**Trabajar de forma declarativa**

apiVersion: v1

kind: Pod

metadata:

name: nginx

labels:

zone: prod

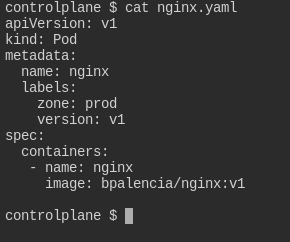
version: v1

spec:

containers:

- name: nginx

image: bpalencia/nginx:v1

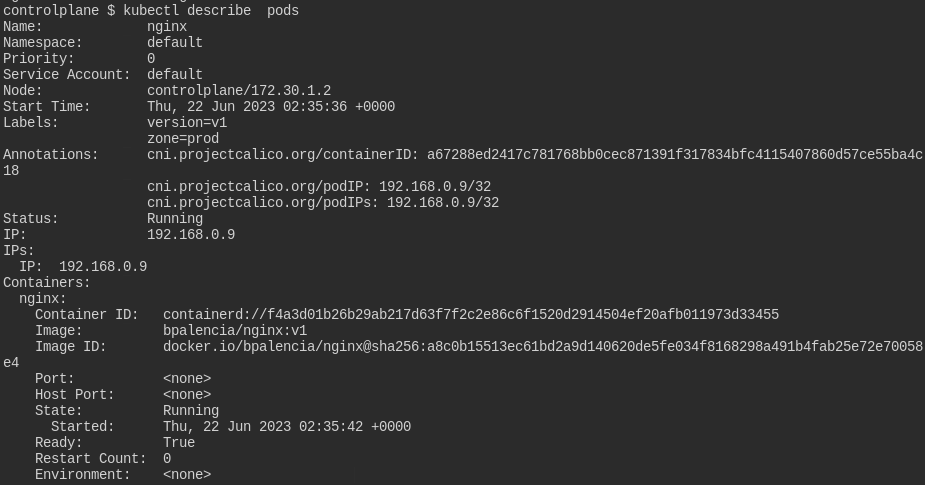
****

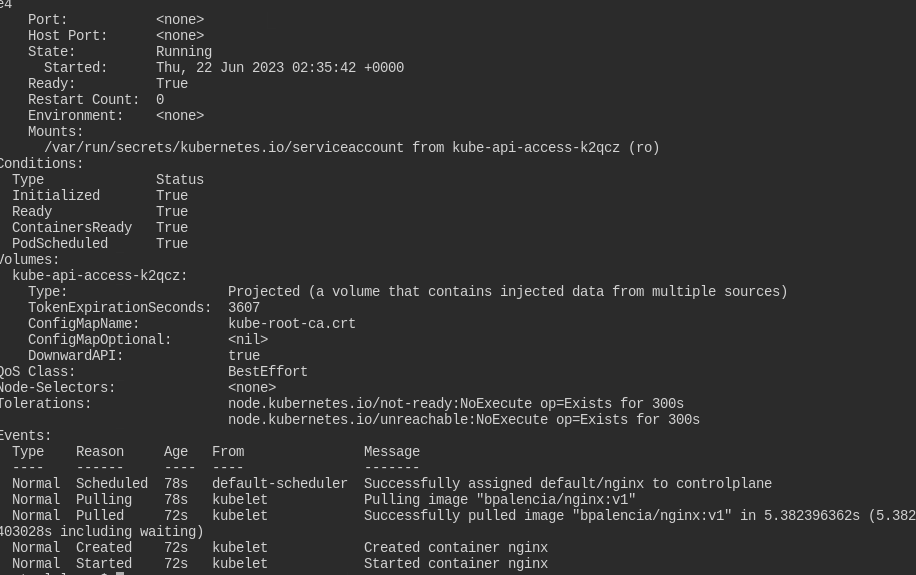
**kubectl create -f nginx.yaml**

****

**kubectl get pods**

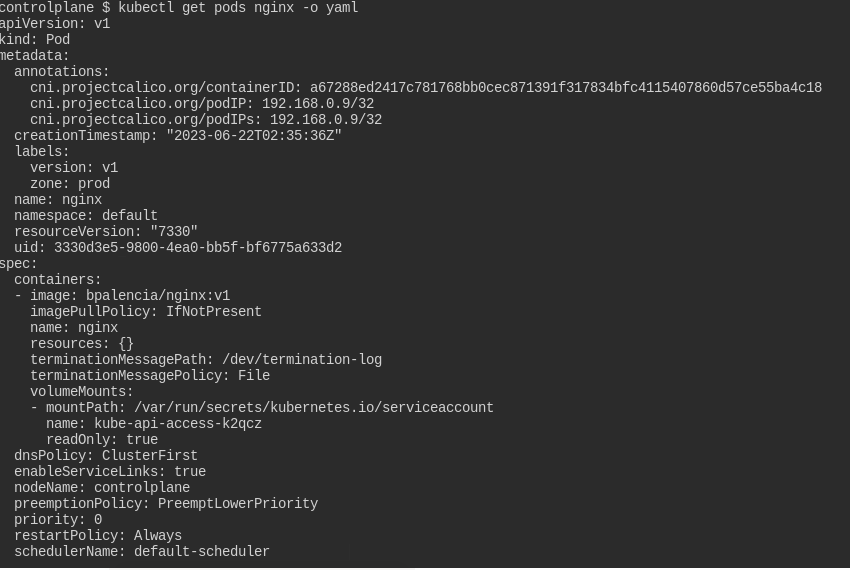
****

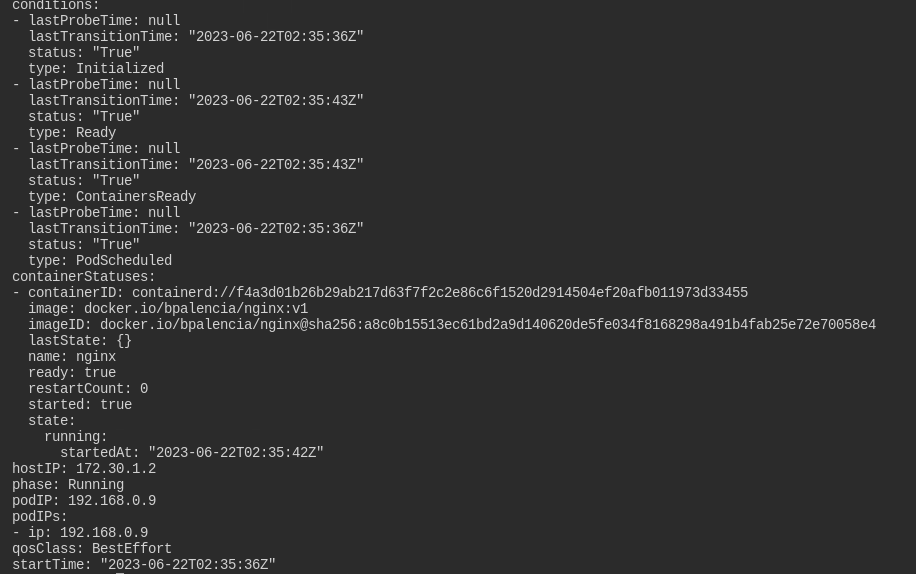
**c**

**c**

**kubectl create -f nginx.yaml**

Muestra la informacion en formado .yaml

