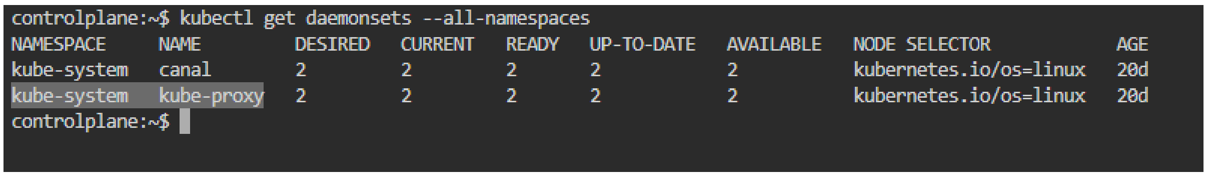
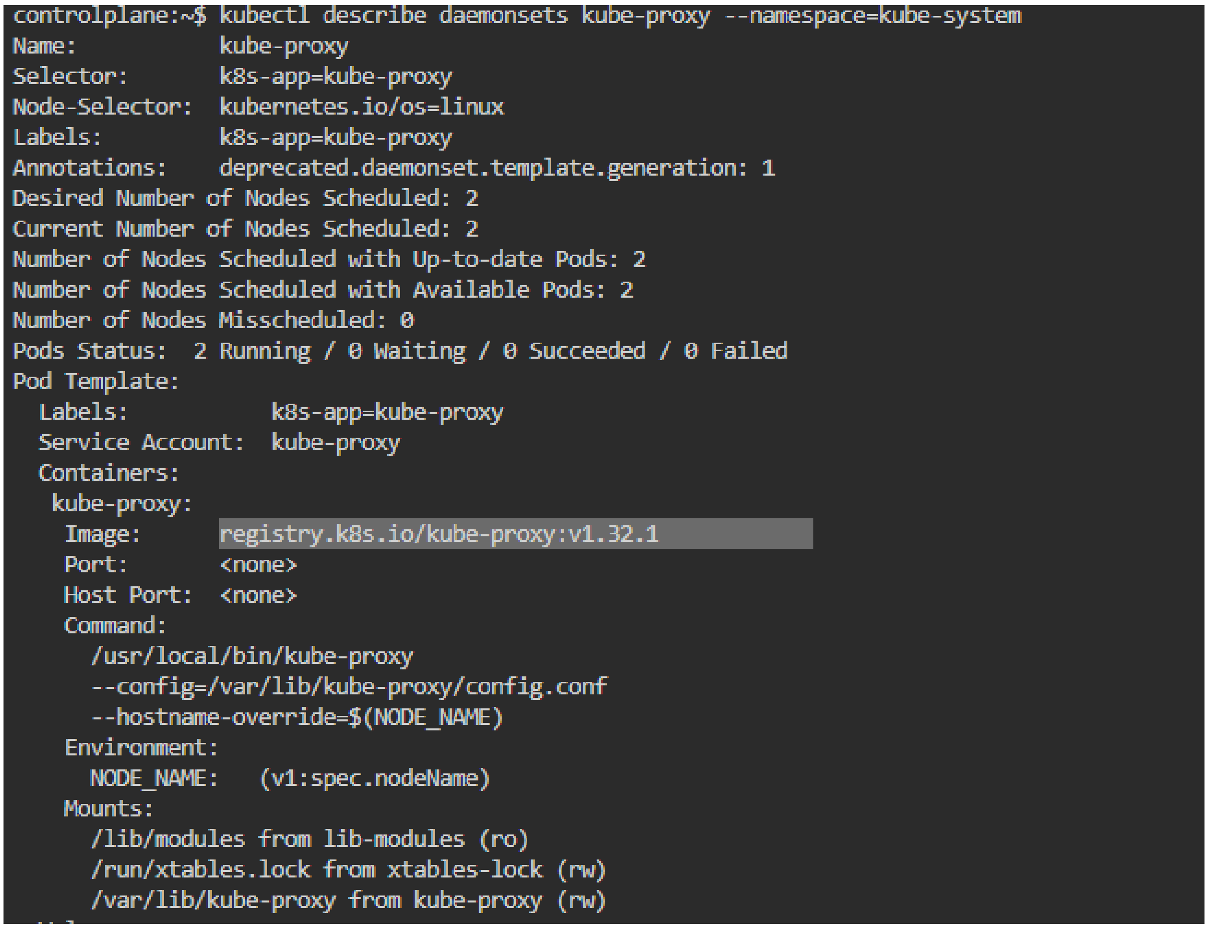
1- How many DaemonSets are created in the cluster in all namespaces?

