

Refactor Monolith to Microservices and Deploy

(Lưu ý: tài liệu viết 2 lần deploy khác nhau có thể có giá trị không mapping với nhau vd: name images docker reverseproxy)

Chúc mọi người sớm pass project – có thắc mắc liên hệ NhanNV13

Create an S3 bucket

General configuration

Bucket name

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

AWS Region

Copy settings from existing bucket - *optional*
Only the bucket settings in the following configuration are copied.

Object Ownership [Info](#)

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

☒ **ACLs disabled (recommended)**
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

☐ **ACLs enabled**
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Object Ownership
Bucket owner enforced

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

☐ **Block all public access**
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

☐ **Block public access to buckets and objects granted through new access control lists (ACLs)**
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.

☐ **Block public access to buckets and objects granted through any access control lists (ACLs)**
S3 will ignore all ACLs that grant public access to buckets and objects.

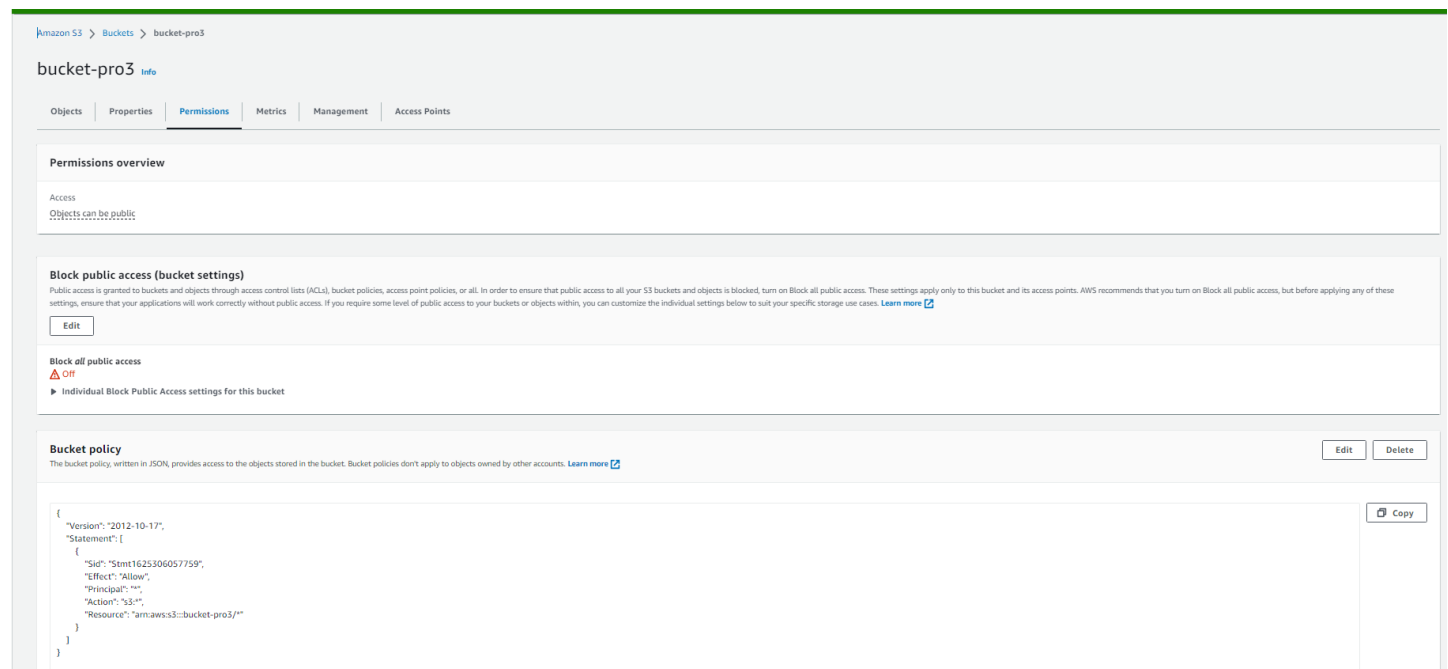
☐ **Block public access to buckets and objects granted through new public bucket or access point policies**
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

Navigate to S3 from the AWS console.

Create a public S3 bucket with default configuration, such as no versioning and disabled encryption.

Once your bucket is created, go to the Permissions tab. Add bucket policy allowing other AWS services (Kubernetes) to access the bucket contents

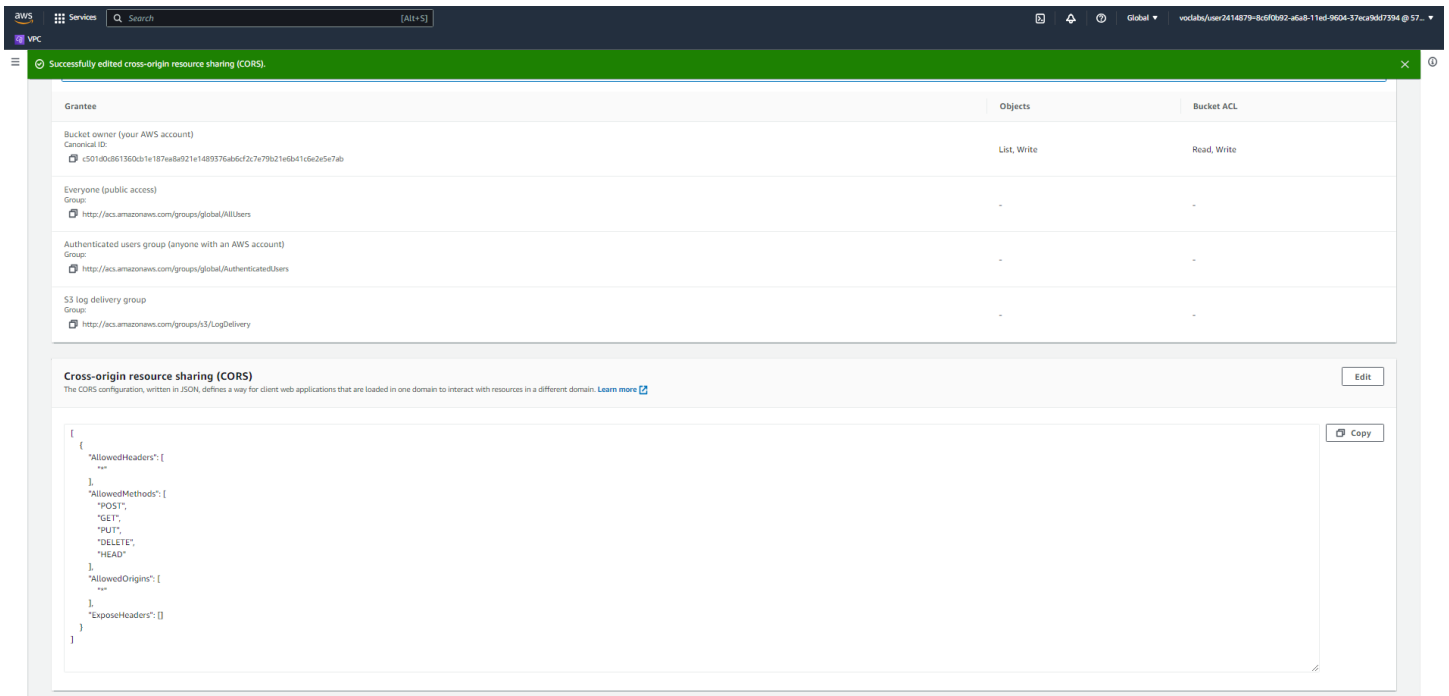
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1625306057759",
      "Principal": "*",
      "Action": "s3:*",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::[bucket-name]"
    }
  ]
}
```



The screenshot shows the AWS S3 console interface for a bucket named 'bucket-pro3'. The 'Permissions' tab is selected, displaying the 'Permissions overview' section. It indicates that 'Access' is set to 'Objects can be public'. Below this, the 'Block public access (bucket settings)' section shows that 'Block all public access' is currently 'Off'. The 'Bucket policy' section is visible at the bottom, showing a JSON policy that grants 's3:*' actions to all principals ('*') on the bucket and its contents. The policy is as follows:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1625306057759",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::bucket-pro3/*"
    }
  ]
}
```

