Kubernetes Objects and Kubernetes command

Kubectl run pod_name – image image-name
Kubectl create/apply -f yaml-file
Kubectl get pods/nodes
Kubectl get pods -o wide
Kubectl delete pod pod-name

Kubeclt create deployment –help

controlplane ~ ➜ kubectl run nginx-pod --image nginx:alpine
kubectl run redis --image redis:alpine --labels tier=db
pod/redis created

POD

Container in a pod can talk directly by referring to localhost since they belong to same network
Also same storage space.

Kubectl run nginx — this command deploy a docker container by creating a pod
First it create a pod automatically and then deploy an ionstance of nginx docker

Kubectl run nginx –image nginx
Kubectl get pods

Yaml file

```
apiVersiion: v1                    pod & service ==v1 , rs & deployment = apps/v1
Kind: Pod
Metadata:    ///data about obhect
         Name: myapp-pod
         Labels:
                  App: myapp
Spec:    /// what inside this object
   Container:    /// list or array
    -    Name: nginx-container
         Image: nginx
```

Kubectl create -f name-of-yaml-file

Kubectl get pods
Kubectl describe pod pod-name

```
apiVersion: v1
Kind: Pod
Metadata:
        Name:  nginx
        Labels:
                App: nginx
                Tier: frontend

Spec:
        containers:
                -  name:nginx
                   Image: nginx
                -  name : nginx2  // how to add more conatiner
                   Image: busybox
```

Kubectl create -f name-of-yaml-file
Kubectl apply -f name-of-yaml-file // when we are creating new object
Kubectl get pods
Kubectl describe pod pod-name

Kubectl get pods -o wide == view more information

Kubectl delete pod pod-name

—------
How to generate yaml file

Kubectl run redis –image=redis123 –dry-run -o yaml == depricated
Kubectl run redis –image=redis123 –dry-run=client -o yaml == new version

Kubectl run redis –image=redis123 –dry-run=client -o yaml > redis.yaml
Cat radis.yaml
Kubectl create -f redis.yaml == create pod

Change and then run pod
1 . if you have yaml file go to file and change the requirement
And then run the following the command

Kuubectl apply -f redis.yaml == this not create new pod but onlu change the updated version
Ffff

