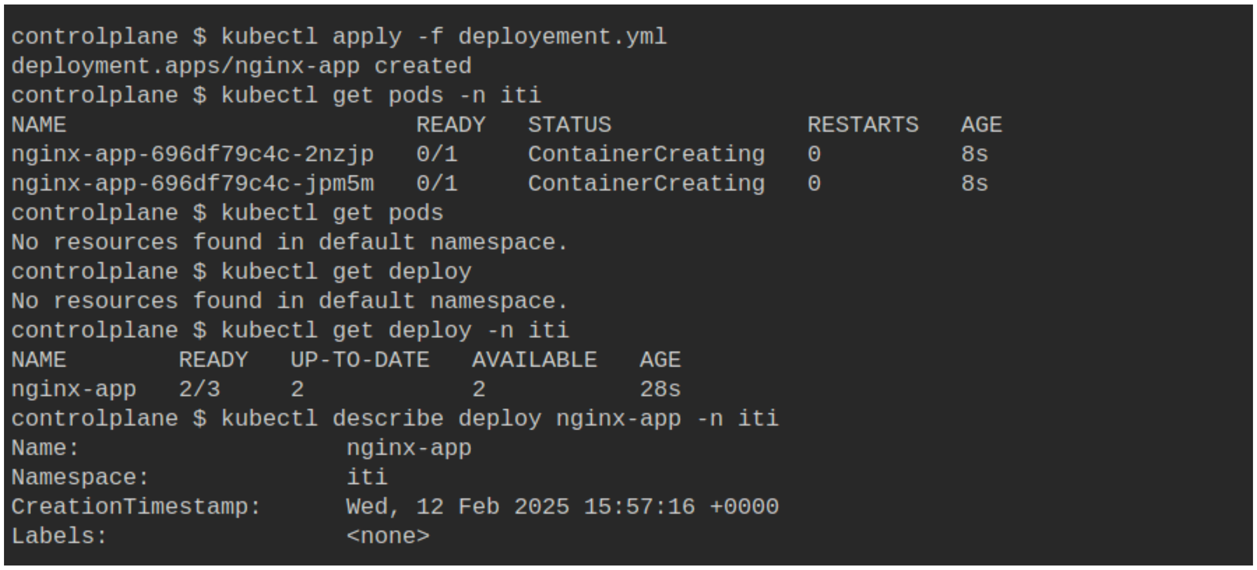
```
resourcequota/iti-resource configured
controlplane $ kubectl get quota --namespace=iti
NAME           AGE   REQUEST     LIMIT
iti-resource   40s   pods: 0/2
```

7- creating nginx deployment with 3 replicas



```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-app
  namespace: iti
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      name: finance-redis
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx
```

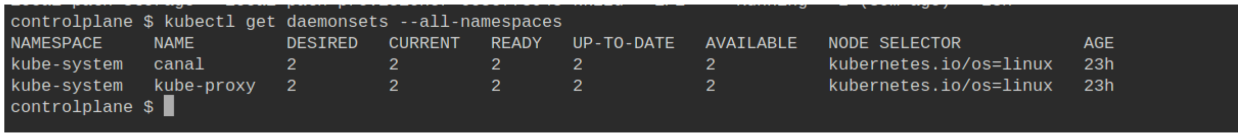
8- how many pods were created and why



```
controlplane $ kubectl apply -f deployement.yml
deployment.apps/nginx-app created
controlplane $ kubectl get pods -n iti
NAME                         READY   STATUS              RESTARTS   AGE
nginx-app-696df79c4c-2nzjp   0/1     ContainerCreating   0          8s
nginx-app-696df79c4c-jpm5m   0/1     ContainerCreating   0          8s
controlplane $ kubectl get pods
No resources found in default namespace.
controlplane $ kubectl get deploy
No resources found in default namespace.
controlplane $ kubectl get deploy -n iti
NAME        READY   UP-TO-DATE   AVAILABLE   AGE
nginx-app   2/3     2            2           28s
controlplane $ kubectl describe deploy nginx-app -n iti
Name:                   nginx-app
Namespace:              iti
CreationTimestamp:      Wed, 12 Feb 2025 15:57:16 +0000
Labels:                 <none>
```

