1. introduction

In this assignment, we will deploy a cluster on the Google Kubernetes Engine(GKE) to do the training job and inference job for a CNN model.

2. prerequisites

2.1 Prepare the images

In order to deploy the cluster and enable the inference service, we need to build the images and push the images into docker hub, so that we can easily use the images to create containers in K8S. Here attached the structure of total repo.

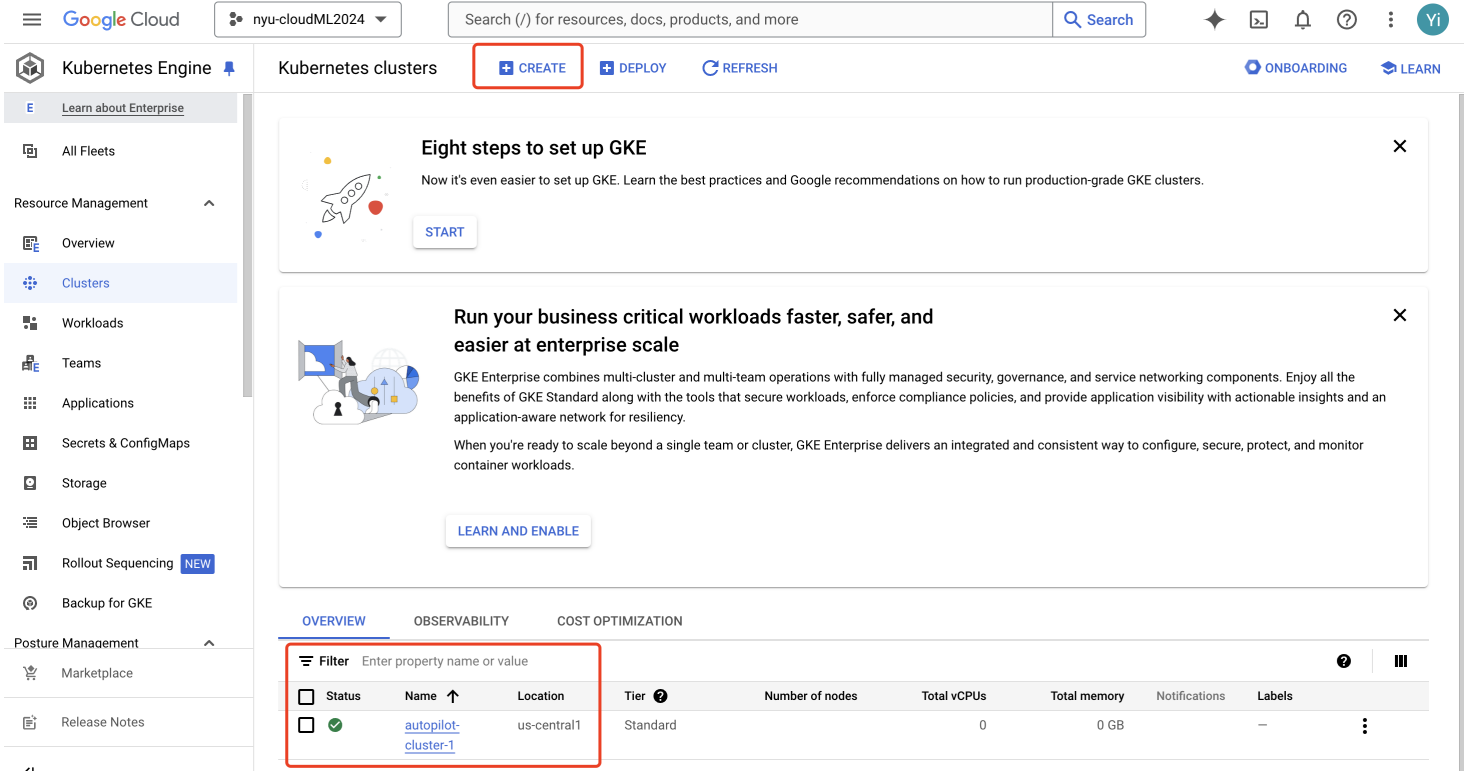
```
.
├── inference
│   ├── Dockerfile
│   ├── inference.py
│   └── requirements.txt
├── inference.yaml
├── pvc.yaml
├── service.yaml
├── train
│   ├── Dockerfile
│   └── train.py
└── train.yaml
```

After we finished the code writing, we can use docker to build the image and push it to docker hub.

