19953 Zhiyu Zhang

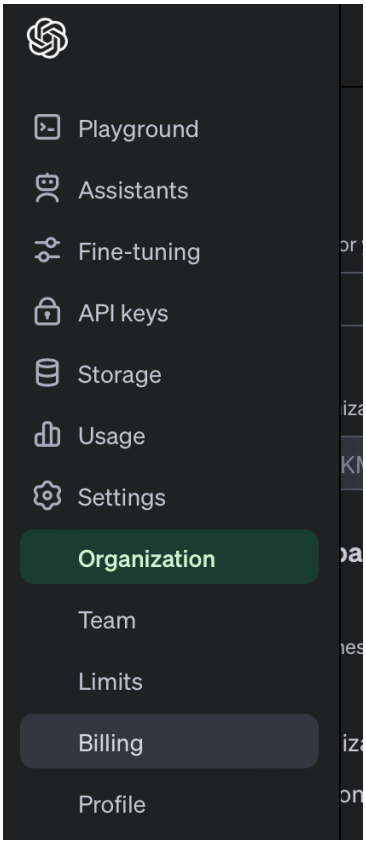
CS571: Week 11: Homework 2: GenAI - Containerized video transcription and chat app

1. Prerequisites

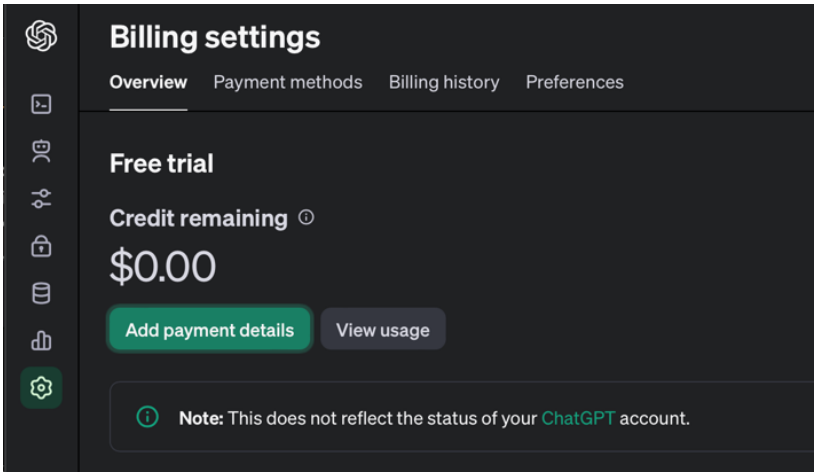
After the last project, you should already have installed Docker Desktop and had a git client. Make sure your Docker Desktop is running. Now we need to get an OpenAI API key and a Pinecone API Key.

