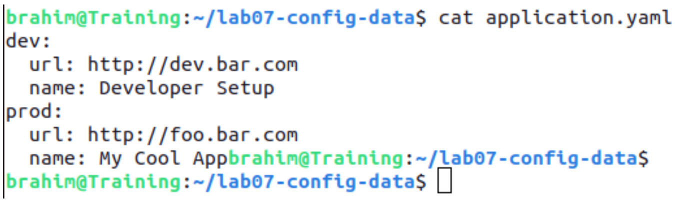
Lab07 – Configuring a Pod to use ConfigMap

In this exercise, you will first create a ConfigMap from a YAML configuration file as a source. Later, you'll create a Pod, consume the ConfigMap as Volume and inspect the key-value pairs as files. Also you will first create a Secret from literal values. Next, you'll create a Pod and consume the Secret as environment variables. Finally, you'll print out its values from within the container.

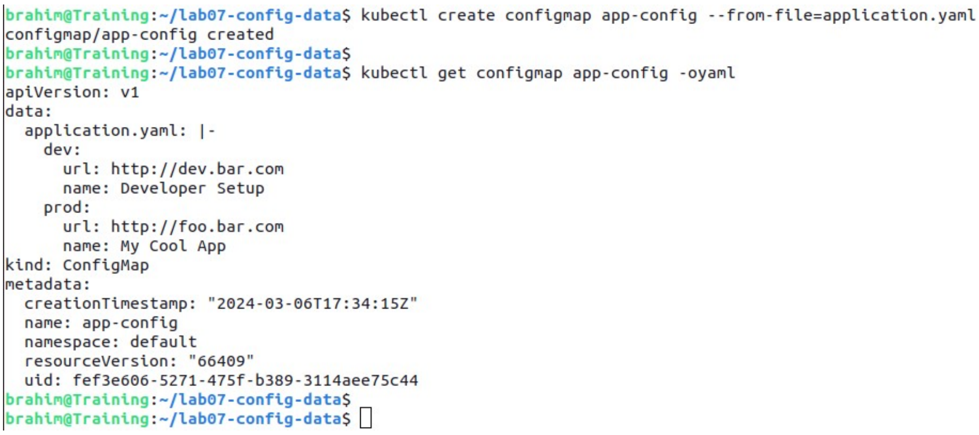
ConfigMap

1. Inspect the YAML configuration file named `application.yaml`



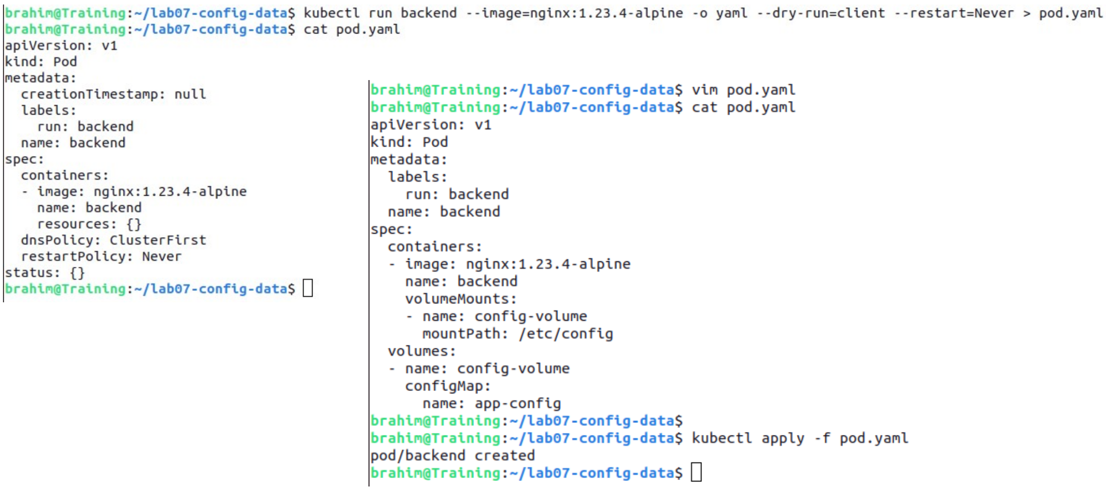
```
brahim@Training:~/lab07-config-data$ cat application.yaml
dev:
  url: http://dev.bar.com
  name: Developer Setup
prod:
  url: http://foo.bar.com
  name: My Cool Appbrahim@Training:~/lab07-config-data$
brahim@Training:~/lab07-config-data$
```

