ConfigMaps

Ref - https://kubernetes.io/docs/concepts/configuration/configmap/

A ConfigMap is an API object used to store non-confidential data in key-value pairs. Pods can consume ConfigMaps as environment variables, command-line arguments, or as configuration files in a volume.

A ConfigMap allows you to decouple environment-specific configuration from your container images, so that your applications are easily portable.

Creating a ConfigMap

- Create the manifest

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: ecom-backend-config
data:
  database_url: "mysql://ecom-db:3306/shop"
  cache_size: "1000"
  payment_gateway_url: "https://payment-gateway.example.com"
  max_cart_items: "50"
  session_timeout: "3600"
```

- Apply the manifest

```
kubectl apply -f cm.yml
```

- Get the configmap

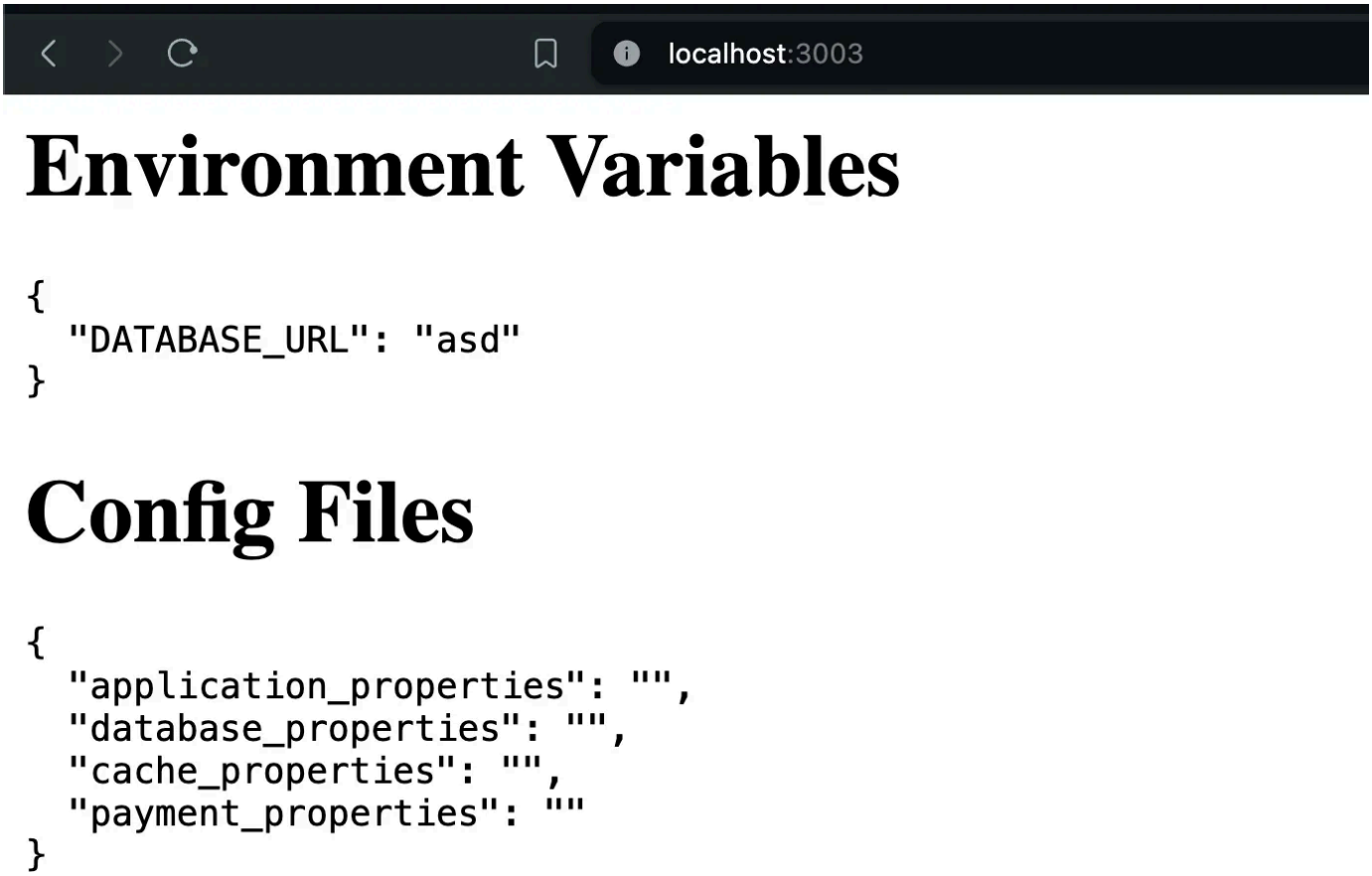
```
kubectl describe configmap ecom-backend-config
```

C…nat exposes env variables

- Express app code
- Deploy to dockerhub - https://hub.docker.com/repository/docker/100xdevs/env-backend/general

Trying the express app using docker locally

```
docker run -p 3003:3000 -e DATABASE_URL=asd  100xdevs/env-backend
```



Try running using k8s locally

- Create the manifest (express-app.yml)

```
apiVersion: apps/v1
kind: Deployment
metadata:
```

```
  replicas: 1
  selector:
    matchLabels:
      app: ecom-backend
  template:
    metadata:
      labels:
        app: ecom-backend
    spec:
      containers:
      - name: ecom-backend
        image: 100xdevs/env-backend
        ports:
        - containerPort: 3000
        env:
        - name: DATABASE_URL
          valueFrom:
            configMapKeyRef:
              name: ecom-backend-config
              key: database_url
        - name: CACHE_SIZE
          valueFrom:
            configMapKeyRef:
              name: ecom-backend-config
              key: cache_size
        - name: PAYMENT_GATEWAY_URL
          valueFrom:
            configMapKeyRef:
              name: ecom-backend-config
              key: payment_gateway_url
        - name: MAX_CART_ITEMS
          valueFrom:
            configMapKeyRef:
              name: ecom-backend-config
              key: max_cart_items
        - name: SESSION_TIMEOUT
          valueFrom:
            configMapKeyRef:
              name: ecom-backend-config
              key: session_timeout
```

```
kubectl apply -f express-app.yml
```

- Create the service (express-service.yml)

```
apiVersion: v1
kind: Service
metadata:
  name: ecom-backend-service
spec:
  type: NodePort
  selector:
    app: ecom-backend
  ports:
    - port: 3000
      targetPort: 3000
      nodePort: 30007
```

- Apply the service

```
kubectl apply -f express-service.yml
```

- Try visiting the website

Environment Variables


```
{
  "DATABASE_URL": "mysql://ecom-db:3306/shop",
  "CACHE_SIZE": "1000",
  "PAYMENT_GATEWAY_URL": "https://payment-gateway.example.com",
  "MAX_CART_ITEMS": "50",
  "SESSION_TIMEOUT": "3600"
}
```


Config Files


```
{
  "application_properties": "app.name=ecom-backend\napp.environment=production\nlogging.level=INFO\nmax.connections=100\n",
  "database_properties": "db.driverClassName=com.mysql.cj.jdbc.Driver\ndb.username=ecom_user\ndb.password=securepassword\ndb.maxPoolSize=20\n",
  "cache_properties": "cache.type=redis\ncache.host=redis-cache\ncache.port=6379\ncache.ttl=600\n",
  "payment_properties": "gateway.url=https://payment-gateway.example.com\ngateway.apiKey=your_api_key_here\ngateway.timeout=30\n"
}
```