2. Get an OpenAI API Key

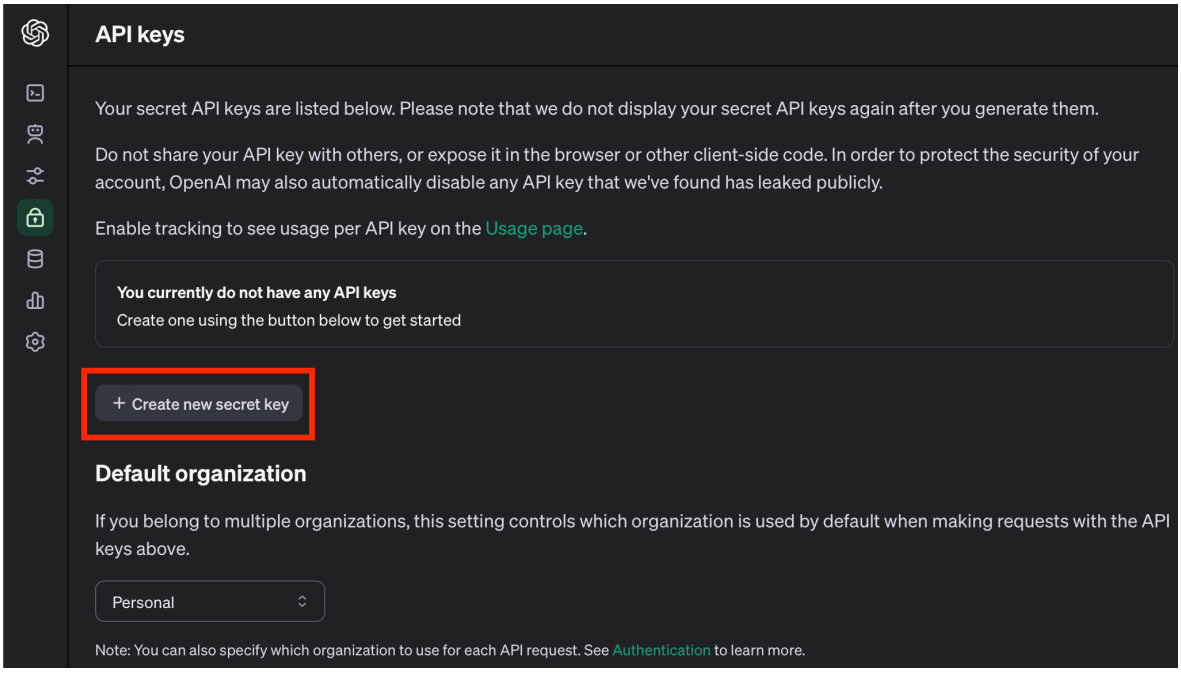
Use this link, log in your OpenAI account, then from the sidebar menu go to "Settings"-"Billing":



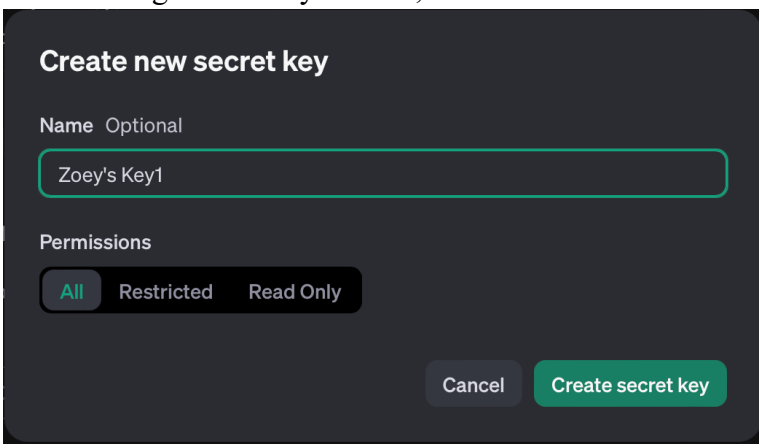
Select "Add payment details" and add some credit. The minimal amount is $5, which should be more than enough for this project. You can turn off auto recharge to avoid future charging.

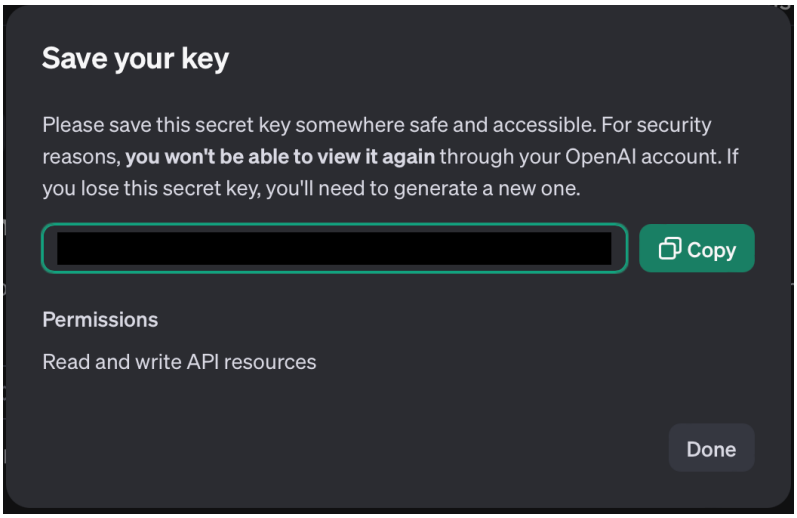


Now that we have enough credit, go to "API keys" from the sidebar menu and select "Create new secret key":



