**< ------------------------------ KUBERNETES COMMNADS ------------------------->**

**for all type of resource & complete list of suppourcesorted res { apiVersion , namespaced , kind }**

1. kubectl api-resources | less ( list of complete resource )
2. kubectl api-resources |grep -i pod ( eg: pod)
3. kubectl api-resources | grep -i configmaps

**Note ::: for making short cmnd use ‘alias’**

1. alias **kgn**="kubectl get nodes"
2. alias **kgp**="kubectl get pods"

**Note ::: for knowing something**

1. kubectl explain pods |less (for ApiVersions)
2. kubectl explain pod.metadata |less (check what is inside metadatafor yml)
3. kubectl explain rs --recursive |less
4. kubectl explain deploy --recursive |less

**Note :: create pod without yml ( with run comnd or imperatice comnds)**

1. kubectl run nginx --image=nginx --port=80 -v=7
2. kubectl run mypodname --image=nginx:1.16 --restart=Never
3. kubectl run mypodname --dry-run --image=nginx:1.16 --restart=Never (drrun)
4. kubectl run -i -t LOADgenerator --rm --image=busybox /bin/bash     { temporary pod }  (inside container)
5. kubectl delete pods --all ( delete all running pods )

Note ::

1. kubectl explain pod --recursive |less ( how to write YML )

2 ) kubectl run mypodname --dry-run --image=nginx:1.16 --restart=Never -o yaml

: --- cmnd no 2 show u output of YAML file

3 ) kubectl run mypodname --dry-run --image=nginx:1.16 --restart=Never -o yaml > mysecondpod.yml

:--- Cmnd no 3 show u output of YAML file in mysecondpod.yml

kubectl edit pods myfirstpod.yml (edit pods)

**apply is decleative ::: create is imperative**

Note : for finding some thing in .yml file. eg : find name in container

grep name second.yml

note :: port expose run cmnd (jo ‘**cluster-IP’** milegi wo worker node pe expose hogi ”curl **cluster-IP**:port”

kubectl expose pod thirrd --port=8000 --target-port=80 --name myfirst**service**

**nodeport :** bhr se serice use krsakte hain ‘**NodePort**’

kubectl expose pod thirrd --**type=NodePort** --port=8000 --target-port=80 --name mnodeportsvc

**kubectl -n kube-system get pods ( all control plane pods )**

**kubectl delete pods --all (delete pods alll)**

cp mypod-1.yml mypod-2.yml (copy 1 yml file to 2yml ile)

**S T A R T** ---------> Note :: 1 yml file copy into other file

note: all cluster what u create

kubectl get all -o wide

**Note : how to use ‘tmux’ in linux**

1 sudo apt-get install tmux

2 tmux

Now devide/split terminal

3 pres Ctrl+b & shift+%

Now 1 terminal to another

4 press Ctrl+b & o

5 exit

kubectl cluster-info

kubectl get events ( wht happened in background )

kubectl get nodes

kubectl get pods

kubectl config view kubectl config --help

kubectl get all -o wide

kubectl get pods -o wide ( full detail of PODS)

Follow technical guftugu LEC-48

hostnamectl set-hostname MASRTER ::: hostnamectl set-hostname WORKER-1

21 kubectl describe node <name/IP> ( cpu limit, memory limit, node status ,kubelet status )

kubectl describe pod < pod-name >

22 vi pod1.yml (for creat 1st POD, description box script copy in VI editor)

23 kubectl apply -f pod1.yml --dry-run ( aftr pasting script in VI apply this pods) pod1.yml podname

23 kubectl apply --dry-run -f pod1.yml

23 kubectl apply -f pod1.yml --dry-run=client

23 kubectl create -f pod1.yml

kubectl get all ( showing all pods , service , rs ,rc )

kubectl get endpoint

24 kubectl get pods ( checking pods running condition or not)

24 kubectl get pods -w ( watch pods)

24 kubectl get all ( give all pods , rs running condition )

25 kubectl get pods -o wide ( full detail of PODS name ready statud ip node endpoint )

curl -v/V 12.34.5.0.6:8080/mypodname ( internally access the pod or application )

wget -q0- 12.34.5.0.6:8080/mypodname ( internally access the pod or application )

26 kubectl describe pod testpod |less ( full detils like IP, events pods,ready,environment )

kubectl describe pod testpod -n testns (if your pod in testns name space)

kubectl get pods testpod -o yaml (dtails of yml format)

kubectl get pods testpod -o json (dtails of jsonformat)

27 kubectl logs -f testpod (full container details inside)

kubectl logs -f testpod -c c00

(more container details c00 is container) after cmnd prs CTRL+z

29 kubectl delete pod testpod kube (for deleting pod)

30 kubectl delete pods --all ( delete all running pods )

kubectl delete all --all ( delete all things like, pod , service , daemon set , rc , rs )

**\*\*\*\*\*\*\*Annotations\*\*\*\*\*\*\* ( for messege and descriptions, owner details)**

1 vi pod2.yml ( open pod in VI) and paste annotation script from tech guftugu LEC 48 )

2 kubectl apply -f pod2.yml

3 kubectl describe pod testpod ( inside pod2.yml pod name is testpod)

4 kubectl delete pod testpod delete ( inside pod2.yml pod name is testpod)

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* multicontainer pod \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

4 vi pod3.yml ( paste MULTI CONTAINER SCRIPT in vi pod3.yml )

5 kubectl apply -f pod3.yml

6 kubectl get pods ( bcz multi container **testpod3 2/2 Running)**

7 kubectl describe pod testpod3 (testpod3 is inside VI name)

8 kubectl logs -f testpod3 -c c00 (go to contianer ( c00 in container name MULTI CONTAINER)

9 kubectl exec testpod3 -c c00 -- hostname -i ( IP of containers pod ) all IP same bcz ip gave to pod not containeer ↓ **if u want to know container ip first go inside container)**

10 kubectl exec testpod3 -it -c c03 -- /bin/bash **( u r inside container) cmnd 11 12 inside continer)**

**RESULT-🡪 root@testpod3:/#**

11 ps

12 ps -ef

13 kubectl delete -f pod3.yml ( delete pod through .yml file)

1 . kubectl exec -it multicontainer-pod -- /bin/bash (bydefault going to 1st contaomer)

2 . kubectl exec -it multicontainer-pod -c **2nd** -- /bin/bash (2nd cont)

**\*\*\*\*\*\*\*\* ENVIRONMENT VARIABLE IN pods \* \*\*\*\*\*\*\*\*\*\*\***

14 vi pod4.yml (env variable script paste in this script)

