Udagram Image Filtering Microservice

Name and Surname: Lekgotla Motaung

In this project I have completed the following:

- Refactor the monolith application to microservices
- Set up each microservice to be run in its own Docker container
- Set up a Travis CI pipeline to push images to Dockerhub
- Deploy the Dockerhub images to the Kubernetes cluster

Part 1 - Run the project locally as a Monolithic application

The application works well as a monolith application.

Part 2 - Run the project locally in a multi-container environment

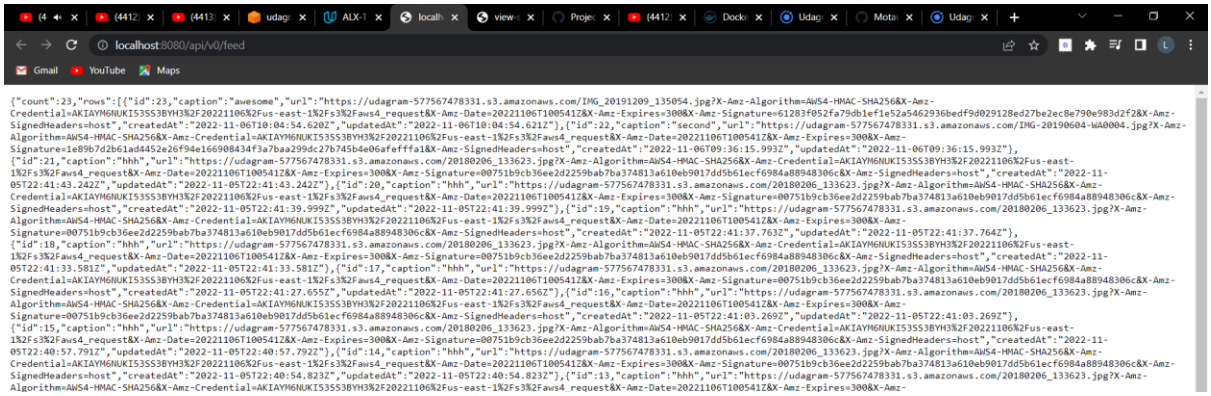
(The work that it required starting point)

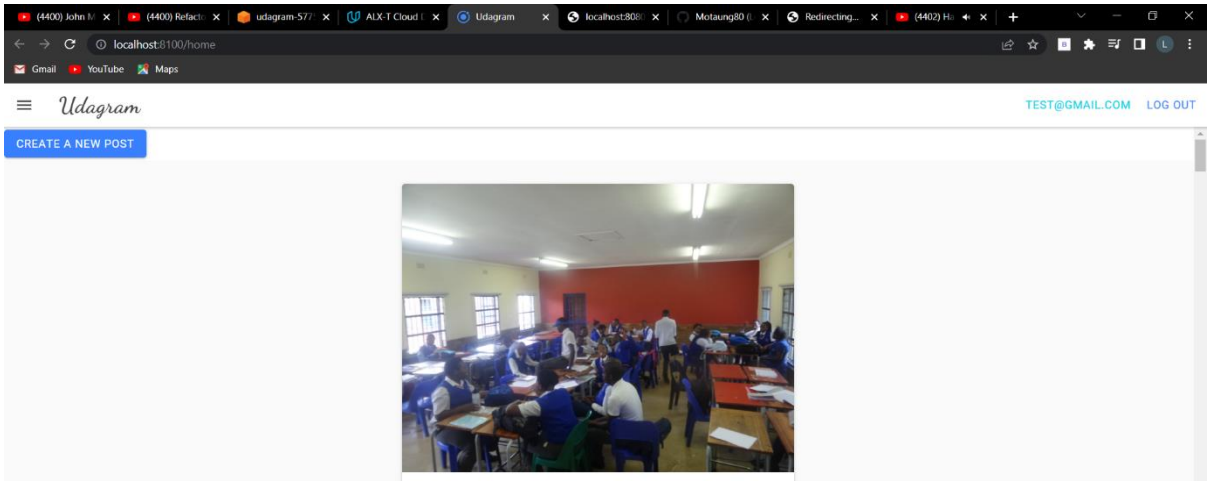
1. Refractor Backend and Frontend
2. Create Docker files
3. Use Docker compose to build and run the application.

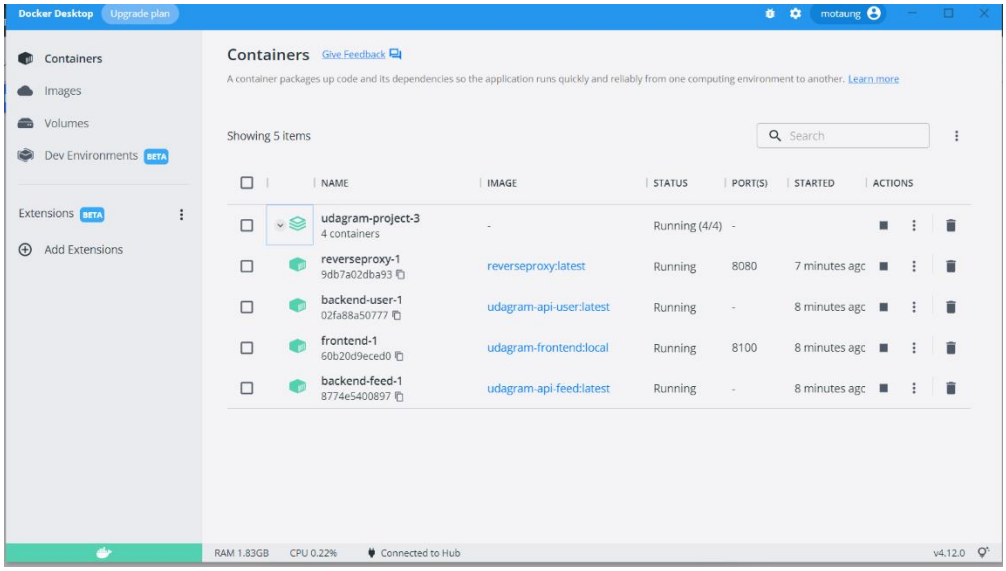
Command Lines used in part 2

- source set_env.sh
- docker image prune –all
- docker-compose -f docker-compose-build.yaml build –parallel
- docker-compose up
- docker-compose down

