SCD-Assignment-03

Prepared by: Abad Naseer

Submitted to: Laiba Imran

Submission Date

12 December 2024

Comprehensive Report on Deployment and Management of a Multi-Episode Containerized Application

Assignment Details:

This report documents the deployment and management process for a multi-episode containerized application. The application consists of five services, each with a frontend and backend, developed as separate modules. The deployment was carried out in both Docker and Kubernetes environments as per the assignment's requirements.

Part 1: Individual Deployment (80 Points)

Task Overview

1. **Deploying Individual Modules Locally**
    - Each service's frontend and backend modules were containerized and run locally using Docker.
    - Modules were made accessible and responsive to HTTP requests.

Dockerfile Details

- Each module has its own `Dockerfile` for frontend and backend.

Episode-01:

- **Backend Dockerfile**: Located at `C:\Users\DELL\Downloads\i201815_A3\episode-01\backend\Dockerfile`.

```
PROBLEMS 1   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS                                                    powershell - backend

PS C:\Users\DELL\Downloads\i201815_A3\episode-01\backend> docker build -t episode-01-backend .
>>
[+] Building 123.5s (12/12) FINISHED                                                                                  docker:default
 => [internal] load build definition from Dockerfile                                                                            0.1s
 => => transferring dockerfile: 517B                                                                                            0.0s
 => [internal] load metadata for docker.io/library/node:18                                                                      3.8s
 => [auth] library/node:pull token for registry-1.docker.io                                                                     0.0s
 => [internal] load .dockerignore                                                                                               0.0s
 => => transferring context: 2B                                                                                                 0.0s
 => [1/6] FROM docker.io/library/node:18@sha256:b57ae84fe7880a23b389f8260d726b784010ed470c2ee26d4e2cbdb955d25b12               91.4s
 => => resolve docker.io/library/node:18@sha256:b57ae84fe7880a23b389f8260d726b784010ed470c2ee26d4e2cbdb955d25b12                0.1s
 => => sha256:b57ae84fe7880a23b389f8260d726b784010ed470c2ee26d4e2cbdb955d25b12 6.41kB / 6.41kB                                  0.0s
 => => sha256:9371b74049cb4bb6422e1f25c93f564ce9524a647fc951a35f0945efe3e96d4d 2.49kB / 2.49kB                                  0.0s
 => => sha256:b2da2cb649e9f22d68ecc32c89d6833e6dc1705f0162c3f8a64df5772a470884 6.39kB / 6.39kB                                  0.0s
 => => sha256:551df7f94f9c131f2fec0e8063142411365f0a1c88b935b9fac22be91af227e0 64.39MB / 64.39MB                               10.2s
 => => sha256:5bd71677db44bb63b94de61b6f1f95d5540b4ba2d6a8a6bc4d19f422b25e0c2b 23.87MB / 23.87MB                               22.3s
 => => sha256:fdf894e782a221820acf469d425b802be26aedb5e5d26ea80a650ff6a974d488 48.50MB / 48.50MB                               50.0s
 => => sha256:ce82e98d553dd62ca6a12bebfe83992ae9f9ae2748275e74b66a68cc094f868b 211.31MB / 211.31MB                             50.0s
 => => sha256:6399a464889d3eae2913051cb98c35d0b6bfa20ec77d6b3a04617d4a298a2a56 3.32kB / 3.32kB                                 22.9s
 => => sha256:a3c94c84d15dfc1c2c202acca56d7327f541d62c10f9bc1dfb013a618aebd5f1 45.70MB / 45.70MB                               46.5s
 => => sha256:2cd8c50fd8ca9ed98f596afc5d92d00b4492b7b069d2d339a6ed8682fc568961 1.25MB / 1.25MB                                 47.9s
 => => sha256:247468edfd9afcf43bf96caab52a1d979edd5eb13afcaf570c1513f4a35fa43f 446B / 446B                                     48.4s
 => => extracting sha256:fdf894e782a221820acf469d425b802be26aedb5e5d26ea80a650ff6a974d488                                        4.6s
 => => extracting sha256:5bd71677db44bb63b94de61b6f1f95d5540b4ba2d6a8a6bc4d19f422b25e0c2b                                        1.3s
 => => extracting sha256:551df7f94f9c131f2fec0e8063142411365f0a1c88b935b9fac22be91af227e0                                        6.1s
 => => extracting sha256:ce82e98d553dd62ca6a12bebfe83992ae9f9ae2748275e74b66a68cc094f868b                                       19.5s
 => => extracting sha256:6399a464889d3eae2913051cb98c35d0b6bfa20ec77d6b3a04617d4a298a2a56                                        0.0s
 => => extracting sha256:a3c94c84d15dfc1c2c202acca56d7327f541d62c10f9bc1dfb013a618aebd5f1                                        6.2s
 => => extracting sha256:2cd8c50fd8ca9ed98f596afc5d92d00b4492b7b069d2d339a6ed8682fc568961                                        0.2s
 => => extracting sha256:247468edfd9afcf43bf96caab52a1d979edd5eb13afcaf570c1513f4a35fa43f                                        0.0s
 => [internal] load build context                                                                                               0.1s
 => => transferring context: 1.44kB                                                                                             0.0s
```

Now below is the command to run docker container for the backend image:

```
  View a summary of image vulnerabilities and recommendatidocker run -d -p 8080:8080 --name episode-01-backend  episode-01-backend
>> C:\Users\DELL\Downloads\i201815_A3\episode-01\backend>
fe0fcee67376ad0647f40fc4e14c12b8627d8f338a1b4af6117baaefd87e1f93
PS C:\Users\DELL\Downloads\i201815_A3\episode-01\backend> docker ps -a
CONTAINER ID   IMAGE                                COMMAND                  CREATED          STATUS                      PORTS
                                                                                                                 NAMES
fe0fcee67376   episode-01-backend                   "docker-entrypoint.s…"   24 seconds ago   Up 8 seconds                0.0.0.0:8080->8080/tcp
                                                                                                                 episode-01-backend
7027825f2f2d   gcr.io/k8s-minikube/kicbase:v0.0.45   "/usr/local/bin/entr…"   7 days ago       Exited (255) 54 minutes ago   127.0.0.1:32772->22/tcp, 12
7.0.0.1:32771->2376/tcp, 127.0.0.1:32770->5000/tcp, 127.0.0.1:32769->8443/tcp, 127.0.0.1:32768->32443/tcp   minikube
PS C:\Users\DELL\Downloads\i201815_A3\episode-01\backend>
```

