

4:30 45

5:30 65

6:30 95

7:30 105

Docker image

Docker images –a

Docker search python:3.6

docker search --filter "is-official=true" registry

docker images ls

docker image pull –all-tags nginx

docker image inspect ubuntu:16.04

docker image history ubuntu:16.04

docker image rm nginx:1-alpine-perl

docker rmi –f imagename

container

docker container create -it --name cc-busybox-A busybox:latest

docker container run -itd --rm --name cc\_busybox\_B busybox:latest

docker container stop cc\_busybox\_B

docker container start cc-busybox-A

docker container restart --time 5 cc-busy-a

docker container rename my\_busybox my-busybox

docker container attach my-busybox

docker container inspect my-busybox

docker inspect --format='{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' my-busybox

docker exec -it my-busybox bash

docker container run -itd --name cont\_nginx -p 8080:80/tcp img\_expose

docker container run -itd --name cont\_nginx-A -p img\_expose

docker rm –f containername

docker container prune

docker network

docker network create --driver bridge my-bridge

docker network create --driver bridge --subnet=192.168.0.0/16 --ip-range=192.168.5.0/24 my-bridge-1

docker network ls --filter driver=bridge

docker network connect my-bridge-1 cont\_nginx

docker container inspect cont\_nginx

docker container run -itd --network host --name cont\_nginx nginx:latest

docker network inspect my-bridge-1

docker network inspect --format "{{.Scope}}" bridge

docker network inspect --format "{{.ID}}: {{.Name}}" bridge

docker network disconnect my-bridge-1 cont\_nginx

docker container inspect cont\_nginx

docker network prune

docker network create --driver bridge --subnet 172.20.0.0/16 --ip-range 172.20.240.0/20 net-bridge

docker run -it --net=net-bridge --name=cont\_database redis

docker inspect --format='{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' cont\_database

docker run -itd --net=net-bridge --name=server-A busybox

docker inspect --format='{{json .Containers}}' net-bridge | python -m json.tool

docker run -itd --name=server-B busybox

docker container exec -it cont\_database bash

ping [www.google.com](http://www.google.com)

apt-get update

apt-get install iputils-ping

ping 172.20.240.1

docker volumes is the storage of container

docker volume create vol-busybox

docker run -d --volume vol-ubuntu:/temp ubuntu

docker volume ls

docker volume ls --filter "dangling=true"

docker volume inspect vol-ubuntu

docker volume rm vol-ubuntu

docker container rm goofy\_khorana

docker volume rm vol-ubuntu

docker volume ls

docker run -itd --name cont-ubuntu --volume vol-ubuntu:/var/log ubuntu:latest

docker container inspect --format "{{json .Mounts}}" cont-ubuntu | python -m json.tool

docker exec -it cont-ubuntu bash

cd /var/lib/docker/volumes

mkdir bind-data

docker run -itd \

> --name bind-ubuntu \

> --mount type=bind,source="/root/bind-data",target=/tmp \

> ubuntu:latest

docker container inspect bind-ubuntu

docker exec -it bind-ubuntu bash

touch /tmp/foo.txt

https://github.com/cerulean-canvas/2048.git

[root@ip-172-31-45-84 2048]# docker run -itd --name 2048 \

> --mount type=bind,source="$(pwd)"/,target=/usr/share/nginx/html \

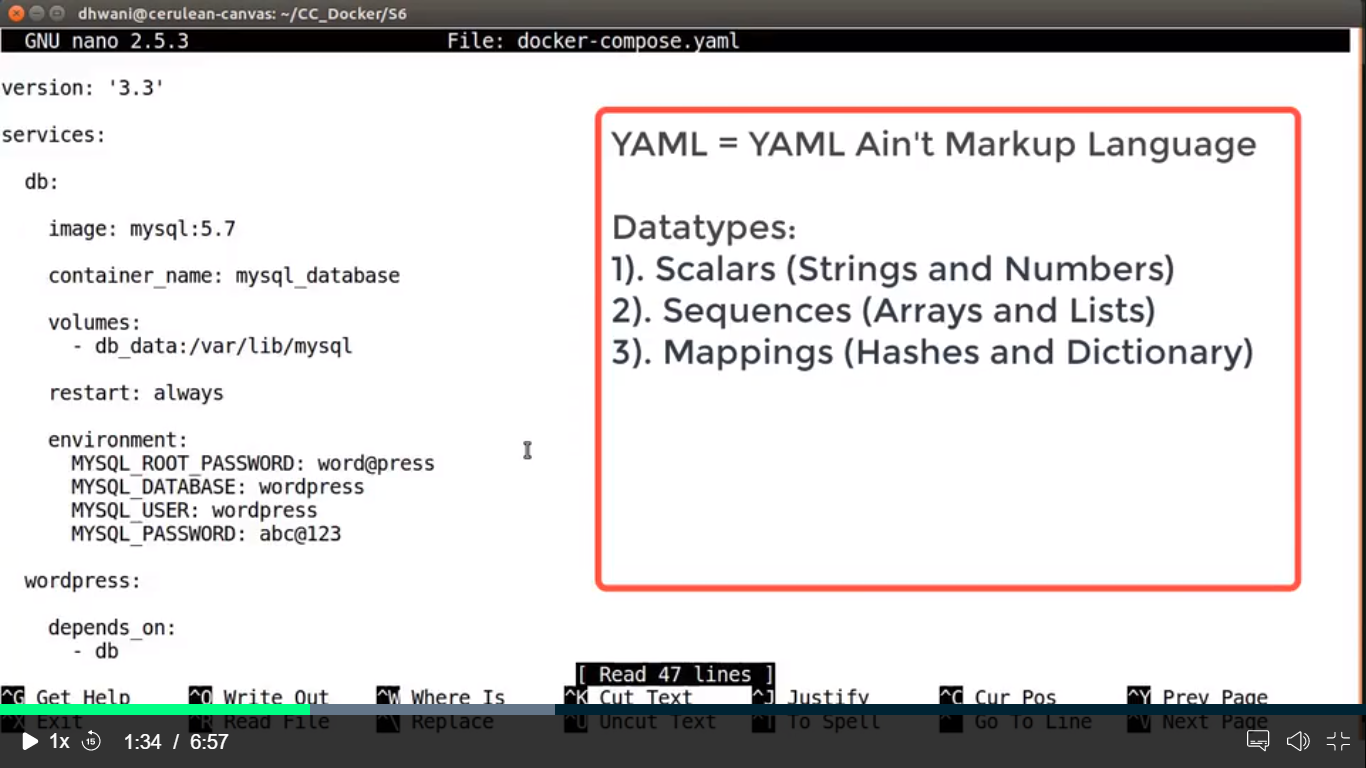
> -p 8080:80 \

> nginx:latest

Docker compose

sudo curl -L https://github.com/docker/compose/releases/download/1.22.0/docker-compose-$(uname -s)-$(uname -m) -o /usr/local/bin/docker-compose

