Report for SCD Lab Exam

21i-1111

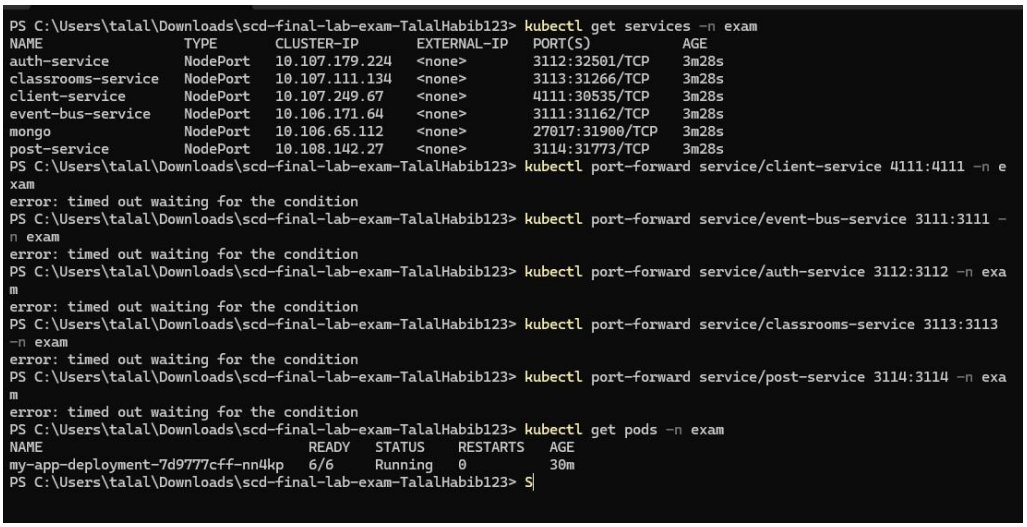
Talal Habib Malik

SE-P

The YAML Files are for Kubernetes, Jenkinsfile for Jenkins and Dockerfile and docker-compose.yml file for docker parts

ISSUE:

Aoa Maam, I mailed you and also talked to you about the issue regarding the port forwarding and engress due to which it does not allow me to access any of my deployed services. Whenerver I do try its time out and the following picture proves it that I cant do port forwarding on anything.



```
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> kubectl get services -n exam
NAME                TYPE       CLUSTER-IP       EXTERNAL-IP   PORT(S)           AGE
auth-service        NodePort   10.107.179.224   <none>        3112:32501/TCP    3m28s
classrooms-service  NodePort   10.107.111.134   <none>        3113:31266/TCP    3m28s
client-service      NodePort   10.107.249.67    <none>        4111:30535/TCP    3m28s
event-bus-service   NodePort   10.106.171.64    <none>        3111:31162/TCP    3m28s
mongo               NodePort   10.106.65.112    <none>        27017:31900/TCP   3m28s
post-service        NodePort   10.108.142.27    <none>        3114:31773/TCP    3m28s
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> kubectl port-forward service/client-service 4111:4111 -n e
xam
error: timed out waiting for the condition
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> kubectl port-forward service/event-bus-service 3111:3111 -
n exam
error: timed out waiting for the condition
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> kubectl port-forward service/auth-service 3112:3112 -n exa
m
error: timed out waiting for the condition
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> kubectl port-forward service/classrooms-service 3113:3113
-n exam
error: timed out waiting for the condition
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> kubectl port-forward service/post-service 3114:3114 -n exa
m
error: timed out waiting for the condition
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> kubectl get pods -n exam
NAME                                  READY   STATUS    RESTARTS   AGE
my-app-deployment-7d9777cff-nn4kp   6/6     Running   0          30m
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> S
```

This issue started when I did the SCD assignment 3 where I did some things to make the assignment work but when I deleted it andreinstalled Minikube for paper its started having issue and I wasn't able to resolve it from a day before paper and even within paper and playground was out of the questions since I had limited time and playground would have taken a major portion of it.

SO because of the I wasn't able to do the testing part in Kubernetes but you can cross verify the yaml files with other people because its defined correctly.

Step:3 Creating Docker Containers:

Assumptions:

Since my roll number is 1111
I added 2000 to the roll number for the backend ports and added 1 to make them separate.

For Frontend I couldn't use 1111 since Window detects anything running as trojan on it for some reason so I added 3000 since 2111 is already used and 3111 is being used for backend. So my frontend port is 4111

These were modifications were made directly to the code given to us.