2 pods were created instead of the 3 specified by the deployment because the resource quota hard limit specified for the namespace are 2 pods only

9- get daemonsets on all namespaces

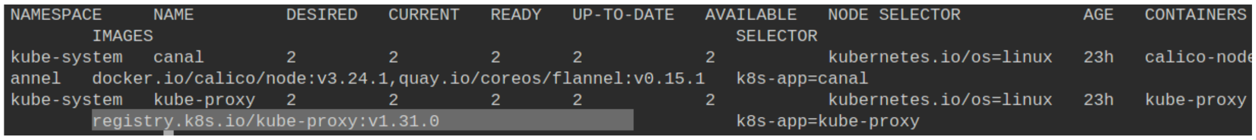


```
controlplane $ kubectl get daemonsets --all-namespaces
NAMESPACE     NAME         DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR            AGE
kube-system   canal        2         2         2       2            2           kubernetes.io/os=linux   23h
kube-system   kube-proxy   2         2         2       2            2           kubernetes.io/os=linux   23h
controlplane $
```

10- what daemon sets exist on the kube-system

The canal daemon-set and the kube-proxy exists on all nodes

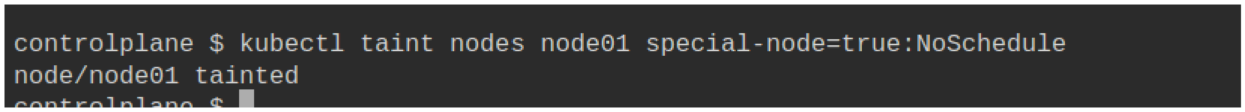
11- the image used by the kube-proxy daemonset



```
NAMESPACE     NAME         DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR            AGE   CONTAINERS
      IMAGES                                                                    SELECTOR
kube-system   canal        2         2         2       2            2           kubernetes.io/os=linux   23h   calico-node
annel   docker.io/calico/node:v3.24.1,quay.io/coreos/flannel:v0.15.1   k8s-app=canal
kube-system   kube-proxy   2         2         2       2            2           kubernetes.io/os=linux   23h   kube-proxy
      registry.k8s.io/kube-proxy:v1.31.0                                  k8s-app=kube-proxy
```

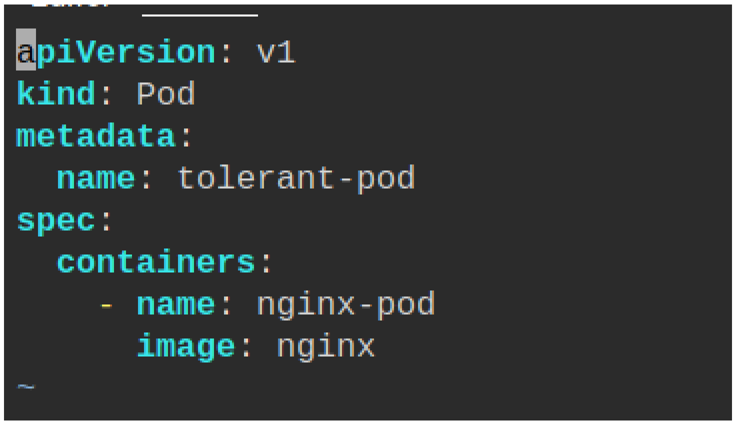
registry.k8s.io/kube-proxy:v1.31.0

12- taint node 1



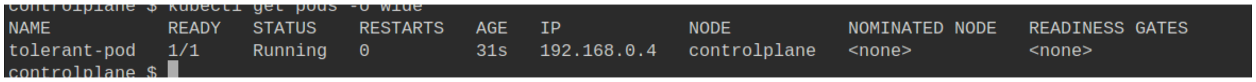
```
controlplane $ kubectl taint nodes node01 special-node=true:NoSchedule
node/node01 tainted
controlplane $
```

13- create a pod called tolerant pod



```yaml
apiVersion: v1
kind: Pod
metadata:
  name: tolerant-pod
spec:
  containers:
    - name: nginx-pod
      image: nginx
~
```

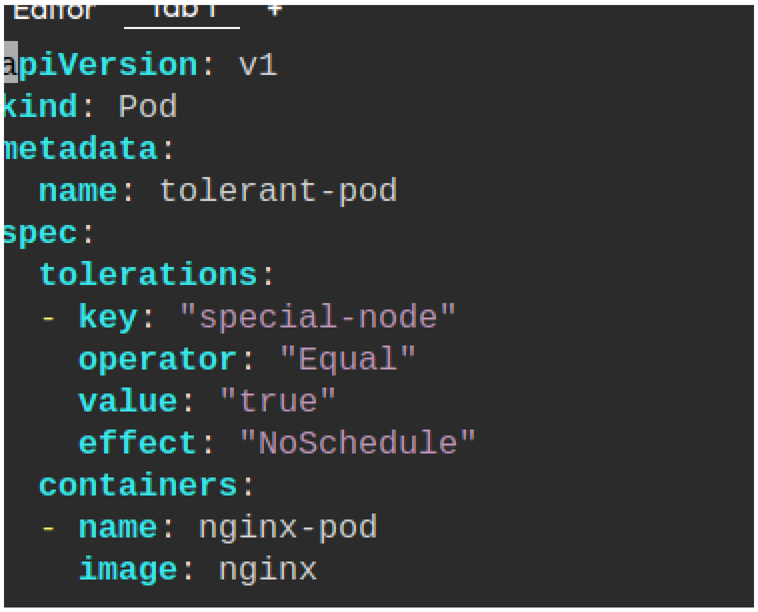
14- which node is it deployed on?



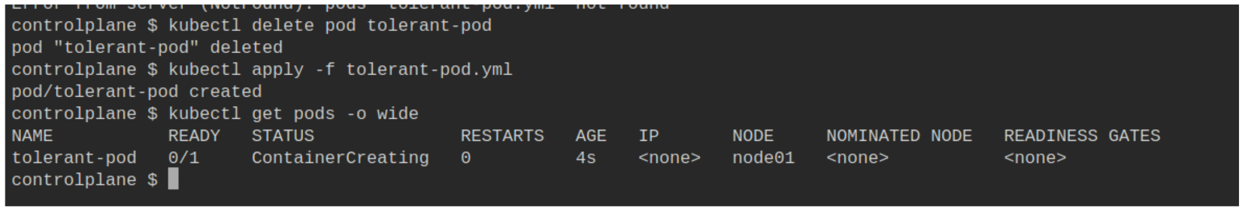
```
controlplane $ kubectl get pods -o wide
NAME           READY   STATUS    RESTARTS   AGE   IP            NODE           NOMINATED NODE   READINESS GATES
tolerant-pod   1/1     Running   0          31s   192.168.0.4   controlplane   <none>           <none>
controlplane $
```

Deployed on the controlplane node because the pod doesnt have a toleration that matches node1 so its deployed on the controlplane node

15- deleting the pod and rescheduling



```yaml
apiVersion: v1
kind: Pod
metadata:
  name: tolerant-pod
spec:
  tolerations:
  - key: "special-node"
    operator: "Equal"
    value: "true"
    effect: "NoSchedule"
  containers:
  - name: nginx-pod
    image: nginx
```



```
Error from server (NotFound): pods "tolerant-pod.yml" not found
controlplane $ kubectl delete pod tolerant-pod
pod "tolerant-pod" deleted
controlplane $ kubectl apply -f tolerant-pod.yml
pod/tolerant-pod created
controlplane $ kubectl get pods -o wide
NAME           READY   STATUS              RESTARTS   AGE   IP       NODE     NOMINATED NODE   READINESS GATES
tolerant-pod   0/1     ContainerCreating   0          4s    <none>   node01   <none>           <none>
controlplane $
```

16- on which node is it scheduled

The pod is scheduled on the node01 because its tolerations now the node01 which are special-node=true:NoSchedule