15 kubectl apply -f pod4.yml (apply this)

16 kubectl exec environments -it -- /bin/bash **{result (root@environments:/)#}**

17 kubectl exec environments -it -c c00 -- /bin/bash **result (root@environments:/)#**

18 env

19 # echo $MYNAME (SHOWING ENV variable. In this eg: echo $MYNAME → AAMIR IMRAN AHMED bcz hum ne wahi dala hai)

20 exit (AND) kubectl delete -f pod4.yml ( .yml POD DELTE)

**\*\*\*\*\*\*\*\*\*\* POD WITH PORTS\*\*\*\*\*\*\*\***

1 vi pod5.yml

2 kubectl apply -f pod5.yml

3 kubectl get pods

4 kubectl get pods -o wide (pod IP and instance IP)

5 curl <pod IP>:80 ( not working )

Follow technical guftugu LEC-49

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* LABELS \*\*\*\*\*\*\*\*\*\*\*\*\*\***

When machine stop and u start the machine connect with putty

Sudo su → minikube status → ex

1 vi pod6.yml ( paste script of LABEL in this) MAKING manifest DECLERATIVE LEBL

kubectl label pods podname env=testing type=frontend ( imperative method to label )

kubectl label --overwrite pods podname cc=dd bb=aa ( overwrite label imperative )

kubectl label pods podname env- ( for unlabeled pod )

kubectl label pods --all new=pod ( label all pods in same ns mperative )

2 kubectl apply -f pod6.yml result ( pod/delhipod created )

3 kubectl get pods (**pod name is delhipod)**

3 kubectl describe pod podname (seeing label. And diifent things)

4 kubectl get pods -o wide ( full pod details IP of pod)

5 kubectl get pods --show-labels ***class=pods,env=developments*** **(check labels . findout pod with labels)**

**kubectl get nodes --show-labels** --------------- **check labels on nodes**

6 kubectl label pods delhipod myname=AAmir pod/delhipod labeled **( imperative label comnd)**

7 kubectl get pods --show-labels

8 kubectl get pods -l env=developments **( find pod with LABEL)**

kubectl get pods -l class=pods,env=developments (jis pod pe y 2 label ho use dikhao)

9 kubectl get pods -l env!=developments ( ! this marks showing jis pr ye label n alga ho wo pod)

10 kubectl delete pod -l env=developments ( pod delete with jispr ye label lga ho)

11 kubectl get pods -l 'env in(developments,testing)' { find pod with multiple label}

12 kubectl get pods -l 'env notin(developments,testing)' { jis me ye label nhi lga ho wo pod}

13 kubectl delete pods -l 'env notin(developments,testing)' {jis pod pr ye labelna lga ho usko delete}

13 kubectl delete pods -l 'env in(developments,testing)' {jis pod pr ye label lga ho usko delete}

For remove the labels

**kubectl label nodes ip-172-31-44-220(worker-1) node-**

**\*\*\*\*\*\*\*\*\*\*\*\*\* NODE SELECTOR \*\*\*\*\*\*\*\*\*\*\*\*\*\***

14 vi pod7.yml

15 kubectl apply -f pod7.yml

16 kubectl get pods PODnme= nodelabels result( nodelabels 0/1 Pending 0) bcz NODESELECTOR T2-medium

17 kubectl describe pod nodelabels (details of pod nodelabels ) didn't match Pod's node

kubectl describe nodes ip-172-31-43-101 (worker-1) | less

18 kubectl get nodes  ***( for labelling on nodes 1st checknodes)***

19 kubectl label nodes minikube hardware=t2-medium **( minikube is node) now labeleing on it**

20 kubectl describe pod nodelabels POD-name= nodelabels ( ab use ne krdiya)

21 kubectl get pods result ( nodelabels 1/1 Running 0)

21 kubectl delete -f pod7.yml

**\*\*\*\*\*REPLICATION CONTROLLER or POD CONTROLLER \*\*\*\*\*\*\*\*\*\*\***

(RC WORKS EQUALITY BASED SELECTOR) (It can only monitor 1-st of abels ….eg: team=dev )

**REPLICATION CONTROLLER will support for rolling update of the applications**

**kubectl explain rc | less kubectl explain rc --recursive | less (for yml file )**

kubectl run web-server --image=nginx

kubectl label pod web-server team=dev ( assign label on pod)

kubectl label pod web-server run- (un labels )

kubectl label pod web-server team=dev

kubectl label pod web-server team- (un label)

**kubectl get rc,pods --show-labels**

kubectl delete pods --all

1 vi pod8.yml

2 kubectl apply -f pod8.yml result (replicationcontroller/myreplica created)

3 kubectl get rc NAME DESIRED CURRENT READY AGE

myreplica 5 5 5 112s

**kubectl describe rc**

3. kubectl get rc,pods

3. watch kubectl get rc,pods --show-labels

3. kubectl describe pod rc-demo-8mf6m

4 kubectl describe rc myreplica ( full details Rc) myreplica = name of (ReplicationController

5 kubectl get pods ( pods running status) 5 pods with replications

**Note :::**  **If any 1 pode will down . due to RC new pod (replicaton take place lets see)**

6 kubectl delete pod myreplica-7cx2r (myreplica-7cx2r one of RC node)

7 kubectl get pods ( aftr deleting 1pod stil new pod creat with new IP .5 pod prnt showing bcz RC)

8 kubectl get rc

9 kubectl describe rc myreplica ( full details Rc) showing 6 replica 1 deletedkubectl

10 kubectl get pods --show-labels (showing label)

11 kubectl scale --replicas=8 rc -l myname=Bhupinder **label=Bhupinder** **( scale up replicas imperative )**

kubectl get pods

kubectl get rc ( check desired and current,

**(cmnd 11 scale up myname=Bhupinder { ye RC label ke replica ko } this labelled 8 repl)**

**Note:** **RC scale through ‘edit’ Open vi editor**

11 kubectl edit rc myreplica **replicationcontroller/myreplica edited** ( myreplica = RC name inside yml)

12 kubectl get rc ( check desired and current, showing 8 )

13 kubectl get pods (8replica showing)

14 kubectl scale --replicas=1 rc -l myname=Bhupinder

**( cmnd 14 scale down myname=Bhupinder this labelled, 1 repl)**

15 kubectl get rc

16 kubectl get pods

17 kubectl get pods --show-labels

18 kubectl delete -f pod8.yml **(if u want to delete RC delte yml file )**

19 kubectl get pods (showing terminate pods)