sudo chmod +x /usr/local/bin/docker-compose



docker-compose up -d

docker exec –it mysql\_database bash

docker-compose config

docker-compose config --services

docker-compose images

docker-compose logs

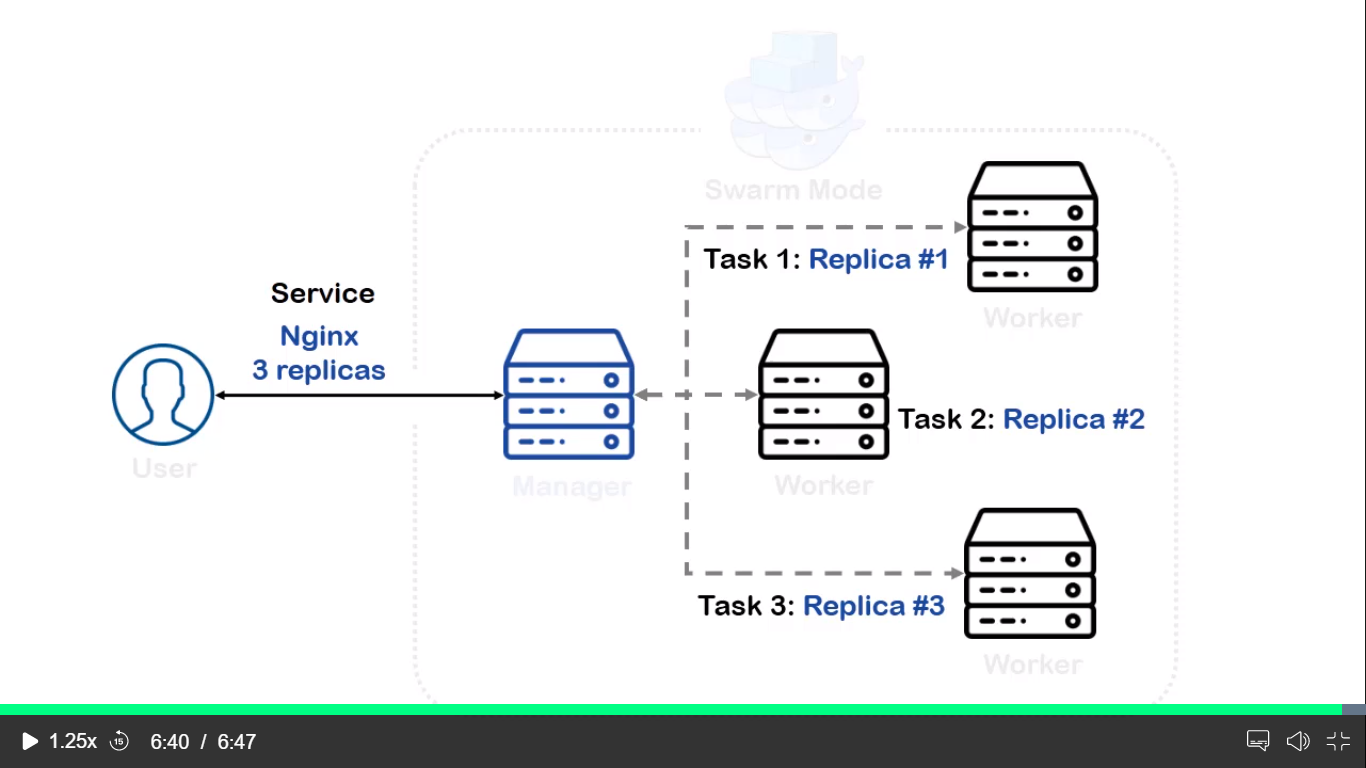
docker-compose logs --tail=10

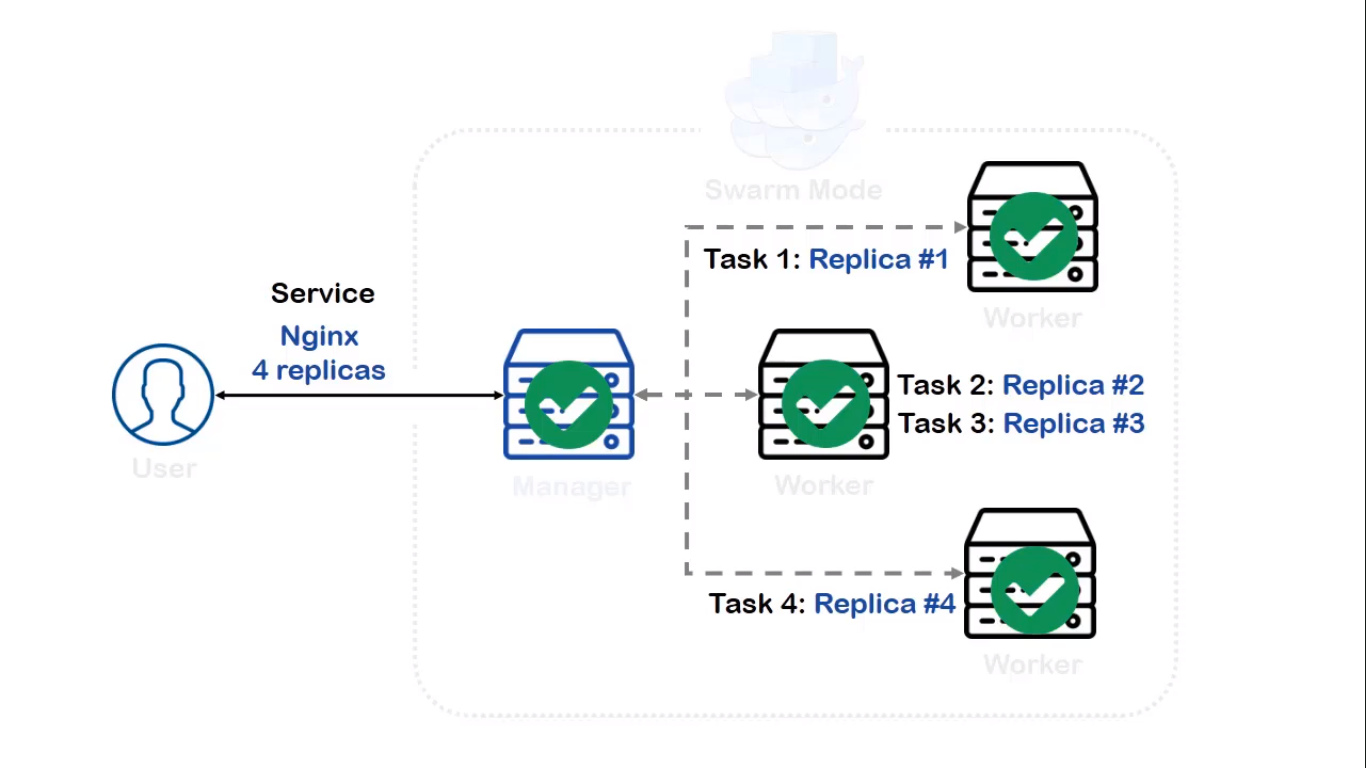
docker-compose ps

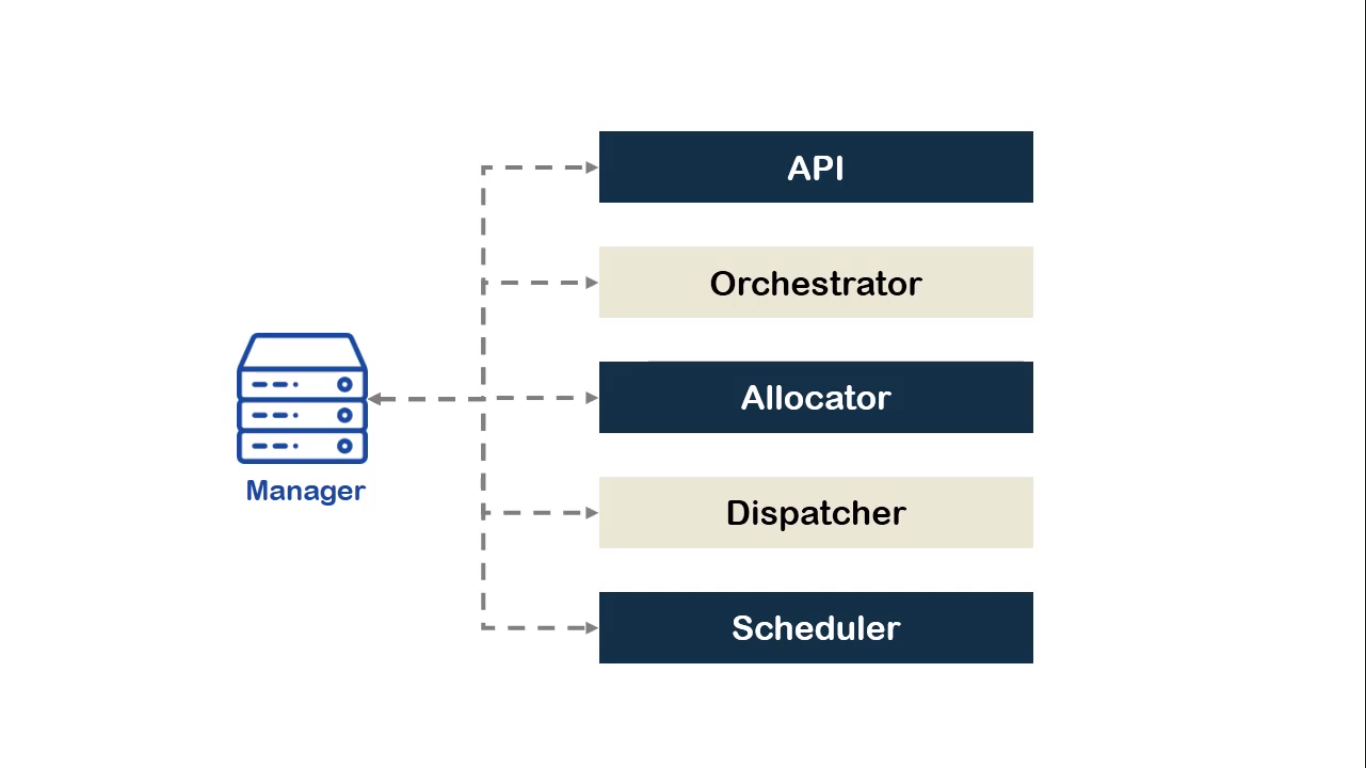
docker-compose top

docker-compose down

docker swarm







Sudo nano /etc/apt/source.list

docker swarm init --advertise-addr 13.59.104.199

docker swarm join-token worker

docker service create --name web-server -p 8080:80 --replicas 3 nginx:latest

docker service ls

docker service ps web-server

docker service inspect web-server

docker ps –a

docker node update --availability drain worker-2

docker service ps web-server

docker node rm worker-2

docker swarm leave

docker service scale web-server=6

docker service ps web-server

docker service update --image nginx:alpine web-server

docker service inspect --pretty web-server

docker service rm web-server

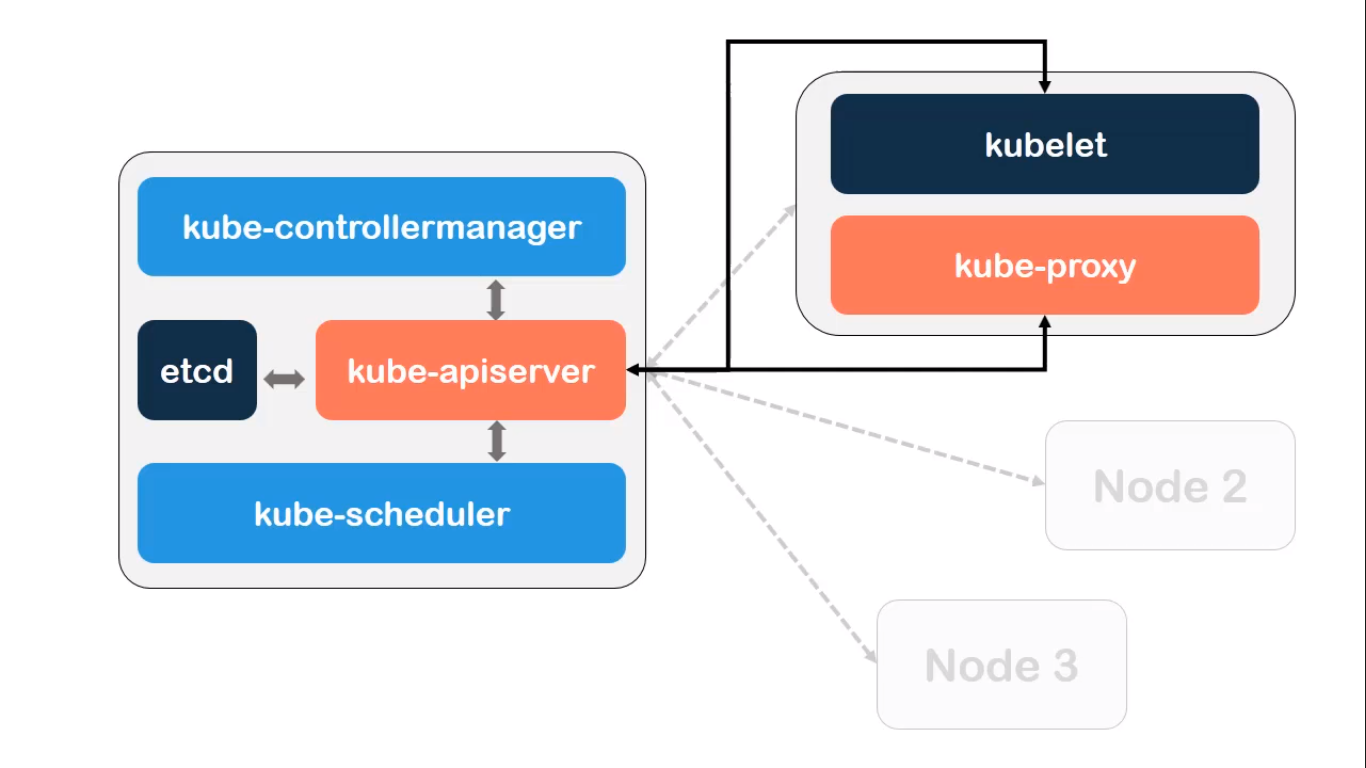
docker swarm leave

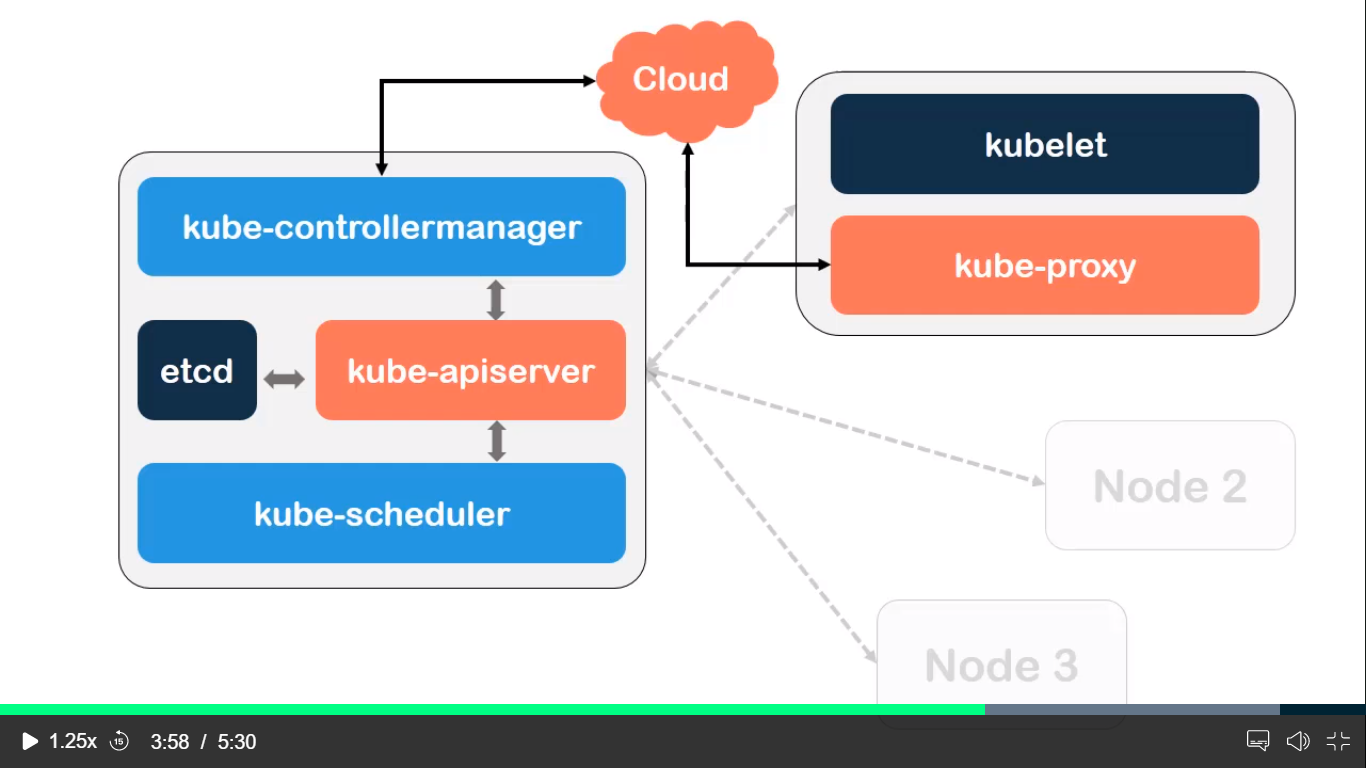