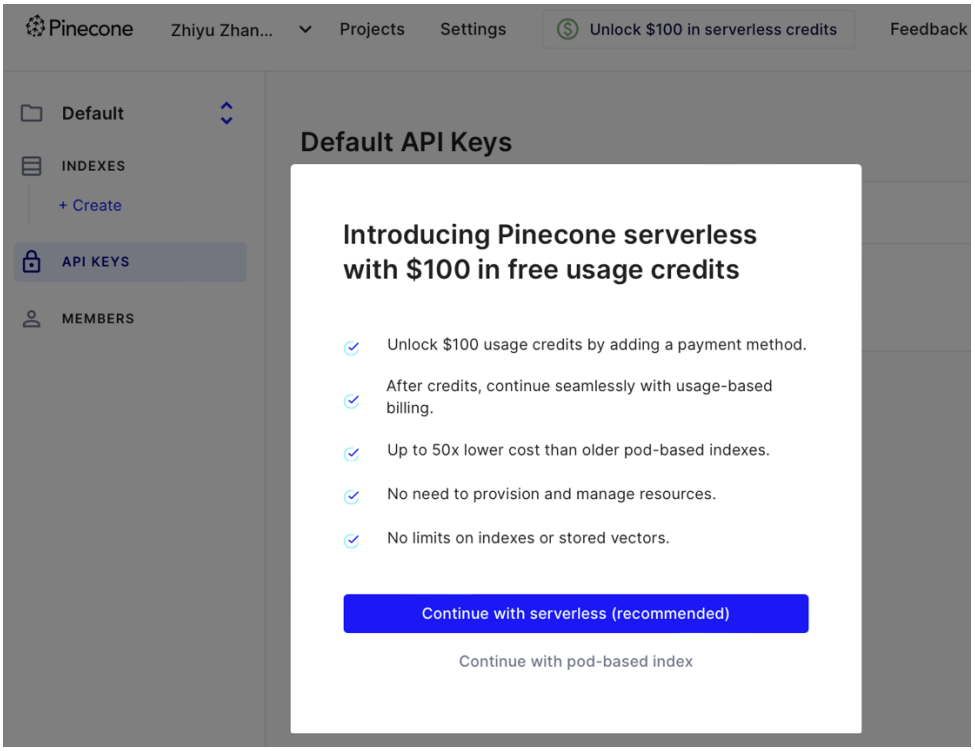
You can choose to give this key a name, then select “Create secret key” to get your API key:



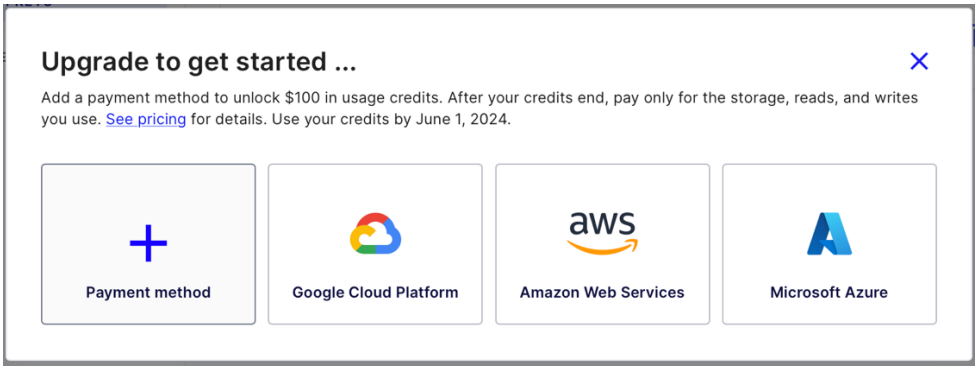


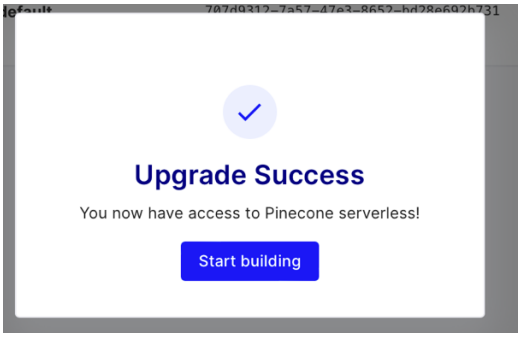
3. Get a Pinecone API Key

Use this link, log in or create your Pinecone account, then select “Continue with serverless” in the pop-up window, which gives you $100 serverless usage credit. You’ll need Pinecone serverless for this project since our application needs to use Pinecone to create serverless indexes later.

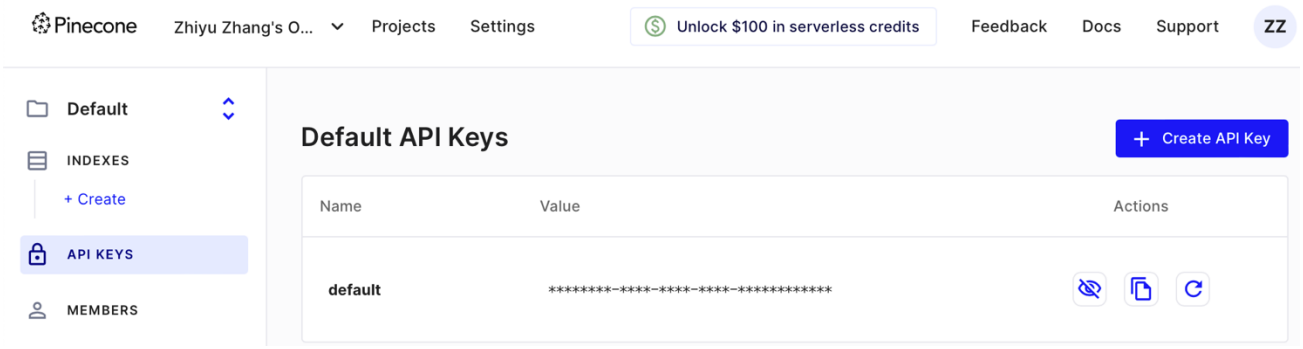


Add a payment method to unlock the credits:





Now go to API KEYS from the side bar to get your default API Key:



4. Get the sample application

Now we have all the prerequisites we need, we can start with our application. In terminal, go to a desired directory:

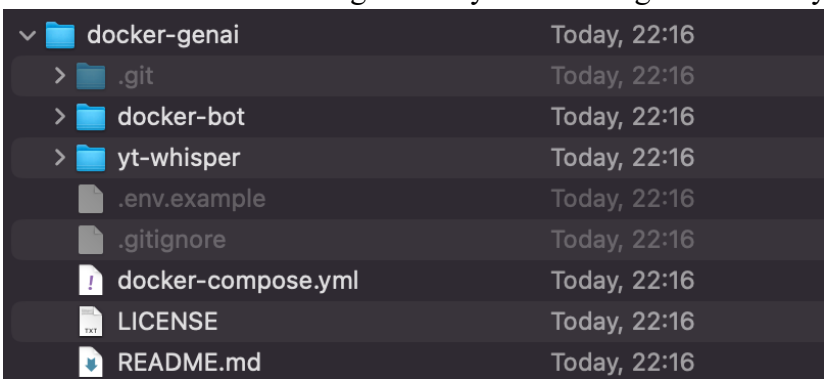
```
% cd ~/SFBU/2024Spring/CS571/w11
```

Clone the sample application:

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

```
(base) zhangzhiyu@shengchligongju ~ % cd ~/SFBU/2024Spring/CS571/w11
(base) zhangzhiyu@shengchligongju w11 % git clone https://github.com/Davidnet/do
cker-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 | 498.00 KiB/s, done.
Resolving deltas: 100% (24/24), done.
```

You should now have the following files in your docker-genai directory:



5. Specify your API keys

Go to docker-genai directory:

```
% cd docker-genai
```

Create a text file ".env":

```
% vim .env
```

```
(base) zhangzhiyu@shengchligongju w11 % cd docker-genai
(base) zhangzhiyu@shengchligongju docker-genai % vim .env
```