Add the CORS configuration to allow the application running outside of AWS to interact with your bucket. You can use the following configuration:










The screenshot shows the AWS IAM console interface. At the top, there's a navigation bar with the AWS logo, 'Services' link, a search bar, and a user profile dropdown. Below the navigation bar, a green banner indicates 'Successfully edited cross-origin resource sharing (CORS)'. The main content area is divided into two sections. The first section is a table with three columns: 'Grantee', 'Objects', and 'Bucket ACL'. It lists four grantees: 'Bucket owner (your AWS account)', 'Everyone (public access)', 'Authenticated users group (anyone with an AWS account)', and 'SS log delivery group'. The second section is titled 'Cross-origin resource sharing (CORS)' and contains a JSON configuration for the bucket's CORS settings. The JSON defines allowed headers, methods (POST, GET, PUT, DELETE, HEAD), and origins. There is an 'Edit' button in the top right of the CORS section and a 'Copy' button next to the JSON code.

Grantee	Objects	Bucket ACL
Bucket owner (your AWS account) Canonical ID: a50180d861360b1e187ea8d21e1489376ab6f2c7e79b21e6b41c6e2e5e7ab	List, Write	Read, Write
Everyone (public access) Group: http://acs.amazonaws.com/groups/global/AllUsers	-	-
Authenticated users group (anyone with an AWS account) Group: http://acs.amazonaws.com/groups/global/AuthenticatedUsers	-	-
SS log delivery group Group: http://acs.amazonaws.com/groups/h3/LogDelivery	-	-

Cross-origin resource sharing (CORS)
The CORS configuration, written in JSON, defines a way for client web applications that are loaded in one domain to interact with resources in a different domain. [Learn more](#)

```
[
  {
    "AllowedHeaders": [
      "*"
    ],
    "AllowedMethods": [
      "POST",
      "GET",
      "PUT",
      "DELETE",
      "HEAD"
    ],
    "AllowedOrigins": [
      "*"
    ],
    "ExposeHeaders": []
  }
]
```

Create PostgreSQL database

<input type="radio"/> Aurora (MySQL Compatible) 	<input type="radio"/> Aurora (PostgreSQL Compatible) 	<input type="radio"/> MySQL 
<input type="radio"/> MariaDB 	<input checked="" type="radio"/> PostgreSQL 	<input type="radio"/> Oracle 
<input type="radio"/> Microsoft SQL Server 		

▼ Hide filters

☒ Show versions that support the Multi-AZ DB cluster [Info](#)
Create a Multi-AZ DB cluster with one primary DB instance and two readable standby DB instances. Multi-AZ DB clusters provide up to 2x faster transaction commit latency and automatic failover in typically under 35 seconds.

Engine Version

PostgreSQL 13.7-R1 ▼

Templates

Choose a sample template to meet your use case.

<input type="radio"/> Production Use defaults for high availability and fast, consistent performance.	<input type="radio"/> Dev/Test This instance is intended for development use outside of a production environment.	<input checked="" type="radio"/> Free tier Use RDS Free Tier to develop new applications, test existing applications, or gain hands-on experience with Amazon RDS. Info
--	--	---

Settings

DB instance identifier [Info](#)

Type a name for your DB instance. The name must be unique across all DB instances owned by your AWS account in the current AWS Region.

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 60 alphanumeric characters or hyphens. First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

▼ Credentials Settings


Master username [Info](#)


Type a login ID for the master user of your DB instance.

1 to 16 alphanumeric characters. The first character must be a letter.

☐ **Manage master credentials in AWS Secrets Manager**

Manage master user credentials in Secrets Manager. RDS can generate a password for you and manage it throughout its lifecycle.

 If you manage the master user credentials in Secrets Manager, some RDS features aren't supported.

[Learn more](#) 

☐ **Auto generate a password**

Amazon RDS can generate a password for you, or you can specify your own password.

Master password [Info](#)

Constraints: At least 8 printable ASCII characters. Can't contain any of the following: / (slash), ' (single quote), " (double quote) and @ (at sign).

Confirm master password [Info](#)

Default VPC (vpc-076f8e9b95b27e930)

6 Subnets, 6 Availability Zones

Only VPCs with a corresponding DB subnet group are listed.

i After a database is created, you can't change its VPC.

DB subnet group [Info](#)

Choose the DB subnet group. The DB subnet group defines which subnets and IP ranges the DB instance can use in the VPC that you selected.

default

Public access [Info](#)

☒ Yes

RDS assigns a public IP address to the database. Amazon EC2 instances and other resources outside of the VPC can connect to your database. Resources inside the VPC can also connect to the database. Choose one or more VPC security groups that specify which resources can connect to the database.

☐ No

RDS doesn't assign a public IP address to the database. Only Amazon EC2 instances and other resources inside the VPC can connect to your database. Choose one or more VPC security groups that specify which resources can connect to the database.

VPC security group (firewall) [Info](#)

Choose one or more VPC security groups to allow access to your database. Make sure that the security group rules allow the appropriate incoming traffic.

☐ Choose existing

Choose existing VPC security groups

☒ Create new

Create new VPC security group

New VPC security group name

postgres

Availability Zone [Info](#)

No preference

RDS Proxy

RDS Proxy is a fully managed, highly available database proxy that improves application scalability, resiliency, and security.

☐ Create an RDS Proxy [Info](#)

RDS automatically creates an IAM role and a Secrets Manager secret for the proxy. RDS Proxy has additional costs. For more information, see [Amazon RDS Proxy pricing](#).

Certificate authority - optional [Info](#)

Using a server certificate provides an extra layer of security by validating that the connection is being made to an Amazon database. It does so by checking the server certificate that is automatically installed on all databases that you provision.

rds-ca-2019 (default)

If you don't select a certificate authority, RDS chooses one for you.

Monitoring

☐ Turn on Performance Insights

► Additional configuration

Enhanced Monitoring

► Additional configuration

Database options, encryption turned on, backup turned on, backtrack turned off, maintenance, CloudWatch Logs, delete protection turned off.


Estimated monthly costs

The Amazon RDS Free Tier is available to you for 12 months. Each calendar month, the free tier will allow you to use the Amazon RDS resources listed below for free:

- 750 hrs of Amazon RDS in a Single-AZ db.t2.micro, db.t3.micro or db.t4g.micro Instance.
- 20 GB of General Purpose Storage (SSD).
- 20 GB for automated backup storage and any user-initiated DB Snapshots.

