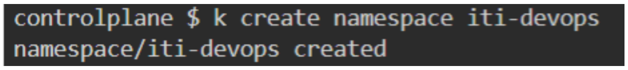
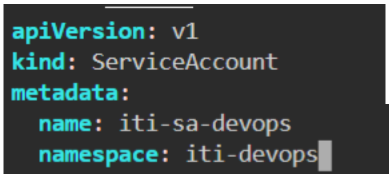
FINAL LAB

1- create a namespace iti-devops

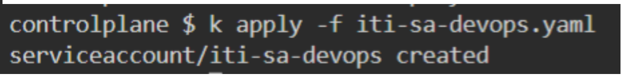


```
controlplane $ k create namespace iti-devops
namespace/iti-devops created
```

2- create a service account iti-sa-devops under the same namespace

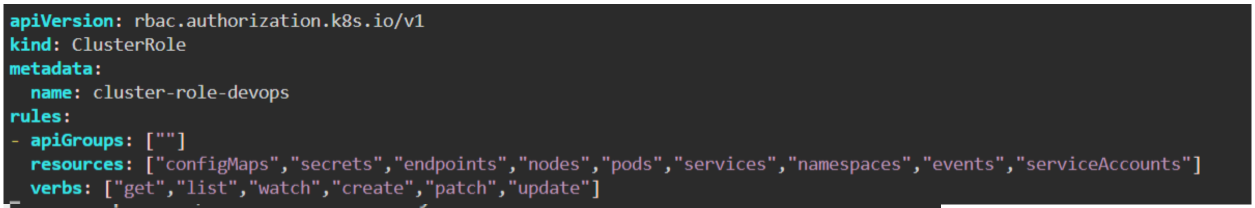


```yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: iti-sa-devops
  namespace: iti-devops
```

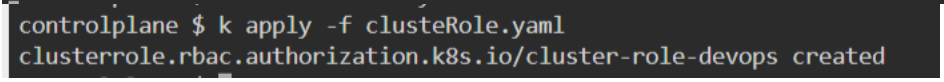


```
controlplane $ k apply -f iti-sa-devops.yaml
serviceaccount/iti-sa-devops created
```

3- create a clusteRole which should be named as cluster-role-devops to grant permissions “get”,”list”,”watch”,”create”,”patch”,”update” to “configMaps”,”secrets”,”endpoints”,”nodes”,”pods”,”services”,”namespaces”,”events”,”servicnts”.

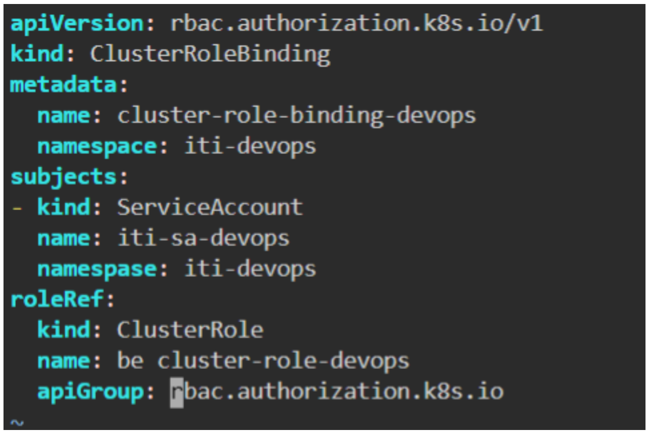


```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: cluster-role-devops
rules:
- apiGroups: [""]
  resources: ["configMaps","secrets","endpoints","nodes","pods","services","namespaces","events","serviceAccounts"]
  verbs: ["get","list","watch","create","patch","update"]
```

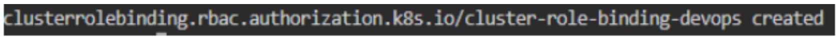


```
controlplane $ k apply -f clusteRole.yaml
clusterrole.rbac.authorization.k8s.io/cluster-role-devops created
```

4- create a ClusterRoleBinding which should be named as cluster-role-binding-devops under the same namespace. Define roleRef apiGroup should be rbac.authorization.k8s.io . Kind should be ClusterRole, name should be cluster-role-devops and subjects kind should be ServiceAccount: name should be iti-sa-devops and namespace should be iti-devops



```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: cluster-role-binding-devops
  namespace: iti-devops
subjects:
- kind: ServiceAccount
  name: iti-sa-devops
  namespase: iti-devops
roleRef:
  kind: ClusterRole
  name: be cluster-role-devops
  apiGroup: rbac.authorization.k8s.io
~
```

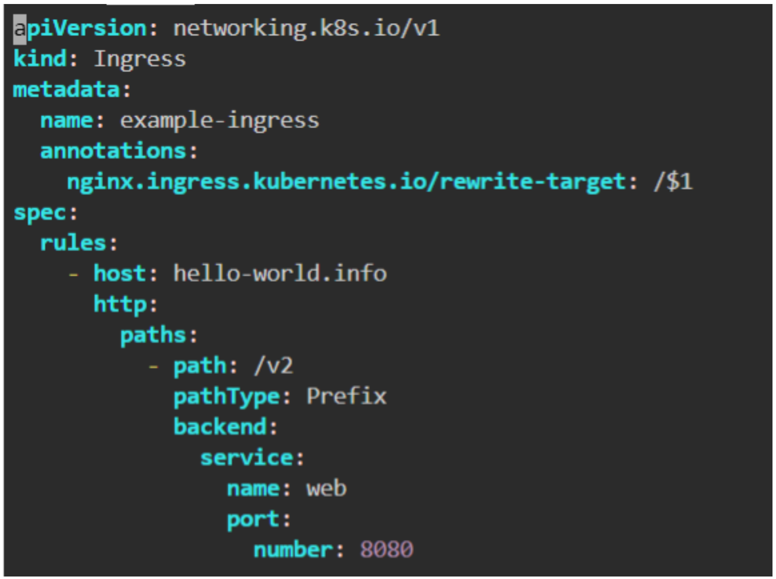


```
clusterrolebinding.rbac.authorization.k8s.io/cluster-role-binding-devops created
```

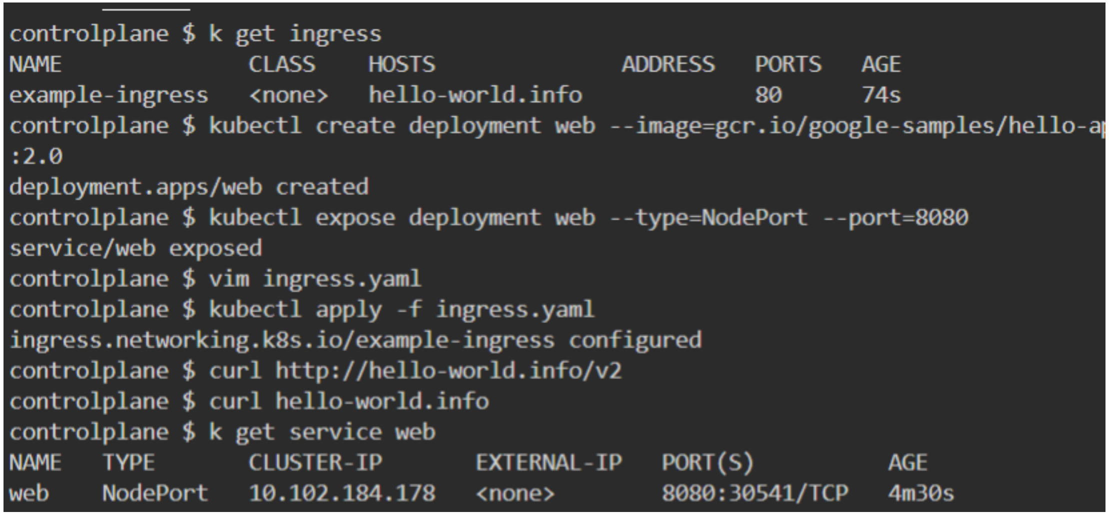
5- What is the difference between statefulSets and deployments?

- Deployment manages multiple pods by automating the creation, updating, and deletion of
ReplicaSet. Deployment, on the other hand, is suitable for stateless workloads that use multiple
replicas of one pod, such as web servers like Nginx and Apache
- StatefulSet helps orchestrate stateful pods by guaranteeing the ordering and uniqueness of pod
replicas. StatefulSet is better suited to stateful workloads that require persistent storage on each
cluster node, such as databases and other identity-sensitive workloads

6- Set up Ingress on Minikube with the NGINX Ingress Controller play around with paths , you can create more than 2 deployments if you like
https://kubernetes.io/docs/tasks/access-application□cluster/ingress-minikube/



```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: example-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /$1
spec:
  rules:
    - host: hello-world.info
      http:
        paths:
          - path: /v2
            pathType: Prefix
            backend:
              service:
                name: web
                port:
                  number: 8080
```



```
controlplane $ k get ingress
NAME              CLASS    HOSTS              ADDRESS   PORTS   AGE
example-ingress   <none>   hello-world.info             80      74s
controlplane $ kubectl create deployment web --image=gcr.io/google-samples/hello-ap
:2.0
deployment.apps/web created
controlplane $ kubectl expose deployment web --type=NodePort --port=8080
service/web exposed
controlplane $ vim ingress.yaml
controlplane $ kubectl apply -f ingress.yaml
ingress.networking.k8s.io/example-ingress configured
controlplane $ curl http://hello-world.info/v2
controlplane $ curl hello-world.info
controlplane $ k get service web
NAME   TYPE       CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
web    NodePort   10.102.184.178   <none>        8080:30541/TCP   4m30s
```