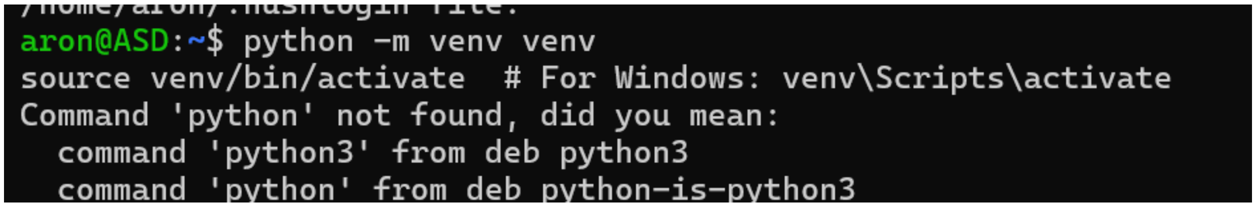
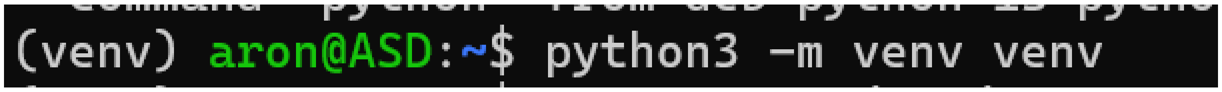
## Step 1: Set Up Your Environment

1. **Create a Project Folder**: Organize your files in a project folder, such as `gpt4o_chatbot`.
2. **Set Up a Virtual Environment**: Inside the folder, create a virtual environment:

```
python -m venv venv
source venv/bin/activate  # For Windows: venv\Scripts\activate
```



```
aron@ASD:~$ python -m venv venv
source venv/bin/activate  # For Windows: venv\Scripts\activate
Command 'python' not found, did you mean:
  command 'python3' from deb python3
  command 'python' from deb python-is-python3
```

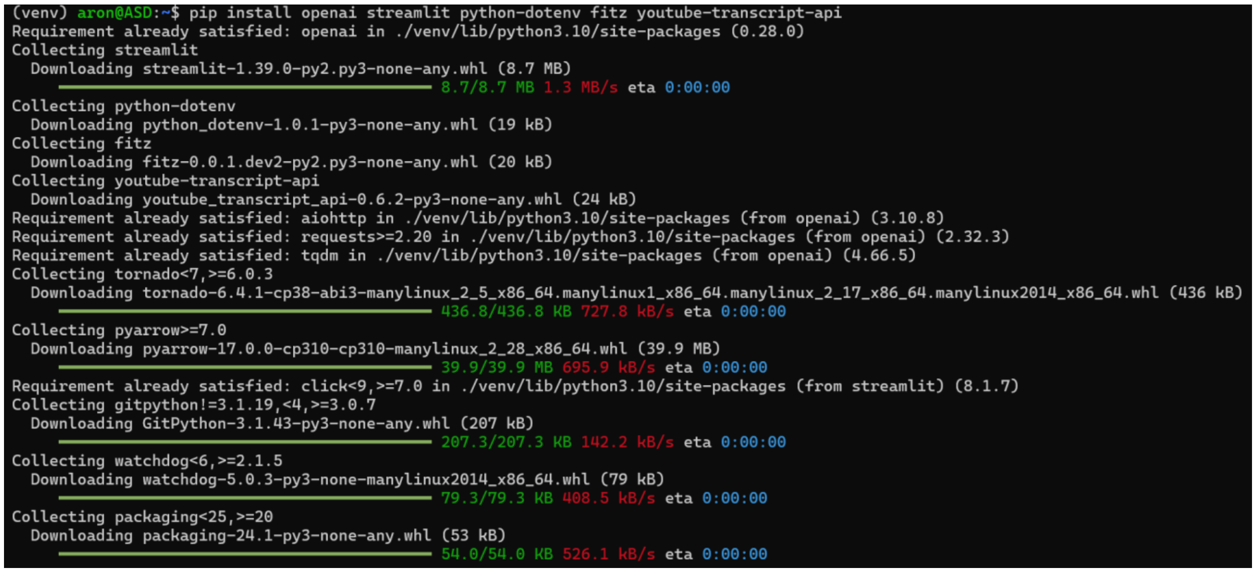


```
(venv) aron@ASD:~$ python3 -m venv venv
```

3. **Install Dependencies**: Install the required packages using pip:

```
pip install openai streamlit python-dotenv fitz youtube-transcript-api
```