Screenshots for Part 2



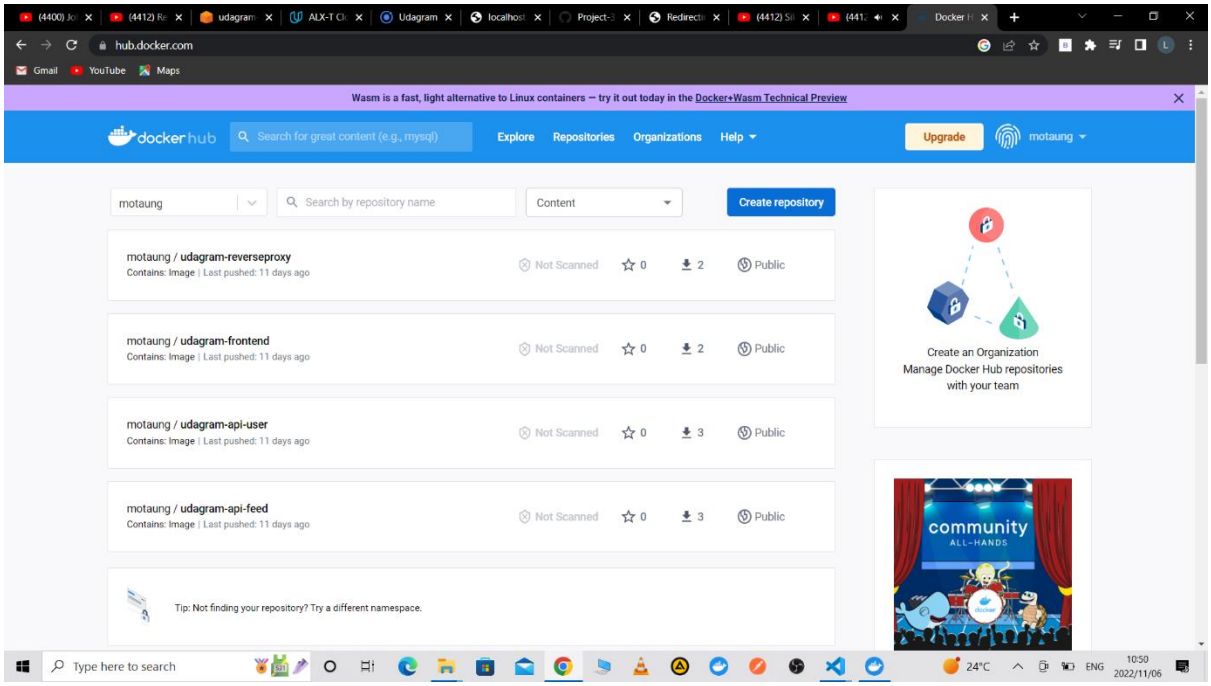


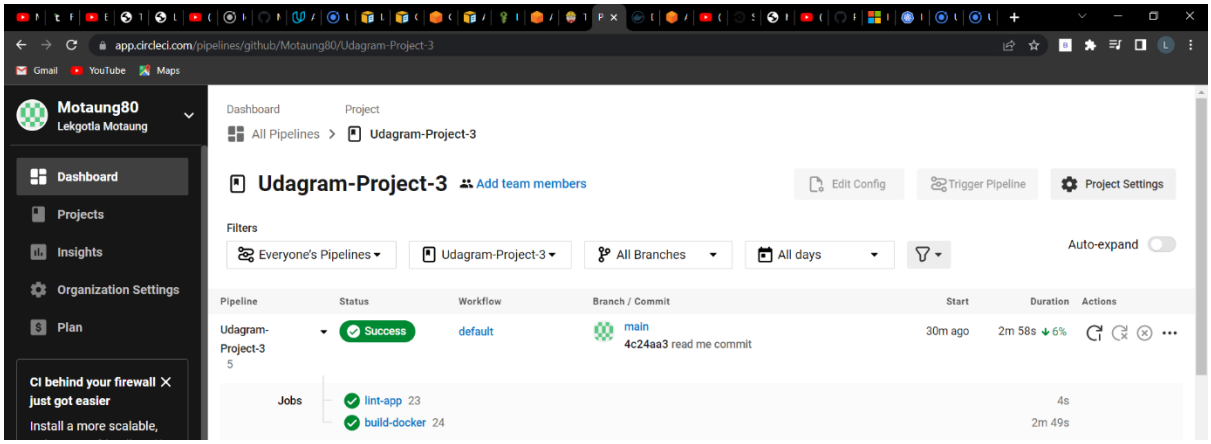


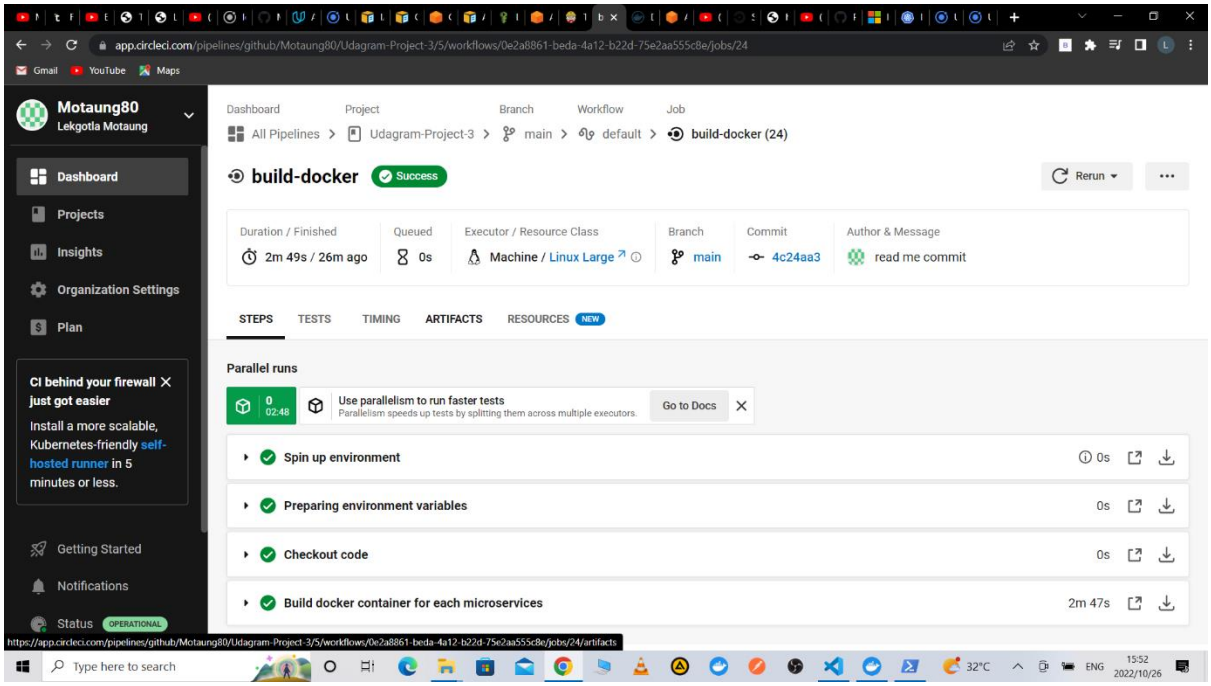
Part 3 - Set up Travis continuous integration pipeline

