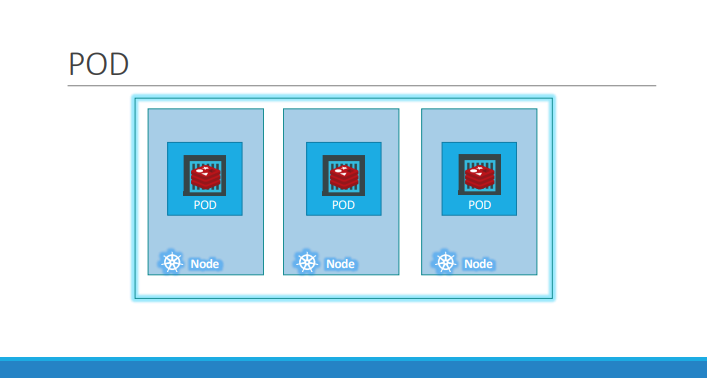
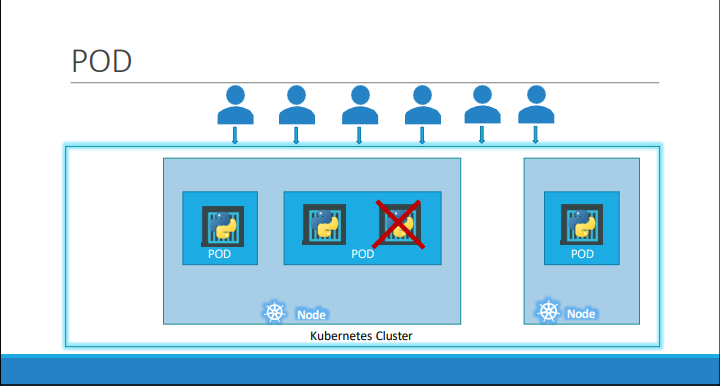
**Kubernetes Concepts**

**Kubernetes pods :**



As we discussed before, with kubernetes our ultimate aim is to deploy our application in the form of containers on a set of machines that are configured as worker nodes in a cluster. However, kubernetes does not deploy containers directly on the worker nodes. The containers are encapsulated into a Kubernetes object known as PODs. A POD is a single instance of an application. A POD is the smallest object, that you can create in kubernetes.

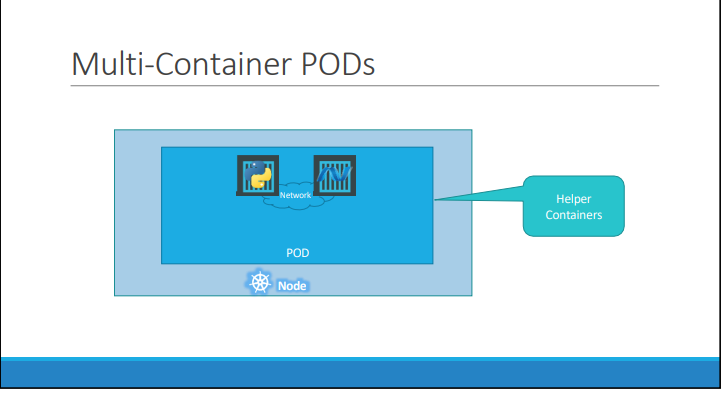


Here we see the simplest of simplest cases were we have a single node kubernetes cluster with a single instance of our application running in a single docker container encapsulated in a POD. What if the number of users accessing our application increase and we need to scale our application? We need to add additional instances of our web application to share the load.

Now, were would we spin up additional instances? Do we bring up a new container instance within the same POD? No! We create a new POD altogether with a new instance of the same application. As we can see, now we have two instances of our web application running on two separate PODs on the same kubernetes system or node.

What if the user base FURTHER increases and our current node has no sufficient capacity? Well THEN we can always deploy additional PODs on a new node in the cluster.

We will have a new node added to the cluster to expand the cluster’s physical capacity., PODs usually have a one-to-one relationship with containers running our application. To scale UP we create new PODs and to scale down we delete PODs. We do not add additional containers to an existing POD to scale our application.



Now we just said that PODs usually have a one-to-one relationship with the containers, but, are we restricted to having a single container in a single POD? No! A single POD CAN have multiple containers, except for the fact that they are usually not multiple containers of the same kind.

As we discussed , if our intention was to scale our application, then we would need to create additional PODs. But sometimes we might have a scenario were we have a helper container, that might be doing some kind of supporting task for our web application such as processing a user entered data, processing a file uploaded by the user etc. and we want these helper containers to live along side our application container.

In that case, we can have both of these containers part of the same POD, so that when a new application container is created, the helper is also created and when it dies the helper also dies since they are part of the same POD. The two containers can also communicate with each other directly by referring to each other as ‘localhost’ since they share the same network namespace. Plus they can easily share the same storage space as well.

Some Terms :

Kubectl => Kubectl is a command-line tool that allows users to manage Kubernetes clusters.

VirtualBox =>VirtualBox is a virtualization software that allows users to run multiple operating systems on a single computer

MiniKube =>Minikube is a tool that allows you to run Kubernetes locally on your computer

A note about creating pods using **kubectl run**.

To create a pod from the command line, use the command:

**Create an NGINX Pod**

kubectl run nginx --image=nginx

As of version 1.18, kubectl run (without any arguments such as --generator ) will create a pod instead of a deployment.

To create a deployment using imperative command, use **kubectl create:**

kubectl create deployment nginx --image=nginx

//Additional commands for information

kubectl get pod

kubectl describe pod

kubectl get pod -o wide

Kubernetes Concepts – <https://kubernetes.io/docs/concepts/>

Pod Overview- <https://kubernetes.io/docs/concepts/workloads/pods/pod-overview/>