2- what DaemonSets exist on the kube-system namespace?



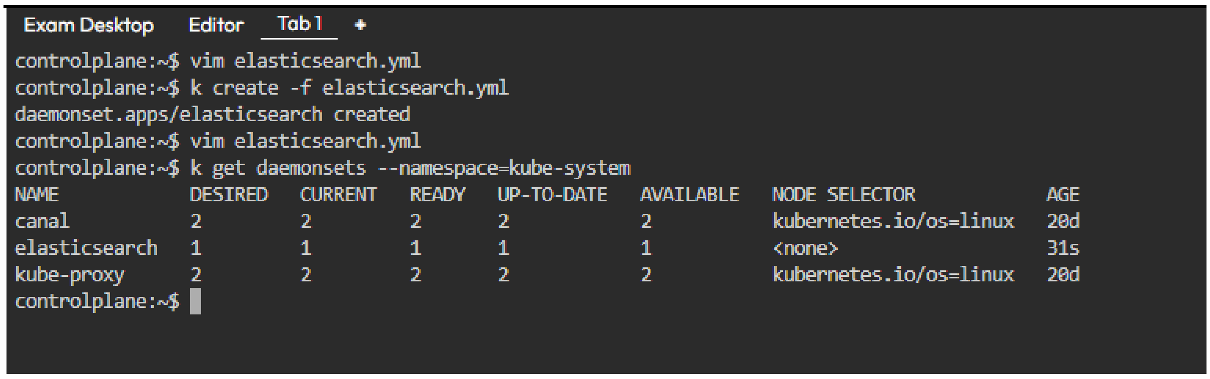
```
controlplane:~$ kubectl get daemonsets --all-namespaces
NAMESPACE     NAME         DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR            AGE
kube-system   canal        2         2         2       2            2           kubernetes.io/os=linux   20d
kube-system   kube-proxy   2         2         2       2            2           kubernetes.io/os=linux   20d
controlplane:~$
```

