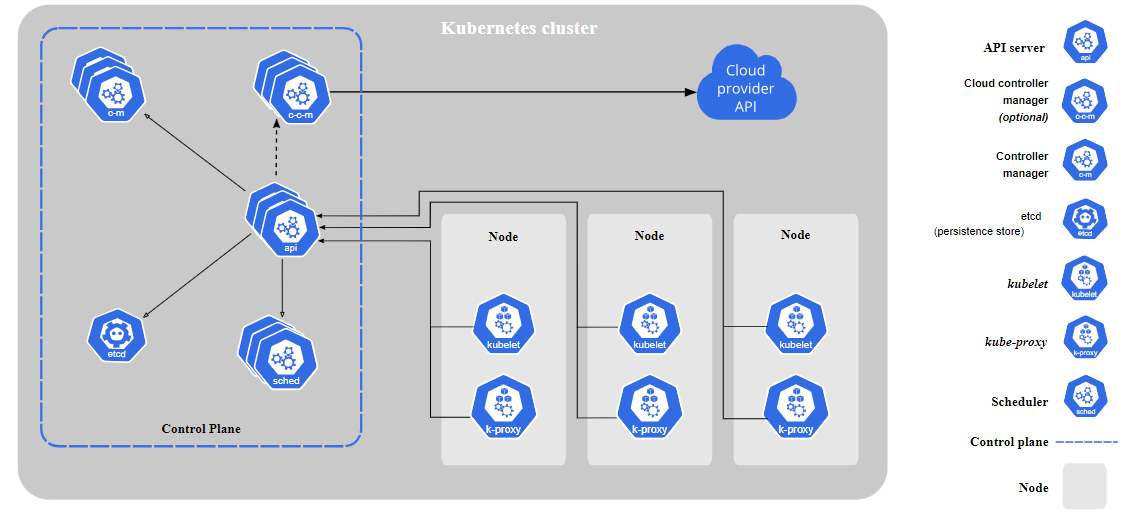
Arch & comps of k8s:

K8s is depends on any one of the containerization platforms like docker.

K8s is like Master slave arch setup. We have Master nodes and slave nodes to handle microservices.



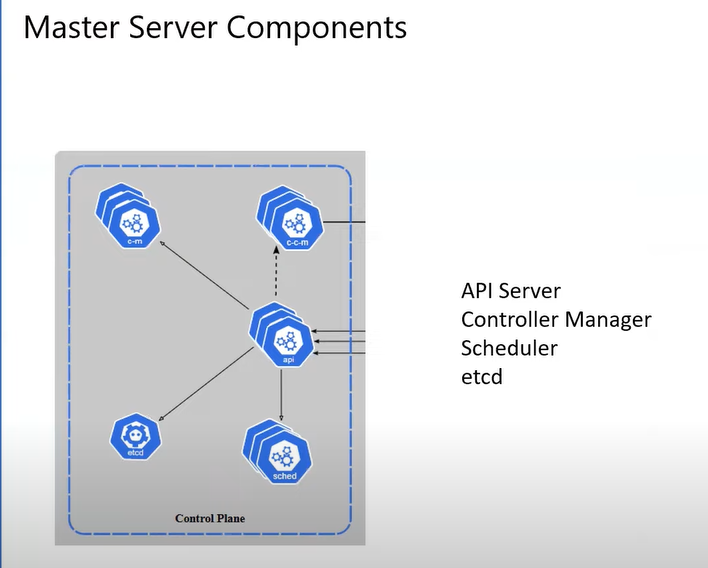
What is cluster?

Collection of multiple systems is called cluster.

K8s is a cluster-based arch in which we have **Master** node is the brain of the infrastructure of k8s, and **worker** node is the actual working systems to handle microservices in k8s.

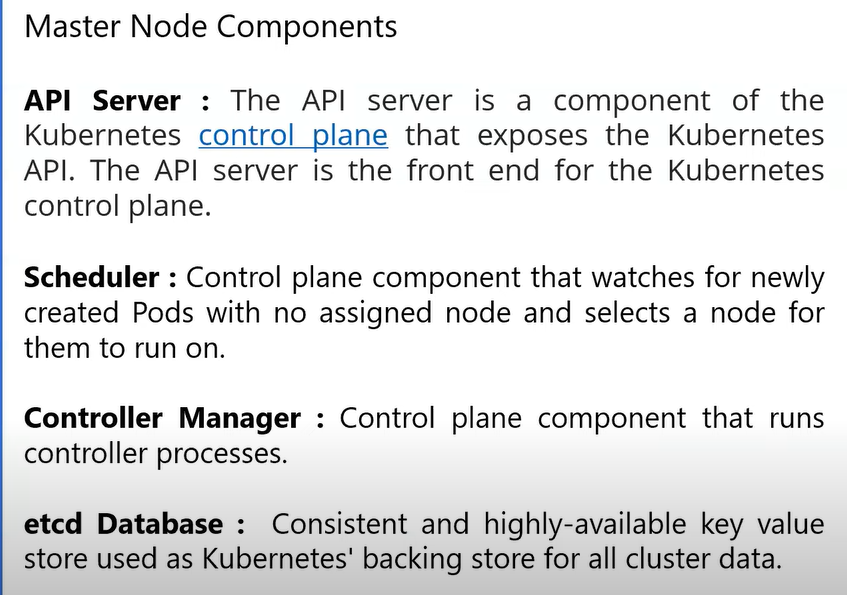
**Master Node/Control plane**:

It may be VM or physical server or any cloud ec2 instance or VM and it would be any flavours of Linux OS machine and if we install **Master Node** components then it becomes Master Node**.**

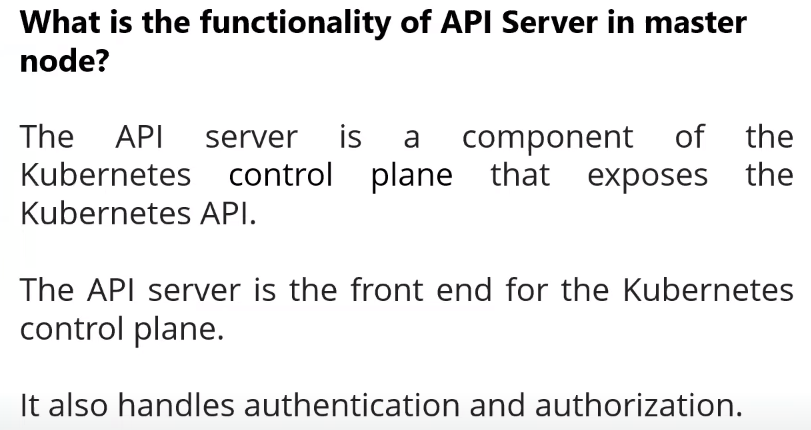


In cloud provider k8s services like AKS, EKS, GKS the responsibility of master node is with the cloud providers only i.e aws ,azure,gcp.

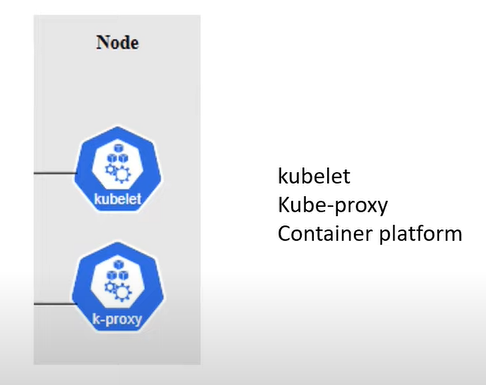
If we set up our own master node using separate VM or servers, we need to maintain and responsible for master node.



Authentication & Authorization is handled by API server.



**Worker Node Components**:



worker node responsibility is to handle workloads i.e pods & deployments.

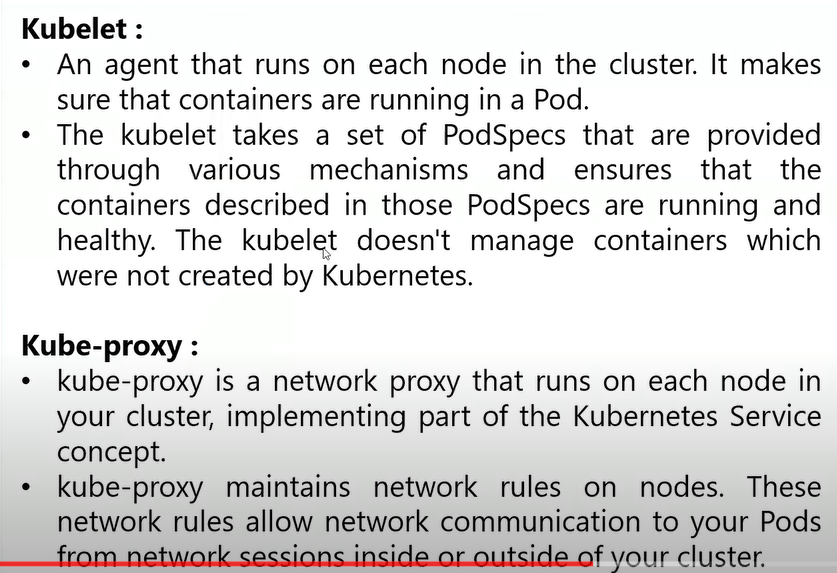
Kubelet is continuously running agent kind of process running on each worker node. It will post /upload details related to node to master node through API Server.