[Learn more about AWS Free Tier.](#)

When your free usage expires or if your application use exceeds the free usage tiers, you simply pay standard, pay-as-you-go service rates as described in the [Amazon RDS Pricing page](#).

 You are responsible for ensuring that you have all of the necessary rights for any third-party products or services that you use with AWS services.

Cancel

Create database

Connectivity & security

Endpoint & port

Endpoint
database-1.chptxenne7vq.us-east-1.rds.amazonaws.com
Port
5434

Networking

Availability Zone
us-east-1a
VPC
vpc-076f8e9b95b27e930
Subnet group
default-vpc-076f8e9b95b27e930
Subnets
subnet-06b8bfab50e1f7a08
subnet-06b2a5c53ba2da95
subnet-02f8741173b943caf
subnet-078daa305a4424834
subnet-02eff58b4c319ab6c
subnet-08675fb2db5734d7a
Network type
IPv4

Security

VPC security groups
postgres (sg-0d26d0f328721a3b4)
Active
Publicly accessible
Yes
Certificate authority
info
rds-ca-2019
Certificate authority date
August 23, 2024, 00:08 (UTC+07:00)
DB instance certificate expiration date
August 23, 2024, 00:08 (UTC+07:00)

Security group rules (2)

< 1 >

Security group	Type	Rule
postgres (sg-0d26d0f328721a3b4)	CIDR/IP - Inbound	203.205.32.81/32
postgres (sg-0d26d0f328721a3b4)	CIDR/IP - Outbound	0.0.0.0/0

Security Groups (1/1) info

search: postgres X Clear filters

☒
Name
Security group ID
Security group name
VPC ID
Description
Owner
Inbound rules count
Outbound rules count

<input checked="" type="checkbox"/>	-	sg-0d26d0f328721a3b4	postgres	vpc-076f8e9b95b27e930	Created by RDS manag...	572919122255	1 Permission entry	1 Permission entry
-------------------------------------	---	----------------------	----------	-----------------------	-------------------------	--------------	--------------------	--------------------

sg-0d26d0f328721a3b4 - postgres

Details
Inbound rules
Outbound rules
Tags

You can now check network connectivity with Reachability Analyzer

Run Reachability Analyzer X

Inbound rules (1/1)

< 1 >

<input checked="" type="checkbox"/>	Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
<input checked="" type="checkbox"/>	-	sg-06278198a1672ee...	IPv4	All traffic	All	All	0.0.0.0/0	-

```
ubuntu@ip-172-31-40-56:~$ psql -h database-1.chptxenne7vq.us-east-1.rds.amazonaws.com -U postgres
Password for user postgres:
psql (15.4 (Ubuntu 15.4-1.pgdg20.04+1), server 13.7)
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, compression: off)
Type "help" for help.

postgres=> |
```

Install docker

```
https://docs.docker.com/engine/install/ubuntu/
curl -fsSL https://get.docker.com -o get-docker.sh
sudo sh get-docker.sh
```



```
ubuntu@ip-172-31-40-56:~$ curl -fsSL https://get.docker.com -o get-docker.sh
ubuntu@ip-172-31-40-56:~$ sudo sh get-docker.sh
# Executing docker install script, commit: c2de0811708b6d9015ed1a2c80f02c9b70c8ce7b
+ sh -c apt-get update -qq >/dev/null
+ sh -c DEBIAN_FRONTEND=noninteractive apt-get install -y -qq apt-transport-https ca-certificates curl >/dev/null
+ sh -c install -m 0755 -d /etc/apt/keyrings
+ sh -c curl -fsSL "https://download.docker.com/linux/ubuntu/gpg" | gpg --dearmor --yes -o /etc/apt/keyrings/docker.gpg
+ sh -c chmod a+r /etc/apt/keyrings/docker.gpg
+ sh -c echo "deb [arch=amd64 signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu focal stable" > /etc/apt/sources.list.d/docker.list
+ sh -c apt-get update -qq >/dev/null
+ sh -c DEBIAN_FRONTEND=noninteractive apt-get install -y -qq docker-ce docker-ce-cli containerd.io docker-compose-plugin docker-ce-rootless-extras docker-buildx-plugin >/dev/null
```

sudo chmod 666 /var/run/docker.sock

```
ubuntu@ip-172-31-40-56:~$ docker -v
Docker version 24.0.5, build ced0996
ubuntu@ip-172-31-40-56:~$
```

<https://hub.docker.com/>

export DOCKER_USERNAME=<your_username>

export DOCKER_PASSWORD=<your_password>

docker login -u="\${DOCKER_USERNAME}" -p="\${DOCKER_PASSWORD}"

```
ubuntu@ip-172-31-40-56:~$ docker login -u="${DOCKER_USERNAME}" -p="${DOCKER_PASSWORD}"
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /home/ubuntu/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
ubuntu@ip-172-31-40-56:~$
```

Install kubectl

<https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/>

sudo apt-get update

sudo apt-get install -y apt-transport-https ca-certificates curl

curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.28/deb/Release.key | sudo gpg --dearmor -o
/etc/apt/keyrings/kubernetes-apt-keyring.gpg

echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.28/deb/ ' |
sudo tee /etc/apt/sources.list.d/kubernetes.list

sudo apt-get update

sudo apt-get install -y kubectl

```
ubuntu@ip-172-31-40-56:~$ kubectl version --client
Client Version: v1.28.0
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
ubuntu@ip-172-31-40-56:~$
```

Install awscli

sudo apt install awscli

```
ubuntu@ip-172-31-40-56:~/project3$ aws -v
aws-cli/2.0.18 Python/3.6.5 Linux/x86_64 prompt/off
Note: AWS CLI version 2, the latest major version of the AWS CLI is available at
https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip.
For more information, see https://awscli.amazonaws.com/awscli-install-cliv2.html

Usage: aws [options] <command> [<subcommand> [<subcommand> ...]]
For more information on a specific command, see the command help text.
For more information on awscli, see help text, you can run:
aws help
aws <command> help
aws <command> <subcommand> help
aws: error: the following arguments are required: command
ubuntu@ip-172-31-40-56:~/project3$
```