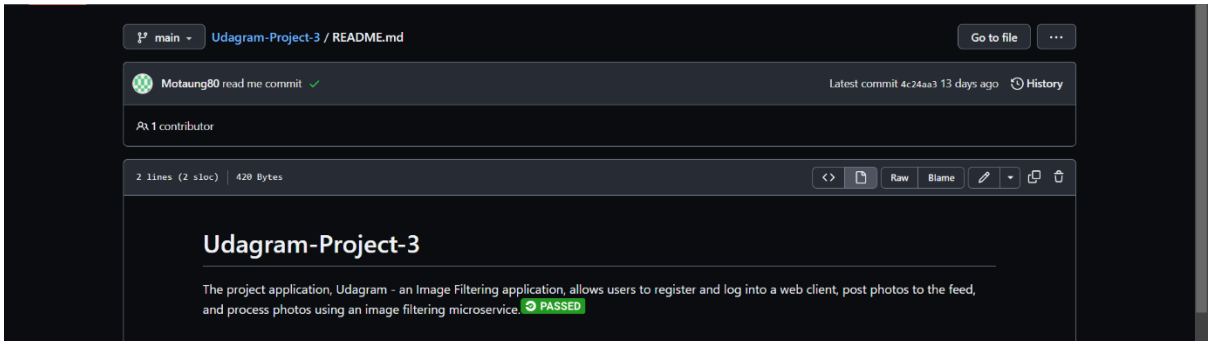
1. Create Dockerhub Repositories
2. Set up Travis CI pipeline

