Task02-K8s

OverView:-

1. Integrate MongoDB with MongoDB-Express
2. Deploy it on a K8s cluster
3. Secure the instance with authentication

Steps:-

Step 1: Prepare the working space

1.1. Create a Directory called Zerosploit

```
mkdir ~/zerosploit
```

1.2. Create all files needed for the application

##Create a mongoex-deployment.yaml file to specify mongo-express deployment needed

```
Vim mongoex-deployment.yaml
```

##Create a mongoex-clusterip.yaml file to specify mongo-express service (ClusterIP) needed to be accessible

```
Vim mongoex-clusterip.yaml
```

##Create an ingress.yaml file to specify the ingress service for the mongo-express deployment

```
vim ingress.yaml
```

##Create a mongo-statefulset.yaml file to specify mongo Statefulset component, As Statefulset is more suitable for database pods as it provides the sticky identity feature needed for database configurations

```
Vim mongo-statefulset.yaml
```

##Create a mongoex-headless-svc.yaml file to specify mongo service, which will be headless service, as it's more suitable to work with Statefulset, as it doesn't have the load balancing feature of the ClusterIP service.

```
Vim mongo-headless-svc.yaml
```

##Create a mongo-pvc.yaml file to specify the persistentvolumeclaim needed for the database. The environment has already StorageClass configured for rook-cephfs

```
vim mongo-pvc.yaml
```

##Create a configmap.yaml file to specify the base_url to connect the mongo-express and mongodb applications

```
vim configmap.yaml
```

##Create a secrets.yaml file to specify the username and password of the user

```
vim secrets.yaml
```

Step 2: Write the mongoex-deployment.yaml file

```
apiVersion: apps/v1

kind: Deployment

metadata:

  name: mongoex-deployment
```

```
  labels:
    app: mongoex-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
    app: mongoex-deployment
  template:
    metadata:
    labels:
    app: mongoex-deployment
    spec:
    containers:
    - name: mongoex
          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_URL
                #   value: mongodb://mongo-headless-svc:27017
                valueFrom:
                configMapKeyRef:
                name: mongodb-configmap
                key: database_url
```

This will create a deployment for the mongo-express pods

Step 3: Write the ingress.yaml file

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: mongoex-ingress
  annotations:
     nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  ingressClassName: nginx
  rules:
  - host: mongoex.com
     http:
     paths:
```

```
    - path: /
    pathType: Prefix
    backend:
          service:
          name: mongoex-clusterip
          port:
                number: 8081
```

This will create a ingres service with a domain called (mongoex.com)

ingress uses the internal service (ClusterIP)

ingress external IP is assigned by a load balancer

Step 4: Write the mongoex-clusterip.yaml file

```
apiVersion: v1
kind: Service
metadata:
  name: mongoex-clusterip
spec:
  selector:
    app: mongoex-deployment
  ports:
    - protocol: TCP
    port: 8081
    targetPort: 8081
```

This will create ClusterIP service for the mongo-express pods

Step 5: Write the mongo-statefulset.yaml file

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mongo-statefulset
  labels:
     app: mongo-pod
spec:
  serviceName: "mongo-headless-svc"
  replicas: 1
  selector:
     matchLabels:
     app: mongo-pod
  template:
     metadata:
     labels:
     app: mongo-pod
     spec:
     containers:
     - name: mongo
     image: mongo:latest
     ports:
     - containerPort: 27017
```

```
    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
    volumeMounts:
    - name: mongo-data
          mountPath: /data/db
    volumes:
    - name: mongo-data
          persistentVolumeClaim:
          claimName: mongo-pvc
```

This will create the mongodb Statefulset component

Step 6: Write the mongo-headless-svc.yaml file

```
apiVersion: v1
kind: Service
metadata:
```

```
  name: mongo-headless-svc
  labels:
      app: mongo-headless-svc
spec:
  ports:
      - name: mongo-port
      port: 27017
      targetPort: 27017
  clusterIP: None
  selector:
      app: mongo-pod
```

This will create Headless service for the mongodb pods, which is more suitable to work with Statefulsets

Step 7: Write the mongo-pvc.yaml file

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mongo-pvc
spec:
  storageClassName: rook-cephfs
  accessModes:
      - ReadWriteMany
  resources:
      requests:
```

```
      storage: 10Gi
```

This will create persistentvolumeclaim for the mongodb pods (Statefulset)

There is already a Storage Class created with the name of “ rook-cephfs ”

Step 8: Write the configmap.yaml file

```
apiVersion: v1

kind: ConfigMap

metadata:

  name: mongodb-configmap

data:

  database_url: "mongodb://mongo-headless-svc:27017"
```

This will specify the “database_url” for connecting mongodb and mongo-express

The reason why i used the entire connection string “`mongodb://mongo-headless-svc:27017`” instead of only the name of the service “mongo-headless-svc” is to provide a complete and directly usable connection string for applications or services that need to connect to the MongoDB database.

Step 9: Write the secrets.yaml file

```
apiVersion: v1

kind: Secret

metadata:

  name: mongodb-secret

type: Opaque

data:

  mongo-root-username: YWRtaW4=
```

```
  mongo-root-password: cGFzc3dvcmQ=
```

This will create a secrets file to store your root username and password

But before you put the username and password you need to encrypt them bu (base64) and not put them in plain-text or it will end with an error