Frontend Docker files:

```
# Use an official Node runtime as the base image
FROM node:14 as build

# Set the working directory in the container to /app
WORKDIR /app

# Copy package.json and package-lock.json to the working directory
COPY package*.json ./

# Install any needed packages specified in package.json
RUN npm install

# Copy the current directory contents into the container at /app
COPY . .

# Build the app
RUN npm run build

# Start a new stage from scratch
FROM node:14

# Install serve
RUN npm install -g serve

# Set the working directory in the container to /app
WORKDIR /app

# Copy the build directory from the previous stage
COPY --from=build /app/build .

# Make port 5000 available to the world outside this container
EXPOSE 4111

# Run the app when the container launches
CMD ["serve", "-s", ".", "-l", "4111"]
```

This is the frontend docker file for the client folder react app.

Backend Docker files:

```
FROM node:14

# Set the working directory in the contai
WORKDIR /app

# Copy package.json and package-lock.json
COPY package*.json ./

# Install any needed packages specified i
RUN npm install

# Bundle the app source inside the Docker
# (Make sure you have a .dockerignore fil
COPY . .

# Make port 4009 available to the world o
EXPOSE 3112

# Define the command to run the app
CMD [ "node", "index.js" ]
```

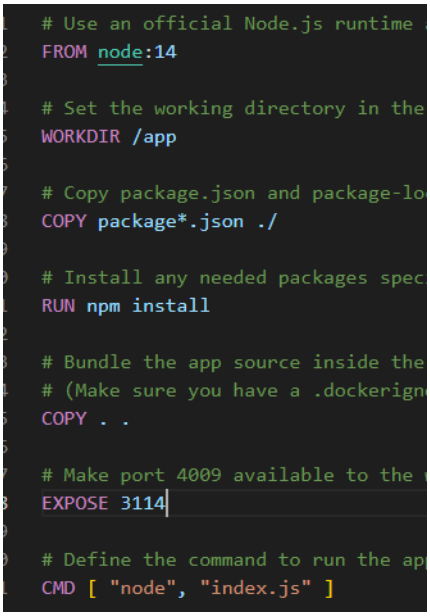
File for Auth folder.

```
FROM node:14

# Set the working directory in the c
WORKDIR /app

# Copy package.json and package-lock
COPY package*.json ./

# Install any needed packages specif
RUN npm install

# Bundle the app source inside the D
# (Make sure you have a .dockerignor
COPY . .

# Make port 4009 available to the wo
EXPOSE 3113

# Define the command to run the app
CMD [ "node", "index.js" ]
```

File for classrooms backend

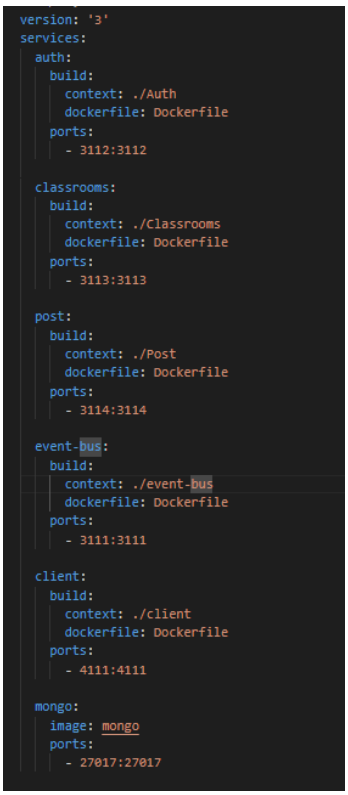
```
# Use an official Node.js runtime
FROM node:14

# Set the working directory in the
WORKDIR /app

# Copy package.json and package-lo
COPY package*.json ./

# Install any needed packages spec
RUN npm install

# Bundle the app source inside the
# (Make sure you have a .dockerign
COPY . .

# Make port 4009 available to the
EXPOSE 3111

# Define the command to run the ap
CMD [ "node", "index.js" ]
```