3- What is the image used by the POD deployed by the kube-proxy DaemonSet

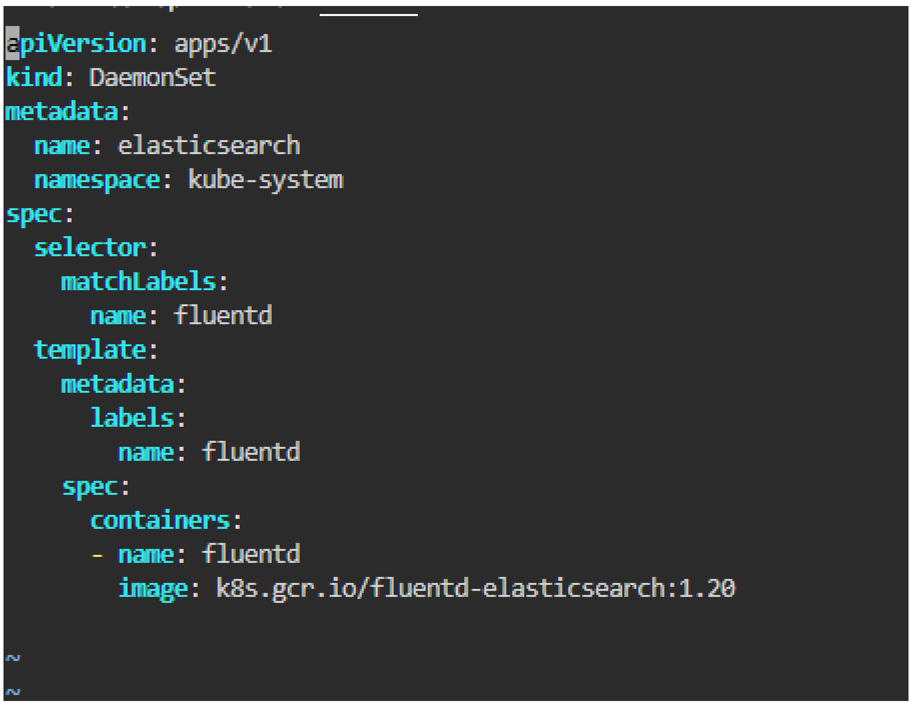


```
controlplane:~$ kubectl describe daemonsets kube-proxy --namespace=kube-system
Name:           kube-proxy
Selector:       k8s-app=kube-proxy
Node-Selector:  kubernetes.io/os=linux
Labels:         k8s-app=kube-proxy
Annotations:    deprecated.daemonset.template.generation: 1
Desired Number of Nodes Scheduled: 2
Current Number of Nodes Scheduled: 2
Number of Nodes Scheduled with Up-to-date Pods: 2
Number of Nodes Scheduled with Available Pods: 2
Number of Nodes Misscheduled: 0
Pods Status:  2 Running / 0 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:           k8s-app=kube-proxy
  Service Account:  kube-proxy
  Containers:
   kube-proxy:
    Image:      registry.k8s.io/kube-proxy:v1.32.1
    Port:       <none>
    Host Port:  <none>
    Command:
      /usr/local/bin/kube-proxy
      --config=/var/lib/kube-proxy/config.conf
      --hostname-override=$(NODE_NAME)
    Environment:
      NODE_NAME:   (v1:spec.nodeName)
    Mounts:
      /lib/modules from lib-modules (ro)
      /run/xtables.lock from xtables-lock (rw)
      /var/lib/kube-proxy from kube-proxy (rw)
```

4- Deploy a DaemonSet for FluentD Logging. Use the given specifications. Name: elasticsearch Namespace: kube-system Image: k8s.gcr.io/fluentd-elasticsearch:1.20

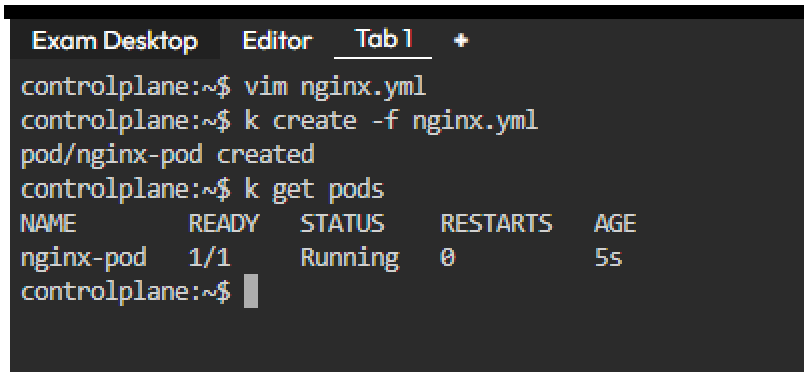


```
controlplane:~$ vim elasticsearch.yml
controlplane:~$ k create -f elasticsearch.yml
daemonset.apps/elasticsearch created
controlplane:~$ vim elasticsearch.yml
controlplane:~$ k get daemonsets --namespace=kube-system
NAME            DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR            AGE
canal           2         2         2       2            2           kubernetes.io/os=linux   20d
elasticsearch   1         1         1       1            1           <none>                   31s
kube-proxy      2         2         2       2            2           kubernetes.io/os=linux   20d
controlplane:~$
```