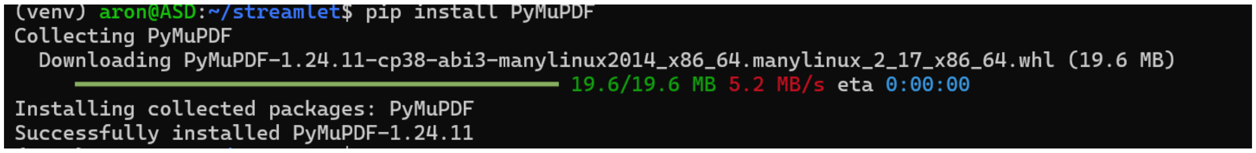
- `openai`: For interacting with the OpenAI API.
- `streamlit`: To create a web-based user interface.
- `python-dotenv`: To handle environment variables.
- `fitz`: For PDF processing.
- `youtube-transcript-api`: To extract transcripts from YouTube videos.



```
(venv) aron@ASD:~$ pip install openai streamlit python-dotenv fitz youtube-transcript-api
Requirement already satisfied: openai in ./venv/lib/python3.10/site-packages (0.28.0)
Collecting streamlit
  Downloading streamlit-1.39.0-py2.py3-none-any.whl (8.7 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.7/8.7 MB 1.3 MB/s eta 0:00:00
Collecting python-dotenv
  Downloading python_dotenv-1.0.1-py3-none-any.whl (19 kB)
Collecting fitz
  Downloading fitz-0.0.1.dev2-py2.py3-none-any.whl (20 kB)
Collecting youtube-transcript-api
  Downloading youtube_transcript_api-0.6.2-py3-none-any.whl (24 kB)
Requirement already satisfied: aiohttp in ./venv/lib/python3.10/site-packages (from openai) (3.10.8)
Requirement already satisfied: requests>=2.20 in ./venv/lib/python3.10/site-packages (from openai) (2.32.3)
Requirement already satisfied: tqdm in ./venv/lib/python3.10/site-packages (from openai) (4.66.5)
Collecting tornado<7,>=6.0.3
  Downloading tornado-6.4.1-cp38-abi3-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_2_17_x86_64.manylinux2014_x86_64.whl (436 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 436.8/436.8 KB 727.8 kB/s eta 0:00:00
Collecting pyarrow>=7.0
  Downloading pyarrow-17.0.0-cp310-cp310-manylinux_2_28_x86_64.whl (39.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 39.9/39.9 MB 695.9 kB/s eta 0:00:00
Requirement already satisfied: click<9,>=7.0 in ./venv/lib/python3.10/site-packages (from streamlit) (8.1.7)
Collecting gitpython!=3.1.19,<4,>=3.0.7
  Downloading GitPython-3.1.43-py3-none-any.whl (207 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 207.3/207.3 KB 142.2 kB/s eta 0:00:00
Collecting watchdog<6,>=2.1.5
  Downloading watchdog-5.0.3-py3-none-manylinux2014_x86_64.whl (79 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 79.3/79.3 KB 408.5 kB/s eta 0:00:00
Collecting packaging<25,>=20
  Downloading packaging-24.1-py3-none-any.whl (53 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 54.0/54.0 KB 526.1 kB/s eta 0:00:00
```

Install PyMuPDF Correctly: Install PyMuPDF using:

```
pip install PyMuPDF
```



```
(venv) aron@ASD:~/streamlet$ pip install PyMuPDF
Collecting PyMuPDF
  Downloading PyMuPDF-1.24.11-cp38-abi3-manylinux2014_x86_64.manylinux_2_17_x86_64.whl (19.6 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 19.6/19.6 MB 5.2 MB/s eta 0:00:00
Installing collected packages: PyMuPDF
Successfully installed PyMuPDF-1.24.11
```

## Step 2: Obtain an API Key

1. **Create/Open a `.env` file** in your project folder and add your OpenAI API key:

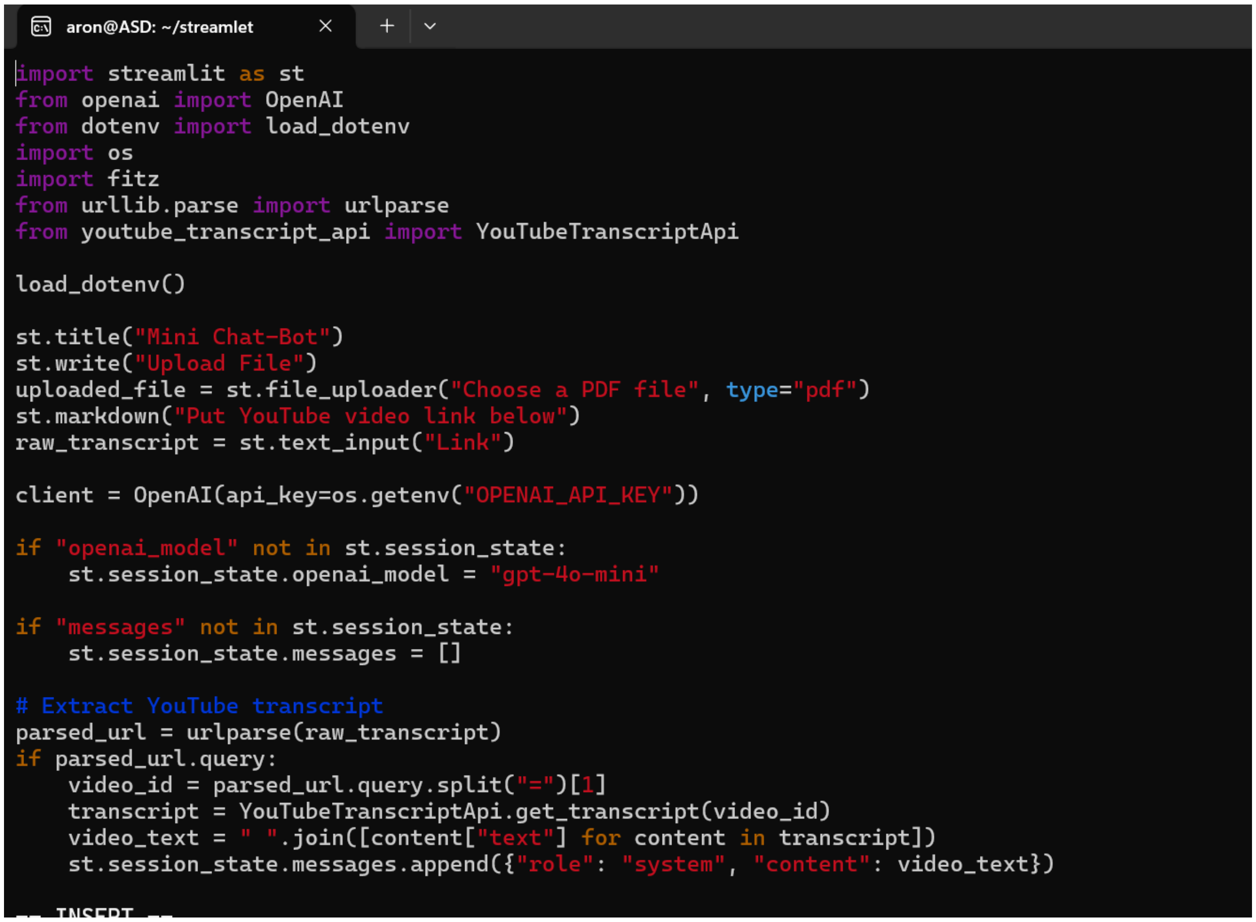
```
OPENAI_API_KEY=your_openai_api_key
```

```
(venv) aron@ASD:~$ OPENAI_API_KEY=sk-pro
```