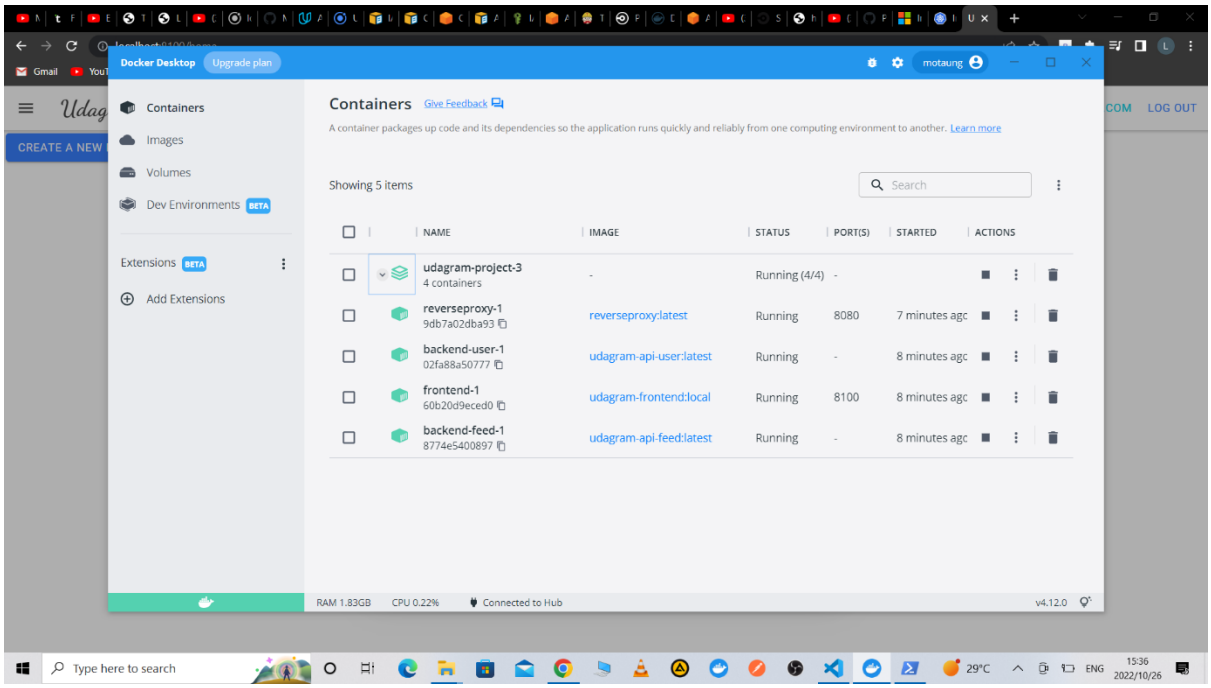
Screenshots for part 3

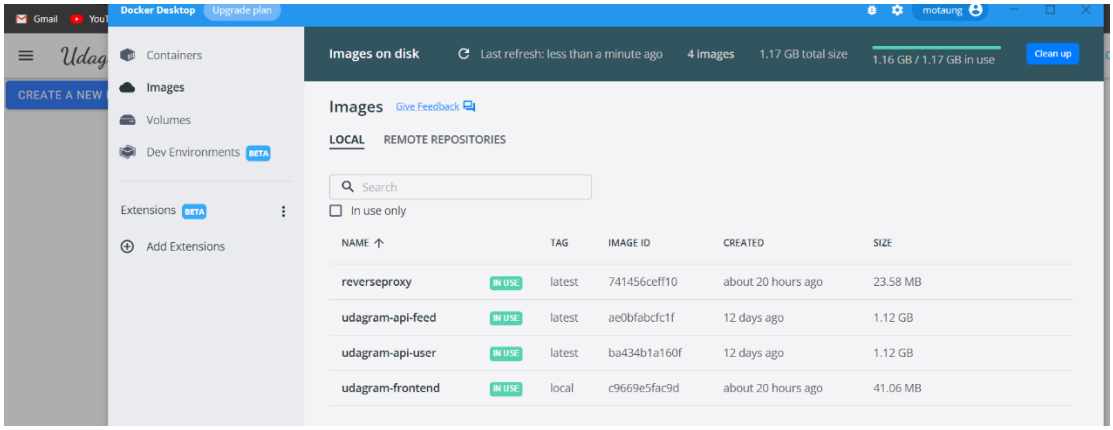





app.circleci.com/pipelines/github/Motaung80/Udagram-Project-3/5/workflows/0e2a8861-beda-4a12-b22d-75e2aa555c8e/jobs/24
Motaung80
Lekgotla Motaung
Dashboard
Projects
Insights
Organization Settings
Plan
CI behind your firewall just got easier
Install a more scalable, Kubernetes-friendly self-hosted runner in 5 minutes or less.
Getting Started
Notifications
Status
Dashboard
Project
Branch
Workflow
Job
All Pipelines > Udagram-Project-3 > main > default > build-docker (24)
build-docker
Success
Rerun
Duration / Finished
2m 49s / 26m ago
Queued
0s
Executor / Resource Class
Machine / Linux Large
Branch
main
Commit
4c24aa3
Author & Message
read me commit
STEPS
TESTS
TIMING
ARTIFACTS
RESOURCES
NEW
Parallel runs
Use parallelism to run faster tests
Parallelism speeds up tests by splitting them across multiple executors.
Go to Docs
Spin up environment
0s
Preparing environment variables
0s
Checkout code
0s
Build docker container for each microservices
2m 47s


main
Udagram-Project-3 / README.md
Go to file
Motaung80 read me commit
Latest commit 4c24aa3 13 days ago
History
1 contributor
2 lines (2 sloc)
420 Bytes
Raw
Blame
Udagram-Project-3
The project application, Udagram - an Image Filtering application, allows users to register and log into a web client, post photos to the feed, and process photos using an image filtering microservice.
PASSED


Docker Desktop
Upgrade plan
motaung
Containers
Images
Volumes
Dev Environments
BETA
Extensions
BETA
Add Extensions
Containers
Give Feedback
A container packages up code and its dependencies so the application runs quickly and reliably from one computing environment to another. Learn more
Showing 5 items
Search
NAME
IMAGE
STATUS
PORT(S)
STARTED
ACTIONS
udagram-project-3
4 containers
Running (4/4)
reverseproxy-1
9db7a02dba93
reverseproxy:latest
Running
8080
7 minutes ago
backend-user-1
02fa88a50777
udagram-api-user:latest
Running
8 minutes ago
frontend-1
60b20d9eced0
udagram-frontend:local
Running
8100
8 minutes ago
backend-feed-1
8774e5400897
udagram-api-feed:latest
Running
8 minutes ago
RAM 1.83GB
CPU 0.22%
Connected to Hub
v4.12.0
CREATE A NEW
LOG OUT



Part 4 - Container Orchestration with Kubernetes

1. Create the EKS Cluster
2. Create the EKS Node Groups
3. Connect kubectl with EKS
    - Deployment
    - Connect to the Kubernetes Services to Access the Application
    - Expose External IP
    - Update the Environment Variables and Re-Deploy the Application
    - Troubleshoot

Command Lines for Part 4

/*Creating the cluster and its node groups*/

eksctl create cluster --name myCluster --region=us-east-1 --nodes-min=2 --nodes-max=3

docker image prune --all

docker-compose -f docker-compose-build.yaml build --parallel

docker-compose up

docker-compose down

source set_env.sh

kubectl apply -f aws-secret.yaml

kubectl apply -f env-secret.yaml

kubectl apply -f env-configmap.yaml

kubectl apply -f backend-feed-deployment.yaml

kubectl apply -f backend-user-deployment.yaml

kubectl apply -f frontend-deployment.yaml

kubectl apply -f reverseproxy-deployment.yaml

kubectl apply -f backend-feed-service.yaml

kubectl apply -f backend-user-service.yaml

kubectl apply -f frontend-service.yaml

kubectl apply -f reverseproxy-service.yaml

kubectl get nodes

kubectl get deployments

kubectl expose deployment frontend --type=LoadBalancer --name=publicfrontend2

kubectl expose deployment reverseproxy --type=LoadBalancer --name=publicreverseproxy2