File for event bus backend

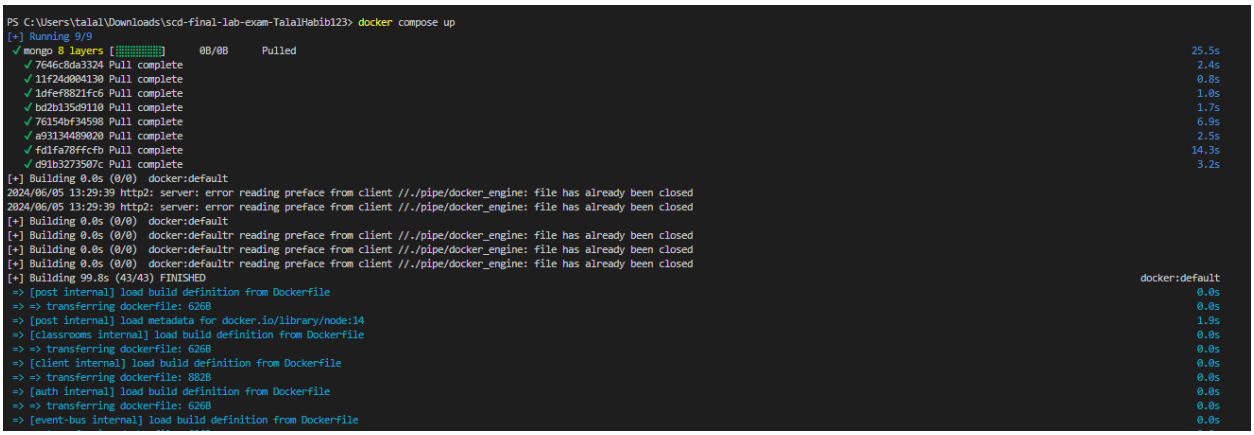


Posts folder docker file

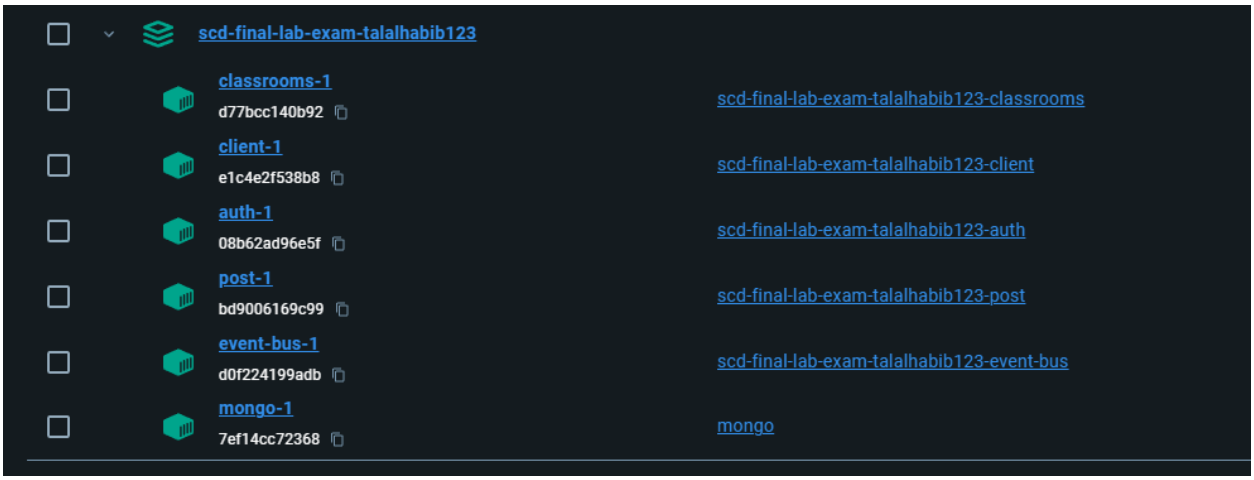
Docker Compose File:



Docker compose file for the whole application to containerize it.



Docker compose up



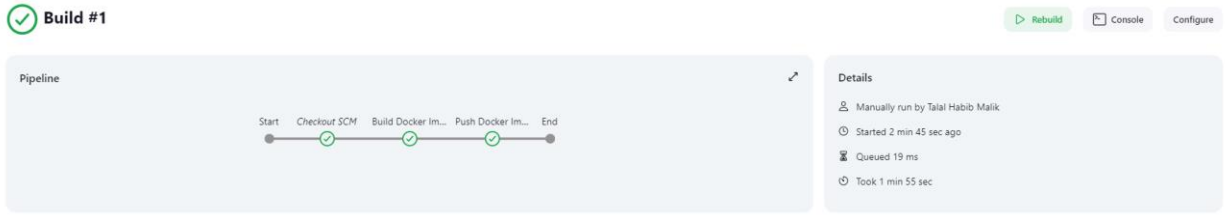
Running Docker Container with an mongo container for the local database.