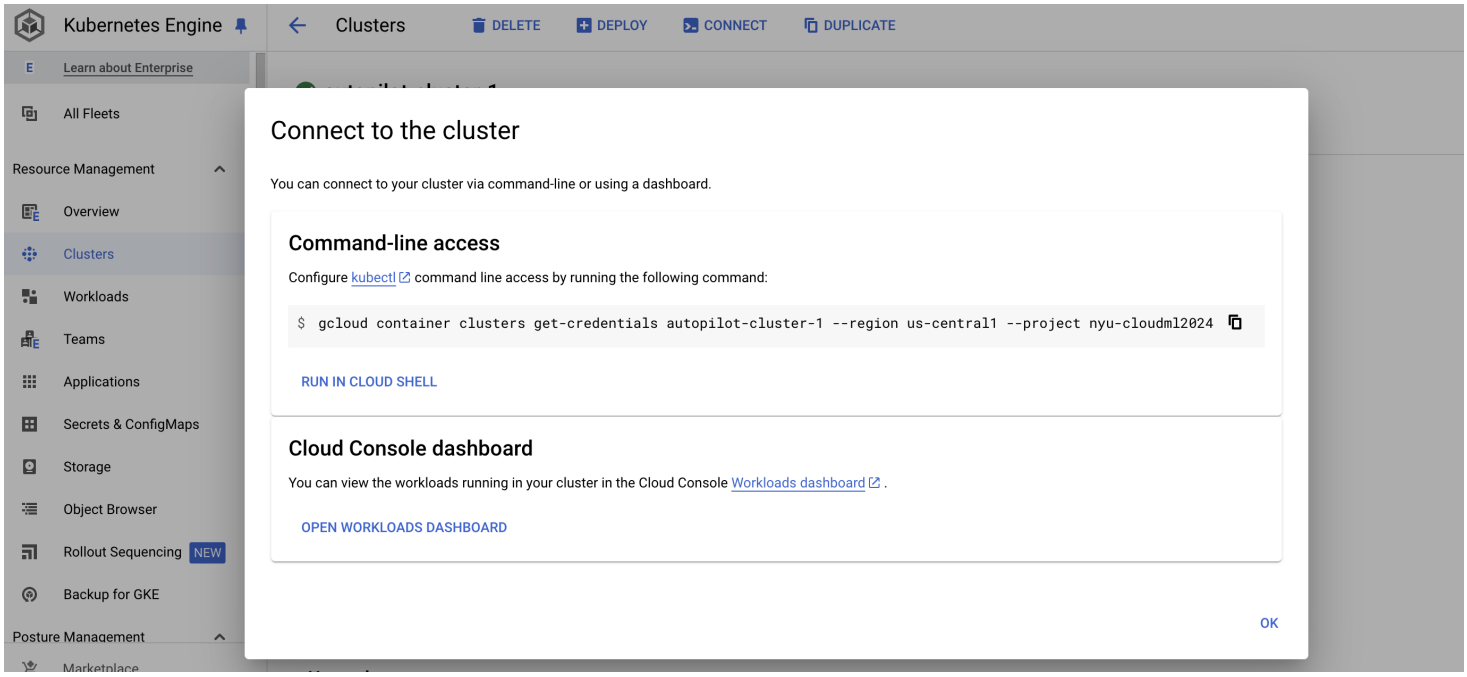
```
docker login -u <username>

docker build -t inferencejob:1.0 inference/
docker build -t trainjob:1.0 train/

docker tag inferencejob:1.0 yp2141/inferencejob:1.0
docker tag trainjob:1.0 yp2141/trainjob:1.0

docker push yp2141/inferencejob:1.0
docker push yp2141/trainjob:1.0
```

2.2 Create a cluster on GKE

First, we need to create a cluster on google cloud, and after we successfully created the cluster, we can see a cluster in **OVERVIEW** part.



Then clicking the cluster and entering into **Clusters** page, we can connect to it under cloud shell.

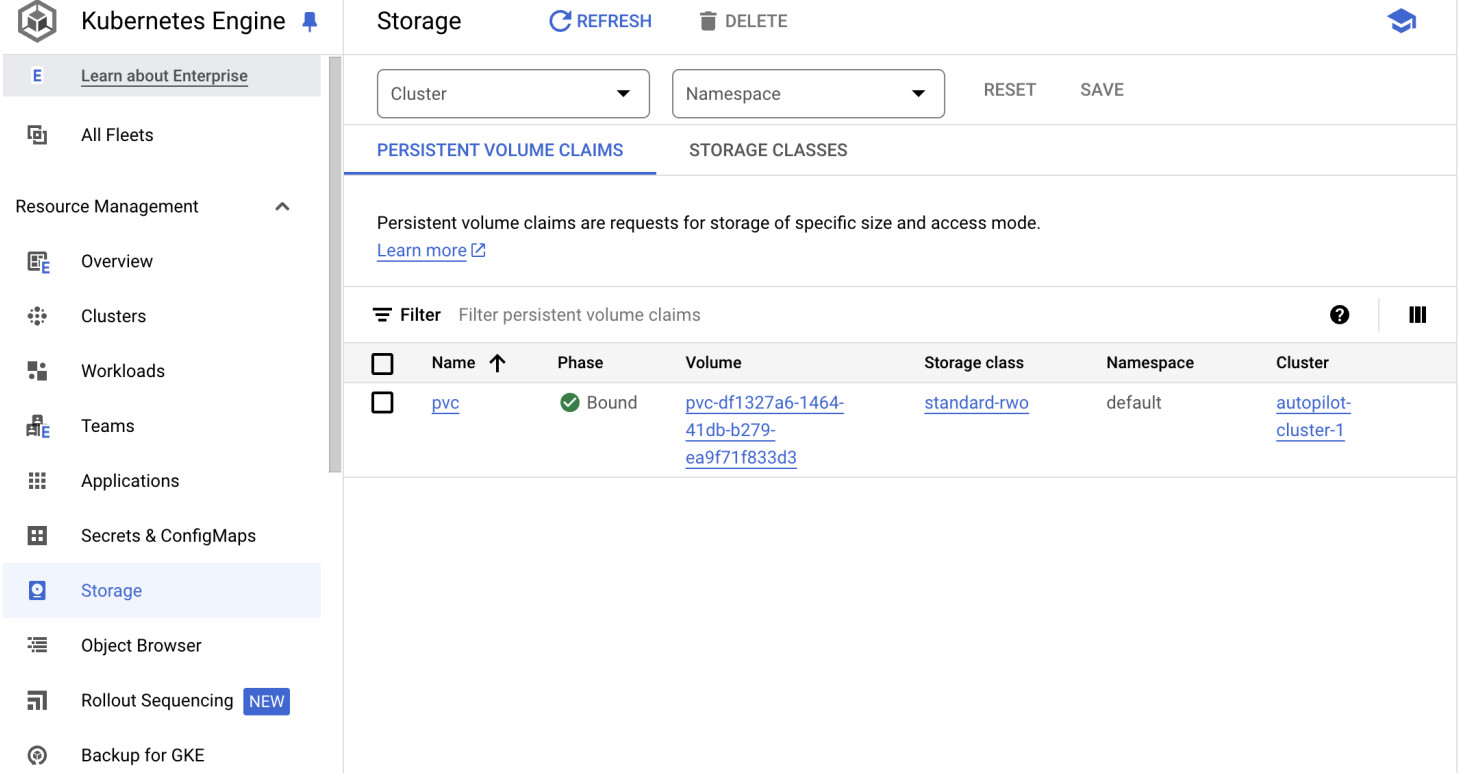


Then, use `git clone` to clone the repository and deploy the pods

```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to nyu-cloudml2024.
Use “gcloud config set project [PROJECT_ID]” to change to a different project.
gcloud container clusters get-credentials autopilot-cluster-1 --region us-central1 --project nyu-cloudml2024yp2141@cloudshell:~ (nyu-cloudml2024)$ gcloud container clusters get-creden
yp2141@cloudshell:~ (nyu-cloudml2024)$ gcloud container clusters get-credentials autopilot-cluster-1 --region us-central1 --project nyu-cloudml2024
Fetching cluster endpoint and auth data.
kubeconfig entry generated for autopilot-cluster-1.
yp2141@cloudshell:~ (nyu-cloudml2024)$ git clone https://github.com/yipeng0016/mnistK8S.git
Cloning into 'mnistK8S'...
remote: Enumerating objects: 13, done.
remote: Counting objects: 100% (13/13), done.
remote: Compressing objects: 100% (13/13), done.
Receiving objects: 100% (13/13), 4.52 KiB | 1.51 MiB/s, done.
remote: Total 13 (delta 0), reused 13 (delta 0), pack-reused 0 (from 0)
yp2141@cloudshell:~ (nyu-cloudml2024)$ ls
main.tf  mnistK8S  README-cloudshell.txt  terraform.tfstate  terraform.tfstate.backup
```