kubectl get services

kubectl get pods

/*Frontend Directory*/

docker build . -t motaung/udagram-frontend:v4

docker push motaung/udagram-frontend:v4

/*Deployment Directory*/

kubectl set image deployment frontend frontend=motaung/udagram-frontend:v4

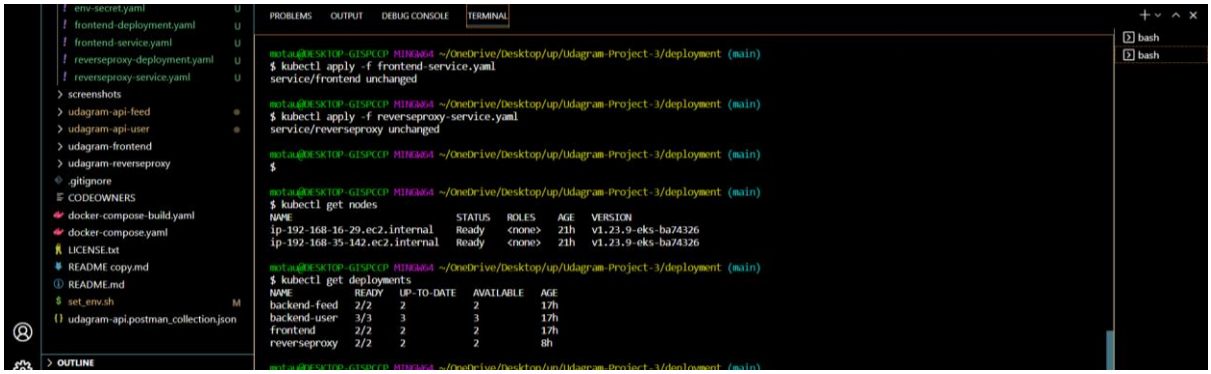
kubectl describe services

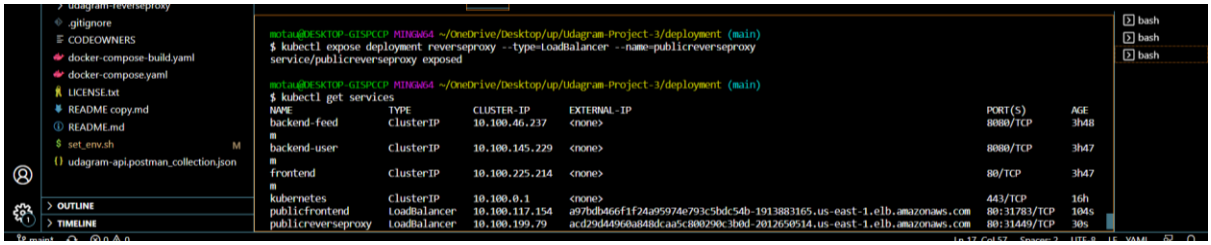
/*Horizontal Scaling*/

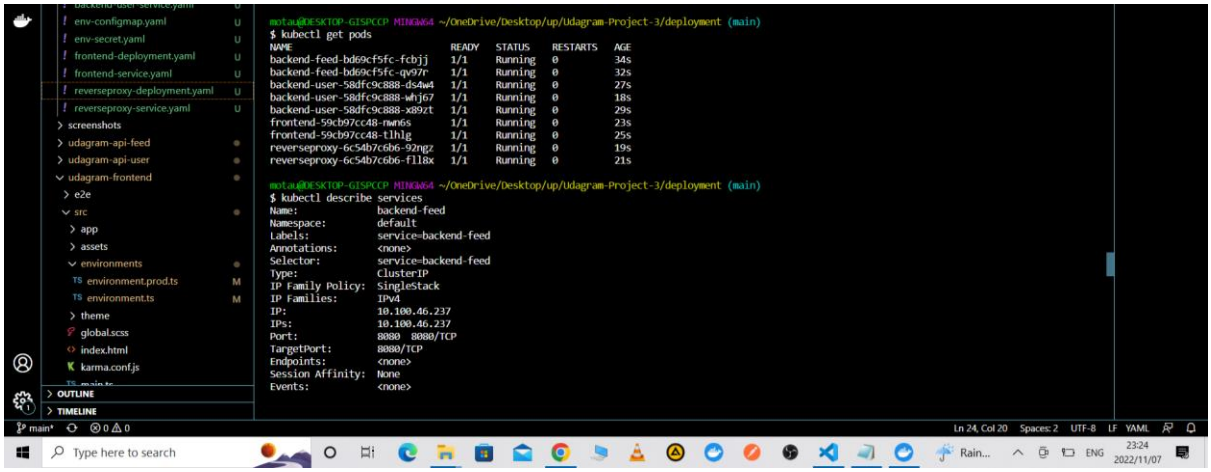
eksctl create cluster --name LCluster --region=us-east-1 --nodes-min=2 --nodes-max=5

