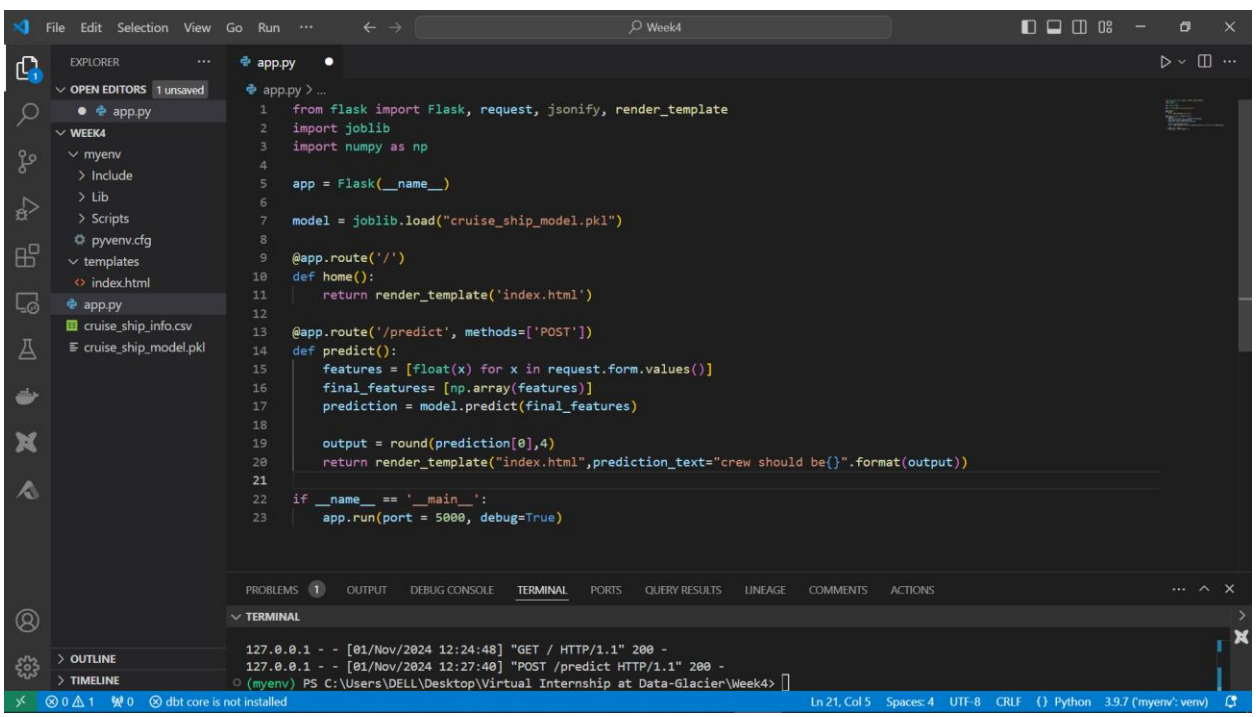
Week 5 Report
Cloud Deployment

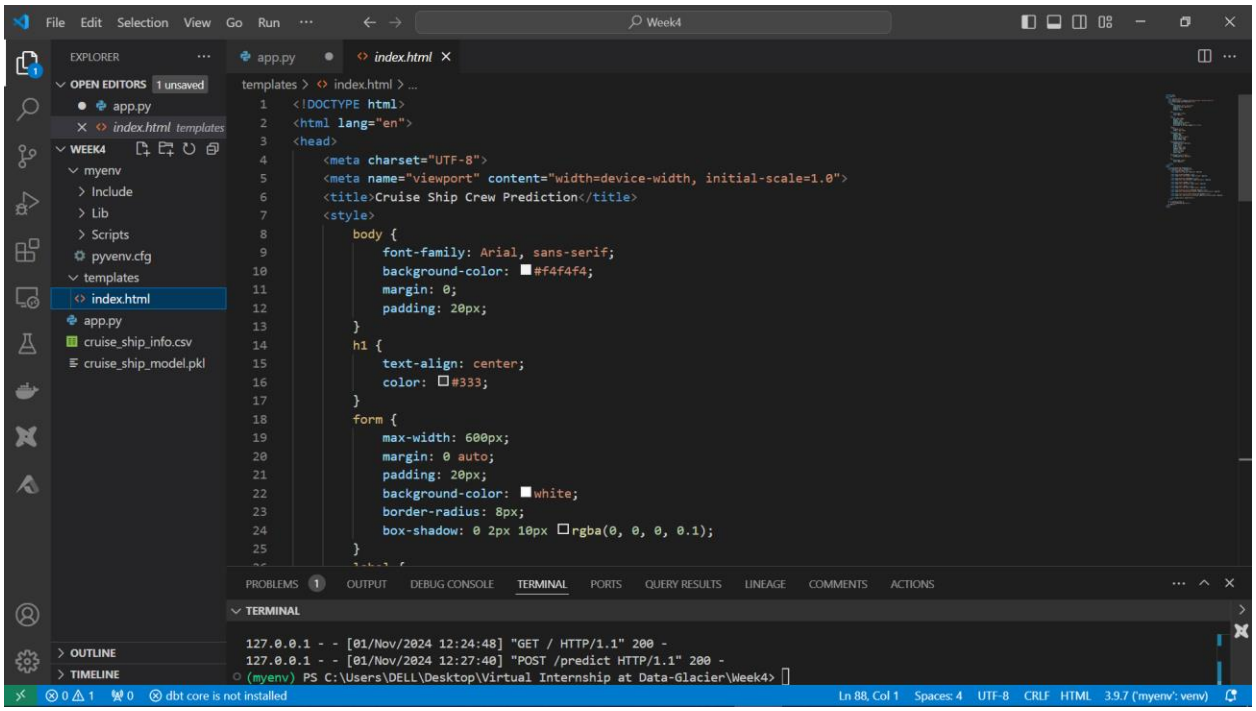
Name: Cloud Deployment
Report date: 11-04-2024