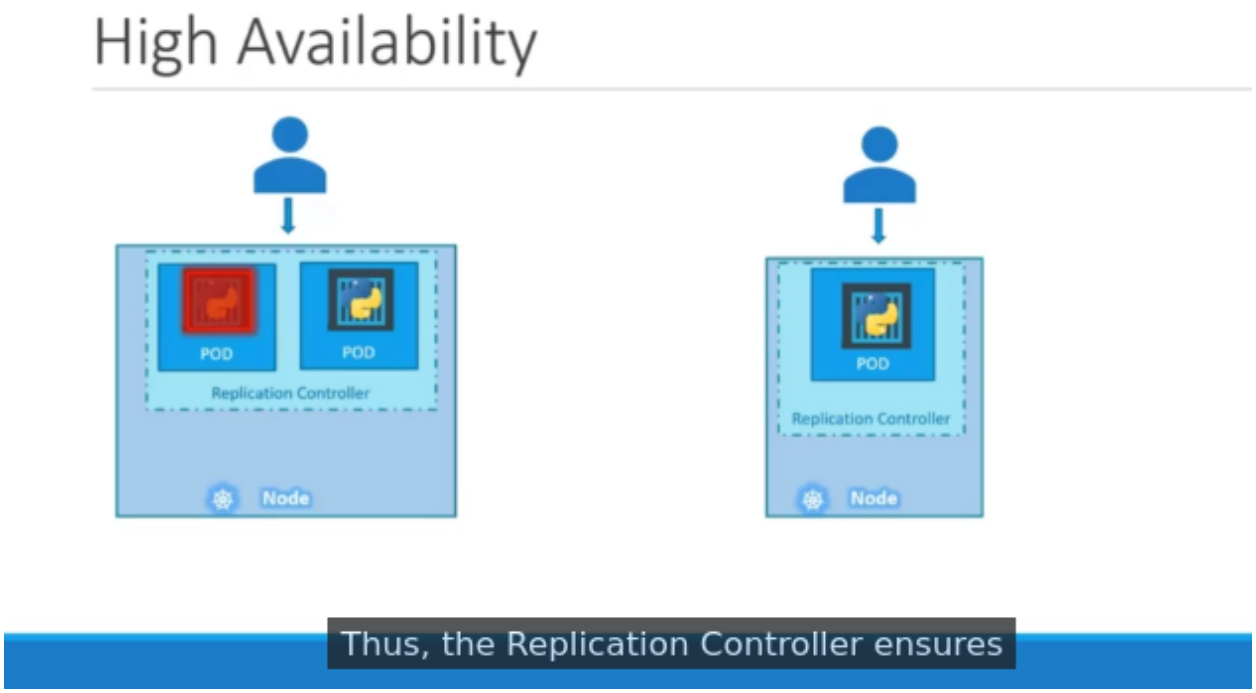
ReplicaController

If pod fails then user will no lonmger able to access application

To prevent this we uses replicaController or replica set

For single pod or more than one pod

Existing fails rc or rs brings new pod automaticaally

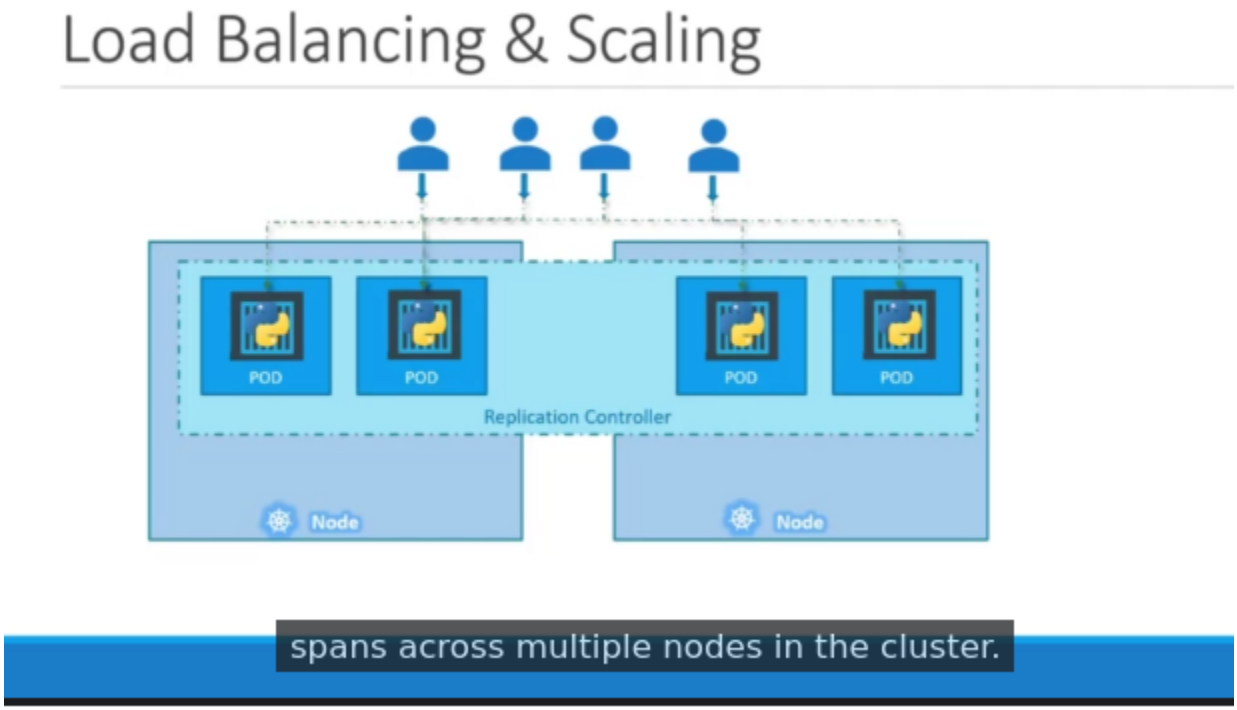


It provides

Load balancing
Scaling
Auto scaling

Same node or multiple node

Rc or rs haS the ability to span over node



Rc or rs

Older technology - new version

Rc

apiVersion: v1
Kind: ReplicationController
Metadat:
Name: myapp-rc
‘ labels:
App:myapp
Type: front-end

