kura labs

Kubernetes!!

Objective:

Create an application environment with K3D.

Task: Complete the list of tasks below.

Task 1:

- Create your cluster with a loadbalancer with port mapping “8088:8081” via K3d and then create a Mongo deployment yaml file with a service.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongodb-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mongodb
  template:
    metadata:
      labels:
        app: mongodb
    spec:
      containers:
        - name: mongodb
          image: mongo
          ports:
            - containerPort: 27017
          env:
          - name: MONGO_INITDB_ROOT_USERNAME
            valueFrom:
              secretKeyRef:
                name: mongodb-secret
                key: mongo-root-username
          - name: MONGO_INITDB_ROOT_PASSWORD
            valueFrom:
              secretKeyRef:
                name: mongodb-secret
                key: mongo-root-password
---
apiVersion: v1
kind: Service
metadata:
  name: mongo-service
spec:
  selector:
    app: mongodb
  ports:
    - protocol: TCP
      port: 27017
      targetPort: 27017
```

```

```

- Take note of the container port, root username and password field.

Task 2:

- Now it's time to create a secret yaml file for your username and password
- You will need to open WSL (Windows) or Mac terminal and use this command to encode your string to base64:

```
$echo -n example_username | base64
```

- Run the same command for the password and copy and paste the encoded username and password in a text doc.
- Now enter the base64 encoded string into the username and password field

```
apiVersion: v1
kind: Secret
metadata:
    name: mongodb-secret
type: Opaque
data:
    mongo-root-username: {base64 encoded string}
    mongo-root-password: {base64 encoded string}
```

Task 3:

- Time to deploy the mongo database but first you must create the secret yaml file with:

```
kubectl apply -f {secret_filename}
```

- Now you can create the deployment

```
kubectl apply -f {deployment_filename}
```

- Take note of what type of service the container is using

Task 4:

- Time to create a config_map file that will allow your application to connect to your database:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: mongodb-configmap
data:
  database_url: mongo-service
```

- Take note of the database_url
- Next it's time to create the mongo-express application deployment:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongoexp-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mongo-express
  template:
    metadata:
      labels:
        app: mongo-express
    spec:
      containers:
        - name: mongo-express
          image: mongo-express
          ports:
            - containerPort: 8081
          env:
          - name: ME_CONFIG_MONGODB_ADMINUSERNAME
            valueFrom:
              secretKeyRef:
                name: mongodb-secret
```

```
                key: mongo-root-username
          - name: ME_CONFIG_MONGODB_ADMINPASSWORD
            valueFrom:
              secretKeyRef:
                name: mongodb-secret
                key: mongo-root-password
          - name: ME_CONFIG_MONGODB_SERVER
            valueFrom:
              configMapKeyRef:
                name: mongodb-configmap
                key: database_url
---
apiVersion: v1
kind: Service
metadata:
  name: mongo-exp-service
spec:
  selector:
    app: mongo-express
  type: LoadBalancer
  ports:
    - protocol: TCP
      port: 8081
      targetPort: 8081
```

- A few things to take note of:
  - The env section of this file and the mongodb file.
  - The names and keys.
  - You will have to choose the service (loadbalancer), target port (8081) and port for the service section (8081).

- First create the configmap with:

```
kubectl apply -f {configmap_filename}
```

- Now you can create the mongo-express deployment application

Task 5:

- Now access the mongo-express application