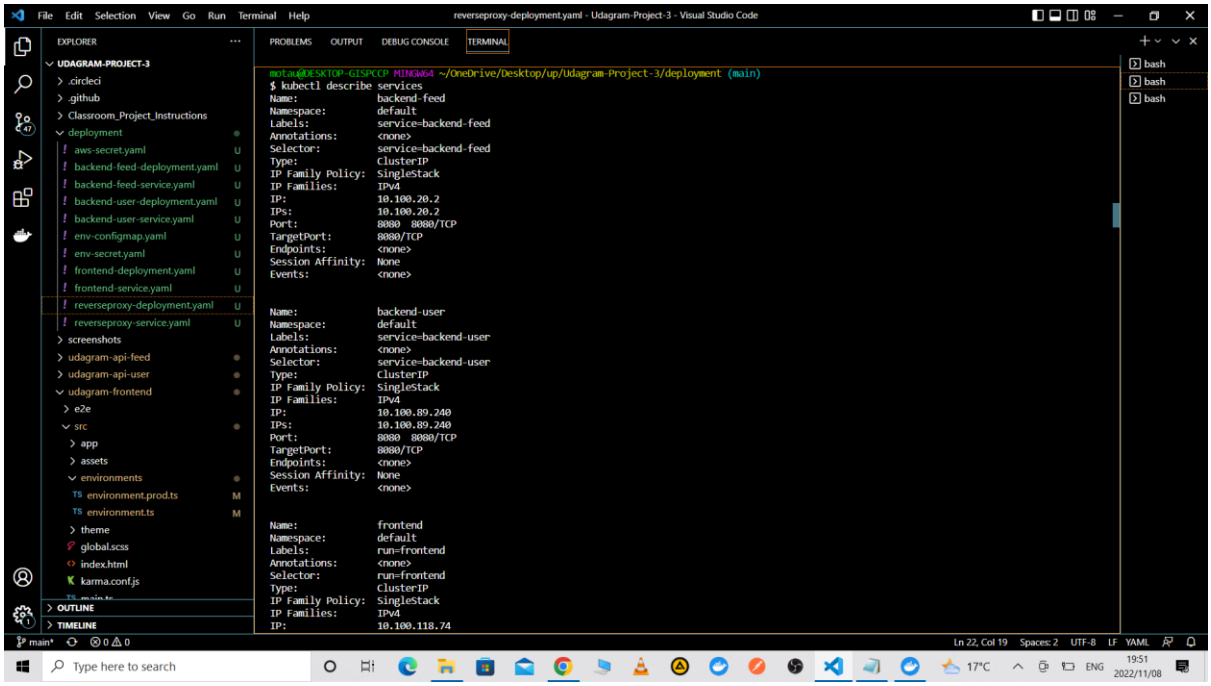
kubectl describe hpa

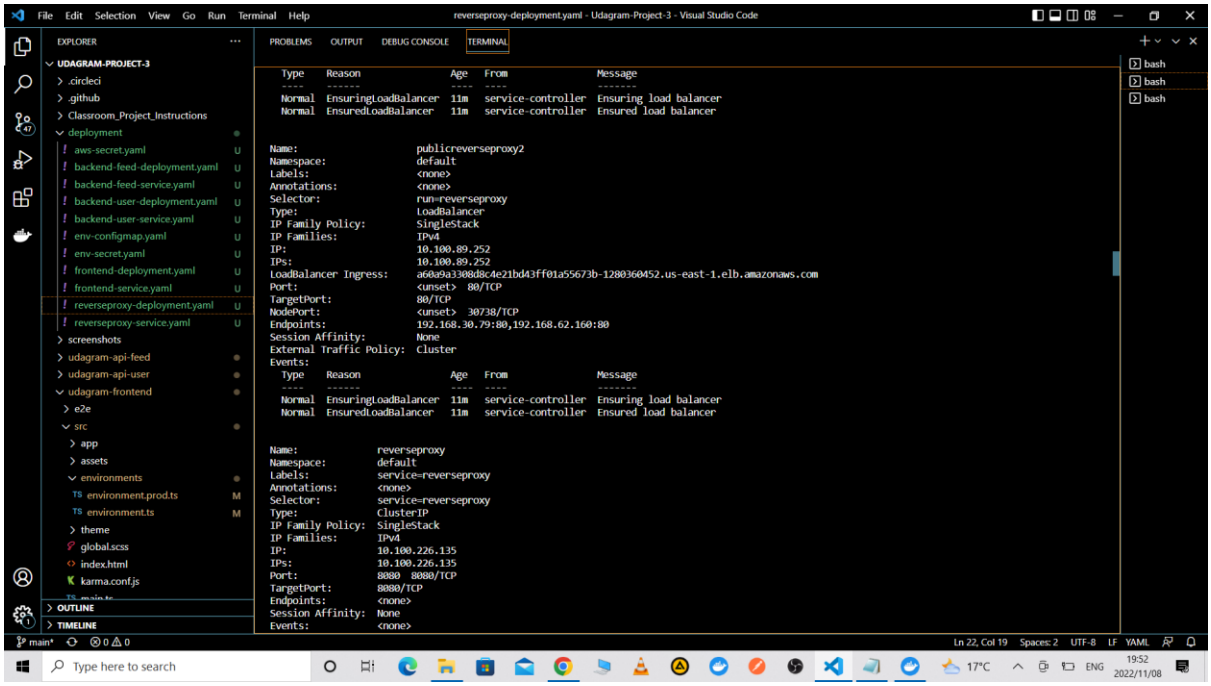
Screenshots for Part 4

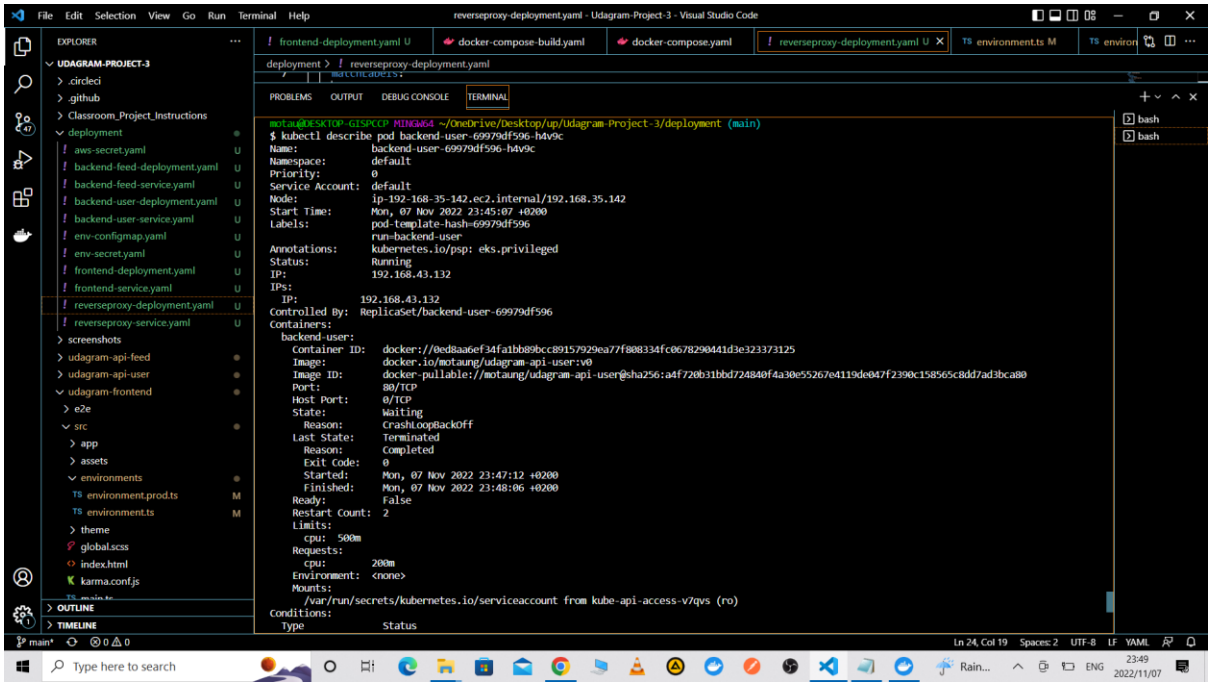


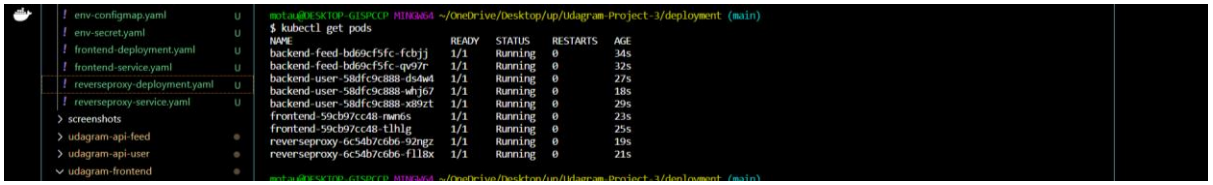


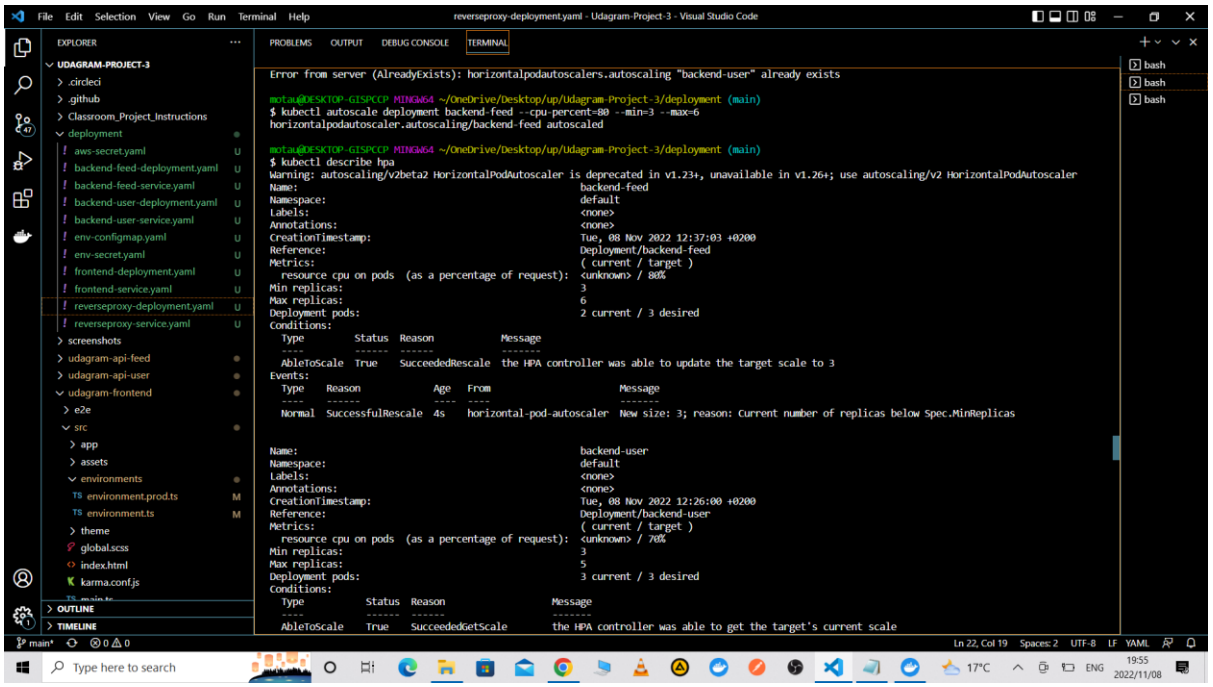


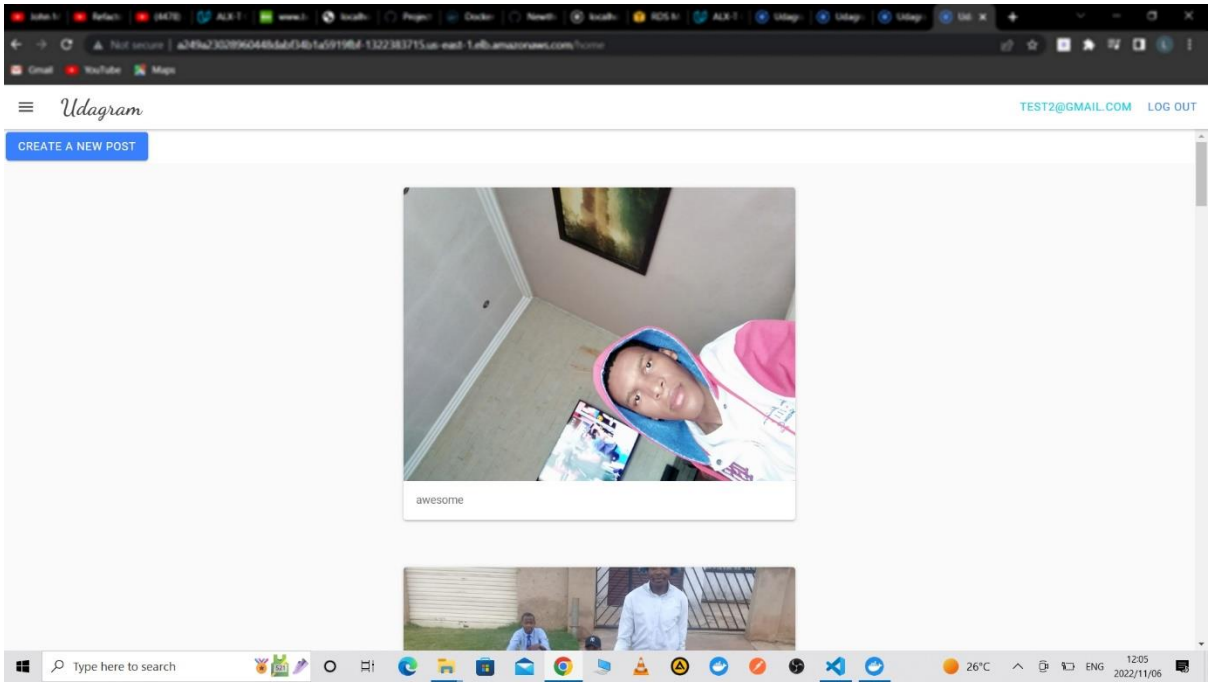

File Edit Selection View Go Run Terminal Help
reverseproxy-deployment.yaml - Udagram-Project-3 - Visual Studio Code
EXPLORER
UDAGRAM-PROJECT-3
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
motau@DESKTOP-GI5PCCP MINGW64 ~/OneDrive/Desktop/up/Udagram-Project-3/deployment (main)
$ kubectl describe services
Name: backend-feed
Namespace: default
Labels: service=backend-feed
Annotations: <none>
Selector: service=backend-feed
Type: ClusterIP
IP Family Policy: SingleStack
IP Families: IPv4
IP: 10.100.20.2
IPs: 10.100.20.2
Port: 8080 8080/TCP
TargetPort: 8080/TCP
Endpoints: <none>
Session Affinity: None
Events: <none>
Name: backend-user
Namespace: default
Labels: service=backend-user
Annotations: <none>
Selector: service=backend-user
Type: ClusterIP
IP Family Policy: SingleStack
IP Families: IPv4
IP: 10.100.89.240
IPs: 10.100.89.240
Port: 8080 8080/TCP
TargetPort: 8080/TCP
Endpoints: <none>
Session Affinity: None
Events: <none>
Name: frontend
Namespace: default
Labels: run=frontend
Annotations: <none>
Selector: run=frontend
Type: ClusterIP
IP Family Policy: SingleStack
IP Families: IPv4
IP: 10.100.118.74
bash
bash
bash
Ln 22, Col 19 Spaces: 2 UTF-8 LF YAML
Type here to search
