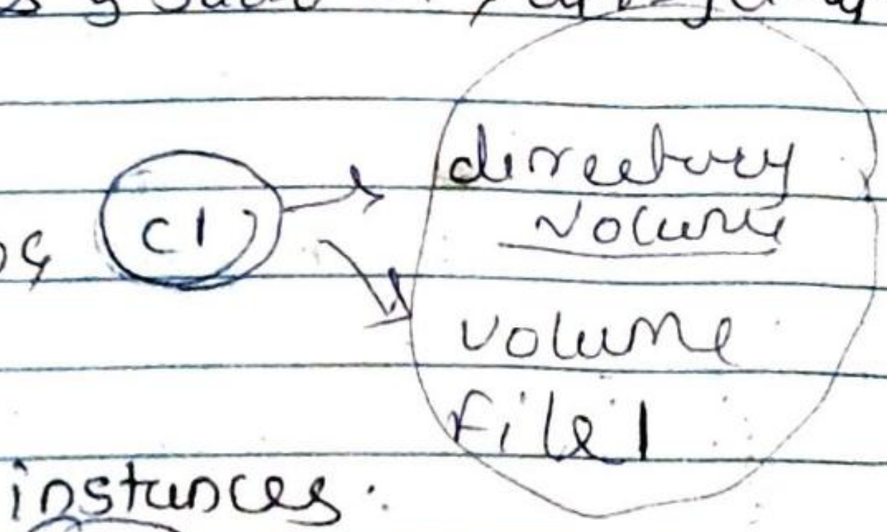
(1)

launch instance.

1) - K8S
2) ubontu
3) 18.04
4) t2. medium [2cpu, 4Gbram]
5) key
6) edit All traffic anywhere.
7) 3 number of instances
8) launch instances.

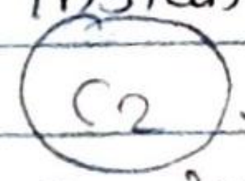
9) Connect all 3 instances & sudo -i, apt-get-update

10) go to google install docker on ubuntu 18.04



1st link.

1) old version Run on 3 instances.
2) sudo apt update. 3
3) sudo apt install. 3 instances
4) key Commands exe 3 instances
5) Run next
6) apt get update.
7) install.
8) execute just Command [deman service].
9) sudo systemctl enable docker
10) demon Service Reload
11) demon Service Reload
12) restart Command on file.



go to google serch install kubeadm.
1st link.

BMIT

1) sudo apt-get update
2) certificate curl
3) key Command
4) Repositery "echo"
5) Run 3 Commands {3 services}
   [Kubeate set on hold] → shows.

6) make mastees
   edit instance name.
   master - K8S
   node1 - K8S
   node2 - K8S
   * on master root
   ~~kubeadm limit~~
   rm /etc/containerd/config.toml
   systemctl restart Containerd.
   kubedm ~~limit~~ init.
   new window open.
   * copy control pannel.

   Paste it ~~on drive~~ ~~folder~~ document.
   on node rm /etc/containerd/config.toml (txt file)
   * systemctl restart containerd.

   Run 3 Commands on master,
1) mkdir 2) sudo cp -i 3) sudo chown.

   Run node
   kubedm join 172.31.8.175:6443

   on master
1) cube-ctl get nodes

   Run one command on file.
   they are ready to in nodes.

21-11-2022

B.M.I.T

(2).

sudo -i

cub kubectl get nodes (To check nodes it shows (Ready)

vi mypods.yml

api Version : v1

kind : Pod. → capital

Metadata : ← small

name : hello-pod

spec:

Containers:

- name : first-container ← small

image : nginx

Ports: ← small

- containerPort : 80

exit :wq !

* kubctl create -f mypods.yml (creating pods)
* kubctl get pods -o wide (To check the pods on which node it is) (It will installed 2ed node) we did not change & the pods creating rights is totally in hand with kubernetess
* kubctl describe pods (details of pods) (vs code)
* Create folder . (K8s-Peiya)
* go to vs → open folder → desktop → k8s → go to extenstion → kubernetes → install
* create new file . (mypods.yml)
* go to github → new repository → K8S-Tpm → create repository enter code in terminal.

on root.

* kubctl delete pod/hell-pod (podname) → To delete the pod on root.

git clone (code)
cd k8s_7pm/
ls
kubctl create -f mypods.yml
(3)rd. need (To maintain desired ~~state~~ -22
kubctl delete pod /(podname) → delete pod
kubectl describe pod → full details of pods

we need to multiple pods & we need desired state
desired state :- it will create same pods in another server ~~whi~~ when the pods are deleted or nodes are deleted.

Repllication Controller :-
in VS
New file → myRC.yml
apiVersion : v1
kind : ReplicationController → Process
metadata :
name : myonlin vertex-rc → name.
spec :
replicas: 3 → no. of podes
selector :
app : myonlineapp (myvertex pod)
version : 2.6.2 → The pod which will be used for repllication.
template :
metadata :
labels :
app : myvertexpod.
version : 2.6.2

Repllication Controller.

Pod part
~~resp~~ after rep
repllication pod

containers:
- name : myngnix
  image : nginx
  ports :
  - containerport : 80

} inside Pod container name & image

NOTE :- pod name & selector name should be same.

git add -A
git commit -m "mydata"
git push.
git pull on - server.
kubectl create -f myrc.yml
kubectl get rc.
kubectl get pods
delete one pod.
kubectl delete pod/veetex-rc-bxvc99
it will create new pod within time.

kubectl get pods -o wide → on which node will be Running
kubectl get rc
kubectl delete rc/veetex-rc
(Rc-name)
kubectl get pods → shows 0 Pods.

* (6)th.
sudo -i
get RC.
del kubectl delete rc/myonlineapp-rc

if we losts ip of pods so we can achive that using names