**Note:: this cmnd 20 delete RC but don’t delete pods**

20 kubectl delete rc --cascade=false myreplica (myreplica = RC name )

21 kubectl delete -f first-rc.yml ( delete YML file amd all RC )

22 kubectl delete rc --all

**\*\* EXAMPLE OF REPLICA SET \*\* replication controller advance version is REPLICA SET \*\*\***

**(EQUALITY AND SET BASED SELECTOR**) (It can monitor diffrnt labels like ….eg: team=dev …. Team=prod )

**REPLICA-SET will not support for rolling update of the applications running in pods**

**kubectl label pod dev run- (for unlabeling pod …. dev=pod name)**

**kubectl label pod test team=test (for label podname=test ,, label ;;test )**

1. vi pod9.yml

1 cat pod9.yml ( for seeing inside manifest or yml)

2 kubectl apply -f pod9.yml result( replicaset.apps/myrs created)

kubectl get pods

**kubectl get all ( showing all pods , service , rs )**

1. kubectl get rs
2. **kubectl get rs,pods --show-labels**

4 kubectl scale --replicas=1 rs/myrs ***scale down*** ( myrs is inside my pod9.yml Repliaset name)

kubectl scale rs myrs --replicas 5 ***scale up*** ( myrs is inside my Repliaset name)

5 kubectl get pods ( showing after replica scale state how many pods are there)

6 kubectl get rs ( showing desired and current state)

7 kubectl delete pod myrs-9jnpg ***kubectl get pods*** (pod kubectl delete pod myrs-9jnpg one of RS node)

8 kubectl get rs ( showing desired and current state)

9 kubectl get pods ( after deeting 1pode you will see 1 new pod created due to RS)

10 kubectl delete rs/myrs result (replicaset.apps "myrs" deleted)

11 kubectl get rs ( No resources found in default namespace.)

12 rm -rf rm -rf pod1.yml pod2.yml ( deleting .yml / manifest )

**kubectl –help**

**kubectl --help | less**

Follow technical guftugu LEC-50

**\*\*\*\*\*\* EXAMPLE OF DEPLOYMENT (work with Replic Set)\*\*\*\*\*\*\*\*\***\*

1 vi mydeployment.yml

2 kubectl apply -f mydeployment.yml result(deployment.apps/mydeployments created)

**(kind=Deployment , name= mydeployments)**

3 cat mydeployment.yml

1. kubectl get pods
2. kubectl get all
3. kubectl get all --show-labels
4. kubectl describe deployment nginx-deployments (deploymemnt name= nginx-deployments ) **for old new**

kubectl get deploy ( check deployment was created or not) mydeployments 2/2 2 2 deploy=type

6 kubectl describe deploy mydeployments ( mydeployments=deploy name iside yml ( how deployment creats Rs and pods)

7 kubectl get rs ( showing desired and current state

8 kubectl get pods ( bcz u describing in .yml mydeployments= mydeployments-646ddb49c-mmc

9 kubectl delete pods mydeployments-646ddb49

10 kubectl get pods ( after delete 1 pod due to RS create new onepod)

11 kubectl get rs (showing 2 pods running)

11 watch "kubectl get rs" ( watching )

12 kubectl scale --replicas=1 deploy mydeployments **( recently created pod delete) (scaledown ) deploy=**type

12 kubectl scale --replicas=4 deploy mydeployments **( recently created pod delete) (scaledUP ) deploy=**type

kubectl get rs **((( NAME DESIRED CURRENT READY AGE )))**

13 kubectl get deploy **((( NAME READY UP-TO-DATE AVAILABLE AGE )))**

(due to scledown 1/1 this is desired we obain)

14 kubectl get rs -o wide

15 kubectl get pods (copy **pod name** here before checking logs)

16 kubectl logs -f <podname> ***(what is running inside container ) print inside the container***

**Note ::: Nowchange some thing in mydeployment.yml thorough vi (2 replica)**

17 vi mydeploy.yml

18 kubectl apply -f mydeploy.yml

19 kubectl get deploy NAME READY UP-TO-DATE AVAILABLE AG

mydeployments 2/2 2 2 70m

20 kubectl get rs NAME DESIRED CURRENT READY AGE

New RS create hua hai ---------------🡪 mydeployments-55568dc64b 2 2 2 6m4s

**old code ko stop krdiya h--------------🡪 mydeployments-6b5557447c 0 0 0 88m**

21 kubectl get pods NAME READY STATUS RESTARTS AGE

mydeployments-55568dc64b-hqnnq 1/1 Running 0 12m

mydeployments-55568dc64b-th95g 1/1 Running 0 12m

22 kubectl logs -f mydeployments-55568dc64b-hqnnq (see inside the container)

23 kubectl exec mydeployments-55568dc64b-hqnnq -- cat /etc/os-release **(check os-releas)**

**Note :: when image vesion change**

Eg : image: coolgourav147/nginx-custom  ***to*** image: coolgourav147/nginx-custom:v2

24 kubectl apply -f first-deploy.yml;watch "kubectl get rs -o wide"

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ROLLOUT \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

24 kubectl rollout status deployment mydeployments ( deployment = type….. mydeployments =name)

Result (deployment "mydeployments" successfully rolled out)

25 kubectl rollout history deployment mydeployments (check rollout history)

26k kubectl rollout undo deploy/mydeployments **RESULT**  deployment.apps/mydeployments rolled back

( pichle version me jane keliye) undo

26 kubectl get pods

**Note: (pods utne hi rhenge jitney current me hong… Result** ( deployment.apps/mydeployments rolled back)

27 kubectl get deploy

28 kubectl get pods ( pod new bnega ka same hoga)

29 kubectl logs -f mydeployments-6b5557447c-qxtwf **( pichle walek k code dikhega)**

30 kubectl exec mydeployments-6b5557447c-qxtwf -- cat /etc/os-release ( due to ‘roleout’ previous image showed to you)

**Note :: rollout**

31 kubectl rollout

31 kubectl rollout history deploy firstdeploy ( firstdeploy=deploy name inside yml ) --showing none—

31. kubectl rollout history deployment nginx-deployments --revision=1 (more details )

32 kubectl apply -f first-deply.yml --record=true --showing record name when ur image version change—

33 kubectl get deploy firstdeploy -o yaml | less result ( kubernetes.io/change-cause: )

34 kubectl rollout undo --to-revision=2 deploy/deployment firstdeploy ( when u want to roll back which u want )

35 kubectl rollout pause deployment firstdeploy

36 kubectl rollout resume deploy firstdeploy ( firstdeploy= deploy name inside yml)

