https://github.com/DennisPYu/sit323-2024-t1-prac9p

Install MongoDB into Kubernetes cluster

createStatefulSet.yaml > { } spec > { } template > { } spec > [ ] containers > { } 0 > { } startupProbe > # timeoutSeconds

```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mongo
spec:
  selector:
    matchLabels:
      app: mongo
  serviceName: "mongo"
  replicas: 3
  template:
    metadata:
      labels:
        app: mongo
    spec:
      containers:
        - name: mongo
          image: mongo:4.0.8
          startupProbe:
            exec:
              command:
                - mongo
                - --eval
                - "db.adminCommand('ping')"
            initialDelaySeconds: 1
            periodSeconds: 10
            timeoutSeconds: 5
            successThreshold: 1
            failureThreshold: 2
          livenessProbe:
            exec:
              command:
                - mongo
                - --eval
                - "db.adminCommand('ping')"
            initialDelaySeconds: 1
            periodSeconds: 10
            timeoutSeconds: 5
            successThreshold: 1
            failureThreshold: 2
          readinessProbe:
            exec:
              command:
                - mongo
                - --eval
                - "db.adminCommand('ping')"
            initialDelaySeconds: 1
            periodSeconds: 10
            timeoutSeconds: 5
            successThreshold: 1
            failureThreshold: 2
          env:
            - name: MONGO_INITDB_ROOT_USERNAME
              valueFrom:
```

PROBLEMS OUTPUT TERMINAL PORTS POLYGLOT NOTEBOOK AZURE DEBUG CONSOLE

```
PS I:\Study\BachelorD\2024T1\SIT323-Cloud Native Application Development\9.1p\docker_calculator-upgrade> kubectl apply -f createStatefulSet.yaml
```

```
kubectl apply -f createStatefulSet.yaml
statefulset.apps/mongo configured
PS I:\Study\BachelorD\2024T1\SIT323-Cloud Native Application Development\9.1p\docker_calculator-upgrade>
```

Create a MongoDB user

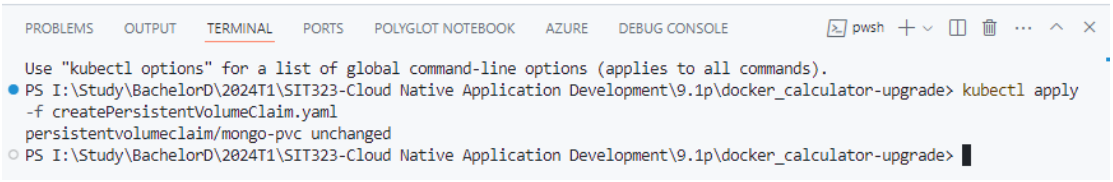
createConfigMap.yaml > { } data > mongodb.conf

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: mongodb-config
immutable: false
data:
  username: admin1
  mongodb.conf: |
    storage:
      dbPath: /data/db
    replication:
      replSetName: "rs0"
```

Create a Persistent Volume

createPersistentVolumeClaim.yaml > ...

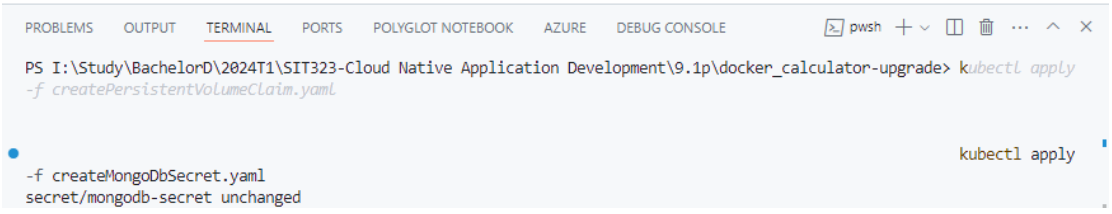
```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mongo-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: "demo-storage"

```



Create a Kubernetes Secret

createMongoDbSecret.yaml > { } data > password

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: mongodb-secret
immutable: false
type: Opaque
data:
  password: cGFzc3dvcmQxMjM=
```



