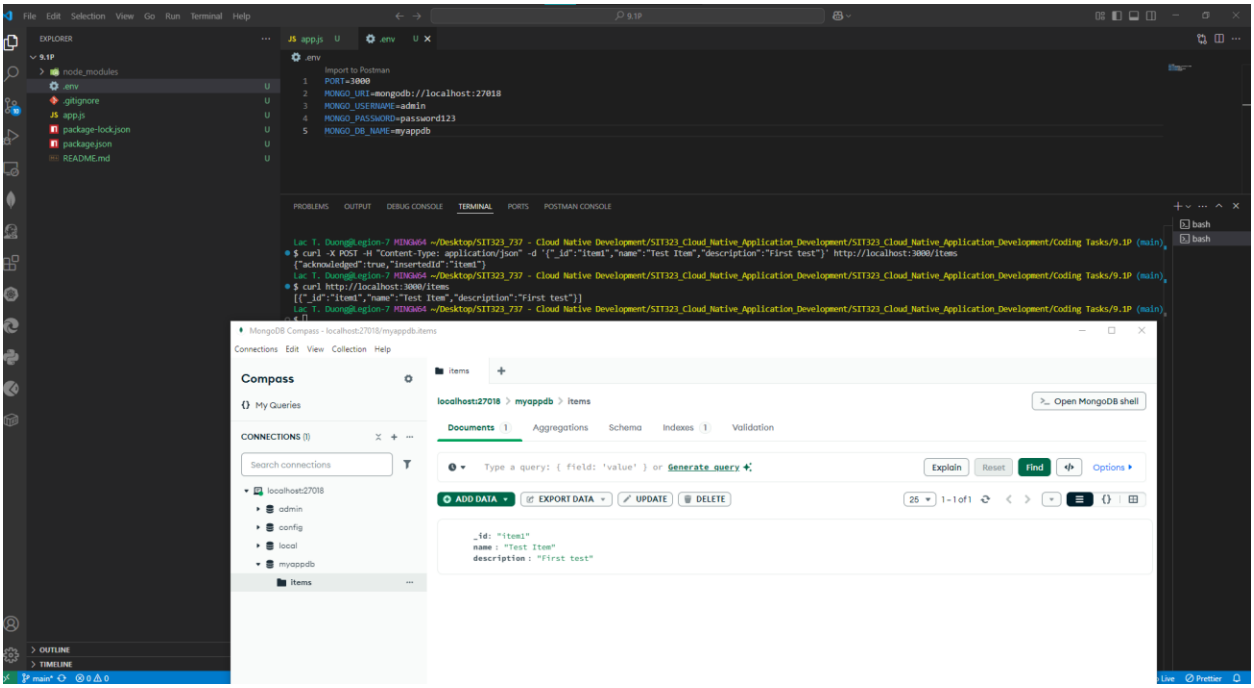
Github :
https://github.com/HaydenDuong/SIT323_Cloud_Native_Application_Development/tree/main/Coding%20Tasks/9.1P

Video Demo:

https://deakin.au.panopto.com/Panopto/Pages/Viewer.aspx?id=b0128c10-18b4-4079-bca5-b2d500ec8c60

Pictures & Explanation:

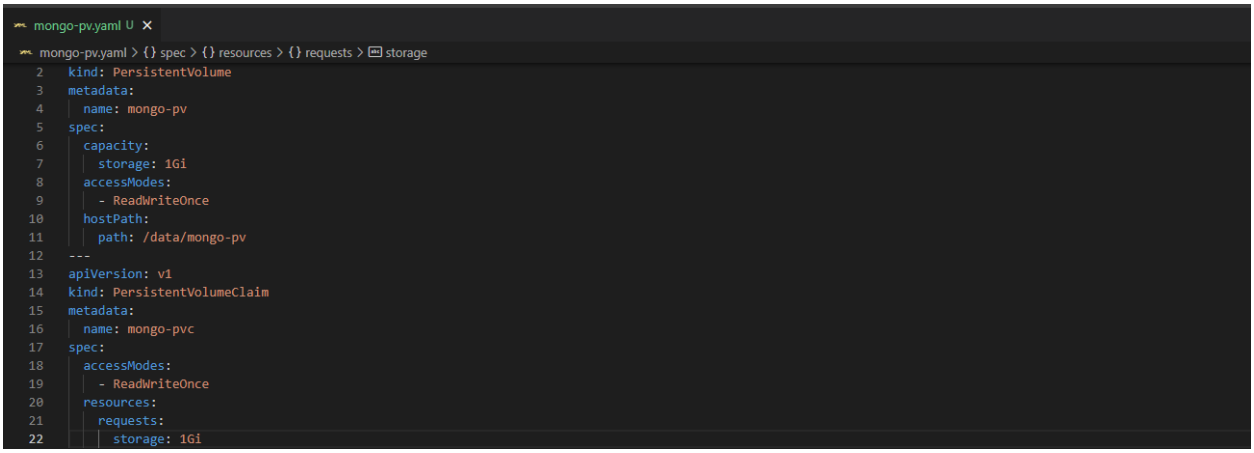
Testing the application to determine whether it is connected to mongoDB or not and to test CRUD endpoints.



