Task9.1P

Jiakun Li *sit323*

223313972

Instructions:

Define a StorageClass (createStorageClass.yaml): This is used to define the storage type and manage policies for persistent volumes.

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: demo-storage
provisioner: docker.io/hostpath
volumeBindingMode: Immediate
reclaimPolicy: Delete
```

Create a PersistentVolume (createPersistentVolume.yaml), specifying the storage capacity, access mode, and physical storage path.

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: mongo-pv
spec:
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteMany
  local:
    path: /run/desktop/mnt/host/C:\Sam\DEAKIN\T1_24\SIT323 D\Task 9.1
  nodeAffinity:
    required:
      nodeSelectorTerms:
        - matchExpressions:
            - key: kubernetes.io/hostname
              operator: In
              values:
                - docker-desktop
```

Next, create a PersistentVolumeClaim (createPersistentVolumeClaim.yaml) that requests storage resources matching the PersistentVolume.

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mongo-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: "demo-storage"
```

The deployment file (createDeployment.yaml) sets up a single replica for MongoDB, binds the previously created PVC, and sets environment variables for initializing the database user.

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongo
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mongo
  template:
    metadata:
      labels:
        app: mongo
    spec:
      containers:
        - image: mongo
          name: mongo
          args: ["--dbpath", "/data/db"]
          env:
            - name: MONGO_INITDB_ROOT_USERNAME
              value: "admin"
            - name: MONGO_INITDB_ROOT_PASSWORD
              value: "password"
          volumeMounts:
            - mountPath: /data/db
              name: mongo-volume
      volumes:
        - name: mongo-volume
          persistentVolumeClaim:
            claimName: mongo-pvc
```

Finally, define a Service (createService.yaml) to expose MongoDB's port within the cluster, allowing other applications to connect to the database.

```yaml
apiVersion: v1
kind: Service
metadata:
  name: mongo-svc
spec:
  ports:
    - port: 27017
      protocol: TCP
      targetPort: 27017
      nodePort: 32000
  selector:
    app: mongo
  type: NodePort
```

Verification and Testing:

Ensure all the YAML files are correctly deployed.

Use kubectl get pods to check if the MongoDB Pod's status is Running.

Use kubectl get svc to check if the service correctly exposes port 27017.

Use a MongoDB client to test the connection to the MongoDB instance.

Configuration and Deployment of MongoDB:

Step 1: Create StorageClass

First, create the StorageClass:

Step 2: Create PersistentVolume

Create the PersistentVolume:

Step 3: Create PersistentVolumeClaim

Create the PersistentVolumeClaim:

Step 4: Deploy MongoDB

Apply the deployment file:

Step 5: Create Service

Create the Service to expose MongoDB

```
C:\Users\12771>cd C:\Sam\DEAKIN\T1_24\SIT323 D\Task 9.1P

C:\Sam\DEAKIN\T1_24\SIT323 D\Task 9.1P>kubectl apply -f ./createPersistentVolume.yaml
persistentvolume/mongo-pv unchanged

C:\Sam\DEAKIN\T1_24\SIT323 D\Task 9.1P>kubectl apply -f ./createPersistentVolumeClaim.yaml
persistentvolumeclaim/mongo-pvc unchanged

C:\Sam\DEAKIN\T1_24\SIT323 D\Task 9.1P>kubectl apply -f ./createStorageClass.yaml
storageclass.storage.k8s.io/demo-storage unchanged

C:\Sam\DEAKIN\T1_24\SIT323 D\Task 9.1P>kubectl apply -f ./createService.yaml
service/mongo-svc unchanged

C:\Sam\DEAKIN\T1_24\SIT323 D\Task 9.1P>kubectl apply -f ./createDeployment.yaml
deployment.apps/mongo unchanged
```

Step 6: Verify the Deployment:

a). Verify the MongoDB Pod is running

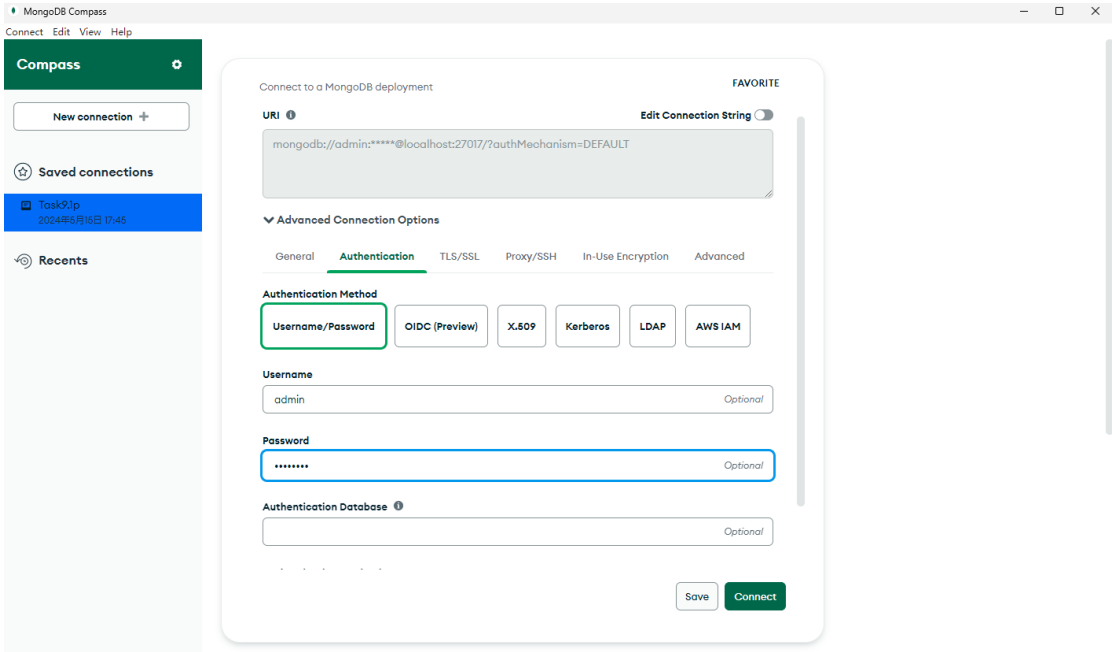
```
C:\Sam\DEAKIN\T1_24\SIT323 D\Task 9.1P>kubectl get pods
NAME                                        READY   STATUS             RESTARTS        AGE
mongo-54f7c77856-xr5wf                      1/1     Running            1 (86m ago)     106m
mongodb-enterprise-operator-6dcd58f895-l5rcz   1/1     Running            2 (86m ago)     114m
nginx-deployment-5cd667c897-pvm4v           0/1     ImagePullBackOff   0               16d
nginx-deployment-7c79c4bf97-8gpjh           1/1     Running            6 (86m ago)     16d
task6-6b56f87c56-q9rf9                      1/1     Running            5 (86m ago)     16d
task9-787f5d6645-5bck6                      0/1     CrashLoopBackOff   68 (3m13s ago)  23h
```

b). Verify the Service exposes the port

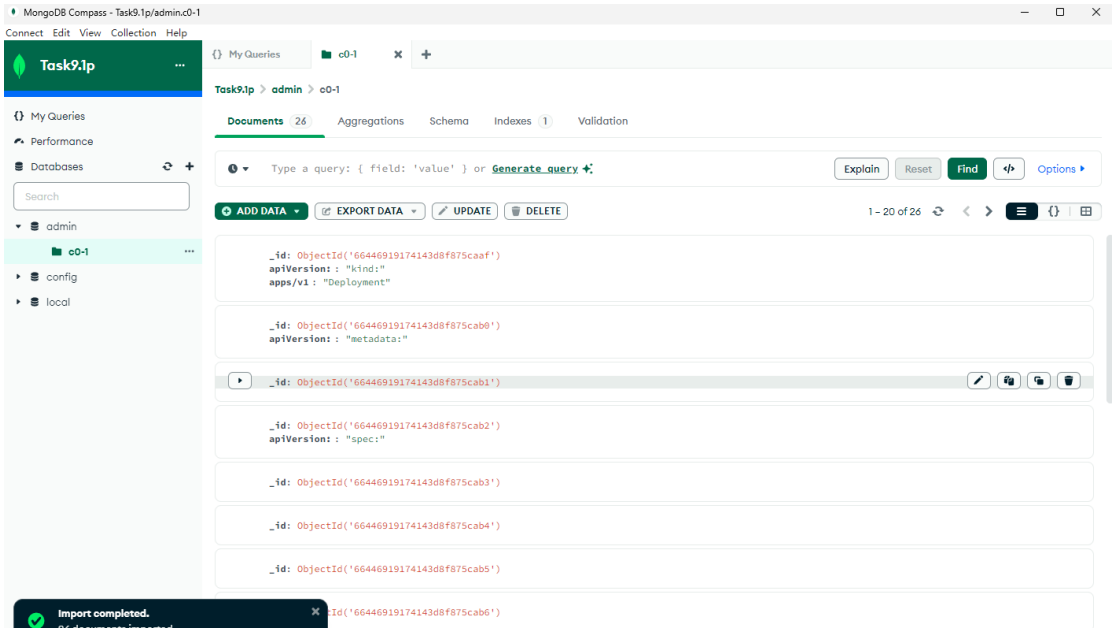
```
C:\Sam\DEAKIN\T1_24\SIT323 D\Task 9.1P>kubectl get svc
NAME               TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)           AGE
kubernetes         ClusterIP      10.96.0.1        <none>        443/TCP           17d
mongo-svc          NodePort       10.102.44.169    <none>        27017:32000/TCP   24h
my-mongodb         ClusterIP      10.108.66.27     <none>        27017/TCP         42h
my-mongodb-1       ClusterIP      10.110.236.11    <none>        27017/TCP         41h
nginx-service      ClusterIP      10.108.156.226   <none>        80/TCP            16d
nodejs-service     LoadBalancer   10.100.151.217   localhost     80:31787/TCP      16d
operator-webhook   ClusterIP      10.101.170.141   <none>        443/TCP           40h
```

And we can see the mongo-svc has been deployed.

Step 7: Configure the Application to Connect to MongoDB



Step 8: Testing CRUD



Documents 24 Aggregations Schema Indexes 1 Validation

Type a query: { field: 'value' } or Generate query

Explain Reset Find Options

ADD DATA EXPORT DATA UPDATE DELETE

1 - 18 of 24

```
_id: ObjectId('66446919174143d8f875cab1')

1 _id: ObjectId('66446919174143d8f875cab2')                ObjectId
  apiVersion: "spec:/"                                     String
```

Document modified. CANCEL UPDATE

```
_id: ObjectId('66446919174143d8f875cab3')
```