Step 4: Jenkins Pipeline

```groovy
pipeline {
    stages {
        stage('Build Docker Images') {
            steps {
                script {
                    def services = ['auth', 'classrooms', 'post', 'event-bus', 'client']
                    for (service in services) {
                        // Start of step for 21I-1111
                        bat "docker build -t ${service} ./${service}"
                        // End of step for 21I-1111
                    }
                }
            }
        }

        stage('Push Docker Images') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'badd24f7-ad18-40df-8508-73eff65a3b05', usernameVariable: 'User', passwordVariable: 'Pass')]) {
                    script {
                        def services = ['auth', 'classrooms', 'post', 'event-bus', 'client']
                        for (service in services) {
                            // Start of step for 21I-1111
                            bat "docker login -u %User% -p %Pass%"
                            bat "docker tag ${service} %User%/${service}"
                            bat "docker push %User%/${service}"
                            // End of step for 21I-1111
                        }
                    }
                }
            }
        }
    }
}
```

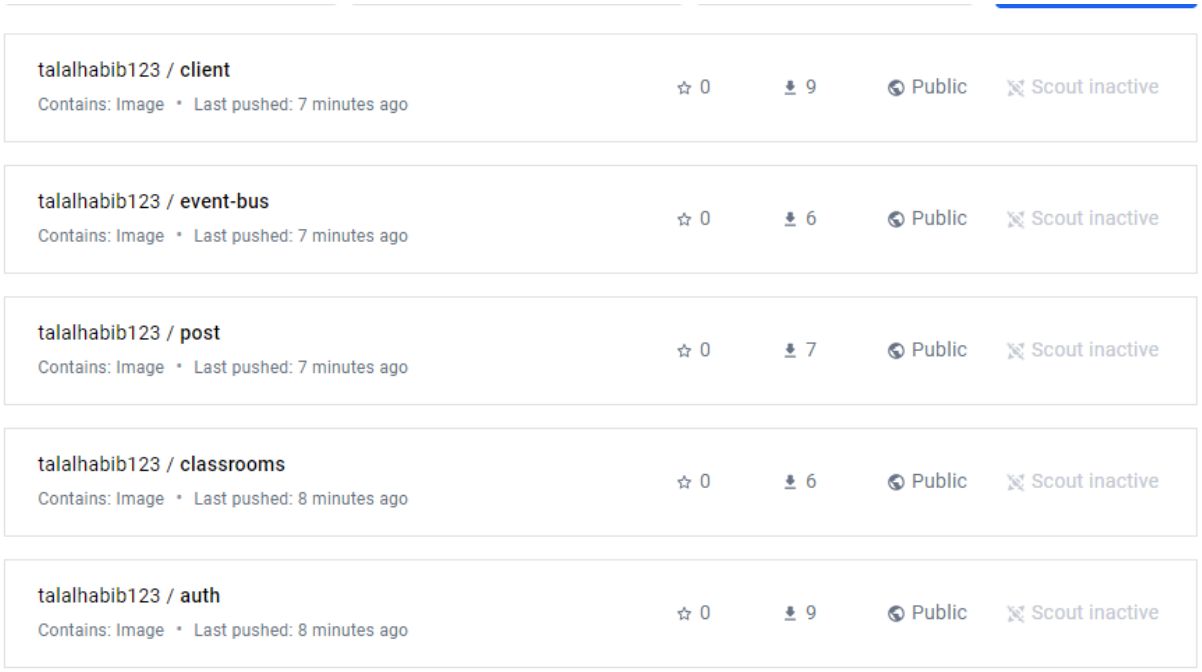
This pipeline loops through all the folder of my repo and makes the images and tags them with my docker hub username and pushes it to the Docker Hub.