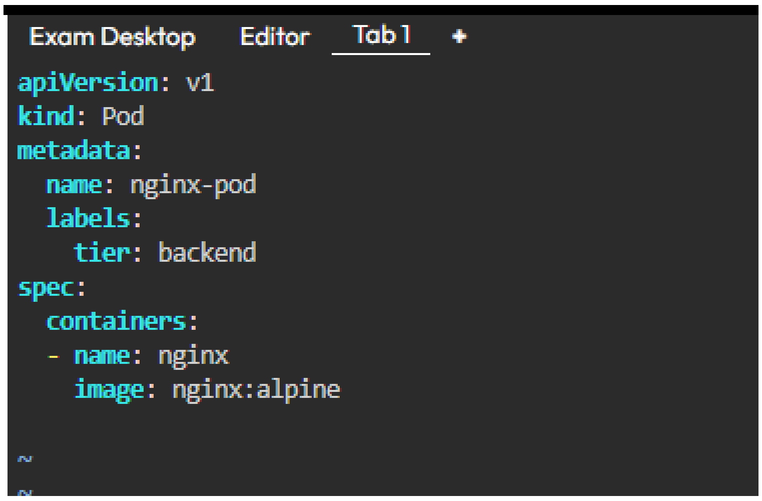


```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: elasticsearch
  namespace: kube-system
spec:
  selector:
    matchLabels:
      name: fluentd
  template:
    metadata:
      labels:
        name: fluentd
    spec:
      containers:
      - name: fluentd
        image: k8s.gcr.io/fluentd-elasticsearch:1.20

~
~
```

5- Deploy a pod named nginx-pod using the nginx:alpine image with the labels set to tier=backend.

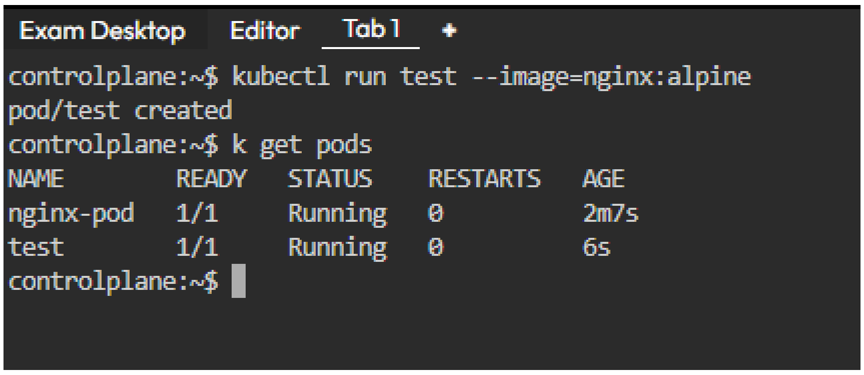


```
controlplane:~$ vim nginx.yml
controlplane:~$ k create -f nginx.yml
pod/nginx-pod created
controlplane:~$ k get pods
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   0          5s
controlplane:~$
```



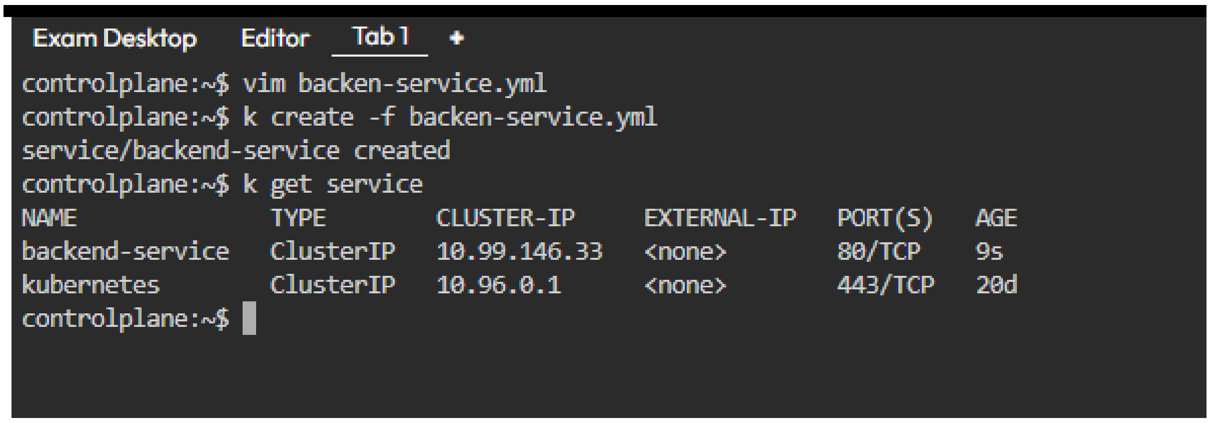
```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
  labels:
    tier: backend
spec:
  containers:
  - name: nginx
    image: nginx:alpine

~
~
```

