GENAI - CONTAINERIZED VIDEO TRANSCRRIPTION AND CHAT

Week 11 Homework

CourseCloud Computing Infrastructure

Name: Hartina Vonyee Cleon

ID: 20145

Due Date： August 3, 2024

Introduction

In this task, I deployed the Streamlit application inside a Docker container on Google Cloud Kubernetes. The aim was to set up the necessary environment, obtain and apply various API keys, create the Docker image, and ensure the application functions optimally within a containerized environment. I also addressed how to handle issues such as Docker Compose file validation, environment variables, and obtaining information about specific containers, such as IP addresses. By following a planned approach to each design step, I aimed to create a functional, cloud-based application. Specifically, this project involves designing a chatbot that can respond to questions from a video. It demonstrates the integration of technologies such as Docker, OpenAI, Whisper, Embeddings, Chat completions, Pinecone, and Retrieval-Augmented Generation.

Prerequisites

- OpenAI API Key
- Pinecone API Key
- The latest version of Docker Desktop
- GitHub repository

Steps to Build and Run the Application

1. Clone Repository: Use Git to clone the project's repository.

Now that we have all the necessary resources for the task, we can proceed with developing our application. In the terminal, navigate to the desired directory:

git clone https://github.com/Davidnet/docker-genai.git

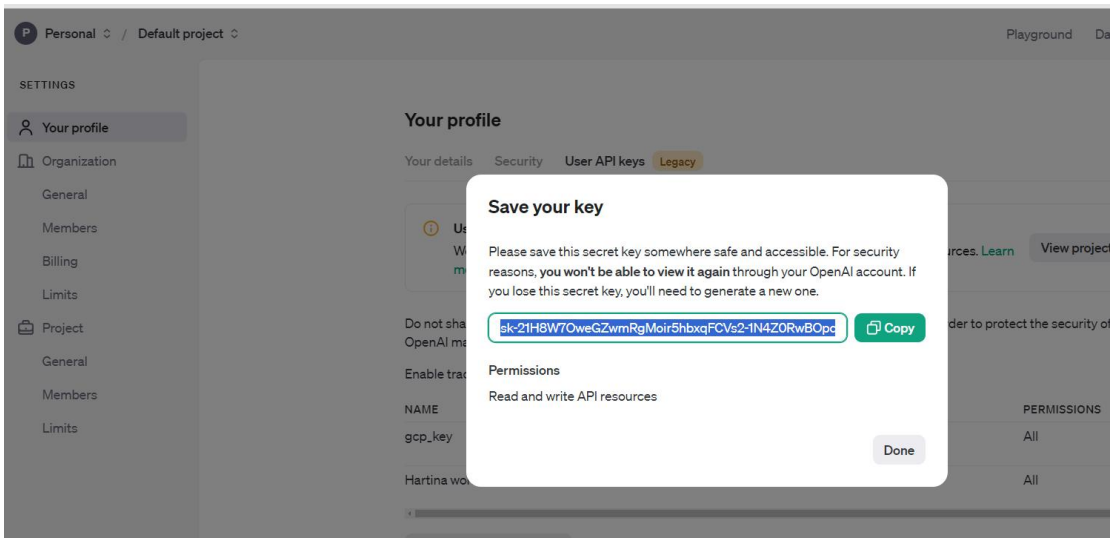
```
C:\Users\HP\Documents\SEM 2\CS571>clone https://github.com/Davidnet/docker-genai.git
'clone' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\HP\Documents\SEM 2\CS571>git clone https://github.com/Davidnet/docker-genai.git
Cloning into 'docker-genai'...
remote: Enumerating objects: 66, done.
remote: Counting objects: 100% (66/66), done.
remote: Compressing objects: 100% (43/43), done.
remote: Total 66 (delta 24), reused 60 (delta 20), pack-reused 0
Receiving objects: 100% (66/66), 114.38 KiB | 329.00 KiB/s, done.
Resolving deltas: 100% (24/24), done.
```

Get an OpenAI API Key

To obtain an OpenAI API key, follow these steps:

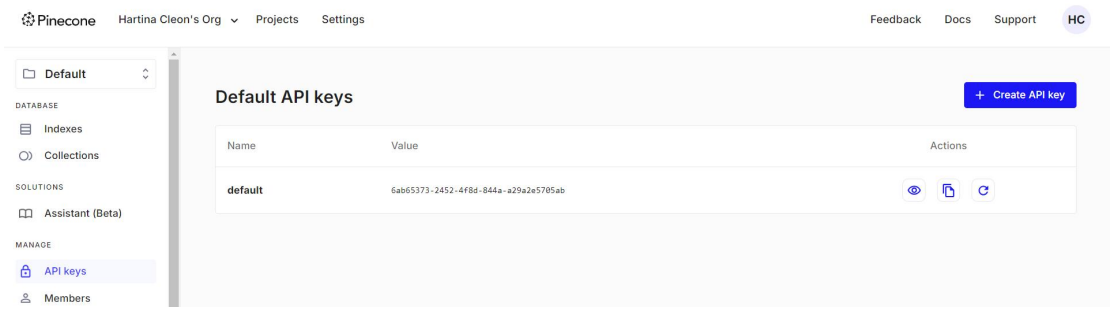
1. Click on the provided link to access the OpenAI website.
2. Enter your OpenAI login details.
3. Navigate to the 'Billing' section.
4. Click on 'Add Payment Details' and proceed to add the required credit.
5. Once you have successfully added enough credit, go to the sidebar menu.
6. From the menu, select "Create a New Secret Key."