Each step has my roll number at the start and at the end




```
latest: digest: sha256:b53a33ed8085e8cd54d71e4abd1
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

My pipeline was successfully built and all my images were pushed to my docker hub.

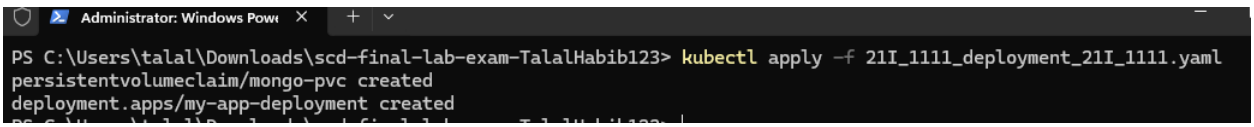


My docker images on docker hub.

Step 5 Deployment and Services

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mongo-pvc
  namespace: exam
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi

---

apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app-deployment
  namespace: exam
spec:
  replicas: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
      - name: auth
        image: talalhabib123/auth
        ports:
        - containerPort: 3112
      - name: classrooms
        image: talalhabib123/classrooms
        ports:
        - containerPort: 3113
      - name: client
        image: talalhabib123/client
        ports:
        - containerPort: 4111
      - name: event-bus
        image: talalhabib123/event-bus
        ports:
        - containerPort: 3111
      - name: post
        image: talalhabib123/post
        ports:
        - containerPort: 3114
      - name: mongodb
        image: mongo
        ports:
        - containerPort: 27017
        volumeMounts:
        - name: mongo-storage
          mountPath: /data/db
      volumes:
      - name: mongo-storage
        persistentVolumeClaim:
          claimName: mongo-pvc
```

Created a Kubernetes deployment file which specifies all the container of the application to run inside one pod and also to create a mongo container as well which



Deployment Created.

```
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> kubectl get deployments -n exam
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
my-app-deployment   1/1     1            1           5m4s
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> kubectl get pods -n exam
NAME                                READY   STATUS    RESTARTS   AGE
my-app-deployment-7d9777cff-nn4kp   6/6     Running   0          5m15s
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123>
```

This verifies that our deployment was done successfully and we have 6 deployments inside one pod, which include Frontend (client), Backend (auth, event-bus, posts, classrooms), and MongoDB as well.

All of this is being performed inside a Namespace to isolate it in the cluster and namespace being used is exam

```
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> kubectl get
NAME              STATUS   AGE
default           Active   159m
exam              Active   11m
kube-node-lease   Active   159m
kube-public       Active   159m
kube-system       Active   159m
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123>
```

Creating Services:

```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: auth-service
  namespace: exam
spec:
  selector:
    app: my-app
    name: auth
  ports:
    - protocol: TCP
      port: 3112
      targetPort: 3112
  type: NodePort
---
apiVersion: v1
kind: Service
metadata:
  name: classrooms-service
  namespace: exam
spec:
  selector:
    app: my-app
    name: classrooms
  ports:
    - protocol: TCP
      port: 3113
      targetPort: 3113
  type: NodePort
---
```

These are not all of our services but this yaml file specifices the services of our deployments.

```
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> kubectl apply -f 21I_1111_service_21I_1111.yaml
service/auth-service created
service/classrooms-service created
service/client-service created
service/event-bus-service created
service/post-service created
service/mongo created
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> kubectl get services -n exam
NAME                TYPE       CLUSTER-IP       EXTERNAL-IP   PORT(S)           AGE
auth-service        NodePort   10.107.179.224   <none>        3112:32501/TCP    21s
classrooms-service  NodePort   10.107.111.134   <none>        3113:31266/TCP    21s
client-service      NodePort   10.107.249.67    <none>        4111:30535/TCP    21s
event-bus-service   NodePort   10.106.171.64    <none>        3111:31162/TCP    21s
mongo               NodePort   10.106.65.112    <none>        27017:31900/TCP   21s
post-service        NodePort   10.108.142.27    <none>        3114:31773/TCP    21s
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123>
```

The above command in the screenshot is used to create the services for all of our deployments from the yaml file above.

```
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> kubectl get deployments -n exam
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
my-app-deployment   1/1     1            1           31m
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> kubectl scale deployment my-app-deployment --replicas=3 -n
 exam
deployment.apps/my-app-deployment scaled
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123>
```

Here we are scaling the application and all of it components within the pod.