6- Deploy a test pod using the nginx:alpine image

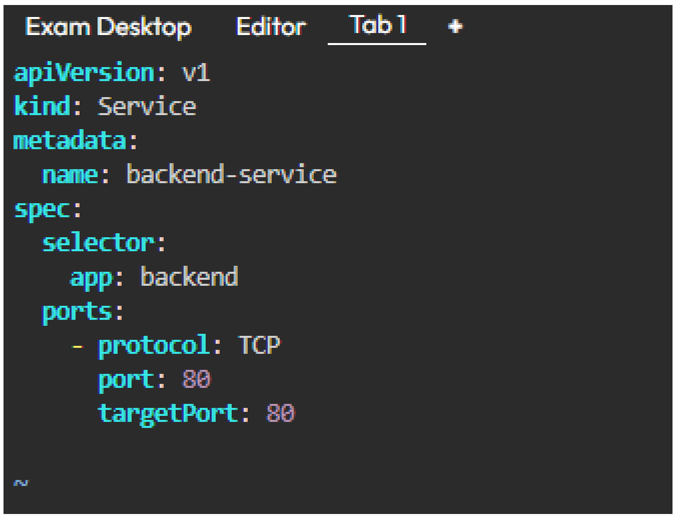


```
controlplane:~$ kubectl run test --image=nginx:alpine
pod/test created
controlplane:~$ k get pods
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   0          2m7s
test        1/1     Running   0          6s
controlplane:~$
```

7- Create a service backend-service to expose the backend application within the cluster on port 80.

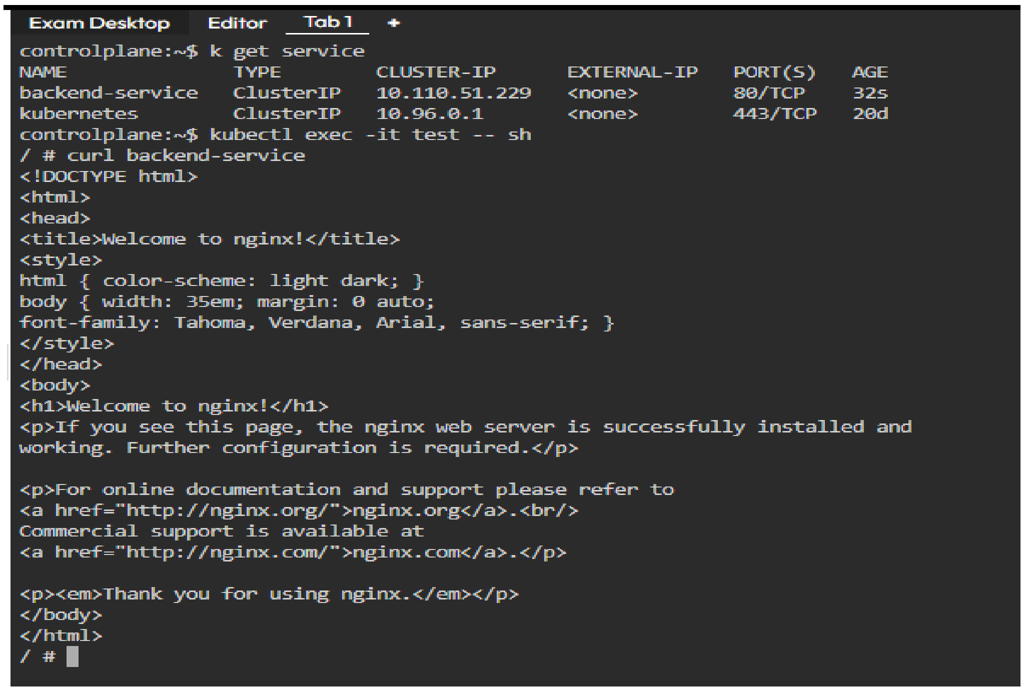


```
controlplane:~$ vim backen-service.yml
controlplane:~$ k create -f backen-service.yml
service/backend-service created
controlplane:~$ k get service
NAME              TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
backend-service   ClusterIP   10.99.146.33    <none>        80/TCP    9s
kubernetes        ClusterIP   10.96.0.1       <none>        443/TCP   20d
controlplane:~$
```