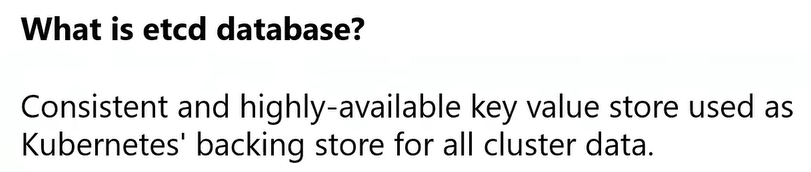
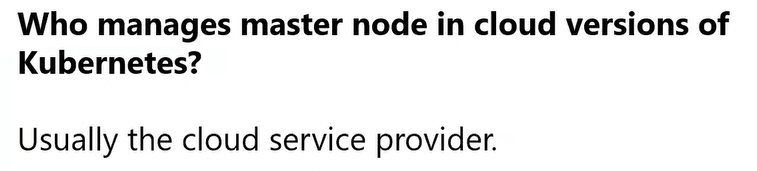
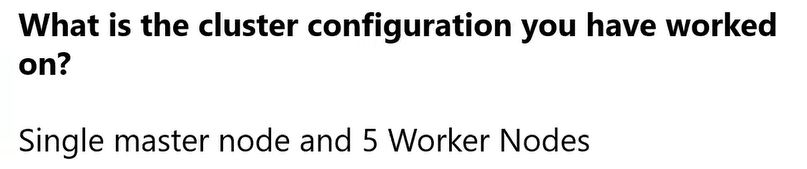
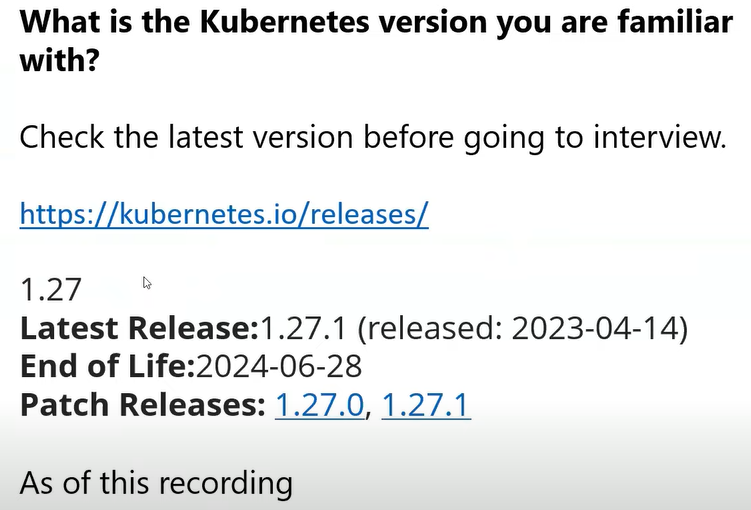
Kube-proxy is continuously running background process on each worker node.when multiple containers are deployed across multiple nodes in a k8s cluster,inorder to select the best path for communication between the containers kube-proxy will helps inorder to reduce the latency.

Kube-proxy is also known as communication optimizer only for containers.

Container platform like docker/linux containers/ containerD also continuously running process on each worker node.

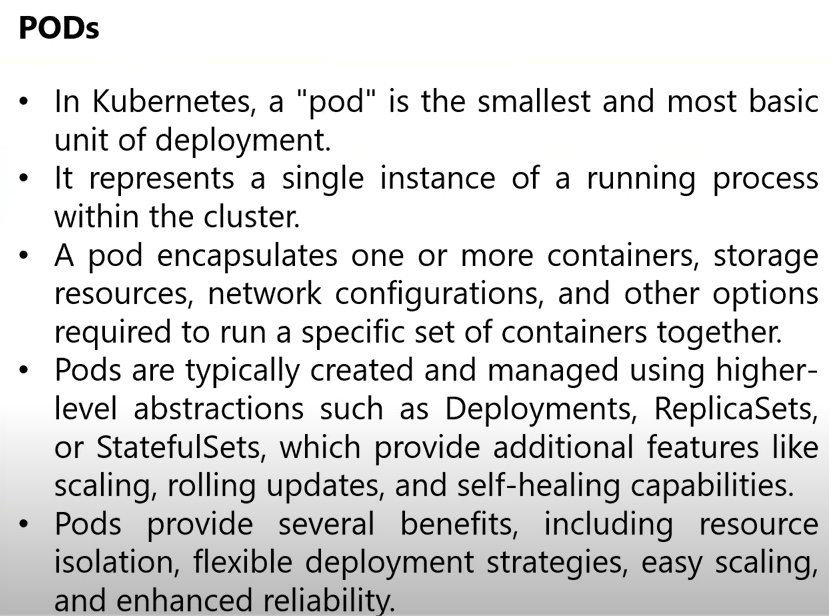
We must install any Container platform like docker/linux containers/ containerD in the worker node so that it will create containers and perform all the operations related to containers.



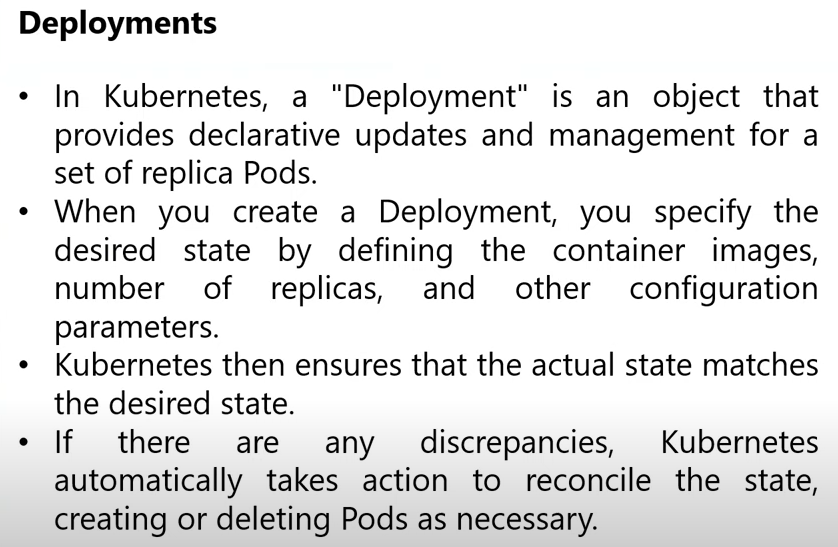
   

**Workload Components** :

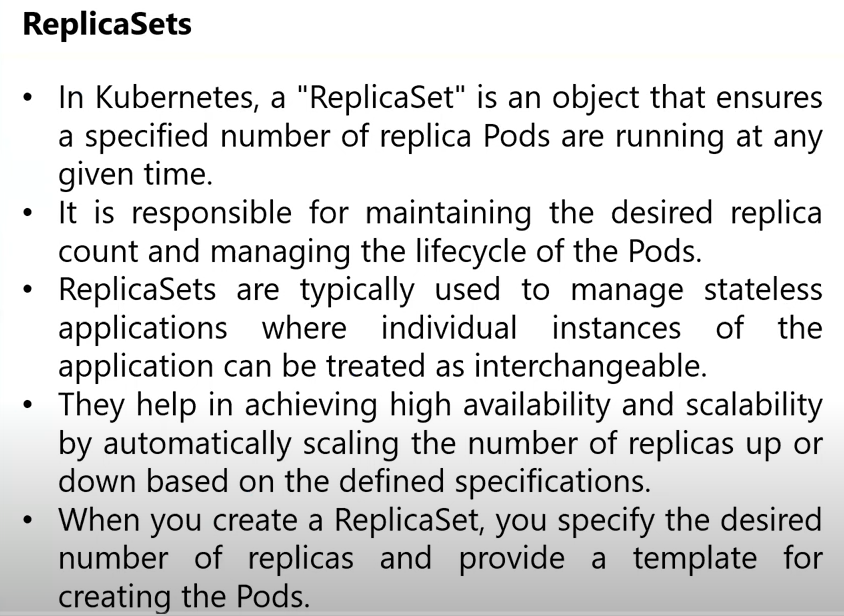
1. **Pods**:

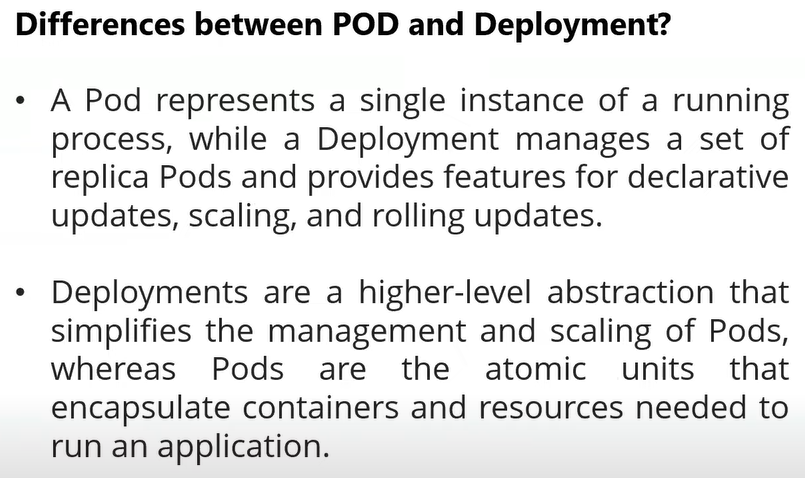


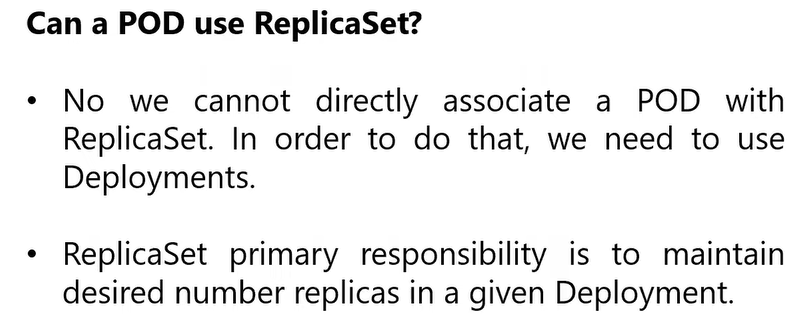
1. **Deployments**:

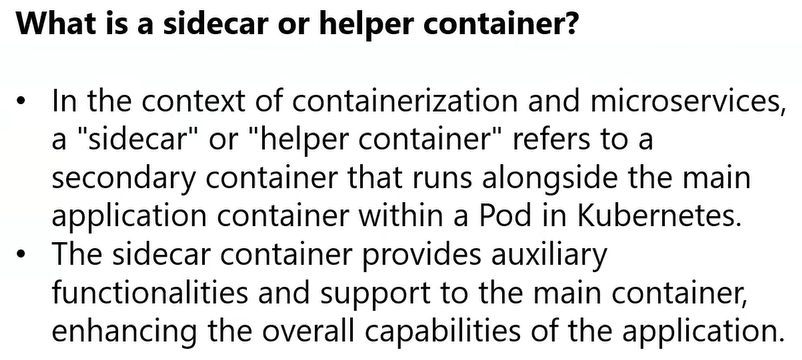


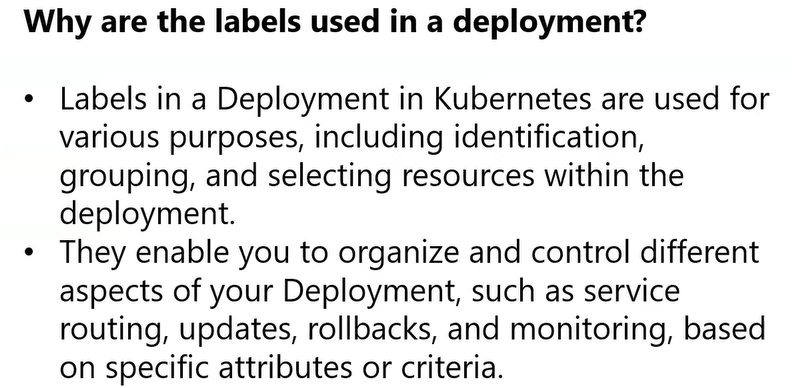
1. **Replica Set**:



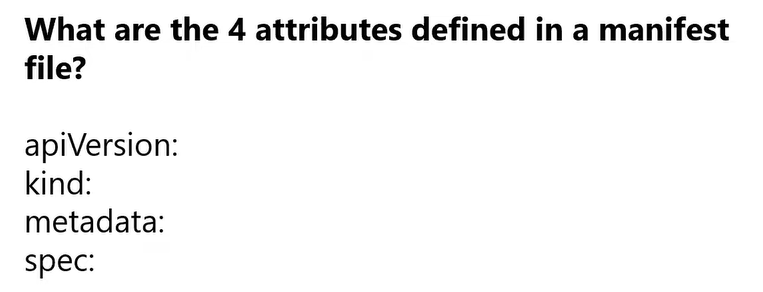


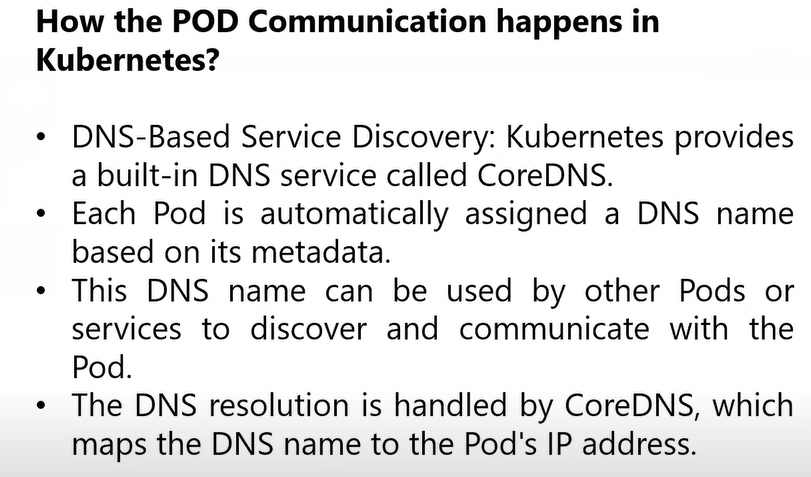


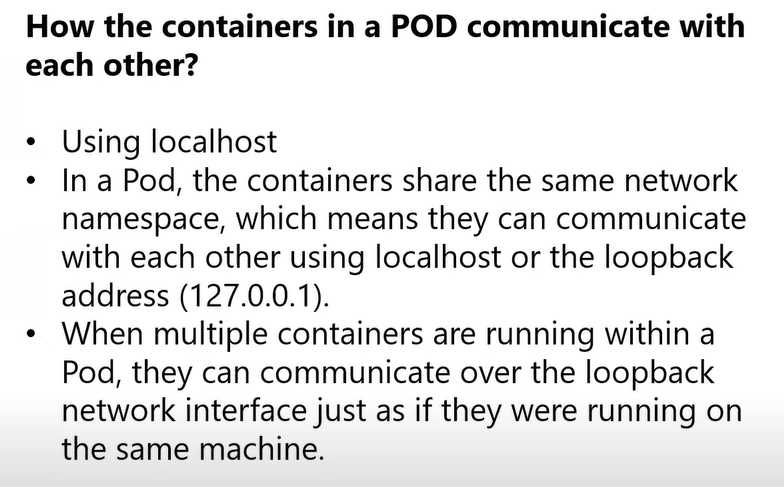










s

**Commands**:

1. Kubectl get all -a: it will show all the workloads/components/resources in the cluster

What is Kubectl?