2. Create a new ConfigMap named `app-config` from that file.



```
brahim@Training:~/lab07-config-data$ kubectl create configmap app-config --from-file=application.yaml
configmap/app-config created
brahim@Training:~/lab07-config-data$
brahim@Training:~/lab07-config-data$ kubectl get configmap app-config -oyaml
apiVersion: v1
data:
  application.yaml: |-
    dev:
      url: http://dev.bar.com
      name: Developer Setup
    prod:
      url: http://foo.bar.com
      name: My Cool App
kind: ConfigMap
metadata:
  creationTimestamp: "2024-03-06T17:34:15Z"
  name: app-config
  namespace: default
  resourceVersion: "66409"
  uid: fef3e606-5271-475f-b389-3114aee75c44
brahim@Training:~/lab07-config-data$
brahim@Training:~/lab07-config-data$
```

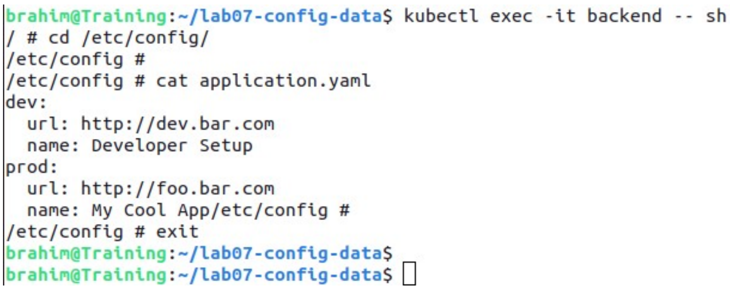
3. Create a Pod named `backend` that consumes the ConfigMap as Volume at the mount path `/etc/config`. The container runs the image `nginx:1.23.4-alpine`.



```
brahim@Training:~/lab07-config-data$ kubectl run backend --image=nginx:1.23.4-alpine -o yaml --dry-run=client --restart=Never > pod.yaml
brahim@Training:~/lab07-config-data$ cat pod.yaml
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: backend
  name: backend
spec:
  containers:
  - image: nginx:1.23.4-alpine
    name: backend
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Never
status: {}
brahim@Training:~/lab07-config-data$
```

```
brahim@Training:~/lab07-config-data$ vim pod.yaml
brahim@Training:~/lab07-config-data$ cat pod.yaml
apiVersion: v1
kind: Pod
metadata:
  labels:
    run: backend
  name: backend
spec:
  containers:
  - image: nginx:1.23.4-alpine
    name: backend
    volumeMounts:
    - name: config-volume
      mountPath: /etc/config
  volumes:
  - name: config-volume
    configMap:
      name: app-config
brahim@Training:~/lab07-config-data$
brahim@Training:~/lab07-config-data$ kubectl apply -f pod.yaml
pod/backend created
brahim@Training:~/lab07-config-data$
```

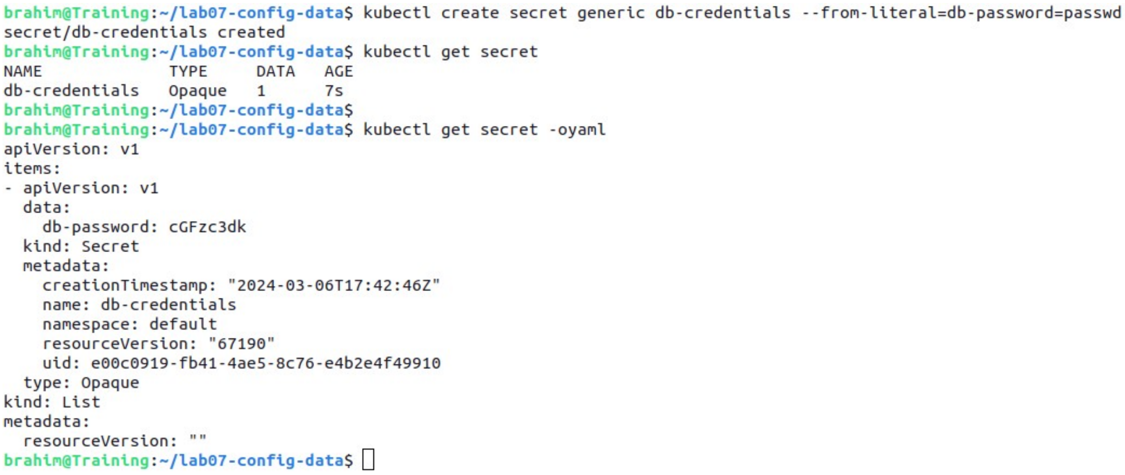
4. Shell into the Pod and inspect the file at the mounted Volume path. You must find the file `application.yaml` with the expected YAML content.



```
brahim@Training:~/lab07-config-data$ kubectl exec -it backend -- sh
/ # cd /etc/config/
/etc/config #
/etc/config # cat application.yaml
dev:
  url: http://dev.bar.com
  name: Developer Setup
prod:
  url: http://foo.bar.com
  name: My Cool App/etc/config #
/etc/config # exit
brahim@Training:~/lab07-config-data$
brahim@Training:~/lab07-config-data$
```