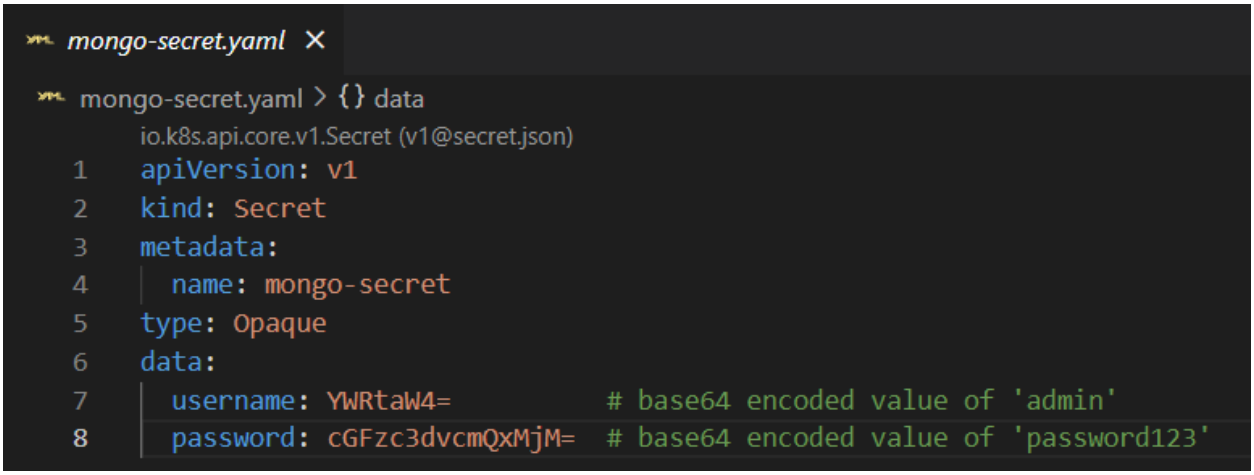
Picture 1 – App.js is connected to MongoDB, shown in MongoDB Compass.

Because this application will simulate how microservice application look like so the application-side and the database-side will have their own YAML files like Deployment, and Service files. As for the database, it will have additional Secret, and Persistent Volume – Persistent Volume Claim files.

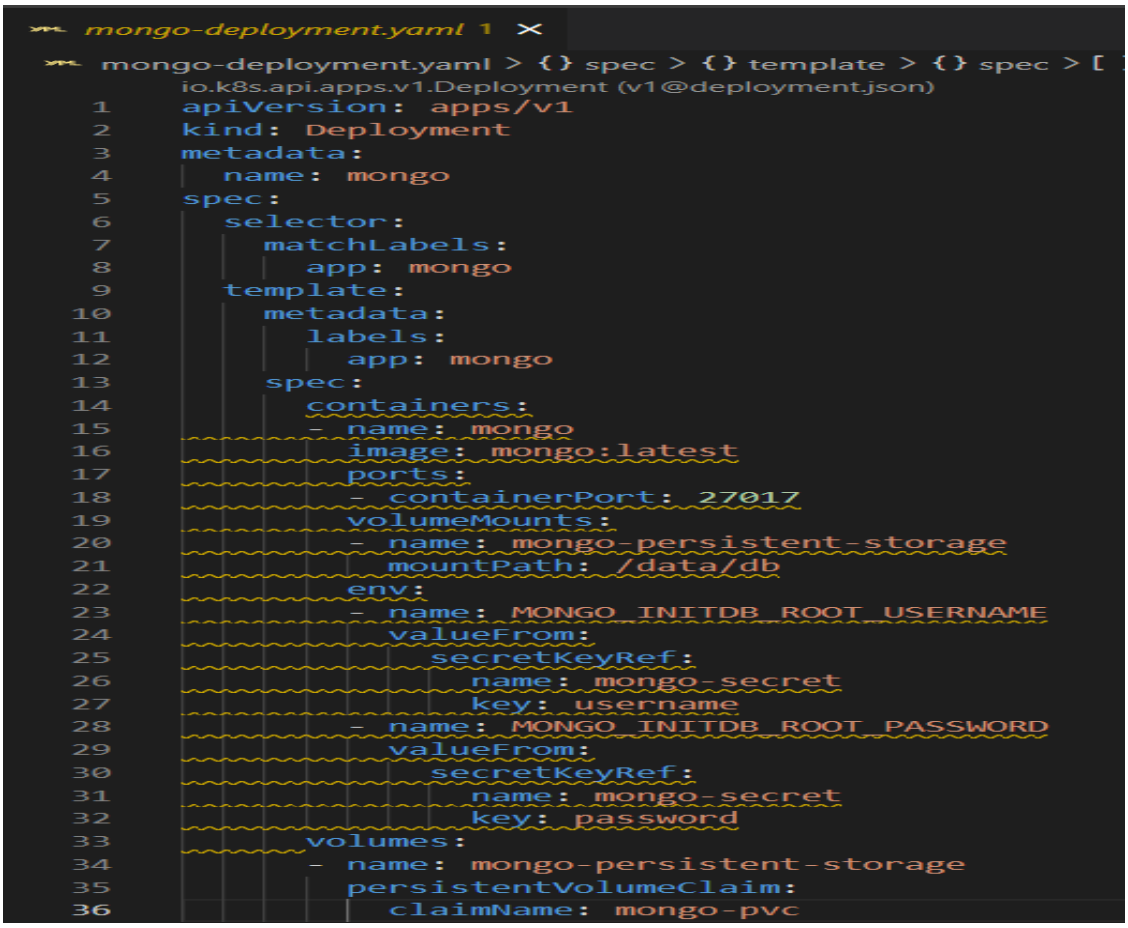
Their contents are shown as follows (or can be accessed publicly through Github link)