Internship Batch: LISUM38
Version:<1.0>
Data intake by: Ky Dang
Data intake reviewer:
Data storage location:

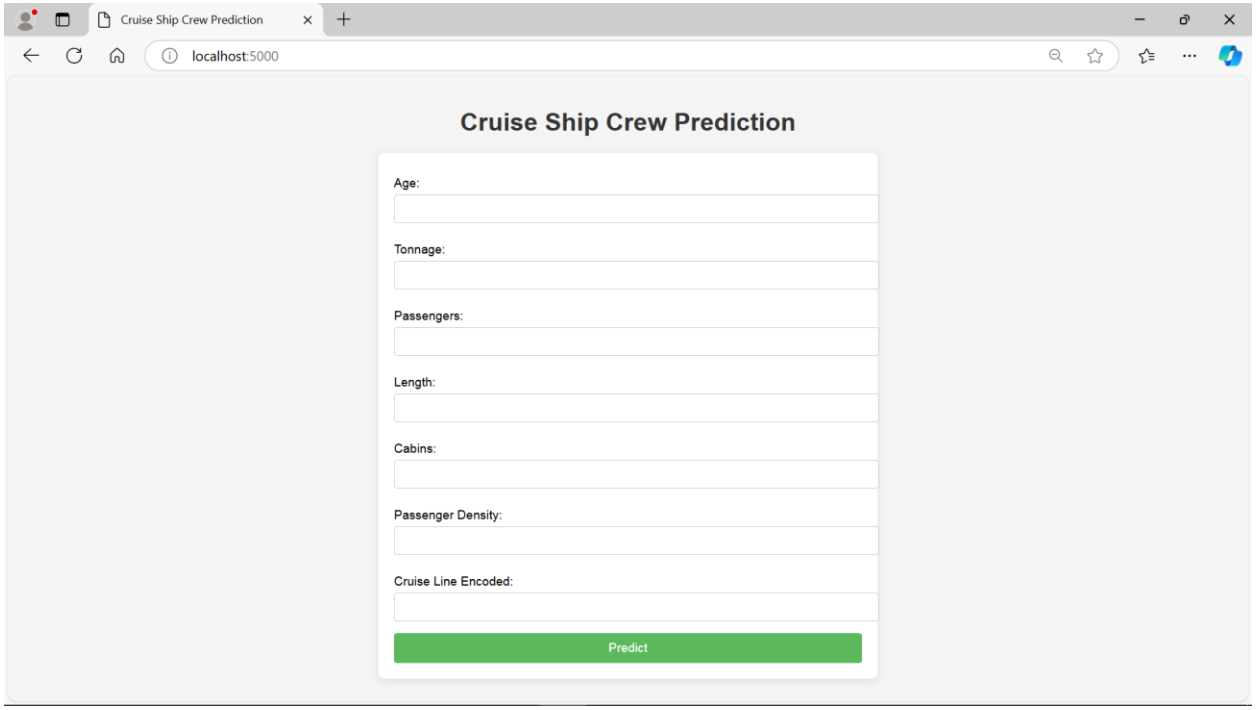
Tabular data details:

Total number of observations	158
Total number of files	1
Total number of features	10
Base format of the file	.csv
Size of the data	9Kb

1. Flask API

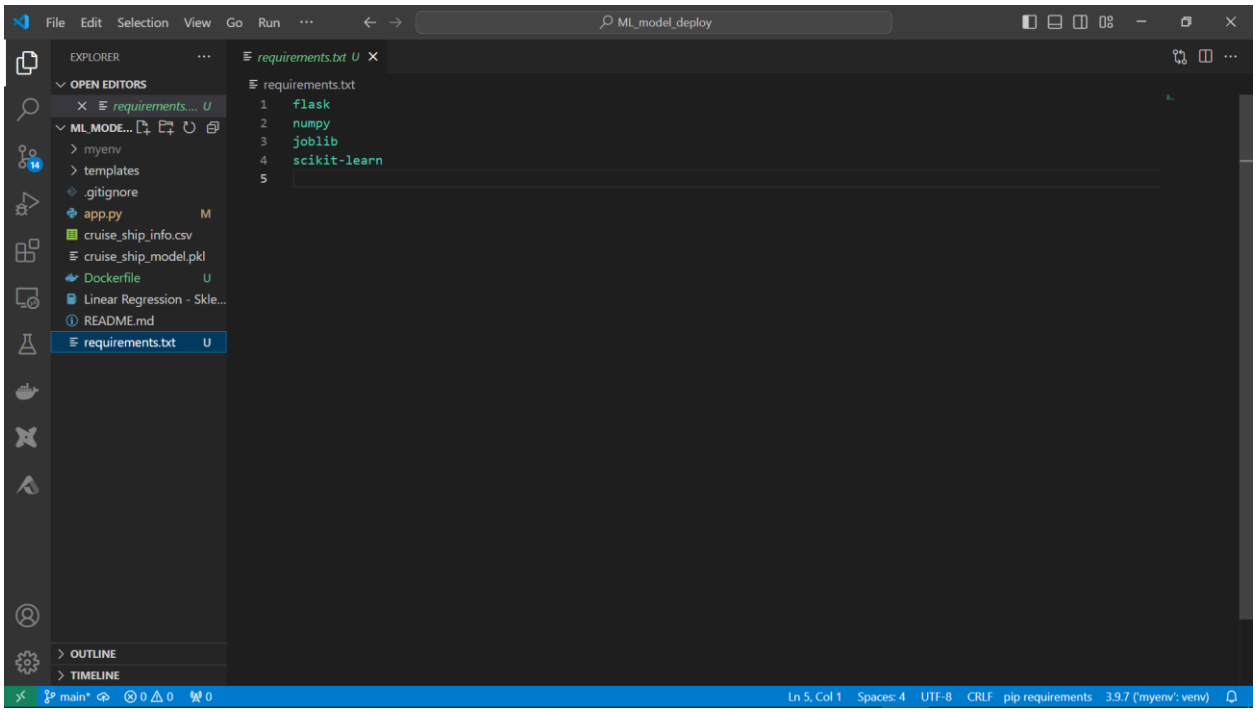






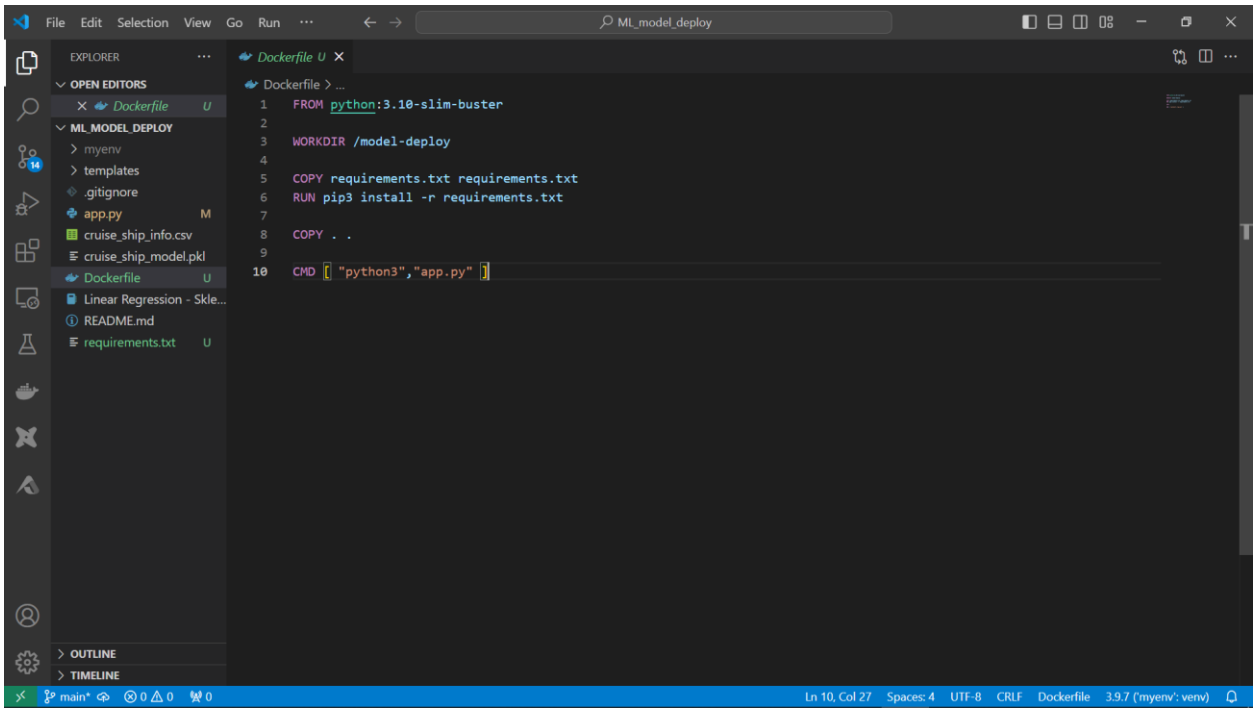
Preparing the index.html in templates folder and app.py file, then deploying Flask run on local host with port 5000.

2. Prepare requirements.txt for cloud deployment



Requirements.txt contans these essential labraries such as: Flask, numpy, joblib and skit-learn

3. Prepare Docker file



To reduce the overall size of the Docker container while still providing a Python environment suitable for the application, I made use of **`python 3:10-slim-buster`** (lightweight Docker image based on the Debian Buster distribution).
The docker file will be used to create an image for contaniner in cloud run.