7. Optionally, you can provide a name for this key.
8. Fill in the necessary information in the "Create Secret Key" section to generate your API key.



Obtain a Pinecone API Key

To get a Pinecone API Key, follow the link provided. This will take you to the Pinecone web app where you can either enter your account credentials or create a new account. The link will open in a new window. Go to the side menu and navigate to API KEYS to retrieve your default API Key.



Specify your API keys

Go to docker-genai directory:
cd docker-genai
Create a text file ".env":
code .env
Give the below script in the .env file and replace the keys with your Actual keys:

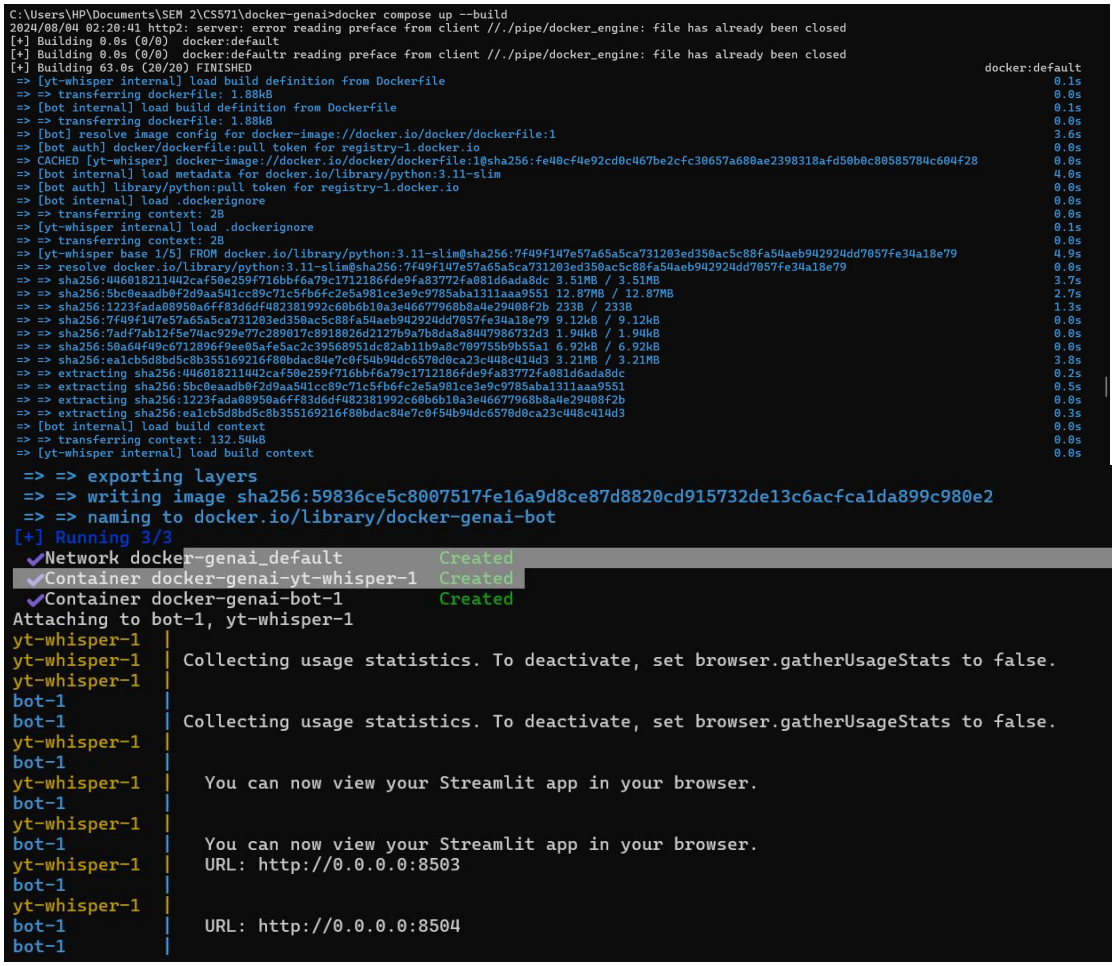
Specify your API keys in ".env" using this template:
#----------
OpenAI

#--
OPENAI_TOKEN=your-api-key # Replace your-api-key with your personal API key
#--
Pinecone
#--
PINECONE_TOKEN=your-api-key # Replace your-api-key with your personal API key