Follow technical guftugu LEC-51

**\*\*\*\*\* container to container communication on Same pods \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

1 vi pod1.yml

2 kubectl apply -f pod1.yml

3 cat pod1.yml

4 kubectl get pods result(testpod 2/2 Running)

5 kubectl exec testpod -it -c c00 -- /bin/bash result(root@testpod:/#)

6 apt update && install curl ( for communition to container install curl)

OR

7 apt install curl -y

8 curl localhost:80 (checking local host replying or not)

**Result** → <html><body><h1>It works!</h1></body></html>

Exit & delete tespod pod1.yml

kubectl delete -f pod1.yml result (pod "testpod" deleted)

**\*\*\*\*\* \*\*\*\*\*\*\*\*\*\* pod to pod communication on Same Node \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Create 2 different pod**

1 vi pod2.yml

2 vi pod3.yml

3 kubectl apply -f pod2.yml result (pod/testpod1 created) cont name c01

4 kubectl apply -f pod3.yml result (pod/testpod4 created) cont name c03

5 kubectl get pods testpod1 1/1 Running

testpod4 1/1 Running

6 kubectl get pods -o wide for POD IP

testpod1 1/1 Running 0 2m38s **172.17.0.3**

testpod4 1/1 Running 0 39s **172.17.0.4**

**NOTE : *after command ‘’ 6 ‘’ you can enter in the container like c001 install curl and use curl command with other POD IP address. You will get it's work. For going to inside the continer,must know to pod name & container no***

7 cat pod2.yml  result(podname: testpod1) (container - name: c01)

8 cat pod3.yml result(podname: testpod4) (container - name: c03)

9 kubectl exec testpod1 -it -c c01 -- /bin/bash result(root@testpod1:/#)

Now we are root@testpod1:/# here ,,,, instal curl

10 apt install curl

11 curl < **172.17.0.3** ip of pod1>:80 LIKE → curl 172.17.0.8:80

12 curl < **172.17.0.4** ip of of pod4>:80

exit

result <html><body><h1>It works!</h1></body></html>

**\*\*\*\*1 NODDE'S POD TO 2 NODE'S POD COMMUNICATION THROUGH CLUSTER ip \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  *service applied to RS* give virtual IP within a cluster**

1 vi deployhttpd.yml **%%%%%**

2 kubectl apply -f deployhttpd.yml

3 kubectl get pods result (mydeployments-84c5754d58-wbwwd 1/1 Running)

4 kubectl get pods -o wide NAME READY STATUS AGE POD.IP NODE

172.17.0.5 same

5 cat deployhttpd.yml result(podname: testpod1 ) (container - name: c00 )

6 kubectl exec testpod1 -it -- /bin/bash result ( root@testpod1:/# )

**NOTE : this command for going to PODS not to container**

Note:Already install curl

7 curl 172.17.0.5:80 result <html><body><h1>It works!</h1></body></html>

exit

**NOTE**:: but some time pod delte and IP change. Now create **service ( cluster ip )**

8 vi service.yml

9 kubectl apply -f service.yml

10 kubectl get svc NAME TYPE CLSTR-IP EXTRNL-IP PORT(S)

demoservice ClusterIP **10.97.138.235 (v-IP)(static-IP)** <none> 80/TCP

kubernetes ClusterIP 10.96.0.1 <none> 443/TCP

10 kubectl get pods

11 kubectl exec testpod1 -it -- /bin/bash cat deployhttpd.yml (podname: testpod1 result ( root@testpod1:/# )

**NOTE : this command for going to PODS not to container**

12 curl 10.97.138.235:80 resut <html><body><h1>It works!</h1></body></html> \*\*\*

exit

13 kubectl get pods (check pod name, when pod delete name chnagee)

Name of pod = mydeployments-787cfcbc8d-64bsc \*\*

14 kubectl delete pod mydeployments-787cfcbc8d-64bsc ( pod delete)

**Note :: But due to Replica Set create new one**

15 kubectl get pods (new name of pod= mydeployments-84c5754d58-4x4w9 \*\* namechnge but static IP remain same ) check IP same ,,,,,

16 kubectl exec testpod1 -it -- /bin/bash result( root@testpod1:/#)

17 curl 10.97.138.235:80 resut <html><body><h1>It works!</h1></body></html> \*\*\*

Exit

18 kubectl delete pods testpod4

19 kubectl delete pods testpod1

20 kubectl delete -f deployhttpd.yml

.yml file are delete kubectl delete -f service.yml

**\*\*\*\*\*\*\*NODEPORT (service acceble from outside the cluster.like internet)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

1 vi deployhttpd.yml **%%%%% <----- textfile**

2 kubectl apply -f deployhttpd.yml ---🡪 deployment.apps/mydeployments created

3 vi svc.yml (create servive)

4 kubectl apply -f svc.yml

5 kubectl get svc NAME TYPE CLUTR-IP EXTRNL-IP PORT

demoservice **NodePort**  10.110.179.182 <none> **80:32282/ TCP**

kubernetes ClusterIP 10.96.0.1 <none> 443/TCP

6 kubectl describe svc demoservice Name: demoservice

Annotations: <none>

Selector: name=deployment

Type: NodePort

IP: 10.110.179.182

IPs: 10.110.179.182

Port: <unset> 80/TCP

TargetPort: 80/TCP

NodePort: <unset> 32282/TCP

Now go to AWS and copy Public-DNS and paste to google with port no like this

ec2-3-110-210-98.ap-south-1.compute.amazonaws.com: 32282

**not response to google**

7 kubectl delete -f svc.yml (for removing service )

**\*\*\*\*\*\*\*\*\*\*\*\*\* VOLUMES \*\*\*\*\*\*\*\*\*\*\*\*\*with EmptyDir**

1 vi emptydir.yml

2 kubectl apply -f emptydir.yml ---🡪 pod/myvolemptydir created

3 kubectl get pods result (myvolemptydir 2/2 Running)

4 cat emptydir.yml

***For going inside the 1 cpnatiner***

5 kubectl exec myvolemptydir -it -c c1 -- /bin/bash ( name: myvolemptydir # --- pod name )

Now ur inside the pod’s container [root@myvolemptydir /]#

For mount go to /tmp in yml------ (**mountPath: "/tmp/xchange")**

6 cd tmp

7 ls

8 cd xchange ( VOLUME = xchange ) root@myvolemptydir xchange]#

Create file throuhh vi

9 vi abc.yml

10 ls (abc.yml ) & pwd & exit

NOW For going inside the 2 cpnatiner

11 kubectl exec myvolemptydir -it -c c2 -- /bin/bash

12 cd tmp mountPath: "/tmp/data"