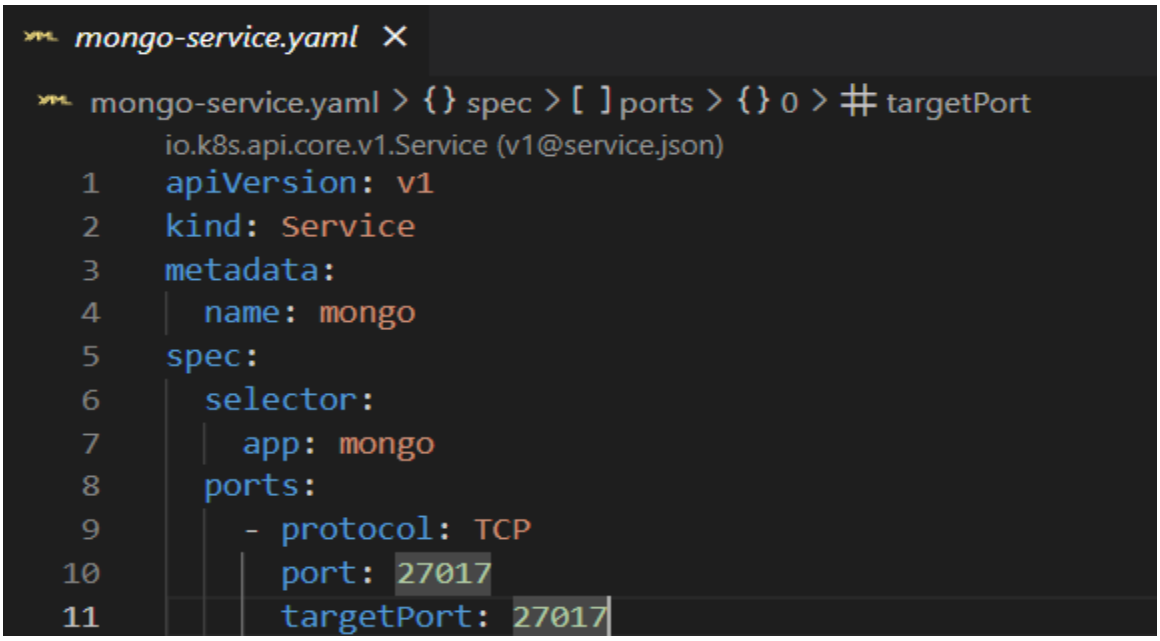
Picture 2 - mongo-pv.yaml



Picture 3 – mongo-secret.yaml



Picture 4 – mongo-deployment.yaml



Picture 5 – mongo-service.yaml

app-deployment.yaml 1

app-deployment.yaml > {} spec > {} template > {} spec > [ ] containers > io.k8s.api.apps.v1.Deployment (v1@deployment.json)