Observing the how the cluster has redistributed the workload

```
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> kubectl get pods -n exam -o wide
NAME                             READY   STATUS             RESTARTS       AGE     IP           NODE       NOMINATED NODE   READINESS GATES
my-app-deployment-7d9777cff-6cbr4   5/6     CrashLoopBackOff   5 (107s ago)   5m24s   10.244.0.7   minikube   <none>           <none>
my-app-deployment-7d9777cff-nn4kp   6/6     Running            0              37m     10.244.0.6   minikube   <none>           <none>
my-app-deployment-7d9777cff-nsbbs   5/6     CrashLoopBackOff   5 (111s ago)   5m24s   10.244.0.8   minikube   <none>           <none>
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123>
```

mongodb:

Container ID: docker://3da015facbd938cf36c66f0ccf1d3ba1a75716ea6b1e124f3f94e2846955ecf3

Image: mongo

Image ID: docker-pullable://mongo@sha256:108cd0d7867ba32559a3a2c4b353183f7076042369a85c67aab3f7c52dfc2783

Port: 27017/TCP

Host Port: 0/TCP

State: Waiting

Reason: CrashLoopBackOff

Last State: Terminated

Reason: Error

Exit Code: 100

Started: Wed, 05 Jun 2024 14:49:00 +0500

Finished: Wed, 05 Jun 2024 14:49:00 +0500

Ready: False

Restart Count: 4

Environment: <none>

Mounts:

/data/db from mongo-storage (rw)

/var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-xkdkz (ro)

During scaling it is observed that the cluster is waiting for the MongoDB to scale since it crashed.

The testing part wasn't able to do because of the issue I mentioned before. And in the email as well so kindly consider and cross verify.

Kubernetes Dashboard:

Using Kubernetes dashboard to monitor deployed applications and cluster health.

Setting it up

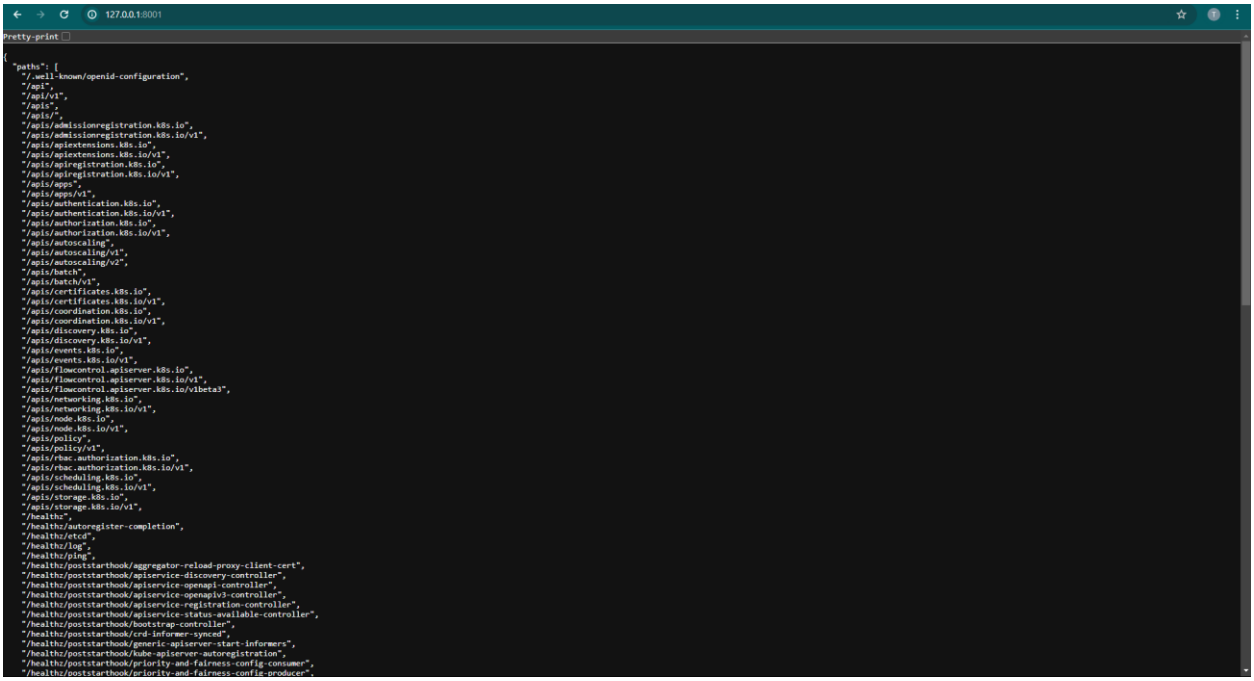
```
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.0.0-beta8/aio/deploy/recommende
d.yaml
namespace/kubernetes-dashboard created
serviceaccount/kubernetes-dashboard created
service/kubernetes-dashboard created
secret/kubernetes-dashboard-certs created
secret/kubernetes-dashboard-csrf created
secret/kubernetes-dashboard-key-holder created
configmap/kubernetes-dashboard-settings created
role.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrole.rbac.authorization.k8s.io/kubernetes-dashboard created
rolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
Warning: spec.template.spec.nodeSelector[beta.kubernetes.io/os]: deprecated since v1.14; use "kubernetes.io/os" instead
deployment.apps/kubernetes-dashboard created
service/dashboard-metrics-scraper created
Warning: spec.template.metadata.annotations[seccomp.security.alpha.kubernetes.io/pod]: non-functional in v1.27+; use the "seccompProfile" field instead
deployment.apps/dashboard-metrics-scraper created
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123>
```