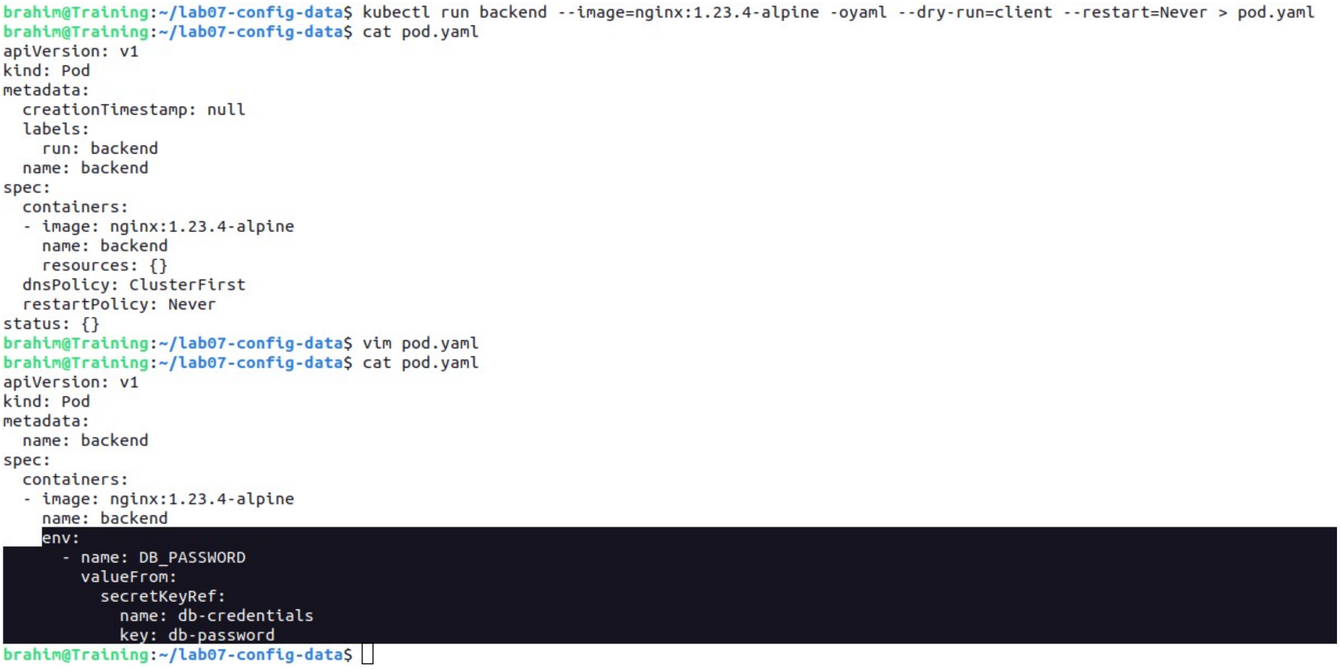
Secret

5. Create a new Secret named `db-credentials` with the key/value pair `db-password=passwd`.

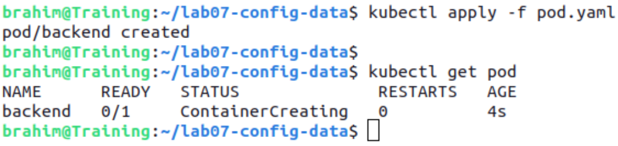


```
brahim@Training:~/lab07-config-data$ kubectl create secret generic db-credentials --from-literal=db-password=passwd
secret/db-credentials created
brahim@Training:~/lab07-config-data$ kubectl get secret
NAME             TYPE     DATA   AGE
db-credentials   Opaque   1      7s
brahim@Training:~/lab07-config-data$
brahim@Training:~/lab07-config-data$ kubectl get secret -oyaml
apiVersion: v1
items:
- apiVersion: v1
  data:
    db-password: cGFzc3dk
  kind: Secret
  metadata:
    creationTimestamp: "2024-03-06T17:42:46Z"
    name: db-credentials
    namespace: default
    resourceVersion: "67190"
    uid: e00c0919-fb41-4ae5-8c76-e4b2e4f49910
  type: Opaque
kind: List
metadata:
  resourceVersion: ""
brahim@Training:~/lab07-config-data$
```

6. Create a Pod named `backend` that uses the Secret as environment variable named `DB_PASSWORD` and runs the container with the image `nginx:1.23.4-alpine`.

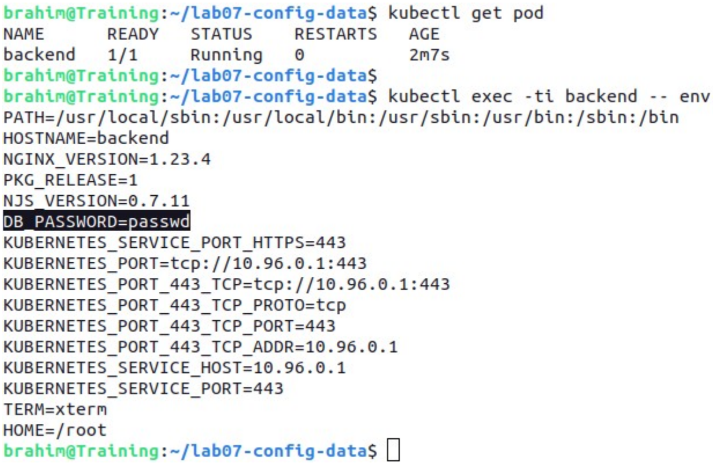


```
brahim@Training:~/lab07-config-data$ kubectl run backend --image=nginx:1.23.4-alpine -oyaml --dry-run=client --restart=Never > pod.yaml
brahim@Training:~/lab07-config-data$ cat pod.yaml
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: backend
  name: backend
spec:
  containers:
  - image: nginx:1.23.4-alpine
    name: backend
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Never
status: {}
brahim@Training:~/lab07-config-data$ vim pod.yaml
brahim@Training:~/lab07-config-data$ cat pod.yaml
apiVersion: v1
kind: Pod
metadata:
  name: backend
spec:
  containers:
  - image: nginx:1.23.4-alpine
    name: backend
    env:
      - name: DB_PASSWORD
        valueFrom:
          secretKeyRef:
            name: db-credentials
            key: db-password
brahim@Training:~/lab07-config-data$ 
```



```
brahim@Training:~/lab07-config-data$ kubectl apply -f pod.yaml
pod/backend created
brahim@Training:~/lab07-config-data$
brahim@Training:~/lab07-config-data$ kubectl get pod
NAME      READY   STATUS              RESTARTS   AGE
backend   0/1     ContainerCreating   0          4s
brahim@Training:~/lab07-config-data$ 
```

7. Shell into the Pod and print out the created environment variables. You should find `DB_PASSWORD` variable.



```
brahim@Training:~/lab07-config-data$ kubectl get pod
NAME      READY   STATUS    RESTARTS   AGE
backend   1/1     Running   0          2m7s
brahim@Training:~/lab07-config-data$
brahim@Training:~/lab07-config-data$ kubectl exec -ti backend -- env
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
HOSTNAME=backend
NGINX_VERSION=1.23.4
PKG_RELEASE=1
NJS_VERSION=0.7.11
DB_PASSWORD=passwd
KUBERNETES_SERVICE_PORT_HTTPS=443
KUBERNETES_PORT=tcp://10.96.0.1:443
KUBERNETES_PORT_443_TCP=tcp://10.96.0.1:443
KUBERNETES_PORT_443_TCP_PROTO=tcp
KUBERNETES_PORT_443_TCP_PORT=443
KUBERNETES_PORT_443_TCP_ADDR=10.96.0.1
KUBERNETES_SERVICE_HOST=10.96.0.1
KUBERNETES_SERVICE_PORT=443
TERM=xterm
HOME=/root
brahim@Training:~/lab07-config-data$ 
```