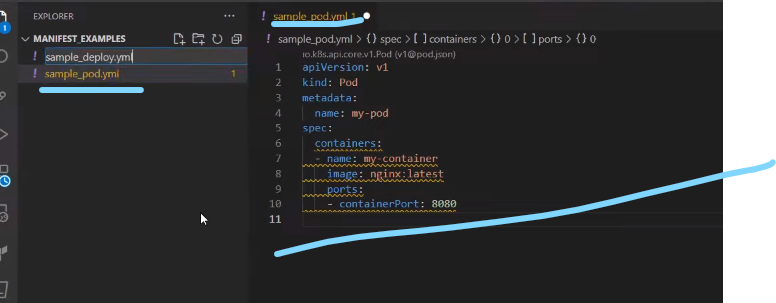
Kubectl is a client-side application, to interact with k8s cluster Kubectl is mandatory.

1. Kubectl get pods: it will show all the pods in the cluster.
2. Kubectl get deployments: it will show all the deployments in the cluster.
3. Kubectl get replica sets: it will show all the replica sets in the cluster.
4. Kubectl apply -f <yamlfile>: create the workload (i.e pod/pods, deployments, replica sets etc) in the cluster
5. Kubectl delete -f <yamlfile>: deletes the workload in the cluster.

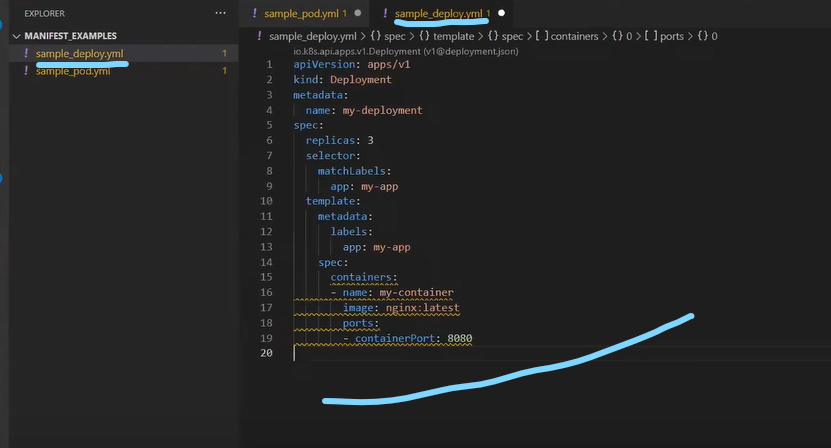
Now create 3 yaml files to practice the pods,deployments and replicasets.

Ingeneral we create deployments only in which we maintained pods and replicasets.