4. Cloud Deployment (Google Cloud Run)

Utilize the Google Cloud Run with Cloud Build API and Artifacts Registry to deploy the web application (in case of Machine Learning model prediction).

4.1 Cloud Run

Cloud Run is a fully managed compute platform that automatically scales your containerized applications. It allows you to run stateless HTTP-driven containers in a serverless environment, which means you don't have to manage the underlying infrastructure.

4.2 Cloud Build

Cloud Build is a continuous integration and delivery (CI/CD) platform that automates the building, testing, and deploying of applications. It allows developers to create and manage build pipelines for their code.

4.3 Artifact Registry

Artifact Registry is a service for storing, managing, and securing artifacts like container images and language packages. It is designed to work with Google Cloud's container and package management ecosystem.

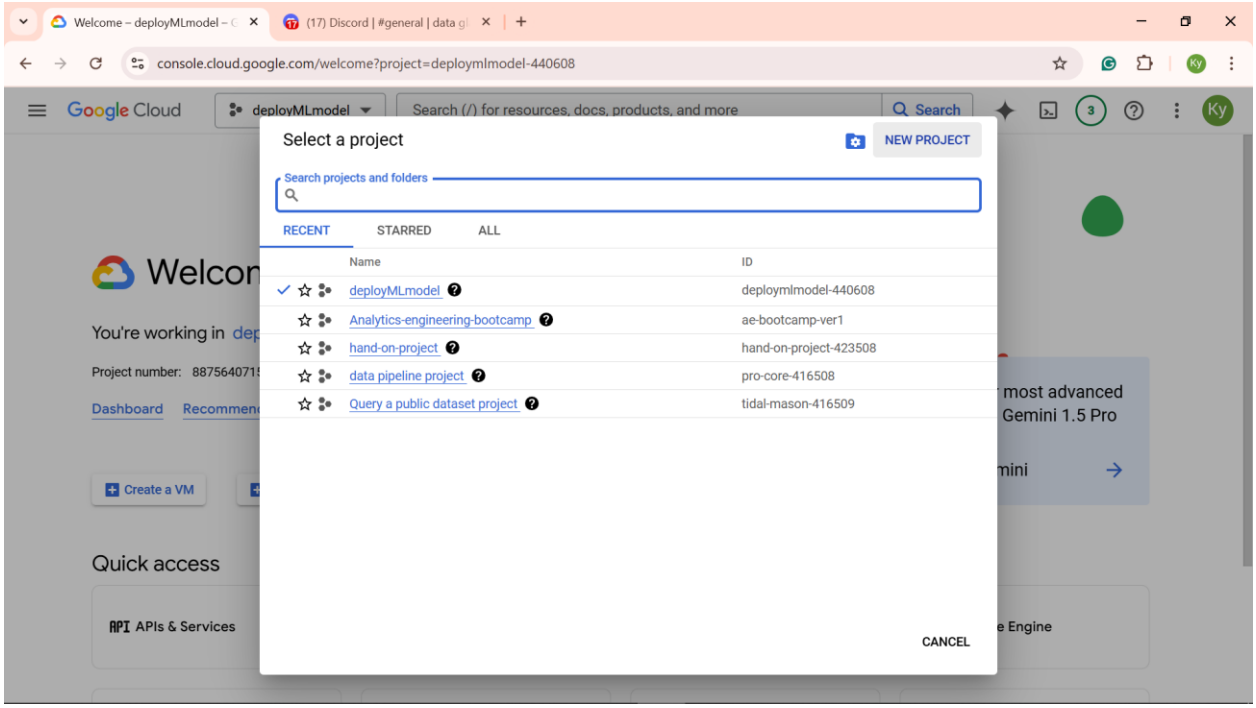
How they work together:

1) **Development and Building**: Write the code and define your CI/CD pipeline in Cloud Build. When pushing the codes to the repository, Cloud Build automatically triggers the build process.
2) **Containerization**: Cloud Build builds the application into a container image and stores it in Artifact Registry.
3) **Deployment**: Deploying the containerized application directly to Cloud Run, which automatically scales and manages the application in a serverless environment.