13 ls

14 cd data [root@myvolemptydir data]#

15 ls **( abc.yml) 🡨------ hm ne ye file c01 bnayi thi but volume share ki waja se c02 me bhi bn gyi**

Note I am in container 2 and when doing ls showing same abc,yml like container 1,

cd /tmp/ cd data/ ls

Now change something in abc.yml file

16 vi abc.yml ( same contet like container 1 file)

exit

**Now** we are changes in container2 file , go to container 1 file and do cross verify

17 kubectl exec myvolemptydir -it -c c1 -- /bin/bash [root@myvolemptydir /]#

18 cat /tmp/xchange/abc.yml **( content change)**

19 kubectl delete -f emptydir.yml

Now we are in container 1 and showing changes content perfectly…

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* HOSTPATH \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**1 POD wants to access 2nd pods data and mapped with HOST machine,,..**

1 vi hostpath.yml

2 kubectl apply -f hostpath.yml

3 kubectl get pods

4 vi hostpath.yml name of pod: myvolhostpath

5 ls /tmp

After this data directory not showing

6

7

8

9

10

Follow technical guftugu LEC-52

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* PERSINSTANT VOLUMES \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

First configured minikube, when minikube configured go to AWS make volume EBS → vlomes → create volumes with same az , when volume available copy volume ID and paste to yml script infront of volume ID vol- vol-08ebb4393908cd4a4

1 vi mypv.yml

2 kubectl apply -f mypv.yml --🡪result (persistentvolume/myebsvol created)

3 kubectl get pv NAME CAPACITY ACCESS MODES RECLAIM POLICY

myebsvol 1Gi RWO Recycle

4 vi mypvc.yml **( after making PV. Now claiming for PV) how many gb you want**

5 kubectl apply -f mypvc.yml result(persistentvolumeclaim/myebsvolclaim created)

6 kubectl get pvc NAME STATUS VOLUME CAPCTY

(name: myebsvolclaim # -- name of "PVC" ) myebsvolclaim  **Bound** pvc-e9-1b3b 1Gi

7 vi deploypvc.**yml ( for using PVC Now creat pod)**

8 kubectl apply -f deploypvc.yml (deployment.apps/pvdeploy created)

9 kubectl get deploy result(pvdeploy 1/1 1 1)

10 kubectl get rs ( desired vs output)

11 kubectl get pods ( name of pod )

12 kubectl exec pvdeploy-f74845446-4jwgr -it -- /bin/bash

**Now** you are in [root@pvdeploy-7dc98b5bcd-jr2p9 /]# container

For path open .yml in cat befor command no “”10””

13 cd /tmp/persistent/ result([root@b5bcd-jr2p9 persistent]#

14 ls

**Now** make some file through vi

15 vi testfile.yml

16 ls result(testfile.yml)

exit and delete pod

17 kubectl delete pod pvdeploy-7dc98b5bcd-jr2p9

**Now** due to Deployment object create new pod

18 kubectl get pods result( pvdeploy-7dc98b5bcd-cs9ln) 🡨--- **new pod**

19 kubectl exec pvdeploy-f74845446-zqjbx -it -- /bin/bash

Now you are in [root@pvdeploy-7dc98b5bcd-cs9ln /]# container

20 cd /tmp/persistent/ [root@pvdeploy-7dc98b5bcd-cs9ln persistent]#

21 ls result(testfile.yml

22 cat testfile.yml **(show content)**

Exit and **delete the volume in AWS**

kubectl delete -f deploypvc.yml

kubectl delete -f mypvc.yml

kubectl delete -f mypv.yml

Delete file

**\*\*\*\*\*\*\*\*\*\*\*\*\*HEALTHCHECK/LIVENESSPROBE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

1 vi liveness.yml

2 kubectl apply -f liveness.yml

3 kubectl get pods result (mylivenessprobe 1/1 Running)

4 kubectl describe pod mylivenessprobe (all liveness details)

RESULT --- Liveness: exec [cat /tmp/healthy] delay=5s timeout=30s period=5s #success=1 #failure=3

5 kubectl get pods

6 kubectl exec mylivenessprobe -it -- /bin/bash result( root@mylivenessprobe:/#)

7 cat /tmp/healthy

8 **echo $?** **→ showing ‘0’ means status healthy**

**→ showing ‘1’ means status not healthy**

**Now we trying with error ( wrong file search)**

9 cat /tmp/aamir **result(cat: /tmp/aamir: No such file or directory) k8s consider error**

10 echo $? → showing ‘1’ means status unhealthy

11 ls /tmp/healthy

Now delete this /tmp/healthy

12 rm /tmp/healthy (after delete and after 5 sec k8s check /tmp/healthy & cant find )

13 cat /tmp/healthy

14 echo $? After removing showing “”1””

15 kubectl describe pod mylivenessprobe (all liveness details)

16 kubectl delete -f liveness.yml

Follow technical guftugu LEC-53

**\*\*\*\*\*\*\*\*\* CONFIGMAP (create VIRTUAL memory) (readable) & SECRTES (hidden)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

1. **%%%%As a volume add**

1 vi sample.conf (configuration file)

2 kubectl create configmap mymap --from-file=sample.conf **(create object) for file**

Type=configmap GIVE name =mymap

2 kubectl create cm cm5 --from-file=data-base --from-file=application\_file

2 kubectl create configmap cm1 --from-literal=database\_ip="192.30.45" --from-literal=database\_name="root"

3 kubectl get configmap/cm NAME DATA AGE

kube-root-ca.crt 1 5h4m

mymap 1 4m4s

4 kubectl describe configmap mymap **(shwing .conf file content)**

**Now we want**, when cluster create pod or container this configuration is accessible..

5 vi deployconfig.yml

6 kubectl apply -f deployconfig.yml (creating pod)

7 kubectl get pods ( myvolconfig 1/1 Running )

8 kubectl exec myvolconfig -it -- /bin/bash (root@myvolconfig /]#)

9 cd /tmp

10 ls result( **config** ks-script-o23i7rc2)

11 ls -l (**d**rwxrwxrwx 3 root root 4096 Sep 22 11:46 config)

12 cd config

13 ls result( sample.conf )

14 cat sample.conf

15 kubectl delete cm --a

15 kubectl delete -f deployconfig.yml

1. **%%%%As a environment variable**

1 vi deployenv.yml

2 kubectl apply -f deployenv.yml

3 cat deployenv.yml

4 kubectl get pods (myenvconfig 1/1 Running)

