To expose the pods by using deployment and services

Steps to create

Step 1:

1. Create the server with ubuntu
    - Select the ubuntu
    - Select the t2 micro
    - Increase the storage to 25
2. Change to the root user
3. Update the server apt update -y

Step 2:

1. Install the docker packages by using commands

sudo apt install curl wget apt-transport-https -y
sudo curl -fsSL https://get.docker.com -o get-docker.sh
chmod 777 get-docker.sh
sh get-docker.sh

2. Start the docker by command systemctl start docker
3. Check the status of docker by command systemctl status docker

Step 3:

1. Install the kubectl packages

sudo curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
chmod +x kubectl

2. It can provide the executable permissions to the kubectl file

Step 4:

1. Give **aws configure**
2. Give snap info **aws-cli**
3. Give snap install **aws-cli --channel=v1/stable --classic** to install the aws cli
4. Create user in iam and give access permissions to the user
5. Provide access key and secret key in cli

Step 5:

1. To install the kops by command is

curl -LO https://github.com/kubernetes/kops/releases/download/v1.25.0/kops-linux-amd64
chmod +x kops-linux-amd64
mv kops-linux-amd64 /usr/local/bin/kops
mv kubectl /usr/local/bin/kubectl

2. After performing all the commands in terminal as shown in below figure

```
aws  Services  Search [Alt+S]
EC2
root@ip-172-31-6-184:~# curl -Lo https://github.com/kubernetes/kops/releases/download/v1.25.0/kops-linux-amd64
curl: (2) no URL specified
curl: try 'curl --help' or 'curl --manual' for more information
root@ip-172-31-6-184:~# curl -LO https://github.com/kubernetes/kops/releases/download/v1.25.0/kops-linux-amd64
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100  155M  100  155M    0     0  51.6M      0  0:00:03  0:00:03 --:--:-- 98.6M
root@ip-172-31-6-184:~# ll
total 374056
drwx------  5 root root      4096 Nov 19 12:19 ./
drwxr-xr-x 22 root root      4096 Nov 19 09:48 ../
drwxr-xr-x  2 root root      4096 Nov 19 12:05 .aws/
-rw-r--r--  1 root root      3106 Apr 22  2024 .bashrc
-rw-------  1 root root        20 Nov 19 11:45 .lesshst
-rw-r--r--  1 root root       161 Apr 22  2024 .profile
drwx------  2 root root      4096 Nov 19 09:48 .ssh/
-rwxrwxrwx  1 root root     22115 Nov 19 11:05 get-docker.sh*
-rwxr-xr-x  1 root root 163291319 Nov 19 12:10 kops*
-rw-r--r--  1 root root 163291319 Nov 19 12:19 kops-linux-amd64
-rwxr-xr-x  1 root root  56381592 Nov 19 11:46 kubectl*
drwx------  4 root root      4096 Nov 19 12:05 snap/
root@ip-172-31-6-184:~# chmod +x kops-linux-amd64
root@ip-172-31-6-184:~# ll
total 374056
```

3. Give ll -a to list the files

4. Go inside of the file vi .bashrc to add the content in file

```
aws  Services  Search [Alt+S]
EC2
    alias grep='grep --color=auto'
    alias fgrep='fgrep --color=auto'
    alias egrep='egrep --color=auto'
fi

# some more ls aliases
alias ll='ls -alF'
alias la='ls -A'
alias l='ls -CF'

# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi
export PATH=$PATH:/usr/local/bin
# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
#if [ -f /etc/bash_completion ] && ! shopt -oq posix; then
#    . /etc/bash_completion
-- INSERT --
```

5. Save and quit from the editor

6. Type **source .bashrc** command

7. Type **kubectl version --client --output=yaml** it will show the version in yaml file format as shown in below figure

```
root@ip-172-31-6-184:~# vi .bashrc
root@ip-172-31-6-184:~# source .bashrc
root@ip-172-31-6-184:~# kubectl version --client --output=yaml
clientVersion:
  buildDate: "2024-10-22T20:35:25Z"
  compiler: gc
  gitCommit: 5864a4677267e6adeae276ad85882a8714d69d9d
  gitTreeState: clean
  gitVersion: v1.31.2
  goVersion: go1.22.8
  major: "1"
  minor: "31"
  platform: linux/amd64
kustomizeVersion: v5.4.2

root@ip-172-31-6-184:~#
```

Step 6:

1. Create the s3 bucket from cli and store the entire cluster information in the created bucket
2. Create the bucket from cli by the command

   aws s3api create-bucket --bucket mani420 --region us-east-1

3. To enable the versioning from cli by through command

   aws s3api put-bucket-versioning --bucket mani420 --region us-east-1 --versioning-configuration Status=Enabled