4.4 Workflow:

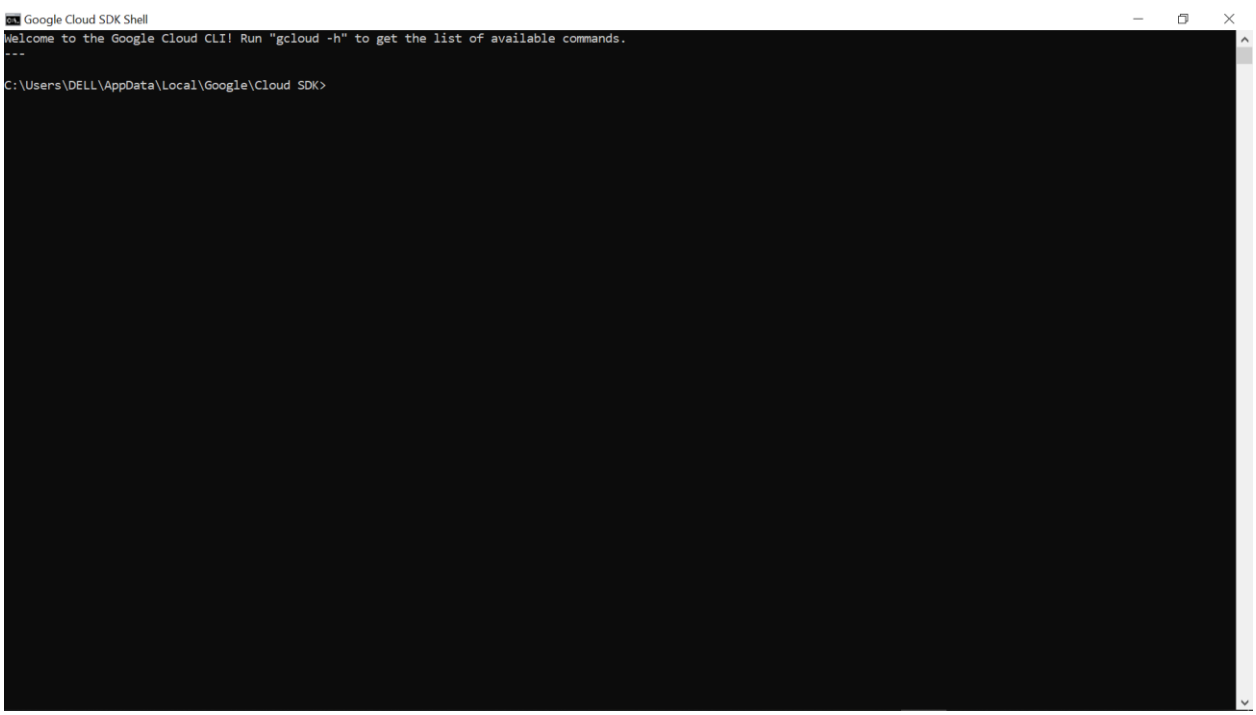
4.4.1 Open the google console

4.4.2 Create a new project and name regardingly to the application



4.4.3 Enable Cloud Build API and Artifact Registry (they aslo require billing account)

4.4.4 Installation Google Cloud SDK (GG Cloud CLI) for proper OS.



I am using google cloud CLI to communicate with Google platform (in this case is Artifacts and Cloud build). The workflow can be done with UI of these features with non-coding.