5 kubectl exec myenvconfig -it -- /bin/bash ( [root@myenvconfig /]# )

6 env result( MYENV=this is my configuration file for any application)

**(env read the content and accces the content)**

7 echo $MYENV content (this is my configuration file for any application)

Exit and delete kubectl delete -f deployenv.yml

Note :: configmap for directory… create directory name configma00 and exit

kubectl create configmap mymap --from-file=configmap00/ ← foldername

Note :: configmap inject into pod

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*SECRET ( for all secret file to hide)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Note ::** Create file for **username or password**:

1 echo "root" > **username.txt**; echo "mypassword123" > **password.txt**

filename: username.txt password.txt

**↓ ↓**

root my password123

for checking---

1cat **username.txt** , cat **password.txt**

**Now hide file**

2 kubectl create secret generic mysecret --from-file=username.txt --from-file=password.txt

(given name)

Result secret/mysecret created

3 kubectl get secret NAME TYPE DATA AGE

mysecret Opaque 2 2m26s

4 kubectl describe secret mysecret Data

password.txt: 14 bytes

username.txt: 5 bytes

**Note :: don’t give content inside these file**

**for pod or container—**

1 vi deploysecret.yml script

1. kubectl apply -f deploysecret.yml
2. kubectl get pods

2 kubectl delete -f deploysecret.yml ( for deleteing )

3 ubuntu# kubectl get pods

4 kubectl exec myvolsecret -it -- /bin/bash

5 ls

6 cd /tmp

7 ls result( mysecrets)

8 cd mysecrets/ [root@myvolsecret mysecrets]#

9 ls resullt ( password.txt username.txt)

10 cat password.txt RESULT (mypassword123)

11 cat username.txt RESULT (root)

Follow technical guftugu LEC-54

**\*\*\*\*\*\*\*\*\* NAMESPACES ( isolation of PODS ) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**NOTE:: RESOURCE QUOTA is always set to the Namespace**

**Limt range is only applicabe for {PODS ,, CONTAINERS ,, STRORAGE}**

First check Default Namespace for PODS…..

Kubectl run web-server --image=nginx (imerative method)

kubectl run web-server --image=nginx -n project-1 (create pod in PROJECT-1 Namespace)

kubectl get pods -n project-1 -o wide

Kubectl delete pod web-server -n PROJECT- 1

kubectl describe ns project-1 ( for checking RESOURCE QUOTAS)

kubectl get quota -n project-1

kubectl describe quota project-1-quota -n project-1 (display used and unsed quota for project-1 )

kubectl delete pods --all -n project-1 (delete all pods umder project-1 namespace )

kubectl get pods -n project-1

kubectl describe pod web-server -n project-1

Kubectl edit namespace project-1 (describe project-1 quota)

kubectl delete pods --all -n project-1 **( delete all rununig pods in project-1)**

1 kubectl get pods (No resources found in default namespace.)

2 kubectl get namespace default Active 11m

kube-node-lease Active 11m

kube-public Active 11m

kube-system Active 11m

**Note** ::**now we create own namespace ….**

1 kubectl create ns testns ( imperative method , testns = ns name)

3 vi devns.yml ( declaerative method of creating Namespace )

4 kubectl apply -f devns.yml -v=7 ( namespace/dev created ( -v=7 🡪 verbose mod )

5 kubectl get namespace ( namespace created )

5- kubectl get all -n kube-public ( we don’t have anything )

: - for testing make pod

6 vi pod1.yml

7 kubectl apply -f pod1.yml -n dev ( pod/testpod created in your namespace) dev=namespace

7 kubectl apply -f pod1.yml --namespace dev

8 kubectl get pods -n dev (name spce pod) NAME READY STATUS RESTARTS AGE

testpod 1/1 Running 0 2m5s

**Note : if u deleteing the pod 1st mention the namespace**

10 kubectl delete -f pod1.yml -n dev

**Now aply again** kubectl apply -f pod1.yml -n dev

**Note : If you want k8s search pod in your name space directly ( jaha hm kaam kr rahewaha dekhe)**

kubectl config get-contexts project-1 added (check namespce admin)

kubectl config delete-contexts project-1 project-1 delete

11 **kubectl config set-context $(kubectl config current-context) --namespace=dev**

(Context "minikube" modified. )

**11 kubectl config set-context $(kubectl config current-context) --namespace=kubernetes-admin@kubernetes**

**(for set the default )**

**Note :: Now u fired simple command ” kubectl get pods” s0 k8s treat as a default namespcace**

12 kubectl get pods NAME READY STATUS RESTARTS AGE

testpod 1/1 Running 0 20s

**Note ::now u want to check which namespace u seeing now (jis namespace me dekh rha hai wo bteya)**

13 kubectl config view | grep namespace: (namespace: dev)

14 kubectl config view name: context\_info

namespace: dev

Note :: If you want k8s search pod in Default namespace

15 kubectl get pods -n default

16 kubectl delete -f pod1.yml -n dev

Note :: create namespace throuuhh cmnd line

16 kubectl create ns my-personal

17 kubectl config set-context --current --namespace=my-personal ( my-personal moved to defalt name space)

18 kubectl get pods --all -namespaces ( all name space) \\ kubectl get all --all -namespaces

**\*\*\*\*\*\*\*\* RESOURCES \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

1 vi poderesources.yml

2 kubectl apply -f poderesources.yml ( pod/resources created)

**No imp** : check which namespace currently you working

3 kubectl config view | grep namespace: (check which Nmaespace are uworking )

4 kubectl get pods ( resources 1/1 Running)

5 kubectl describe pods resources ( resource= pod name)

6 kubectl delete -f poderesources.yml

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* RESOURCE QUOTA \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**NOTE:: RESOURCE QUOTA is always set to the Namespace**

**Limt range is only applicabe for {PODS ,, CONTAINERS ,, STRORAGE}**

kubectl get quota -n project-1

kubectl delete -f project-rq.yml

kubectl describe quota project-1-quota -n project-1 (display used and unsed quota for project-1 )

kubectl get quota (No resources found in default namespace. )

1 vi resourcequota.yml

2 kubectl describe ns myns ( describe quota in myns ns)

2 kubectl apply -f resourcequota.yml

3 vi testpod.yml ( here we obly aply reqest not limit)

3 kubectl apply -f testpod.yml

4 kubectl get deploy (deployments 0/3) **not ready**

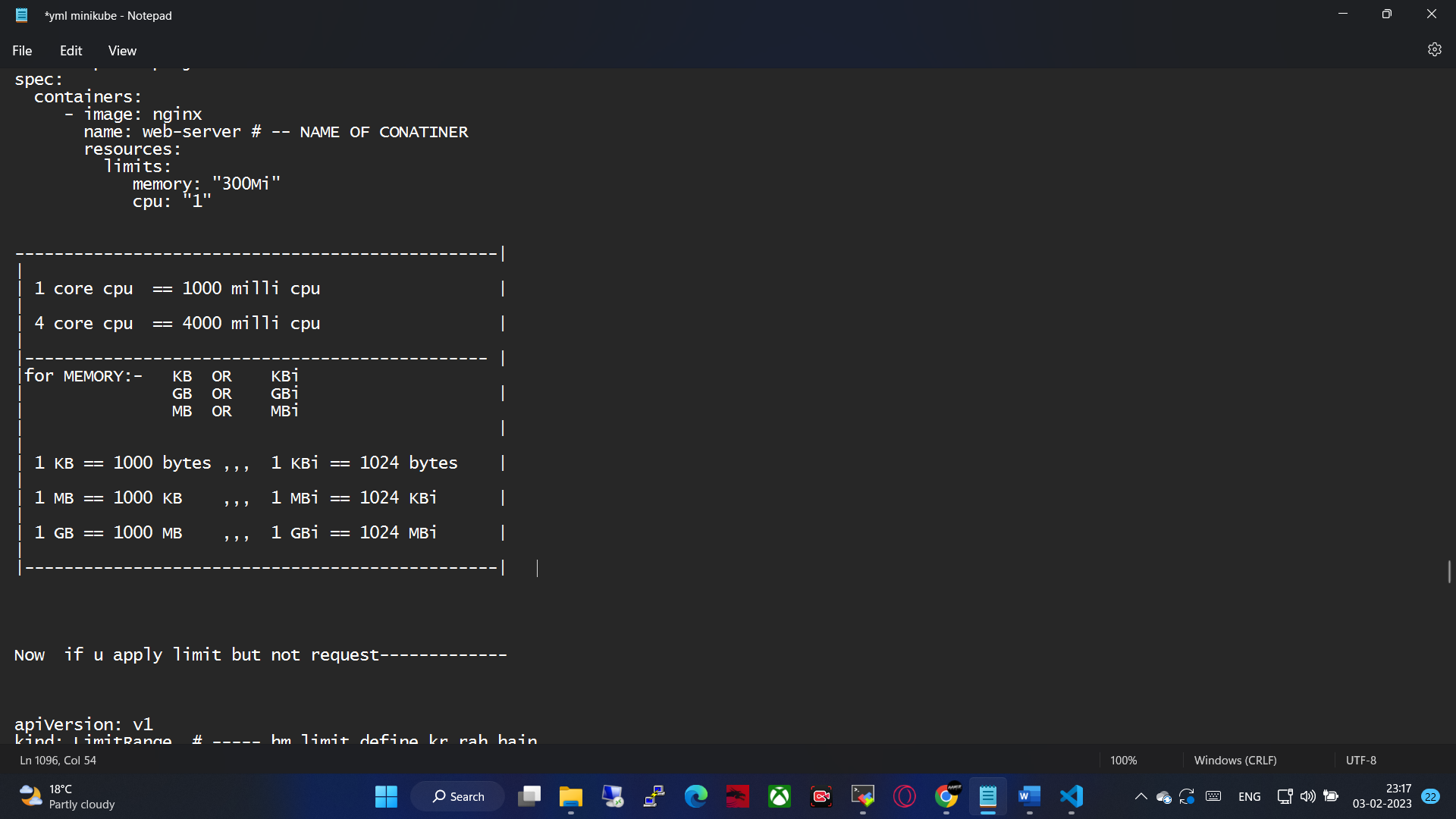
5 kubectl get pods o resources found in dev namespace

Once see yml file plzz spec:

replicas: 3

6 kubectl get rs NAME DESIRED CURRENT READY AGE@@

deployments-8579bf8f96 3 0 0 11m

go and watch vi resourcequota.yml

hard:

limits.cpu: "400m" (max quota)

limits.memory: "400Mi"

requests.cpu: "200m"

requests.memory: "200Mi"

now check this once vi testpod.yml

replicas: 3

cpu: "200m" (per head)

over to limit thats reason not achived @@

***for limitrange ---------------------rajesh***

***type :: container***

vi limit-cpu-container.yml

kubectl apply -f limit-cpu-container.yml

kubectl get limitrange (for limit range name )

kubectl describe limitrange limit-mem-cpu-per-container

check….

7 kubectl describe rs deployments-8579bf8f96 (Warning FailedCreate)

8 kubectl describe ns myns Name: myns

Labels: kubernetes.io/metad

9 kubectl delete -f resourcequota.yml

10 kubectl delete -f testpod.yml

**If we apply LIMIT but not REQUEST------------------**

1 vi cpu.yml

2 kubectl apply -f cpu.yml

2 kubectl get pods testpod =name

3 kubectl describe pod <pod name> **Limits: Requests:**

**cpu: 1 cpu:1**

4 kubectl delete -f cpu1.yml

**If we apply REQEUST but not LIMIT-----------------:WQ**

4 vi cpu2.yml

5 kubectl apply -f cpu2.yml

6 kubectl describe pod default-cpu-demo-3 Requests: limit

cpu: **750m 1**

**7** kubectl delete -f cpu2.yml

**For memory--------------------------**

1 vi memdafaut.yml

2 kubectl apply -f memdafaut.yml

3 vi meme1.yml

4 kubectl apply -f mem1.yml

5 kubectl get pods

7 kubectl describe pod constraints-mem-demo ( Limits:

Limt < 1000 memory: 800Mi

Request >500 Requests:

memory: 600Mi

check if limit request lopw or high

8 kubectl delete -f mem1.yml

9 vi mem2.yml (content+ meme1.yml ) (limit exceeding)

10 kubectl apply -f mem2.yml

**((maximum memory usage per Cntainer is 1Gi, but limit is 1800Mi)**

11 vi mem3.yml (content+ meme1.yml ) (request reducing)