Spec; it contains template to create pod

Move all field of pods except kind and apiversion

Metadata:

Name: myapp-pod

Labels:

App: myapp

Type: front-end

Spec:

Containers:

-Name: nginx-controller

Image: nginx

Selector: optional auto tales the labels of same file

Replicas: 3 how many pods we want

After this we run

Kubeset create -f rs-yamil-file-.yaml it first create pod by using template file

Kubectl get relicationcontroller or rc

Kubectl get pods == name of pod is not what we given in pod metadata but name of pod is equal to name of rs or rc and some random number

ReplicaSet

```
apiVersion:apps/v1
Kind: ReplicaSet
Metadata:
        Name: myapp-replicaset
        Labels:
                App: myapp
                Type: front-end
Spec:
        Template:
                Metadata:
                        Name:myapp-pod
                        Labels:
                                App: myapp
                                Type: front-end
                Spec:
                        Containers:
                                - Name: mginx-conatiner
                                  Image: nginx
        Replicas: 3
```

Rs requires a selector definition compulsory

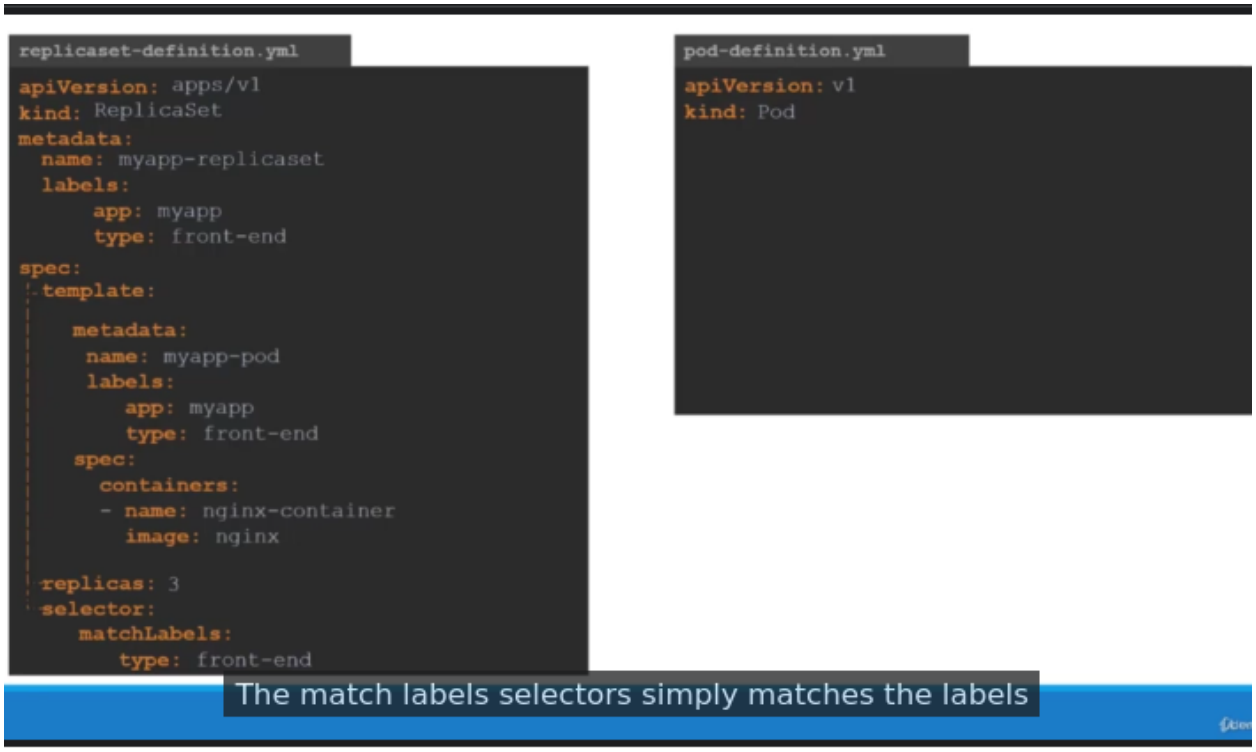
Rc me optional hota hai by default it takes label from pod definition file

Selector section helps to identify what pods fall under it
Because rs also consider pods which are not part of template
It world on labels.