4.4.5 Initialize the Google Cloud environment

Set up my local environment to interact with Google Cloud services by configuring various settings such as my project, account…

```
gcloud init
```

4.4.6 Authenticate my gcloud SDK session

By allowing me to log in with my google account and enabling me to securely access and manage my Google Cloud resources using the gcloud command-line tool.

```
gcloud auth login
```

4.4.7 Create a new Docker repository in Google Cloud Artifact Registry

```
gcloud artifacts repositories create mldeploy --repository-format=docker --location=us-east1 --description="deploy model" --immutable-tags –async
```

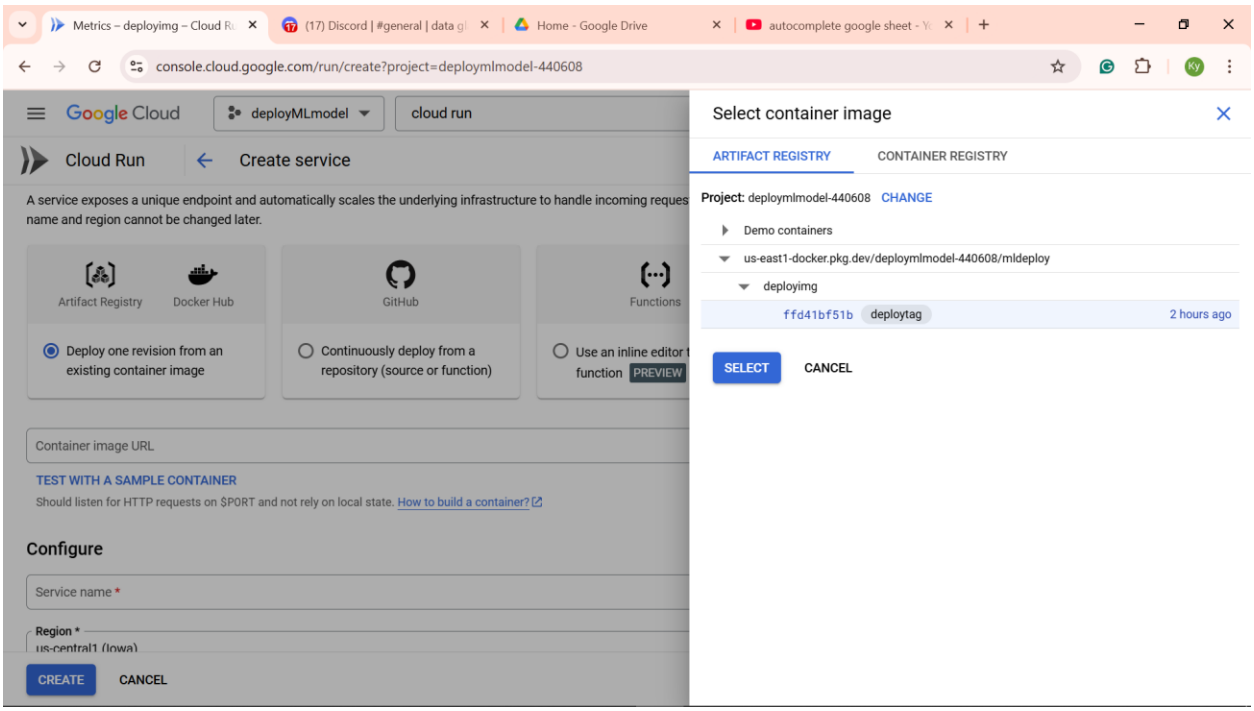
4.4.8 Authenticate Docker with Google Cloud's Artifact Registry

```
gcloud auth configure-docker us-east1-docker.pkg.dev?
```

4.4.9 Submit a build to Google Cloud Build

```
gcloud builds submit --tag us-east1-docker.pkg.dev/deploymlmodel-440608/mldeploy/deployimg:deploytag
```

4.4.10 Create the cloud run service



The final step of deploying on Google cloud with Cloud run.

Now I can refer to this link and using this model (I will stop running on this gcloud due to the incidental costs)

Link: [Cruise Ship Crew Prediction]