- **Frontend Dockerfile**: Located at `C:\Users\DELL\Downloads\i201815_A3\episode-01\fronted\Dockerfile`.

```
PROBLEMS 2   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS                                   powershell - fronted
PS C:\Users\DELL\Downloads\i201815_A3\episode-01\fronted> docker build -t episode-01-frontend .
>>
[+] Building 98.8s (13/15)                                                                          docker:default
 => [internal] load build definition from Dockerfile                                                          0.0s
 => => transferring dockerfile: 762B                                                                          0.0s
 => [internal] load metadata for docker.io/library/nginx:alpine                                               2.0s
 => [internal] load metadata for docker.io/library/node:18                                                    1.7s
 => [auth] library/nginx:pull token for registry-1.docker.io                                                  0.0s
[+] Building 98.8s (13/15)                                                                          docker:default
 => [internal] load build definition from Dockerfile                                                          0.0s
 => => transferring dockerfile: 762B                                                                          0.0s
 => [internal] load metadata for docker.io/library/nginx:alpine                                               2.0s
 => [internal] load metadata for docker.io/library/node:18                                                    1.7s
 => [auth] library/nginx:pull token for registry-1.docker.io                                                  0.0s
[+] Building 98.9s (13/15)                                                                          docker:default
 => [internal] load build definition from Dockerfile                                                          0.0s
 => => transferring dockerfile: 762B                                                                          0.0s
 => [internal] load metadata for docker.io/library/nginx:alpine                                               2.0s
 => [internal] load metadata for docker.io/library/node:18                                                    1.7s
 => [auth] library/nginx:pull token for registry-1.docker.io                                                  0.0s
 => [internal] load build definition from Dockerfile                                                          0.0s
 => => transferring dockerfile: 762B                                                                          0.0s
 => [internal] load metadata for docker.io/library/nginx:alpine                                               2.0s
 => [internal] load metadata for docker.io/library/node:18                                                    1.7s
 => [auth] library/nginx:pull token for registry-1.docker.io                                                  0.0s
 => [internal] load metadata for docker.io/library/nginx:alpine                                               2.0s
 => [internal] load metadata for docker.io/library/node:18                                                    1.7s
 => [auth] library/nginx:pull token for registry-1.docker.io                                                  0.0s
```

Here is the screenshot of the frontend service running in a container:

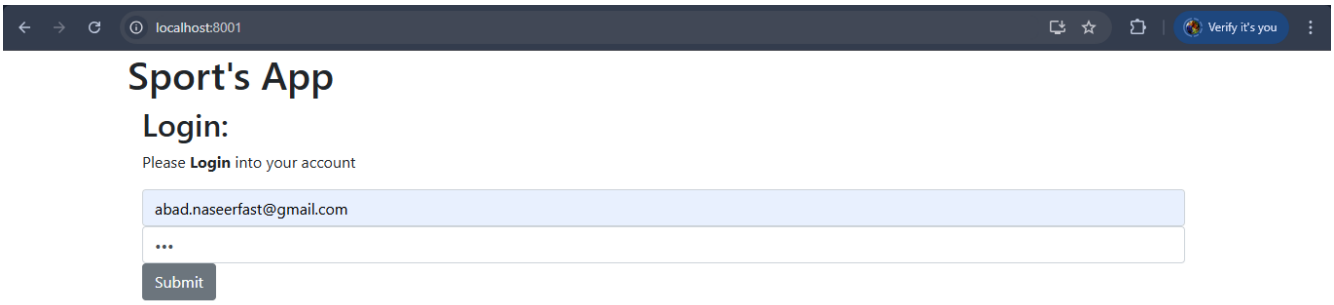
```
PS C:\Users\DELL\Downloads\i201815_A3\episode-01\fronted> docker run -d -p 8001:8001 --name episode-01-frontend episode-01-frontend
bf2e9b75e22646eacd87861846abfe8d7c375efa59856086b688c6e1dcc3e1c0
PS C:\Users\DELL\Downloads\i201815_A3\episode-01\fronted> docker ps -a
CONTAINER ID   IMAGE                                COMMAND                  CREATED          STATUS                          PORTS
                                                                                                                   NAMES
bf2e9b75e226   episode-01-frontend                  "/docker-entrypoint.…"   9 seconds ago    Up 8 seconds                    80/tcp, 0.0.0.0:8001->80
01/tcp                                                                                                             episode-01-frontend
fe0fcee67376   episode-01-backend                   "docker-entrypoint.s…"   25 minutes ago   Up 25 minutes                   0.0.0.0:8080->8080/tcp
                                                                                                                   episode-01-backend
7027825f2f2d   gcr.io/k8s-minikube/kicbase:v0.0.45  "/usr/local/bin/entr…"   7 days ago       Exited (255) About an hour ago  127.0.0.1:32772->22/tcp,
 127.0.0.1:32771->2376/tcp, 127.0.0.1:32770->5000/tcp, 127.0.0.1:32769->8443/tcp, 127.0.0.1:32768->32443/tcp   minikube
```

But docker-compose will handle the mongodb part as well by making the communication b/w the frontend, backend and the mongodb for each service. Here is the completed image and running container of the service (episode-01).