17°C
ENG
19:51
2022/11/08


File Edit Selection View Go Run Terminal Help
reverseproxy-deployment.yaml - Udagram-Project-3 - Visual Studio Code
EXPLORER
UDAGRAM-PROJECT-3
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Type Reason Age From Message
Normal EnsuringLoadBalancer 11m service-controller Ensuring load balancer
Normal EnsuredLoadBalancer 11m service-controller Ensured load balancer
Name: publicreverseproxy2
Namespace: default
Labels: <none>
Annotations: <none>
Selector: run=reverseproxy
Type: LoadBalancer
IP Family Policy: SingleStack
IP Families: IPv4
IP: 10.100.89.252
IPs: 10.100.89.252
LoadBalancer Ingress: a60a9a3308d8c4e21bd43ff01a55673b-1280360452.us-east-1.elb.amazonaws.com
Port: <unset> 80/TCP
TargetPort: 80/TCP
NodePort: <unset> 30738/TCP
Endpoints: 192.168.30.79:80,192.168.62.160:80
Session Affinity: None
External Traffic Policy: Cluster
Events:
Type Reason Age From Message
Normal EnsuringLoadBalancer 11m service-controller Ensuring load balancer
Normal EnsuredLoadBalancer 11m service-controller Ensured load balancer
Name: reverseproxy
Namespace: default
Labels: service=reverseproxy
Annotations: <none>
Selector: service=reverseproxy
Type: ClusterIP
IP Family Policy: SingleStack
IP Families: IPv4
IP: 10.100.226.135
IPs: 10.100.226.135
Port: 8080 8080/TCP
TargetPort: 8080/TCP
Endpoints: <none>
Session Affinity: None
Events: <none>
bash
bash
bash
Ln 22, Col 19 Spaces: 2 UTF-8 LF YAML
Type here to search
17°C
ENG
19:52
2022/11/08