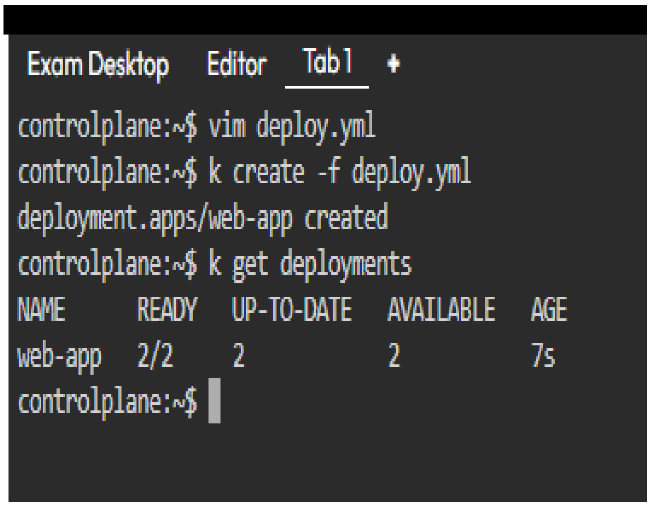
```
apiVersion: v1
kind: Service
metadata:
  name: backend-service
spec:
  selector:
    app: backend
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80

~
```

8- try to curl the backend-service from the test pod. What is the response?

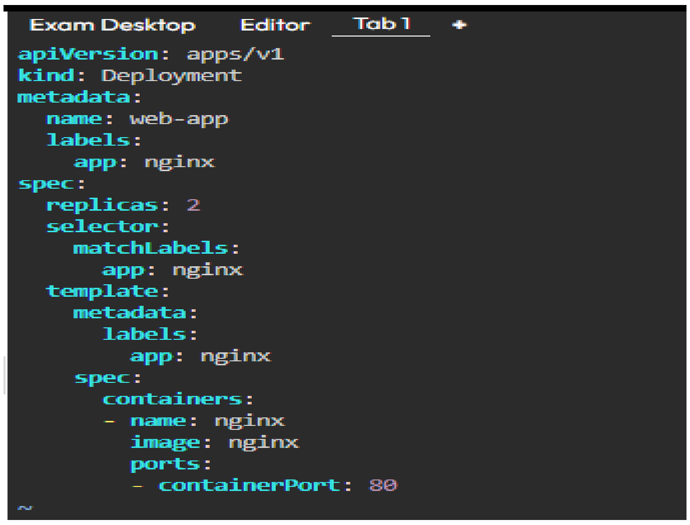


```
controlplane:~$ k get service
NAME              TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
backend-service   ClusterIP   10.110.51.229   <none>        80/TCP    32s
kubernetes        ClusterIP   10.96.0.1       <none>        443/TCP   20d
controlplane:~$ kubectl exec -it test -- sh
/ # curl backend-service
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
/ #
```

9- Create a deployment named web-app using the image nginx with 2 replicas

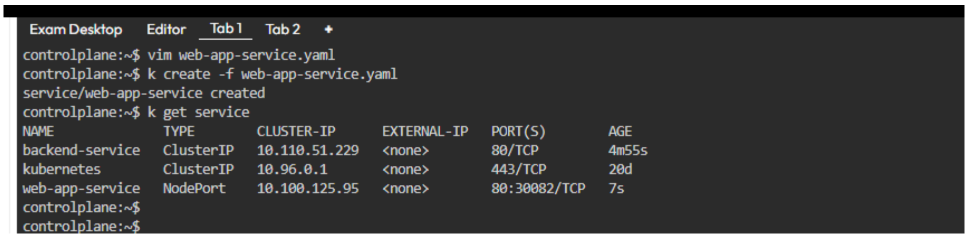


```
controlplane:~$ vim deploy.yml
controlplane:~$ k create -f deploy.yml
deployment.apps/web-app created
controlplane:~$ k get deployments
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
web-app   2/2     2            2           7s
controlplane:~$
```

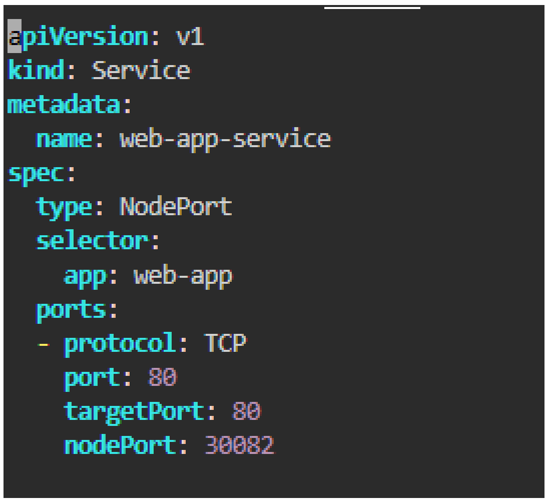


```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: web-app
  labels:
    app: nginx
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
~
```

10- Expose the web-app as service web-app-service application on port 80 and nodeport 30082 on the nodes on the cluster

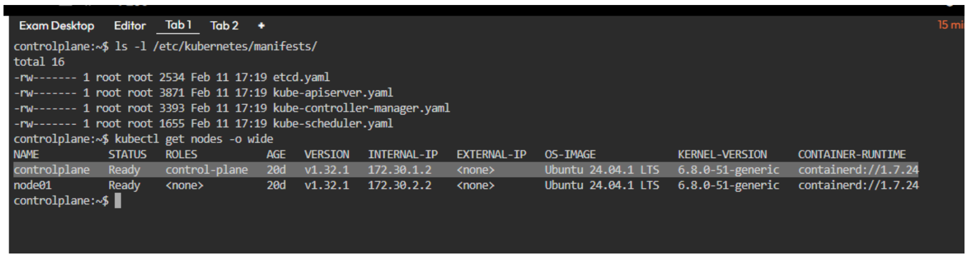


```
controlplane:~$ vim web-app-service.yaml
controlplane:~$ k create -f web-app-service.yaml
service/web-app-service created
controlplane:~$ k get service
NAME              TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
backend-service   ClusterIP   10.110.51.229   <none>        80/TCP         4m55s
kubernetes        ClusterIP   10.96.0.1       <none>        443/TCP        20d
web-app-service   NodePort    10.100.125.95   <none>        80:30082/TCP   7s
controlplane:~$
controlplane:~$
```



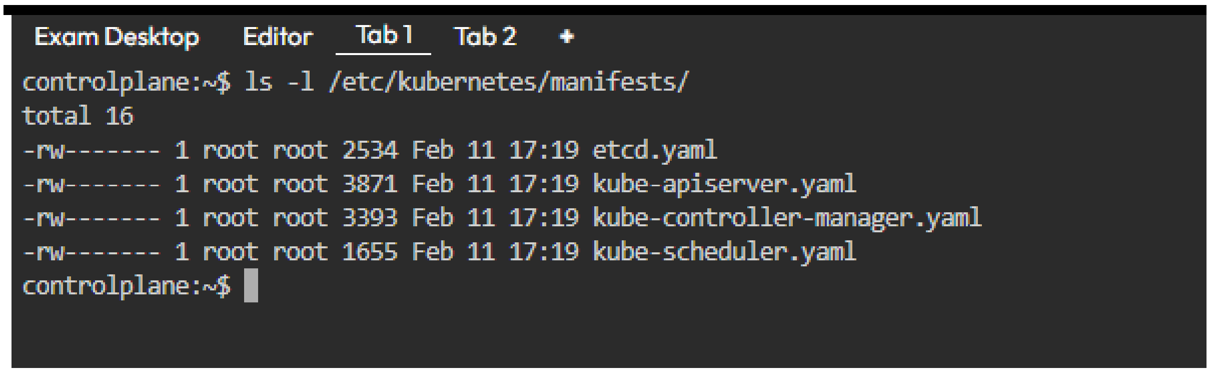
```
apiVersion: v1
kind: Service
metadata:
  name: web-app-service
spec:
  type: NodePort
  selector:
    app: web-app
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
    nodePort: 30082
```