All pods come in rs ==== same label

All pods which are created before the creation of rs comes unser rs if they have created with same label

```
        Selector:
                matchLabels:
                        Type: front-end
```



Selector in rs provides many options for matching that are not available in rc

Kubectl create -f rs-file-name.yaml
Kubectl get rs or replicaset
Kubectl get pods

Here also name of pod === name of rs+some random number

Rs monitor existing pod == in tis case also we need t provide pod template
And if not created then create it

Rs is a process that monitor a pod
Provides label to filter

How to scale

Currently replica = 3
Future replica = 6

1 . go to yaml file update the replica section to 6
And then run the command

Kubectl replace -f rs-name.yaml

2. By input file name

Kubectl scale -- replicas=6 -f rs-file-nsame.yaml

This command will not update replicas in the es-file-name.yaml file automaticall y

3 . by input rs -name in name format

Kubectl scale -- replicas=6 replicas rs-name

4. Scale rs based on load

Kubectl delete rs/replicaset rs-name == it will delete all pods also

Kubectl edit rs rs-name == to edit the rs object
=== it automatically update

Create and recreate pods

Deployments

Same as rs
But it provides additional features

Rolling update == update to newer versions if availale
Rolling back == if updateions failed it roll back completely roll back changes immediately

Update one by one not all at a time because it affect applicatoopn
Upgrade one after other

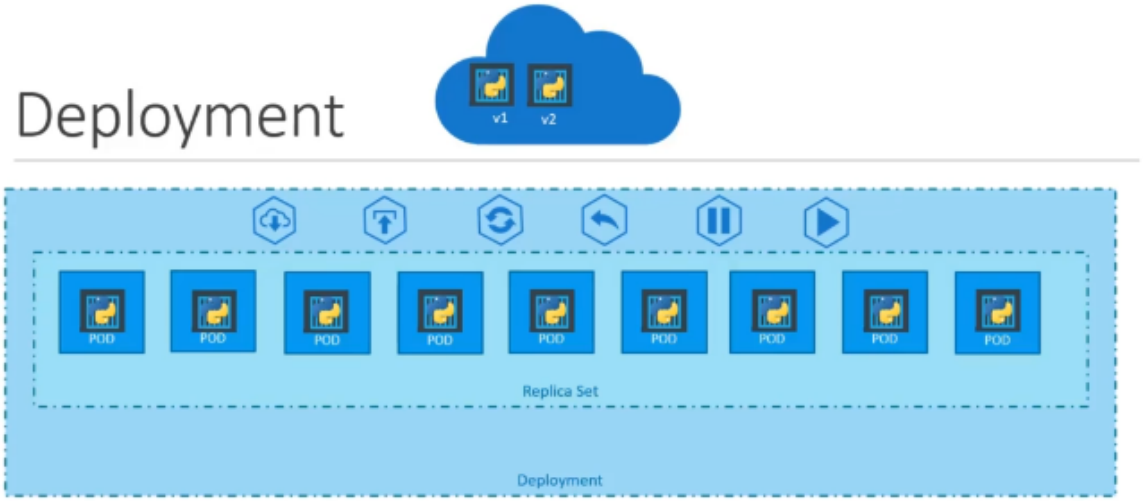
Also it make possible to make multiple changes at a time

Pause the environment
Make changes
resume the environment so that all the changes rolled out together