Pulling yaml files from github

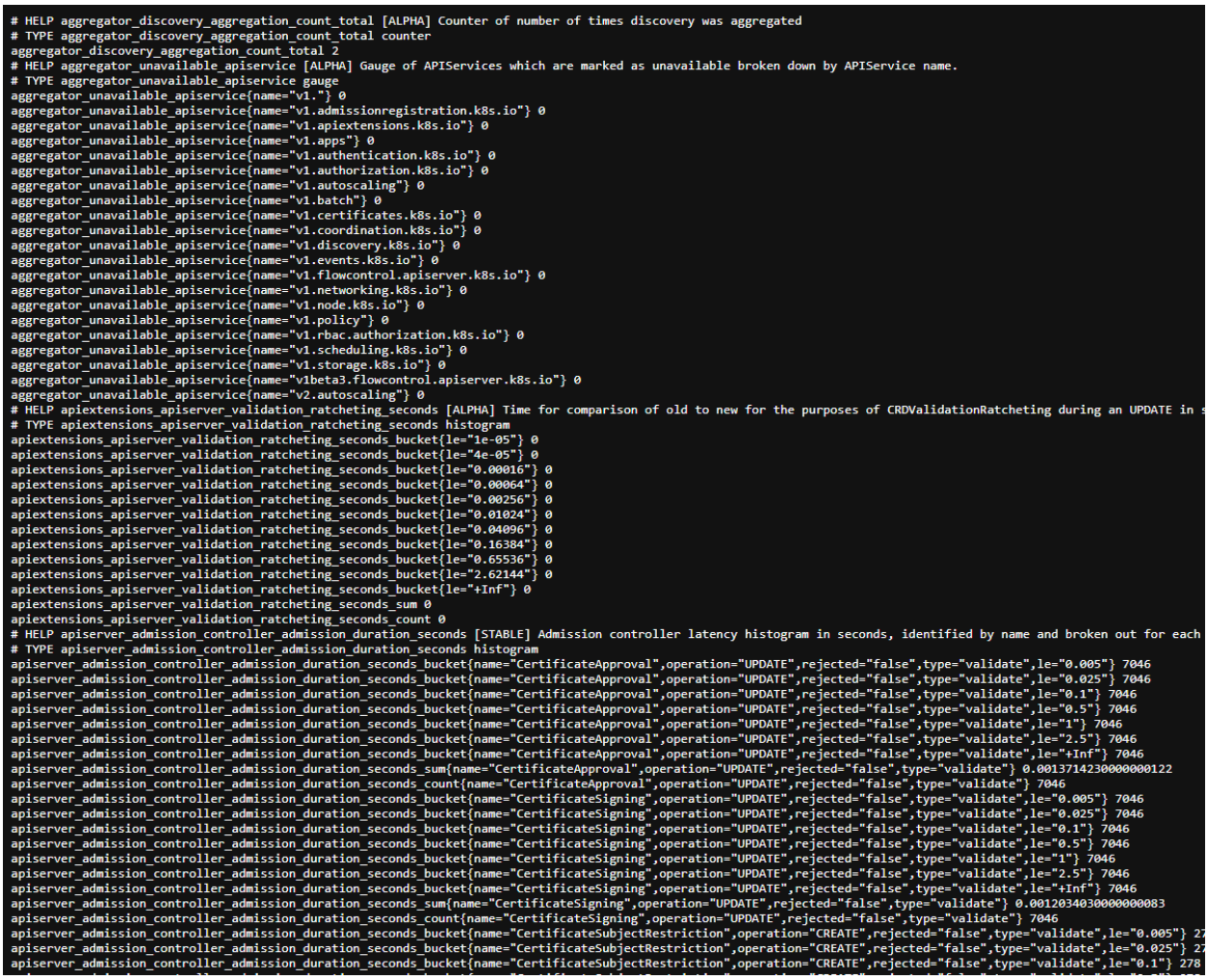
```
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123> kubectl apply -f 21I_1111_dashboard-admin_21I_1111.yaml
serviceaccount/dashboard-admin-sa created
clusterrolebinding.rbac.authorization.k8s.io/dashboard-admin-sa created
PS C:\Users\talal\Downloads\scd-final-lab-exam-TalalHabib123>
```