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
      - name: my-app
        image: hayden2310/sit323-task91p:v1
        ports:
        - containerPort: 3000
        env:
        - name: PORT
          value: "3000"
        - name: MONGO_URI
          value: "mongodb://mongo:27017/myappdb"
        - name: MONGO_USERNAME
          valueFrom:
            secretKeyRef:
              name: mongo-secret
              key: username
        - name: MONGO_PASSWORD
          valueFrom:
            secretKeyRef:
              name: mongo-secret
              key: password
```

Picture 6 – app-deployment.yaml

app-service.yaml

app-service.yaml > {} spec > abc type

io.k8s.api.core.v1.Service (v1@service.json)

```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-app
spec:
  selector:
    app: my-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 3000
  type: LoadBalancer
```

Picture 7 - app-service.yaml

After built an image and pushed it to Docker Hub / localhost repository, run “kubeclt apply -f ….yaml” on those YAML files.

The application, by now, can be tested by:

1. Type in command: “kubectl port-forward svc/my-app 8080:80”.
2. Upon received output message like “Forwarding from 127.0.0.1:8080 → 3000”, then open a new terminal and type in command: “kubectl port-forward svc/mongo 27018:27017” – this command will allow user to view the result of CRUD interaction with mongoDB database on mongoDB Compass.
3. Open mongoDB Compass and choose “Add new connection” button and paste this following URI: “mongodb://admin:password123@localhost:27018/”
4. Upon successful connected, open a new terminal in vscode and try the following CRUD command:
    a. For adding a new document item:

curl -X POST -H "Content-Type: application/json" -d '{"_id":"item1","name":"Test Item","description":"Just testing"}' http://localhost:8080/items

    b. For updating an existing document item:

curl -X PUT -H "Content-Type: application/json" -d '{"name":"Updated Item"}' http://localhost:8080/items/item1

    c. Get a specific document item:

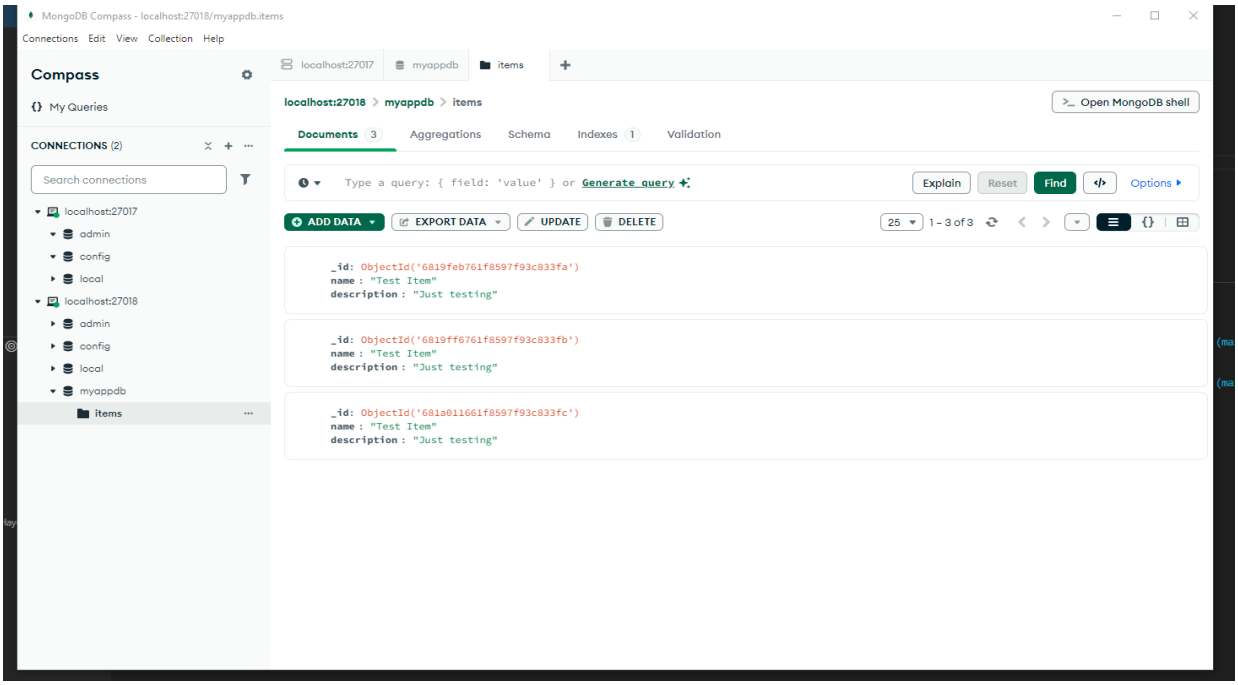
curl http://localhost:8080/items/item1

    d. Get all document items

curl http://localhost:8080/items

    e. Delete a document item:

curl -X DELETE http://localhost:8080/items/item1



Picture 8 – Results after adding three document item.