File Edit Selection View Go Run Terminal Help
reverseproxy-deployment.yaml - Udagram-Project-3 - Visual Studio Code
EXPLORER
UDAGRAM-PROJECT-3
frontend-deployment.yaml U
docker-compose-build.yaml
docker-compose.yaml
reverseproxy-deployment.yaml U
environment.ts M
deployment > ! reverseproxy-deployment.yaml
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
bash
bash
motau@DESKTOP-GISPCCP MINGW64 ~/OneDrive/Desktop/up/Udagram-Project-3/deployment (main)
$ kubectl describe pod backend-user-69979df596-h4v9c
Name: backend-user-69979df596-h4v9c
Namespace: default
Priority: 0
Service Account: default
Node: ip-192-168-35-142.ec2.internal/192.168.35.142
Start Time: Mon, 07 Nov 2022 23:45:07 +0200
Labels: pod-template-hash=69979df596
run=backend-user
Annotations: kubernetes.io/psp: eks.privileged
Status: Running
IP: 192.168.43.132
IPs:
IP: 192.168.43.132
Controlled By: ReplicaSet/backend-user-69979df596
Containers:
backend-user:
Container ID: docker://0ed8aa6ef34fa1bb89bcc89157929ea77f808334fc0678290441d3e323373125
Image: docker.io/motaung/udagram-api-user:v0
Image ID: docker-pullable://motaung/udagram-api-user@sha256:a4f720b31bbd724840f4a30e55267e4119de047f2390c158565c8dd7ad3bca80
Port: 80/TCP
Host Port: 0/TCP
State: Waiting
Reason: CrashLoopBackOff
Last State: Terminated
Reason: Completed
Exit Code: 0
Started: Mon, 07 Nov 2022 23:47:12 +0200
Finished: Mon, 07 Nov 2022 23:48:06 +0200
Ready: False
Restart Count: 2
Limits:
cpu: 500m
Requests:
cpu: 200m
Environment: <none>
Mounts:
/var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-v7qvs (ro)
Conditions:
Type Status
OUTLINE
TIMELINE
main*
Ln 24, Col 19 Spaces: 2 UTF-8 LF YAML
Type here to search
Rain...
ENG
23:49
2022/11/07


env-configmap.yaml U
env-secret.yaml U
frontend-deployment.yaml U
frontend-service.yaml U
reverseproxy-deployment.yaml U
reverseproxy-service.yaml U
screenshots
udagram-api-feed
udagram-api-user
udagram-frontend
motau@DESKTOP-GISPCCP MINGW64 ~/OneDrive/Desktop/up/Udagram-Project-3/deployment (main)
$ kubectl get pods
NAME READY STATUS RESTARTS AGE
backend-feed-bd69cf5fc-fcbjj 1/1 Running 0 34s
backend-feed-bd69cf5fc-qv97r 1/1 Running 0 32s
backend-user-58dfc9c888-ds4w4 1/1 Running 0 27s
backend-user-58dfc9c888-whj67 1/1 Running 0 18s
backend-user-58dfc9c888-x89zt 1/1 Running 0 29s
frontend-59cb97cc48-nwn6s 1/1 Running 0 23s
frontend-59cb97cc48-tlhlg 1/1 Running 0 25s
reverseproxy-6c54b7c6b6-92ngz 1/1 Running 0 19s
reverseproxy-6c54b7c6b6-fll8x 1/1 Running 0 21s


File Edit Selection View Go Run Terminal Help
reverseproxy-deployment.yaml - Udagram-Project-3 - Visual Studio Code
EXPLORER
UDAGRAM-PROJECT-3
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
bash
bash
bash
Error from server (AlreadyExists): horizontalpodautoscalers.autoscaling "backend-user" already exists
motau@DESKTOP-GISPCCP MINGW64 ~/OneDrive/Desktop/up/Udagram-Project-3/deployment (main)
$ kubectl autoscale deployment backend-feed --cpu-percent=80 --min=3 --max=6
horizontalpodautoscaler.autoscaling/backend-feed autoscaled
motau@DESKTOP-GISPCCP MINGW64 ~/OneDrive/Desktop/up/Udagram-Project-3/deployment (main)
$ kubectl describe hpa
Warning: autoscaling/v2beta2 HorizontalPodAutoscaler is deprecated in v1.23+, unavailable in v1.26+; use autoscaling/v2 HorizontalPodAutoscaler
Name: backend-feed
Namespace: default
Labels: <none>
Annotations: <none>
CreationTimestamp: Tue, 08 Nov 2022 12:37:03 +0200
Reference: Deployment/backend-feed
Metrics: (current / target)
resource cpu on pods (as a percentage of request): <unknown> / 80%
Min replicas: 3
Max replicas: 6
Deployment pods: 2 current / 3 desired
Conditions:
Type Status Reason Message
AbleToScale True SucceededRescale the HPA controller was able to update the target scale to 3
Events:
Type Reason Age From Message
Normal SuccessfulRescale 4s horizontal-pod-autoscaler New size: 3; reason: Current number of replicas below Spec.MinReplicas
Name: backend-user
Namespace: default
Labels: <none>
Annotations: <none>
CreationTimestamp: Tue, 08 Nov 2022 12:26:00 +0200
Reference: Deployment/backend-user
Metrics: (current / target)
resource cpu on pods (as a percentage of request): <unknown> / 70%
Min replicas: 3
Max replicas: 5
Deployment pods: 3 current / 3 desired
Conditions:
Type Status Reason Message
AbleToScale True SucceededGetScale the HPA controller was able to get the target's current scale
OUTLINE
TIMELINE
main*
Ln 22, Col 19 Spaces: 2 UTF-8 LF YAML
Type here to search
17°C
ENG
19:55
2022/11/08



Part 5. Logging

kubectl logs <your pod name>