## Step 3: Create the `chatbot.py` Script

1. **Create a Python Script**: Name it `chatbot.py` and add the following code:

```python
import streamlit as st
from openai import OpenAI
from dotenv import load_dotenv
import os
import fitz
from urllib.parse import urlparse
from youtube_transcript_api import YouTubeTranscriptApi

load_dotenv()

st.title("Mini Chat-Bot")
st.write("Upload File")
uploaded_file = st.file_uploader("Choose a PDF file", type="pdf")
st.markdown("Put YouTube video link below")
raw_transcript = st.text_input("Link")

client = OpenAI(api_key=os.getenv("OPENAI_API_KEY"))

if "openai_model" not in st.session_state:
    st.session_state.openai_model = "gpt-4o-mini"

if "messages" not in st.session_state:
    st.session_state.messages = []

# Extract YouTube transcript
parsed_url = urlparse(raw_transcript)
if parsed_url.query:
    video_id = parsed_url.query.split("=")[1]
    transcript = YouTubeTranscriptApi.get_transcript(video_id)
    video_text = " ".join([content["text"] for content in transcript])
    st.session_state.messages.append({"role": "system", "content": video_text})

    if st.button("Summarize Video"):
        summarized_video = client.chat.completions.create(
            model=st.session_state.openai_model,
            messages=[{"role": "user", "content": f"Summarize the following YouTube video content: {video_text}"}]
        )
        st.session_state.messages.append({"role": "assistant", "content": summarized_video.choices[0].message.content})

# Extract text from uploaded PDF
if uploaded_file is not None:
    docs = fitz.open(stream=uploaded_file.read(), filetype="pdf")
    pdf_text = "".join([page.get_text() for page in docs])
    st.session_state.messages.append({"role": "system", "content": pdf_text})

    if st.button("Summarize Text"):
        summarize_text = client.chat.completions.create(
            model=st.session_state.openai_model,
            messages=[{"role": "user", "content": f"Summarize the following content: {pdf_text}"}]
        )
        st.session_state.messages.append({"role": "assistant", "content": summarize_text.choices[0].message.content})

# Display conversation
for message in st.session_state.messages:
    if message["role"] != "system":
        with st.chat_message(message["role"]):
            st.markdown(message["content"])

# Chatbot interaction
if prompt := st.chat_input("Enter message here"):
    st.session_state.messages.append({"role": "user", "content": prompt})
    response = client.chat.completions.create(
        model=st.session_state.openai_model,
        messages=[{"role": msg["role"], "content": msg["content"]} for msg in st.session_state.messages]
    )
    st.session_state.messages.append({"role": "assistant", "content": response.choices[0].message.content})
```

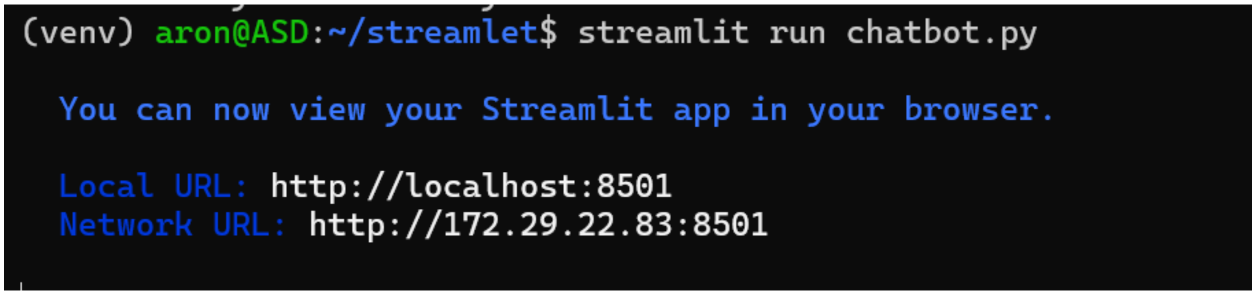


```
aron@ASD: ~/streamlet
import streamlit as st
from openai import OpenAI
from dotenv import load_dotenv
import os
import fitz
from urllib.parse import urlparse
from youtube_transcript_api import YouTubeTranscriptApi

load_dotenv()

st.title("Mini Chat-Bot")
st.write("Upload File")
uploaded_file = st.file_uploader("Choose a PDF file", type="pdf")
st.markdown("Put YouTube video link below")
raw_transcript = st.text_input("Link")

client = OpenAI(api_key=os.getenv("OPENAI_API_KEY"))

if "openai_model" not in st.session_state:
    st.session_state.openai_model = "gpt-4o-mini"

if "messages" not in st.session_state:
    st.session_state.messages = []

# Extract YouTube transcript
parsed_url = urlparse(raw_transcript)
if parsed_url.query:
    video_id = parsed_url.query.split("=")[1]
    transcript = YouTubeTranscriptApi.get_transcript(video_id)
    video_text = " ".join([content["text"] for content in transcript])
    st.session_state.messages.append({"role": "system", "content": video_text})

-- INSERT --
```

## Step 4: Run the Application

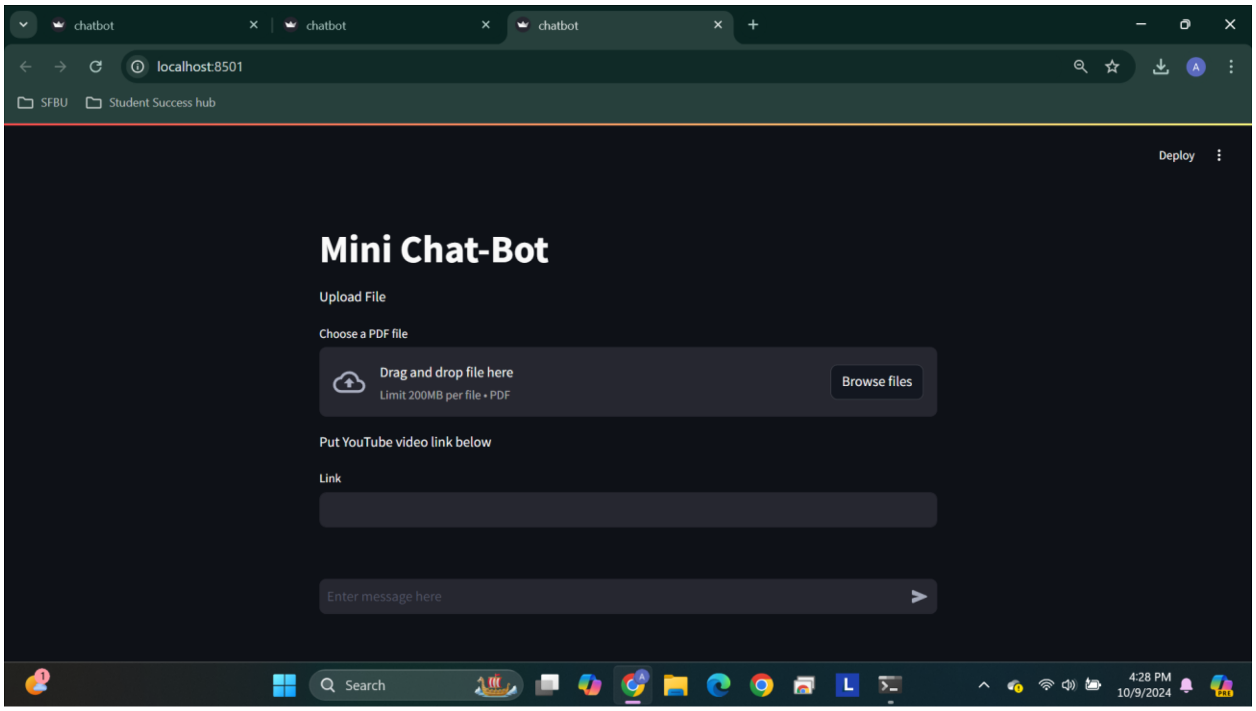
1. **Run the Script**: Start the Streamlit app by running the following command in your terminal:

```
1. streamlit run chatbot.py
```



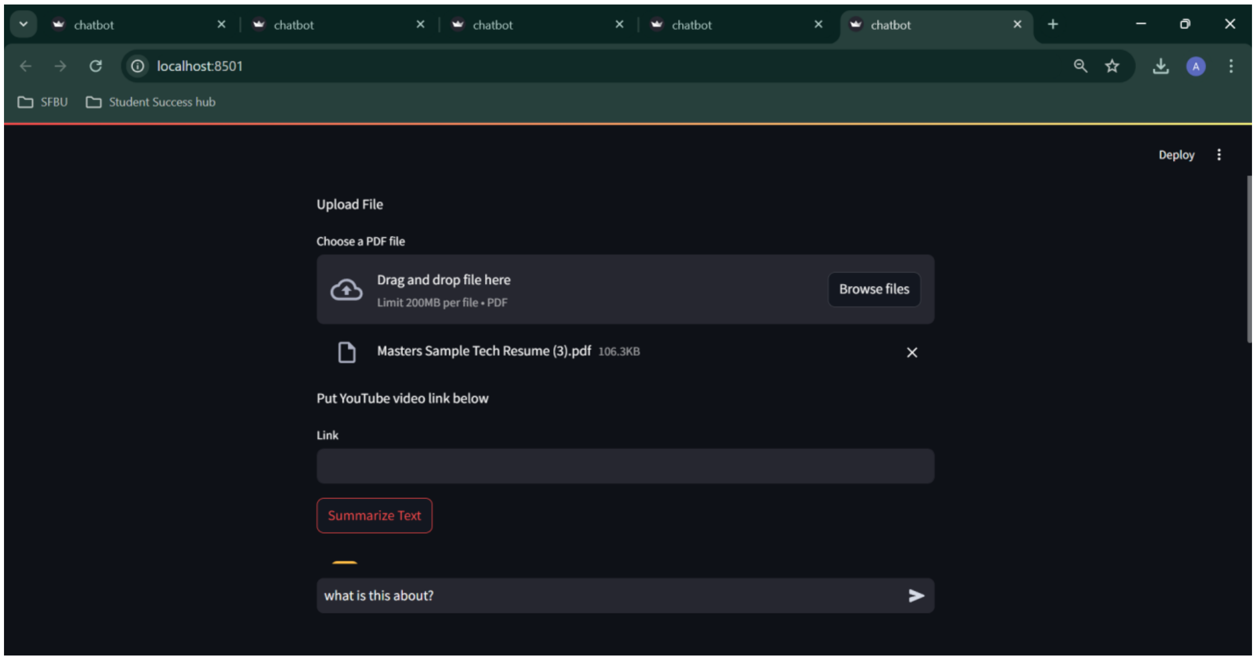
```
(venv) aron@ASD:~/streamlet$ streamlit run chatbot.py

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8501
  Network URL: http://172.29.22.83:8501
```

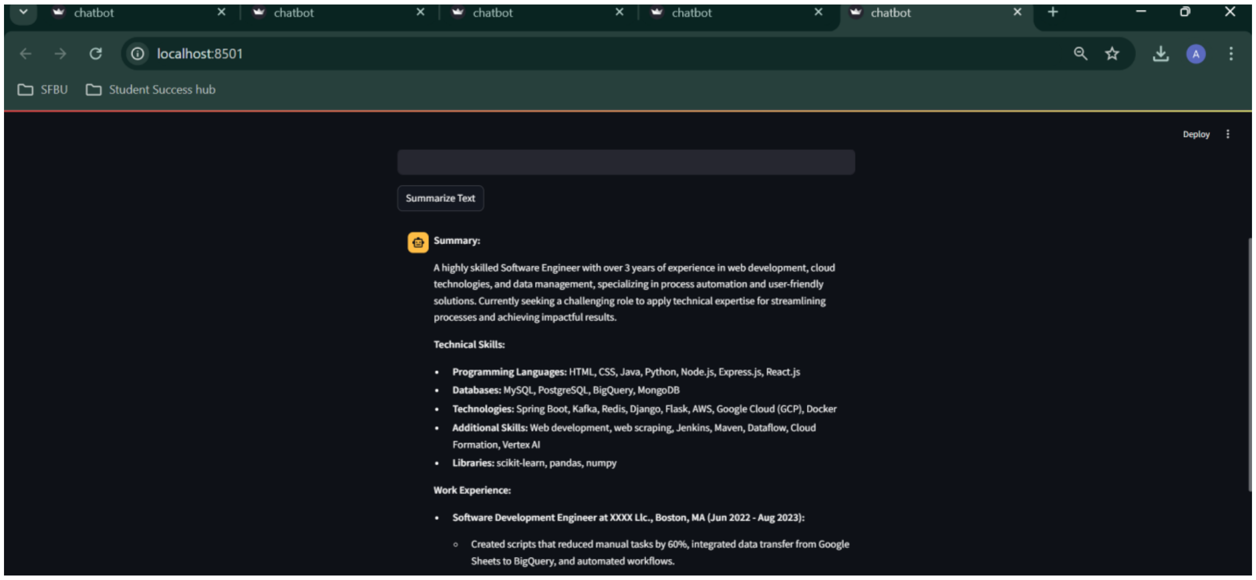
2. **Access the Web Interface**: Open the link provided in your terminal, typically `http://localhost:8501`.



## Step 5: Test the Functionality

1. **Upload a PDF**: Use the "Upload File" section to upload a PDF and click "Summarize Text" to get a summary.





**Summary:**

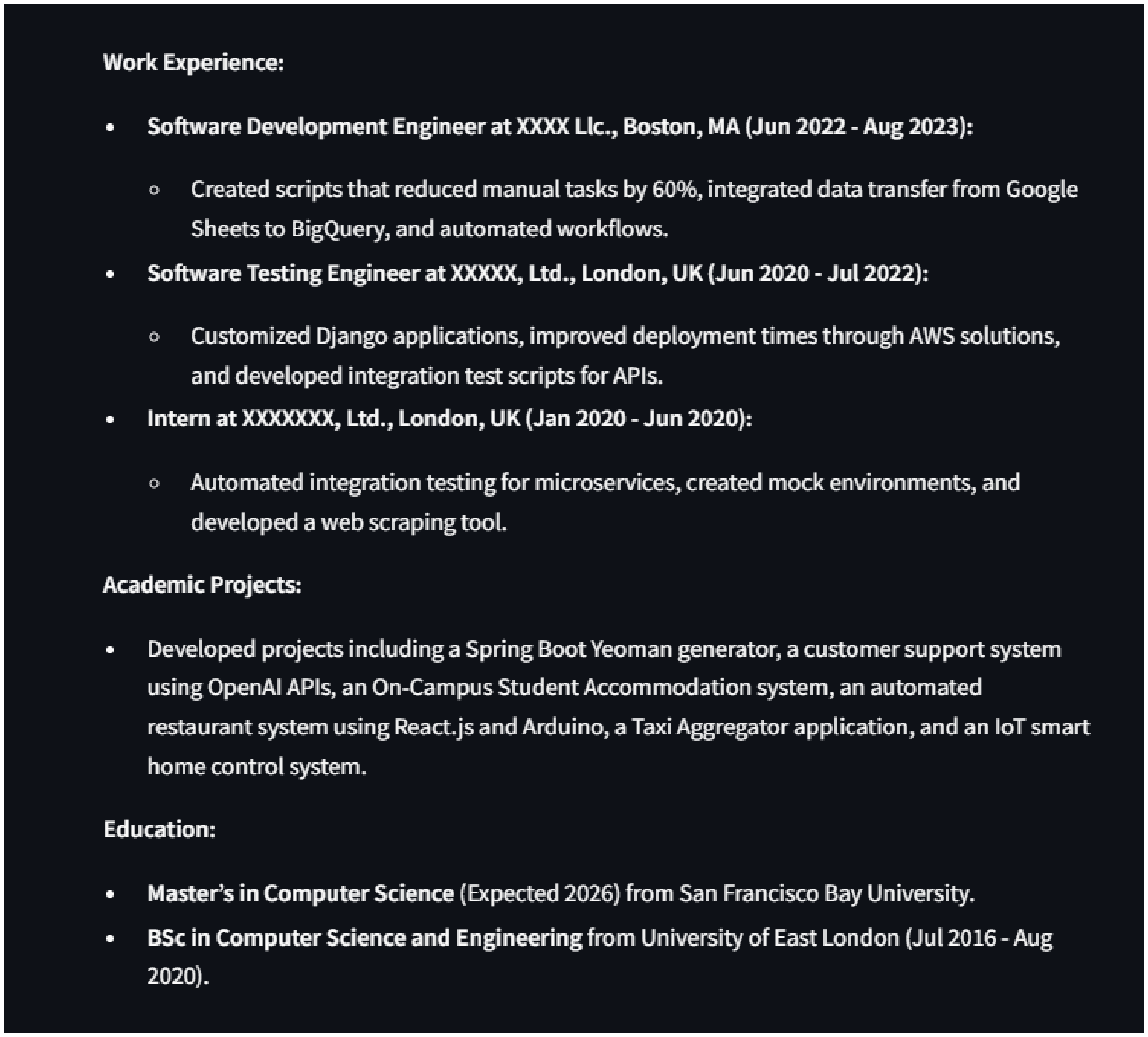
A highly skilled Software Engineer with over 3 years of experience in web development, cloud technologies, and data management, specializing in process automation and user-friendly solutions. Currently seeking a challenging role to apply technical expertise for streamlining processes and achieving impactful results.

**Technical Skills:**

- **Programming Languages:** HTML, CSS, Java, Python, Node.js, Express.js, React.js
- **Databases:** MySQL, PostgreSQL, BigQuery, MongoDB
- **Technologies:** Spring Boot, Kafka, Redis, Django, Flask, AWS, Google Cloud (GCP), Docker
- **Additional Skills:** Web development, web scraping, Jenkins, Maven, Dataflow, Cloud Formation, Vertex AI
- **Libraries:** scikit-learn, pandas, numpy

**Work Experience:**

- **Software Development Engineer at XXXX Llc., Boston, MA (Jun 2022 - Aug 2023):**
  - Created scripts that reduced manual tasks by 60%, integrated data transfer from Google Sheets to BigQuery, and automated workflows.



**Work Experience:**

- **Software Development Engineer at XXXX Llc., Boston, MA (Jun 2022 - Aug 2023):**
  - Created scripts that reduced manual tasks by 60%, integrated data transfer from Google Sheets to BigQuery, and automated workflows.
- **Software Testing Engineer at XXXXX, Ltd., London, UK (Jun 2020 - Jul 2022):**
  - Customized Django applications, improved deployment times through AWS solutions, and developed integration test scripts for APIs.
- **Intern at XXXXXXX, Ltd., London, UK (Jan 2020 - Jun 2020):**
  - Automated integration testing for microservices, created mock environments, and developed a web scraping tool.

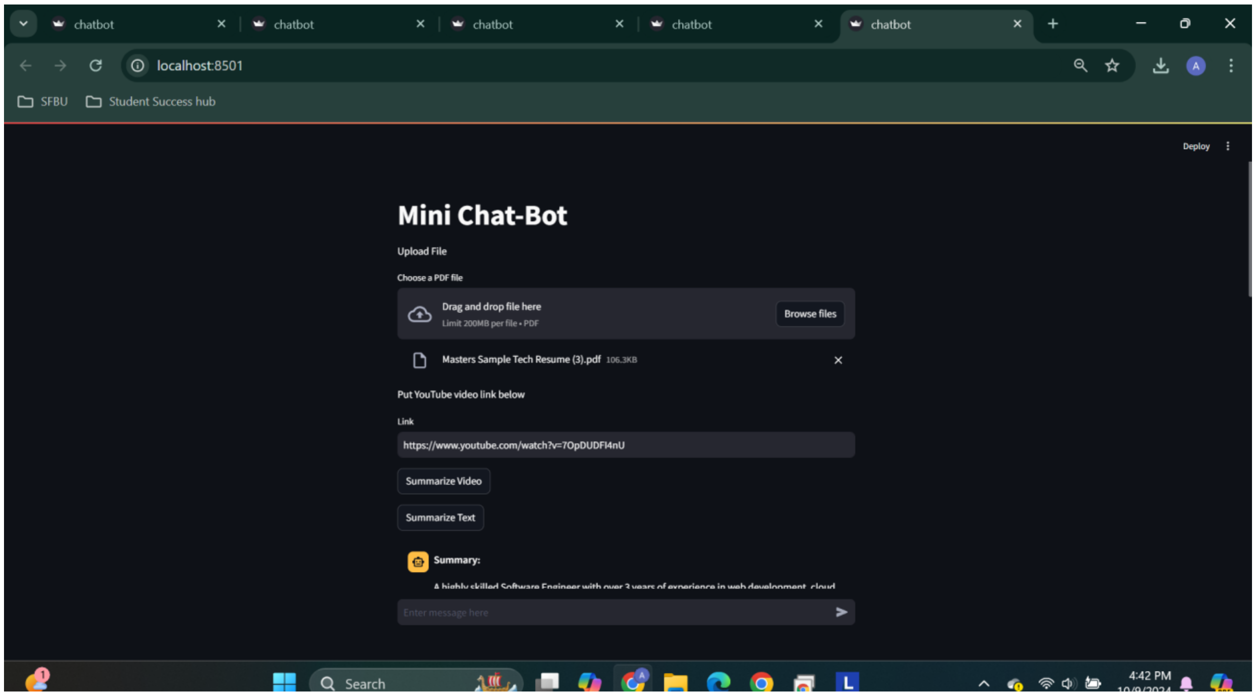
**Academic Projects:**

- Developed projects including a Spring Boot Yeoman generator, a customer support system using OpenAI APIs, an On-Campus Student Accommodation system, an automated restaurant system using React.js and Arduino, a Taxi Aggregator application, and an IoT smart home control system.

**Education:**

- **Master's in Computer Science** (Expected 2026) from San Francisco Bay University.
- **BSc in Computer Science and Engineering** from University of East London (Jul 2016 - Aug 2020).

2. **Provide a YouTube Link**: Enter a YouTube link with a transcript in the input field and click "Summarize Video."