**c**

****

**Pruebas con el comando apply**

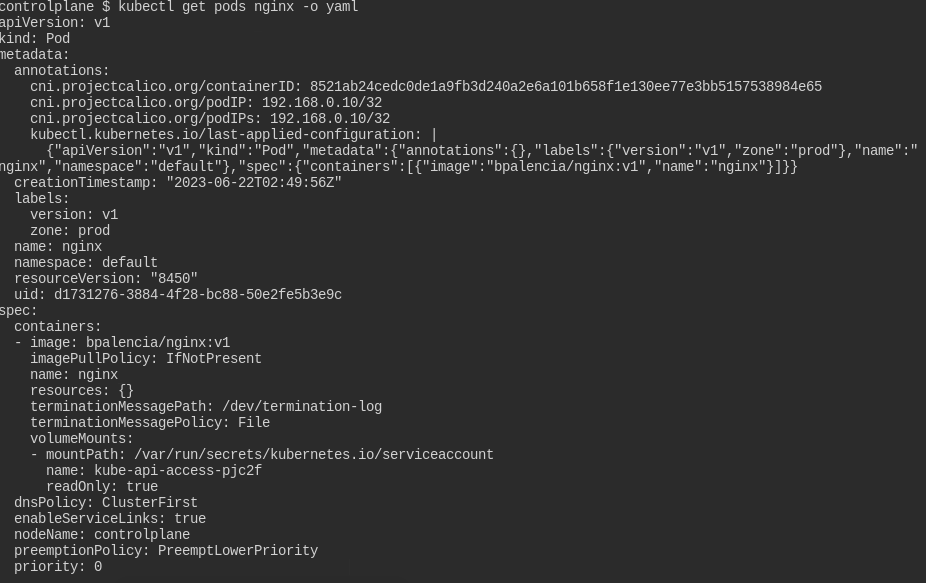
**kubectl delete pod nginx**

****

**kubectl apply -f nginx.yaml**

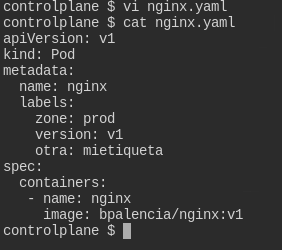
****

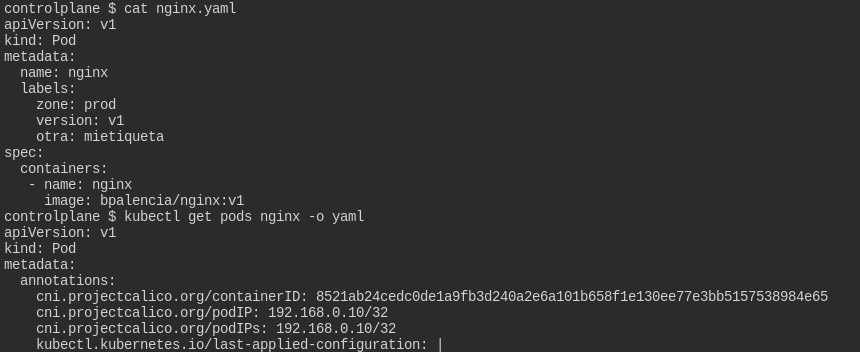
Como no existia creo nuevamente el pod con el cotenedor

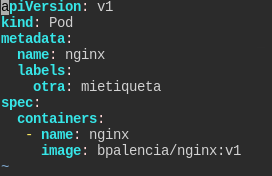
La opcion apiversion tieninformacion de la ultima version aplicada.

A

Camibiar el .yaml para hcaer la prueba



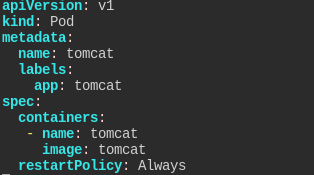
cc