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
```

```
docker-genai — vim .env — 80×24
#------------------------------------------------------------------------------
# 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
~
~
```

6. Build and run the application

```
% docker compose up –build
```

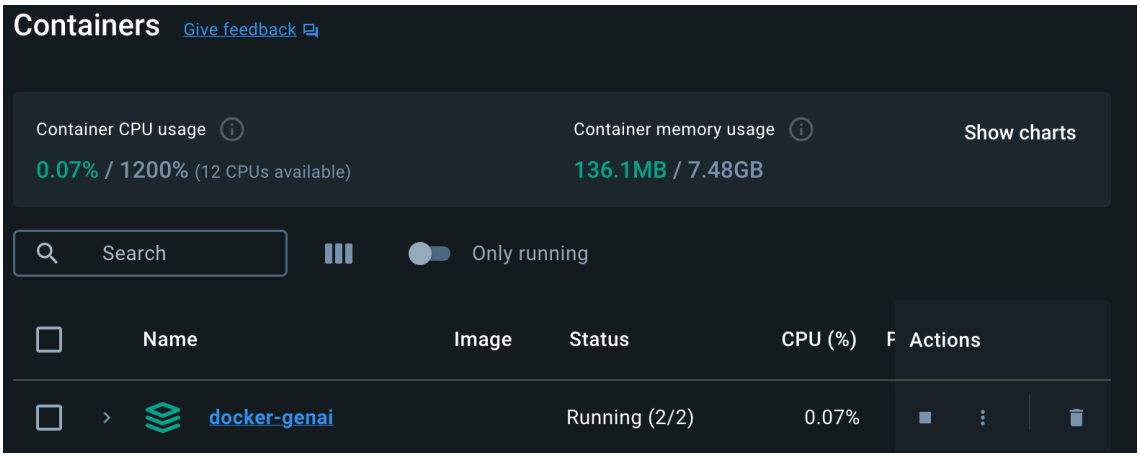
You will see something like this while Docker is building the application:

```
(base) zhangzhiyu@shengchligongju docker-genai % docker compose up --build
[+] Building 194.3s (20/20) FINISHED                        docker:desktop-linux
 => [bot internal] load build definition from Dockerfile                    0.0s
 => => transferring dockerfile: 1.83kB                                      0.0s
 => [yt-whisper internal] load build definition from Dockerfile             0.0s
 => => transferring dockerfile: 1.83kB                                      0.0s
 => [yt-whisper] resolve image config for docker.io/docker/dockerfile:1     1.6s
 => [bot auth] docker/dockerfile:pull token for registry-1.docker.io        0.0s
 => CACHED [yt-whisper] docker-image://docker.io/docker/dockerfile:1@sha2   0.0s
 => [yt-whisper internal] load metadata for docker.io/library/python:3.11   2.1s
 => [bot auth] library/python:pull token for registry-1.docker.io           0.0s
 => [bot internal] load .dockerignore                                       0.0s
 => => transferring context: 2B                                             0.0s
 => [yt-whisper internal] load .dockerignore                                0.0s
 => => transferring context: 2B                                             0.0s
 => [yt-whisper base 1/5] FROM docker.io/library/python:3.11-slim@sha256   25.8s
```

Once the application is running, you'll see something like this:

```
 ✔ Network docker-genai_default          Created                            0.1s
 ✔ Container docker-genai-bot-1          Created                            0.1s
 ✔ Container docker-genai-yt-whisper-1   Created                            0.1s
Attaching to bot-1, yt-whisper-1
yt-whisper-1  |
yt-whisper-1  | Collecting usage statistics. To deactivate, set browser.gatherUs
ageStats to False.
bot-1         |
bot-1         | Collecting usage statistics. To deactivate, set browser.gatherUs
ageStats to False.
bot-1         |
yt-whisper-1  |
yt-whisper-1  |
yt-whisper-1  |   You can now view your Streamlit app in your browser.
yt-whisper-1  |
yt-whisper-1  |   URL: http://0.0.0.0:8503
yt-whisper-1  |
bot-1         |
bot-1         |   You can now view your Streamlit app in your browser.
bot-1         |
bot-1         |   URL: http://0.0.0.0:8504
bot-1         |
```

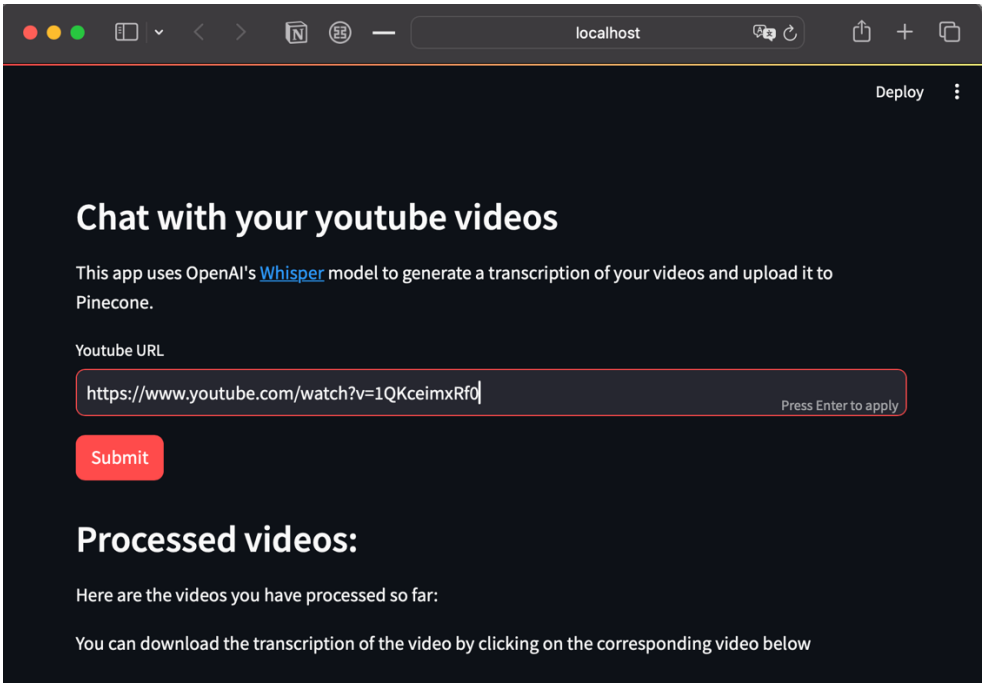
You should also be able to see the container image running in Docker Desktop:



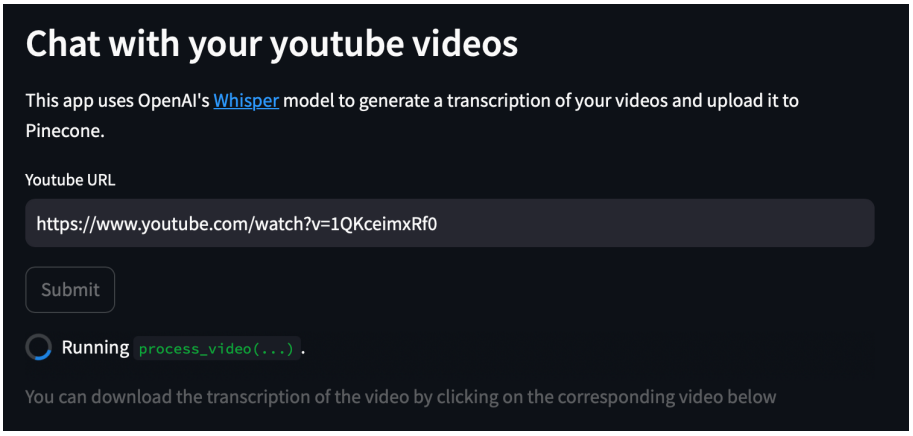
7. Use the yt-whisper service

Open a browser and access the yt-whisper service at http://localhost:8503.

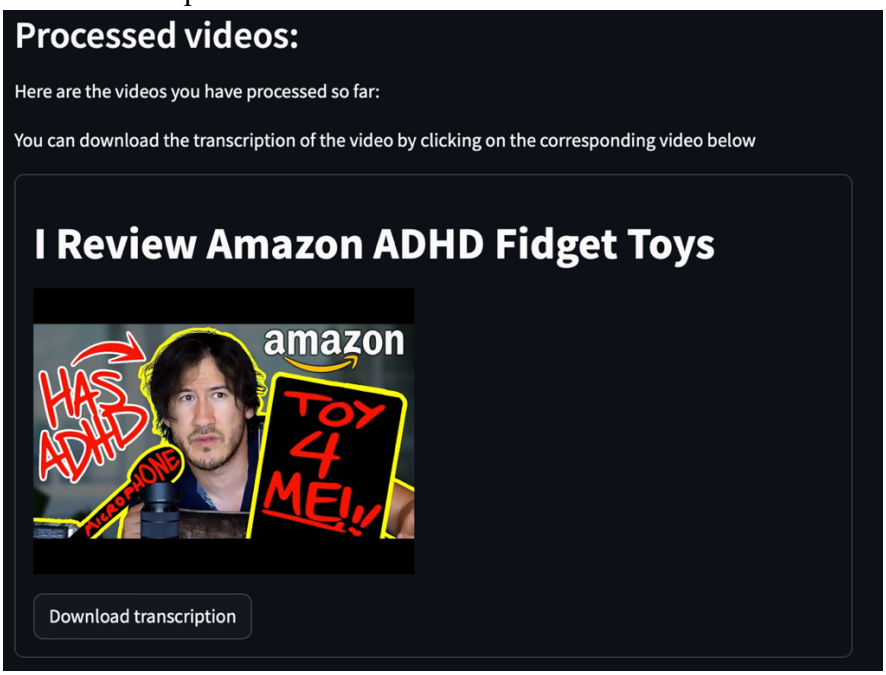
Enter the Youtube video URL you want to use and select “Submit”:



Here I used https://www.youtube.com/watch?v=1QKceimxRf0. Give it a couple minutes to process the video:



After processing, you'll see a list of videos that have been indexed in Pinecone. It also provides a button to download the transcription:



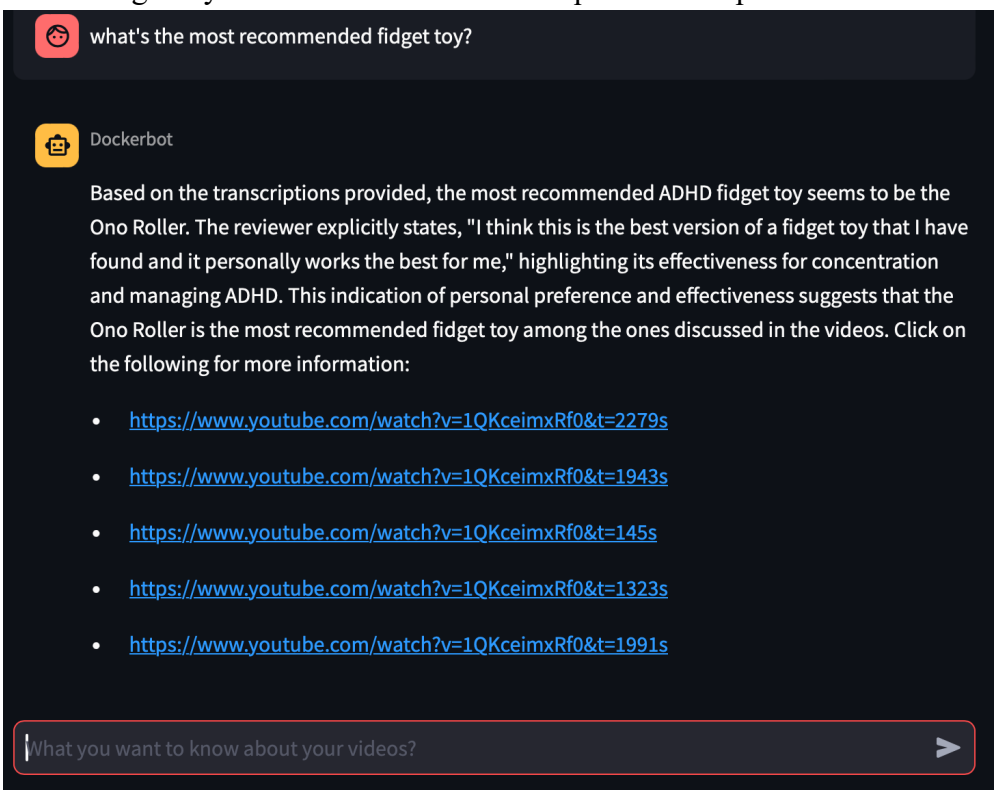
You can also see messages in terminal:

```
yt-whisper-1  | 2024-03-31 06:16:12.216 Processing video: https://youtube.com/watch?v=1QK
ceimxRf0
yt-whisper-1  | 2024-03-31 06:16:24.111 File size(bytes): 16103370
yt-whisper-1  | 2024-03-31 06:16:24.112 File name: /tmp/tmpnybm5sot/I Review Amazon ADHD
Fidget Toys.mp4
yt-whisper-1  | 2024-03-31 06:19:14.873 Transcription done
```

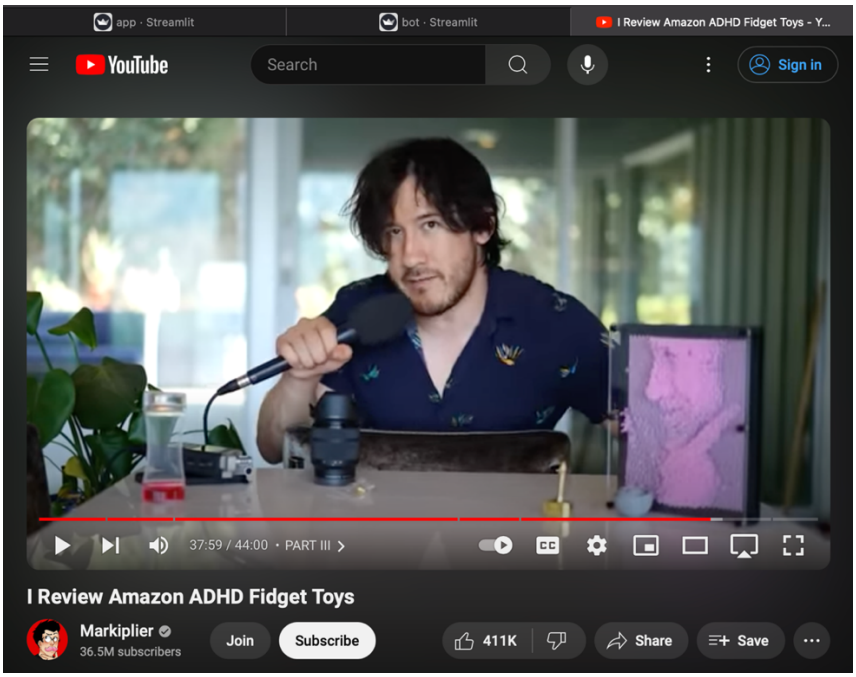
8. Use the dockerbot service

Open a browser and access the dockerbot service at http://localhost:8504.

Enter the question you want to ask about the video that was processed. Here I asked “what's the most recommended fidget toy?”. Dockerbot answers the question and provides links to the video with timestamps:



You can click on any of the links in the answer to go to the part in the video that answers your question:



9. Stop the application

In terminal, press ctrl + C to stop the application:

```
^CGracefully stopping... (press Ctrl+C again to force)
[+] Stopping 0/0
 ⠏ Container docker-genai-bot-1         Stopping                    0.1s
 ⠏ Container docker-genai-yt-whisper-1  St...                       0.1s
[+] Killing 2/22
 ✔ Container docker-genai-yt-whisper-1  Ki...                       0.3s
 ✔ Container docker-genai-bot-1         Killed                      0.2s
canceled
```

You can also see the container stops running in Docker Desktop: