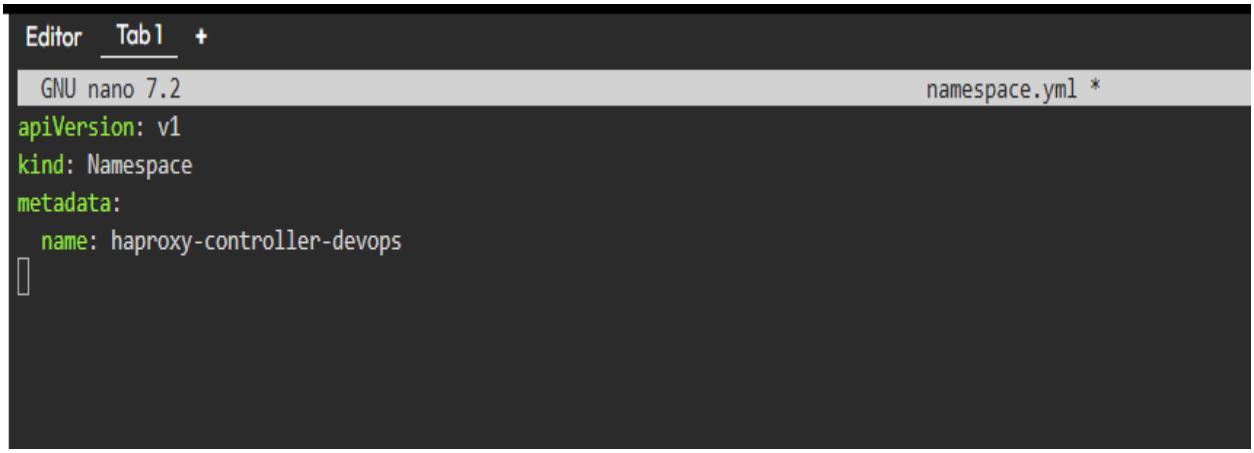
Mohanad Mahmoud Sayed

Lab5

1- Create a namespace haproxy-controller-devops.

Firstly: Create the namespace.yml:



Then, check and apply:

```
controlplane:~$ kubectl apply -f namespace.yml
namespace/haproxy-controller-devops created
controlplane:~$ k get namespace
NAME                        STATUS   AGE
default                     Active   34d
haproxy-controller-devops   Active   13s
kube-node-lease             Active   34d
kube-public                 Active   34d
kube-system                 Active   34d
local-path-storage          Active   34d
controlplane:~$
```

2- Create a ServiceAccount haproxy-service-account-devops under the same namespace.

Firstly: Create the service-account.yml:

```
Editor   Tab 1   +
 GNU nano 7.2                                          service-account.yml *
apiVersion: v1
kind: ServiceAccount
metadata:
  name: haproxy-service-account-devops
  namespace: haproxy-controller-devops
```

Then, check and apply:

```
controlplane:~$ kubectl apply -f service-account.yml
serviceaccount/haproxy-service-account-devops created
```

3- Create a ClusterRole which should be named as haproxy-cluster-role-devops, to grant permissions "get", "list", "watch", "create", "patch", "update" to "Configmaps",”secrets”,"endpoints","nodes","pods","services", "namespaces","events","serviceaccounts".

```
Editor   Tab 1   +
 GNU nano 7.2                                          cluster-role.yml *
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: haproxy-cluster-role-devops
rules:
- apiGroups: [""]
  resources: ["configmaps", "secrets", "endpoints", "nodes", "pods", "services", "namespaces", "events", "serviceaccounts"]
  verbs: ["get", "list", "watch", "create", "patch", "update"]
```

Then, check and apply:

```
controlplane:~$ kubectl apply -f cluster-role.yml
clusterrole.rbac.authorization.k8s.io/haproxy-cluster-role-devops created
```

```
Editor   Tab 1   +
controlplane:~$ kubectl get clusterrole haproxy-cluster-role-devops
NAME                          CREATED AT
haproxy-cluster-role-devops   2025-04-26T19:09:03Z
controlplane:~$
```

4- Create a ClusterRoleBinding which should be named as haproxy-cluster-role-binding-devops under the same namespace. Define roleRef apiGroup should be rbac.authorization.k8s.io, kind should be ClusterRole, name should be haproxy-cluster-role-devops and subjects kind should be ServiceAccount, name should be haproxy-service-account-devops and namespace should be haproxy-controller-devops.

Firstly: Create the clusterrole file:

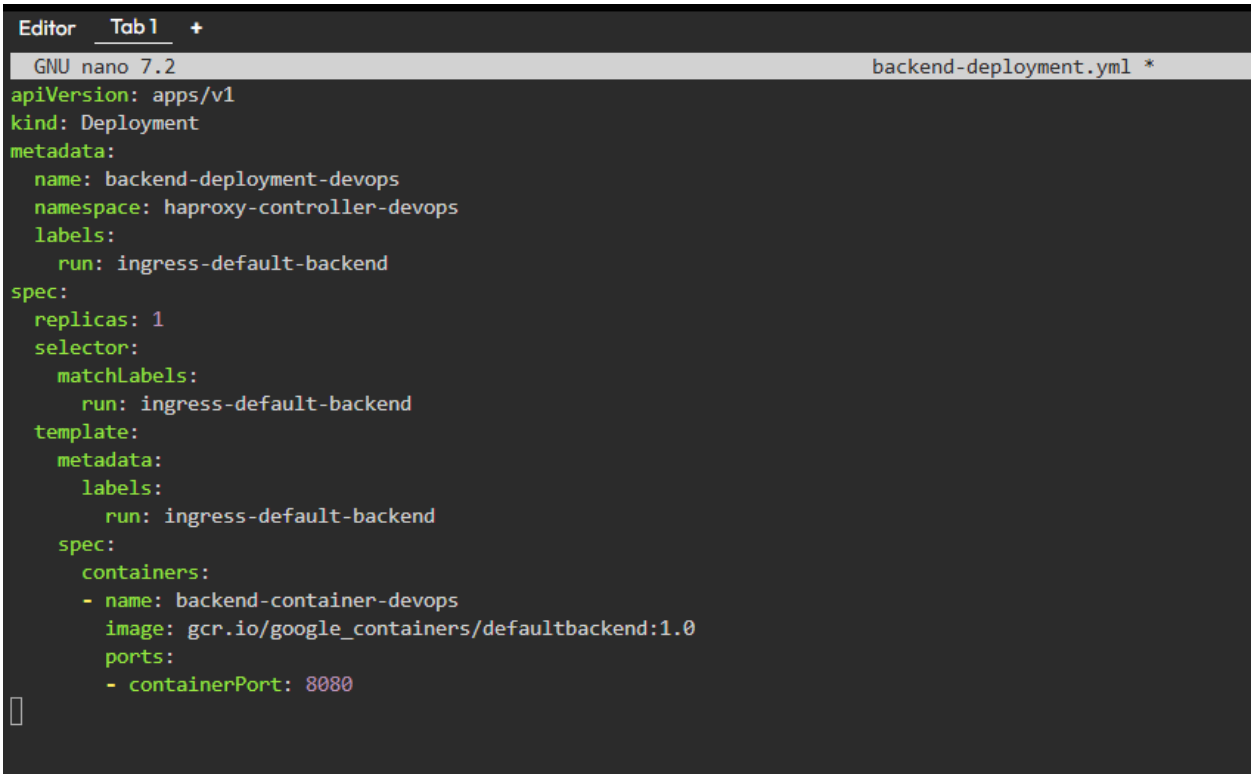
```
Editor   Tab 1   +
  GNU nano 7.2                                          clusterrole-binding.yml *
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: haproxy-cluster-role-binding-devops
subjects:
- kind: ServiceAccount
  name: haproxy-service-account-devops
  namespace: haproxy-controller-devops
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: haproxy-cluster-role-devops
```

Then, check and apply:

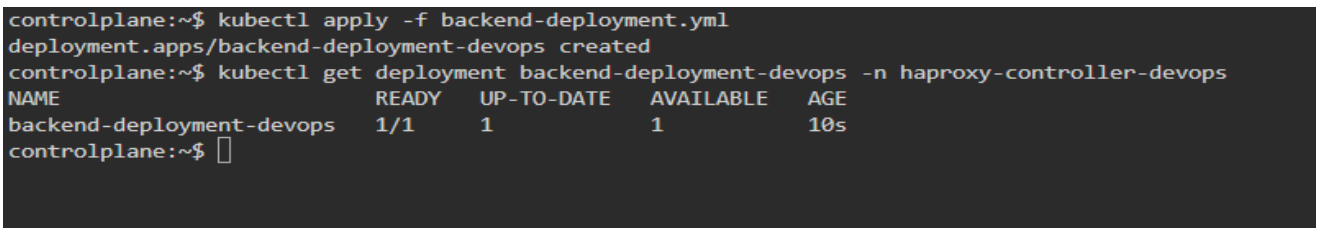
```
controlplane:~$ kubectl apply -f clusterrole-binding.yml
clusterrolebinding.rbac.authorization.k8s.io/haproxy-cluster-role-binding-devops created
controlplane:~$ kubectl get clusterrolebinding haproxy-cluster-role-binding-devops
NAME                                  ROLE                                        AGE
haproxy-cluster-role-binding-devops   ClusterRole/haproxy-cluster-role-devops     14s
controlplane:~$
```

5- Create a backend deployment which should be named as backend-deployment-devops under the same namespace, labels run should be ingress-default-backend under metadata. Configure spec as replica should be 1, selector's matchLabels run should be ingress-default-backend. Template's labels run under metadata should be ingress-default-backend. The container should named as backend-container-devops, use image gcr.io/google_containers/defaultbackend:1.0 (use exact name of image as mentioned) and its containerPort should be 8080.

Firstly: Create the backend-deployment.yml:



Then, check and apply:



6- Create a service for backend which should be named as service-backend-devops under the same namespace, labels run should be ingress-default-backend. Configure spec as selector's run should be ingress-default-backend, port should be named as port-backend, protocol should be TCP, port should be 8080 and targetPort should be 8080.

Firstly: Create the backend-service.yml:

```
Editor   Tab 1   +
 GNU nano 7.2                                          backend-service.yml *
apiVersion: v1
kind: Service
metadata:
  name: service-backend-devops
  namespace: haproxy-controller-devops
  labels:
    run: ingress-default-backend
spec:
  selector:
    run: ingress-default-backend
  ports:
  - name: port-backend
    protocol: TCP
    port: 8080
    targetPort: 8080
```

Then, check and apply:

```
controlplane:~$ kubectl apply -f backend-service.yml
service/service-backend-devops created
```

```
controlplane:~$ kubectl get service service-backend-devops -n haproxy-controller-devops
NAME                     TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)    AGE
service-backend-devops   ClusterIP   10.107.230.209   <none>        8080/TCP   38s
controlplane:~$
```

7- Create a deployment for frontend which should be named haproxy-ingress-devops under the same namespace. Configure spec as replica should be 1, selector's matchLabels should be haproxy-ingress, template's labels run should be haproxy-ingress under metadata. The container name should be ingress-container-devops under the same service account haproxy-service-account-devops, use image haproxytech/kubernetes-ingress, give args as --default-backend-service=haproxy-controller-devops/service-backend-devops, resources requests for cpu should be 500m and for memory should be 50Mi, livenessProbe httpGet path should be /healthz its port should be 1024. The first port name should be http and its containerPort should be 80, second port name should be https and its containerPort should be 443 and third port name should be stat its containerPort should be 1024. Define environment as first env name should be TZ its value should be Etc/UTC, second env name should be POD_NAME its valueFrom: fieldRef: fieldPath: should be metadata.name and third env name should be POD_NAMESPACE its valueFrom: fieldRef: fieldPath: should be metadata.namespace.

Firstly: Create the namespace.yml:

Editor Tab 1 +

GNU nano 7.2 frontend-deployment.yml *

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: haproxy-ingress-devops
  namespace: haproxy-controller-devops
spec:
  replicas: 1
  selector:
    matchLabels:
      run: haproxy-ingress
  template:
    metadata:
      labels:
        run: haproxy-ingress
    spec:
      serviceAccountName: haproxy-service-account-devops
      containers:
      - name: ingress-container-devops
        image: haproxytech/kubernetes-ingress
        args:
        - --default-backend-service=haproxy-controller-devops/service-backend-devops
        resources:
          requests:
            cpu: 500m
            memory: 50Mi
        livenessProbe:
          httpGet:
            path: /healthz
            port: 1024
        ports:
        - name: http
          containerPort: 80
        - name: https
          containerPort: 443
        - name: stat
          containerPort: 1024
        env:
        - name: TZ
          value: Etc/UTC
        - name: POD_NAME
          valueFrom:
            fieldRef:
              fieldPath: metadata.name
        - name: POD_NAMESPACE
          valueFrom:
            fieldRef:
              fieldPath: metadata.namespace
```

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location M-U Undo
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line M-E Redo

Then, check and apply:

```
controlplane:~$ kubectl apply -f frontend-deployment.yml
deployment.apps/haproxy-ingress-devops created
controlplane:~$ kubectl get deployment haproxy-ingress-devops -n haproxy-controller-devops
NAME                     READY   UP-TO-DATE   AVAILABLE   AGE
haproxy-ingress-devops   1/1     1            1           14s
controlplane:~$


controlplane:~$ kubectl describe deployment haproxy-ingress-devops -n haproxy-controller-devops
Name:                   haproxy-ingress-devops
Namespace:              haproxy-controller-devops
CreationTimestamp:      Sat, 26 Apr 2025 19:23:07 +0000
Labels:                 <none>
Annotations:            deployment.kubernetes.io/revision: 1
Selector:               run=haproxy-ingress
Replicas:               1 desired | 1 updated | 1 total | 1 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:           run=haproxy-ingress
  Service Account:  haproxy-service-account-devops
  Containers:
   ingress-container-devops:
    Image:       haproxytech/kubernetes-ingress
    Ports:       80/TCP, 443/TCP, 1024/TCP
    Host Ports:  0/TCP, 0/TCP, 0/TCP
    Args:
      --default-backend-service=haproxy-controller-devops/service-backend-devops
    Requests:
      cpu:     500m
      memory:  50Mi
    Liveness:  http-get http://:1024/healthz delay=0s timeout=1s period=10s #success=1 #failure=3
    Environment:
      TZ:             Etc/UTC
      POD_NAME:        (v1:metadata.name)
      POD_NAMESPACE:   (v1:metadata.namespace)
    Mounts:           <none>
  Volumes:            <none>
  Node-Selectors:     <none>
  Tolerations:        <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Available      True    MinimumReplicasAvailable
  Progressing    True    NewReplicaSetAvailable
OldReplicaSets:  <none>
NewReplicaSet:   haproxy-ingress-devops-6987454f5f (1/1 replicas created)
Events:
  Type    Reason             Age   From                   Message
  ----    ------             ----  ----                   -------
  Normal  ScalingReplicaSet  60s   deployment-controller  Scaled up replica set haproxy-ingress-devops-6987454f5f from 0 to 1
controlplane:~$
```

➔ The deployment has:

- 1 replica
- **Container**: ingress-container-devops
- **Image**: haproxytech/kubernetes-ingress

- **Args**: include --default-backend-service=haproxy-controller-devops/service-backend-devops
- **ServiceAccount**: haproxy-service-account-devops
- **3 container ports**: 80, 443, 1024
- **Liveness probe**: /healthz on port 1024
- **Environment variables**: TZ, POD_NAME, POD_NAMESPACE

8- Create a service for frontend which should be named as ingress-service-devops under same namespace, labels run should be haproxy-ingress. Configure spec as selectors' run should be haproxy-ingress, type should be NodePort. The first port name should be http, its port should be 80, protocol should be TCP, targetPort should be 80 and nodePort should be 32456. The second port name should be https, its port should be 443, protocol should be TCP, targetPort should be 443 and nodePort should be 32567. The third port name should be stat, its port should be 1024, protocol should be TCP, targetPort should be 1024 and nodePort should be 32678.

Firstly: Create the namespace.yml:

Editor Tab 1 +

GNU nano 7.2 frontend-service.yml *

```
apiVersion: v1
kind: Service
metadata:
  name: ingress-service-devops
  namespace: haproxy-controller-devops
  labels:
    run: haproxy-ingress
spec:
  type: NodePort
  selector:
    run: haproxy-ingress
  ports:
  - name: http
    protocol: TCP
    port: 80
    targetPort: 80
    nodePort: 32456
  - name: https
    protocol: TCP
    port: 443
    targetPort: 443
    nodePort: 32567
  - name: stat
    protocol: TCP
    port: 1024
    targetPort: 1024
    nodePort: 32678
```

Then, check and apply:

```
controlplane:~$ kubectl get service ingress-service-devops -n haproxy-controller-devops
NAME                     TYPE       CLUSTER-IP       EXTERNAL-IP   PORT(S)                                  AGE
ingress-service-devops   NodePort   10.102.100.205   <none>        80:32456/TCP,443:32567/TCP,1024:32678/TCP   15s
controlplane:~$ kubectl describe service ingress-service-devops -n haproxy-controller-devops
Name:                     ingress-service-devops
Namespace:                haproxy-controller-devops
Labels:                   run=haproxy-ingress
Annotations:              <none>
Selector:                 run=haproxy-ingress
Type:                     NodePort
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       10.102.100.205
IPs:                      10.102.100.205
Port:                     http  80/TCP
TargetPort:               80/TCP
NodePort:                 http  32456/TCP
Endpoints:
Port:                     https  443/TCP
TargetPort:               443/TCP
NodePort:                 https  32567/TCP
Endpoints:
Port:                     stat  1024/TCP
TargetPort:               1024/TCP
NodePort:                 stat  32678/TCP
Endpoints:
Session Affinity:         None
External Traffic Policy:  Cluster
Internal Traffic Policy:  Cluster
Events:                   <none>
controlplane:~$
```

➔This screenshot shows that the service is type of node port:

- port 80 → targetPort 80 → nodePort 32456
- port 443 → targetPort 443 → nodePort 32567
- port 1024 → targetPort 1024 → nodePort 32678