Build and run the application

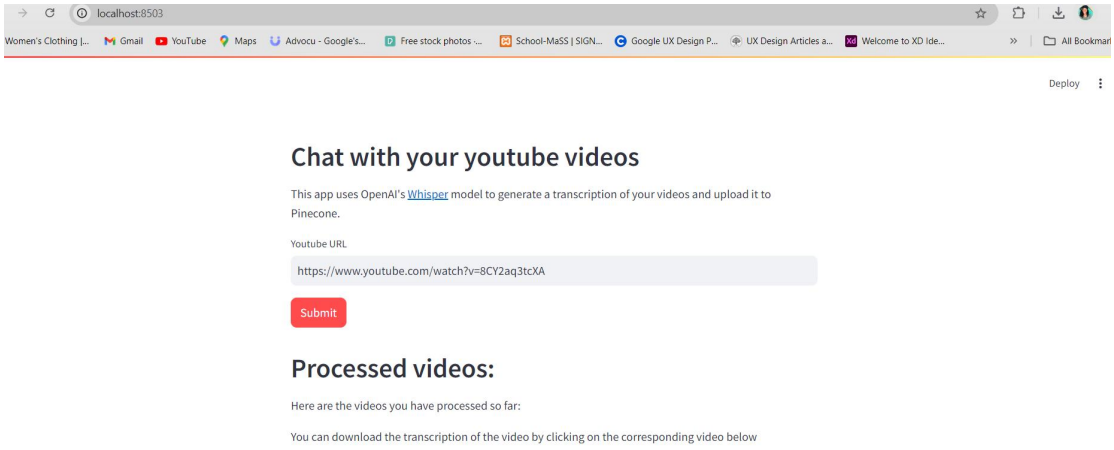
docker compose up - -build

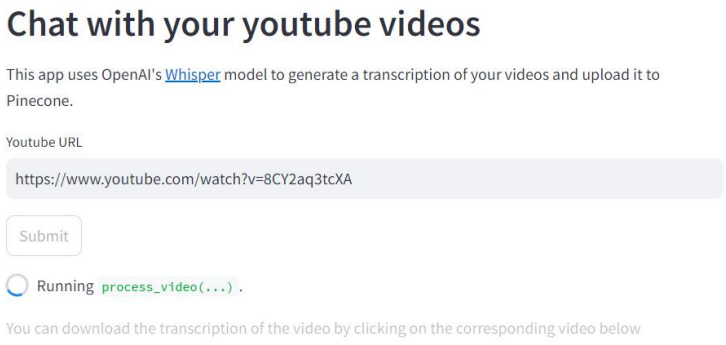
You will notice this while Docker is constructing the application; the following is typical when the application is up. As you can deduce, the Dockerfile is similar to the one used when running the application.



Use the yt-whisper service

Open a browser and access the yt-whisper service at http://localhost:8503.
Enter the Youtube video URL you want to use and select “Submit”:





Error encounter:

RateLimitError: Error code: 429 - {'error': {'message': 'You exceeded your current quota, please check your plan and billing details. For more information on this error, read the docs: https://platform.openai.com/docs/guides/error-codes/api-errors.', 'type': 'insufficient_quota', 'param': None, 'code': 'insufficient_quota'}}

Traceback:

File "/usr/local/lib/python3.11/site-packages/streamlit/runtime/scriptrunner/exec_code.py", line 75, in exec_func_with_error_handling
  result = func()
  ^^^^^^

File "/usr/local/lib/python3.11/site-packages/streamlit/runtime/scriptrunner/script_runner.py", line 574, in code_to_exec
  exec(code, module.__dict__)

File "/app/docker_bot/bot.py", line 151, in <module>
  chat_input()

File "/app/docker_bot/bot.py", line 118, in chat_input
  result = generate_response(user_input)
  ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

File "/app/docker_bot/bot.py", line 34, in generate_response
  question_embedding = client.embeddings.create(
  ^^^^^^^^^^^^^^^^^^^^^^^^^^^^

File "/usr/local/lib/python3.11/site-packages/openai/resources/embeddings.py", line 114, in create
  return self._post(
  ^^^^^^^^^^^^

File "/usr/local/lib/python3.11/site-packages/openai/_base_client.py", line 1266, in post
  return cast(ResponseT, self.request(cast_to, opts, stream=stream, stream_cls=stream_cls))
  ^^

File "/usr/local/lib/python3.11/site-packages/openai/_base_client.py", line 942, in request

```
    return self._request(
           ^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.11/site-packages/openai/_base_client.py", line 1031, in _request
    return self._retry_request(
           ^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.11/site-packages/openai/_base_client.py", line 1079, in _retry_request
    return self._request(
           ^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.11/site-packages/openai/_base_client.py", line 1031, in _request
    return self._retry_request(
           ^^^^^^^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.11/site-packages/openai/_base_client.py", line 1079, in _retry_request
    return self._request(
           ^^^^^^^^^^^^^^
  File "/usr/local/lib/python3.11/site-packages/openai/_base_client.py", line 1046, in _request
    raise self._make_status_error_from_response(err.response) from None
```