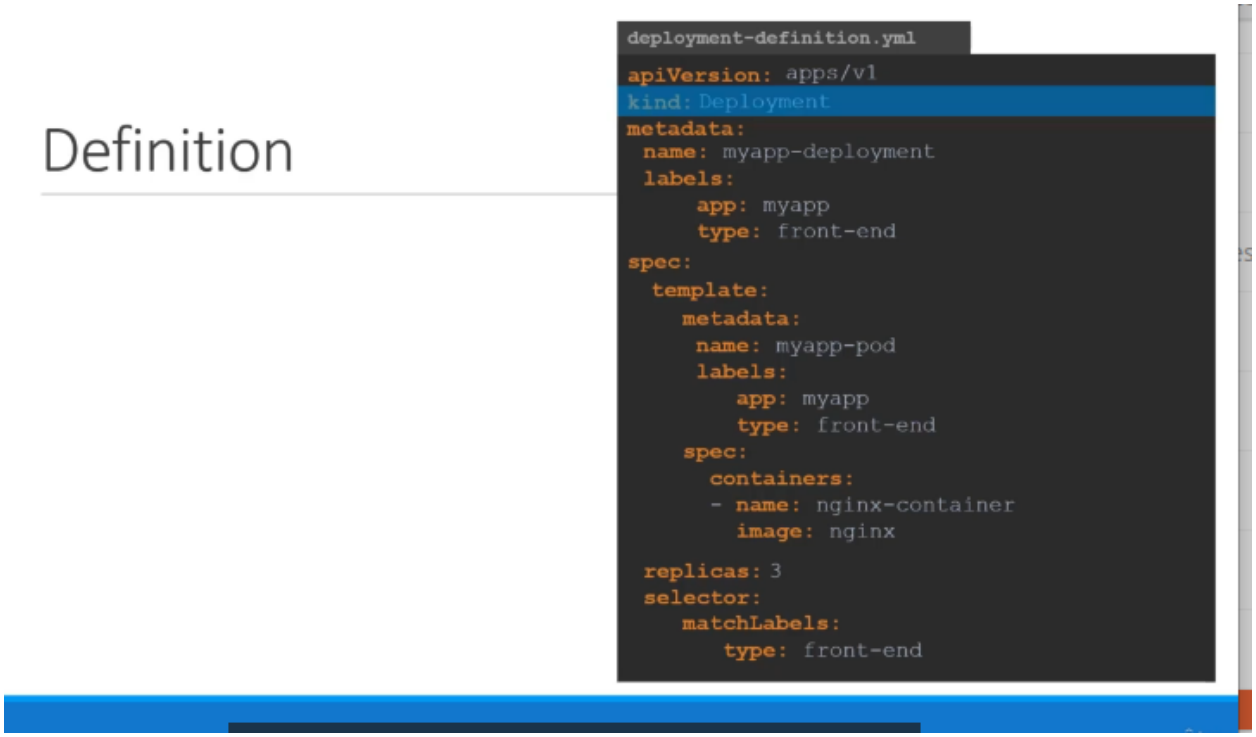
Pods == deploy single instances of our application
Container is insulated in pods

And multiple such pods are deployed using rs or rc

D



Deployment is similar to rs yaml file



Kubectl create -f deploy-file-name.yaml
Kubectl get deployments or deploy

Kubectl create deployments or deploy name-of-deplouyment –image-image-name --replicas=3
Kubectl get rs

Deployment automatically a replica set

Rs then automatically create pod

Pod name = rs-name+randome number

Kubectl get all == to see tha all object

Create an NGINX Pod

kubectl run nginx --image=nginx

Generate POD Manifest YAML file (-o yaml). Don't create it(--dry-run)

kubectl run nginx --image=nginx --dry-run=client -o yaml

Create a deployment

kubectl create deployment --image=nginx nginx

Generate Deployment YAML file (-o yaml). Don't create it(--dry-run)

kubectl create deployment --image=nginx nginx --dry-run=client -o yaml

Generate Deployment YAML file (-o yaml). Don't create it(--dry-run) with 4 Replicas (--replicas=4)

kubectl create deployment --image=nginx nginx --dry-run=client -o yaml > nginx-deployment.yaml

Save it to a file, make necessary changes to the file (for example, adding more replicas) and then create the deployment.

kubectl create -f nginx-deployment.yaml

OR

In k8s version 1.19+, we can specify the --replicas option to create a deployment with 4 replicas.

kubectl create deployment --image=nginx nginx --replicas=4 --dry-run=client -o yaml > nginx-deployment.yaml