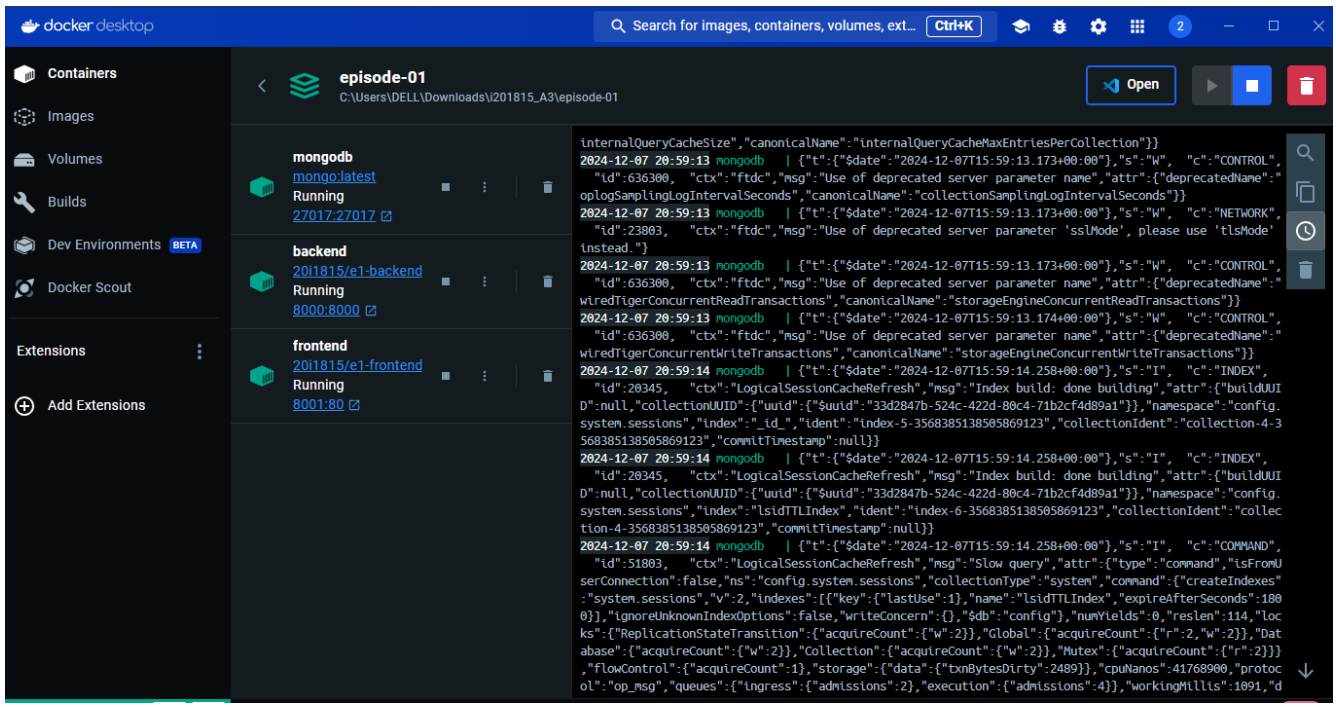
```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS                                   docker-compose - episode-01

PS C:\Users\DELL\Downloads\i201815_A3\episode-01> docker-compose up --build
  >>
  [+] Building 1.4s (11/11)                                                                          docker:default
  [+] Building 30.8s (25/25) FINISHED                                                                docker:default
   => [backend internal] load build definition from Dockerfile                                                0.1s
   => => transferring dockerfile: 517B                                                                        0.0s
   => [backend internal] load metadata for docker.io/library/node:18                                         1.0s
   => [backend internal] load .dockerignore                                                                   0.0s
   => => transferring context: 2B                                                                             0.0s
   => [backend internal] load build context                                                                   0.1s
   => => transferring context: 981B                                                                           0.0s
   => [backend 1/6] FROM docker.io/library/node:18@sha256:b57ae84fe7880a23b389f8260d726b784010ed470c2ee26d4e2cbdb955d25b12   0.0s
   => CACHED [backend 2/6] WORKDIR /app                                                                       0.0s
   => CACHED [backend 3/6] COPY package.json package-lock.json ./                                             0.0s
   => CACHED [backend 4/6] RUN npm install                                                                    0.0s
   => CACHED [backend 5/6] COPY . .                                                                           0.0s
   => CACHED [backend 6/6] WORKDIR /app/src                                                                   0.0s
   => [backend] exporting to image                                                                            0.0s
   => => exporting layers                                                                                     0.0s
   => => writing image sha256:3e3cf27bafbea002a51ad6c3b2c1b9aa959732165829364b11bdafe2195fedd2                0.0s
   => => naming to docker.io/library/episode-01-backend                                                       0.0s
   => [frontend internal] load build definition from Dockerfile                                               0.1s
```

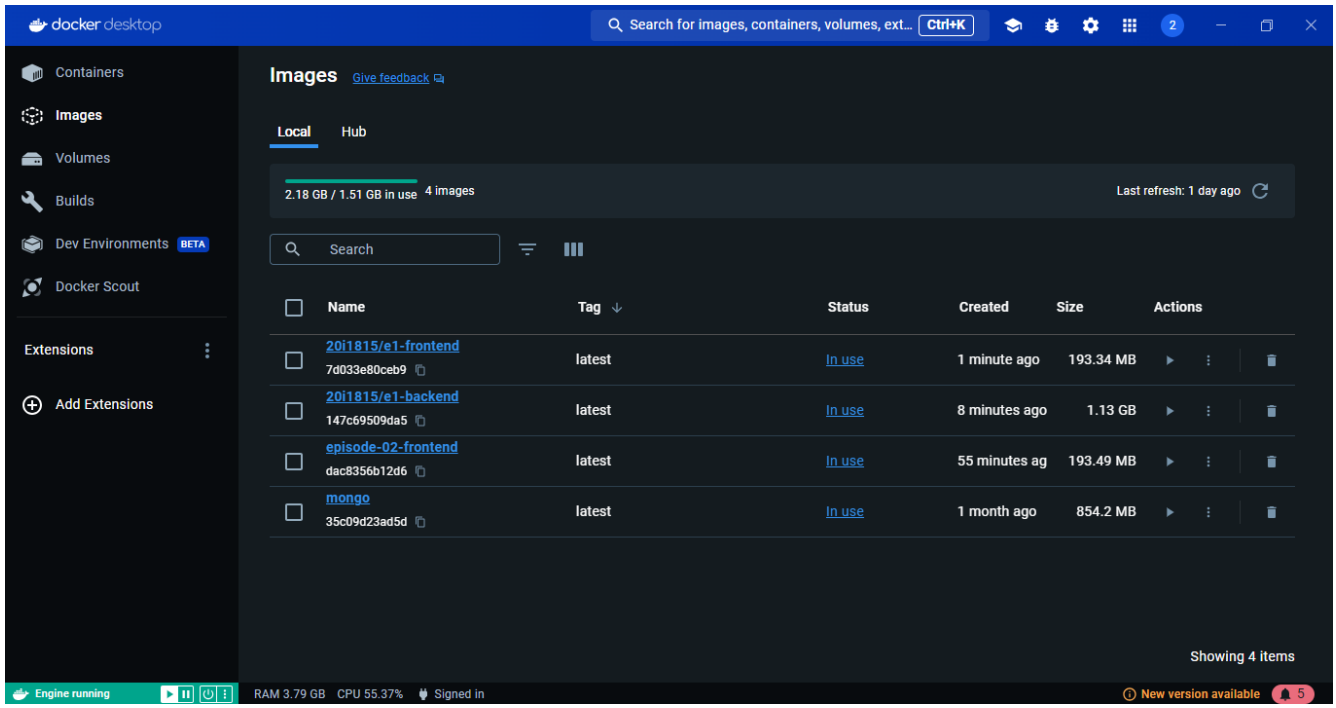
And we can access these routes on localhost:



And here is the docker desktop containers for episode-01 service running:



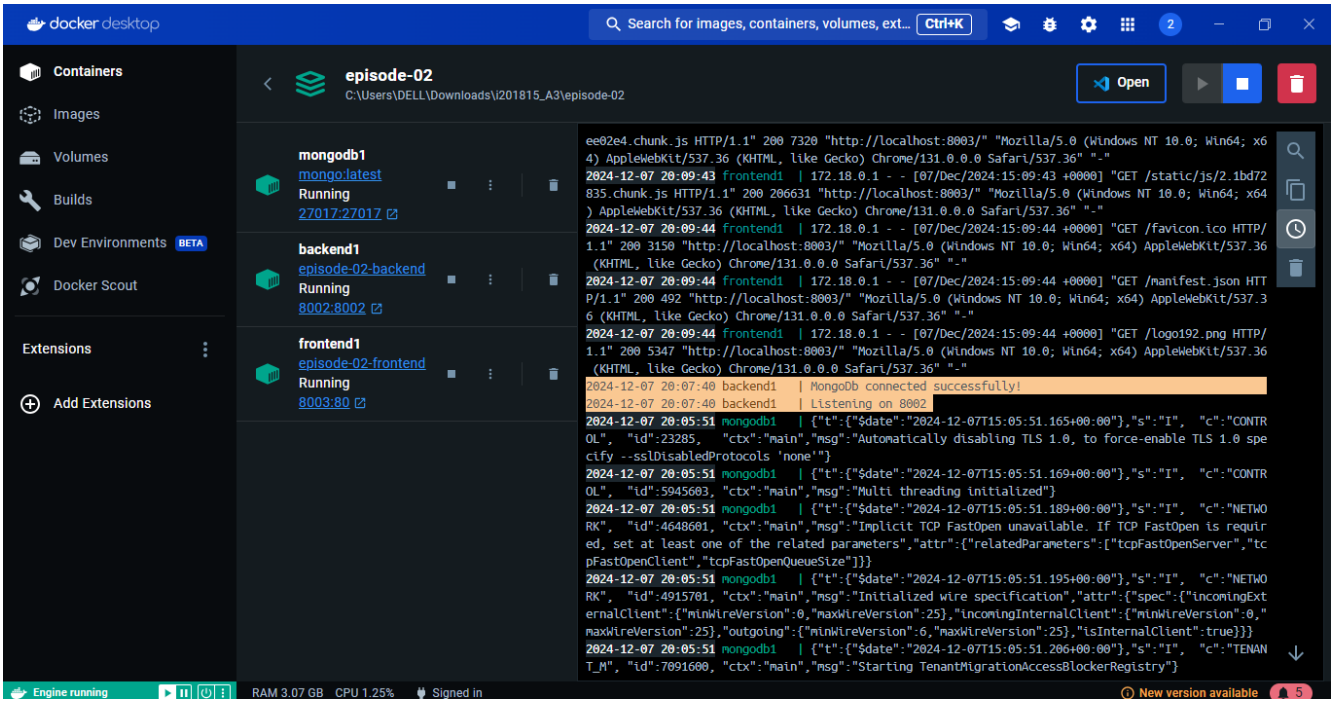
And here are the images:



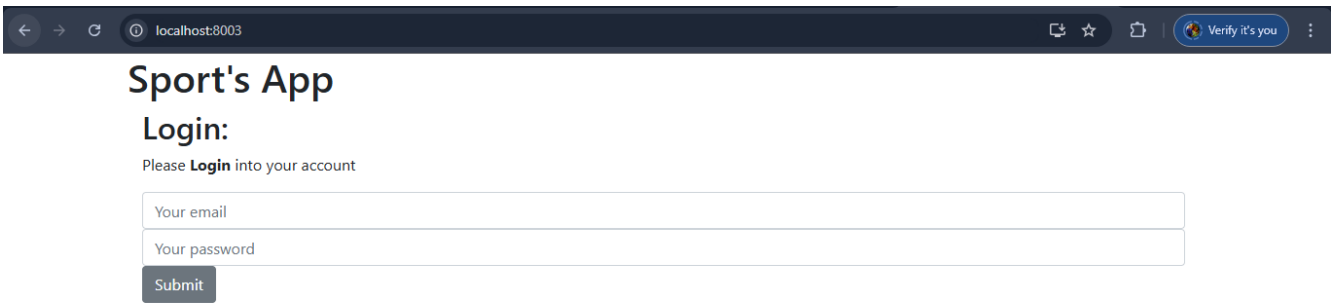
Episode-02:

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS                                   docker-compose - episode-02
PS C:\Users\DELL\Downloads\i201815_A3\episode-02> docker-compose up --build
>>
[+] Building 0.0s (0/0)  docker:default
[+] Building 7.3s (12/12)                                                                          docker:default
[+] Building 12.4s (27/27) FINISHED                                                                docker:default
 => [backend internal] load build definition from Dockerfile                                                0.3s
 => => transferring dockerfile: 517B                                                                        0.2s
 => [backend internal] load metadata for docker.io/library/node:18                                          3.2s
 => [backend auth] library/node:pull token for registry-1.docker.io                                         0.0s
 => [backend internal] load .dockerignore                                                                   0.1s
 => => transferring context: 2B                                                                             0.0s
 => [backend 1/6] FROM docker.io/library/node:18@sha256:b57ae84fe7880a23b389f8260d726b784010ed470c2ee26d4e2cbdb955d25b12   0.0s
 => [backend internal] load build context                                                                   0.1s
 => => transferring context: 981B                                                                           0.0s
 => CACHED [backend 2/6] WORKDIR /app                                                                       0.0s
 => CACHED [backend 3/6] COPY package.json package-lock.json ./                                             0.0s
 => CACHED [backend 4/6] RUN npm install                                                                    0.0s
 => CACHED [backend 5/6] COPY . .                                                                           0.0s
 => CACHED [backend 6/6] WORKDIR /app/src                                                                   0.0s
 => [backend] exporting to image                                                                            0.1s
 => => exporting layers                                                                                     0.0s
 => => writing image sha256:d3b71999c54f6f6b8e3c105798753ac9111d1c36f54267988c5e2d9dfb9cf996                0.0s
 => => naming to docker.io/library/episode-02-backend                                                       0.0s
 => [frontend internal] load build definition from Dockerfile                                               0.2s
 => => transferring dockerfile: 319B                                                                        0.1s
 => [frontend internal] load metadata for docker.io/library/nginx:latest                                    2.0s
 => [frontend internal] load metadata for docker.io/library/node:18-alpine                                  1.4s
 => [frontend auth] library/nginx:pull token for registry-1.docker.io                                       0.0s
```

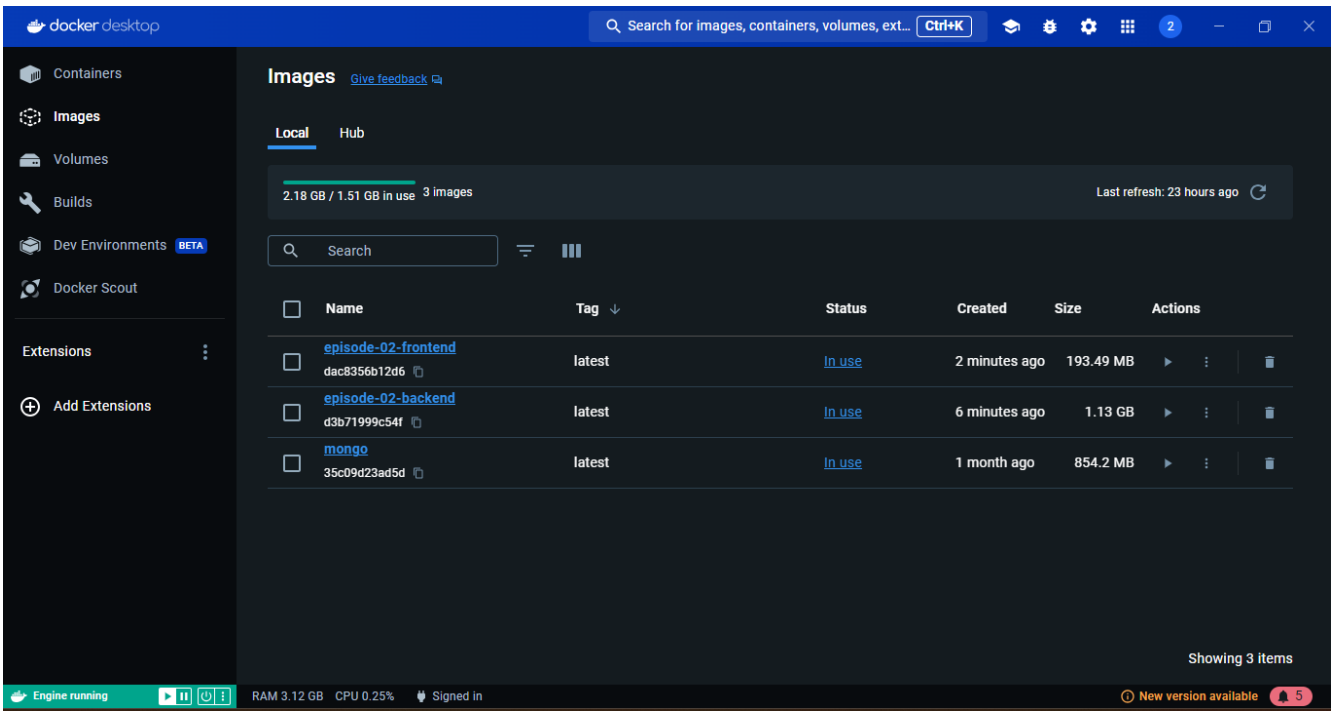
And here is the docker desktop log that shows the episode is running perfectly:



And when accessed on browser, it shows the view:



And here are the images for episodes-02:



Now here we will find the episode-03:

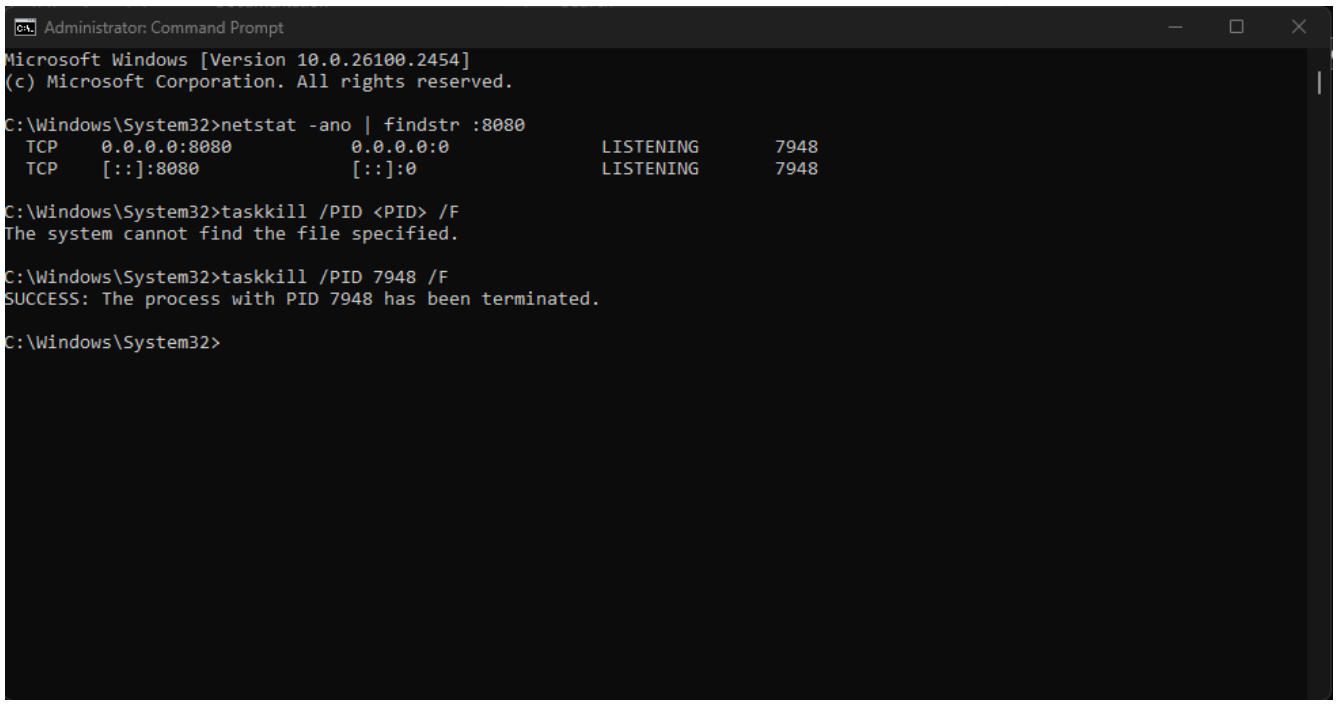
```
PROBLEMS 3   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS                                   docker-compose - episode-04
PS C:\Users\DELL\Downloads\i201815_A3> cd episode-04
PS C:\Users\DELL\Downloads\i201815_A3\episode-04> docker-compose up --build
>>
[+] Building 683.1s (10/10)                                                                          docker:default
[+] Building 1533.8s (21/25)                                                                         docker:default
 => [frontend auth] library/node:pull token for registry-1.docker.io                                          0.1s
 => [frontend auth] library/nginx:pull token for registry-1.docker.io                                         0.1s
 => [frontend internal] load .dockerignore                                                                    0.3s
 => => transferring context: 2B                                                                               0.1s
 => [frontend stage-1 1/2] FROM docker.io/library/nginx:latest@sha256:fb197595ebe76b9c0c14ab68159fd3c08bd067ec62300583543f0e  0.0s
 => [frontend builder 1/6] FROM docker.io/library/node:18-alpine@sha256:7e43a2d633d91e8655a6c0f45d2ed987aa4930f0792f6d9dd3bf  0.0s
 => CACHED [frontend builder 2/6] WORKDIR /app                                                                0.0s
 => [frontend internal] load build context                                                                    3.9s
 => => transferring context: 871.45kB                                                                         3.6s
 => [frontend builder 3/6] COPY package*.json ./                                                              3.5s
```

Here you will find image creation and docker container building for episode-05:

```
PROBLEMS 3   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS                                   docker-compose - episode-01
 ✓ Container mongodb   Created
                       2.1s
 ✓ Container backend   Created
                       1.0s
 ✓ Container frontend  Created
                       1.4s
Attaching to backend, frontend, mongodb
mongodb   | {"t":{"$date":"2024-12-07T17:22:54.122+00:00"},"s":"I",  "c":"CONTROL",  "id":23285,   "ctx":"main","msg":"Automaticall
y disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'"}
mongodb   | {"t":{"$date":"2024-12-07T17:22:54.159+00:00"},"s":"I",  "c":"CONTROL",  "id":5945603, "ctx":"main","msg":"Multi thread
ing initialized"}
mongodb   | {"t":{"$date":"2024-12-07T17:22:54.263+00:00"},"s":"I",  "c":"NETWORK",  "id":4648601, "ctx":"main","msg":"Implicit TCP
 FastOpen unavailable. If TCP FastOpen is required, set at least one of the related parameters","attr":{"relatedParameters":["tcpFa
stOpenServer","tcpFastOpenClient","tcpFastOpenQueueSize"]}}
mongodb   | {"t":{"$date":"2024-12-07T17:22:54.296+00:00"},"s":"I",  "c":"NETWORK",  "id":4915701, "ctx":"main","msg":"Initialized
wire specification","attr":{"spec":{"incomingExternalClient":{"minWireVersion":0,"maxWireVersion":25},"incomingInternalClient":{"mi
nWireVersion":0,"maxWireVersion":25},"outgoing":{"minWireVersion":6,"maxWireVersion":25},"isInternalClient":true}}}
mongodb   | {"t":{"$date":"2024-12-07T17:22:54.361+00:00"},"s":"I",  "c":"TENANT_M", "id":7091600, "ctx":"main","msg":"Starting Ten
antMigrationAccessBlockerRegistry"}
```

Issues and Resolutions

- Issue: Port conflicts during initial runs.
  - Resolution: Adjusted the ports in the `docker run` command.



- Issue: Dependency errors during `npm install`.

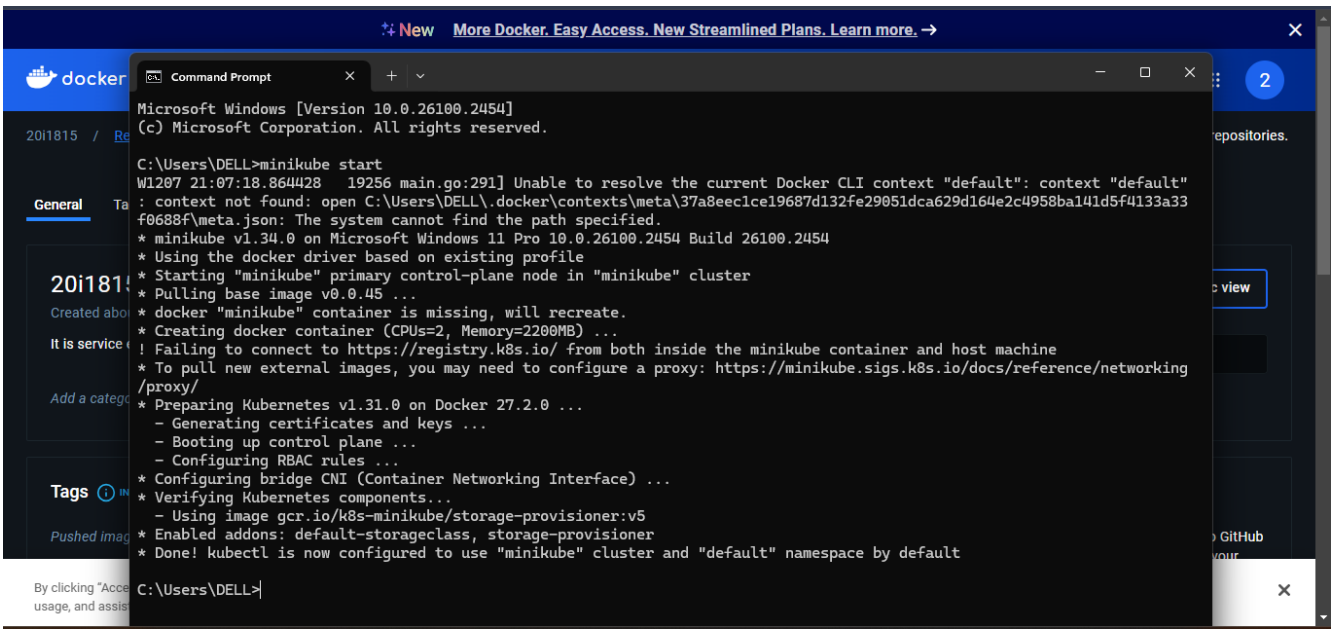
  - Resolution: Updated `package.json` and re-installed dependencies. Error giving me building dependencies so I added this in package.json to resolve (episode-01-frontend): "build": "react-scripts --openssl-legacy-provider build",

Part 2: Kubernetes Deployment (50 Points)

Task Overview

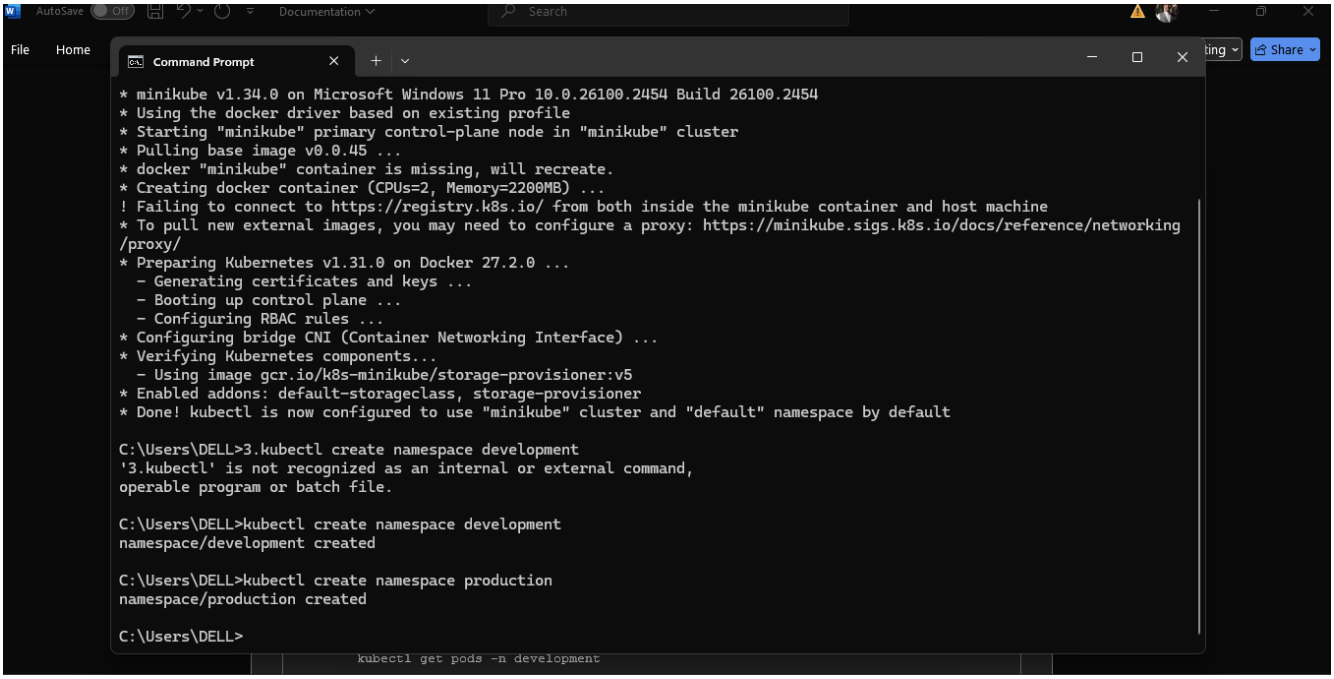
1. **Setting Up a Local Kubernetes Cluster**
   - Minikube was used to set up the cluster.

