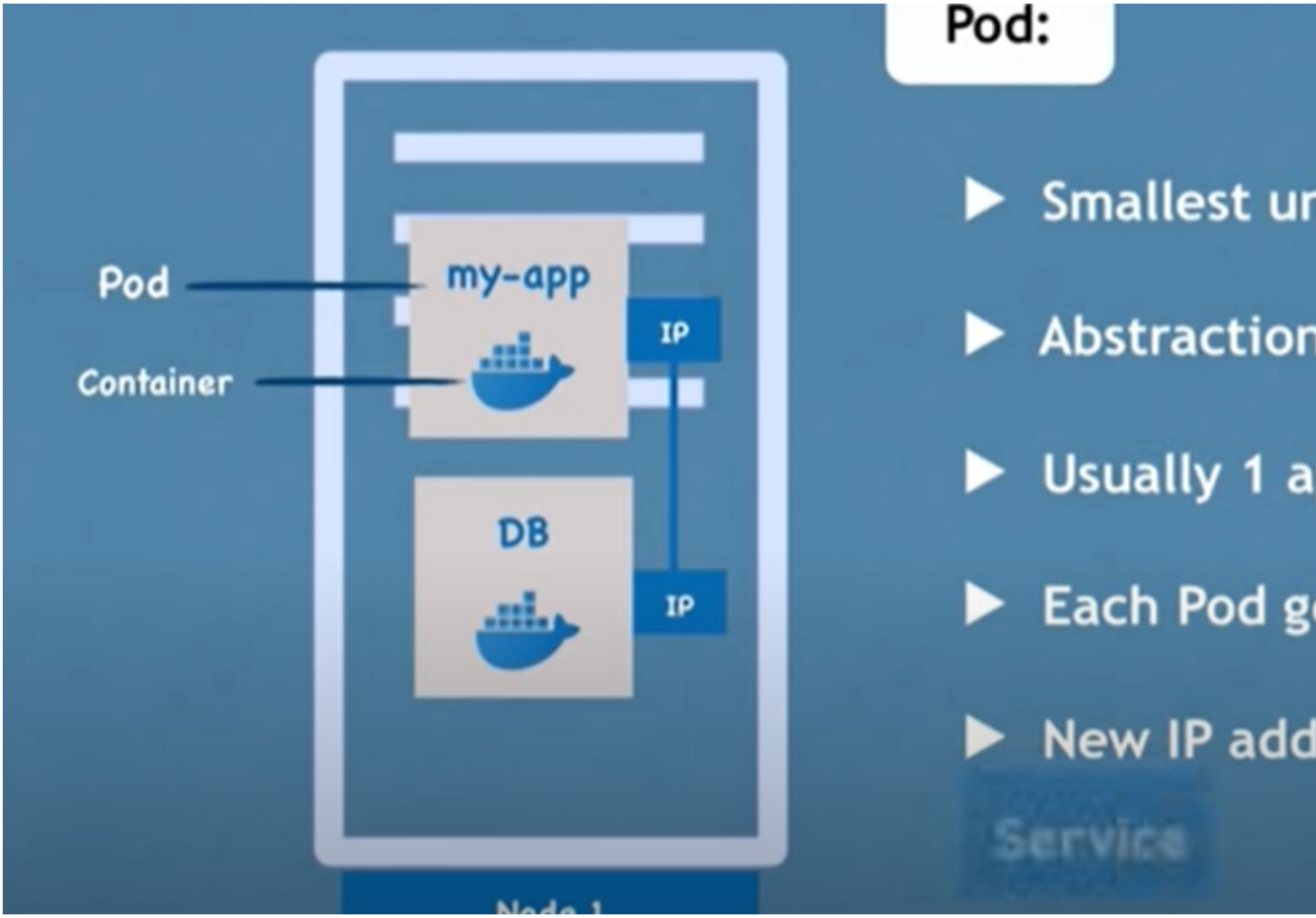
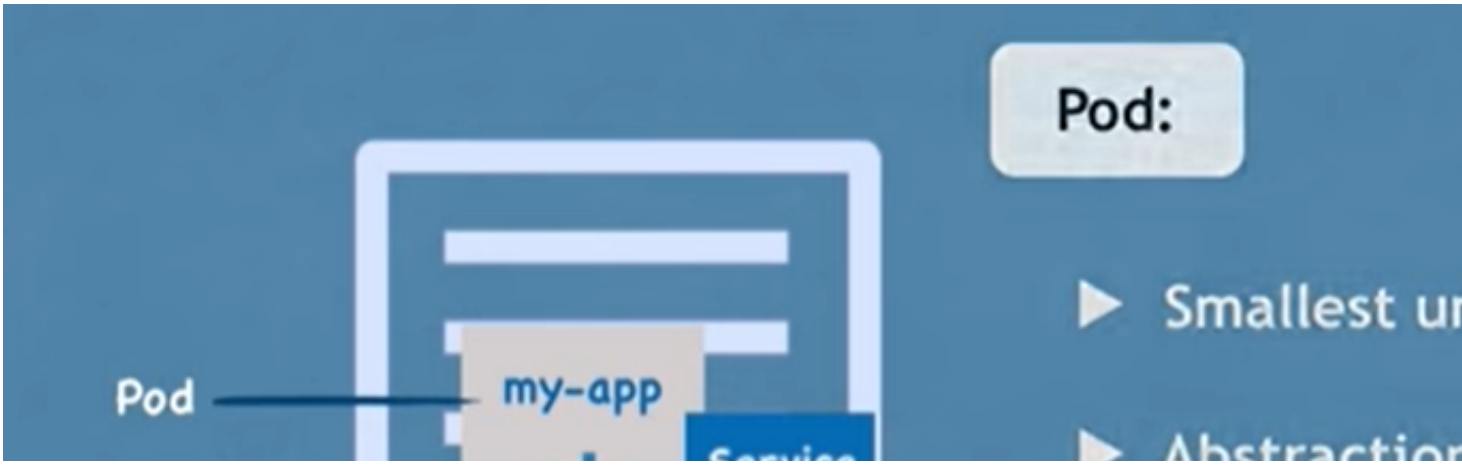
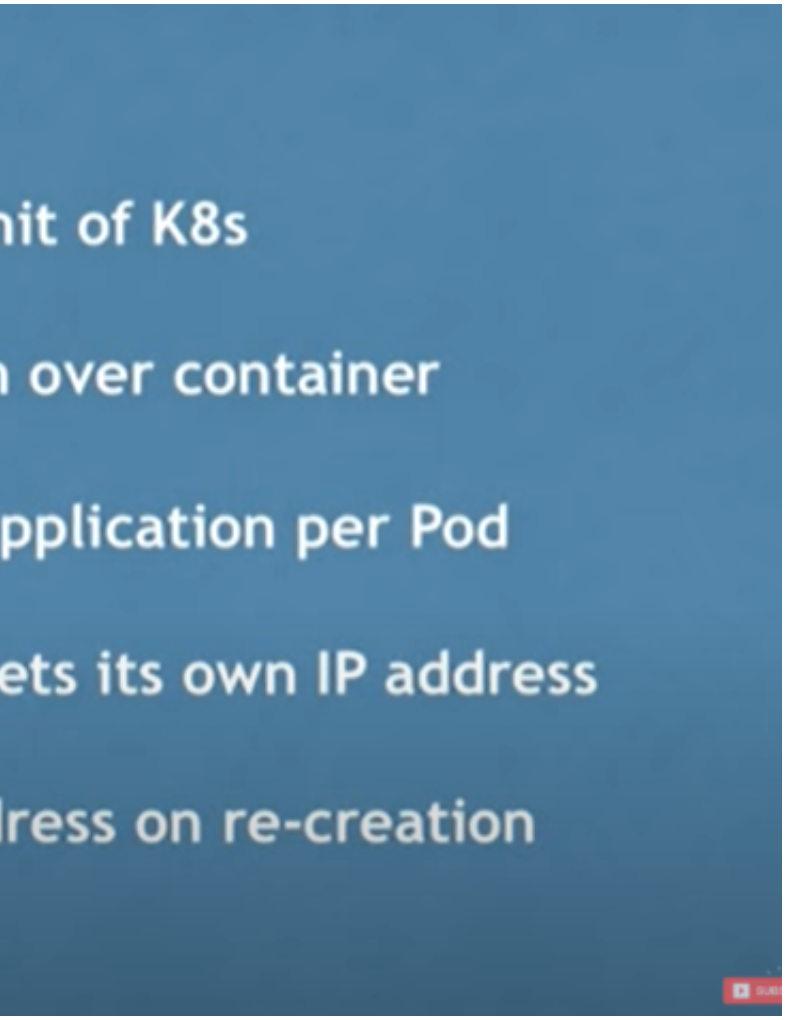
Kubernetes2

Wednesday, 20 March 2024 9:56 AM

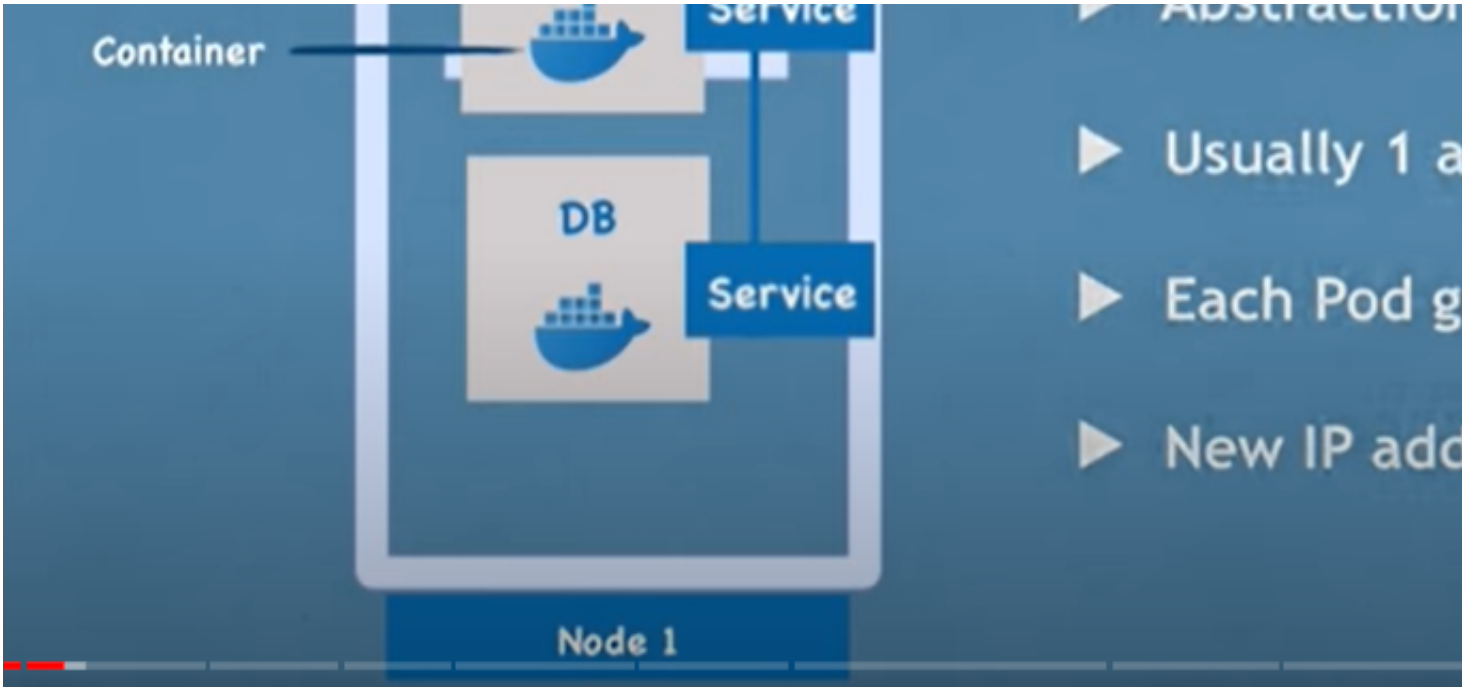


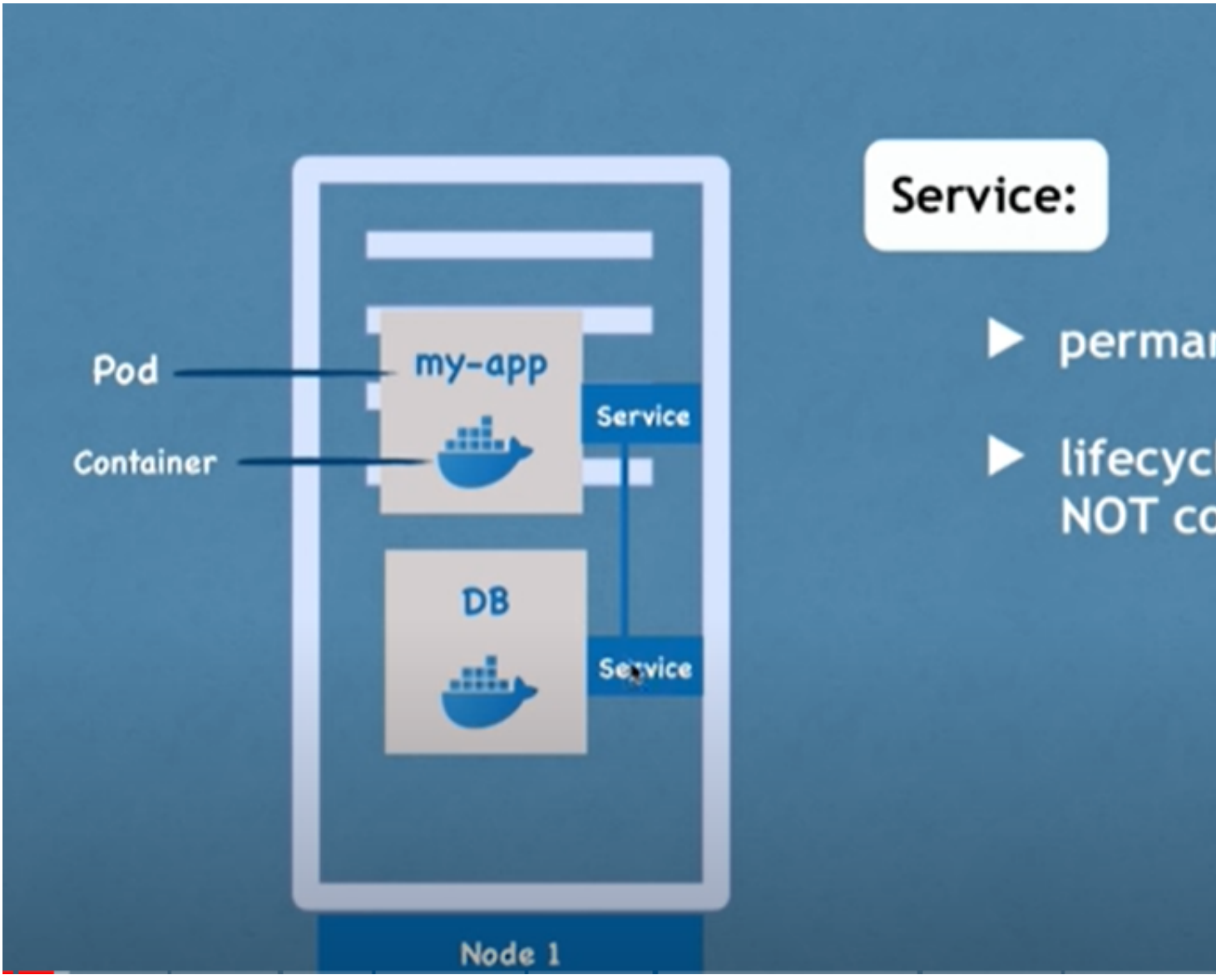
If Pod is down or crashed then new Pod will be created but here again IP address wil change for new Pod to counter this Service is used instead of IP wrt each POD. Thus IP not getting changed every time new POD is created.