12 kubectl apply -f mem3.yml

**(minimum memory usage per Container is 500Mi, but request is 200Mi)**

***for limitrange ---------------------rajesh yml.txt***

***type :: pods***

vi limit-mem-cpu-per-pods1.yml

kubectl apply -f limit-mem-cpu-per-pods1.yml

kubectl get limitrange

kubectl delete limitrange --all

kubectl describe limitrange

kubectl describe limitrange limit-mem-cpu-per-pods

kubectl apply -f limit-mem-cpu-per-pods1.yml

kubectl get pods

kubectl describe pods busybox2

kubectl get limitrange

kubectl create quota myquota -o yaml --dry-run=client > myquota.yaml ( output le rahe hain )

ls → myquota.yaml

vi myquota.yaml ( yml minikube ) last

kubectl create ns mysession ( create ns bcz in yml mention ns )

kubectl get quota -n mysession NAME AGE REQUEST LIMIT

myquota 7m11s cpu: 0/1, memory: 0/2G

Follow technical guftugu LEC-55 ( video start 24:24 )

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* AUTO SCALING \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Download **matric server**

wget https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml -O metrics-server-components.yaml

after download check “” **metrics-server-components.yaml””** this file

1 ls **metrics-server-components.yaml**

Go to inside this

2 vi **lmetrics-server-components.yaml**

Add certificate in DEPLOYMENTS → container → CIRTIFICATION in line no3

- --cert-dir=/tmp

- --secure-port=4443

**--kubelet-insecure-tls** (certificate)

- --kubelet-preferred-ad

3 kubectl apply -f **lmetrics-server-components.yaml**

4 kubectl get pods

5 kubectl get namespaces (kube-system)

6 kubectl get pods -n kube-system (**lmetrics-server-components) serve runing**

7 kubectl logs -f <matric server name> -n kube-system (showing info)

8 vi deployhpa.yml

9 kubectl apply -f deployhpa.yml

10 kubectl get all

**Node :: Paste hpa commad here**

11 kubectl autoscale deployment mydeploy --cpu-percent=20 --min=1 --max=10

(horizontalpodautoscaler.autoscaling/mydeploy autoscaled)

12 kubectl get all ( horizontalpodautoscaler.autoscaling/mydeploy ) running

**Open new terminal with new colour**

13 kubectl get pods

14 kubectl exec <pod name> -it -- /bin/bash

Go to previous terminal and type

15 watch kubectl get all

15 kubectl get all

16 open coloured terminal and

Updattttte u see load increase

################# Mithun technology - Kubernetes - HPA #################

kubectl get hpa ( showing HPA )

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* T A I N T IN K8S \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

1 kubectl run web-server --image=nginx ( pod/web-server created )

2 kubectl port-forward web-server 8081:80 --address 0.0.0.0 ( port forward )

Kubectl describe nodes | grep “Taints”

**Now k8s will not create node on master node now chgeck**

1. kubectl get node (take master node IP)

4 kubectl describe node ip-172-31-35-164(master) | less **Taints: node-role.kubernetes.io/master:NoSchedule**

**Now we want to creat pod on to the master node also so untaint ip-172-31-35-164 = mastr node**

5. kubectl taint nodes ip-172-31-35-164 node-role.kubernetes.io/**master**- (node/ip-172-31-35-164 **untainted** )

6. kubectl describe node ip-172-31-35-164 | grep -i taint **Taints: <none>**

**Now go to and check pod create in mastwr also**

7. kubectl get pods -o wide

**Now u wnt to tained to master node no pod will creat on master**

8. kubectl taint nodes ip-172-31-35-164 node-role.kubernetes.io/master:**NoSchedule**  **node/ip--164 tainted**

**HERE we want to tainted worker node 2 ( ip-172-31-43-101) ( no node create/schedule on node-2 )**

9. kubectl taint nodes ip-172-31-43-101 key1=value:**NoSchedule** ( node/ip-172-31-43-101 tainted )

10. kubectl describe node ip-172-31-43-101 | grep -i taint ( Taints: key1=value:NoSchedule )

**HERE we want to untainted/ or romve taint to worker node 2 ( ip-172-31-43-101)**

11. kubectl taint nodes ip-172-31-43-101 key1=value:**NoSchedule-** ( node/ip-172-31-43-101 untainted )

**kubectl taint nodes ip-172-31-44-220 key:NoExecute ----------🡪 *node/ip-172-44-220 tainted***

**kubectl taint nodes ip-172-31-44-220 key:NoExecute- ---------🡪**  ***node/ip-172-31-44-220 untainted***

Follow technical guftugu LEC-56

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* JOBS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

1 vi job.yml

2 kubectl apply -f job.yml (testjob-tlbp9 0/1 Completed)

3 watch kubectl get pods

4 kubectl delete -f job.yml

5 kubectl apply -f job.yml (this is job wordone and delete that’s why showing o/1)

6 kubectl delete -f job.yml

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* parallisam in job \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Creat 5 6 or more container, after work done delete automatticsly

1 vi job2.yml

2 kubectl apply -f job2.yml

3 kubectl get pods (pods showing)

4 watch kubectl get pods (after work complete terminate itself)

**\*\*\*\*\*\*\*\*\*\*\*\* coron job pattern ( scheduling g) or (periodically)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**1** vi coronjob.yml

2 kubectl apply -f coronjob.yml

3 watch kubectl get pods

4

5

6

7

8

9

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* I N I T CONTAINER \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1 vi initcont.yml

2 kubectl apply -f initcont.yml

3 watch kubectl get pods NAME READY STATUS RESTARTS AGE

myapp-pod 0/1 Init:0/2 0 22s

4 kubectl logs -f <pod/initcontainer>

5 kubectl get all -o wide

6

7

8

9

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* H E L M \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

To download helm

1 curl -fsSL -o get\_helm.sh https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3

2 chmod 700 get\_helm.sh

3 ./get\_helm.sh (helm installed into /usr/local/bin/helm)

4 which helm (/usr/local/bin/helm)

5 helm version

6 helm list

Add repo to helm

7 helm repo add stable https://charts.helm.sh/stable --force-update

("stable" has been added to your repositories)

8 helm repo ( list chart repositories)

9 helm repo list (stable <https://charts.helm.sh/stable>)

If u want to remove Repo

10 helm repo remove stable ("stable" has been removed from your repositories)

11 helm search repo jenkins

12 helm show values stable/jenkin

13 helm show chart stable/tomcat

14 helm show all stable/tomcat

15 apt-get install tree

16 helm create helloworld ( for creat new helm chart)

17 ls ( get\_helm.sh helloworld )

18 tree helloworld/

19 ls ( Chart.yaml charts templates values.yaml)

20 vi values.yaml) manifest

21 helm install jestjenkins stable/jenkins

22

23

24

25

26

27

28

29