```
mohamedhassan@master:~/task-04$ kubectl apply -f secrets.yaml
Error from server (BadRequest): error when creating "secrets.yaml": Secret in version "v1" cannot be handled as a Secret: illegal base64 dat
a at input byte 2
```

So it's better to encrypt them by:

```
echo "root" | base64
```

```
echo "Pa$$w0rd" | base64
```

```
mohamedhassan@master:~/task-04$ echo "root" | base64
cm9vdAo=
mohamedhassan@master:~/task-04$ echo "Pa$$w0rd" | base64
UGEyMTQ0NDg5dzByZAo=
```

Step 10: Apply your manifests

10.1. Run the deployment.yaml file

```
Kubectl apply -f deployment.yaml
```

```
Kubectl get deployments
```

```
mohamedhassan@master:~/task-04$ kubectl apply -f mongoex-deployment.yaml
deployment.apps/mongoex-deployment created
mohamedhassan@master:~/task-04$ kubectl get deployments.apps
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
mongoex-deployment   1/1     1            1           6s
```

10.2. Run the ingress.yaml file

```
kubectl apply -f ingress.yaml
```

```
kubectl get ingress
```

```
mohamedhassan@master:~/task-04$ kubectl apply -f ingress.yaml
ingress.networking.k8s.io/mongoex-ingress created
mohamedhassan@master:~/task-04$ kubectl get ingress
NAME              CLASS   HOSTS         ADDRESS         PORTS   AGE
mongoex-ingress   nginx   mongoex.com   198.96.95.206   80      93s
```

10.3. Run the mongoex-clusterip.yaml file

```
kubectl apply -f mongoex-clusterip.yaml
```

```
kubectl get service
```

```
mohamedhassan@master:~/task-04$ kubectl apply -f mongoex-clusterip.yaml
service/mongoex-clusterip created
mohamedhassan@master:~/task-04$ kubectl get service
NAME                TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)     AGE
mongo-headless-svc  ClusterIP   None            <none>        27017/TCP   147m
mongoex-clusterip   ClusterIP   10.111.75.252   <none>        8081/TCP    11s
```

10.4. Run the mongo-statefulset.yaml file

```
kubectl apply -f mongo-statefulset.yaml
```

```
kubectl get statefulsets.apps
```

```
mohamedhassan@master:~/task-04$ kubectl apply -f mongo-statefulset.yaml
statefulset.apps/mongo-statefulset created
mohamedhassan@master:~/task-04$ kubectl get statefulsets.apps
NAME              READY   AGE
mongo-statefulset   0/1     5s
mohamedhassan@master:~/task-04$ kubectl get statefulsets.apps
NAME              READY   AGE
mongo-statefulset   1/1     9s
```

10.5. Run the mongo-headless-svc.yaml file

```
kubectl apply -f mongo-headless-svc.yaml
```

```
kubectl get service
```

```
mohamedhassan@master:~/task-04$ kubectl apply -f mongo-headless-svc.yaml
service/mongo-headless-svc created
mohamedhassan@master:~/task-04$ kubectl get service
serviceaccounts              servicel2statuses.metallb.io  services
mohamedhassan@master:~/task-04$ kubectl get services mongo-headless-svc -owide
NAME                TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)     AGE   SELECTOR
mongo-headless-svc  ClusterIP   None         <none>        27017/TCP   26s   app=mongo-pod
```

10.6. Run the mongo-pvc.yaml file

```
kubectl apply -f mongo-pvc.yaml
```

```
kubectl get pvc
```

```
mohamedhassan@master:~/task-04$ kubectl apply -f mongo-pvc.yaml
persistentvolumeclaim/mongo-pvc created
mohamedhassan@master:~/task-04$ kube
kubeadm  kubectl  kubelet
mohamedhassan@master:~/task-04$ kubectl get pvc
NAME        STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS      VOLUMEATTRIBUTESCLASS   AGE
mongo-pvc   Bound    pvc-c48e9de9-4de2-427e-ba65-c13456ccbfca   10Gi       RWX            rook-cephfs       <unset>                 20s
```

10.7. Run the configmap.yaml file

```
kubectl apply -f configmap.yaml
```

```
kubectl get configmaps
```

```
mohamedhassan@master:~/task-04$ kubectl apply -f configmap.yaml
configmap/mongodb-configmap created
mohamedhassan@master:~/task-04$ kubectl get con
configmaps                controllerrevisions.apps
mohamedhassan@master:~/task-04$ kubectl get configmaps
NAME               DATA   AGE
kube-root-ca.crt   1      5d
mongodb-configmap  1      13s
```

10.8. Run the secrets.yaml file

```
kubectl apply -f secrets.yaml
```

```
kubectl get secrets
```

```
mohamedhassan@master:~/task-04$ kubectl apply -f secrets.yaml
secret/mongodb-secret created
mohamedhassan@master:~/task-04$ kubectl get secrets
NAME             TYPE     DATA   AGE
mongodb-secret   Opaque   2      11s
```

Step 11: Add the Domain name to your local hosts

11.1. Edit the hosts file on your local system

```
sudo Vim /etc/hosts
```

11.2. Add the new domain to the hosts

```
198.96.95.206 mongoex.com
```

```
127.0.0.1 localhost
127.0.1.1 mohamed-Inspiron-3581

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
198.96.95.206 mongoex.com
```

Step 12: Test to see if the architecture is working

Go to your local browser and search for (mongoex.com)