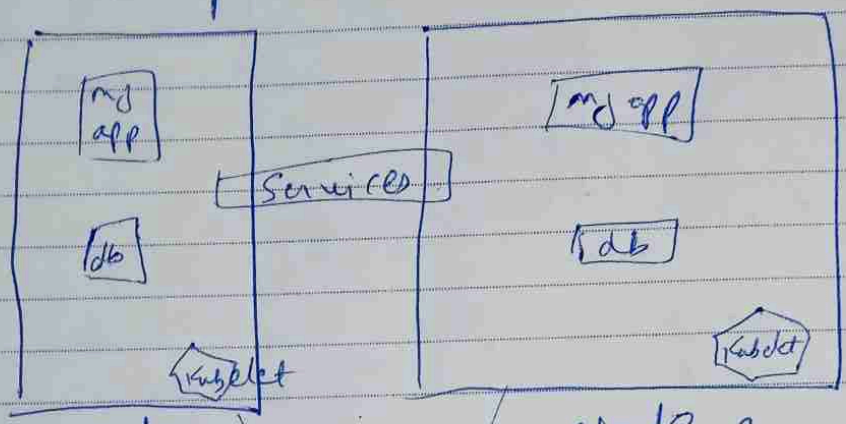
1) First process - container runtime on node

2) Kubelet interacts with both - the container and node

- Kubelet scheduled the pods and container
- Kubelet start the pod with containers inside

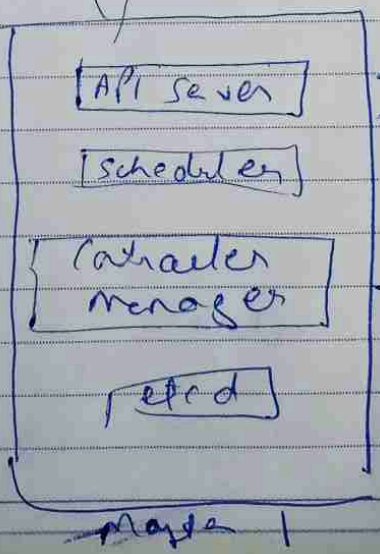


3) Kubeproxy forward the requests.

* MASTER NODE MANAGES the CLUSTER

It has 4 processes

1) Some request → API Server → validate request → other process → Pod



- cluster gateway
- acts as a gatekeeper for authorization
- detects cluster state changes

a) Schedule new Pod → API Server → Scheduler → Pod (where to put the Pod)

* Scheduler just decide on which node new pod should be scheduled.

3) Controller manager → Scheduler → kubelet → new pod

4) Etcd → Key value store of a cluster state
etcd is the cluster brain.
Key value ka the cluster changes

etcd checks,
is cluster healthy, is resource available, did the cluster state change?

Note: App data nahi store in etcd

6) What is MiniKube?
Minikube is open source cluster with one node where master and worker process both run on one node. This node have docker container runtime pre installed.

- Minikube will create a virtual box or hyperviser and node run in that virtual box
- Minikube is a 1 node K8s cluster for testing

Kubectl: CLI for K8s cluster (UI and API)
command line tool

API Server is main entrypoint for K8s cluster

Minikube needs Virtualization

* Basic Kubectl (commands)
Create, edit, delete

LIC
भारतीय जीवन बीमा निगम
LIFE INSURANCE CORPORATION OF INDIA *

Kubectl get nodes
" " pod
" " services

:- Kubectl create deployment NAME --image=image
[--dry-run] [options]

:- Kubectl create deployment nginx-dep --image=nginx

:- kubectl get deployment

:- kubectl get pod

~~kubectl get pod~~ # Layers of Abstraction #

Deployment manages a
Replicaset manages a
pod is an abstraction of
container

} K8s managed this

kubectl edit deployment nginx-depl

* Debugging pod

- kubectl logs <name>
- kubectl exec -it (interactive terminal)

- Kubectl apply -f config-file.yaml
kubectl delete -f "
- Kubectl describe pod [Pod name]

* K8s YAML CONFIGURATION

1) metadata
2) Specification
3) status

metadata contains Labels
specification contains selector

In specification of service we define a selector which makes connection b/w service and deployment

Secret configuration file

```
apiVersion : v1
kind : secret
metadata :
    name : mongodb-secret
type : opaque
data :
    username :
    pwd :
```

(actual content)

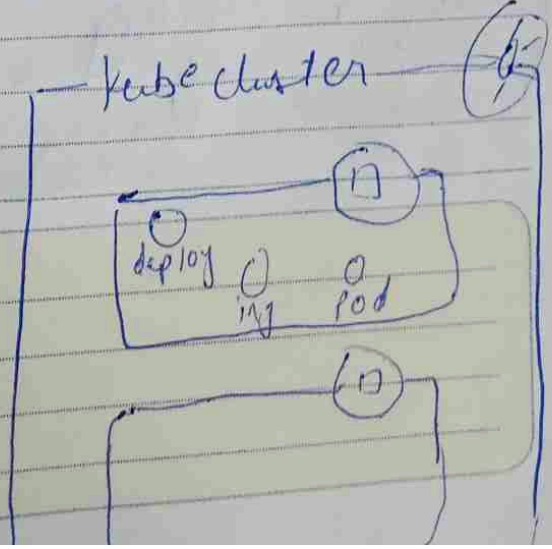
Service configuration file

```
- kind : "service"
- metadata/name : a random name
- selector : to connect to pod through label
- ports :
    port : service port
    targetPort : container port of deployment
```

nodeport : 30000 - 32767
port for external IP address
port you need to put into browser

Q What is Namespace

- Organise resources in namespaces
- Virtual cluster inside a cluster



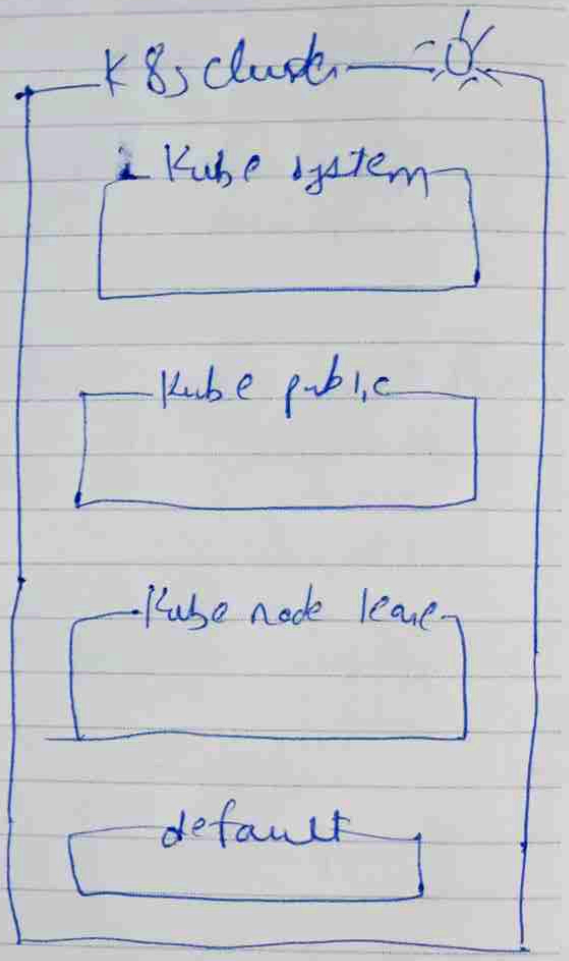
4 namespaces per default

1) Kube system
- do not create or modify
- System processes
- master and kubectl processes

2) Kube public
- publicly accessible data
- A configmap, which contain cluster information

3) Kube - node - lease
- heartbeat of nodes
- each node has associate lease object in namespace
- determines the availability of node

4) default
resources we created are located here



To create,
- kubectl create namespace my-namespace
- also through config file

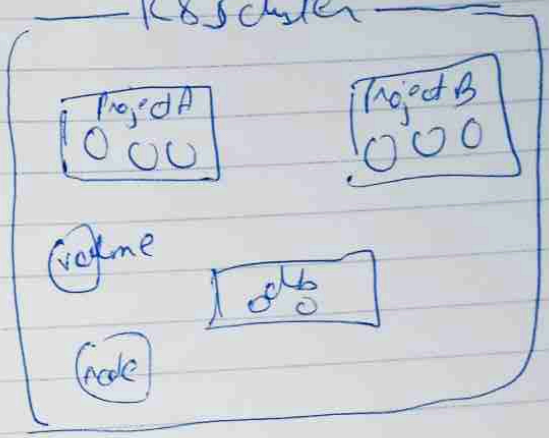
Use of namespace
1) easy to classify
2) avoid conflict btw teams
3) Resource sharing / blue green deployment
4) Access and resource limit

Characteristics

- Each namespace must define own config map and secret
- We can share service

* Components, which can't be created within a namespace

volume || Node



- Kubectl / api-resources --namespaced = false

 # Install kubens for changing active namespace #

→ kubectx, then kubens aage se