Refactor the Backend API

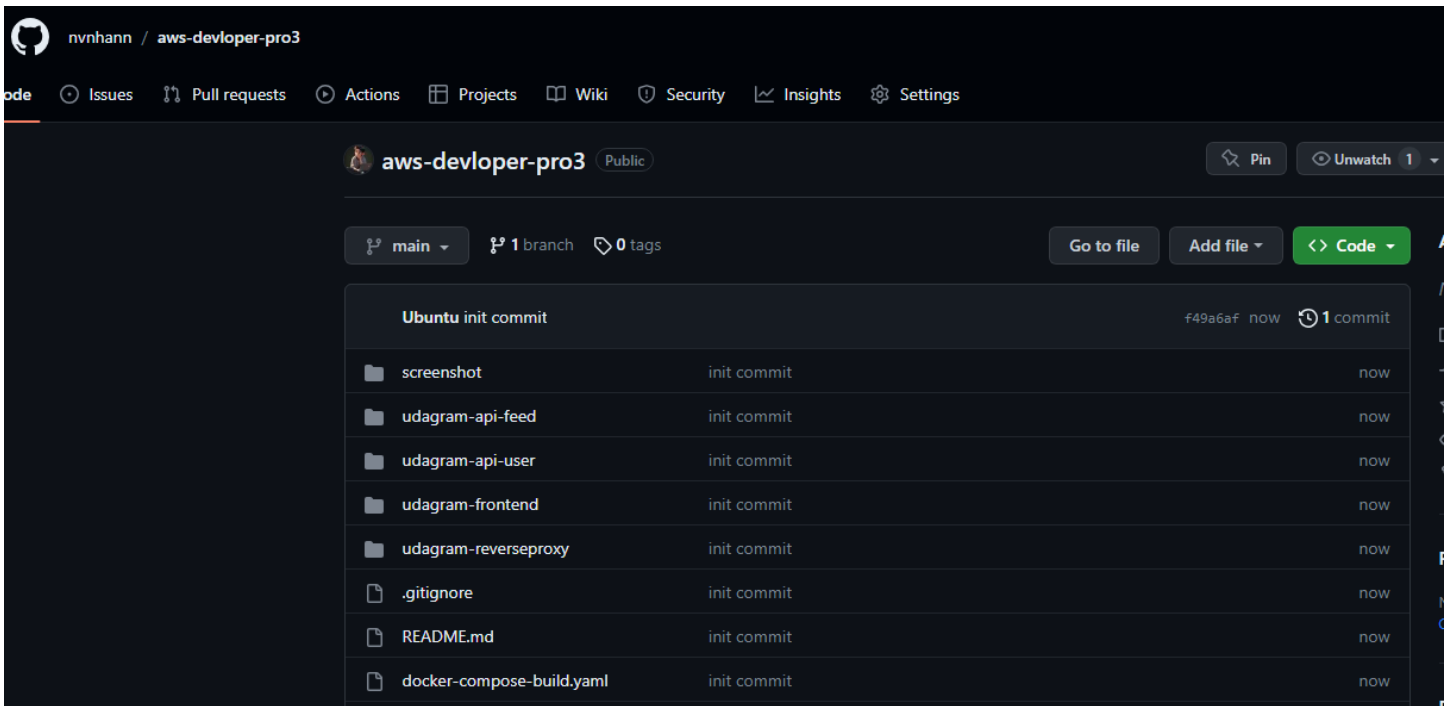
```
ubuntu@ip-172-31-40-56:~$ git clone https://github.com/udacity/cd0354-monolith-to-microservices-project.git project3
Cloning into 'project3'...
remote: Enumerating objects: 197, done.
remote: Total 197 (delta 0), reused 0 (delta 0), pack-reused 197
Receiving objects: 100% (197/197), 5.63 MiB | 30.53 MiB/s, done.
Resolving deltas: 100% (29/29), done.
ubuntu@ip-172-31-40-56:~$ tree -L 2 project3/
project3/
├── CODEOWNERS
├── Classroom_Project_Instructions
│   ├── Part_0_Prerequisites_and_Getting_Started.md
│   ├── Part_III_TravisCI.md
│   ├── Part_II_Microservices_Application.md
│   ├── Part_IV_Container_Orchestration.md
│   ├── Part_I_Monolithic_Application.md
│   └── Part_V_Logging.md
├── LICENSE.txt
├── README.md
├── screenshots
│   └── README.md
├── set_env.sh
├── udagram-api
│   ├── mock
│   ├── package-lock.json
│   ├── package.json
│   ├── src
│   ├── tsconfig.json
│   └── tslint.json
├── udagram-api.postman_collection.json
├── udagram-frontend
│   ├── angular.json
│   ├── e2e
│   ├── ionic.config.json
│   ├── package-lock.json
│   ├── package.json
│   ├── src
│   ├── tsconfig.json
│   ├── tslint.json
│   └── udagram_tests
└── 9 directories, 22 files
ubuntu@ip-172-31-40-56:~$
```

Refer: <https://github.com/nvnhanh/aws-devloper-pro3.git>

```
ubuntu@ip-172-31-40-56:~$ git clone https://github.com/nvnhanh/Cloud-developer.git
Cloning into 'Cloud-developer'...
remote: Enumerating objects: 318, done.
remote: Counting objects: 100% (318/318), done.
remote: Compressing objects: 100% (240/240), done.
remote: Total 318 (delta 56), reused 318 (delta 56), pack-reused 0
Receiving objects: 100% (318/318), 11.96 MiB | 32.47 MiB/s, done.
Resolving deltas: 100% (56/56), done.
ubuntu@ip-172-31-40-56:~$ cp Cloud-developer/Refactor-Monolith-to-Microservices/ .
cp: -r not specified; omitting directory 'Cloud-developer/Refactor-Monolith-to-Microservices/'
ubuntu@ip-172-31-40-56:~$ cp -r Cloud-developer/Refactor-Monolith-to-Microservices/ .
ubuntu@ip-172-31-40-56:~$ mv Refactor-Monolith-to-Microservices/ project3
ubuntu@ip-172-31-40-56:~$ tree -L 2 project3/
project3/
├── README.md
├── docker-compose-build.yaml
├── docker-compose.yaml
├── screenshot
│   ├── build CI.png
│   ├── describe_hpa.png
│   ├── describe_service_1.png
│   ├── describe_service_2.png
│   ├── describe_service_3.png
│   ├── dockerhub.png
│   ├── kubectl_get_info.png
│   ├── newpost.png
│   ├── pod_pogs.png
│   ├── post_success.png
│   └── register.png
├── udagram-api-feed
│   ├── Dockerfile
│   ├── mock
│   ├── package-lock.json
│   ├── package.json
│   ├── src
│   ├── tsconfig.json
│   └── tslint.json
├── udagram-api-user
│   ├── Dockerfile
│   ├── mock
│   ├── package-lock.json
│   ├── package.json
│   ├── src
│   ├── tsconfig.json
│   └── tslint.json
└── udagram-frontend
    ├── Dockerfile
    ├── angular.json
    └── e2e
```

```
ubuntu@ip-172-31-40-56:~$ tree -L 2 project3/
project3/
├── README.md
├── docker-compose-build.yaml
├── docker-compose.yaml
├── screenshot
│   ├── build_CI.png
│   ├── describe_hpa.png
│   ├── describe_service_1.png
│   ├── describe_service_2.png
│   ├── describe_service_3.png
│   ├── dockerhub.png
│   ├── kubectl_get_info.png
│   ├── newpost.png
│   ├── pod_pogs.png
│   ├── post_success.png
│   └── register.png
├── udagram-api-feed
│   ├── Dockerfile
│   ├── mock
│   ├── package-lock.json
│   ├── package.json
│   ├── src
│   ├── tsconfig.json
│   └── tslint.json
├── udagram-api-user
│   ├── Dockerfile
│   ├── mock
│   ├── package-lock.json
│   ├── package.json
│   ├── src
│   ├── tsconfig.json
│   └── tslint.json
├── udagram-frontend
│   ├── Dockerfile
│   ├── angular.json
│   ├── e2e
│   ├── ionic.config.json
│   ├── package-lock.json
│   ├── package.json
│   ├── src
│   ├── tsconfig.json
│   ├── tslint.json
│   └── udagram_tests
└── udagram-reverseproxy
    ├── Dockerfile
    └── nginx.conf

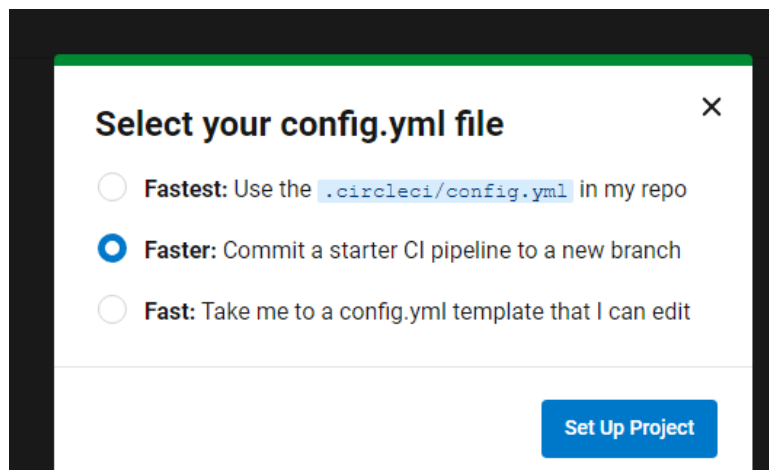
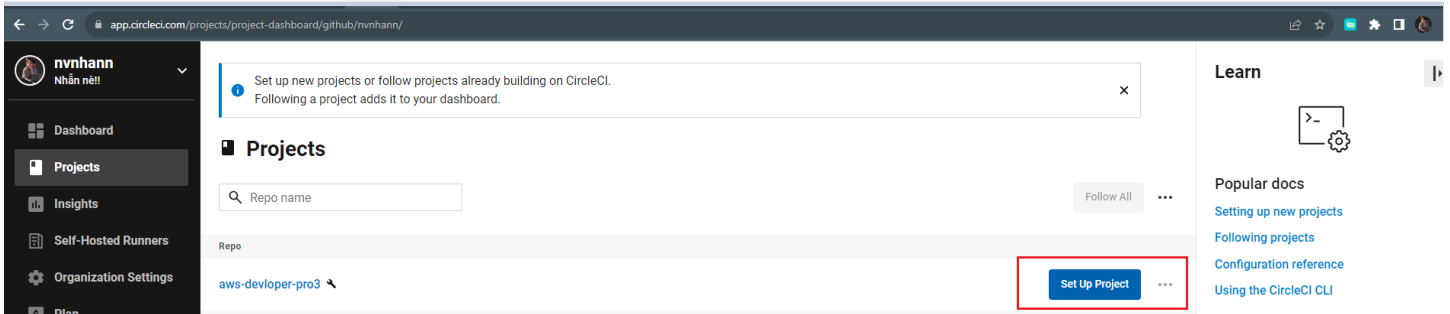
12 directories, 33 files
ubuntu@ip-172-31-40-56:~$
```



Continuous Integration

Setup circleci

<https://circleci.com/signup/>



Add environment

nvnhann

Nhân nết!

Dashboard

Projects FOLLOW 1 NEW

Insights

Self-Hosted Runners

Organization Settings

Plan

Flaky test failures wasting your time?

Only rerun failed tests

1

Maintenance Update

CircleCI scheduled maintenance is rescheduled for September 23, 2023 from 16:00-17:00 UTC instead of August 12, 2023 from 16:00-17:00 UTC. [Read More](#)

Dashboard

Project

All Pipelines

aws-developer-pro3

aws-developer-pro3

Add team members

Edit Config

Trigger Pipeline

Project Settings

Filters

Everyone's Pipelines

aws-developer-pro3

All Branches

All days

Auto-expand

Pipeline	Status	Workflow	Branch / Commit	Start	Duration	Actions
aws-developer-pro3 1	Success	say-hello-workflow	circleci-project-setup 3f1c420	21s ago	6s	

Project Settings

aws-developer-pro3

Overview

Triggers

Advanced

Environment Variables

SSH Keys

API Permissions

Jira Integration

Slack Integration

Insights Snapshot Badge

Status Badges

Webhooks

Docker Layer Caching

Environment Variables

Environment variables let you add sensitive data (e.g. API keys) to your jobs rather than placing them in the repository. The value of the variables cannot be read or edited in the app once they are set.

If you're looking to share environment variables across projects, try [Contexts](#).

Name	Value	Created at	Add Environment Variable	Import Variables
DOCKER_PASSWORD		Aug 23, 2023, 10:21:25 AM		
DOCKER_USERNAME		Aug 23, 2023, 10:21:11 AM		

Config circleci

```

Coding - Untitled-1

1  version: 2.1
2
3  jobs:
4    lint-app:
5      docker:
6        - image: circleci/node:16
7      steps:
8        - checkout
9    build-docker:
10     machine: true
11     steps:
12       - checkout
13       - run:
14         name: Build docker container for each microservices
15         command: |
16           echo "----- Installing dependencies-----"
17           curl -fsSL https://deb.nodesource.com/setup_16.x | sudo -E bash -
18           sudo apt-get install -y nodejs
19           node -v
20           npm -v
21
22           echo "----- The images is Building -----"
23
24           docker build -t udagram-api-feed ./udagram-api-feed
25           docker tag udagram-api-feed nvnhan/udagram-api-feed:v1
26
27           docker build -t udagram-api-user ./udagram-api-user
28           docker tag udagram-api-user nvnhan/udagram-api-user:v1
29
30           docker build -t udagram-frontend ./udagram-frontend
31           docker tag udagram-frontend nvnhan/udagram-frontend:v1
32
33           docker build -t udagram-reverseproxy ./udagram-reverseproxy
34           docker tag udagram-reverseproxy nvnhan/udagram-reverseproxy:v1
35
36           echo "----- All images succesfully built-----"
37
38           echo " ----- login in to hub-----"
39
40           docker login -u $DOCKER_USERNAME -p $DOCKER_PASSWORD
41
42           echo "----- Pushing images to dockerhub-----"
43
44           docker push nvnhan/udagram-api-feed:v1
45           docker push nvnhan/udagram-api-user:v1
46           docker push nvnhan/udagram-frontend:v1
47           docker push nvnhan/udagram-reverseproxy:v1
48
49   workflows:
50     default:
51       jobs:
52         - lint-app
53         - build-docker:
54           requires: [lint-app]

```

Building circleci

The screenshot shows a GitHub Actions workflow running on a self-hosted runner. The workflow is titled "Parallelism speeds up tests by splitting them across multiple executors." and is located in the repository "nvnhan / udagram-pro3". The workflow consists of several steps: "Spin up environment", "Preparing environment variables", "Checkout code", and "Build docker container for each microservices". The "Build docker container for each microservices" step is currently running and shows a list of Docker images being prepared, including "udagram-frontend" and "udagram-reverseproxy". The output of the step shows the digest and size of each image, and the push refers to the repository "docker.io/*****/udagram-frontend".

The screenshot shows the Docker Hub repository page for the user "nvnhan". The page displays a list of repositories, including "udagram-reverseproxy", "udagram-frontend", "udagram-api-user", and "udagram-api-feed". Each repository is shown with its name, a description, and the last pushed time. The page also includes a search bar and a "Create repository" button.

Config awscli

```
untu@ip-172-31-40-56:~/project3$ aws configure
S Access Key ID [None]: 
S Secret Access Key [None]: 
fault region name [None]: us-east-1
fault output format [None]: json
untu@ip-172-31-40-56:~/project3$ cat ~/.aws/credentials
efault]
s_access_key_id = 
s_secret_access_key = 
untu@ip-172-31-40-56:~/project3$
```



```
VPC
GNU nano 4.8 /home/ubuntu/.aws/credentials
[default]
aws_access_key_id = [REDACTED]
aws_secret_access_key = [REDACTED]
aws_session_token = [REDACTED]

ubuntu@ip-172-31-40-56:~$ aws s3 ls
2023-08-23 02:08:27 bucket-pro3
ubuntu@ip-172-31-40-56:~$
```

Container Orchestration with Kubernetes

Create an EKS Cluster using the EKSTCL Tool

```
sudo curl --silent --location "https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$(uname -s)_amd64.tar.gz" | sudo tar xz -C /usr/local/bin
```

Update config

```
aws eks update-kubeconfig --region us-east-1 --name udacity
```

Create eks

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

Deployment

Recall that while splitting the monolithic app into microservices, you used the values saved in the environment variables, as well as AWS CLI was configured locally. Similar values are required while instantiating containers from the Dockerhub images.

ConfigMap: Create env-configmap.yaml, and save all your configuration values (non-confidential environments variables) in that file.

Secret: Do not store the PostgreSQL username and passwords in the env-configmap.yaml file. Instead, create env-secret.yaml file to store the confidential values, such as login credentials. Unlike the AWS credentials, these values do not need to be Base64 encoded.

Secret: Create aws-secret.yaml file to store your AWS login credentials. Replace `__INSERT_AWS_CREDENTIALS_FILE__BASE64__` with the Base64 encoded credentials (not the regular username/password).

Mac/Linux users: If you've configured your AWS CLI locally, check the contents of `~/.aws/credentials` file using `cat ~/.aws/credentials`. It will display the `aws_access_key_id` and `aws_secret_access_key` for your AWS profile(s). Now, you need to select the applicable pair of `aws_access_key` from the output of the `cat` command above and convert that string into base64

```
ubuntu@ip-172-31-40-56:~/project3/deployment$ ll
total 36
drwxrwxr-x 2 ubuntu ubuntu 4096 Aug 23 16:20 ./
drwxrwxr-x 9 ubuntu ubuntu 4096 Aug 23 04:07 ../
-rw-rw-r-- 1 ubuntu ubuntu 803 Aug 23 15:27 aws-secret.yaml
-rw-rw-r-- 1 ubuntu ubuntu 2337 Aug 23 06:02 backend-feed-deployment.yaml
-rw-rw-r-- 1 ubuntu ubuntu 2409 Aug 23 06:07 backend-user-deployment.yaml
-rw-rw-r-- 1 ubuntu ubuntu 630 Aug 23 14:48 frontend.yaml
-rw-rw-r-- 1 ubuntu ubuntu 684 Aug 23 08:41 reverseproxy.yaml
-rw-rw-r-- 1 ubuntu ubuntu 343 Aug 23 04:36 set-env-configmap.yaml
-rw-rw-r-- 1 ubuntu ubuntu 146 Aug 23 05:49 set-env-secret.yaml
-rw-rw-r-- 1 ubuntu ubuntu 0 Aug 23 15:25 test.txt
ubuntu@ip-172-31-40-56:~/project3/deployment$
```

File: aws-secret.yaml

```
Coding - Untitled-3

1  apiVersion: v1
2  kind: Secret
3  metadata:
4    name: nvnhan-secret
5  type: Opaque
6  data:
7    credentials: <credential_base64>
```

File: set-env-configmap.yaml

```
Coding - Untitled-3

1  apiVersion: v1
2  kind: ConfigMap
3  metadata:
4    name: env-config
5  data:
6    POSTGRES_USERNAME: <postgres_username>
7    POSTGRES_PASSWORD: <postgres_password>
8    POSTGRES_HOST: <db_host>
9    POSTGRES_DB: <db_name>
10   AWS_BUCKET: <bucket_name>
11   AWS_REGION: <region>
12   AWS_PROFILE: default
13   JWT_SECRET: testing
14   URL: http://localhost:8100
```

File: set-env-secret.yaml

```
Coding - Untitled-3

1  apiVersion: v1
2  kind: Secret
3  metadata:
4    name: env-secret
5  type: Opaque
6  data:
7    POSTGRESS_USERNAME: <postgres_username_base64>
8    POSTGRESS_PASSWORD: <postgre_password_base64>
```

File: backend-user-deployment.yaml

```

Coding - Untitled-3

1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: api-user
5  spec:
6    selector:
7      matchLabels:
8        run: api-user
9    replicas: 2
10   template:
11     metadata:
12       labels:
13         run: api-user
14     spec:
15       containers:
16         - name: api-user
17           image: nvnhan/udagram-api-user:v1
18           imagePullPolicy: Always # Set the imagePullPolicy to Always
19           env:
20             - name: POSTGRES_USERNAME
21               valueFrom:
22                 configMapKeyRef:
23                   name: env-config
24                   key: POSTGRES_USERNAME
25             - name: POSTGRES_PASSWORD
26               valueFrom:
27                 configMapKeyRef:
28                   name: env-config
29                   key: POSTGRES_PASSWORD
30             - name: POSTGRES_HOST
31               valueFrom:
32                 configMapKeyRef:
33                   name: env-config
34                   key: POSTGRES_HOST
35             - name: POSTGRES_DB
36               valueFrom:
37                 configMapKeyRef:
38                   name: env-config
39                   key: POSTGRES_DB
40             - name: AWS_BUCKET
41               valueFrom:
42                 configMapKeyRef:
43                   name: env-config
44                   key: AWS_BUCKET
45             - name: AWS_REGION
46               valueFrom:
47                 configMapKeyRef:
48                   name: env-config
49                   key: AWS_REGION
50             - name: AWS_PROFILE
51               valueFrom:
52                 configMapKeyRef:
53                   name: env-config
54                   key: AWS_PROFILE
55             - name: JWT_SECRET
56               valueFrom:
57                 configMapKeyRef:
58                   name: env-config
59                   key: JWT_SECRET
60             - name: URL
61               valueFrom:
62                 configMapKeyRef:
63                   name: env-config
64                   key: URL
65       volumeMounts:
66         - name: nvnhan-secret
67           mountPath: "/root/.aws/"
68           readOnly: true
69       ports:
70         - containerPort: 80
71       resources:
72         limits:
73           cpu: 500m
74         requests:
75           cpu: 200m
76       volumes:
77         - name: nvnhan-secret
78           secret:
79             secretName: nvnhan-secret
80
81 ---
82 apiVersion: v1
83 kind: Service
84 metadata:
85   labels:
86     service: api-user
87   name: api-user
88 spec:
89   ports:
90     - name: "8080"
91       port: 8080
92       targetPort: 8080
93   selector:
94     run: api-user

```

File: backend-feed-deployment.yaml

```
Coding - Untitled-3
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: api-feed
5  spec:
6    selector:
7      matchLabels:
8        run: api-feed
9    replicas: 2
10   template:
11     metadata:
12       labels:
13         run: api-feed
14     spec:
15       containers:
16         - name: api-feed
17           image: nvnhan/udagram-api-feed:v1
18           env:
19             - name: POSTGRES_USERNAME
20               valueFrom:
21                 configMapKeyRef:
22                   name: env-config
23                   key: POSTGRES_USERNAME
24             - name: POSTGRES_PASSWORD
25               valueFrom:
26                 configMapKeyRef:
27                   name: env-config
28                   key: POSTGRES_PASSWORD
29             - name: POSTGRES_HOST
30               valueFrom:
31                 configMapKeyRef:
32                   name: env-config
33                   key: POSTGRES_HOST
34             - name: POSTGRES_DB
35               valueFrom:
36                 configMapKeyRef:
37                   name: env-config
38                   key: POSTGRES_DB
39             - name: AWS_BUCKET
40               valueFrom:
41                 configMapKeyRef:
42                   name: env-config
43                   key: AWS_BUCKET
44             - name: AWS_REGION
45               valueFrom:
46                 configMapKeyRef:
47                   name: env-config
48                   key: AWS_REGION
49             - name: AWS_PROFILE
50               valueFrom:
51                 configMapKeyRef:
52                   name: env-config
53                   key: AWS_PROFILE
54             - name: JWT_SECRET
55               valueFrom:
56                 configMapKeyRef:
57                   name: env-config
58                   key: JWT_SECRET
59             - name: URL
60               valueFrom:
61                 configMapKeyRef:
62                   name: env-config
63                   key: URL
64       volumeMounts:
65         - name: nvnhan-secret
66           mountPath: "/root/.aws/"
67           readOnly: true
68       ports:
69         - containerPort: 80
70       resources:
71         limits:
72           cpu: 500m
73         requests:
74           cpu: 200m
75       volumes:
76         - name: nvnhan-secret
77           secret:
78             secretName: nvnhan-secret
79
80   ---
81   apiVersion: v1
82   kind: Service
83   metadata:
84     labels:
85       service: api-feed
86     name: api-feed
87   spec:
88     ports:
89       - name: "8080"
90         port: 8080
91         targetPort: 8080
92     selector:
93       run: api-feed
```

File: reverseproxy.yaml

```
Coding - Untitled-3

1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: reverseproxy
5  spec:
6    selector:
7      matchLabels:
8        run: reverseproxy
9    replicas: 2
10   template:
11     metadata:
12       labels:
13         run: reverseproxy
14     spec:
15       containers:
16       - name: reverseproxy
17         image: nvnhan/reverseproxy:v1
18         imagePullPolicy: Always
19         ports:
20         - containerPort: 80
21       resources:
22         limits:
23           cpu: 1000m
24         requests:
25           cpu: 500m
26
27   ---
28   apiVersion: v1
29   kind: Service
30   metadata:
31     labels:
32       service: reverseproxy
33     name: reverseproxy
34   spec:
35     ports:
36     - name: "8080"
37       port: 8080
38       targetPort: 8080
39     selector:
40       service: reverseproxy
```

File: frontend.yaml

```
Coding - Untitled-3

1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: frontend
5  spec:
6    selector:
7      matchLabels:
8        run: frontend
9    replicas: 2
10   template:
11     metadata:
12       labels:
13         run: frontend
14     spec:
15       containers:
16       - name: frontend
17         image: nvnhan/udagram-frontend:v1
18         imagePullPolicy: Always
19         ports:
20         - containerPort: 80
21       resources:
22         limits:
23           cpu: 500m
24         requests:
25           cpu: 200m
26
27   ---
28   apiVersion: v1
29   kind: Service
30   metadata:
31     name: frontend
32     labels:
33       run: frontend
34   spec:
35     ports:
36     - port: 80
37       protocol: TCP
38     selector:
39       run: frontend
```

```
kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml
kubectl apply -f set-env-secret.yaml
kubectl apply -f set-env-configmap.yaml
kubectl apply -f aws-secret.yaml
kubectl apply -f backend-user-deployment.yaml
kubectl apply -f backend-feed-deployment.yaml
kubectl apply -f reverseproxy.yaml
```

```
kubectl expose deployment reverseproxy --type=LoadBalancer --name=reverseproxy-ep --port=8080
```

```
ubuntu@ip-172-31-40-56:~/project3/deployment$ kubectl get services
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
api-feed	ClusterIP	10.100.43.69	<none>	8080/TCP	5h49m
api-user	ClusterIP	10.100.145.204	<none>	8080/TCP	5h49m
kubernetes	ClusterIP	10.100.0.1	<none>	443/TCP	6h
reverseproxy	ClusterIP	10.100.36.171	<none>	8080/TCP	5h2m
reverseproxy-ep	LoadBalancer	10.100.117.200	a20e2206d47fe40258b463e966fb4004-407380140.us-east-1.elb.amazonaws.com	8080:32370/TCP	13s

```
ubuntu@ip-172-31-40-56:~/project3/deployment$
```

Update `udagram-frontend/src/environments/environment.ts` file - Replace the keyword `localhost` in the `http://localhost:8080/api/v0` string with the External-IP of the reverseproxy deployment

```
ubuntu@ip-172-31-40-56:~/project3$ grep -r "8080/api"
```

```
udagram-frontend/src/environments/environment.prod.ts:  apiHost: 'http://a20e2206d47fe40258b463e966fb4004-407380140.us-east-1.elb.amazonaws.com:8080/api/v0'
```

```
udagram-frontend/src/environments/environment.ts:  apiHost: 'http://a20e2206d47fe40258b463e966fb4004-407380140.us-east-1.elb.amazonaws.com:8080/api/v0'
```

```
ubuntu@ip-172-31-40-56:~/project3$
```

Build a new frontend image, and push it to the Dockerhub. While building a new image, it is recommended to use a different tag each time, as shown in the example below

```
docker build -t udagram-frontend ./udagram-frontend
docker tag udagram-frontend nvnhan/udagram-frontend:v1
docker push nvnhan/udagram-frontend:v1
```

```
nvnhan@nvnhan:~/aws-devloper-pro3$ docker build -t udagram-frontend ./udagram-frontend
```

```
[+] Building 61.8s (7/15)
```

	docker:default
⇒ [internal] load .dockerignore	0.1s
⇒ ⇒ transferring context: 52B	0.0s
⇒ [internal] load build definition from Dockerfile	0.0s
⇒ ⇒ transferring dockerfile: 430B	0.0s
⇒ [internal] load metadata for docker.io/library/nginx:alpine	3.8s
⇒ [internal] load metadata for docker.io/beevelop/ionic:latest	3.4s
⇒ [auth] beevelop/ionic:pull token for registry-1.docker.io	0.0s
⇒ [auth] library/nginx:pull token for registry-1.docker.io	0.0s
⇒ [internal] load build context	0.1s
⇒ ⇒ transferring context: 1.38MB	0.0s
⇒ [stage 1/12] FROM docker.io/library/nginx:alpine@sha256:c9c882be2b73205e0c8d3e3cd0575e2fd58f5fd66dd5d6300665e3a0f2e67c385	58.0s

```
nvnhan@nvnhan:~/aws-devloper-pro3$ docker tag udagram-frontend nvnhan/udagram-frontend:v1
```