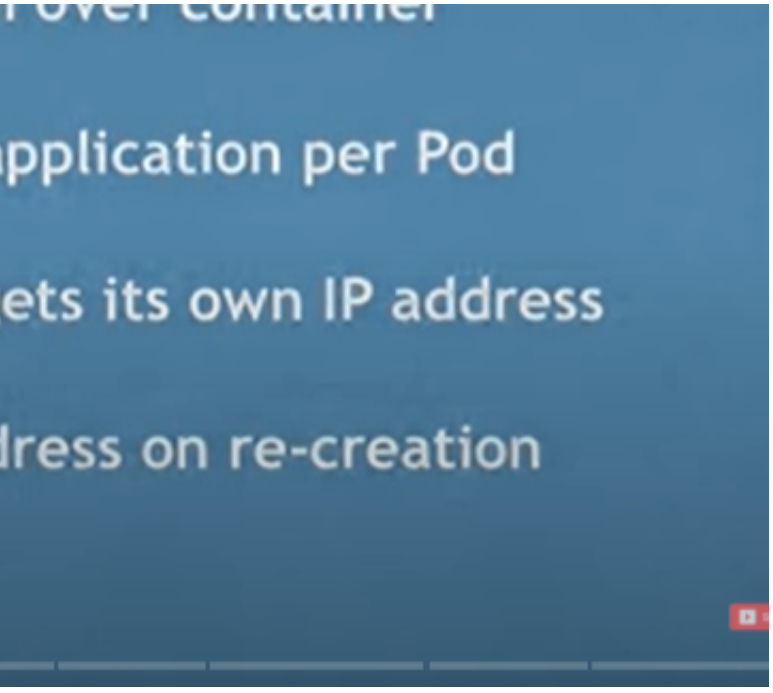

nit of K8s
n over container
pplication per Pod
ets its own IP address
lress on re-creation

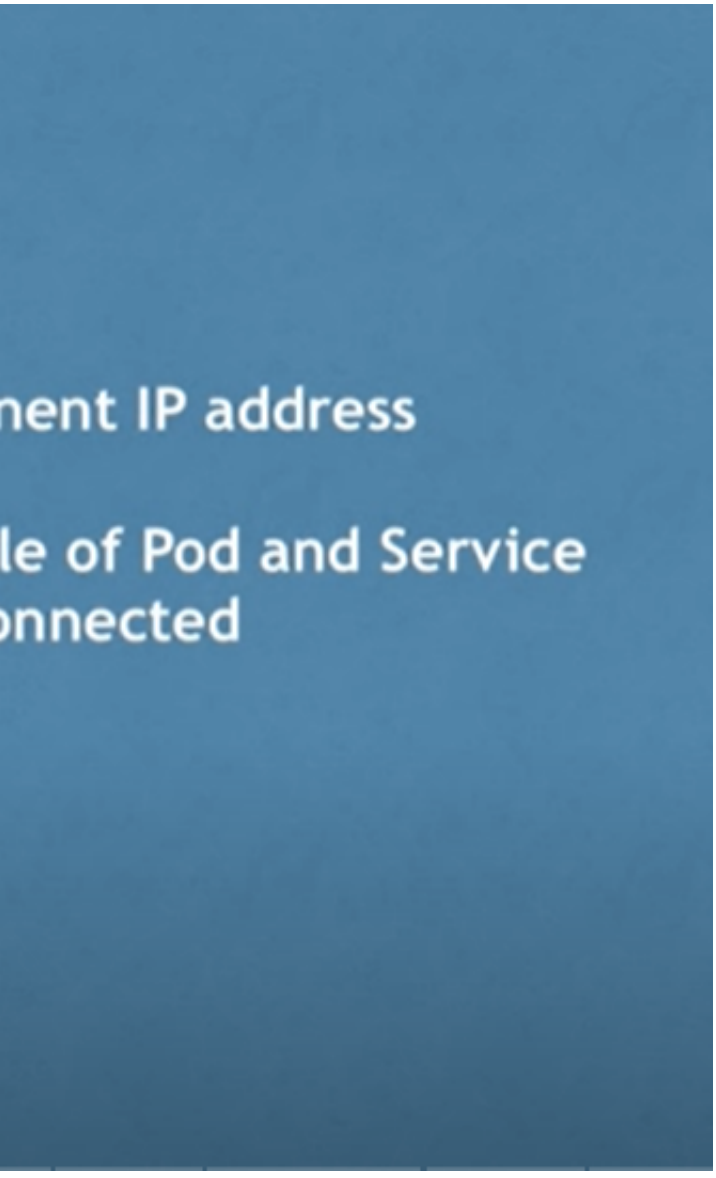

nit of K8s



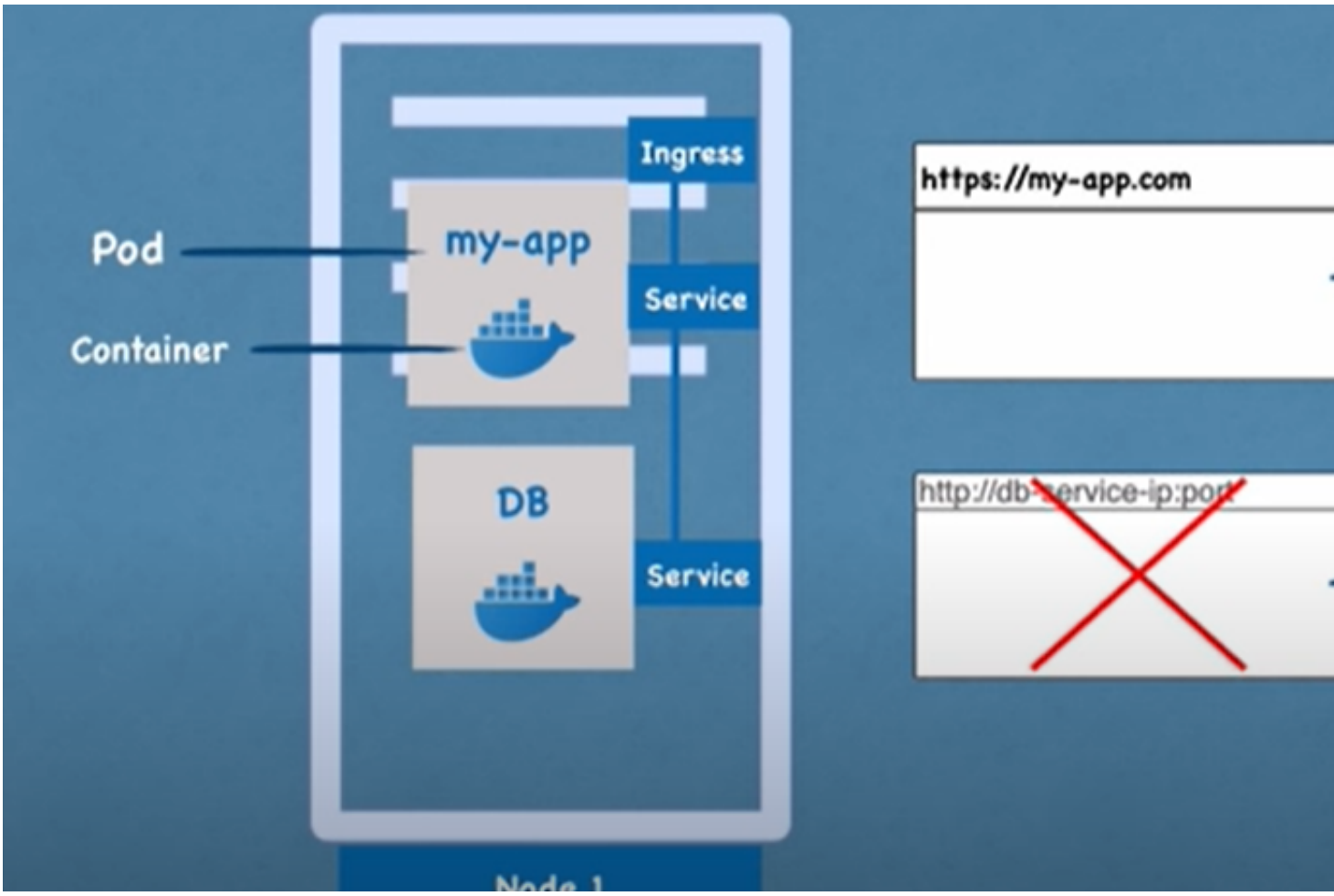


2)Ingress: helps to connect PODS to outside world i.e. outside Node

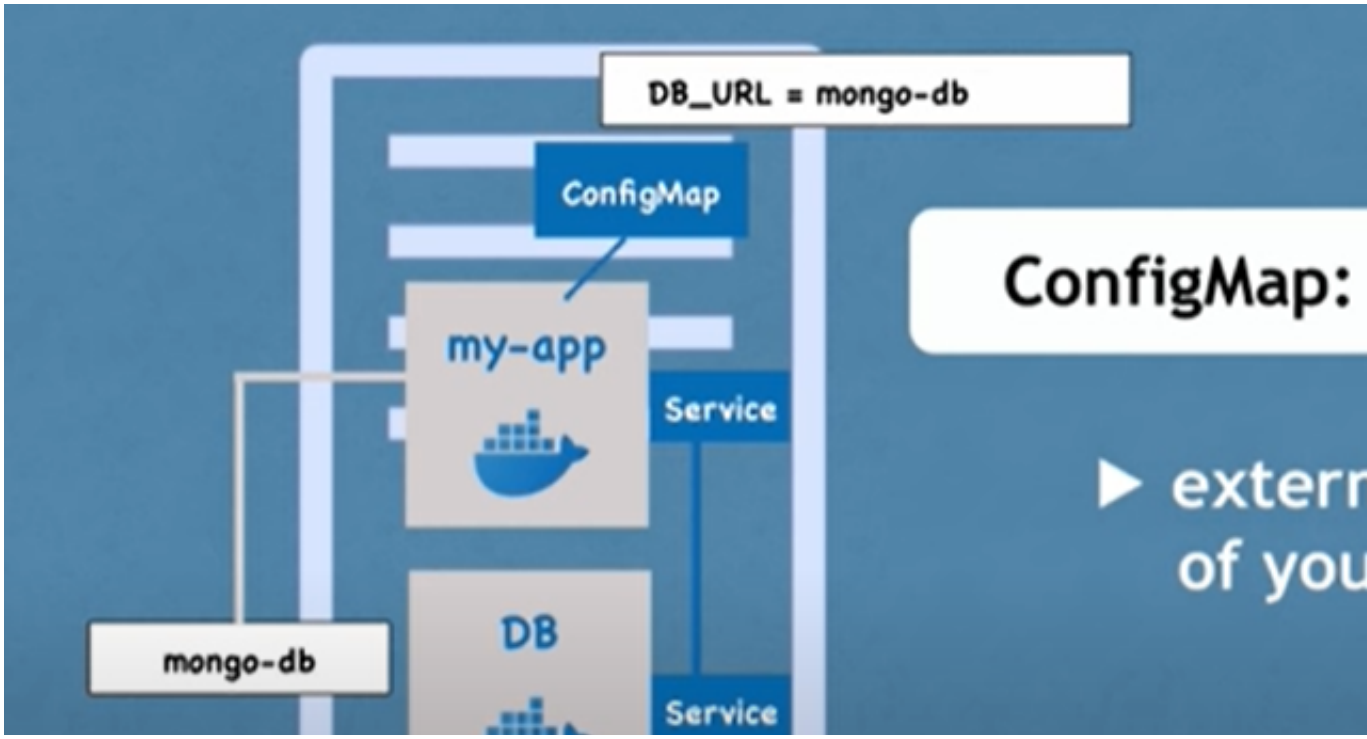

pplication per Pod
ets its own IP address
lress on re-creation

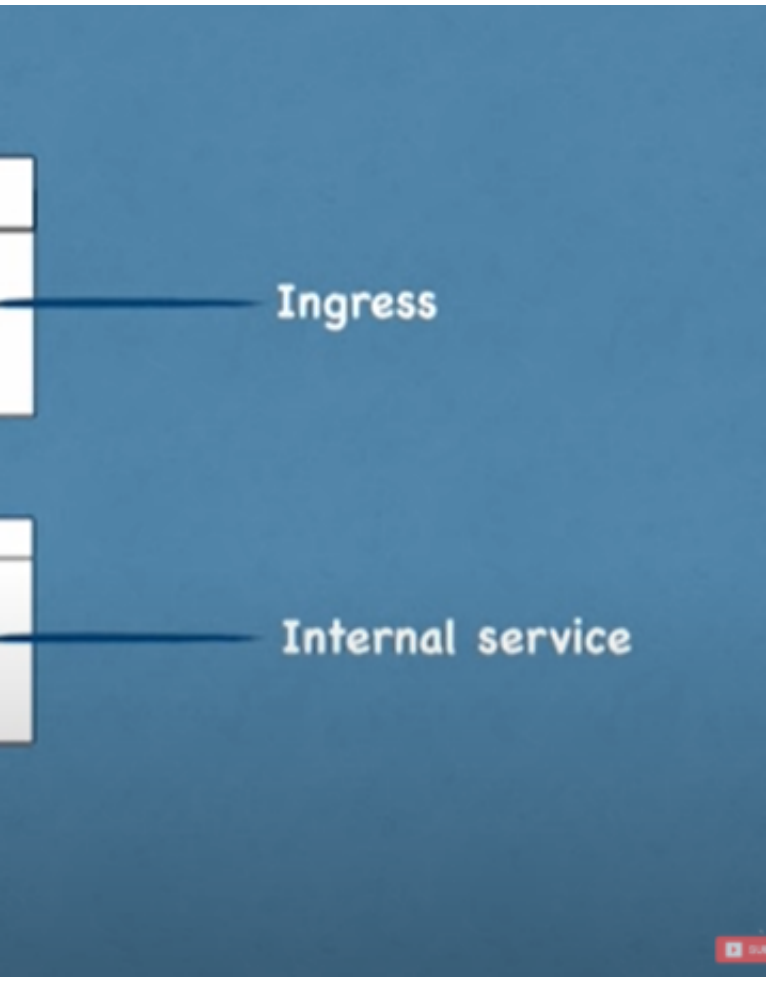

nent IP address
le of Pod and Service
onnected

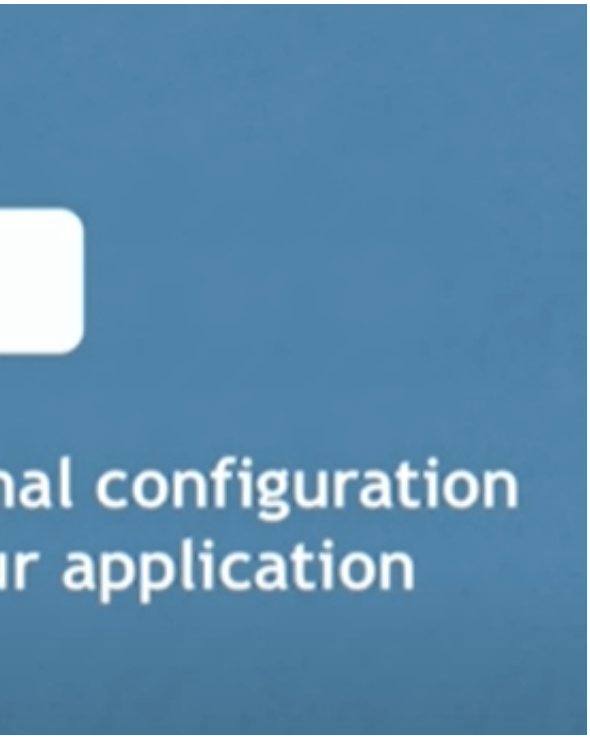
2)Ingress: helps to connect PODS to outside world i.e. outside Node.



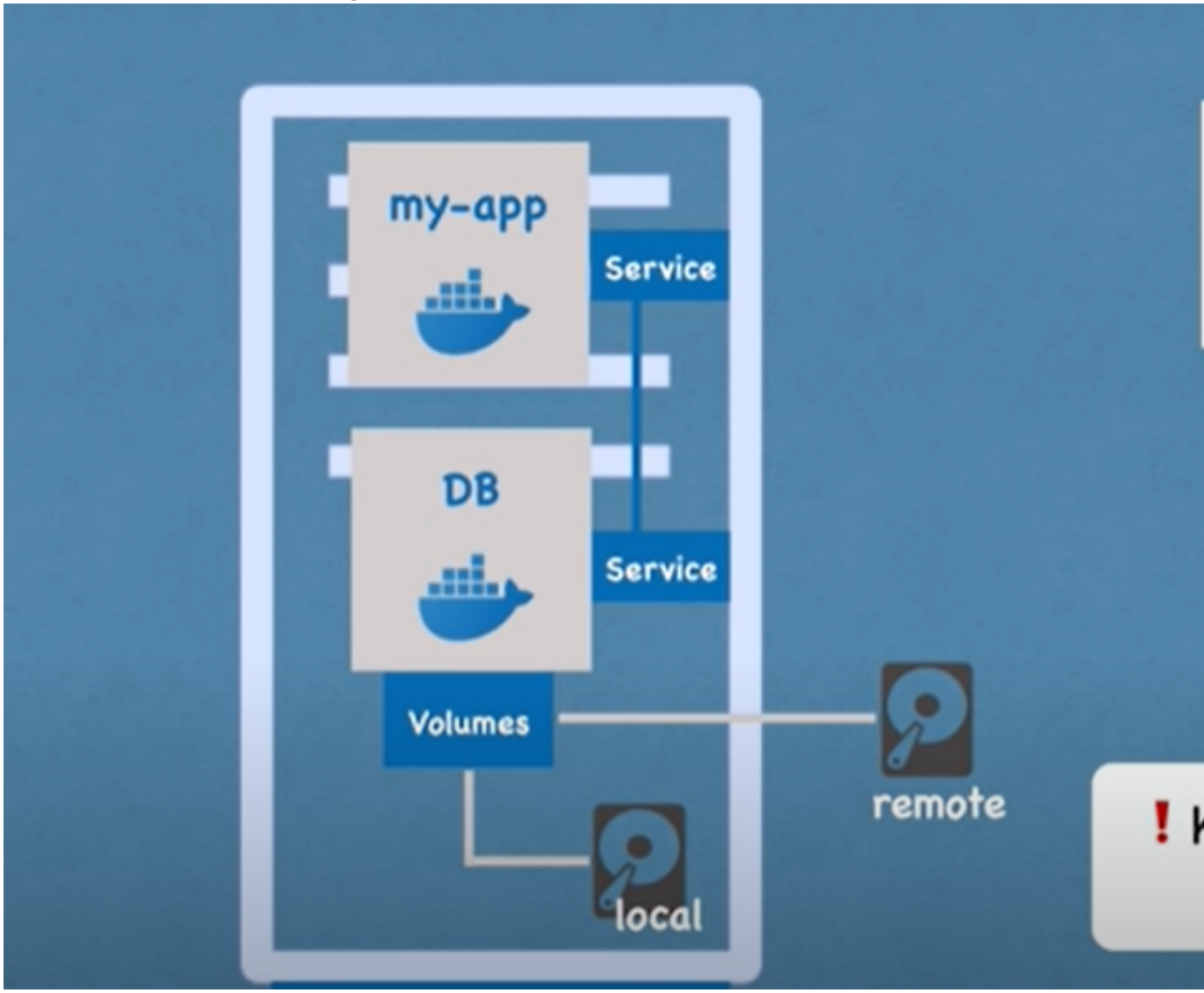
1) ConfigMap and Secret: if DB details are changed then no need to build image again.(Put password in Secret data)