Setting up

```
rs0:SECONDARY> rs.initiate(
...     {
...        _id: "rs0",
...        members: [
...           { _id: 0, host : "mongo-0.mongo.default.svc.cluster.local:27017" },
...           { _id: 1, host : "mongo-1.mongo.default.svc.cluster.local:27017" },
...           { _id: 2, host : "mongo-2.mongo.default.svc.cluster.local:27017" }
...        ]
...     }
... )
{
        "operationTime" : Timestamp(1715620160, 1),
        "ok" : 0,
        "errmsg" : "already initialized",
        "code" : 23,
        "codeName" : "AlreadyInitialized",
        "$clusterTime" : {
                "clusterTime" : Timestamp(1715620160, 1),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        }
}
```

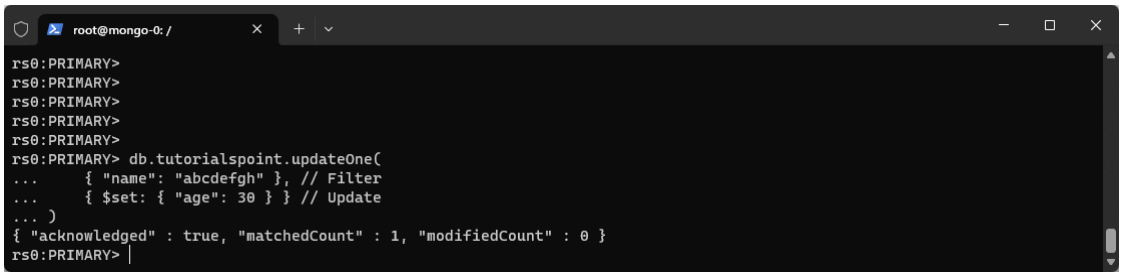
Creating

```
root@mongo-0: /
                        "infoMessage" : "",
                        "configVersion" : 1
                }
        ],
        "ok" : 1,
        "operationTime" : Timestamp(1715615660, 5),
        "$clusterTime" : {
                "clusterTime" : Timestamp(1715615660, 5),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        }
}
rs0:PRIMARY> use test
switched to db test
rs0:PRIMARY> db.createCollection("tutorialspoint")
{
        "ok" : 1,
        "operationTime" : Timestamp(1715615772, 1),
        "$clusterTime" : {
                "clusterTime" : Timestamp(1715615772, 1),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        }
}
rs0:PRIMARY> db.tutorialspoint.insert({"name" : "abcdefgh"})
WriteResult({ "nInserted" : 1 })
rs0:PRIMARY>
```

Reading

```
root@mongo-1: /
---

rs0:SECONDARY> rs.slaveOk()
rs0:SECONDARY> use test
switched to db test
rs0:SECONDARY> db.tutorialspoint.find()
{ "_id" : ObjectId("664238207b1ee7b8316363e1"), "name" : "abcdefgh" }
rs0:SECONDARY>
```

Updating


root@mongo-0: /
rs0:PRIMARY>
rs0:PRIMARY>
rs0:PRIMARY>
rs0:PRIMARY>
rs0:PRIMARY>
rs0:PRIMARY> db.tutorialspoint.updateOne(
...   { "name": "abcdefgh" }, // Filter
...   { $set: { "age": 30 } } // Update
...)
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 0 }
rs0:PRIMARY>

Deleting


root@mongo-0: /
...   { $set: { "age": 30 } } // Update
...)
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 0 }
rs0:PRIMARY>
rs0:PRIMARY>
rs0:PRIMARY>
rs0:PRIMARY>
rs0:PRIMARY>
rs0:PRIMARY> db.tutorialspoint.deleteOne({ "name": "abcdefgh" })
{ "acknowledged" : true, "deletedCount" : 1 }
rs0:PRIMARY>