docker swarm leave --force

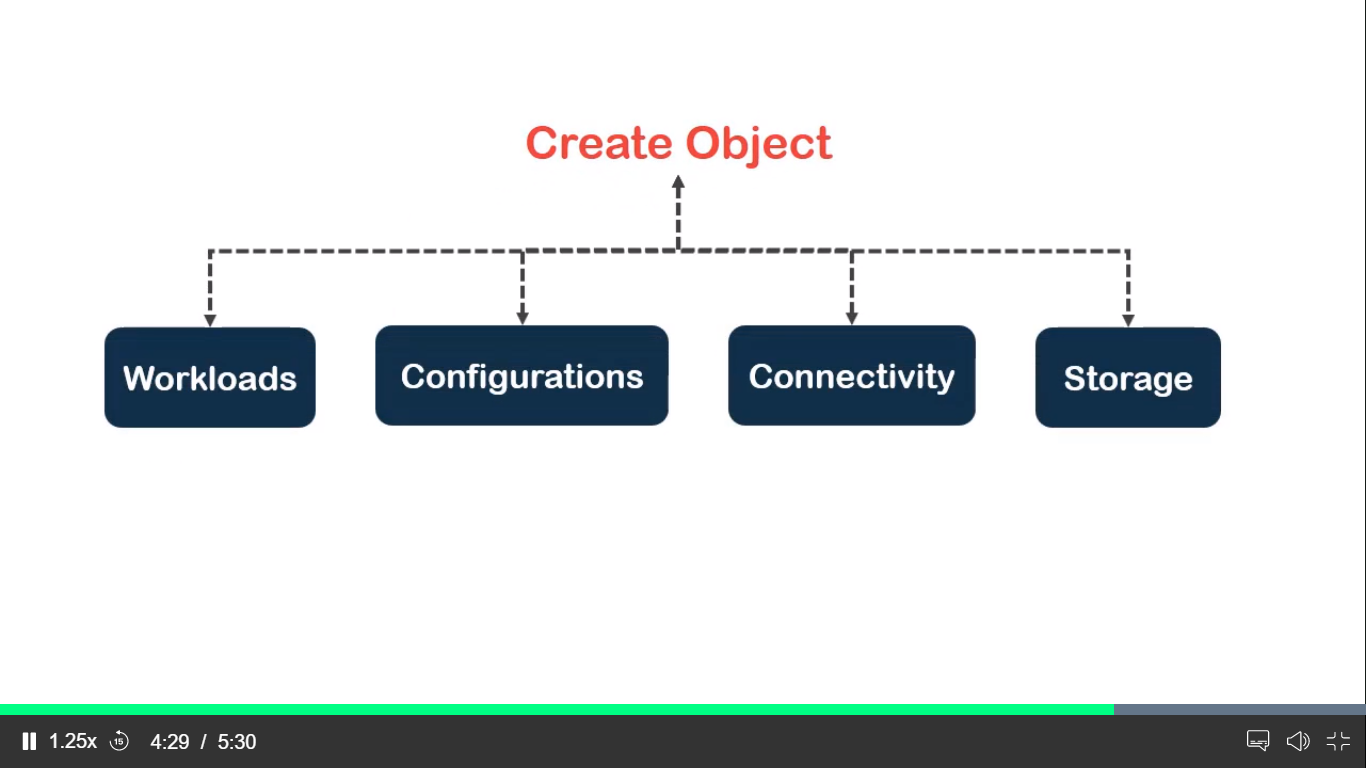
kubernetes

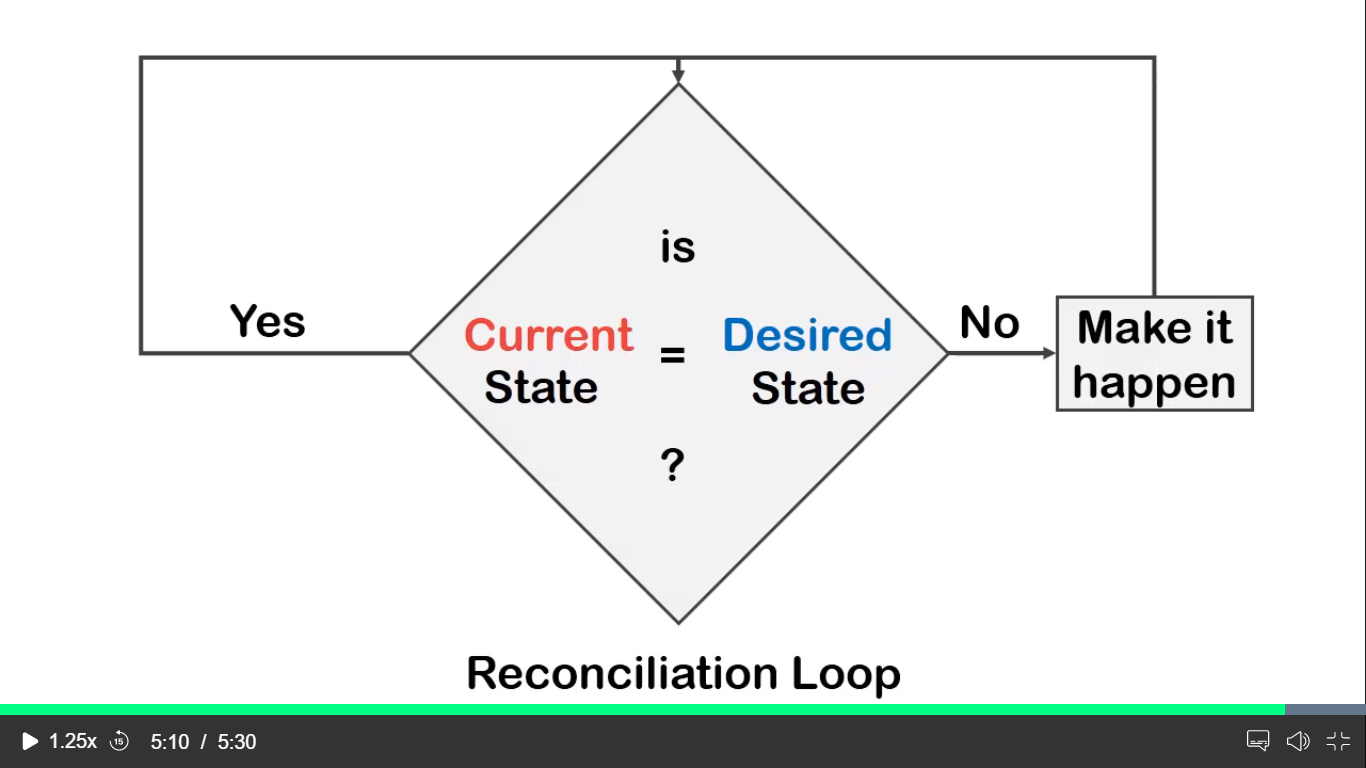
master

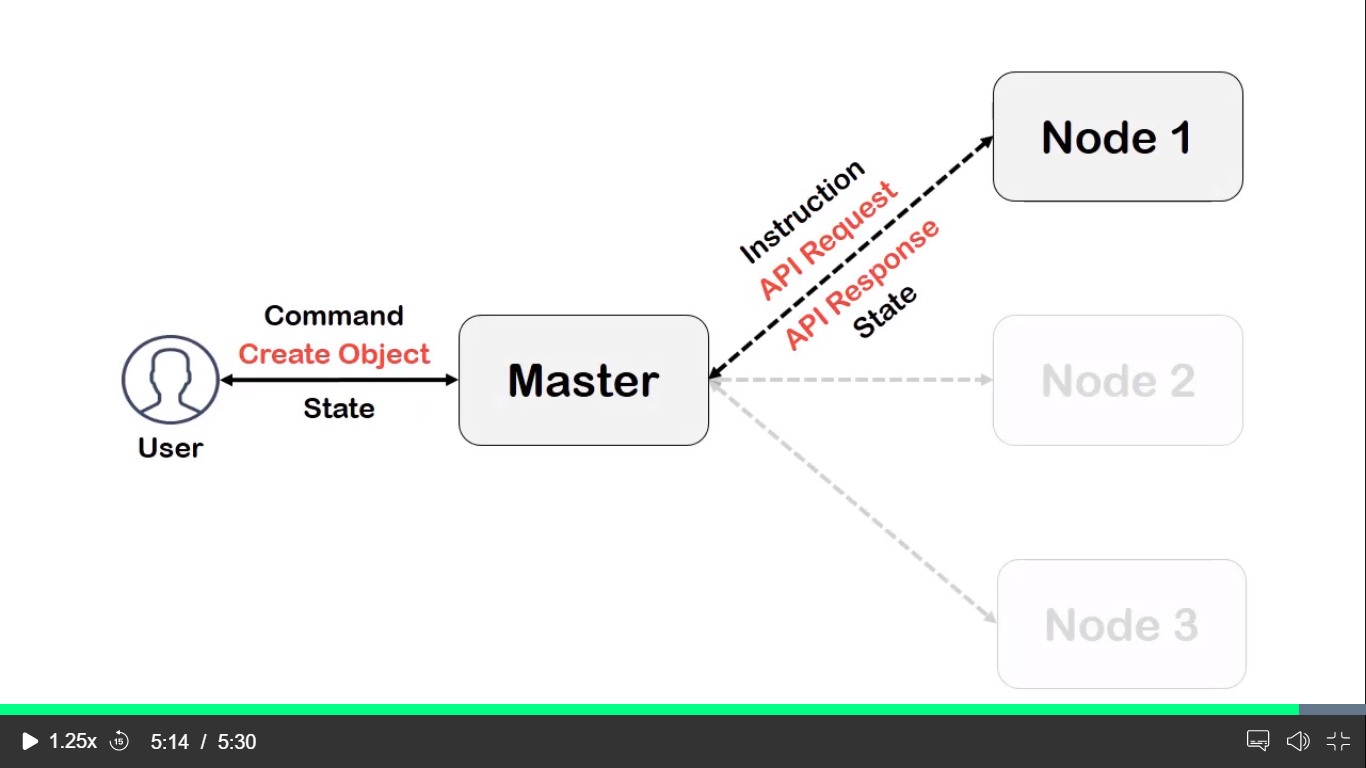












<https://console.cloud.google.com/?pli=1>

creating master in kubernetes

Creating kubernetes master instance in GCP

Sudo su –

Apt-get update

Apt-get install –y docker.io

Docker version

Apt-get update && apt-get install –y apt-transport-https curl

curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add -

cat <<EOF >/etc/apt/sources.list.d/kubernetes.list

> deb http://apt.kubernetes.io/ kubernetes-xenial main

> EOF

root@master:~# apt-get update

apt-get install –y kubelet kubeadm kubectl

exit and login again

root@master:~# sysctl net.bridge.bridge-nf-call-iptables=1

root@master:~# kubeadm init

we must copy all this three commands in our note pad

kubeadm join 10.182.0.8:6443 --token p44flr.p1yak15trztl06so \

--discovery-token-ca-cert-hash sha256:39794fab9ae97bb5a15dbeaccbc20ef37ecd86a030483ae068bad2d9758dacb6

kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-version=$(kubectl version | base64 | tr -d '\n')"