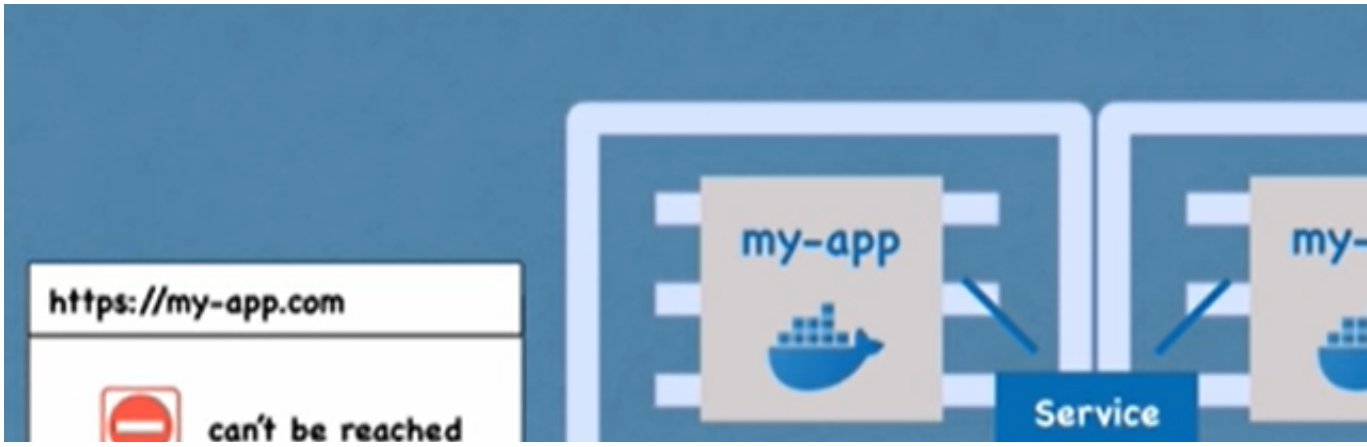

Ingress
Internal service

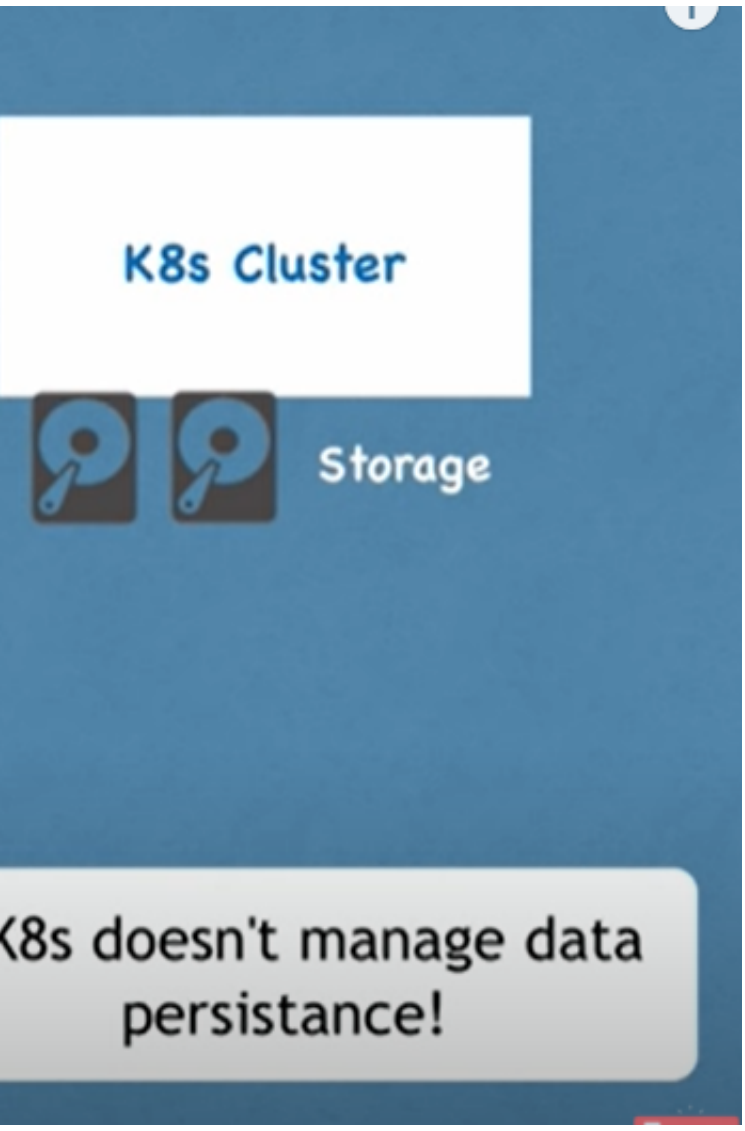

nal configuration
ur application

4)VOLUMES:: Data Storage

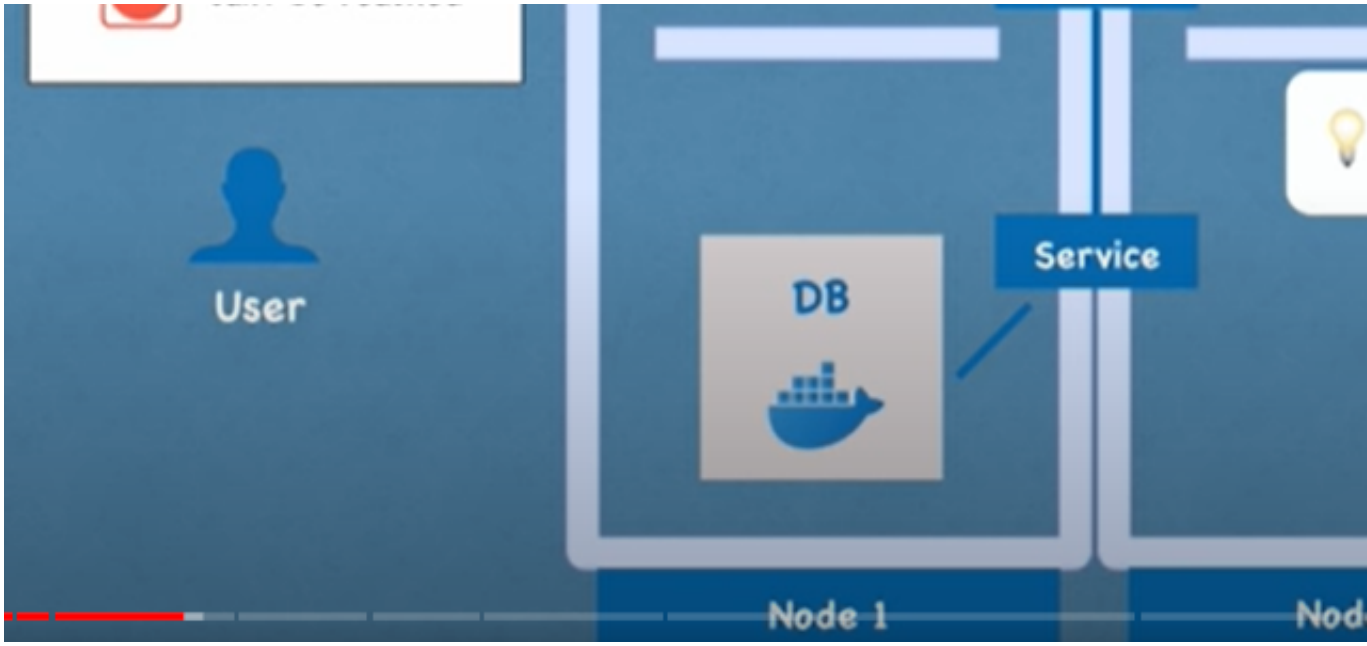


2) Service also act as Load Balancer.

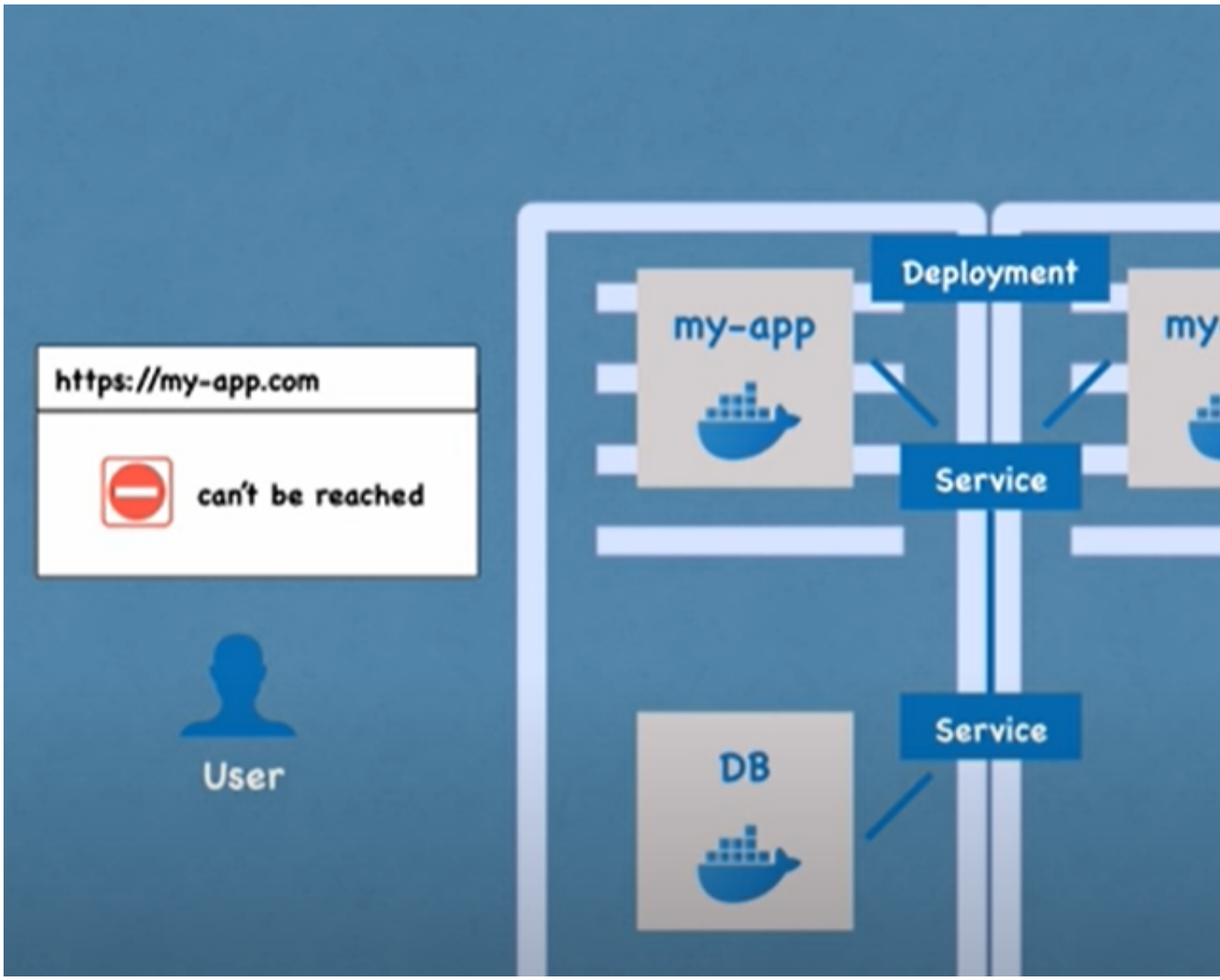


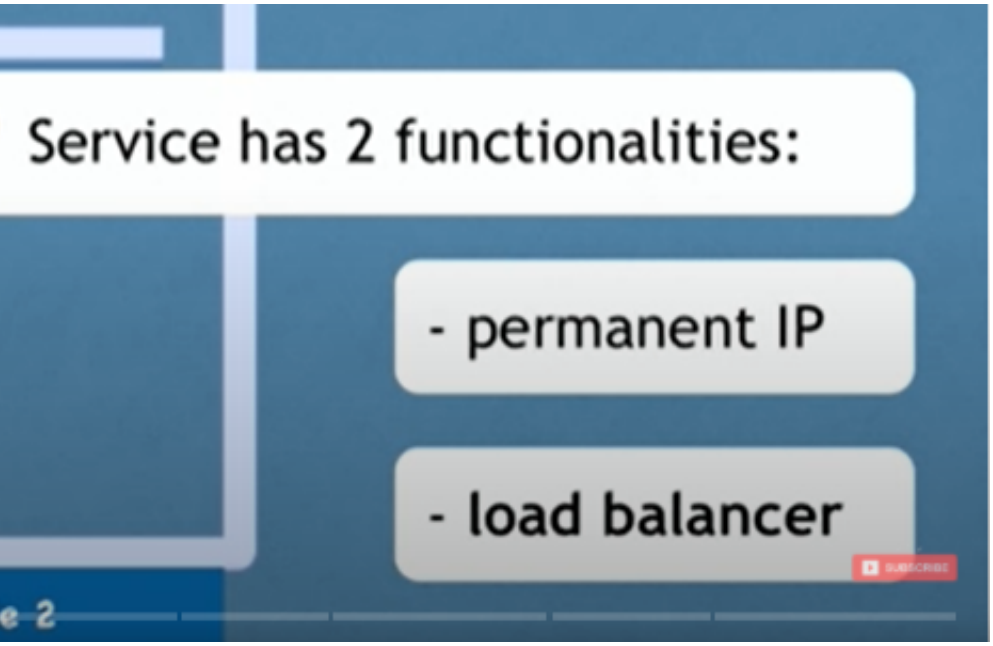

K8s Cluster
Storage
K8s doesn't manage data
persistance!