11- access the web app from the node



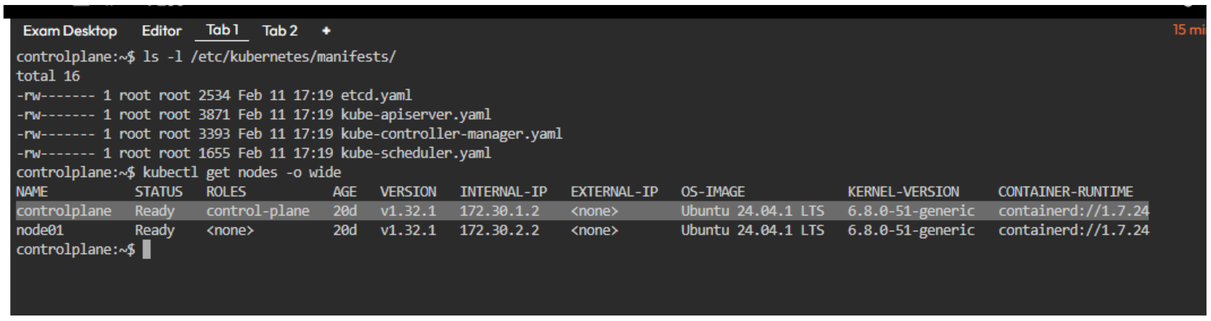
```
controlplane:~$ ls -l /etc/kubernetes/manifests/
total 16
-rw------- 1 root root 2534 Feb 11 17:19 etcd.yaml
-rw------- 1 root root 3871 Feb 11 17:19 kube-apiserver.yaml
-rw------- 1 root root 3393 Feb 11 17:19 kube-controller-manager.yaml
-rw------- 1 root root 1655 Feb 11 17:19 kube-scheduler.yaml
controlplane:~$ kubectl get nodes -o wide
NAME           STATUS   ROLES           AGE   VERSION   INTERNAL-IP   EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION     CONTAINER-RUNTIME
controlplane   Ready    control-plane   20d   v1.32.1   172.30.1.2    <none>        Ubuntu 24.04.1 LTS   6.8.0-51-generic   containerd://1.7.24
node01         Ready    <none>          20d   v1.32.1   172.30.2.2    <none>        Ubuntu 24.04.1 LTS   6.8.0-51-generic   containerd://1.7.24
controlplane:~$
```

12- How many static pods exist in this cluster in all namespaces?



```
controlplane:~$ ls -l /etc/kubernetes/manifests/
total 16
-rw------- 1 root root 2534 Feb 11 17:19 etcd.yaml
-rw------- 1 root root 3871 Feb 11 17:19 kube-apiserver.yaml
-rw------- 1 root root 3393 Feb 11 17:19 kube-controller-manager.yaml
-rw------- 1 root root 1655 Feb 11 17:19 kube-scheduler.yaml
controlplane:~$
```

13-On which nodes are the static pods created currently?



```
controlplane:~$ ls -l /etc/kubernetes/manifests/
total 16
-rw------- 1 root root 2534 Feb 11 17:19 etcd.yaml
-rw------- 1 root root 3871 Feb 11 17:19 kube-apiserver.yaml
-rw------- 1 root root 3393 Feb 11 17:19 kube-controller-manager.yaml
-rw------- 1 root root 1655 Feb 11 17:19 kube-scheduler.yaml
controlplane:~$ kubectl get nodes -o wide
NAME           STATUS   ROLES           AGE   VERSION   INTERNAL-IP   EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION     CONTAINER-RUNTIME
controlplane   Ready    control-plane   20d   v1.32.1   172.30.1.2    <none>        Ubuntu 24.04.1 LTS   6.8.0-51-generic   containerd://1.7.24
node01         Ready    <none>          20d   v1.32.1   172.30.2.2    <none>        Ubuntu 24.04.1 LTS   6.8.0-51-generic   containerd://1.7.24
controlplane:~$
```