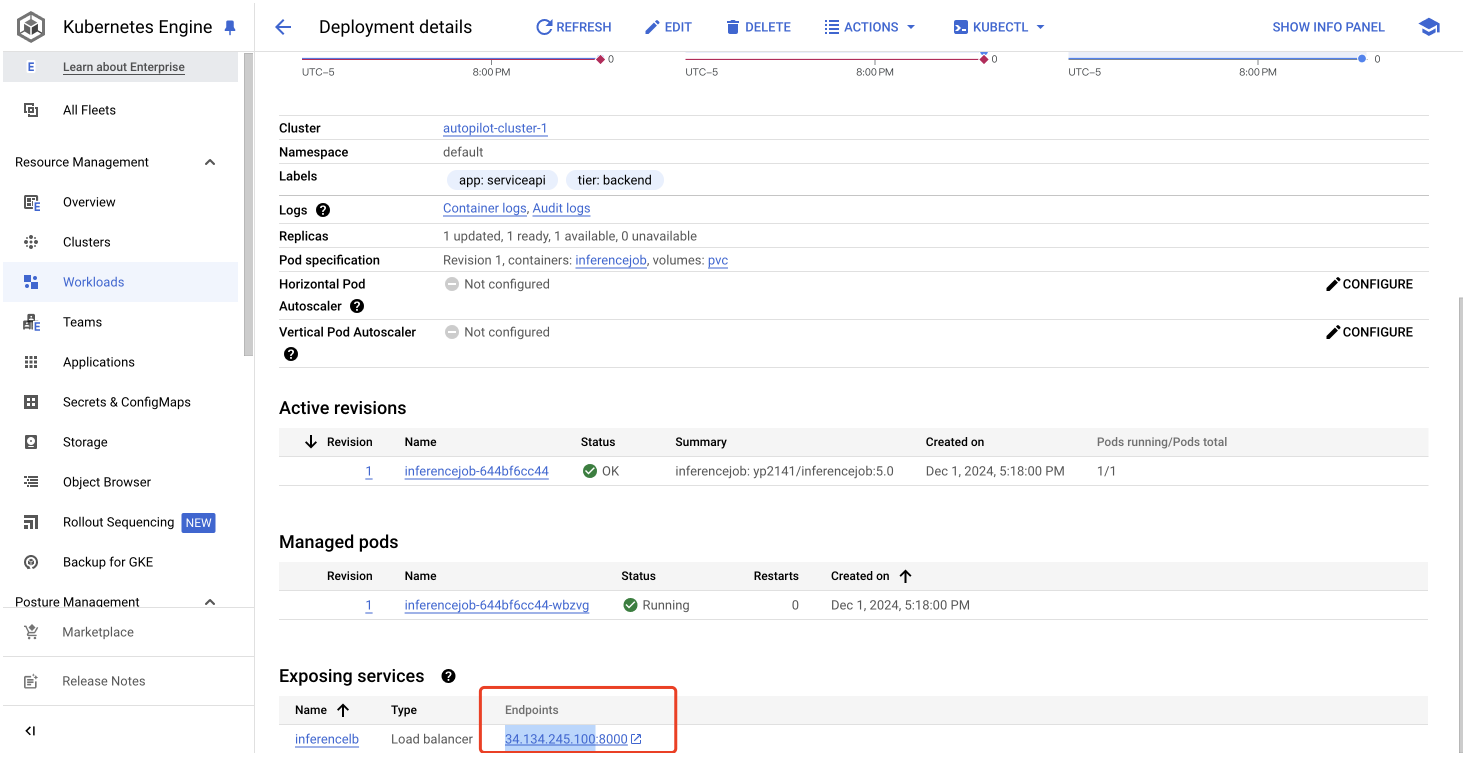
2.3 Deploy service on the cluster

First, we need to use the command `kubectl apply -f pvc.yaml` to deploy the persistence volume claim. It should be noted that pvc will remain in the pending state until an application uses it, at which time the state of pvc will change from pending to bounded.



Then, we need to apply the training, inference and service.yaml.

```
kubectl apply -f training.yaml
kubectl apply -f inference.yaml
kubectl apply -f service.yaml
```



3. Access the inference service

After the deployment, we can use the exposed ip to access the service. In this project, I provided a restful api and you can use curl to access to it.

```
curl -X POST -F "file=@test.png" http://34.134.245.100:8000/predict
```

The final github repo:https://github.com/yipeng0016/mnistK8S

Reference

> The most scalable and fully automated Kubernetes service (https://cloud.google.com/kubernetes-engine?hl=en)
> Pytorch MNIST example (https://github.com/pytorch/examples/tree/main/mnist)
> Deploy an app to a GKE cluster (https://cloud.google.com/kubernetes-engine/docs/deploy-app-cluster)