4. Exporting the bucket in kops by the command

   export kops_state_store=s3://mani420

Step 7:

1. We need to create the cluster
2. We can create the cluster by the command

kops create cluster --name raj.k8s.local --state=s3://mani420 --zones us-east-1a --master-size t2.medium --node-size t2.micro

3. It can create the s3 bucket, autoscaling, load balancer automatically in N.virginia as shown in below figure
4. To edit the node information

kops edit ig --name=raj.k8s.local nodes-us-east-1a --state=s3://mani420

```
Suggestions:
 * list clusters with: kops get cluster
 * edit this cluster with: kops edit cluster raj.k8s.local
 * edit your node instance group: kops edit ig --name=raj.k8s.local nodes-us-east-1a
 * edit your master instance group: kops edit ig --name=raj.k8s.local master-us-east-1a

Finally configure your cluster with: kops update cluster --name raj.k8s.local --yes --admin
```

5. We need to change max and min then save and exit
6. If the instances are not created then update the cluster, then the instances, load balancers, autoscaling will be created by the command

kops update cluster --name raj.k8s.local --yes --admin --state=s3://mani420

7. Create yaml file with vi abhi.yml

```
aws   Services   Search   [Alt+S]
EC2
apiVersion: v1
kind: Pod
metadata:
  name: sam-pod
spec:
  containers:
    - name: cont1
      image: nginx
      ports:
      - containerPort: 80
    - name: cont2
      image: ubuntu
      command: ["sh", "-c", "while true; do echo 'Hello from ubuntu'; sleep 10; done"]
```

8. Type **kubectl create -f abhi.yml** **to create the pods**
9. Type **kubectl describe pods sam-pod** to check on which node pod is created

```
root@ip-172-31-34-141:~# kubectl create -f abhi.yml
pod/sam-pod created
root@ip-172-31-34-141:~# kubectl get pods
NAME      READY   STATUS    RESTARTS   AGE
sam-pod   2/2     Running   0          26s
root@ip-172-31-34-141:~# kubectl describe pod sam-pod
Name:             sam-pod
Namespace:        default
Priority:         0
Service Account:  default
Node:             i-0cd0531d322fbc45d/172.20.46.58
Start Time:       Wed, 20 Nov 2024 07:58:34 +0000
Labels:           <none>
Annotations:      kubernetes.io/limit-ranger: LimitRanger plugin set: cpu request for container cont1; cpu request for container cont2
Status:           Running
IP:               100.96.1.6
IPs:
  IP:  100.96.1.6
Containers:
  cont1:
    Container ID:   containerd://90cc064d634e0ed324270fc00a9cf5d22f45ff0a0f607bff010ba88c5e905cb6
    Image:          nginx
    Image ID:       docker.io/library/nginx@sha256:bc5eac5eafc581aeda3008b4b1f07ebba230de2f27d47767129a6a905c84f470
    Port:           80/TCP
    Host Port:      0/TCP
```

```
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    ConfigMapOptional:       <nil>
    DownwardAPI:             true
QoS Class:                   Burstable
Node-Selectors:              <none>
Tolerations:                 node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                             node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type    Reason     Age    From               Message
  ----    ------     ----   ----               -------
  Normal  Scheduled  3m20s  default-scheduler  Successfully assigned default/sam-pod to i-0cd0531d322fbc45d
  Normal  Pulling    3m20s  kubelet            Pulling image "nginx"
  Normal  Pulled     3m15s  kubelet            Successfully pulled image "nginx" in 4.337609908s (4.337616025s including waiting)
  Normal  Created    3m15s  kubelet            Created container cont1
  Normal  Started    3m15s  kubelet            Started container cont1
  Normal  Pulling    3m15s  kubelet            Pulling image "ubuntu"
  Normal  Pulled     3m13s  kubelet            Successfully pulled image "ubuntu" in 2.14191844s (2.141925254s including waiting)
  Normal  Created    3m13s  kubelet            Created container cont2
  Normal  Started    3m13s  kubelet            Started container cont2
root@ip-172-31-34-141:~# kubectl get nodes
NAME                STATUS   ROLES           AGE   VERSION
i-05b51dbc44e51eff8   Ready    control-plane   28m   v1.25.16
i-0cd0531d322fbc45d   Ready    node            27m   v1.25.16
root@ip-172-31-34-141:~#
```

10. Pod is created in the worker node

Step 8:

1. After updating the cluster create the pods by using deployment yaml file

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: new-dep
spec:
  replicas: 2
  selector:
    matchLabels:
      app: hello-app
  template:
    metadata:
      labels:
        app: hello-app
    spec:
      containers:
       - name: container1
         image: nginx:latest
         ports:
         - containerPort: 80
```

2. Create the deployment by using the command is **kubectl create -f filename.yml**
3. Create the service by yaml file

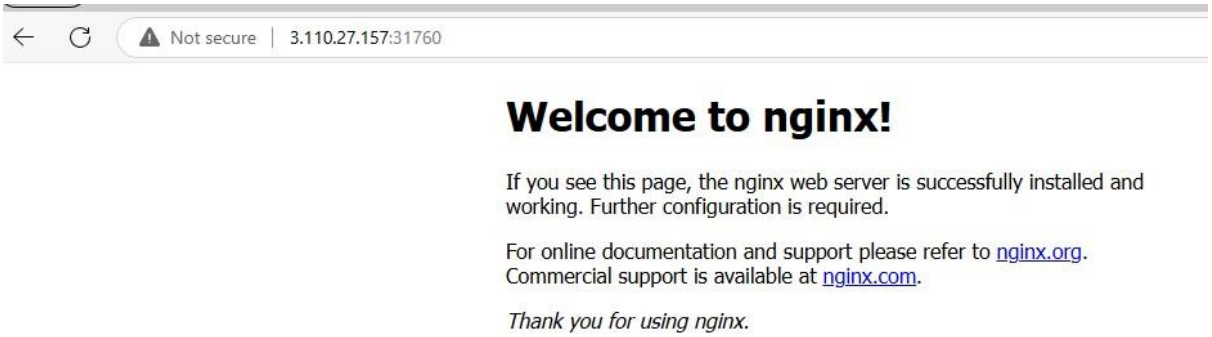
```
apiVersion: v1
kind: Service
metadata:
  name: flipkart
  labels:
    app: hello-app
spec:
```

type: NodePort
ports:
- port: 80
  targetPort: 80
  protocol: TCP
selector:
  app: hello-app

4. Create the service by the command is **kubectl create -f filename.yml**
5. Check the created services by the command is **kubectl get services -o wide**

```
root@ip-172-31-42-15:~# kubectl create -f abhi.yml
deployment.apps/new-dep created
root@ip-172-31-42-15:~# kubectl get deployment
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
new-dep   2/2     2            2           16s
root@ip-172-31-42-15:~# kubectl get deployment -o wide
NAME      READY   UP-TO-DATE   AVAILABLE   AGE   CONTAINERS   IMAGES   SELECTOR
new-dep   2/2     2            2           28s   container1   nginx    app=hello-app
root@ip-172-31-42-15:~# kubectl get pods -o wide
NAME                       READY   STATUS    RESTARTS   AGE   IP           NODE       NOMINATED NODE   READINESS GATES
new-dep-579c9bfb94-9lzcz   1/1     Running   0          58s   10.244.0.3   minikube   <none>           <none>
new-dep-579c9bfb94-vmdxw   1/1     Running   0          58s   10.244.0.4   minikube   <none>           <none>
root@ip-172-31-42-15:~# kubectl create -f mani.yml
service/hello-app created
root@ip-172-31-42-15:~# kubectl get services -o wide
NAME         TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE     SELECTOR
hello-app    NodePort    10.98.207.22   <none>        80:32009/TCP   16s     app=hello-app
kubernetes   ClusterIP   10.96.0.1      <none>        443/TCP        6m16s   <none>
root@ip-172-31-42-15:~# kubectl get nodess -o wide
error: the server doesn't have a resource type "nodess"
root@ip-172-31-42-15:~# kubectl get nodes -o wide
NAME       STATUS   ROLES           AGE     VERSION   INTERNAL-IP    EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION   CONTAINER-RUNTIME
minikube   Ready    control-plane   6m44s   v1.31.0   192.168.49.2   <none>        Ubuntu 22.04.4 LTS   6.8.0-1016-aws   docker://27.2.0
root@ip-172-31-42-15:~# minikube ip
192.168.49.2
root@ip-172-31-42-15:~# ll
total 55136
drwx------  6 root root     4096 Nov 26 06:32 ./
```

6. Change the in bound traffic rules add the nodeport number in indound rules
7. Copy the node IP and provide the nodeport number to that IP
8. It will show the created image on web page as shown in below figure



9. If you wanted to delete the replication set without deleting the pods the command is

   kubectl delete rs my-rs --cascade=orphan

10. If you want to delete all the pods at the same time the command is

    kubectl delete pods --all

11. To see all the api resources **kubectl api-resources**
12. To scale the replication by using command through cli

    kubectl scale rs my-rs --replicas 3

13. To update the image in deployment by through cli command

    kubectl set image deployment/my-dep cont1=new image

14. To roll back the image the command is

    kubectl rollout undo deployment/my-dep --to-revision=1