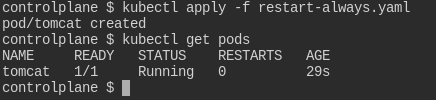
Solo aparece la etiqueta que deje



**Rebotar POD**

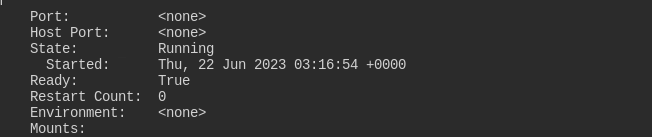


**kubectl pply f restartalways.yaml**



kubectl describe pods tomcat

aparece las veces que se ha reiniciado

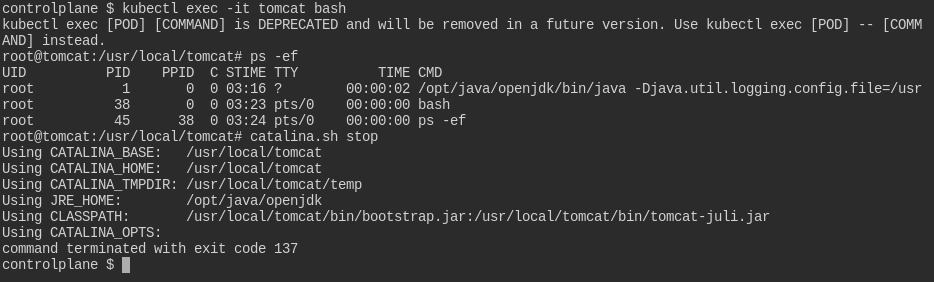


kubectl exec -it tomcat bash

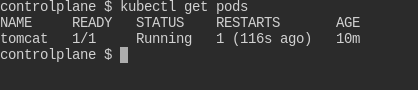
entrar dentro de la shell del contenedor

ps -ef para ver el tomcat corriendo

catalina.sh stop: con este comando podemos parar el toncat y nos devolvera al manejador