```
nvnhan@nvnhan:~/aws-devloper-pro3$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
nvnhan/udagram-frontend	v1	b7c44fd5a7ad	31 seconds ago	60MB
udagram-frontend	latest	b7c44fd5a7ad	31 seconds ago	60MB

```
nvnhan@nvnhan:~/aws-devloper-pro3$ docker push nvnhan/udagram-frontend:v1
```

```
The push refers to repository [docker.io/nvnhan/udagram-frontend]
```

```
fb0c08cd8255: Pushing [=====] 18.16MB
```

```
fe438658cde3: Layer already exists
```

```
5a06aafa23bc: Layer already exists
```

```
5a5177dd8444: Layer already exists
```

```
2f0df4d427b0: Layer already exists
```

```
c92e59d38453: Layer already exists
```

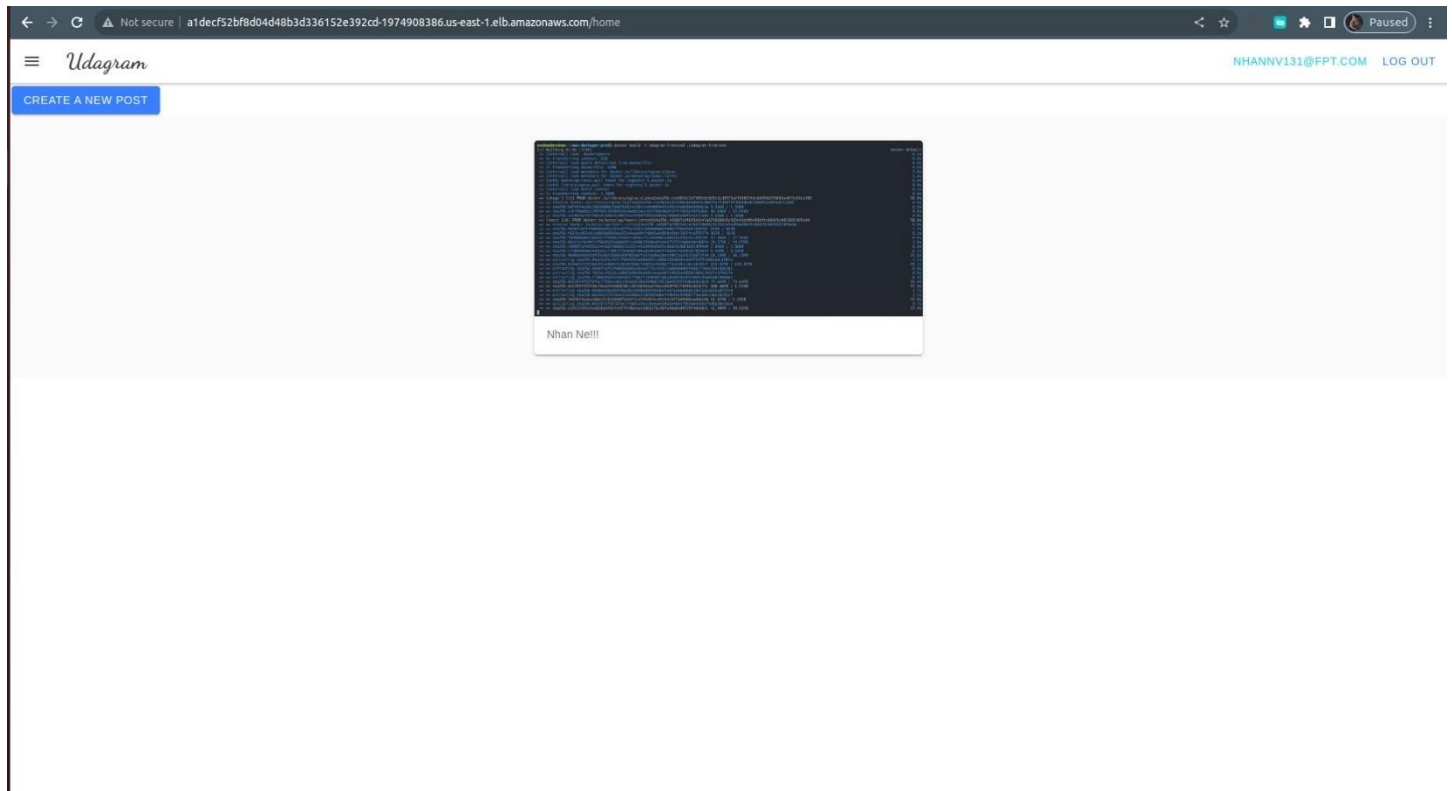
```
53b44a18496e: Layer already exists
```

```
66ff66f91ede: Layer already exists
```

```
4693057ce236: Layer already exists
```

```
kubectl apply -f frontend.yaml
```


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



kubectl autoscale deployment api-user --cpu-percent=70 --min=3 --max=5
kubectl describe hpa

```
abuntu@ip-172-31-40-56:~/project3/deployment$ kubectl describe hpa
Name:                                api-user
Namespace:                           default
Labels:                               <none>
Annotations:                          <none>
CreationTimestamp:                   Wed, 23 Aug 2023 14:53:21 +0000
Reference:                           Deployment/api-user
Metric:                               ( current / target )
  resource cpu on pods  (as a percentage of request): 0% (0) / 70%
Min replicas:                        3
Max replicas:                        5
Deployment pods:                      3 current / 3 desired
Conditions:
  Type            Status  Reason                        Message
  ----            -
  AbleToScale     True    ReadyForNewScale             recommended size matches current size
  ScalingActive   True    ValidMetricFound             the HPA was able to successfully calculate a replica count from cpu resource utilization (percentage of request)
  ScalingLimited  True    TooFewReplicas               the desired replica count is less than the minimum replica count

Events:
  Type    Reason      Age    From                      Message
  ----    -
  Normal  SuccessfulRescale  56m    horizontal-pod-autoscaler  New size: 4; reason: cpu resource utilization (percentage of request) above target
  Normal  SuccessfulRescale  50m    horizontal-pod-autoscaler  New size: 4; reason: All metrics below target
  Warning  FailedGetScale  8m37s (x4 over 17m) horizontal-pod-autoscaler  deployments/scale.apps "api-user" not found
  Normal  SuccessfulRescale  8m22s (x3 over 57m) horizontal-pod-autoscaler  New size: 3; reason: Current number of replicas below Spec.MinReplicas
  Warning  FailedGetResourceMetric  8m7s (x2 over 16m) horizontal-pod-autoscaler  failed to get cpu utilization: did not receive metrics for any ready pods
  Warning  FailedComputeMetricsReplicas  8m7s (x2 over 16m) horizontal-pod-autoscaler  invalid metrics (1 invalid out of 1), first error is: failed to get cpu resource metric values: failed to get cpu utilization: did not receive metrics for any ready pods
  Normal  SuccessfulRescale  7m52s (x3 over 56m) horizontal-pod-autoscaler  New size: 5; reason: cpu resource utilization (percentage of request) above target
  Normal  SuccessfulRescale  111s (x3 over 50m) horizontal-pod-autoscaler  New size: 3; reason: All metrics below target
abuntu@ip-172-31-40-56:~/project3/deployment$
```

Screenshot submission: [here](#)