kubectl get pods --all-namespaces

kubectl get nodes

kubernetes commands

vi imperative-pod.yaml

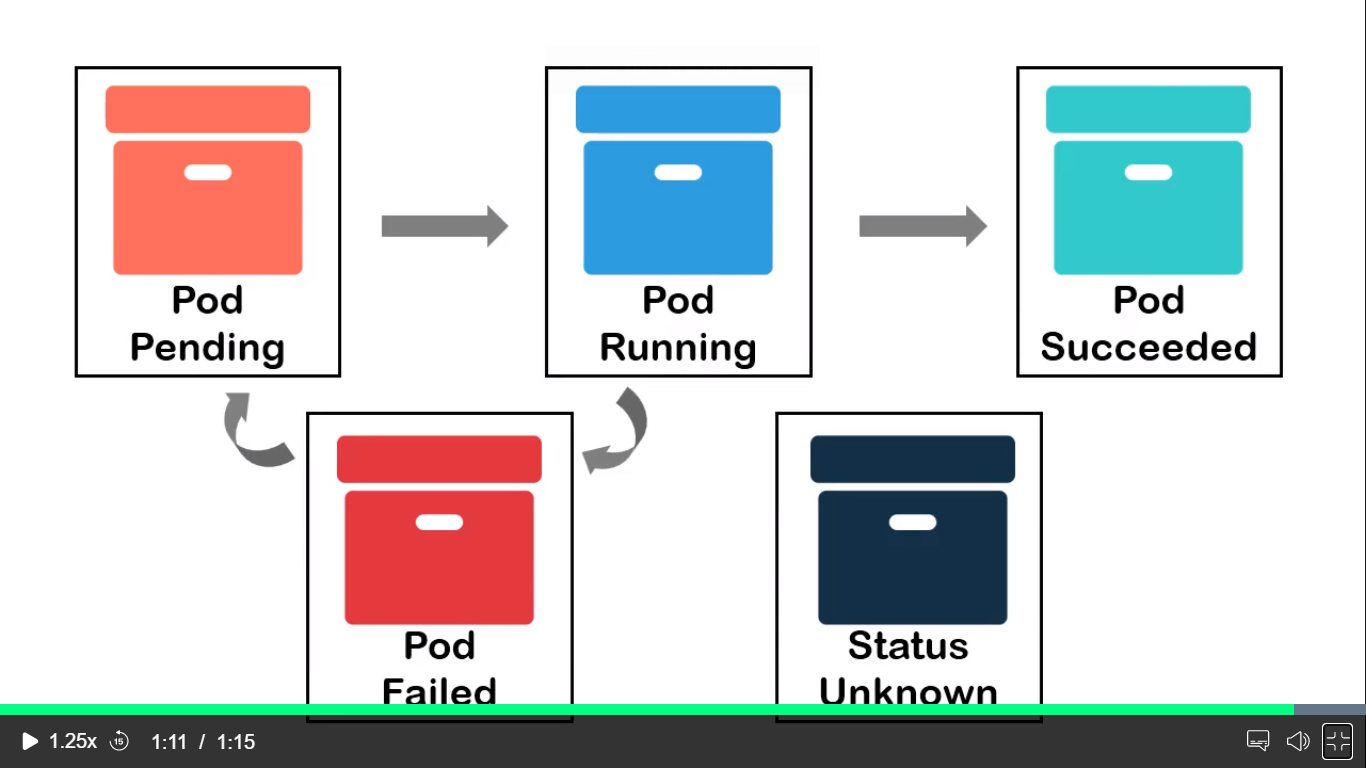
vi declarative-pod.yaml

kubectl create -f imperative-pod.yam

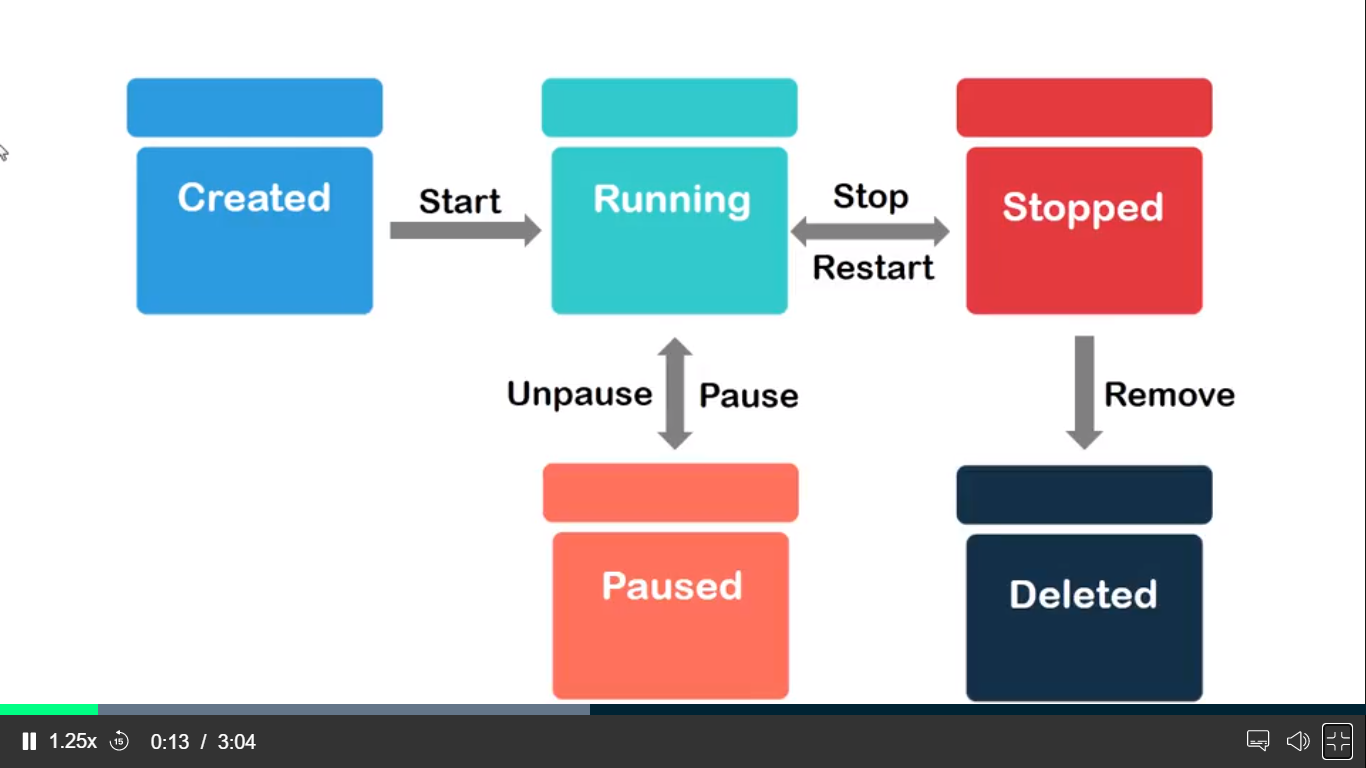
kubectl apply -f declarative-pod.yaml

kubectl get pods

kubectl describe pods imp-pod



Container life cycle



Hooks

Vi lifecyc-pod.yaml

kubectl create -f lifecyc-pod.yaml --validate=false

kubectl get pods

kubectl exec -it lifecyc-pod -- /bin/bash

cat /usr/share/poststart-msg

vi command-pod.yaml

kubectl get pods

kubectl exec -it cmd-pod -- /bin/bash

kubectl describe pod cmd-pod

kubectl logs cmd-pod

vi environment-pod.yaml

kubectl create -f environment-pod.yaml

kubectl get pods

kubectl describe pod env-pod

kubectl exec -it env-pod -- /bin/bash

kubectl exec -it env-pod -- /bin/bash

printenv

namespaces

kubectl get namespaces

kubectl get pods

kubectl get pods --all-namespaces

kubectl create namespace my-namespaces

kubectl create -f imperative-pod.yaml -n my-namespace

kubectl get pods --all-namespaces

resource ( kind of memory to the containers)