Creating an admin dashboard service

Started kubectl proxy to access the dashboard and was given this

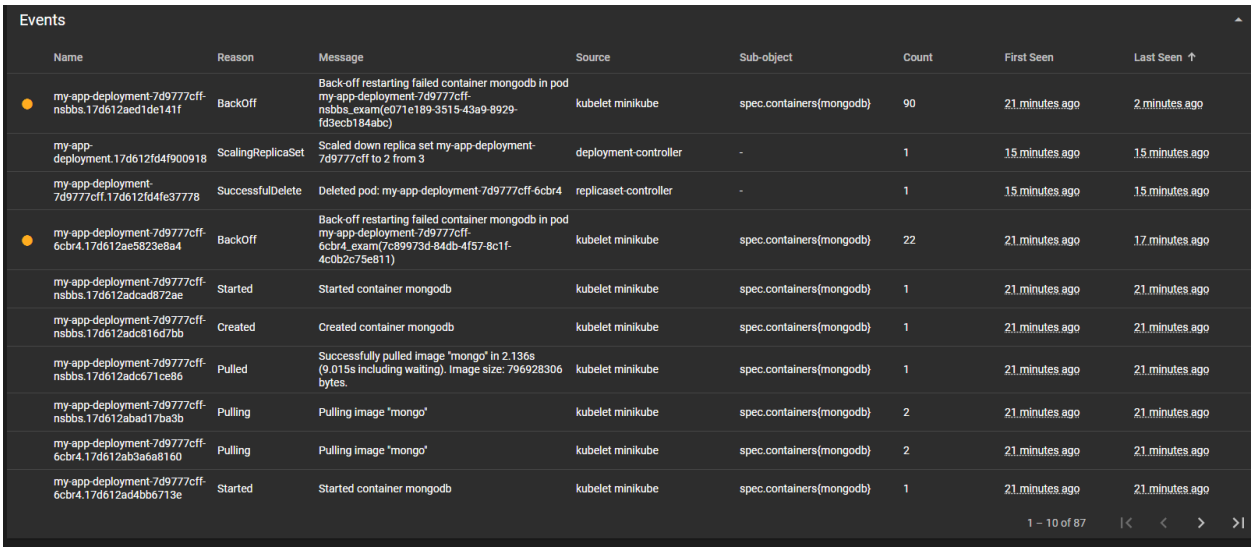


These are paths where I check and see specific things of the cluster



These are the metrics of the cluster

We can also use minkube dashboard to monitor which is a more simplified version of this



Monitoring of my namespace in which my pod is located.

Step 6 Additional Things:

Resource and Requests Limits:

```yaml
spec:
  containers:
  - name: auth
    image: talalhabib123/auth
    ports:
    - containerPort: 3112
    resources:
      requests:
        memory: "64Mi"
        cpu: "250m"
      limits:
        memory: "128Mi"
        cpu: "500m"
  - name: classrooms
    image: talalhabib123/classrooms
    ports:
    - containerPort: 3113
    resources:
      requests:
        memory: "64Mi"
        cpu: "250m"
      limits:
        memory: "128Mi"
        cpu: "500m"
  - name: client
    image: talalhabib123/client
    ports:
```

For this each container was given instructions to take this much memory and cpu at the start and also specify the maximum resource limits for it

Annotations and labels:

```yaml

---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app-deployment
  namespace: exam
  labels:
    app: my-app
    environment: production
  annotations:
    description: "This is my application"
    version: "1.0"
spec:
  replicas: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
        environment: production
      annotations:
        description: "This is my application"
        version: "1.0"
    spec:
      containers:
      - name: auth
        image: talalhabib123/auth
        ports:
        - containerPort: 3112
```

I have added labels specified annotations as well to give it more ease to integrate with other tools.

Health Checks:

For the heath checks we don't specifically need to do anything as cluster is already monitoring our applications and only forward it if they running and also restarts its in the event that there a issue with a container, so its automatically restarts it as well.