Mini Chat-Bot

Upload File

Choose a PDF file

Drag and drop file here
Limit 200MB per file • PDF

Masters Sample Tech Resume (3).pdf 106.3KB

Put YouTube video link below

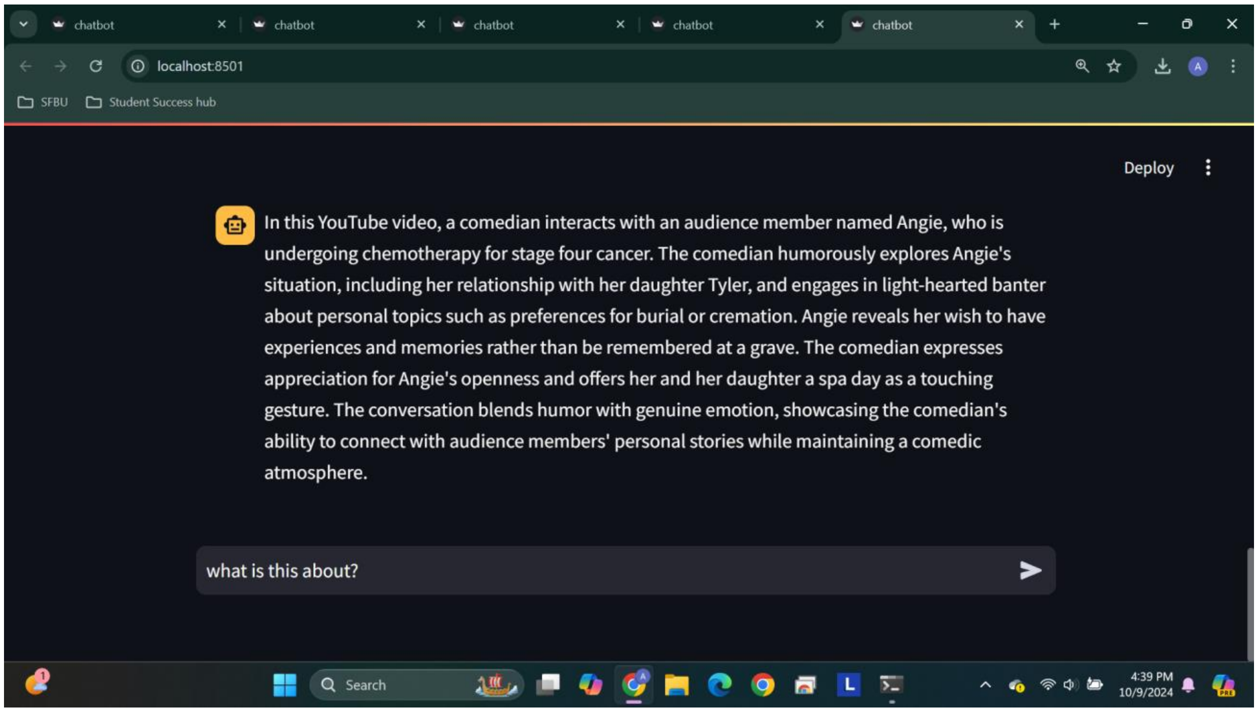
Link

https://www.youtube.com/watch?v=7OpDUDFI4nU

**Summary:**

A highly skilled Software Engineer with over 3 years of experience in web development, cloud

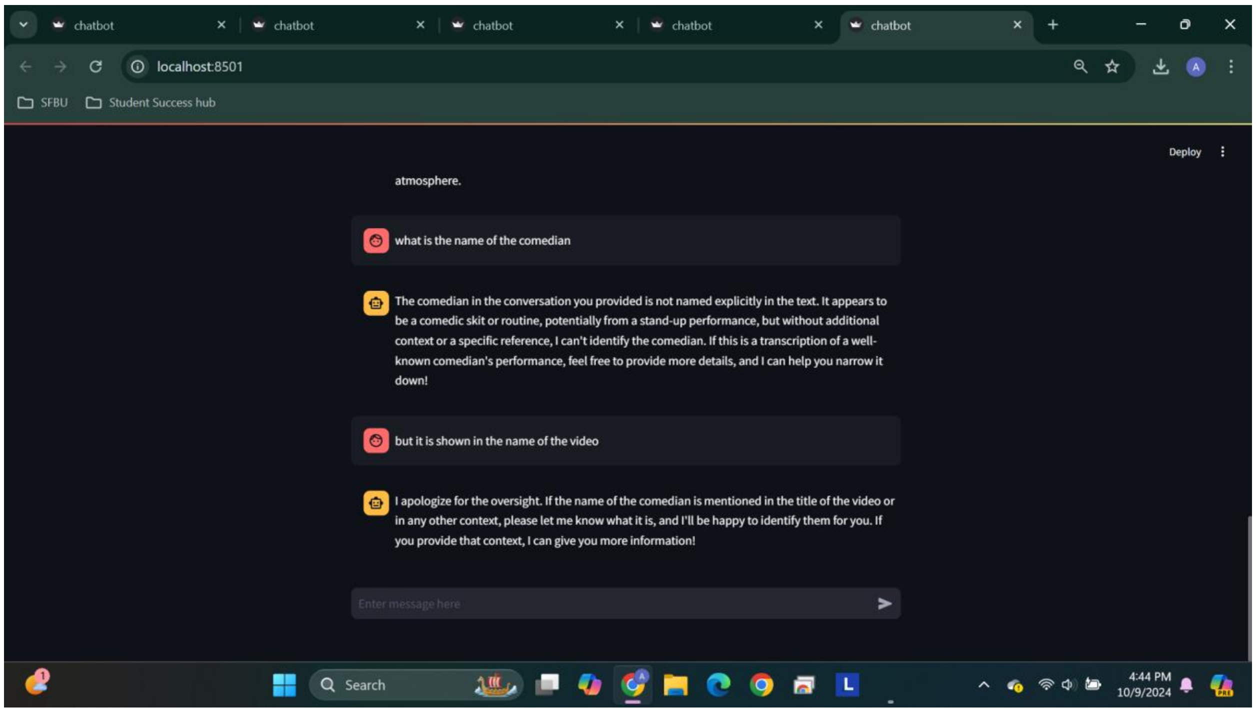
Enter message here



In this YouTube video, a comedian interacts with an audience member named Angie, who is undergoing chemotherapy for stage four cancer. The comedian humorously explores Angie's situation, including her relationship with her daughter Tyler, and engages in light-hearted banter about personal topics such as preferences for burial or cremation. Angie reveals her wish to have experiences and memories rather than be remembered at a grave. The comedian expresses appreciation for Angie's openness and offers her and her daughter a spa day as a touching gesture. The conversation blends humor with genuine emotion, showcasing the comedian's ability to connect with audience members' personal stories while maintaining a comedic atmosphere.

what is this about?

3. **Chat with the Bot**: Use the input field to enter queries, and the bot will respond using GPT-4o Mini.



atmosphere.

what is the name of the comedian

The comedian in the conversation you provided is not named explicitly in the text. It appears to be a comedic skit or routine, potentially from a stand-up performance, but without additional context or a specific reference, I can't identify the comedian. If this is a transcription of a well-known comedian's performance, feel free to provide more details, and I can help you narrow it down!

but it is shown in the name of the video

I apologize for the oversight. If the name of the comedian is mentioned in the title of the video or in any other context, please let me know what it is, and I'll be happy to identify them for you. If you provide that context, I can give you more information!