kubectl get pods

vi resource-pod.yaml

kubctl create –f resource-pod.yaml

kubectl get pods

kubectl describe pods frontend

controller

replication controller

deployment controller ( replication , status , updates and services exposes on the pods )

statefulsets ( arranges the execution of the pods)

jobs ( creates the jobs after the execution of the job)

work with replicatesets

kubectl delete pods --all

vi replica-pod.yaml

kubectl create -f replica-pod.yaml

kubectl create -f replica-pod.yaml

kubectl describe pods frontend-5mcjx

kubectl get rs

kubectl describe rs frontend

kubectl delete pods frontend-5mcjx

deployments

work with deployment

vi deployment.yaml

kubectl create –f deployement.yaml

kubectl get pods

kubectl describe pods deploy-nginx-54cd468594-mpqtj

kubectl get deployment

kubectl describe deployments deploy-nginx

jobs

what are jobs

vi jobs.yaml

kubectl create –f jobs.yaml

kubectl get pods

kubectl describe job-pi-rgrn5

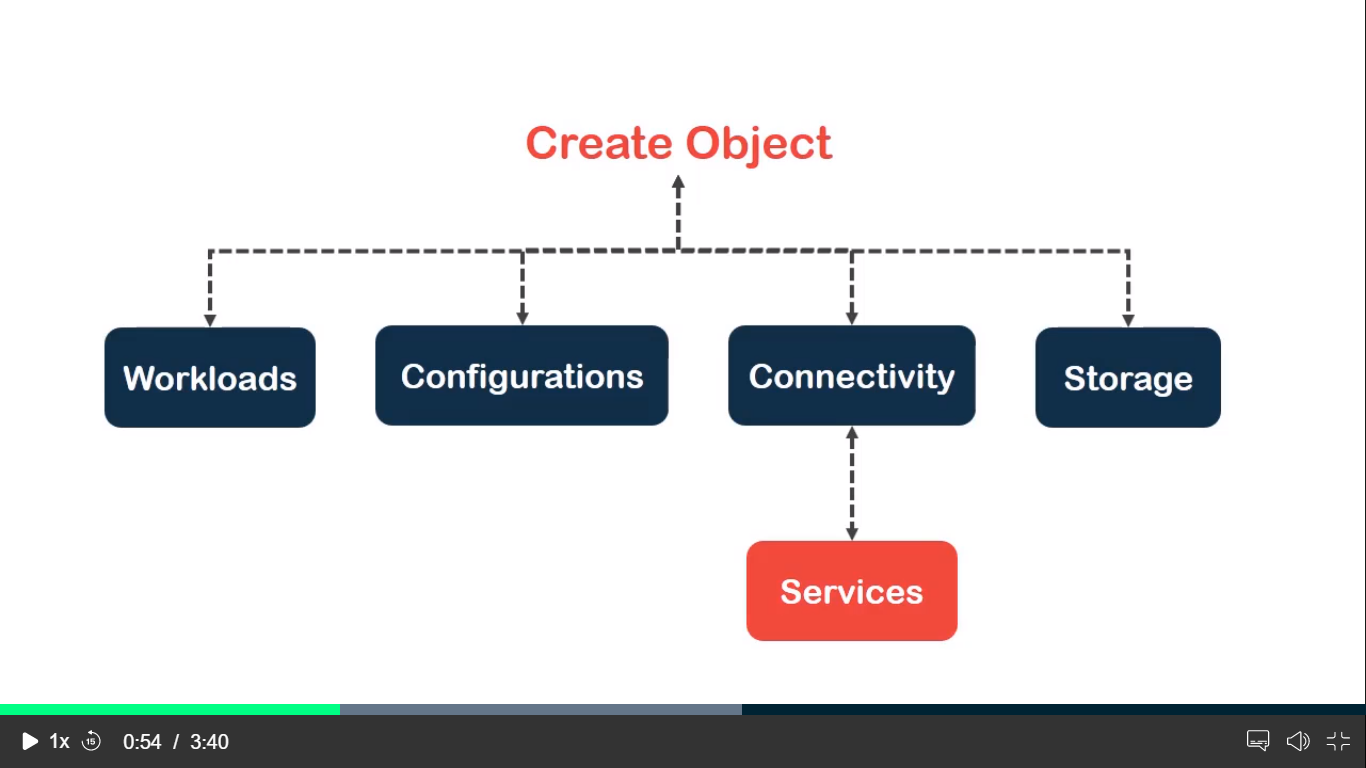
kubectl get jobs

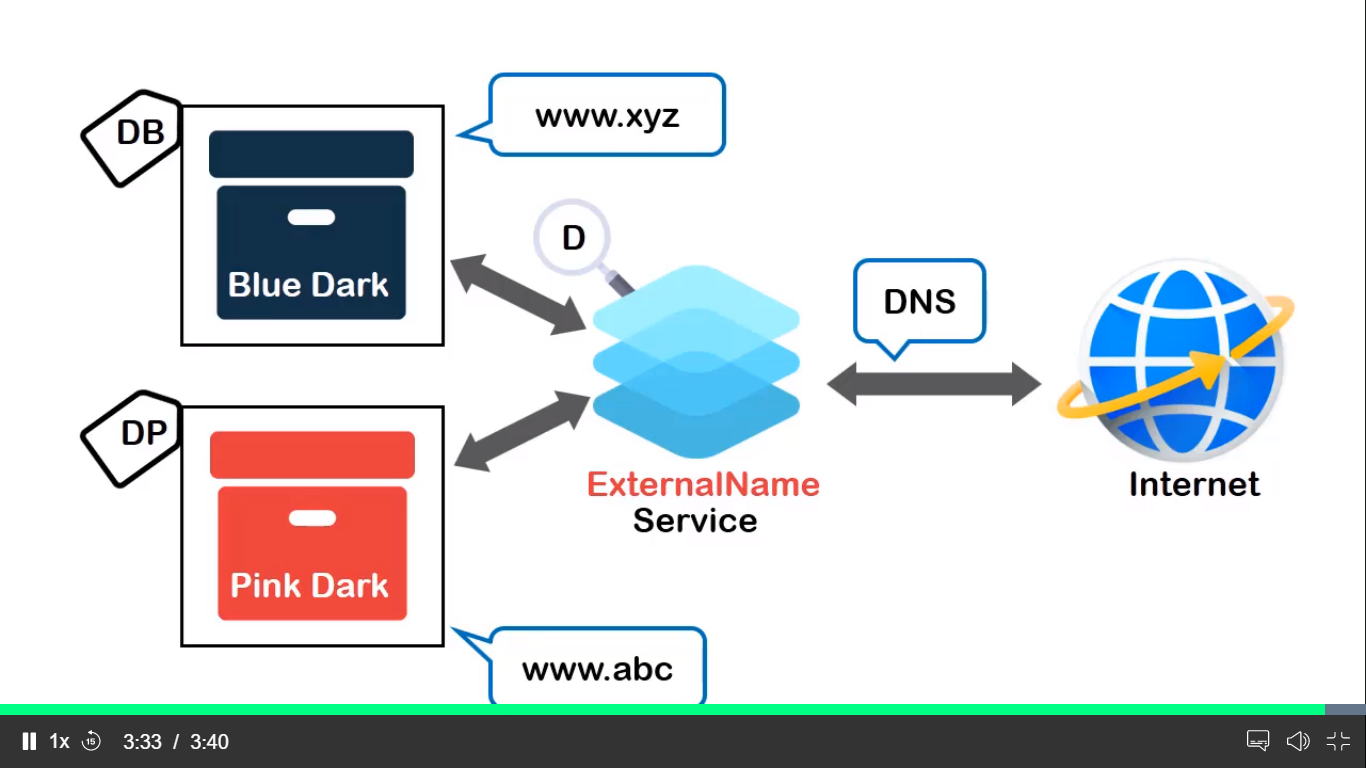
kubectl describe jobs job-pi

logs

kubectl logs job-pi-rgrn5

what are services





Cluster ip

Services

vi deploy-nginx.yaml

vi serve-nginx.yaml

kubectl create –f deploy-nginx.yaml

kubectl create –f serve-nginx.yaml

kubectl get pods

kubectl get svc (clusterip it allows the commincate with in the cluster port 443)