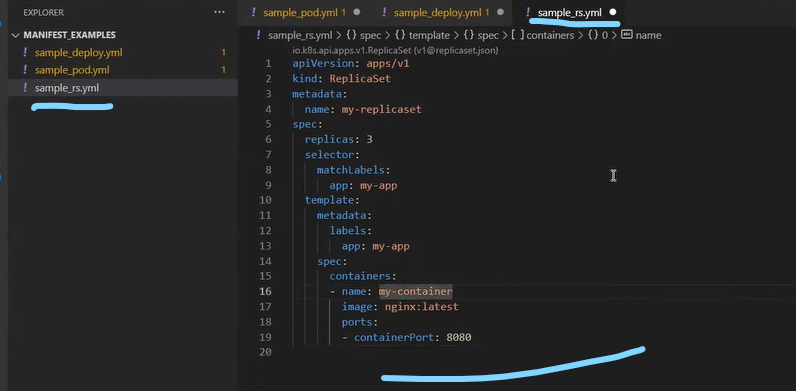
1. Pod yaml file: sample\_pod.yml

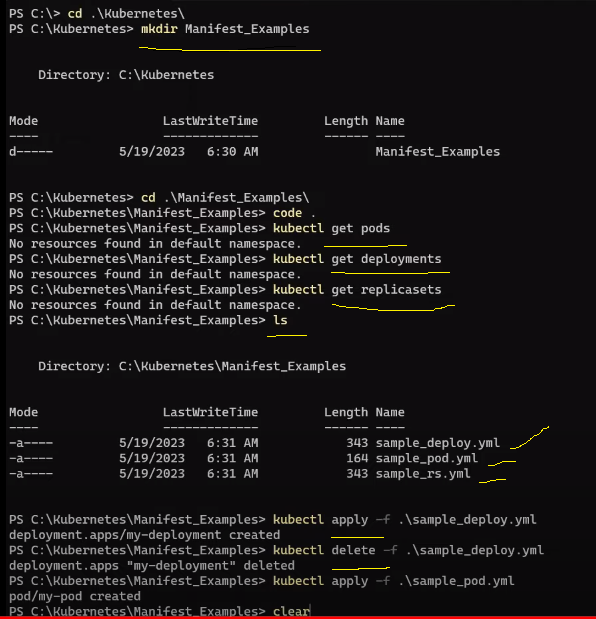


1. Deploy yaml file : sample\_deploy.yml

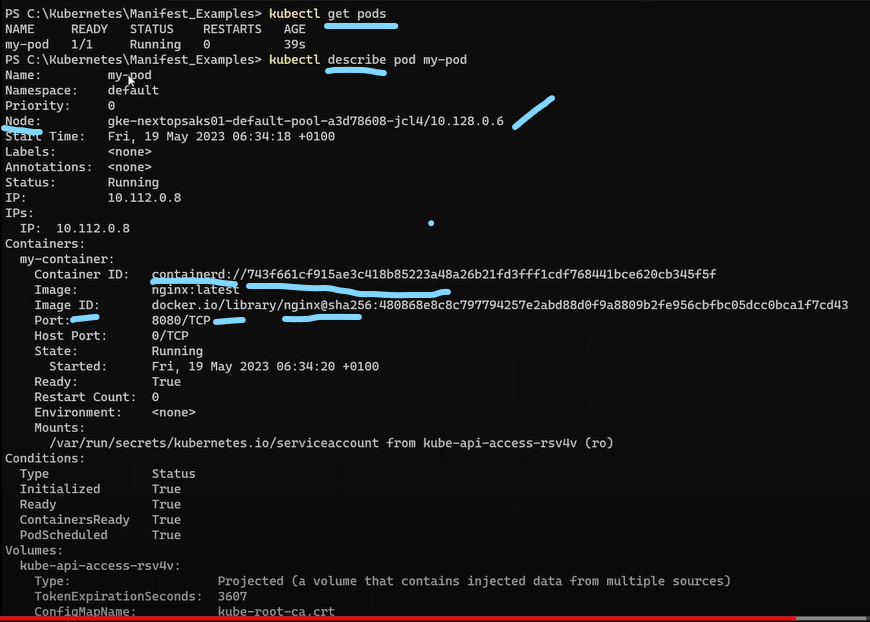


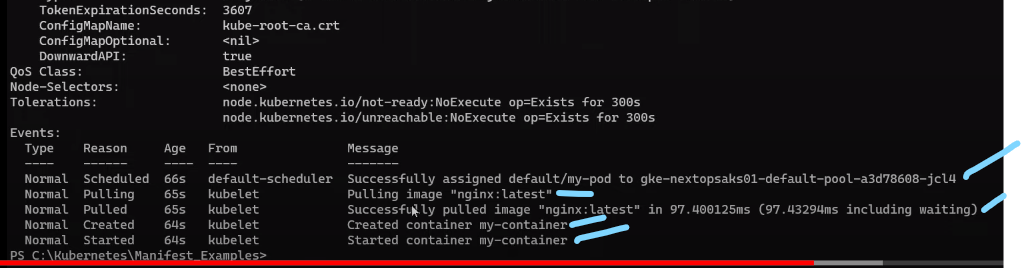
1. Replicaset yaml file : sample\_rs.yml



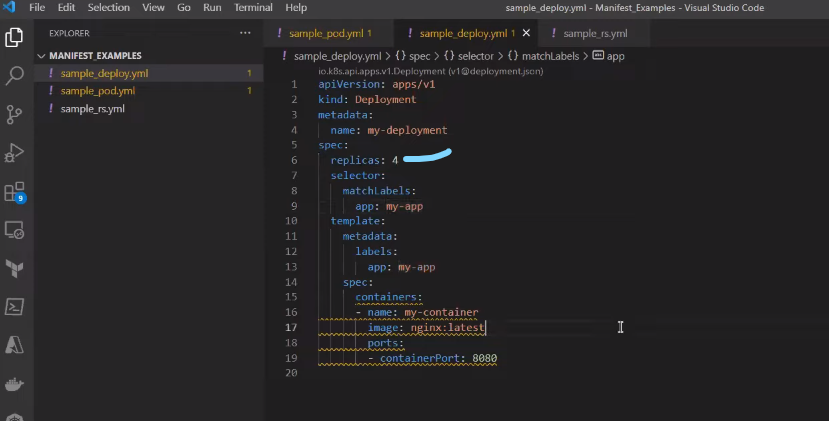


1. Kubectl describe pod <podname>: shows all the detailed info about the pod in the cluster.

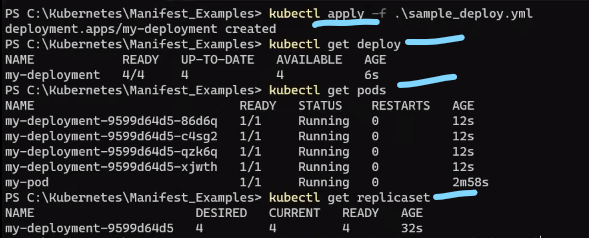


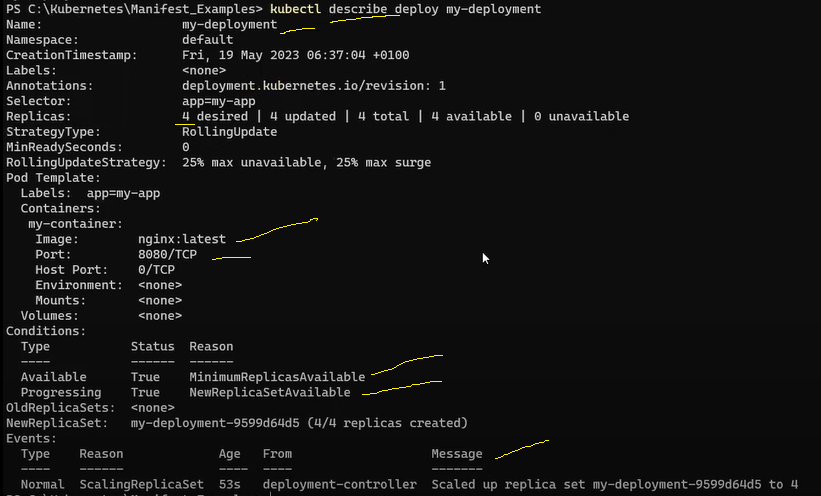


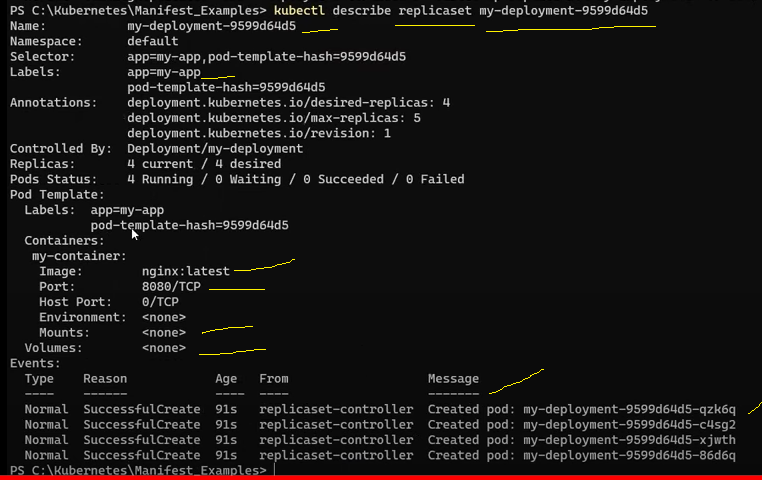
Now change replicas to 4 in deploy.yml



Now create the deployment

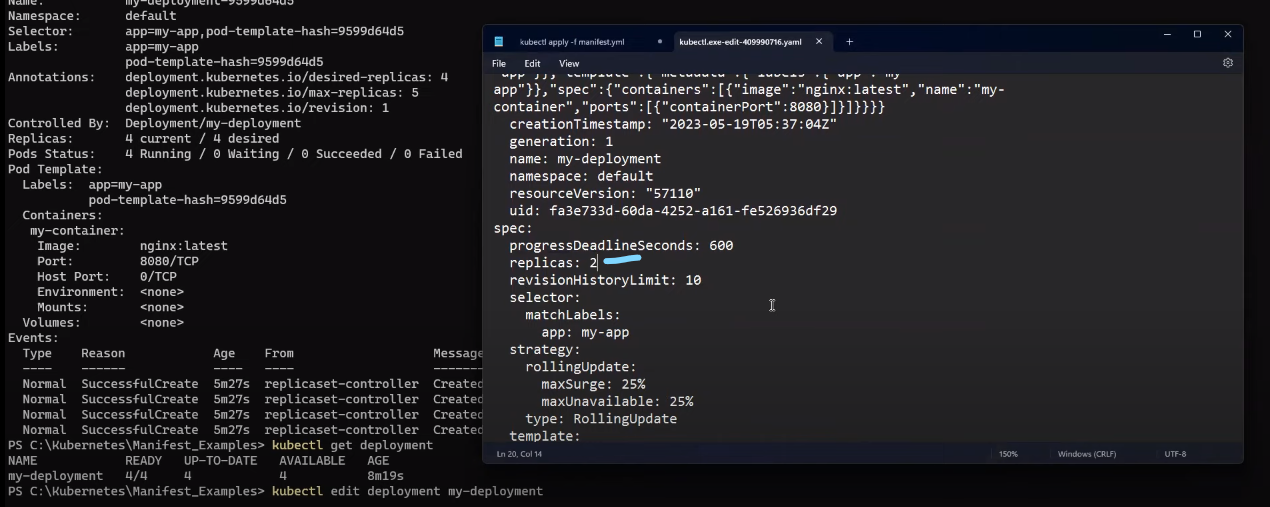


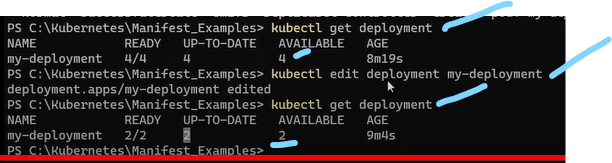




1. Kubectl edit deployment < deployment name>: we can edit the deployments like we can change the pods no and replicas only. e while running the cluster.

After applying this cmd it will open our deployment.yml file where we can change the workload resources values.





1. Kubectl delte pods -w: here -w means watch. Open 2 windows and delete the any pod and observe in the other window it will delete and again automatically creates another pod.