```
minikube start
```



2. **Namespace Configuration**
   - Namespaces were created for `development` and `production` environments.

3. `kubectl create namespace development`
   `kubectl create namespace production`



4. **Module Containerization**

- Each module's dependencies were encapsulated in their respective containers.

Part 3: Integration and Final Deployment (20 Points)

● **Kubectl Deployment:**

○ **Episode 1:**

Kubectl apply -f k8

○ **Episode 2:**

Kubectl apply -f k8

○ **Episode 3:**

Kubectl apply -f k8

○ **Episode 4:**

Kubectl apply -f k8

○ **Episode 5:**

Kubectl apply -f k8

Kubectl apply -f k8

Kubectl apply -f k8

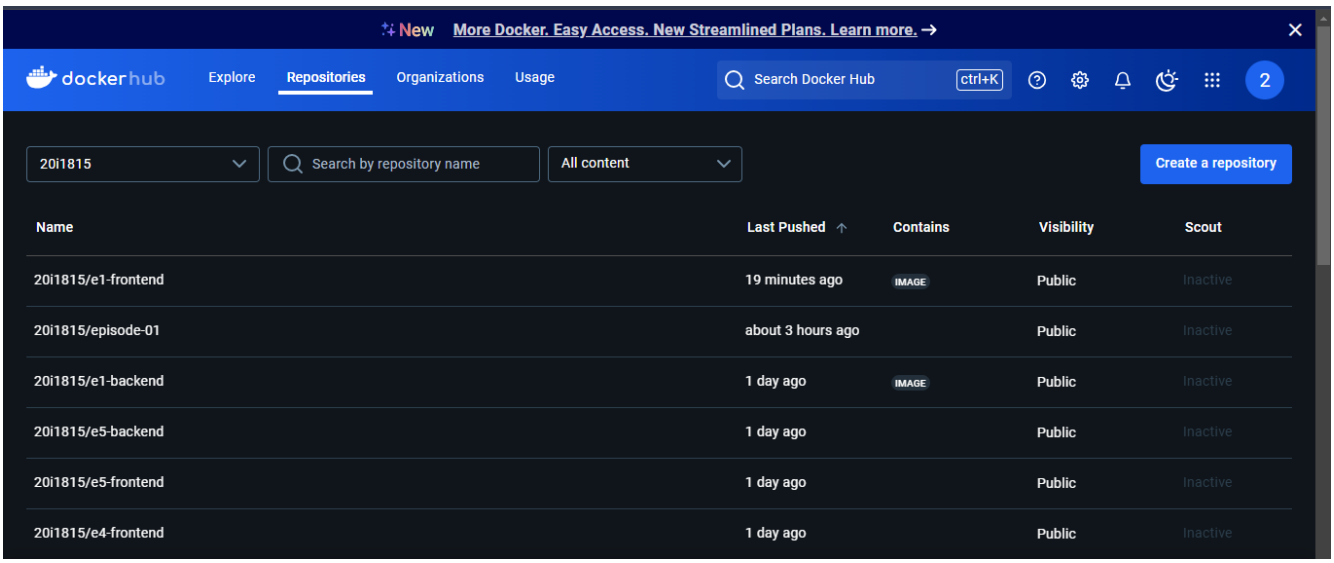
- **Status:**

```
PS C:\Users\DELL\Downloads\i201815_A3\episode-01> kubectl get pods
NAME                        READY   STATUS             RESTARTS        AGE
backend-75b98c4ccc-tphb5    0/1     CrashLoopBackOff   13 (73s ago)    57m
frontend-5b7cdd5fb7-7j66t   0/1     ImagePullBackOff   0               57m
mongodb-7bbb88dc8b-jwcjs    1/1     Running            1 (13m ago)     50m
```

- **Scaling:**

```
PS C:\Users\DELL\Downloads\i201815_A3\episode-01> kubectl scale deployment frontend --replicas=3
deployment.apps/frontend scaled
```

- **Docker hub Images:**

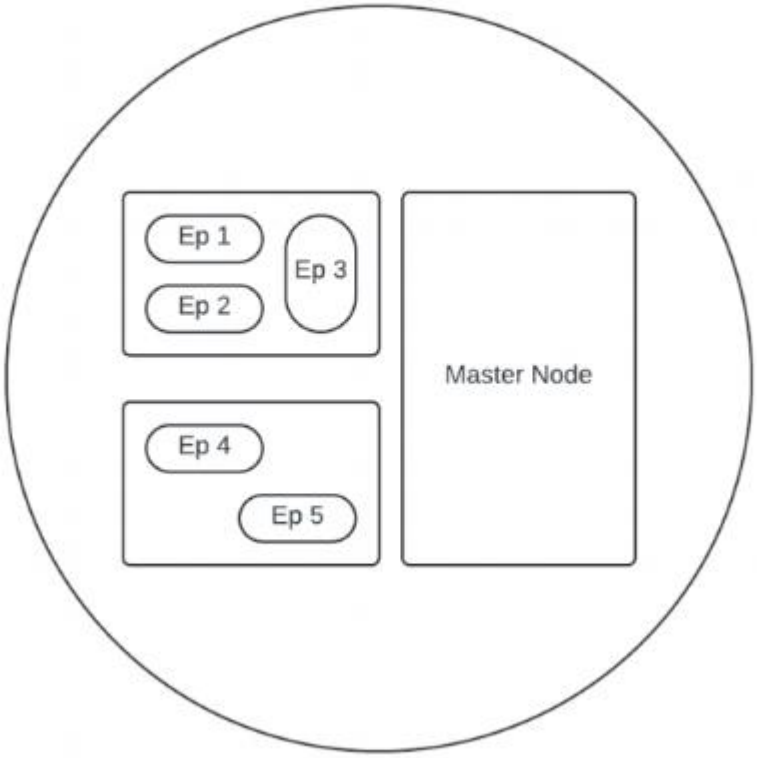


- **Monitoring:**

minikube -p assignment addons enable metrics-server minikube dashboard –profile=assignment

- **Kubernetes Architecture:**

- **Kubernetes Architecture:**



Thank You!