kubectl describe svc serve-nginx

curl <http://10.100.166.73>

working with node port services

kubectl get rs --all-namespaces

kubectl delete deployment deploy-nginx --namespace=default

kubectl delete services serve-nginx --namespace=default

vi deployment.yaml

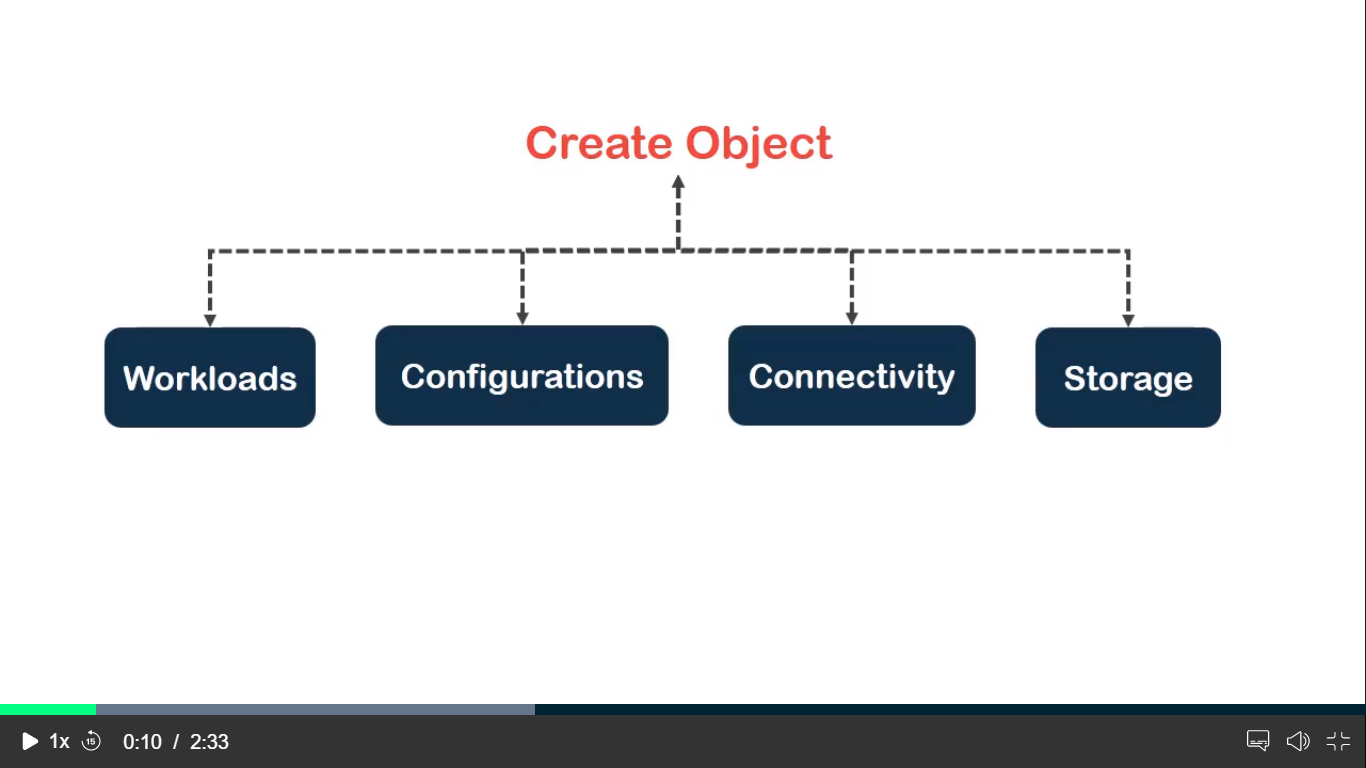
vi serve-nginx.yaml

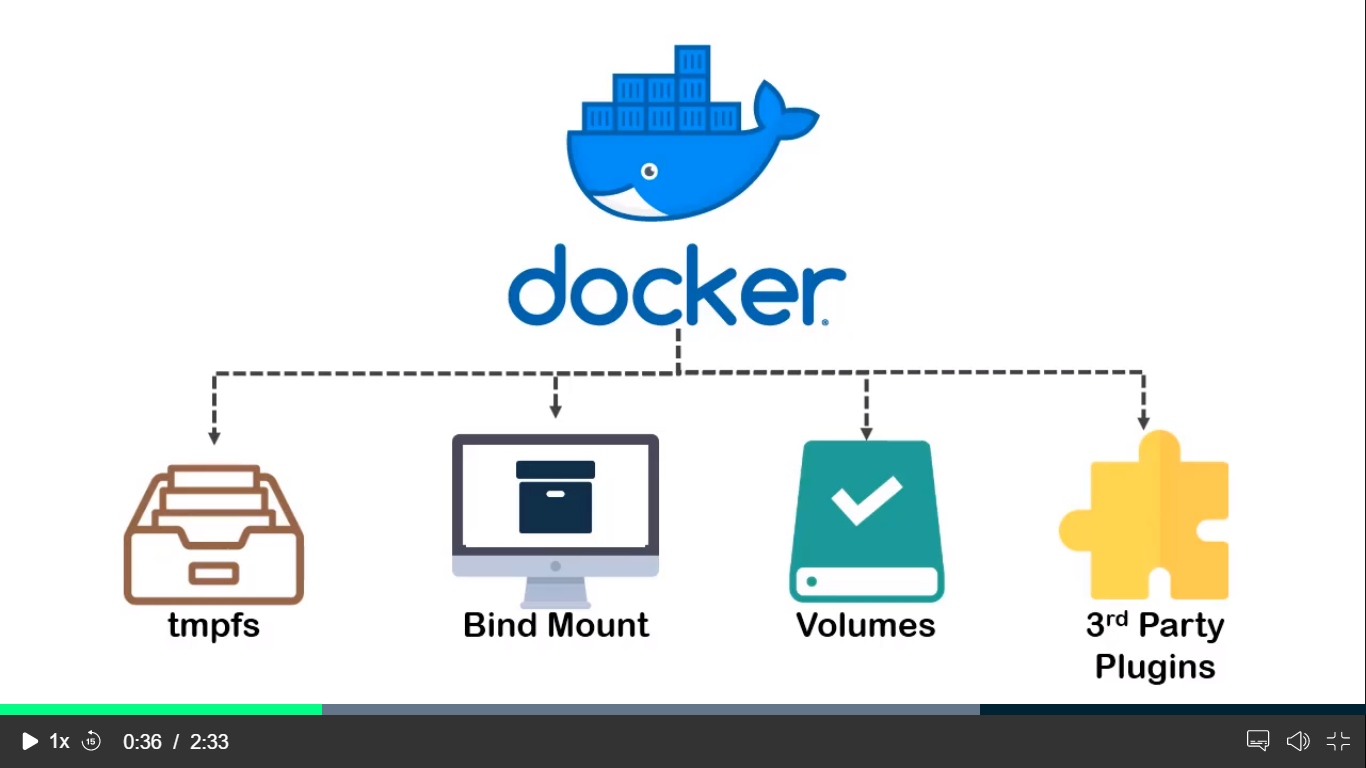
kubectl create -f serve-nginx.yaml

kubectl get svc

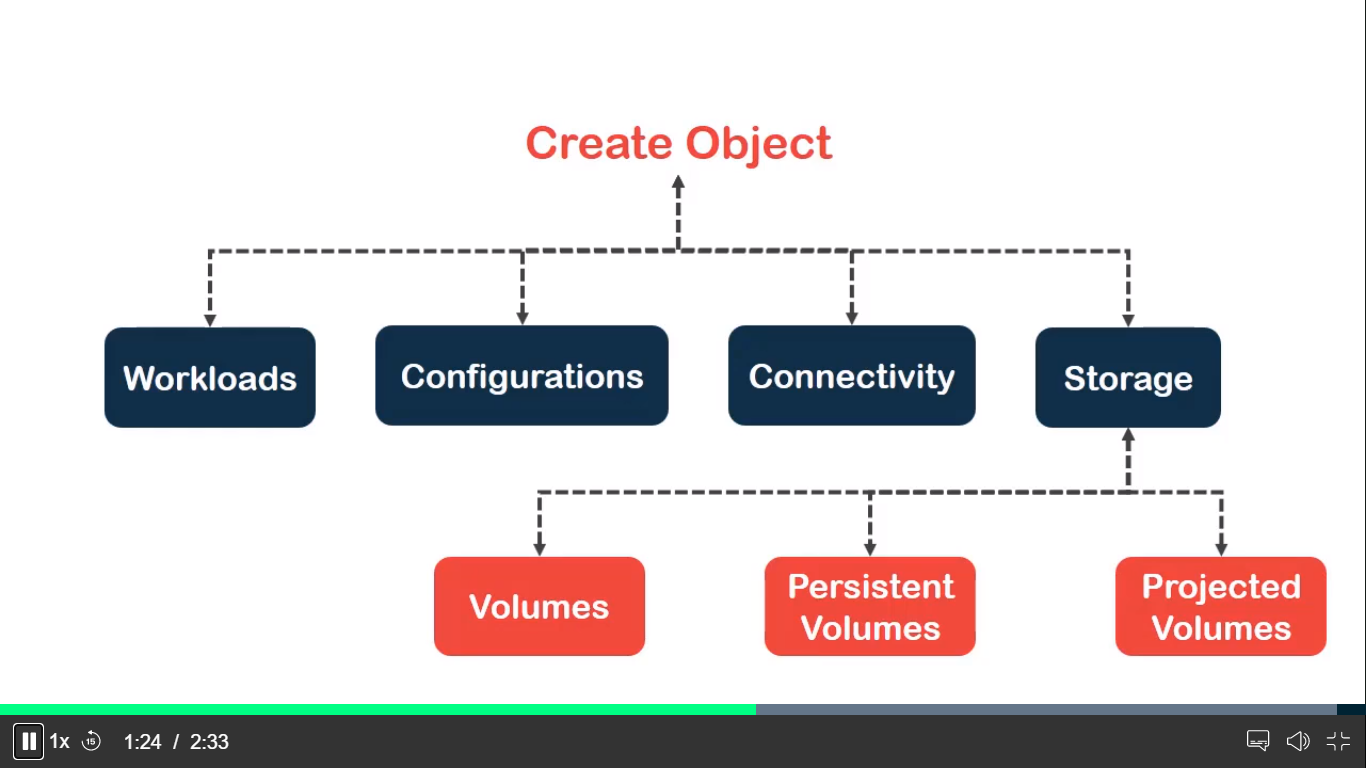
kubectl describe svc serve-nginx

curl -k 34.125.216.3:30778





Kubernets stores aws elastic block storge



kubectl delete replicasets frontend --namespace=default

Mounting volume to a pod

vi redis-pod.yaml

kubectl create -f redis-pod.yaml

kubectl get pods

kubectl exec -it redis-pod -- /bin/bash

apt-get install procps

ps aux

kill 1

kubectl get pods

kubectl exec -it redis-pod -- /bin/bash

vi redis-vol.yaml

kubectl create -f redis-vol.yaml

kubectl describe pods redis-pod

echo -n "admin" > ./username.txt

echo -n "21ssak43adhpn" > ./password.txt

kubectl create secret generic user --from-file=./username.txt

kubectl create secret generic pswd --from-file=./password.txt

kubectl get secrets

kubectl describe secret pswd

vi projected-volume.yaml

kubectl create -f projected-volume.yaml

kubectl get pods

kubectl exec -it projectedvol-pod -- /bin/sh

ls /projected-volume/

cat /projected-volume/password.txt

kubectl create secret generic mysql-pswd --from-literal=password=xyz@123

kubectl get secrets

vi mysql-db.yaml

kubectl apply -f mysql-db.yaml

kubectl get pods

kubectl get deployments

kubectl describe deployments mysql-db

kubectl get svc

kubectl describe service mysql-db

vi wordpress-frontend.yaml

kubectl apply -f wordpress-frontend.yaml (loadbalancer frontend IP)