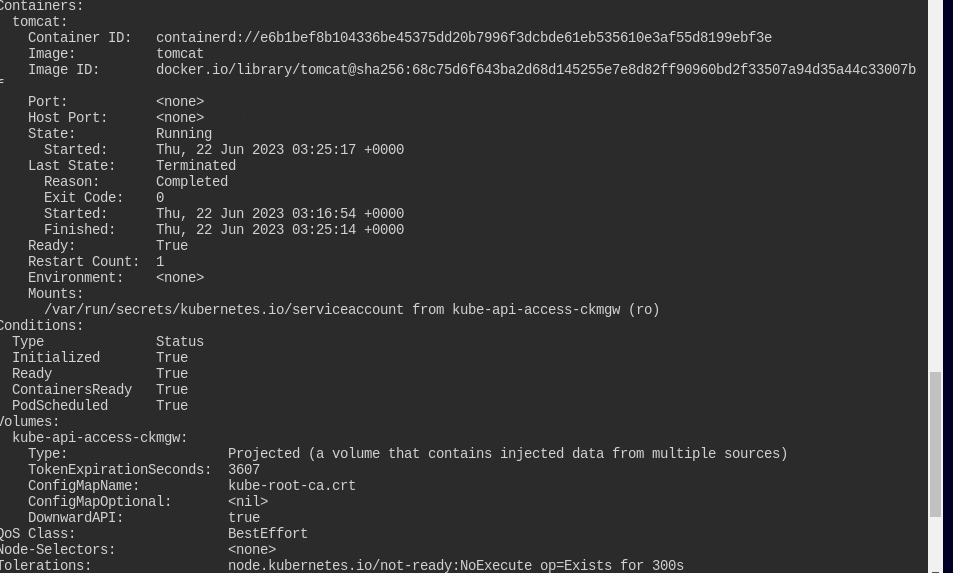
kubectl get pods

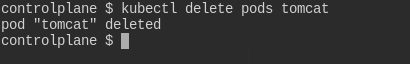
aparece que fue reiniciado

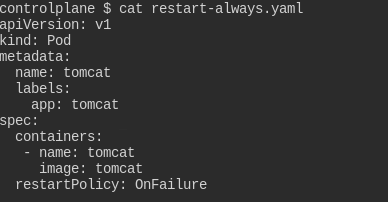


kubectl describe pods tomcat

aparece que fue teminado

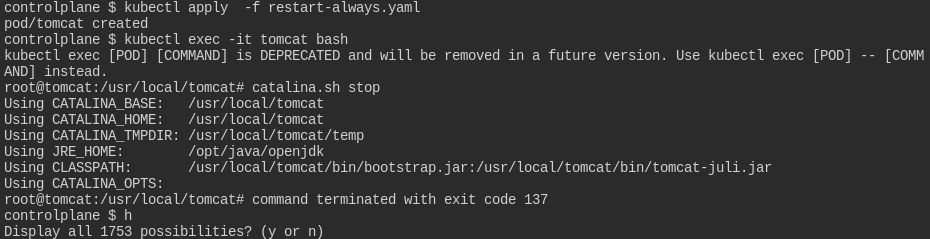
**Ccambio en la configuracion del archivo .yaml Onfailure**

****



c

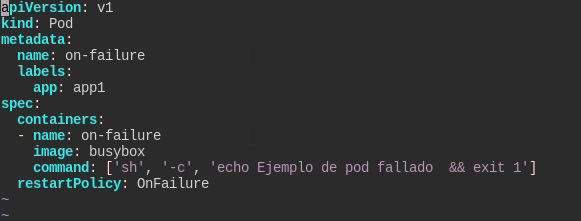


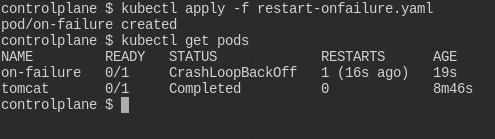
cAparece completa y no reinicia porque se coloco stop no fue por fallo



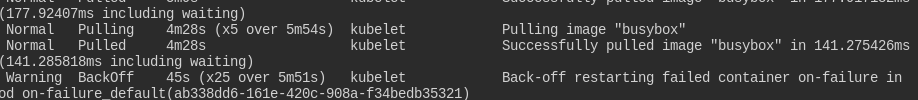
**Ejemplo 3**

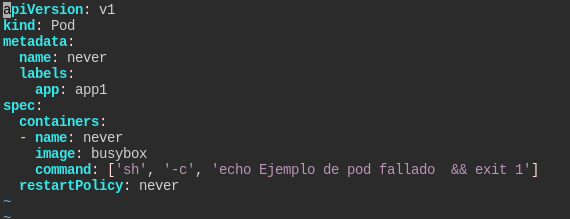
Si falla el POD intenta reiniciarlo

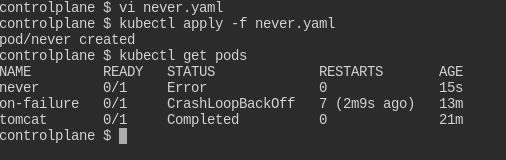
**kubectl apply ~~f restart~~onfailure.yaml**

****

**kubectl describe pods on-failure**

**con politica never**

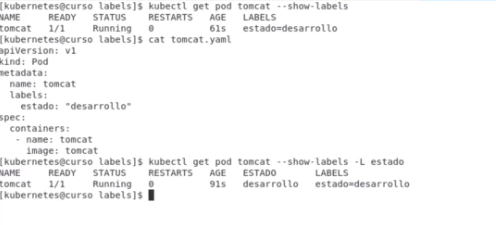
****

****

**Etiquetas**

**kubectl get pod tomcat –show labels**

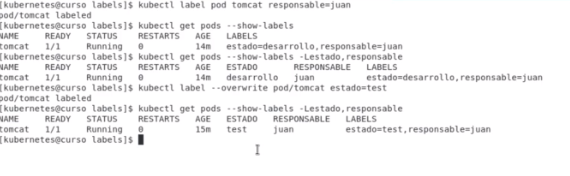
**kubectl get pod tomcat –show labels -L estado**

****

kubectl label –overrite epod/tomcat tiqueta=nuevonombre

kubectl get pods –show labels

kubectl get pods –show labels -L estado,responsable

****

**eliminar la etiqueta**

kubectl label pod/tomcat responsable-



**Nota: Las etiquetas declarativas se colocan en el documento yaml esta es la mejo**

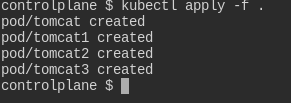
**r forma**

**Selector**

Condiciones que utilizo para localizar objetos

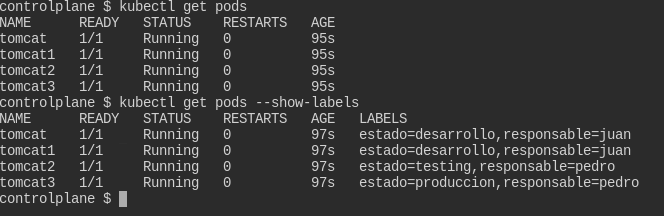
|  |  |
| --- | --- |
|  |  |

Crear todos los nodos y cotenedores



kubectl get pods

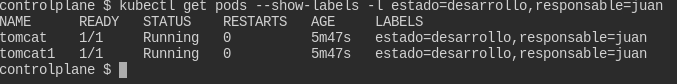
kubectl get pods --show-label



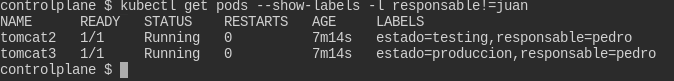
kubectl get pods --show-labels -l estado=desarrollo busca los label que sean de desarrollo



kubectl get pods --show-labels -l estado=desarrollo,responsable=juan

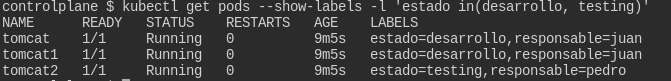
Que seel label no sea juan

get pods --show-labels -l responsable!=juan

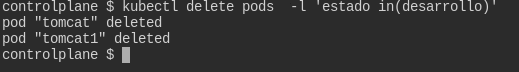


Que este en desarrollo y testing

get pods --show-labels -l 'estado in(desarrollo, testing)'

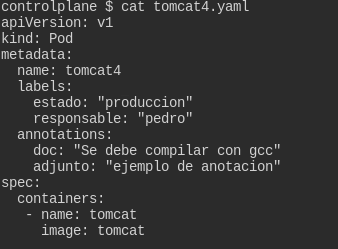


Borrar los dos pods que el estado sea desarrollo



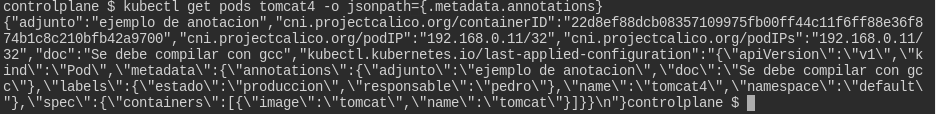
**Anotaciones**

Sirven para describir y documentar nuestros componentes





kubectl get pods tomcat4 -o jsonpath={.metadata.annotations}



**Deployment**

Todos los componentes que van a envolver a los pods y que le dan caracteristicas como son hacer rollback y update

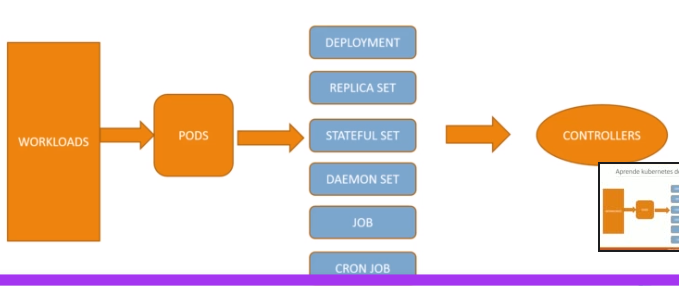
Replicaset: para hccer el escalado, para subir o bajar los pods de acuerdo alo que los cluster tengan indicad

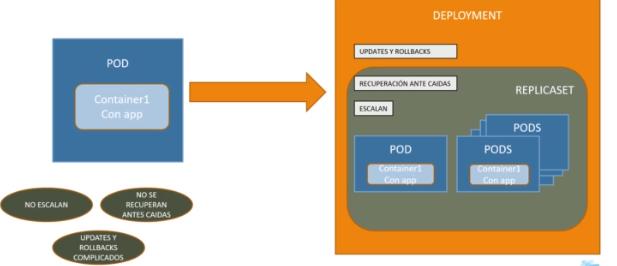
Stefulset: Gestiona el despliegue y el escalado. Garantiza el orden y la unicidad del pods

Daemon set: componente que agefura que todos los nodos del cluster van a tener una copia del pods

jobJob: crea uno o varios pods y se van a sasegurar que de un determinado numero de ellos se termine satisfactoriamente

Cron Job similar al job pero de forma planificada





kubertcl apply -f .

**Crear un deployment y mostrar el deployd, los replicaset y los pod**

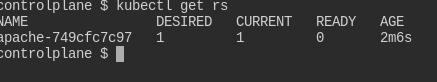
kubectl create deployment apache --image=thhpd



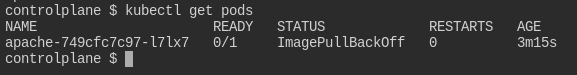
kubectl get deployment



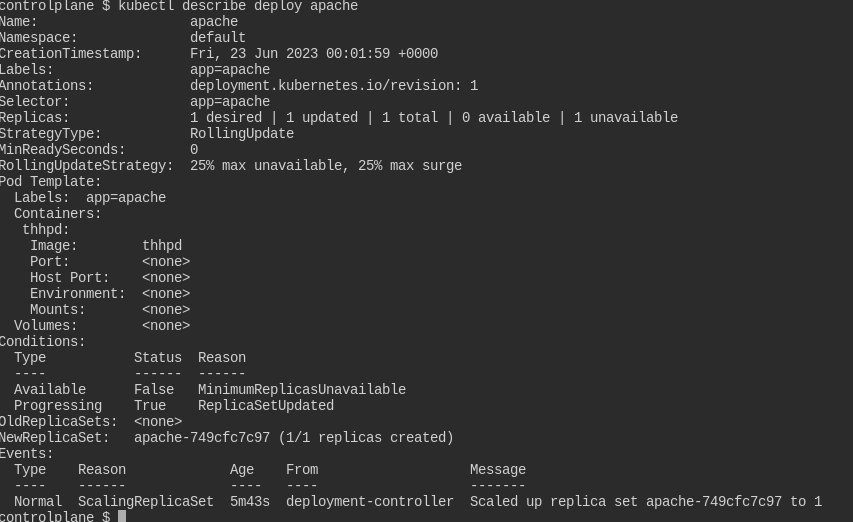
kubectl get rs



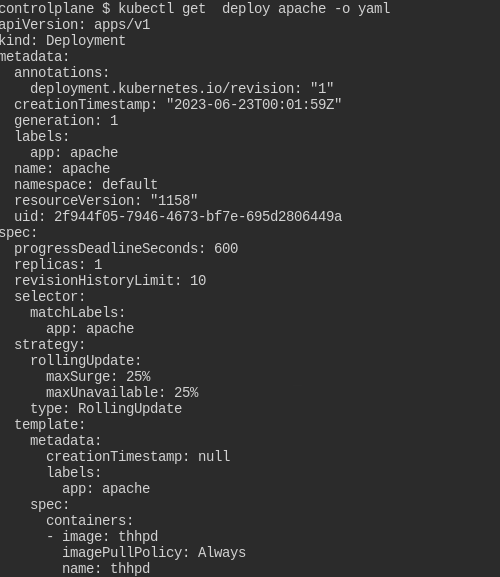
kubectl get pods



kubectl describe deploy apache



kubectl get deploy apache -o yaml



Ejercicio

apiVersion: apps/v1 # i se Usa apps/v1beta2 para versiones anteriores a 1.9.0

kind: Deployment

metadata:

name: nginx-d

spec:

selector: #permite seleccionar un conjunto de objetos que cumplan las condicione

matchLabels:

app: nginx

replicas: 2 # indica al controlador que ejecute 2 pods

template: # Plantilla que define los containers

metadata:

labels:

app: nginx

spec:

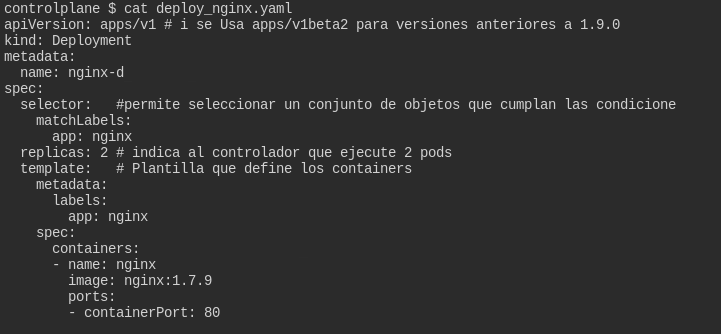
containers:

- name: nginx

image: nginx:1.7.9

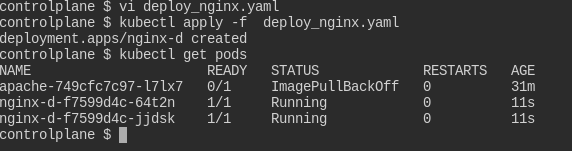
ports:

- containerPort: 80

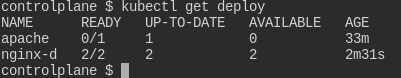


kubectl apply -f deploy\_nginx.yaml

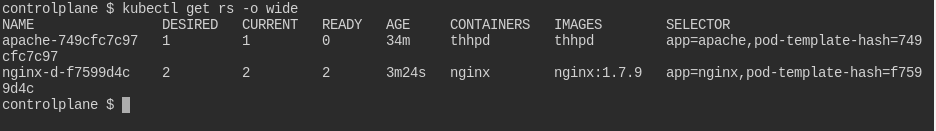
kubectl get pod



kubectl get deploy



kubectl get rs -o wide



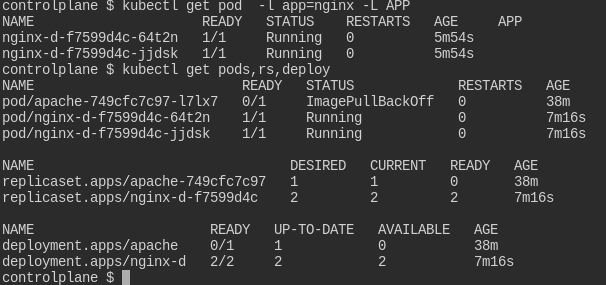
kubectl get -l app=nginx



editar a 5 replicaset

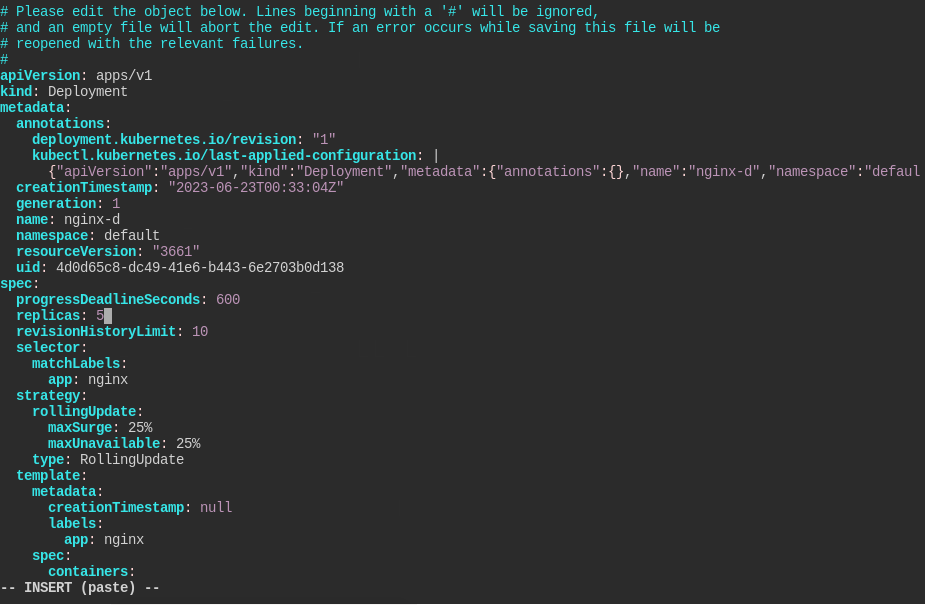
c

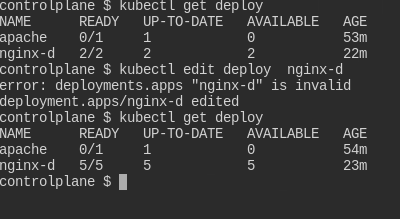
kubectl get pod -l app=nginx



comando edit

Modifica el comportaminto en forma



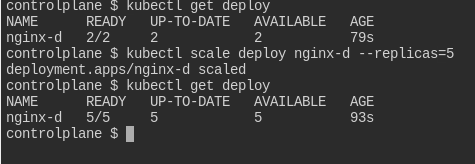


**Comando scale**

**kubectl get deploy**

**kubectl scale deploy nginx-d --replicas=5**

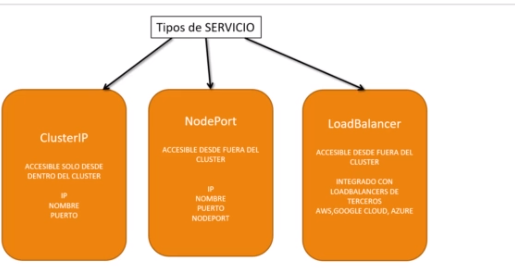
**kubectl get deploy**

****

**Otra forma de escalar es on una etiqueta**

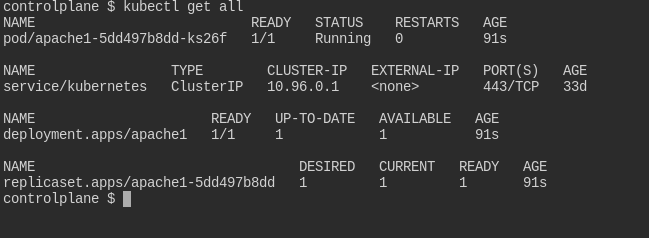
****

**Servicios**

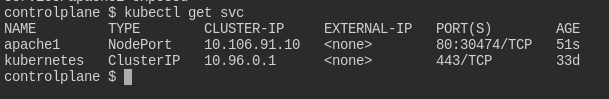
****

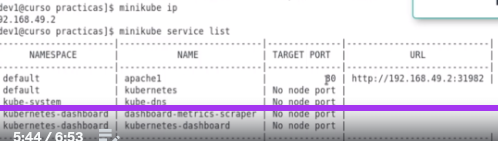
****

**kubectl create deployment apache1 –image=httpd**

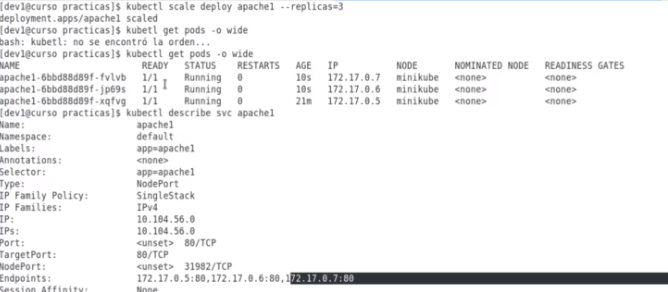
**kubectl expose deploy apache1 --port=80 --type=NodePort**

 **kubectl get svcc**

**Si se esta trabajando con minikube se puede bucar la ip**

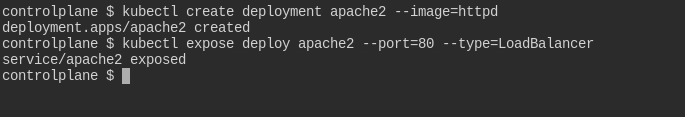
****

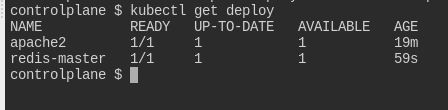
**Si deseas escalar se puede ver en el endpoind las ip de las que se crearon nuevas**

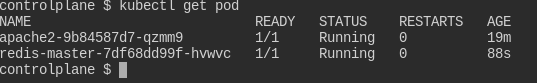
**Ejemplo con Load Balancer**

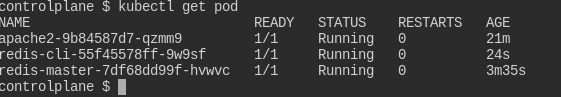
**kubectl create deployment apache2 –image=httpd**

**ubectl expose deploy apache2 --port=80 --type=LoadBalancer**

**El external -ipqueda endiente porque no estoy en un cloud**

****

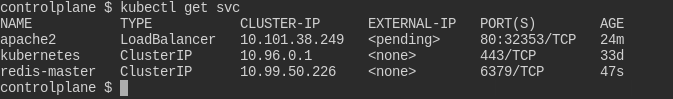
****

****

**Para conectarnos al manestro vamos a necesitar un servicio**

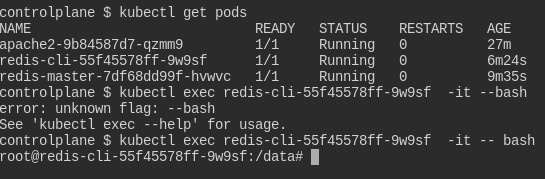
kubectl expose deploy redis-master --port=6379 --type=ClusterIP

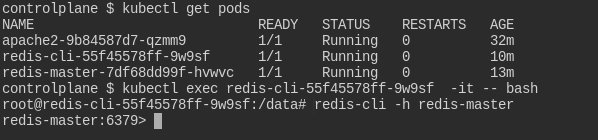
kubectl get svc

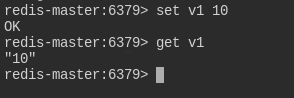
**Conectarme y entrar al al redis cli**

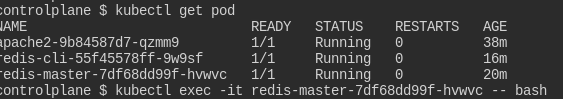
**kubectl get pods**

**kubectl exec redis-cli-55f45578ff-9w9sf -it -- bash**

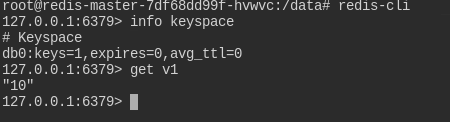
**Conectarme desde nodes cloi a node master**

**Colocar un valor en v1 en master desde cli y luego ingresar a master y verlo en otra tag**

****

****

**x**

****