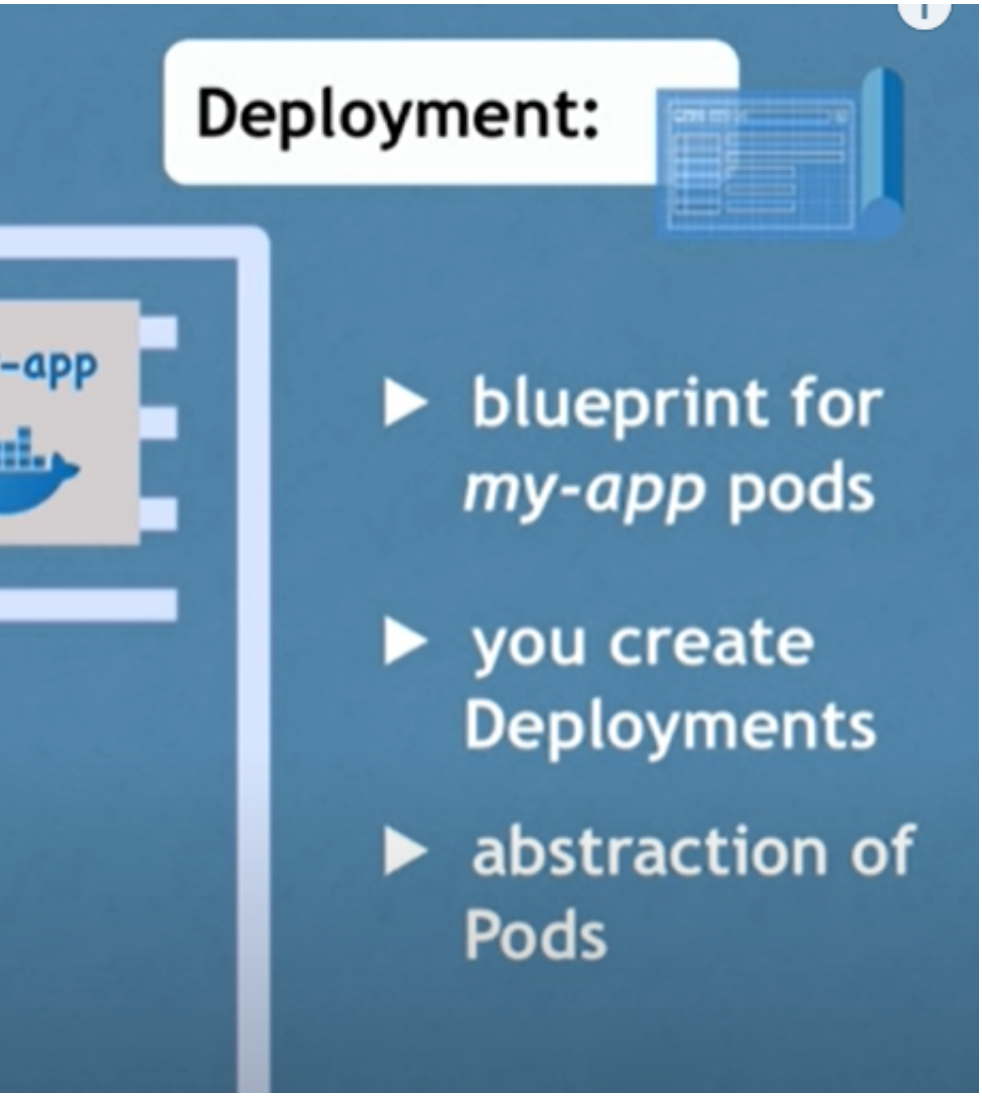

app



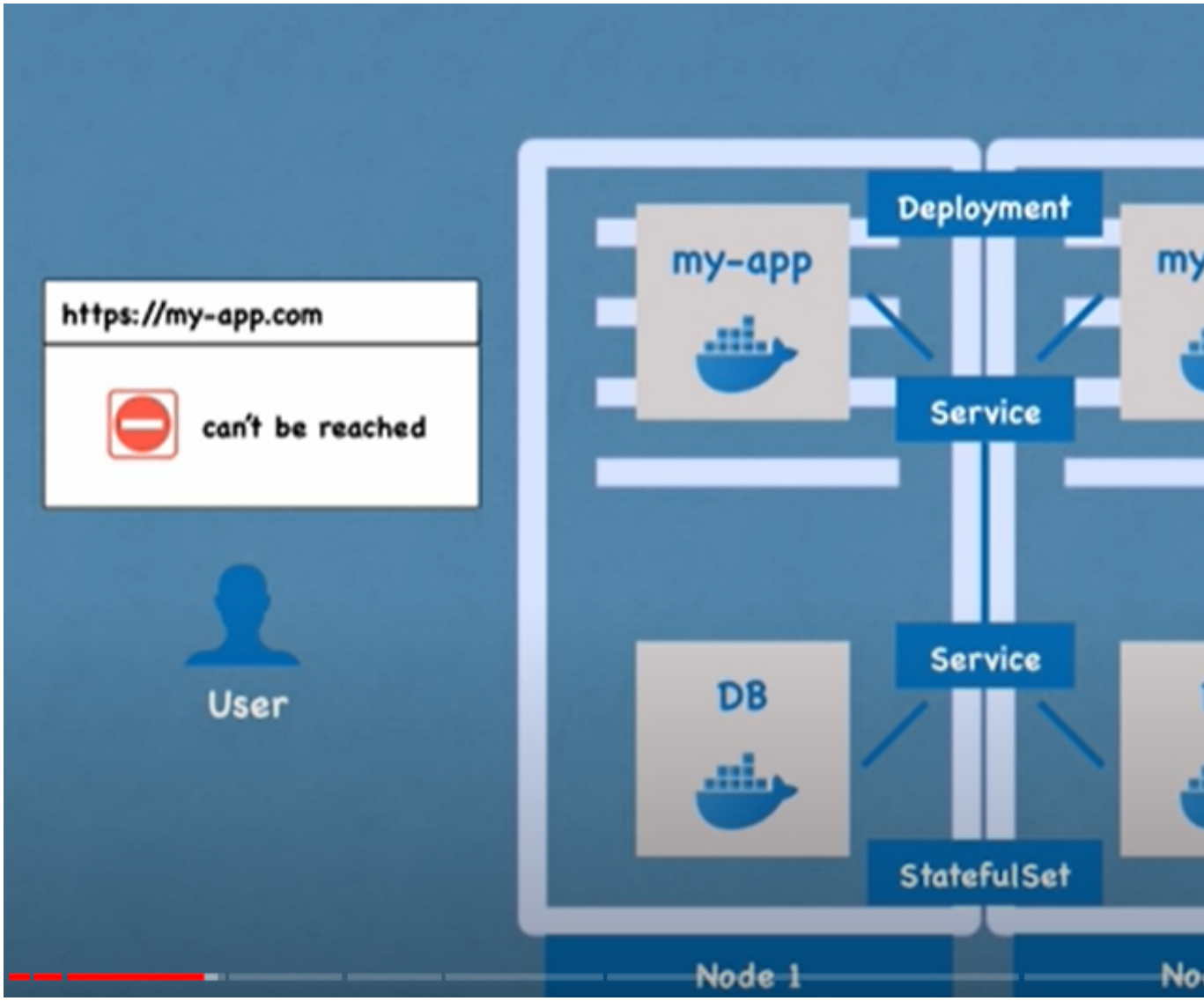
7)We don’t replicate literally i.e. we don’t create same POD on a different Node as a Backup rather we create Deployment where we can define how many replicas of that POD exactly we need.




Service has 2 functionalities:
- permanent IP
- load balancer

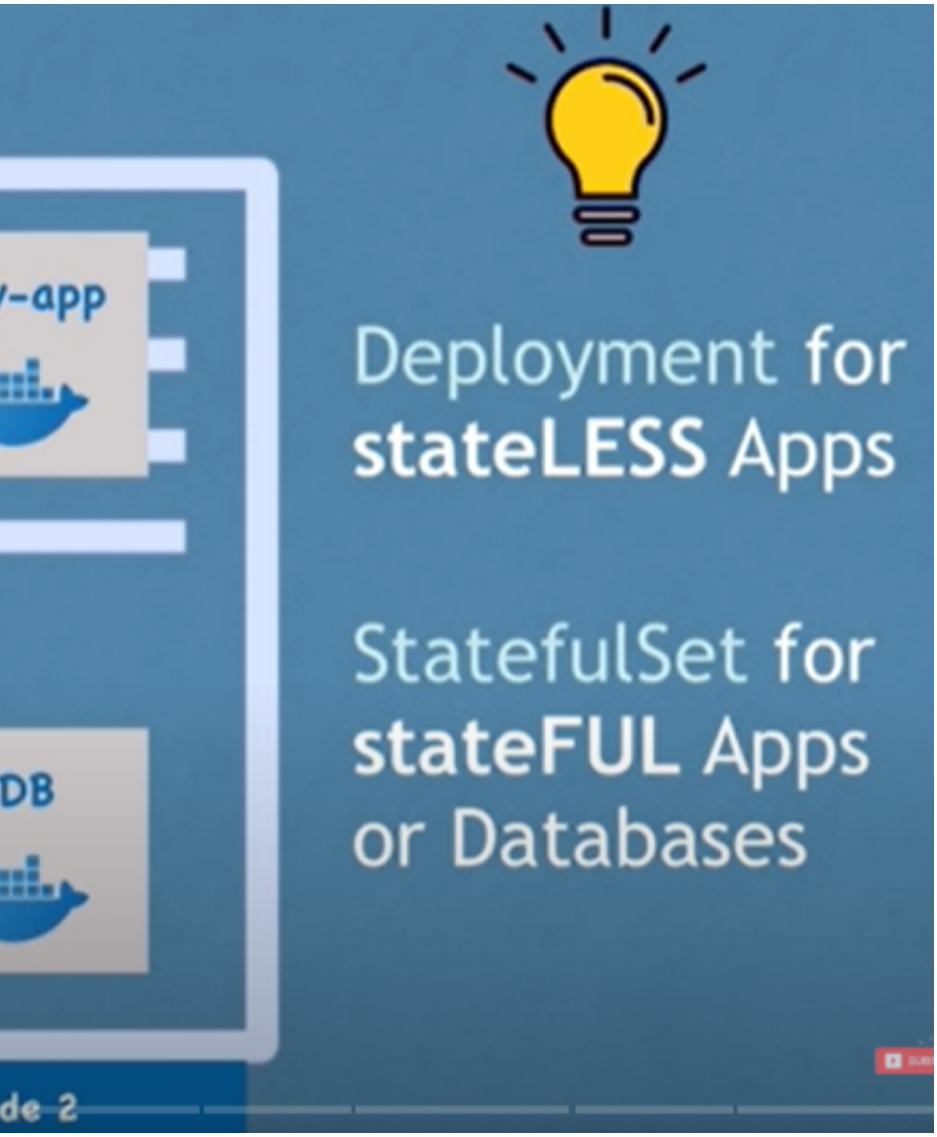

Deployment:
-app
blueprint for my-app pods
you create Deployments
abstraction of Pods

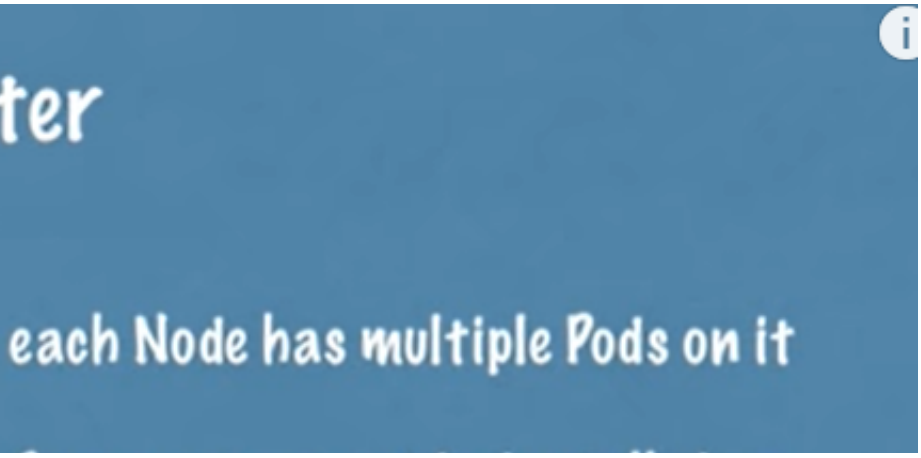
8)

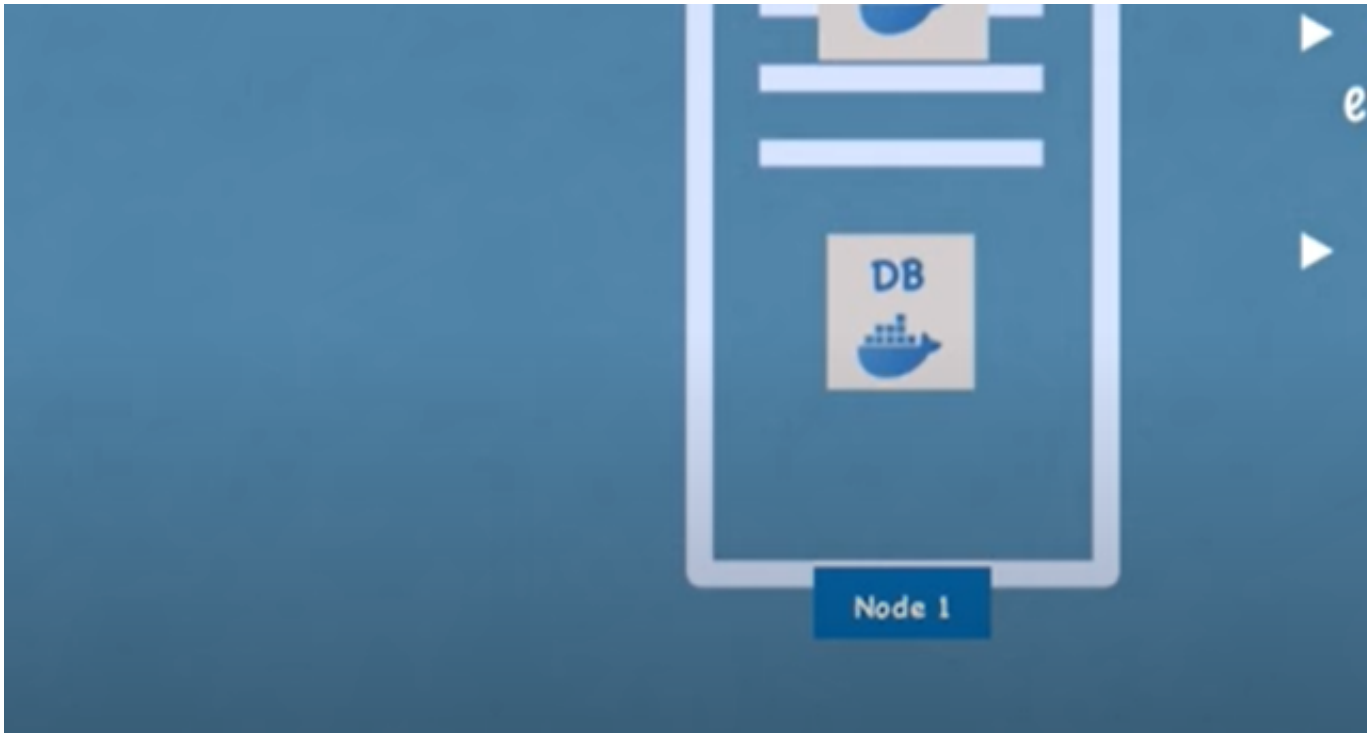


========== Architecture
==========

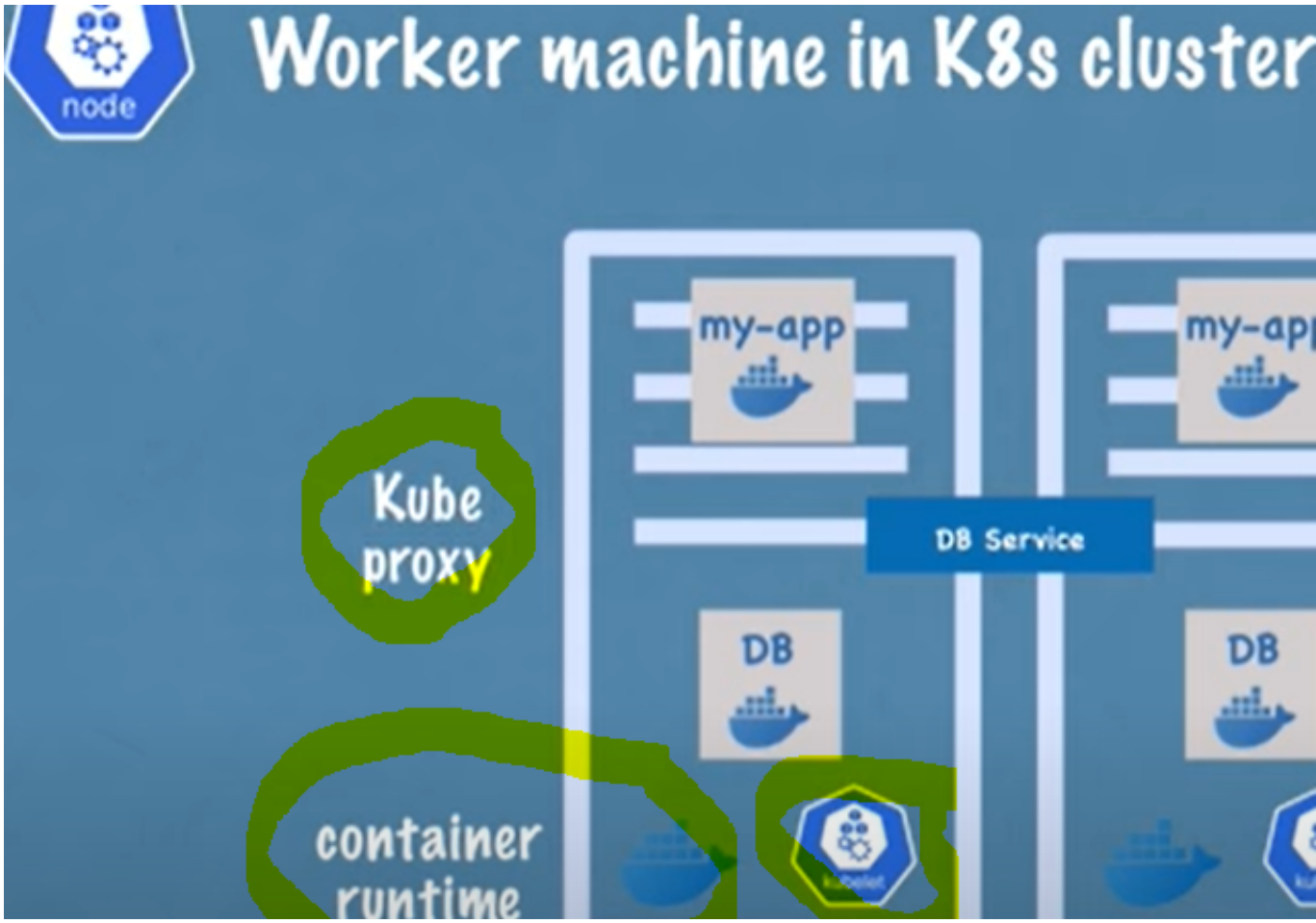
9)Nodes or Worker Nodes:

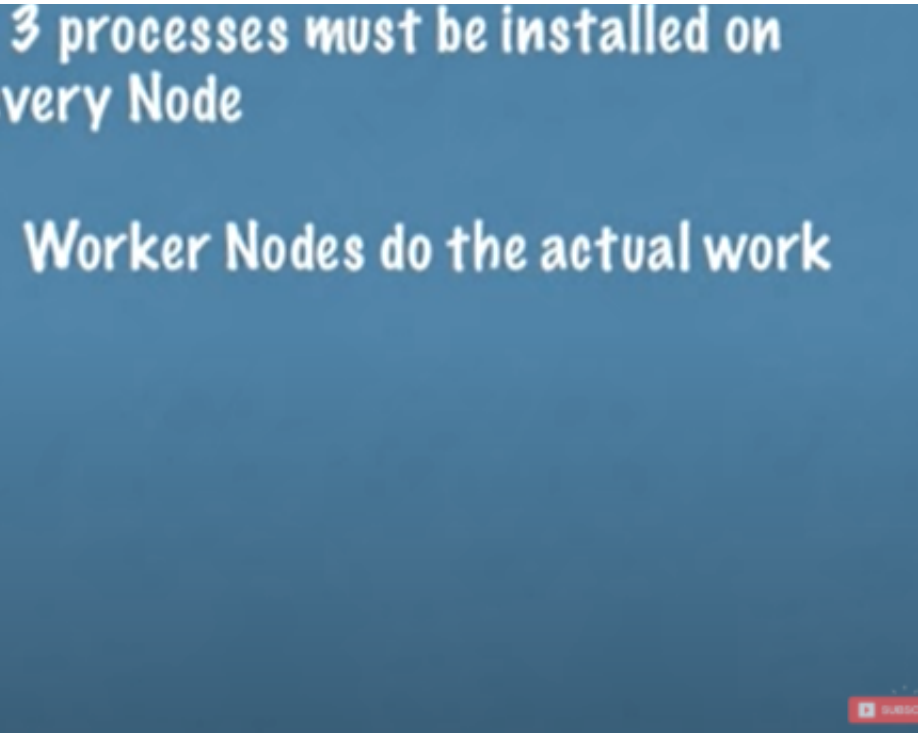

Deployment for
stateLESS Apps
StatefulSet for
stateFUL Apps
or Databases

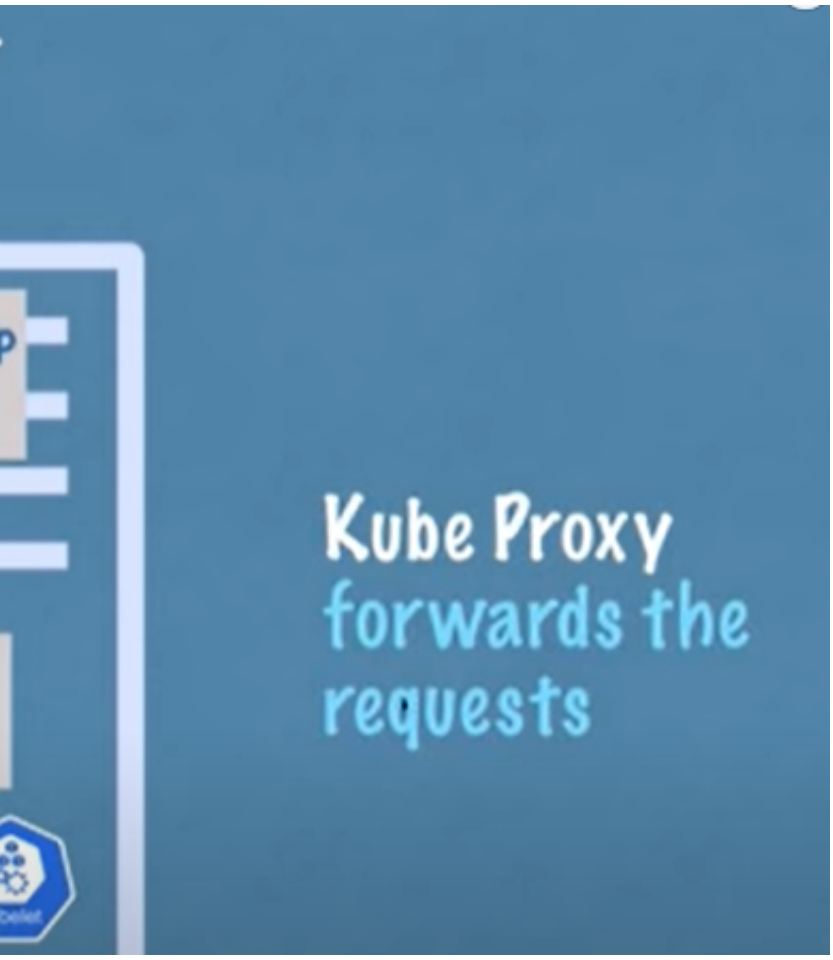

each Node has multiple Pods on it



Kubelet :: interacts with both - the container and Node. It starts the POD with a container inside.
Container Runtime ==>
KubeProxy ==> Communication between Nodes.

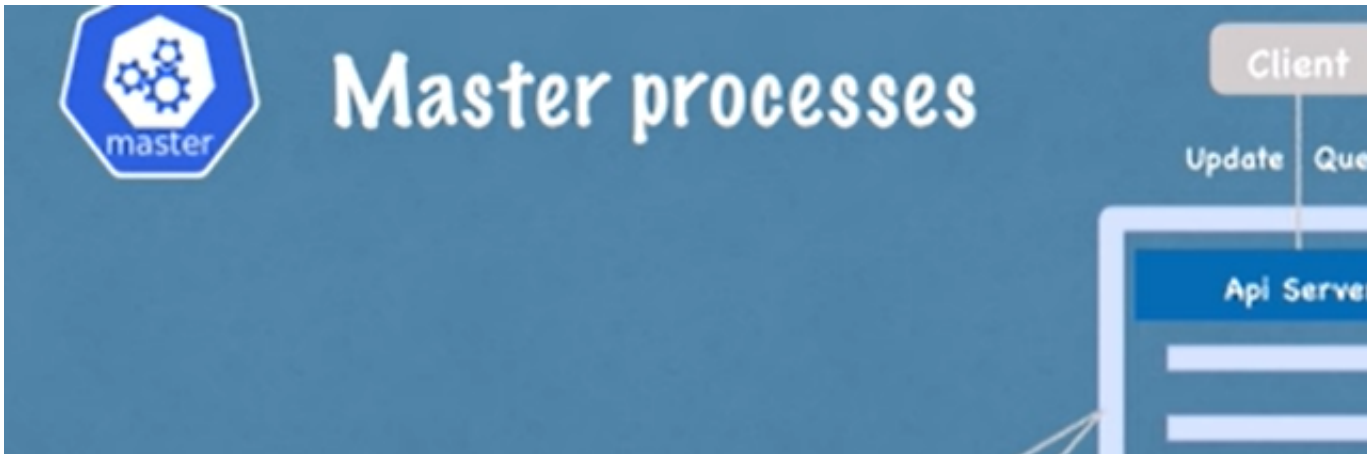


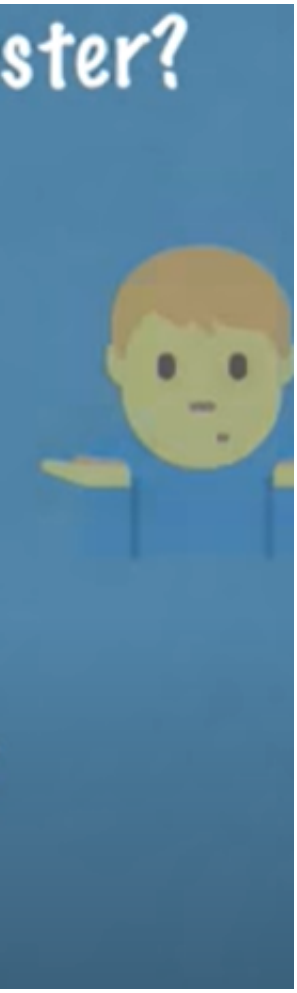

3 processes must be installed on
very Node
Worker Nodes do the actual work


Kube Proxy
forwards the
requests

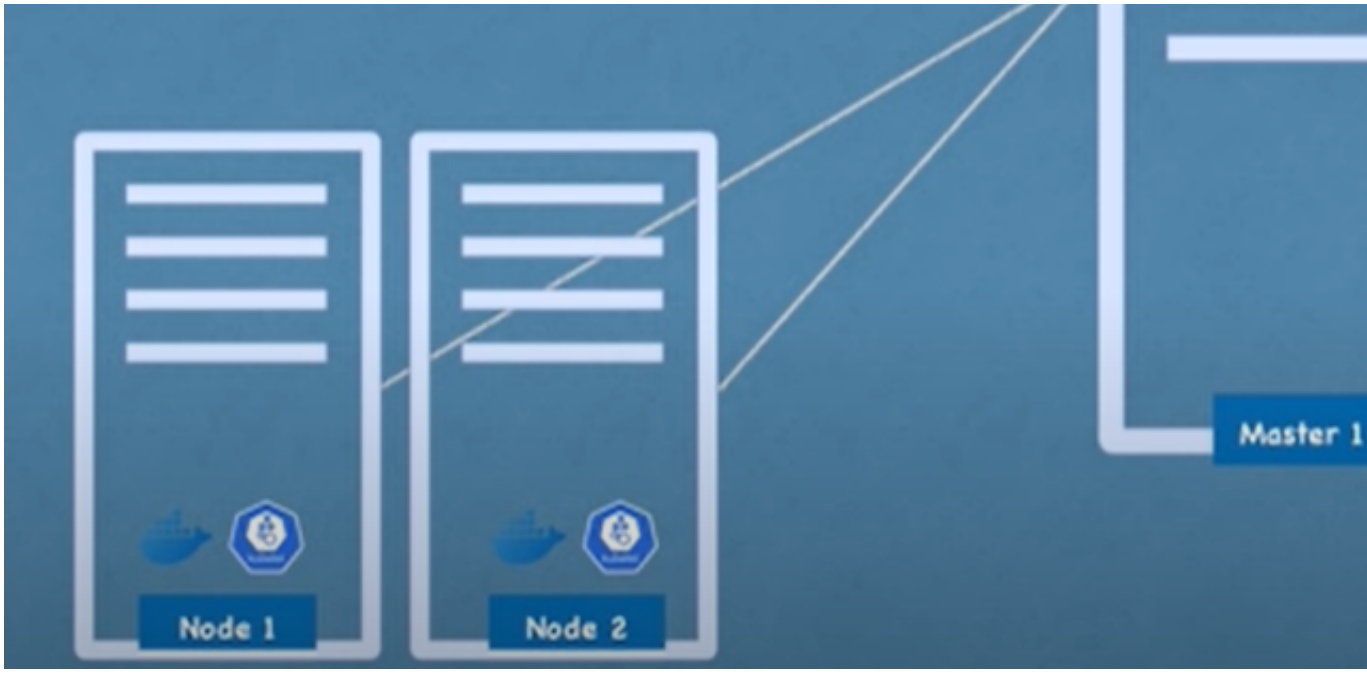
3) Master Node:

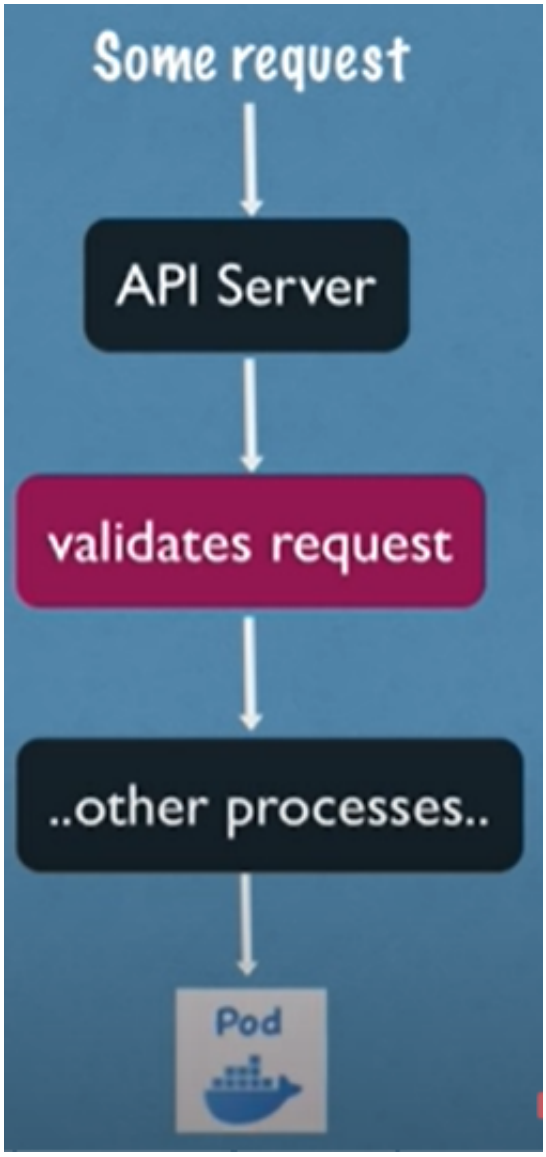
4) API server ::: only 1 entry point in to Cluster.



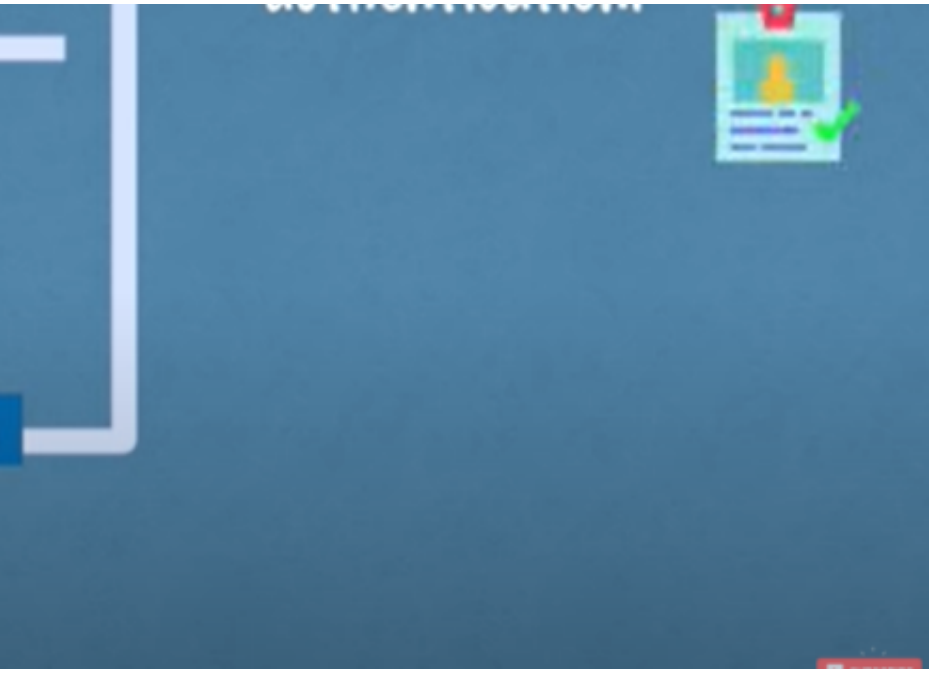

ster?

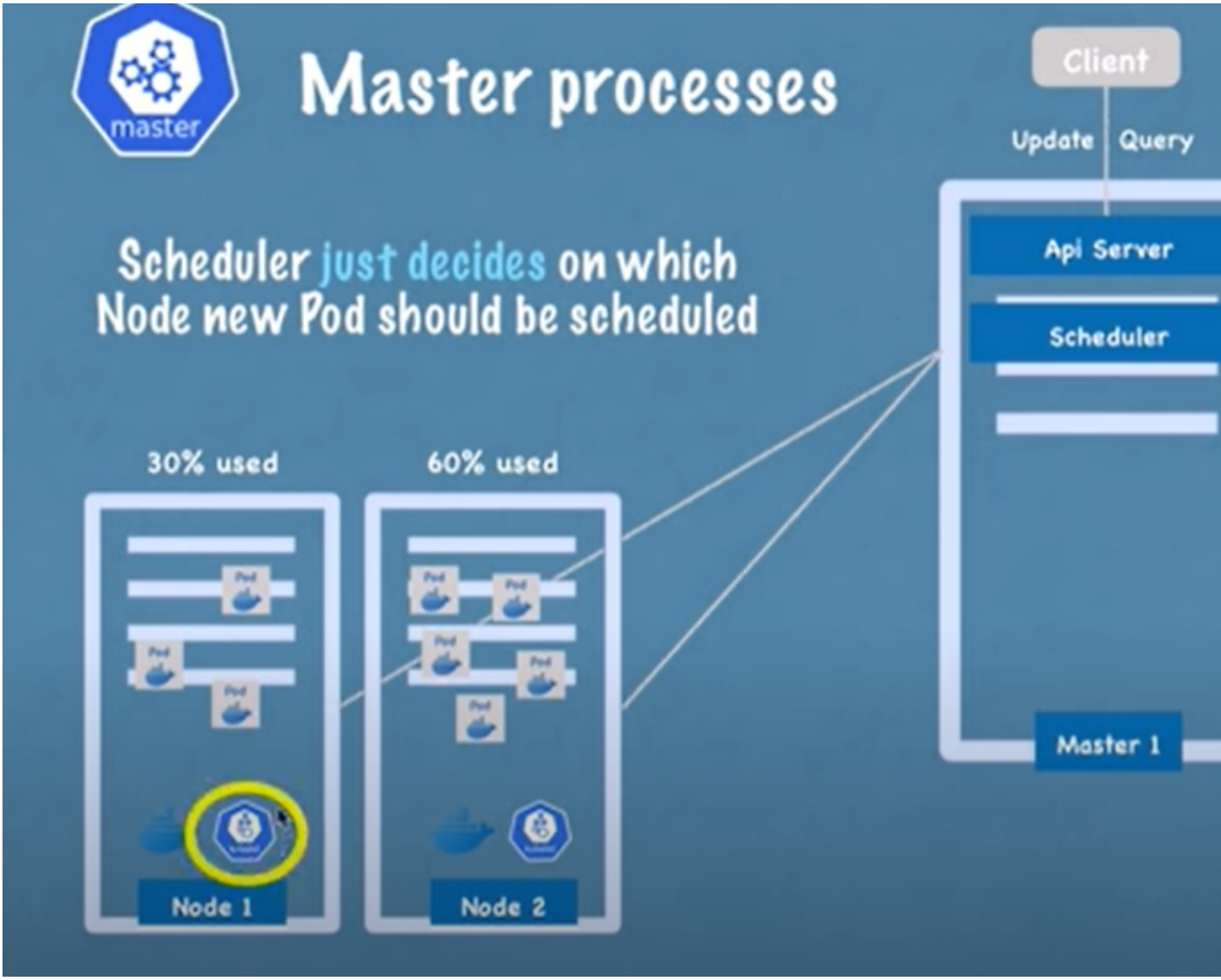

ry
r
cluster gateway
acts as a gatekeeper for
authentication!



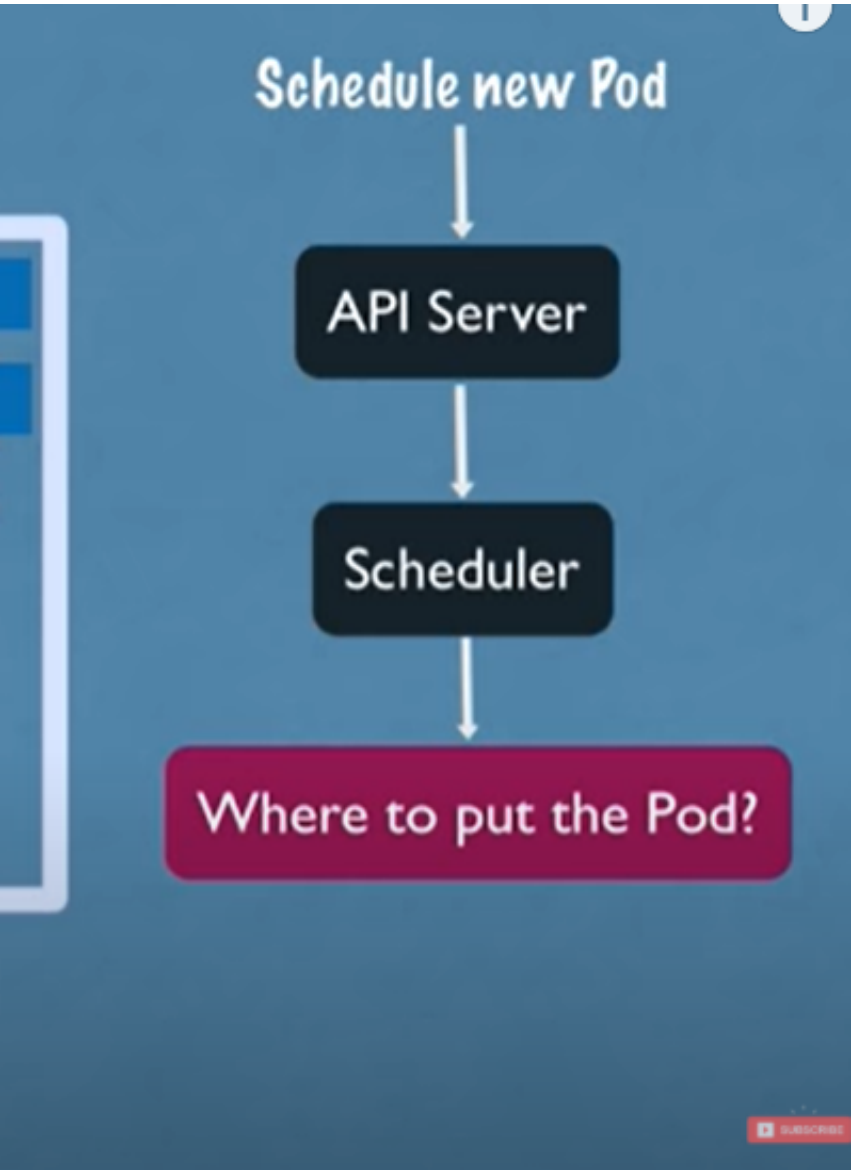


5) Scheduler :: Real processing or starting of POD in to Node is done by Kubelet.

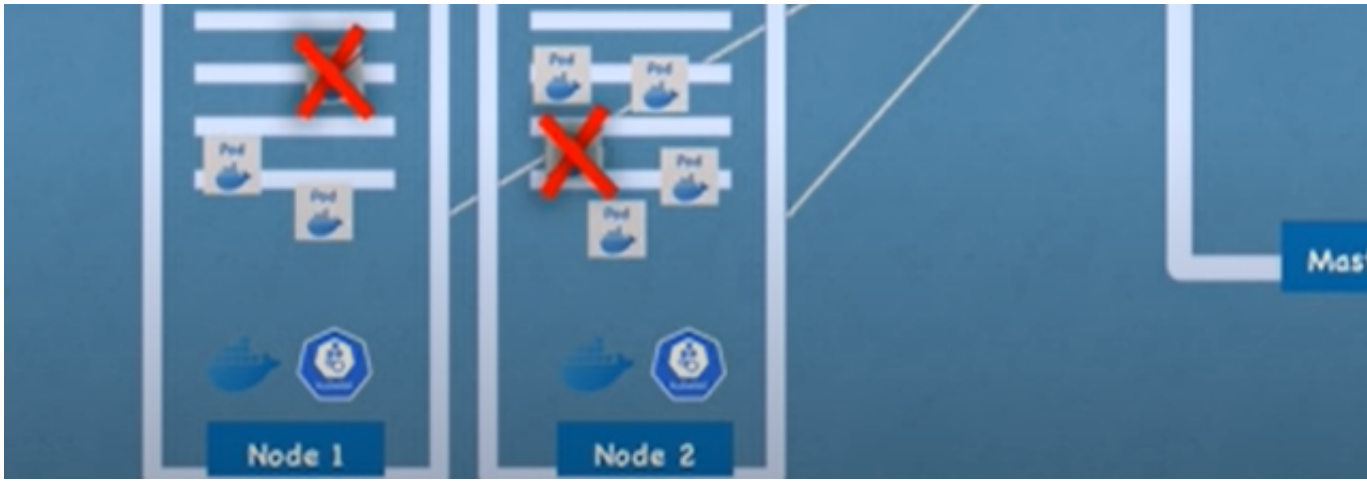




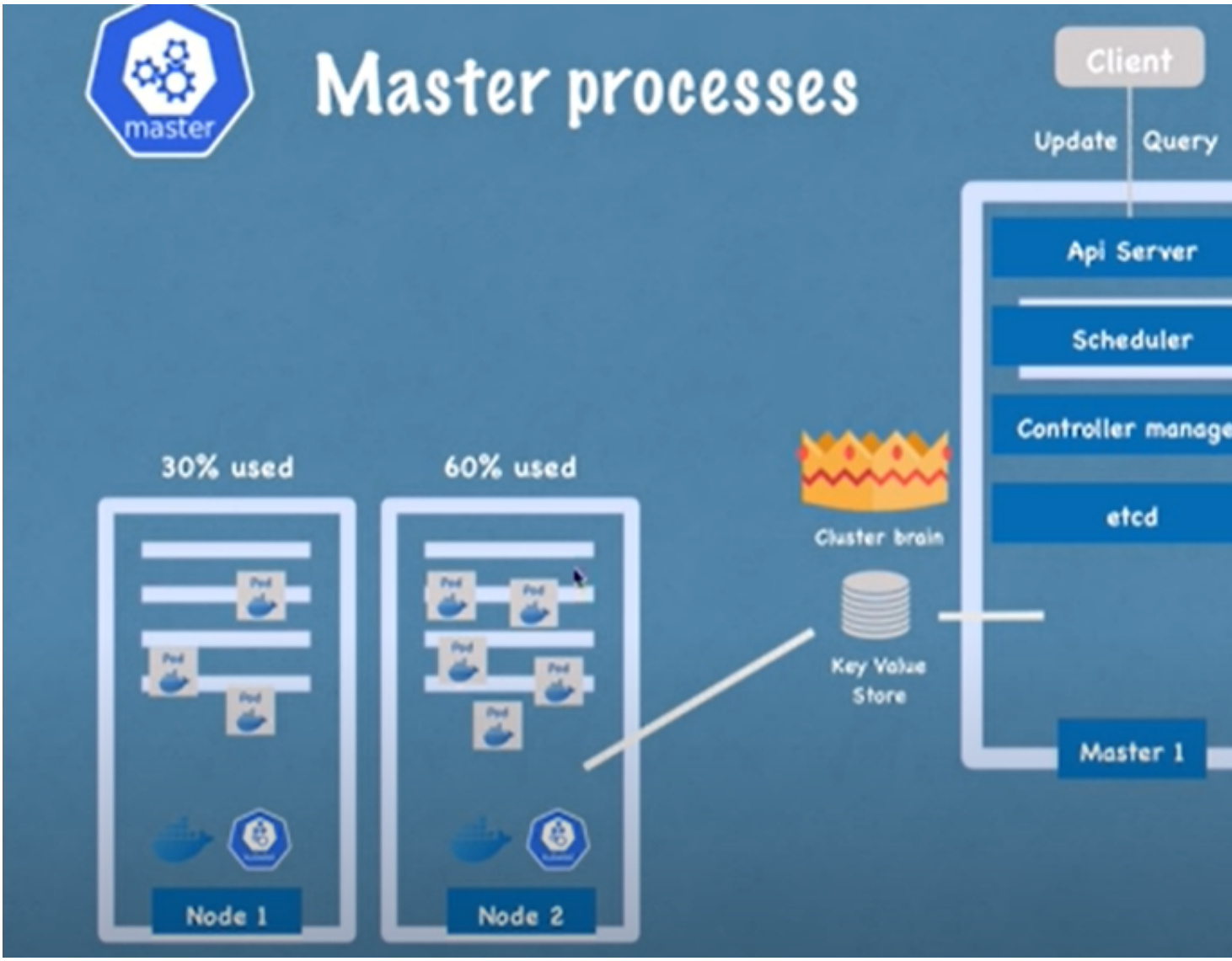
1) Control Manager: Suppose if any POD dies/crashes there should be some way to get an idea for that POD and here comes into picture Controller Manager.


Schedule new Pod
API Server
Scheduler
Where to put the Pod?


Query
manager
detects cluster state changes



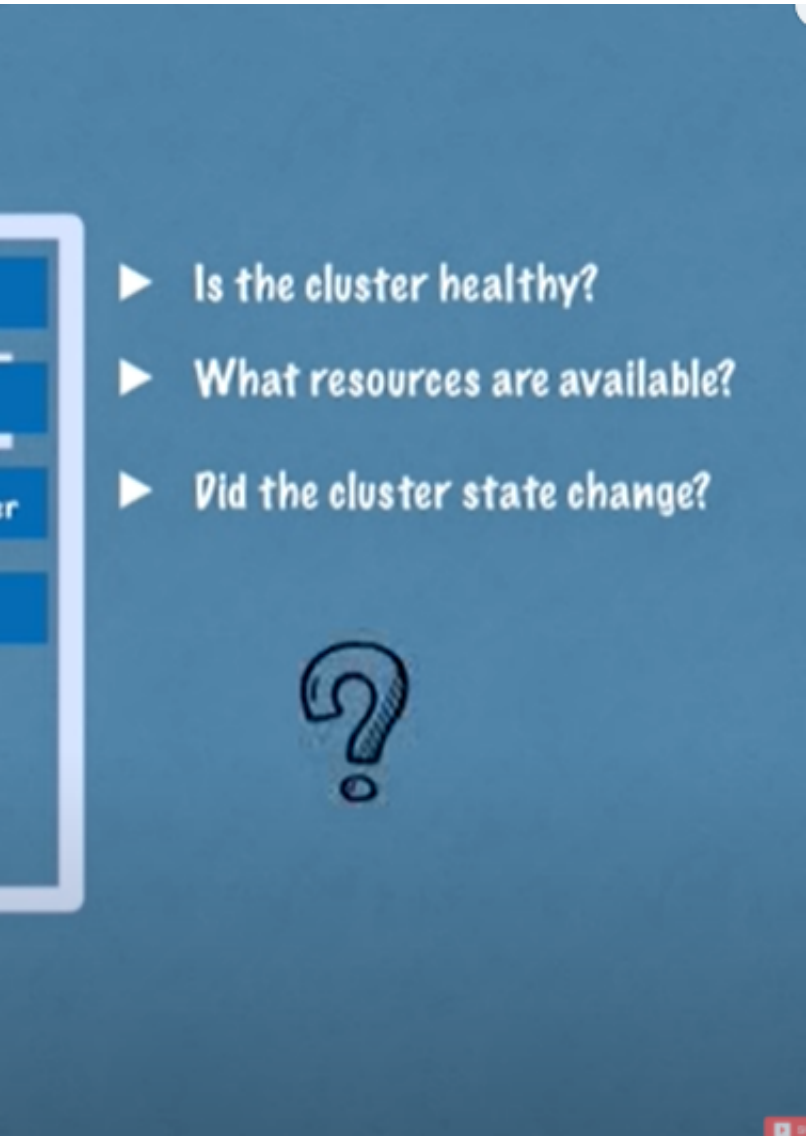
13)etcd: used to store Cluster State Information.

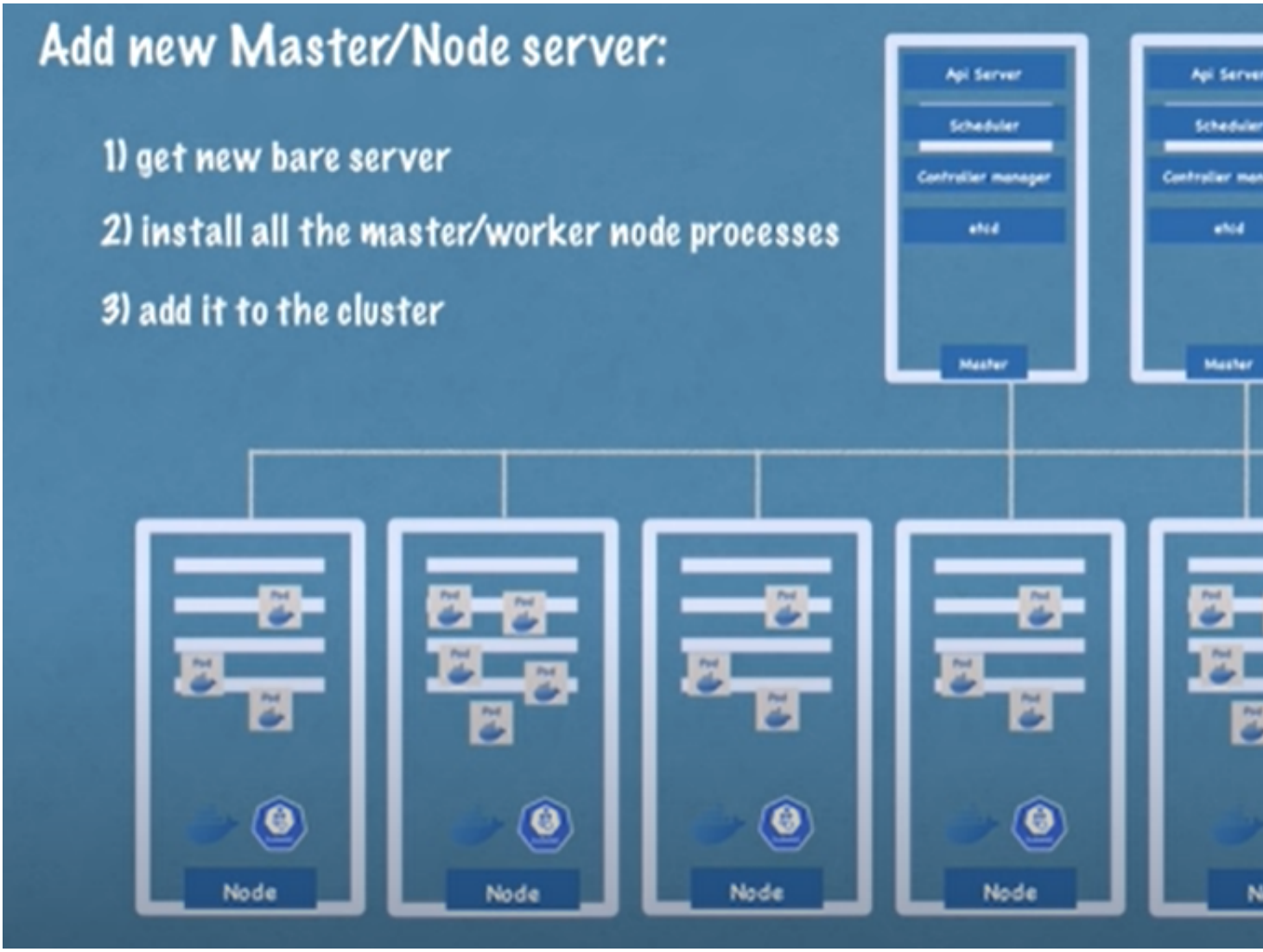


1) So there are multiple Master Nodes present in K8 cluster. API service is load balanced and ETCD is distributed storage across all Master Nodes.

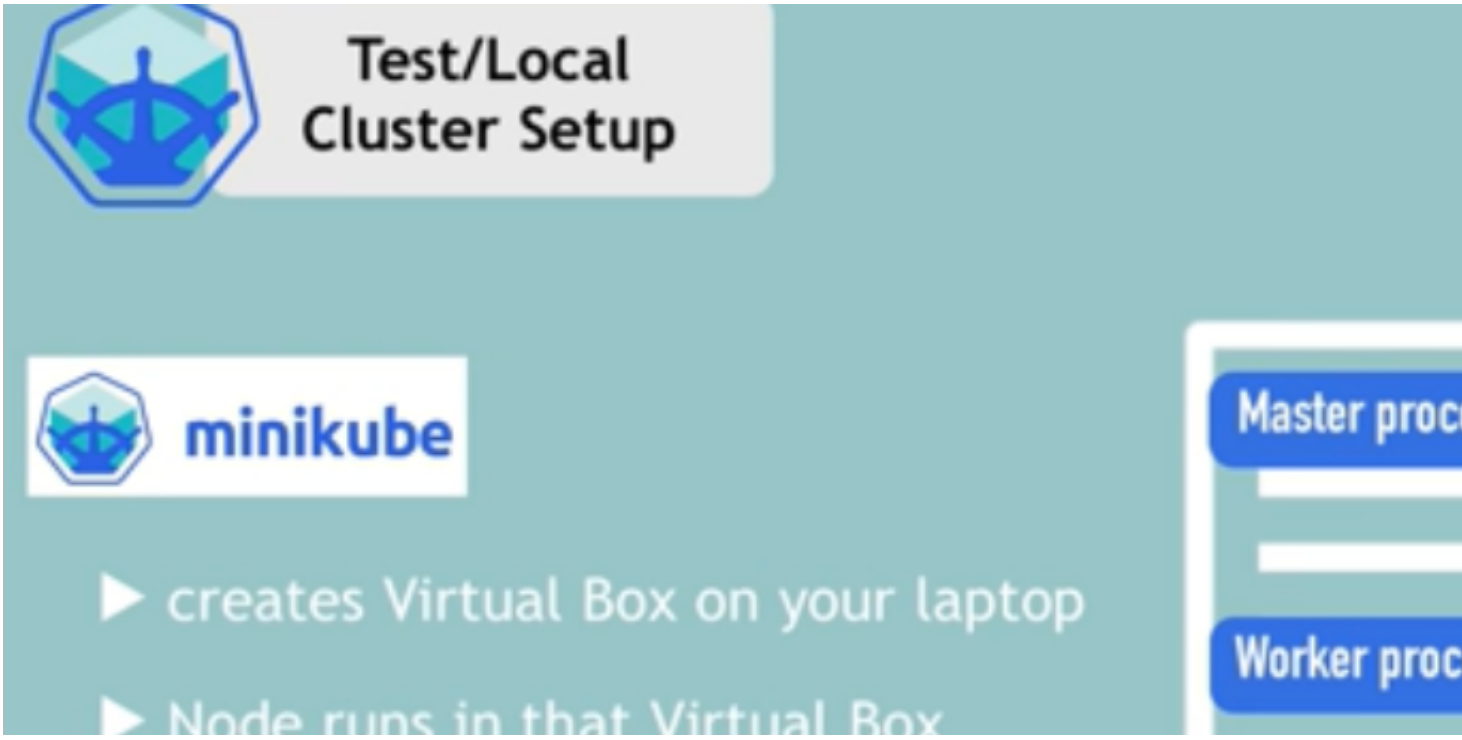
========== Cluster Setup==========


ter 1

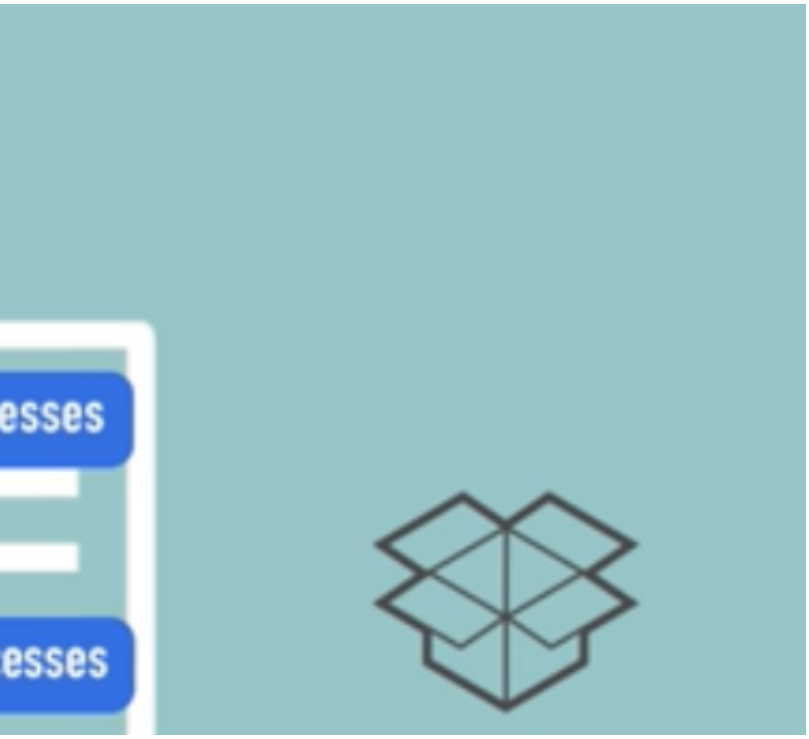

Is the cluster healthy?
What resources are available?
Did the cluster state change?

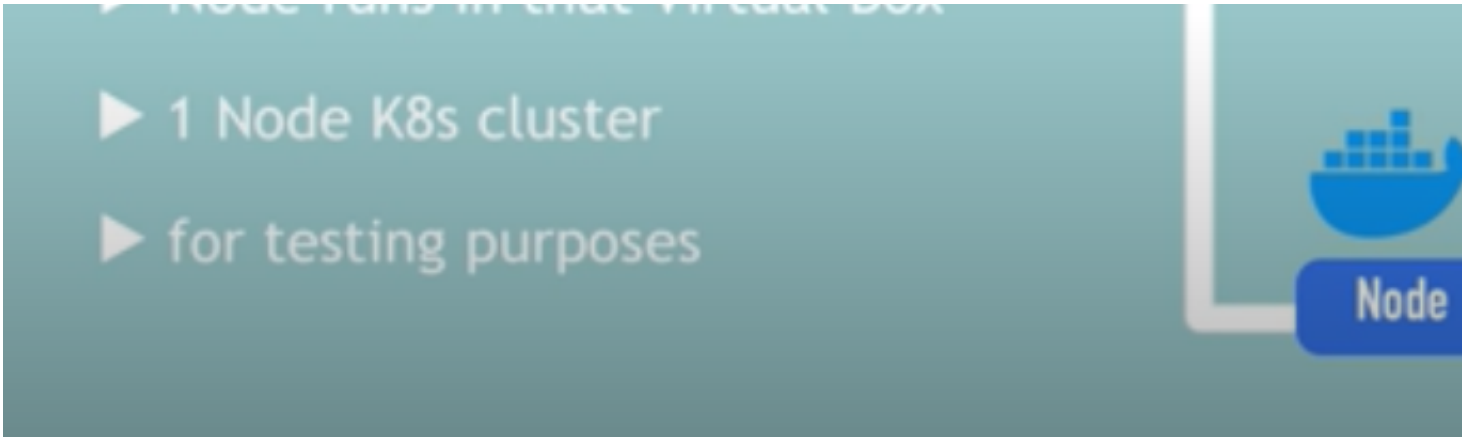

Add new Master/Node server:
1) get new bare server
2) install all the master/worker node processes
3) add it to the cluster
Api Server
Scheduler
Controller manager
etcd
Master
Node
Node
Node
Node

15)

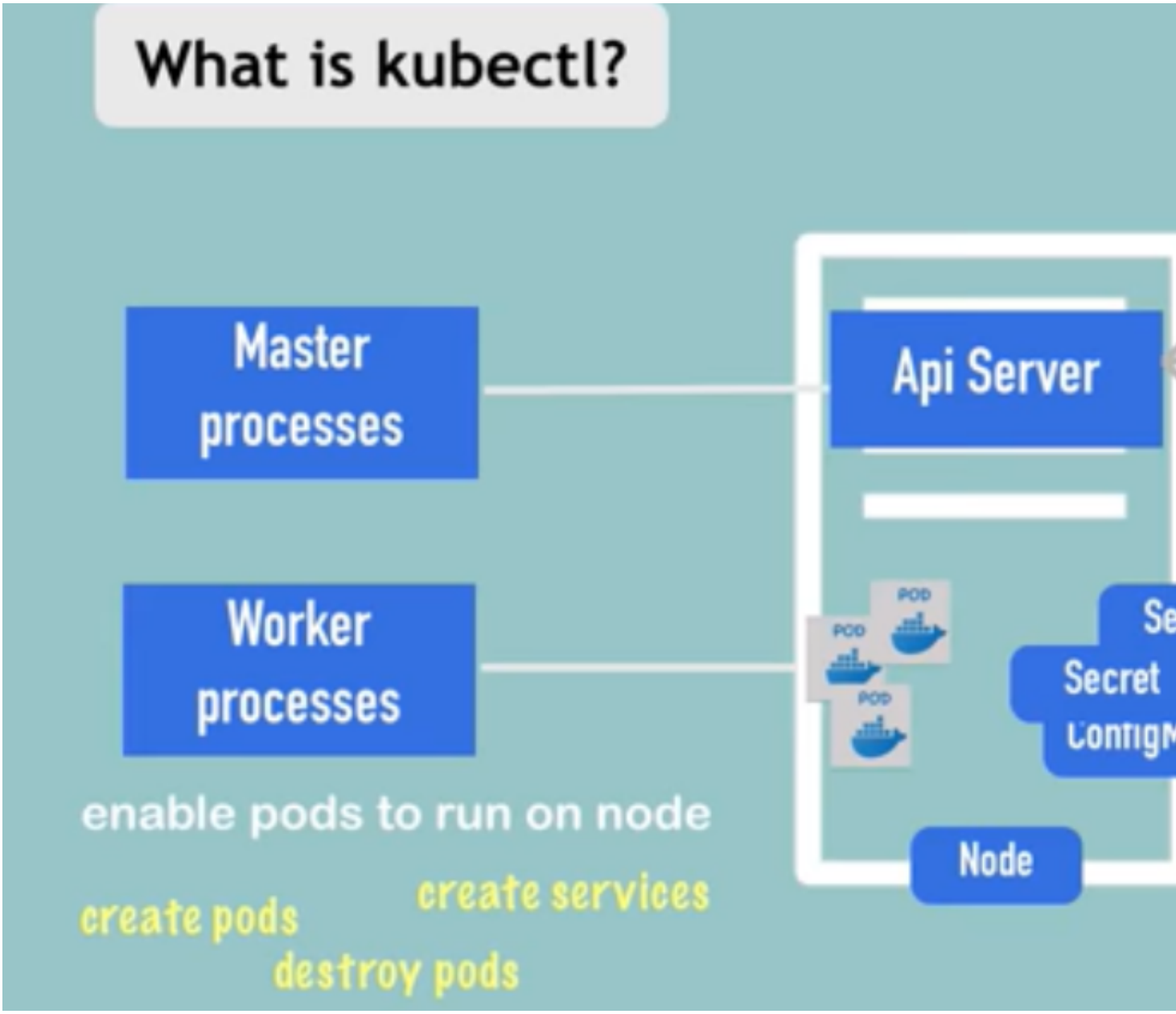

Test/Local
Cluster Setup
minikube
creates Virtual Box on your laptop
Node runs in that Virtual Box
Master proc
Worker proc


Api Server
Scheduler
Controller manager
etcd
Master
Pod
Pod
Pod
Pod
Node
Node

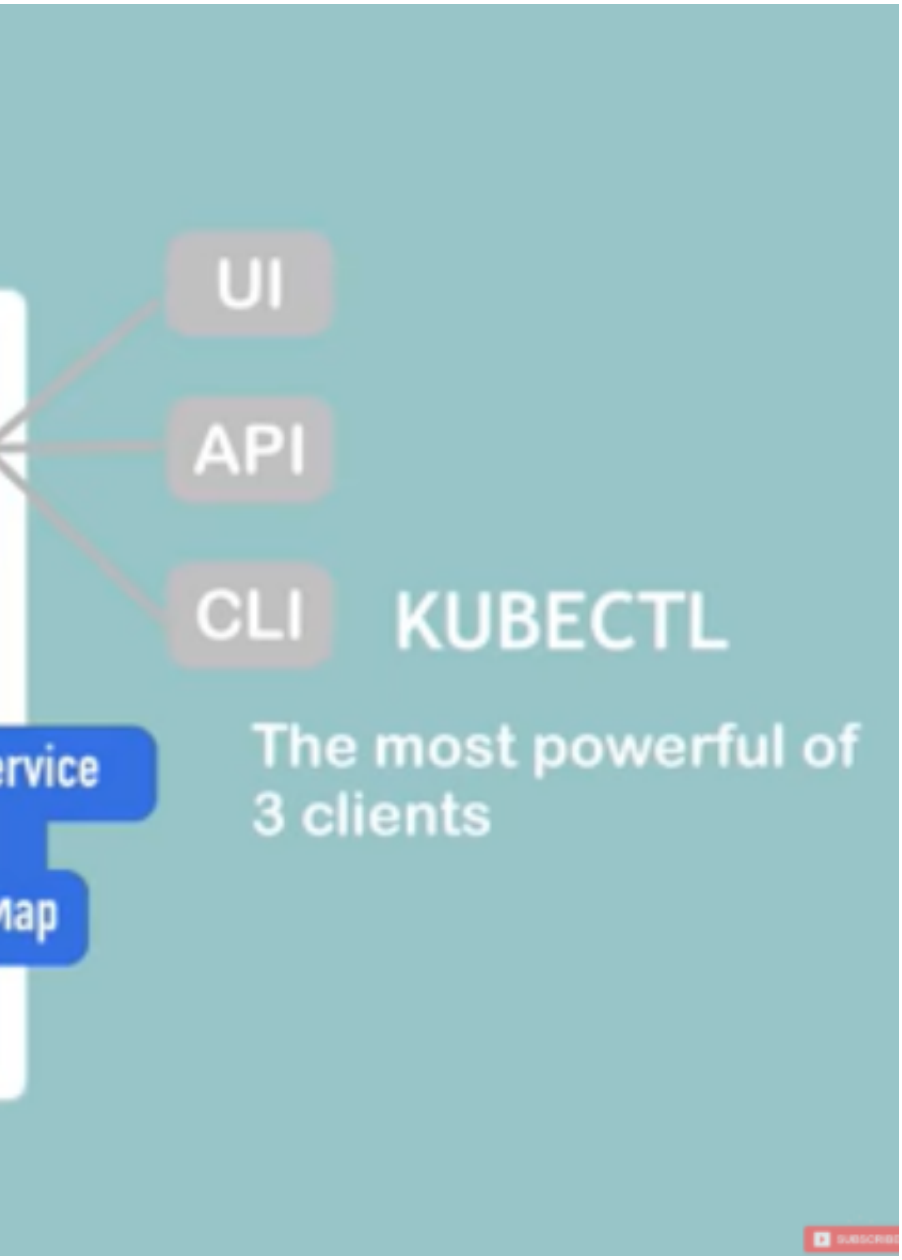

esses
esses


1 Node K8s cluster
for testing purposes
Node

16)


What is kubectl?
Master processes
Worker processes
Api Server
POD
POD
POD
Secret
Node
enable pods to run on node
create pods
create services
destroy pods


Virtual Box


UI
API
CLI
KUBECTL
The most powerful of
3 clients

Kubectl commands:

Kubectl get nodes
Kubectl version ==> both client and server version should be present.
Kubectl get pod
Kubectl get services
Kubectl get -h ==> help command

WHEN WE WANT TO CREATE POD , DIRECTLY THERE IS NO COMMAND TO CREATE POD RATHER WE CREATE DEPLOYEMNT (I.E. Wrapper for POD) ==> it is Blueprint of POD.
-Kubectl create deployment nginx --image =nginx ==> will get nginx image from docker hub.
-Kubectl get deployment
-Kubectl get pod
-Kubectl get replicaset ==> managing replica of POD.
-Kubectl edit deployment nginx ==> we will get autogenerated config file with default values. Here we can edit it and save it(:wq) ==> automatically underlying POD is getting generated/update wrt new config files.==> get it validated using => kubectl get pod == old POD gone and new POD created.

==== Debugging PODS=====
Kubectl logs podName ==> basic log details.
Hubectl describe pod podName ==> more details descriptive one

Kubectl exec -it podName -- bin/bash ==> interactive Terminal ==> now this cmd will lead us to terminat of underlying container/image..run allcommands like ls and all.

Kubectl delete deployment nginx

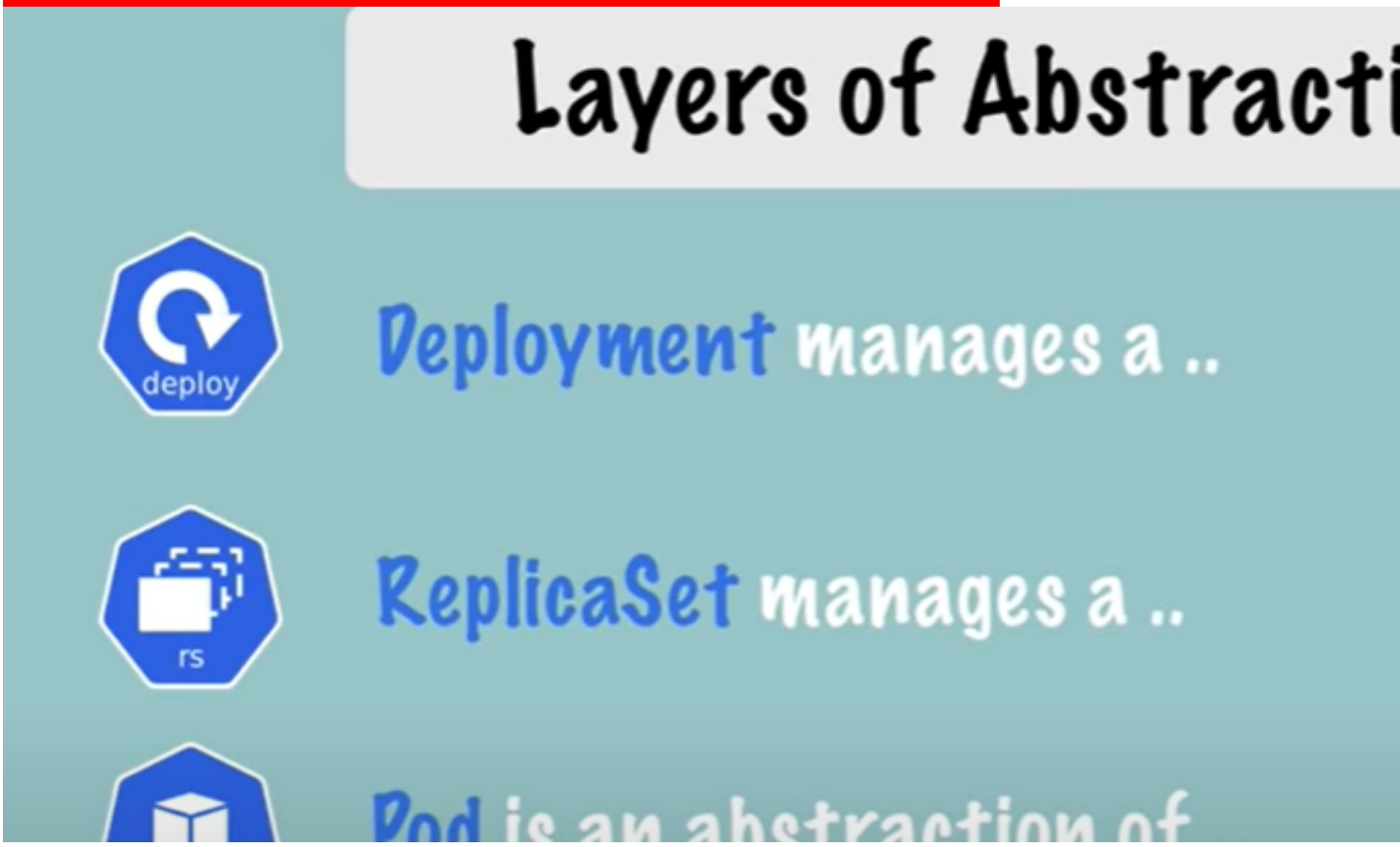
Use config file as part of Config File Name:(Config file is saved as .yaml file)

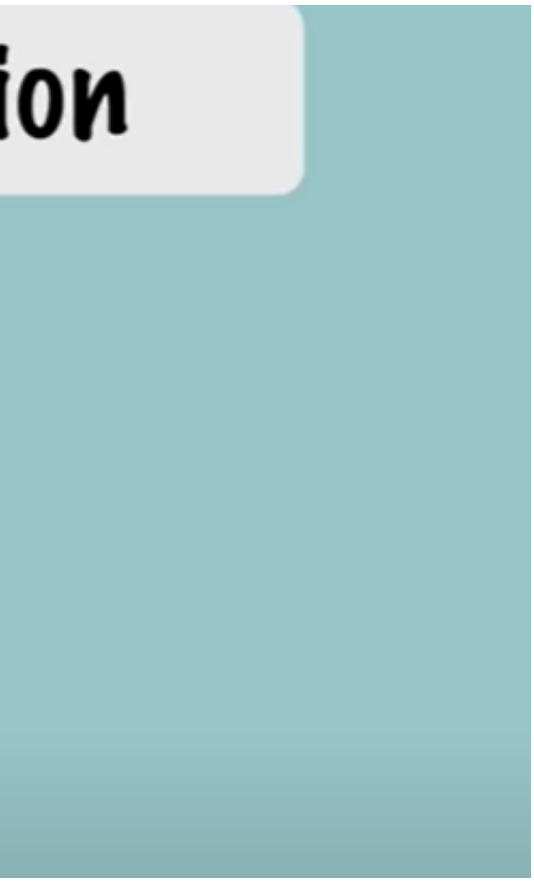
Kubectl apply -f "config file Name" ===> to avoid entering so many values from command prompt..===> Both Create and Deploy.
Name==> name of deployment.
Spec :: ==> 1st One wrt Deployment and 2nd Spec wrt POD.(Specific container).

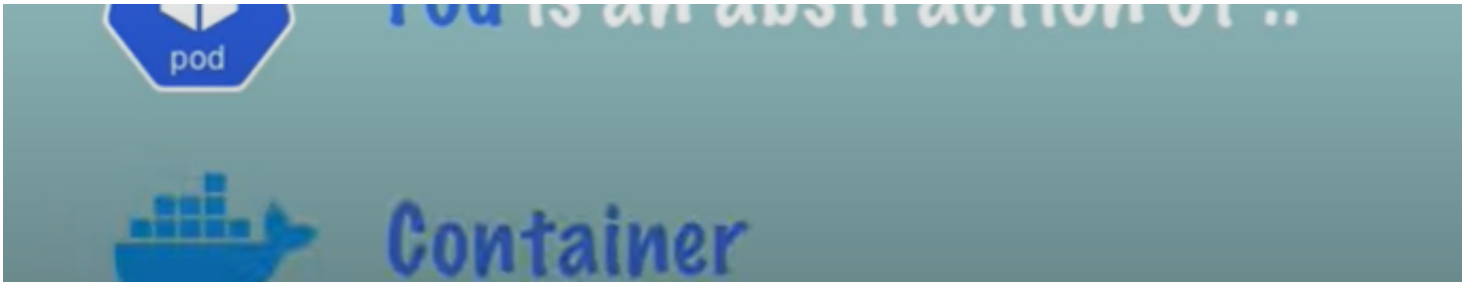
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
```

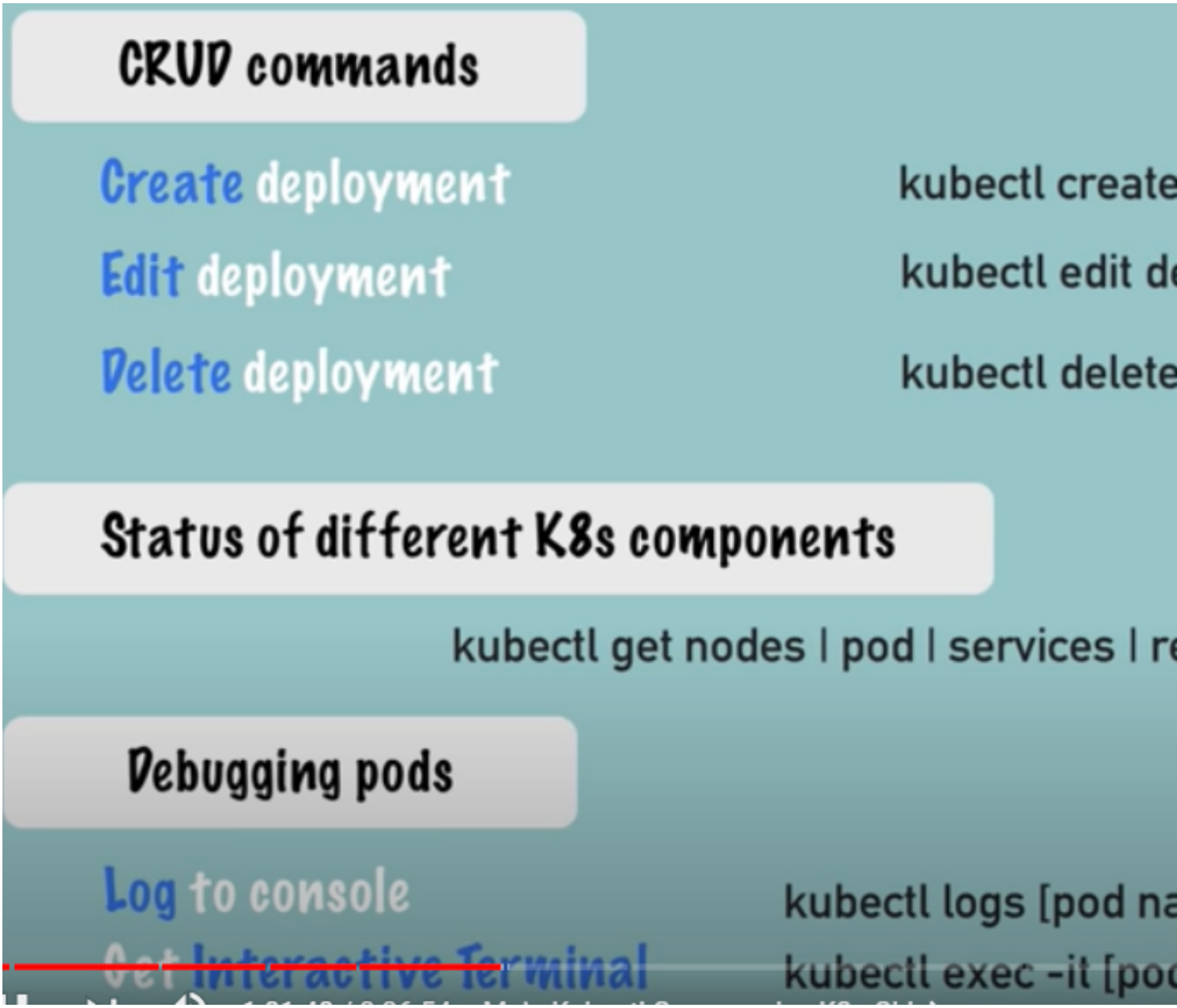
```
  labels:
    app: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.16
        ports:
        - containerPort: 80
```

EVERYTHING BELOW DEPLOYMENT IS HANDLED BY KUBERNETES.

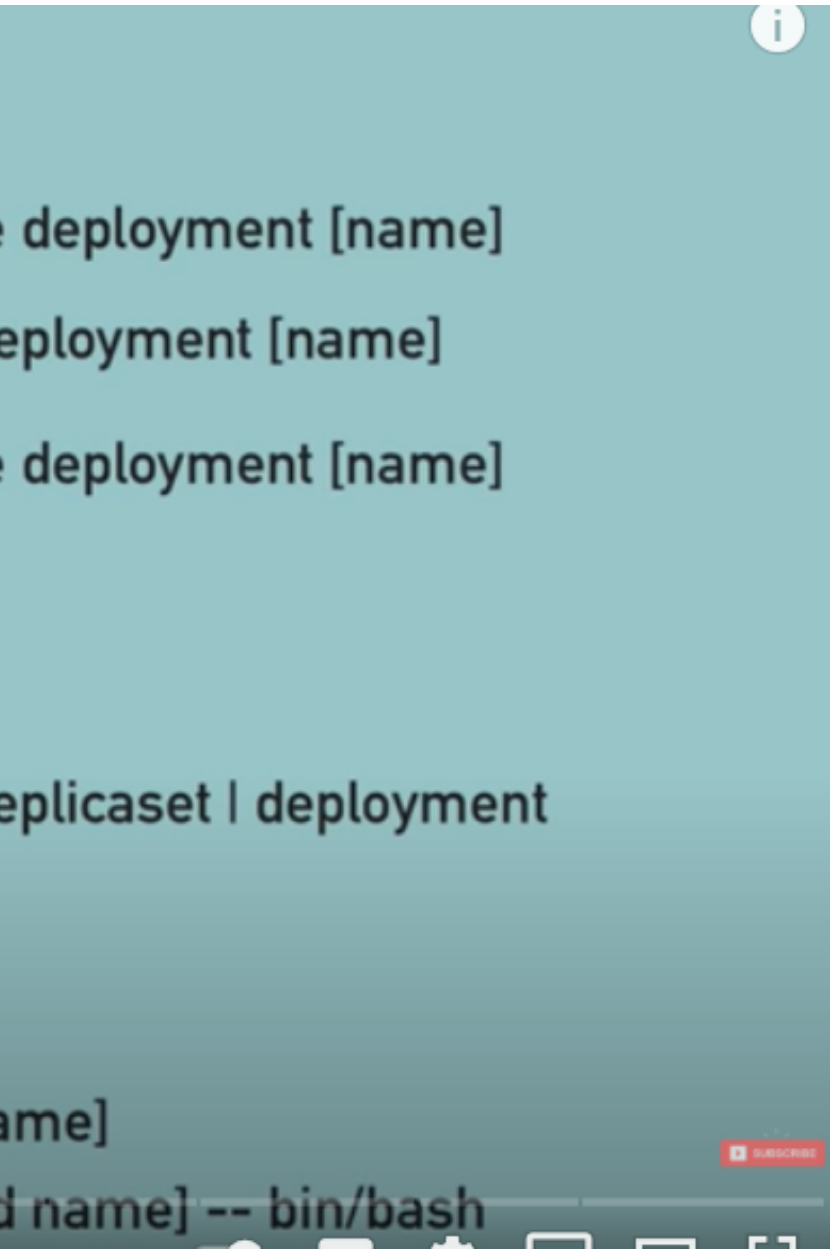


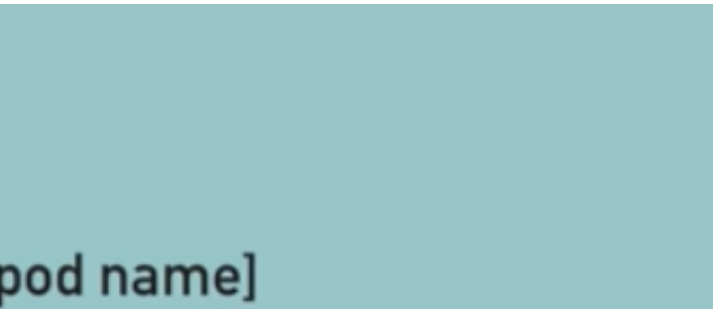

ion

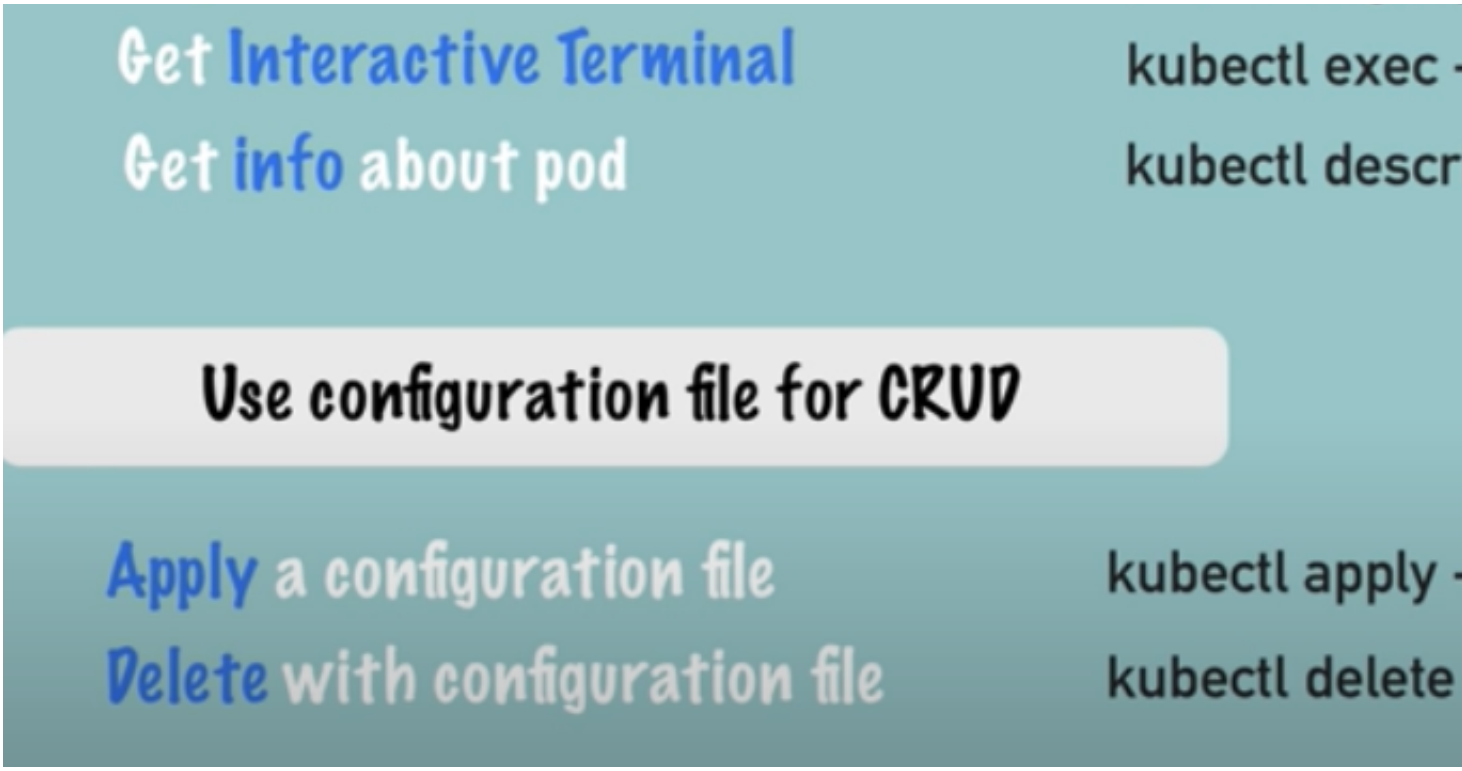

pod
Container


CRUD commands
Create deployment
kubectl create
Edit deployment
kubectl edit d
Delete deployment
kubectl delete
Status of different K8s components
kubectl get nodes | pod | services | r
Debugging pods
Log to console
kubectl logs [pod na
Get Interactive Terminal
kubectl exec -it [pod


Debugging pods
Log to console
kubectl logs [


deployment [name]
eployment [name]
deployment [name]
eplicaset | deployment
ame]
name] -- bin/bash


pod name]


Get Interactive Terminal
kubectl exec
Get info about pod
kubectl descr
Use configuration file for CRUD
Apply a configuration file
kubectl apply
Delete with configuration file
kubectl delete

-it [pod name] -- bin/bash

ibe pod [pod name]

-f [file name]

-f [file name]