kubectl get pods

kubectl get deployments

kubectl get svc

kubectl describe services wp-frontend

BLACKROCK FINANCIAL SERVICES

Demo node Eviction

Kubectl get pods

Vi nginx.yaml

Kubectl create –f nginx.yaml

Kubectl get rs

Kubectl get pods –o wide

Kubectl drain node-2

Kubectl drain node-2 –ignore-daemonsets

Kubectl get nodes

Kubectl get pods –o wide

Kubectl uncordon node-2

Rolling updates

Kubectl get pods

Vi update-pod.yaml

kubectl create -f update-pod.yaml

kubectl get deployments

kubectl describe deployments deploy-nginx

kubectl set image deployment/nginx-deployment nginx=nginx:1.9.1

kubectl rollout status deployment/nginx-deployment

kubectl set image deployment/nginx-deployment nginx=nginx:alpine

kubectl rollout history deployment/nginx-deployment

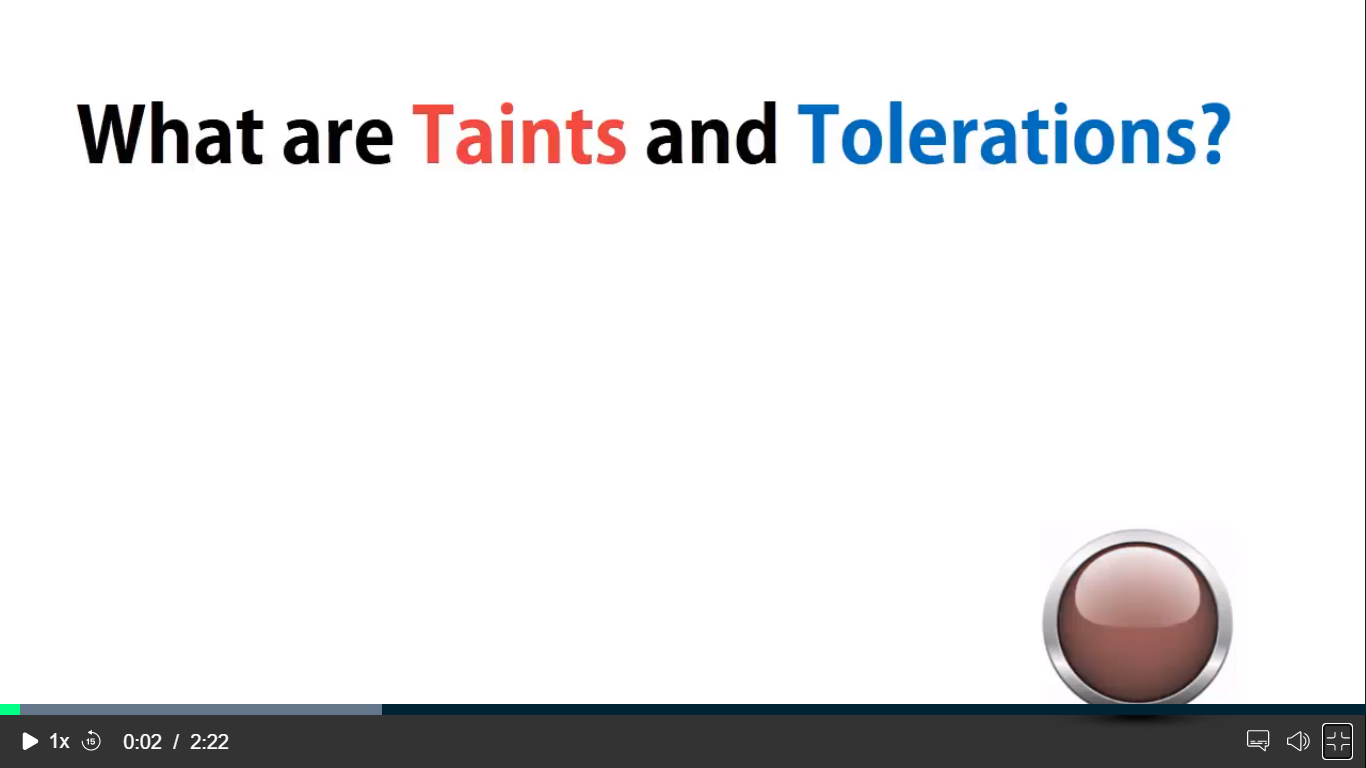
kubectl rollout history deployment/nginx-deployment --revision=2

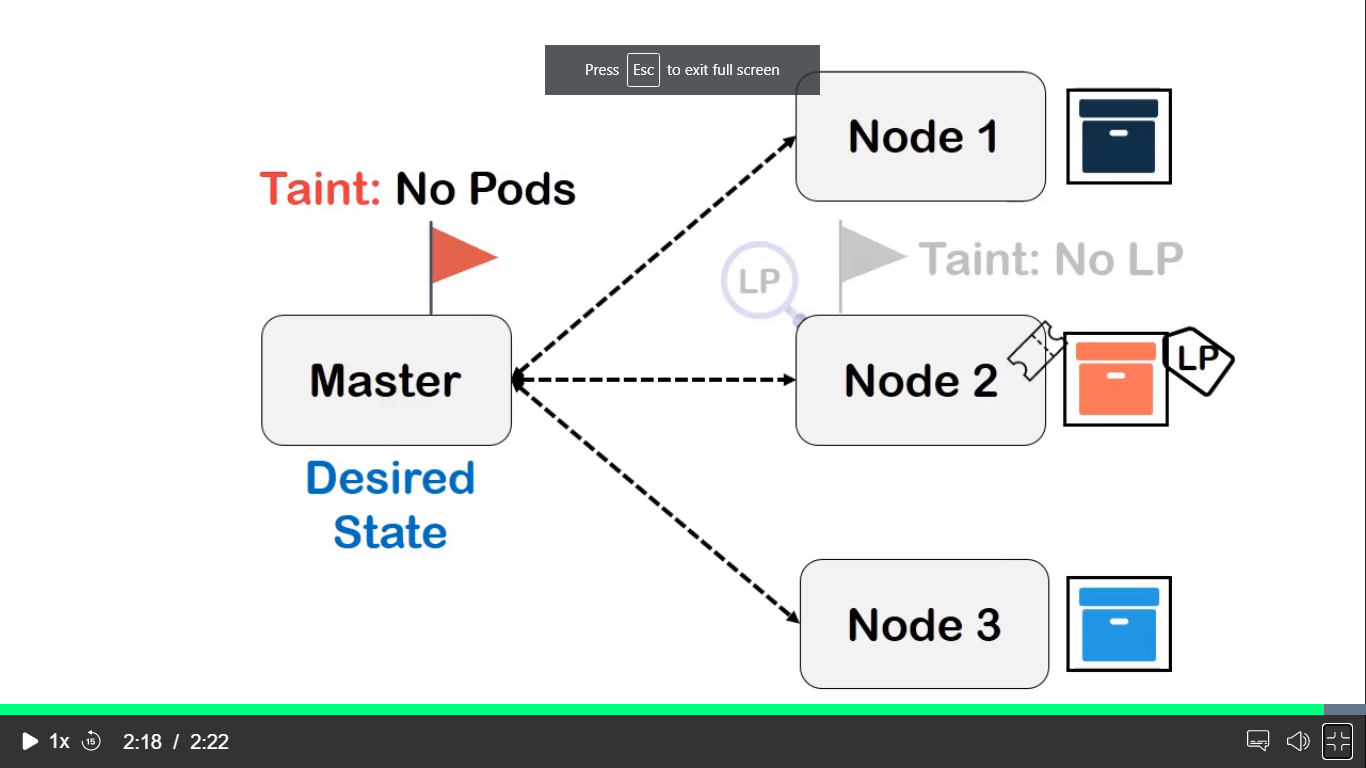
kubectl rollout undo deployment/nginx-deployment

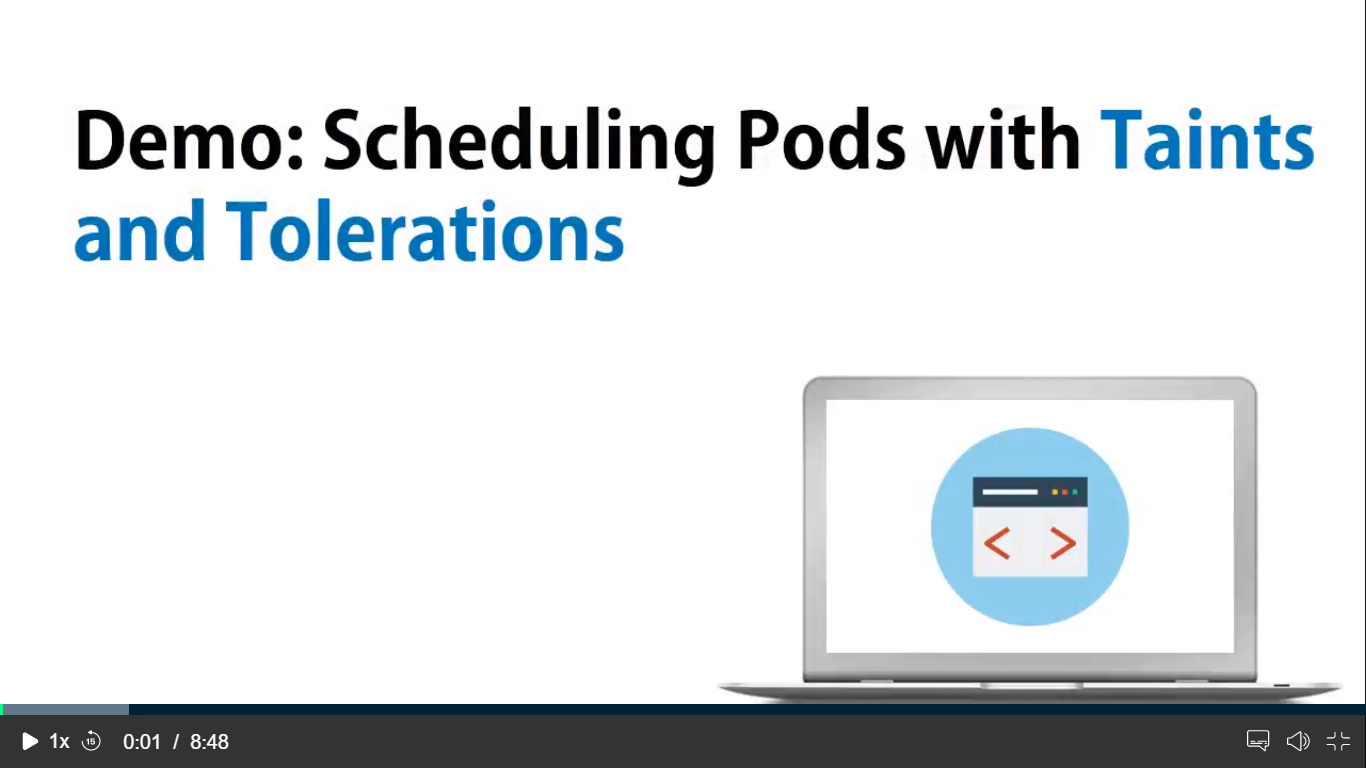
kubectl get pods -o wide

kubectl rollout undo deployment/nginx-deployment --to-revision=2

what are Taints and Tolerations ?







kubectl get nodes -o wide

kubectl get nodes --show-labels

kubectl label nodes node-3 disktype=ssd

kubectl describe nodes node-3

kubectl get pods -o wide

vi test-pod.yaml

kubectl create -f test-pod.yaml

kubectl get pods -o wide

kubectl taint nodes node-3 disk=pd:NoSchedule

kubectl run hdd --image=nginx --replicas=6 --port=8080 -l disk=pd

kubectl taint nodes node-3 disk:NoSchedule-

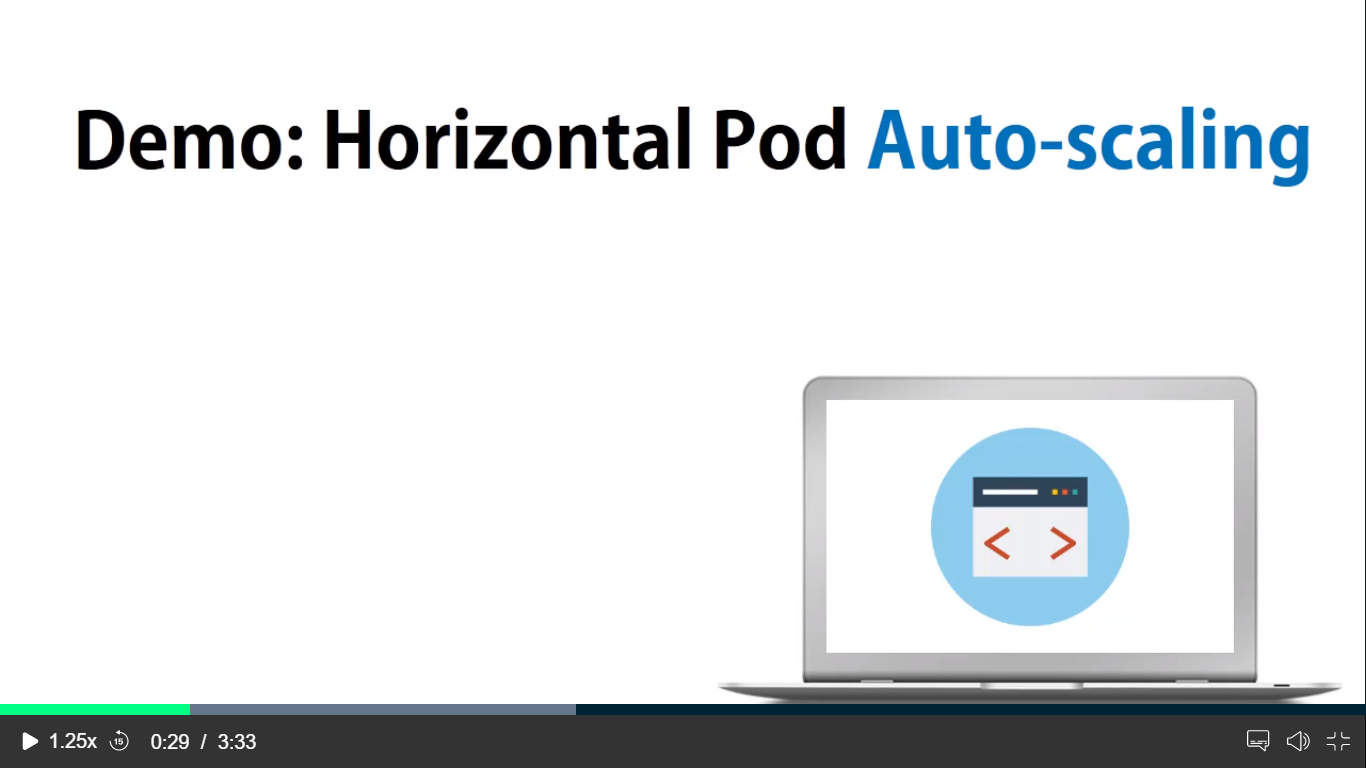
kubectl delete deployments hdd

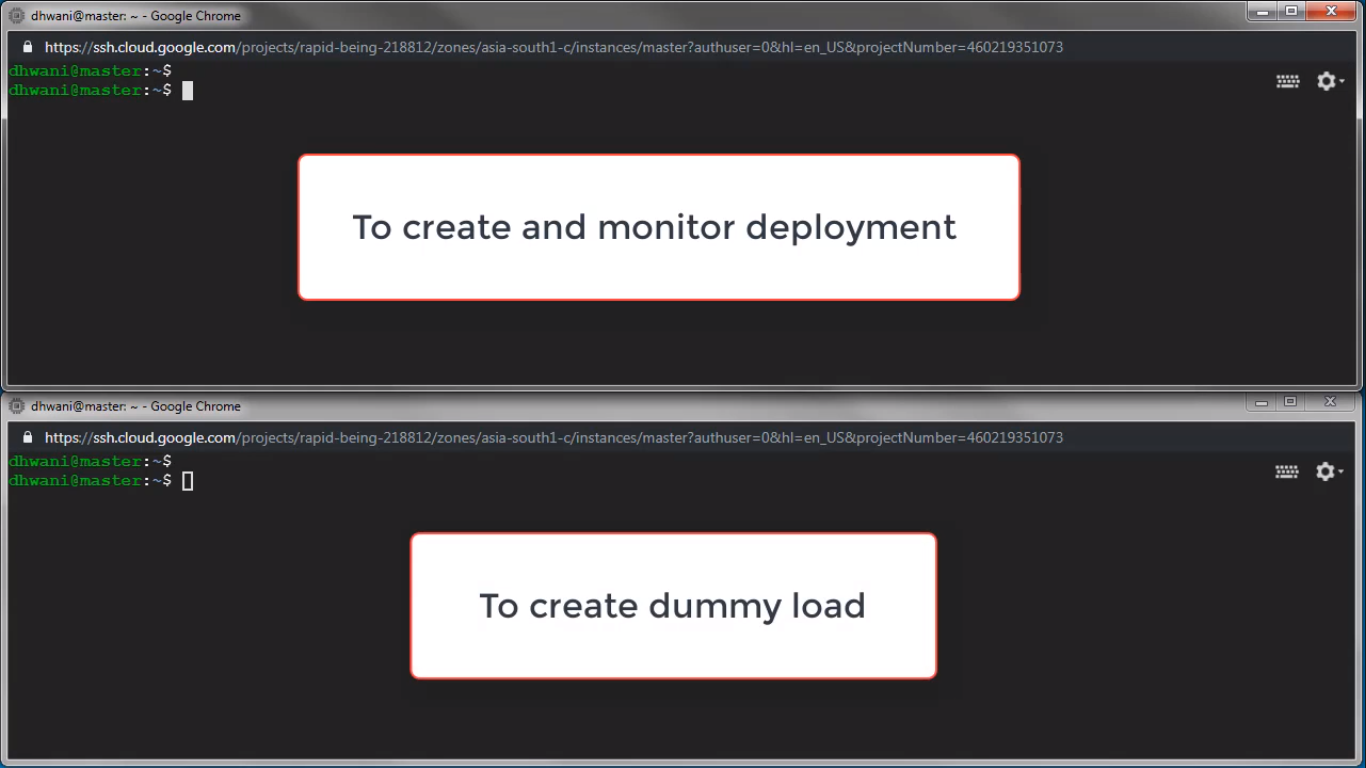
kubectl run hdd --image=nginx --replicas=6 --port=8080 -l disk=pd

kubectl get pods -o wide

HPA

Horizontal pod auto-scaling





kubectl run php-apache --image=k8s.gcr.io/hpa-example --requests=cpu=200m --expose --port=80

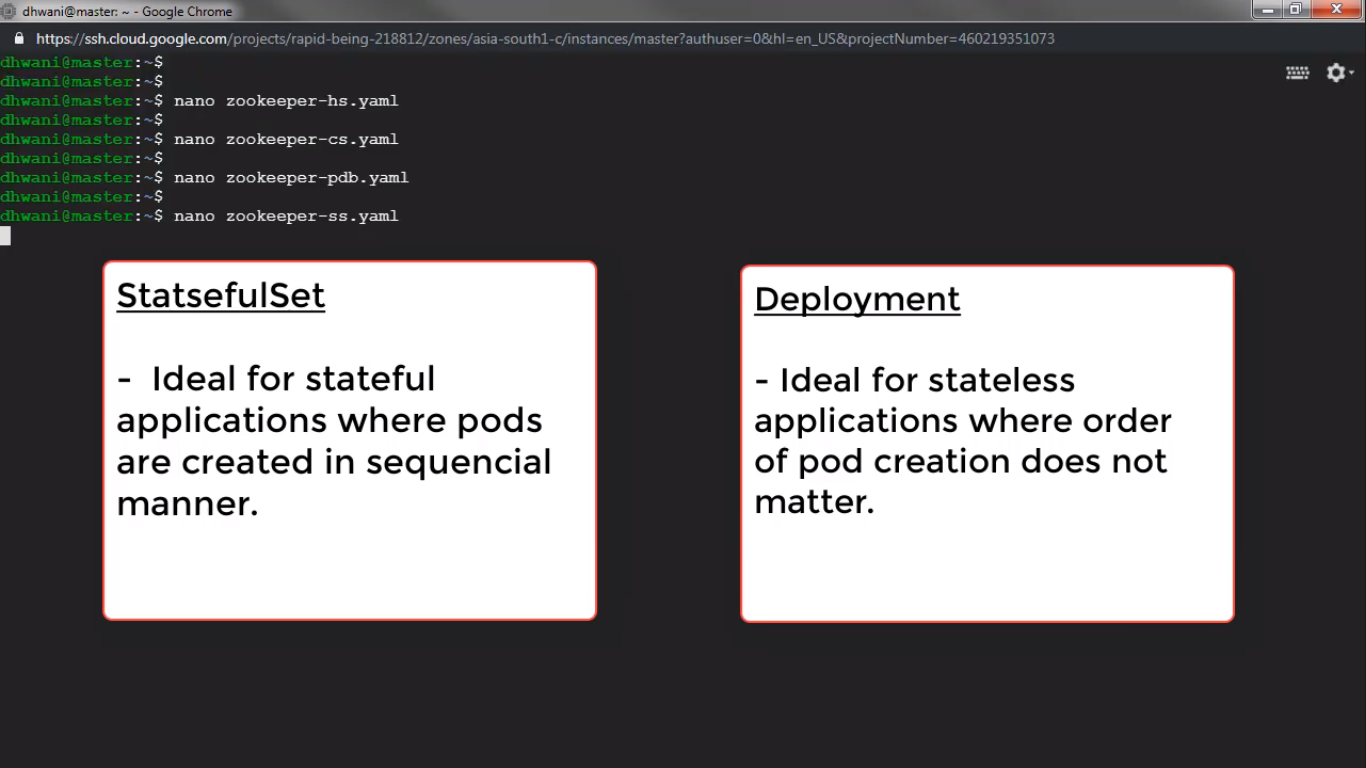
kubectl run -i --tty load-generator --image=busox /bin/sh

kubectl autoscale deployment php-apache --cpu-percent=50

--min=1 --max=5

kubectl get hpa

APACHE ZOOKEEPER ON K8S ( zookeeper cluster)



Vi zookeeper-hs.yaml

Vi zookeeper-cs.yaml

vi zookeeper-pdb.yaml (pod distribution budget)

vi zookeeper-ss.yaml

kubectl create -f zookeeper-hs.yaml

kubectl create -f zookeeper-cs.yaml

kubectl create -f zookeeper-pdb.yaml

kubectl create -f zookeepers-ss.yaml

kubectl get pods -w -l app=zk

for i in 0 1 2; do kubectl exec zk-$i -- hostname; done

for i in 0 1 2; do echo "myid zk-$i";kubectl exec zk-$i --cat /var/lib/zookeeper/data/myid; done

for i in 0 1 2; do kubectl exec zk-$i -- hostname -f; done

kubectl exec zk-0 -- cat /opt/zookeepers/conf/zoo.cfg

kubectl exec zk-0 zkCli.sh create /hi-from-sender hi-from-receiver

kubectl exec zk-1 zkCli.sh get /hi-from-sender

kubectl delete statefulset zk

kubectl get pods -w -l app=zk