\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

STEP: 1 PREPARE AWS UBUNTU INSTANCE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. prepare Ubuntu Server 22.04 LTS (HVM), SSD Volume Type docker instance and add http and custom tcp protocol for port 8080, download the .pem key to the desire directory and connect the aws instance using moba-xterm

2. goto>aws>instance>connect>ssh>copy the link

3. open moba x-term> move to the root directory where you have saved the .pem file

4. paste the copied link and hit enter

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

STEP: 2 DOCKER INSTALLATION ON UBUNTU OS

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

link: https://docs.docker.com/engine/install/ubuntu/

> sudo apt-get update

> sudo apt-get install ca-certificates curl gnupg lsb-release

> curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg

> echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

> sudo apt-get update

> sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin

----------to verify docker installation

> sudo docker -v

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

STEP: 3 INSTALLATION OF KUBERNETES

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

link: https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/

>sudo apt-get update

>sudo apt-get install -y apt-transport-https ca-certificates curl

>sudo curl -fsSLo /usr/share/keyrings/kubernetes-archive-keyring.gpg https://packages.cloud.google.com/apt/doc/apt-key.gpg

> echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/kubernetes.list

>sudo apt-get update

>sudo apt-get install -y kubelet=1.22.8-00 kubectl=1.22.8-00 kubeadm=1.22.8-00

> sudo apt-mark hold kubelet kubeadm kubectl

--to check the version-------

> kubelet --version

> kubeadm version

--- to check cube ctl version

> kubectl version

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

STEP: 1 DISABLING SWAPP

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

> sudo swapoff -a

> sudo sed -i '/ swap / s/^/#/' /etc/fstab

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

STEP: 2 ADDING C GROUP

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

EDIT DOCKER DAEMON FILE

> sudo vi /etc/docker/daemon.json

press 'i' and insert the below mentioned code

{

"exec-opts":["native.cgroupdriver=systemd"]

}

press esc;

type :wq!

hit enter

once the file updated we need to restart the services

>sudo systemctl daemon-reload && sudo systemctl restart docker && sudo systemctl restart kubelet

>sudo docker info |grep -i cgroup

check the cgroup and version if it is coming you are all set to go ahead

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Let's initialize cluster again

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

> sudo kubeadm init --ignore-preflight-errors=all

you will get message your kubernates initiated successfully

\*\*\*\*\*\*\*\*\*if not works

> sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=all

\*\*\*\*you will get some command if you want to start cluster as a regular user

mkdir -p $HOME/.kube

sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

sudo chown $(id -u):$(id -g) $HOME/.kube/config

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

get a node details by executing following command

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

> kubectl get node

if you want to see the cluster details

>kubectl get svc

Once the cluster is started if you want to deploy your any code you can deploy on it.

let's deploy weave network to cluster

>export kubever=$(kubectl version | base64 | tr -d '\n')

>kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-version=$kubever"

Validate the running pods

>kubectl get pods --all-namespaces

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Host a springboot application in kubernetes Pods

----------------------------------------------------------

Step: 1

>git clone https://github.com/Nikunj-Java/SpringBootDockerApp.git

-----------------------------------

Step: 2 let's install Maven

-----------------------------------

>sudo apt-get update && sudo apt install maven -y

-----------------------------------

Step: 3 let's install JDK

-----------------------------------

sudo apt-get update && sudo apt install default-jdk -y

-----------------------------------

> cd SpringBootDocker

>mvn clean install (create jar file)

goto> aws> stop the instance and restart it and connect again using moba x-term

> git clone https://github.com/Nikunj-Java/SpringBootDockerApp.git

> cd SpringBootDocker

> mvn clean install (create jar file)

\*\*\* build image \*\*\*\*\*\*\*\*\*\*

>sudo docker build -t springboot .

>sudo docker images

>kubectl run springbootapp --image=springboot --port=8080

you will get msg that : pod/springbootapp created

>kubectl get pods

\*\*\*\*\*\*\*\*\*\*\* Create Deployment Service \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

kubectl expose pod/springbootapp --port=8081 --target-port=8081 --type=LoadBalancer

you will get msg that : service/springbootapp exposed

kubectl get services

kubectl describe svc springbootapp