



# Local LLM Deployment and Interaction Tutorial

## Demonstrating LLama3 and GPT4all Models

By: Anmol Valecha

June 2024

# Introduction

1

## Overview of Assignment

- Hands-on experience with local LLM deployment
- Importance of practical application in understanding model functionality

2

## Objectives

- Setup and interact with LLama3 and GPT4all locally
- Compare performance and usability of both models



Made with Gamma

# Lliling Lkhnaa



## LLama3 Model Setup

### Download from GitHub

Navigate to the LLama3 GitHub repository at <https://github.com/Ollama/llama3>. Download and extract the repository to your local machine.

1

### Verification using curl

Use the curl command to interact with the LLama3 model:

```
curl -X POST http://localhost:11434/api/chat
  \-H "Content-Type: application/json" \-d
    '{"model": "phi3", "messages": [], "stream":
      false}'
```

2

### Terminal Commands for Setup

Open a terminal and navigate to the LLama3 directory. Run the following commands to set up LLama3:

```
cd llama3 ./setup.sh
```

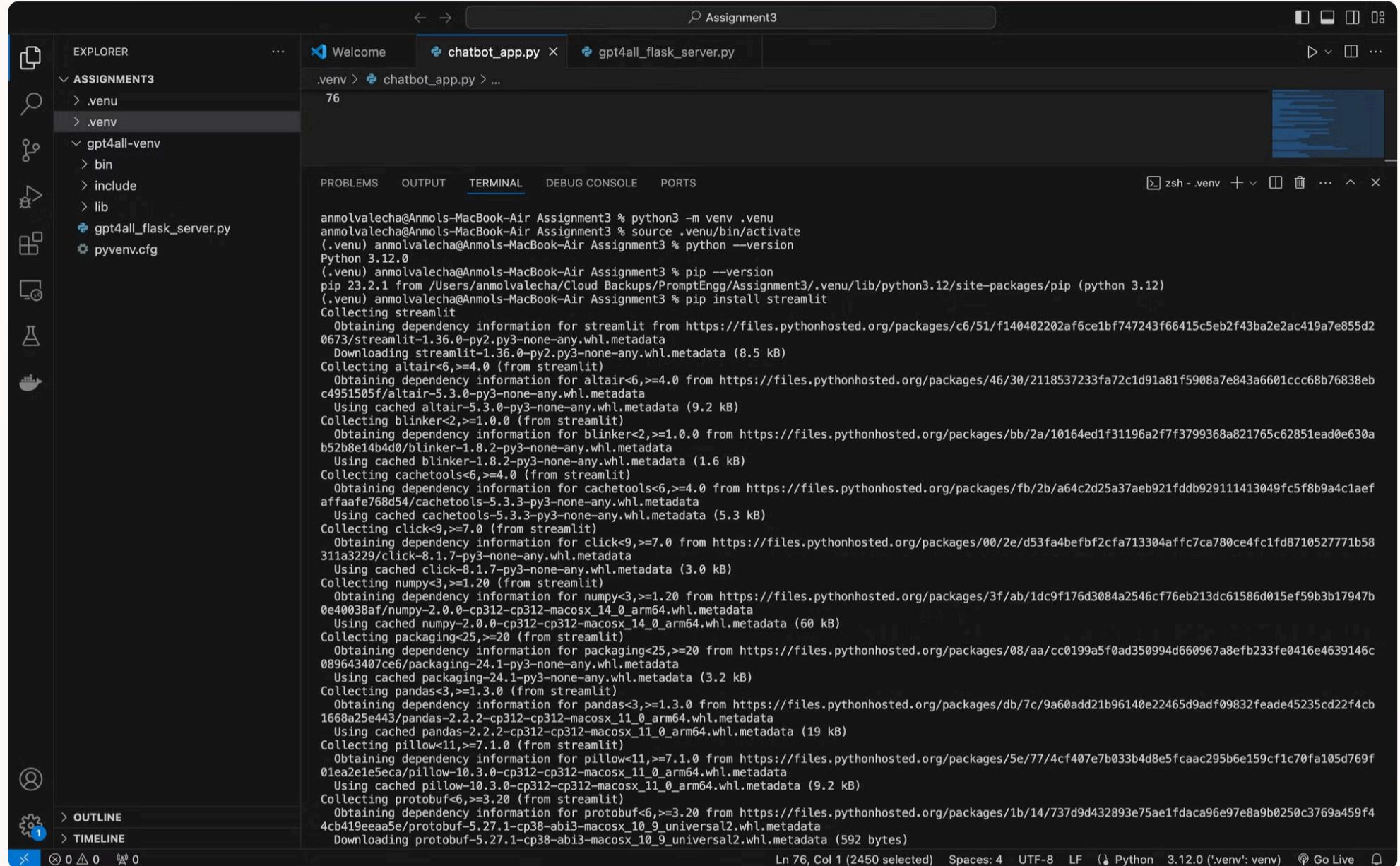
3

# Setup of Streamlit and Ollama3

- Setup a virtual environment

```
anmolvalecha@Anmols-MacBook-Air Assignment3 % python3 -m venv gpt4all-venv  
anmolvalecha@Anmols-MacBook-Air Assignment3 % source gpt4all-venv/bin/activate
```

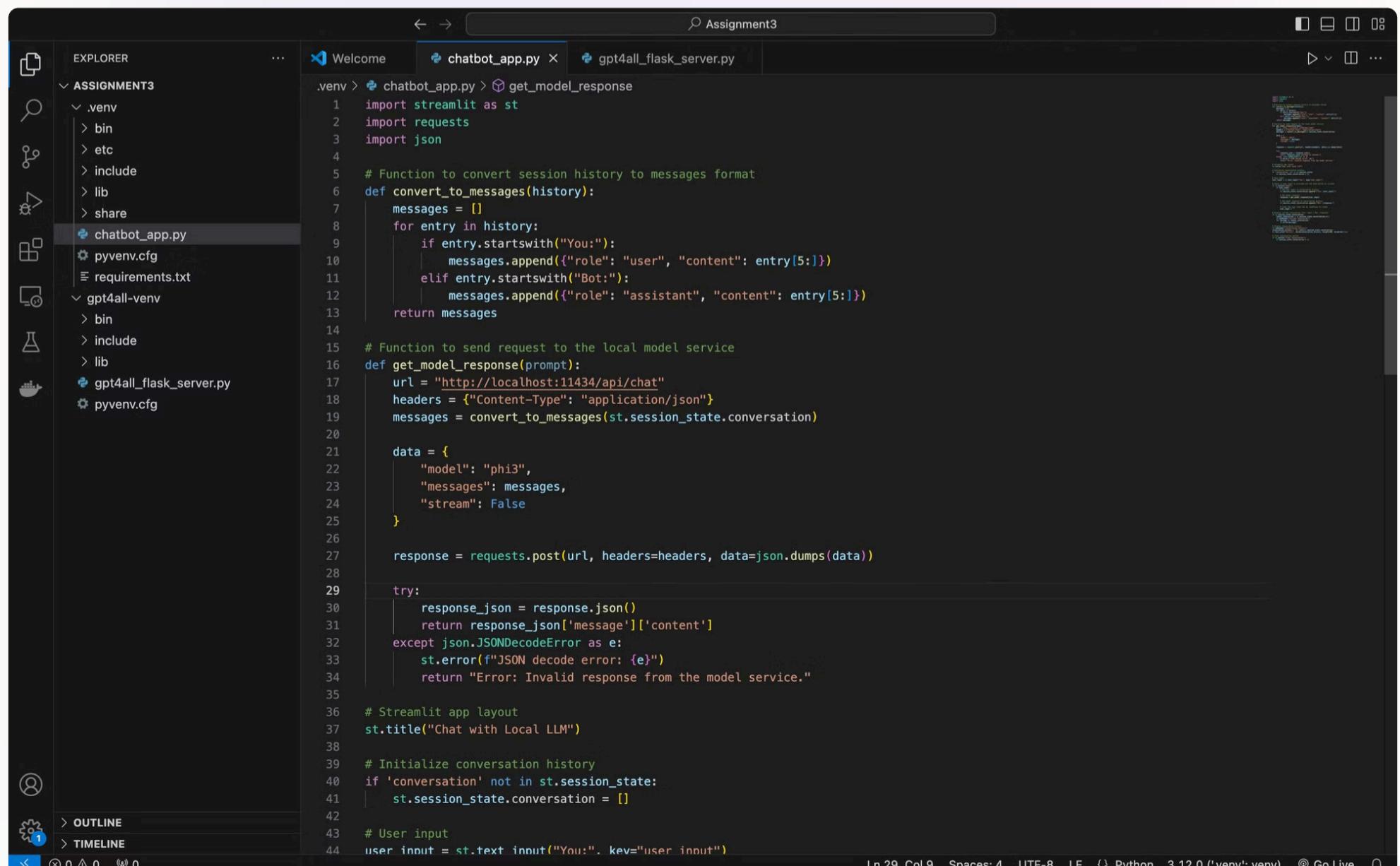
- Install Streamlit and Ollama3 using pip command



The screenshot shows the VS Code interface with the terminal tab selected. The output of the pip command is displayed:

```
anmolvalecha@Anmols-MacBook-Air Assignment3 % python3 -m venv .venv  
anmolvalecha@Anmols-MacBook-Air Assignment3 % source .venv/bin/activate  
(.venv) anmolvalecha@Anmols-MacBook-Air Assignment3 % python --version  
Python 3.12.0  
(.venv) anmolvalecha@Anmols-MacBook-Air Assignment3 % pip --version  
pip 23.2.1 from /Users/anmolvalecha/Cloud Backups/PromptEngg/Assignment3/.venv/lib/python3.12/site-packages/pip (python 3.12)  
(.venv) anmolvalecha@Anmols-MacBook-Air Assignment3 % pip install streamlit  
Collecting streamlit  
  Obtaining dependency information for streamlit from https://files.pythonhosted.org/packages/c6/51/f140402202af6ce1bf747243f66415c5eb2f43ba2e2ac419a7e855d2  
0673/streamlit-1.36.0-py3-none-any.whl.metadata  
    Downloading streamlit-1.36.0-py3-none-any.whl.metadata (8.5 kB)  
Collecting altair<6,>=4.0 (from streamlit)  
  Obtaining dependency information for altair<6,>=4.0 from https://files.pythonhosted.org/packages/46/30/2118537233fa72c1d91a81f5908a7e843a6601ccc68b76838eb  
c4951505f/altair-5.3.0-py3-none-any.whl.metadata  
    Using cached altair-5.3.0-py3-none-any.whl.metadata (9.2 kB)  
Collecting blinker<2,>=1.0.0 (from streamlit)  
  Obtaining dependency information for blinker<2,>=1.0.0 from https://files.pythonhosted.org/packages/bb/2a/10164ed1f31196a2f7f3799368a821765c62851ead0e630a  
b52b8e14b4d0/blinker-1.8.2-py3-none-any.whl.metadata  
    Using cached blinker-1.8.2-py3-none-any.whl.metadata (1.6 kB)  
Collecting cachetools<6,>=4.0 (from streamlit)  
  Obtaining dependency information for cachetools<6,>=4.0 from https://files.pythonhosted.org/packages/fb/2b/a64c2d25a37aeb929111413049fc5f8b9a4c1aef  
affaafe768d54/cachetools-5.3.3-py3-none-any.whl.metadata  
    Using cached cachetools-5.3.3-py3-none-any.whl.metadata (5.3 kB)  
Collecting click<9,>=7.0 (from streamlit)  
  Obtaining dependency information for click<9,>=7.0 from https://files.pythonhosted.org/packages/00/2e/d53fa4befbf2cfa713304affc7ca780ce4fc1fd8710527771b58  
311a3229/click-8.1.7-py3-none-any.whl.metadata  
    Using cached click-8.1.7-py3-none-any.whl.metadata (3.0 kB)  
Collecting numpy<3,>=1.20 (from streamlit)  
  Obtaining dependency information for numpy<3,>=1.20 from https://files.pythonhosted.org/packages/3f/ab/1dc9f176d3084a2546cf76eb213dc61586d015ef59b3b17947b  
0e40038af/numpy-2.0.0-cp312-cp312-macosx_14_0_arm64.whl.metadata  
    Using cached numpy-2.0.0-cp312-cp312-macosx_14_0_arm64.whl.metadata (60 kB)  
Collecting packaging<25,>=20 (from streamlit)  
  Obtaining dependency information for packaging<25,>=20 from https://files.pythonhosted.org/packages/08/aa/cc0199a5f0ad350994d660967a8efb233fe0416e4639146c  
089643407ce6/packaging-24.1-py3-none-any.whl.metadata  
    Using cached packaging-24.1-py3-none-any.whl.metadata (3.2 kB)  
Collecting pandas<3,>=1.3.0 (from streamlit)  
  Obtaining dependency information for pandas<3,>=1.3.0 from https://files.pythonhosted.org/packages/db/7c/9a60add21b96140e22465d9adf09832feade45235cd22f4cb  
1668a25e443/pandas-2.2.2-cp312-cp312-macosx_11_0_arm64.whl.metadata  
    Using cached pandas-2.2.2-cp312-cp312-macosx_11_0_arm64.whl.metadata (19 kB)  
Collecting pillow<11,>=7.1.0 (from streamlit)  
  Obtaining dependency information for pillow<11,>=7.1.0 from https://files.pythonhosted.org/packages/5e/77/4cf407e7b033b4d8e5fcaac295b6e159cf1c70fa105d769f  
01ea2e1e5eca/pillow-10.3.0-cp312-cp312-macosx_11_0_arm64.whl.metadata  
    Using cached pillow-10.3.0-cp312-cp312-macosx_11_0_arm64.whl.metadata (9.2 kB)  
Collecting protobuf<6,>=3.20 (from streamlit)  
  Obtaining dependency information for protobuf<6,>=3.20 from https://files.pythonhosted.org/packages/1b/14/737d9d432893e75ae1fdaca96e97e8a9b0250c3769a459f4  
4cb419eeaa5e/protobuf-5.27.1-cp38-abi3-macosx_10_9_universal2.whl.metadata  
    Downloading protobuf-5.27.1-cp38-abi3-macosx_10_9_universal2.whl.metadata (592 bytes)
```

- Create a Python Script



The screenshot shows the VS Code interface with the code editor tab selected. The file 'chatbot\_app.py' is open, displaying the following Python code:

```
import streamlit as st  
import requests  
import json  
  
# Function to convert session history to messages format  
def convert_to_messages(history):  
    messages = []  
    for entry in history:  
        if entry.startswith("You:"):  
            messages.append({"role": "user", "content": entry[5:]})  
        elif entry.startswith("Bot:"):  
            messages.append({"role": "assistant", "content": entry[5:]})  
    return messages  
  
# Function to send request to the local model service  
def get_model_response(prompt):  
    url = "http://localhost:11434/api/chat"  
    headers = {"Content-Type": "application/json"}  
    messages = convert_to_messages(st.session_state.conversation)  
  
    data = {  
        "model": "phi3",  
        "messages": messages,  
        "stream": False  
    }  
  
    response = requests.post(url, headers=headers, data=json.dumps(data))  
  
    try:  
        response_json = response.json()  
        return response_json['message']['content']  
    except json.JSONDecodeError as e:  
        st.error(f"JSON decode error: {e}")  
        return "Error: Invalid response from the model service."  
  
    # Streamlit app layout  
    st.title("Chat with Local LLM")  
  
    # Initialize conversation history  
    if 'conversation' not in st.session_state:  
        st.session_state.conversation = []  
  
    # User input  
    user_input = st.text_input("You:", key="user_input")
```



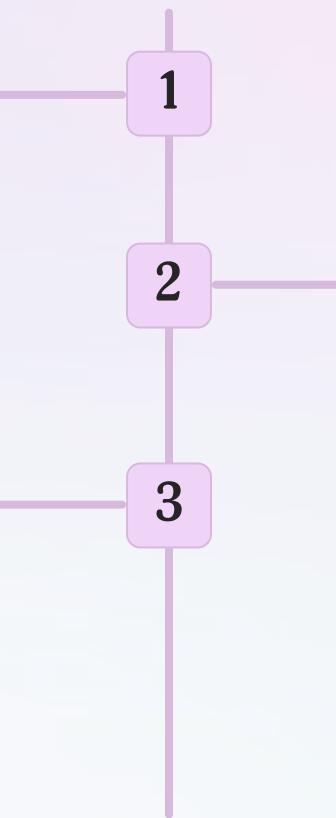
# Interacting with Ollama3

## Run the Python Script

Write a Python script that integrates with the Ollama phi3 language model. This will allow you to interact with the model programmatically.

## Explore the Interface

Use the Ollam3 user interface to select the desired model, such as phi3, and begin a new chat session. Observe how the model responds to your prompts.

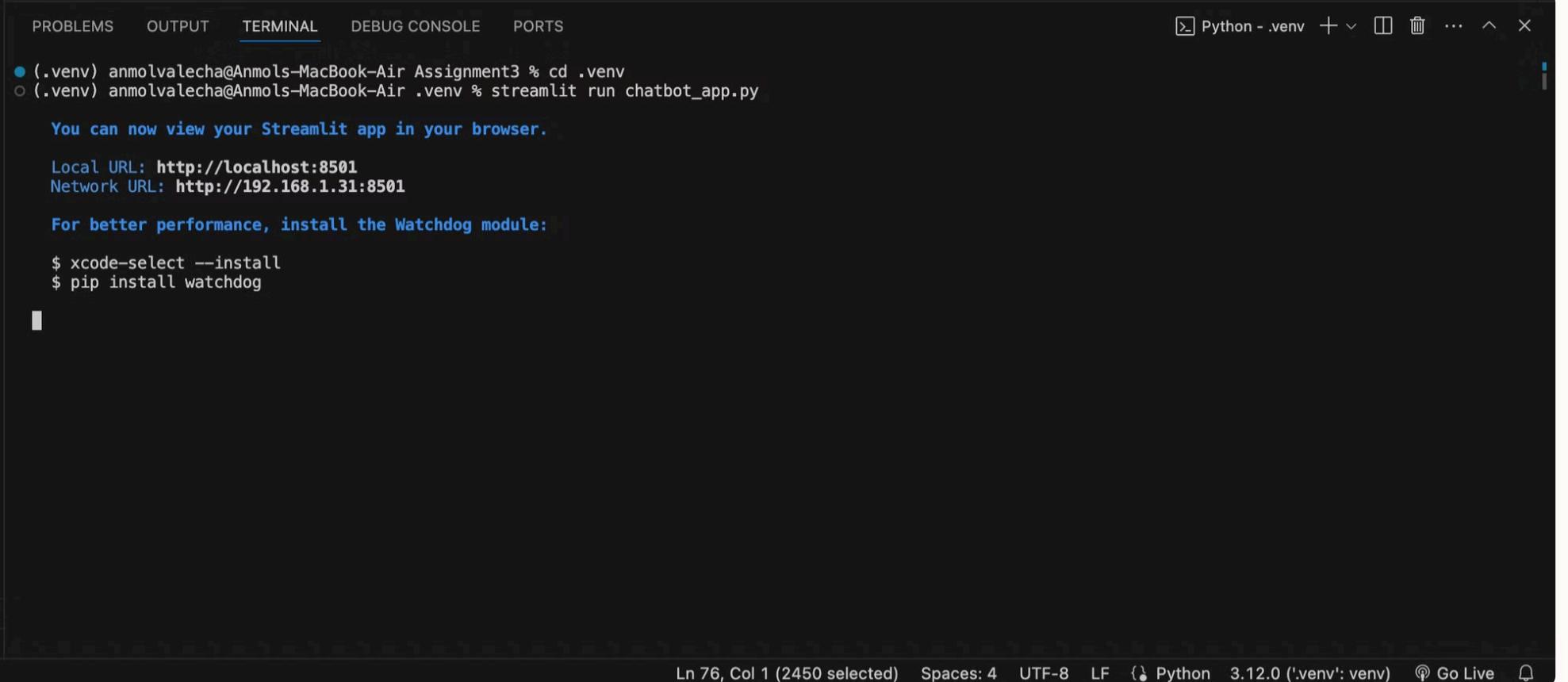


## Leverage API Interactions

For programmatic integration, you can interact with Ollama3 using the curl command to send prompts and receive responses directly on terminal.

# Interaction with Ollama3 via Terminal using Curl Command

- Run the python script



The screenshot shows a terminal window with the following content:

```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE PORTS
Python - .venv + × ... ^ ×

● (.venv) anmolvalecha@Anmols-MacBook-Air Assignment3 % cd .venv
○ (.venv) anmolvalecha@Anmols-MacBook-Air .venv % streamlit run chatbot_app.py

You can now view your Streamlit app in your browser.

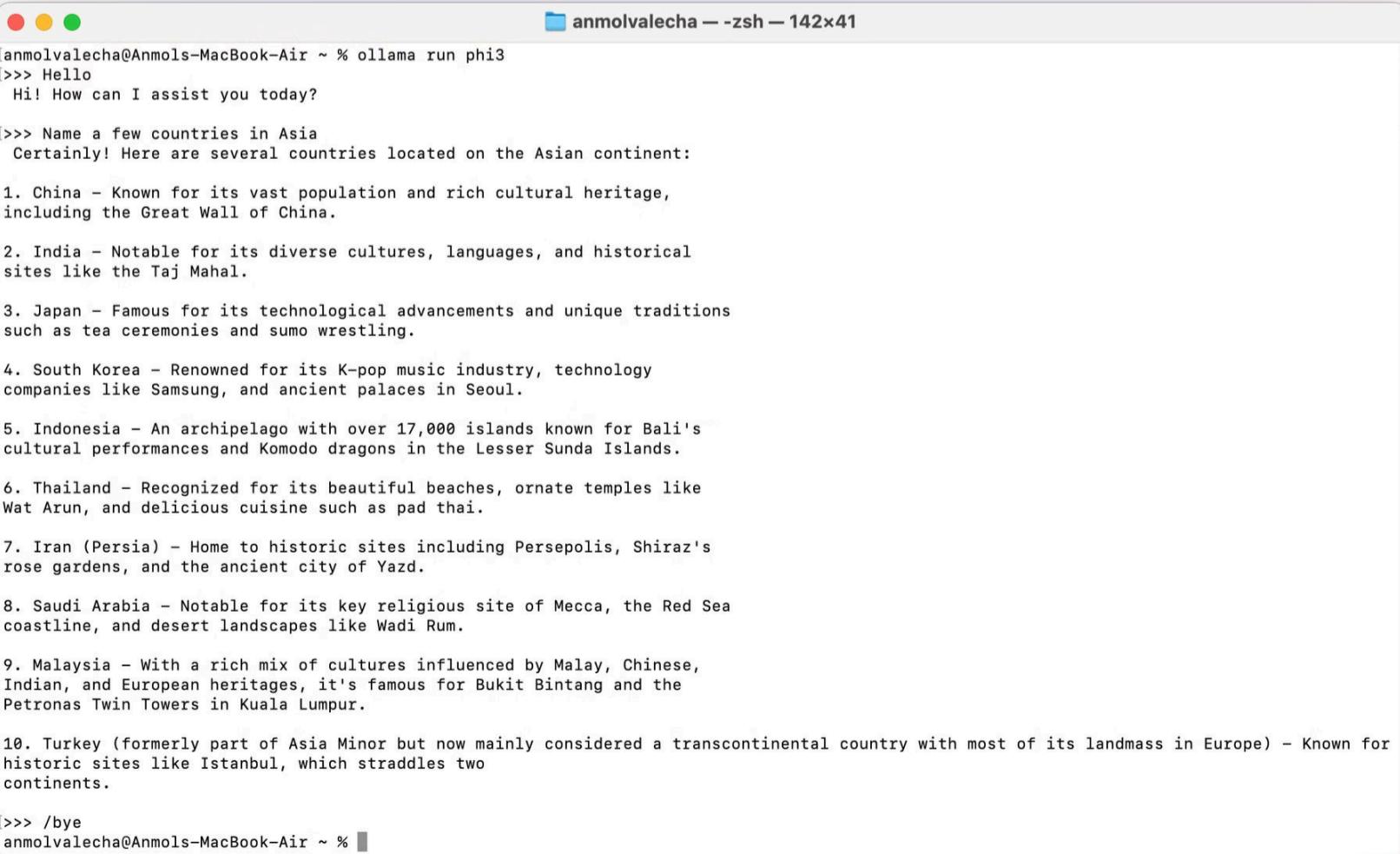
Local URL: http://localhost:8501
Network URL: http://192.168.1.31:8501

For better performance, install the Watchdog module:

$ xcode-select --install
$ pip install watchdog
```

Ln 76, Col 1 (2450 selected) Spaces: 4 UTF-8 LF { Python 3.12.0 ('.venv': venv) ⚡ Go Live 🔍

- Launch Terminal and run command "ollama run phi3"



The screenshot shows a terminal window titled "anmolvalecha — zsh — 142x41" with the following interaction:

```
[anmolvalecha@Anmols-MacBook-Air ~ % ollama run phi3
>>> Hello
    Hi! How can I assist you today?

>>> Name a few countries in Asia
    Certainly! Here are several countries located on the Asian continent:
    1. China – Known for its vast population and rich cultural heritage,
       including the Great Wall of China.
    2. India – Notable for its diverse cultures, languages, and historical
       sites like the Taj Mahal.
    3. Japan – Famous for its technological advancements and unique traditions
       such as tea ceremonies and sumo wrestling.
    4. South Korea – Renowned for its K-pop music industry, technology
       companies like Samsung, and ancient palaces in Seoul.
    5. Indonesia – An archipelago with over 17,000 islands known for Bali's
       cultural performances and Komodo dragons in the Lesser Sunda Islands.
    6. Thailand – Recognized for its beautiful beaches, ornate temples like
       Wat Arun, and delicious cuisine such as pad thai.
    7. Iran (Persia) – Home to historic sites including Persepolis, Shiraz's
       rose gardens, and the ancient city of Yazd.
    8. Saudi Arabia – Notable for its key religious site of Mecca, the Red Sea
       coastline, and desert landscapes like Wadi Rum.
    9. Malaysia – With a rich mix of cultures influenced by Malay, Chinese,
       Indian, and European heritages, it's famous for Bukit Bintang and the
       Petronas Twin Towers in Kuala Lumpur.
    10. Turkey (formerly part of Asia Minor but now mainly considered a transcontinental country with most of its landmass in Europe) – Known for
       historic sites like Istanbul, which straddles two
       continents.

>>> /bye
anmolvalecha@Anmols-MacBook-Air ~ %
```

# Ollama3 Interaction via Streamlit

- Output of python script using Streamlit

You: Name a few countries in Asia

Send

**Current Interaction**

You: Name a few countries in Asia

Bot: Certainly! Here are several countries located within the continent of Asia:

1. China - Known for its vast population and rich cultural history.
2. India - Famous for its diverse cultures, languages, and ancient heritage.
3. Japan - Renowned for its technological advancements and unique blend of traditional and modern culture.
4. South Korea - Notable for its economic development and pop culture influence globally.
5. Indonesia - An archipelagic nation with diverse landscapes, including beaches, mountains, and volcanoes.
6. Thailand - Known for its beautiful beaches, royal palaces, and ornate Buddhist temples.
7. Saudi Arabia - The home of the two holiest cities in Islam: Mecca and Medina.
8. Iran (Persia) - Famous for its rich history and cultural contributions to art and architecture.
9. Turkey - Located at the junction of Europe and Asia, it is a country with deep historical roots.
10. Malaysia - A multicultural country that boasts lush rainforests, beautiful beaches, and vibrant cities.

**Conversation History**

History

You: Hello

Bot: Hi! How can I assist you today? Whether it's answering a question, solving a problem, or

- Conversation History

culture.

4. South Korea - Notable for its economic development and pop culture influence globally.

5. Indonesia - An archipelagic nation with diverse landscapes, including beaches, mountains, and volcanoes.

6. Thailand - Known for its beautiful beaches, royal palaces, and ornate Buddhist temples.

7. Saudi Arabia - The home of the two holiest cities in Islam: Mecca and Medina.

8. Iran (Persia) - Famous for its rich history and cultural contributions to art and architecture.

9. Turkey - Located at the junction of Europe and Asia, it is a country with deep historical roots.

10. Malaysia - A multicultural country that boasts lush rainforests, beautiful beaches, and vibrant cities.

**Conversation History**

History

You: Hello

Bot: Hi! How can I assist you today? Whether it's answering a question, solving a problem, or providing information on a specific topic, feel free to ask.

You: Name a few countries in Asia

Bot: Certainly! Here are several countries located within the continent of Asia:

1. China - Known for its vast population and rich cultural history.
2. India - Famous for its diverse cultures, languages, and ancient heritage.
3. Japan - Renowned for its technological advancements and unique blend of traditional and modern culture.

Clear Conversation

# GPT4all Model Setup

## Installation via pip in Virtual Environment

Activate virtual environment and install GPT4all:

```
python -m venv .venv  
source .venv/bin/activate  
pip install gpt4all
```

1

2

3

## Installing Flask for Web Interface

Install Flask within the same virtual environment:

```
pip install flask
```

## Downloading GPT4all UI from Official Website

Access the GPT4all user interface from <https://gpt4all.io/>. Follow the installation instructions provided on the website.

# Setup of Flask and GPT4All

- Setup a virtual environment

```
anmolvalecha@Anmols-MacBook-Air Assignment3 % python3 -m venv gpt4all-venv  
anmolvalecha@Anmols-MacBook-Air Assignment3 % source gpt4all-venv/bin/activate
```

- Install Flask and GPT4All using pip command

```
Found existing installation: pip 23.2.1  
Uninstalling pip-23.2.1:  
  Successfully uninstalled pip-23.2.1  
Successfully installed pip-24.1  
(gpt4all-venv) anmolvalecha@Anmols-MacBook-Air Assignment3 % pip install gpt4all flask flask-cors  
Collecting gpt4all  
  Using cached gpt4all-2.7.0-py3-none-macosx_10_15_universal2.whl.metadata (4.7 kB)  
Collecting flask  
  Using cached flask-3.0.3-py3-none-any.whl.metadata (3.2 kB)  
Collecting flask-cors  
  Downloading Flask_Cors-4.0.1-py2.py3-none-any.whl.metadata (5.5 kB)  
Collecting requests (from gpt4all)  
  Using cached requests-2.32.3-py3-none-any.whl.metadata (4.6 kB)  
Collecting tqdm (from gpt4all)  
  Using cached tqdm-4.66.4-py3-none-any.whl.metadata (57 kB)  
Collecting Werkzeug>=3.0.0 (from flask)  
  Using cached werkzeug-3.0.3-py3-none-any.whl.metadata (3.7 kB)  
Collecting Jinja2>=3.1.2 (from flask)  
  Using cached jinja2-3.1.4-py3-none-any.whl.metadata (2.6 kB)  
Collecting itsdangerous>=2.1.2 (from flask)  
  Using cached itsdangerous-2.2.0-py3-none-any.whl.metadata (1.9 kB)  
Collecting click>=8.1.3 (from flask)  
  Downloading click-8.1.7-py3-none-any.whl.metadata (3.0 kB)  
Collecting blinker>=1.6.2 (from flask)  
  Downloading blinker-1.8.2-py3-none-any.whl.metadata (1.6 kB)  
Collecting MarkupSafe==2.0 (from Jinja2>=3.1.2->flask)  
  Using cached MarkupSafe-2.1.5-cp312-cp312-macosx_10_9_universal2.whl.metadata (3.0 kB)  
Collecting charset-normalizer<4,>=2 (from requests->gpt4all)  
  Using cached charset_normalizer-3.3.2-cp312-cp312-macosx_11_0_arm64.whl.metadata (33 kB)  
Collecting idna>4,>=2.5 (from requests->gpt4all)  
  Using cached idna-3.7-py3-none-any.whl.metadata (9.9 kB)  
Collecting urllib3<3,>=1.21.1 (from requests->gpt4all)  
  Using cached urllib3-2.2.2-py3-none-any.whl.metadata (6.4 kB)  
Collecting certifi>=2017.4.17 (from requests->gpt4all)  
  Using cached certifi-2024.6.2-py3-none-any.whl.metadata (2.2 kB)  
Using cached gpt4all-2.7.0-py3-none-macosx_10_15_universal2.whl (7.6 MB)  
Using cached flask-3.0.3-py3-none-any.whl (101 kB)  
Downloading Flask_Cors-4.0.1-py2.py3-none-any.whl (14 kB)  
Downloading blinker-1.8.2-py3-none-any.whl (9.5 kB)  
Downloading click-8.1.7-py3-none-any.whl (97 kB)  
  97.9/97.9 kB 2.8 MB/s eta 0:00:00  
Using cached itsdangerous-2.2.0-py3-none-any.whl (16 kB)  
Using cached jinja2-3.1.4-py3-none-any.whl (133 kB)  
Using cached werkzeug-3.0.3-py3-none-any.whl (227 kB)  
Using cached requests-2.32.3-py3-none-any.whl (64 kB)  
Using cached tqdm-4.66.4-py3-none-any.whl (78 kB)  
Using cached certifi-2024.6.2-py3-none-any.whl (164 kB)  
Using cached charset_normalizer-3.3.2-cp312-cp312-macosx_11_0_arm64.whl (119 kB)  
Using cached idna-3.7-py3-none-any.whl (66 kB)  
Using cached MarkupSafe-2.1.5-cp312-cp312-macosx_10_9_universal2.whl (18 kB)  
Using cached urllib3-2.2.2-py3-none-any.whl (121 kB)  
Installing collected packages: urllib3, tqdm, MarkupSafe, itsdangerous, idna, click, charset-normalizer, certifi, blinker, Werkzeug, requests, Jinja2, gpt4all, flask, flask-cors  
Successfully installed Jinja2-3.1.4 MarkupSafe-2.1.5 Werkzeug-3.0.3 blinker-1.8.2 certifi-2024.6.2 charset-normalizer-3.3.2 click-8.1.7 flask-3.0.3 flask-cors-4.0.1 gpt4all-2.7.0 idna-3.7 itsdangerous-2.2.0 requests-2.32.3 tqdm-4.66.4 urllib3-2.2.2  
(gpt4all-venv) anmolvalecha@Anmols-MacBook-Air Assignment3 %
```

- Create a Python Script

```
gpt4all-venv > gpt4all_flask_server.py > ...  
1  from gpt4all import GPT4All  
2  from flask import Flask, request, jsonify  
3  
4  app = Flask(__name__)  
5  model = GPT4All("orca-mini-3b-gguf2-q4_0.gguf")  
6  
7  @app.route('/api/generate', methods=['POST'])  
8  def generate():  
9      try:  
10          data = request.json  
11          prompt = data.get('prompt')  
12          response = model.generate(prompt)  
13          return jsonify({"response": response})  
14      except Exception as e:  
15          return jsonify({"error": str(e)}), 500  
16  
17  if __name__ == '__main':  
18      app.run(port=5000)
```



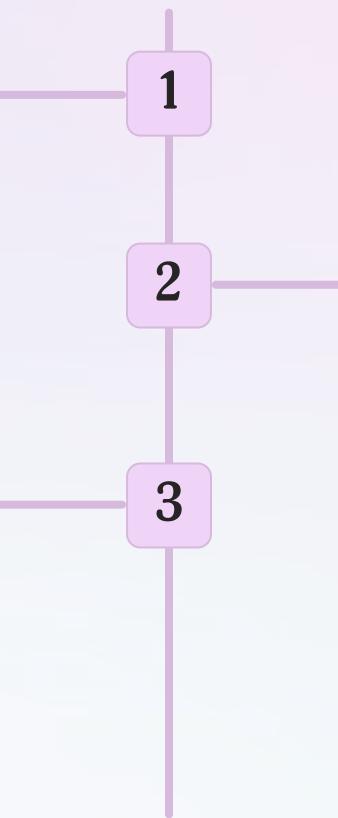
# Interacting with GPT4all

## Run the Python Script

Write a Python script that integrates with the GPT4all language model. This will allow you to interact with the model programmatically.

## Explore the Interface

Use the GPT4all user interface to select the desired model, such as phi3, and begin a new chat session. Observe how the model responds to your prompts.



## Leverage API Interactions

For programmatic integration, you can interact with GPT4all using the curl command to send prompts and receive responses directly on terminal.

# Interaction with GPT4all via Terminal using Curl Command

- Run the python script

```
def generate():
    # Return error message if an exception occurs
    return jsonify({"error": str(e)}), 500

# Endpoint for testing
@app.route('/', methods=['GET'])
def hello():
    return "Hello, World!"

# Run the Flask application
if __name__ == '__main__':
    app.run(debug=True, port=5050)
```

```
anmolvalecha@Anmols-MacBook-Air Assignment3 % source gpt4all-venv/bin/activate
(gpt4all-venv) anmolvalecha@Anmols-MacBook-Air Assignment3 % cd gpt4all-venv
(gpt4all-venv) anmolvalecha@Anmols-MacBook-Air gpt4all-venv % python3 gpt4all_flask_server.py
* Serving Flask app 'gpt4all_flask_server'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5050
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 641-911-403
```

- Launch Terminal and Run a curl command with the prompt

```
(gpt4all-venv) anmolvalecha@Anmols-MacBook-Air Assignment3 % curl -X POST http://localhost:5050/api/generate \
-H "Content-Type: application/json" \
-d '{"prompt": "Hello"}'

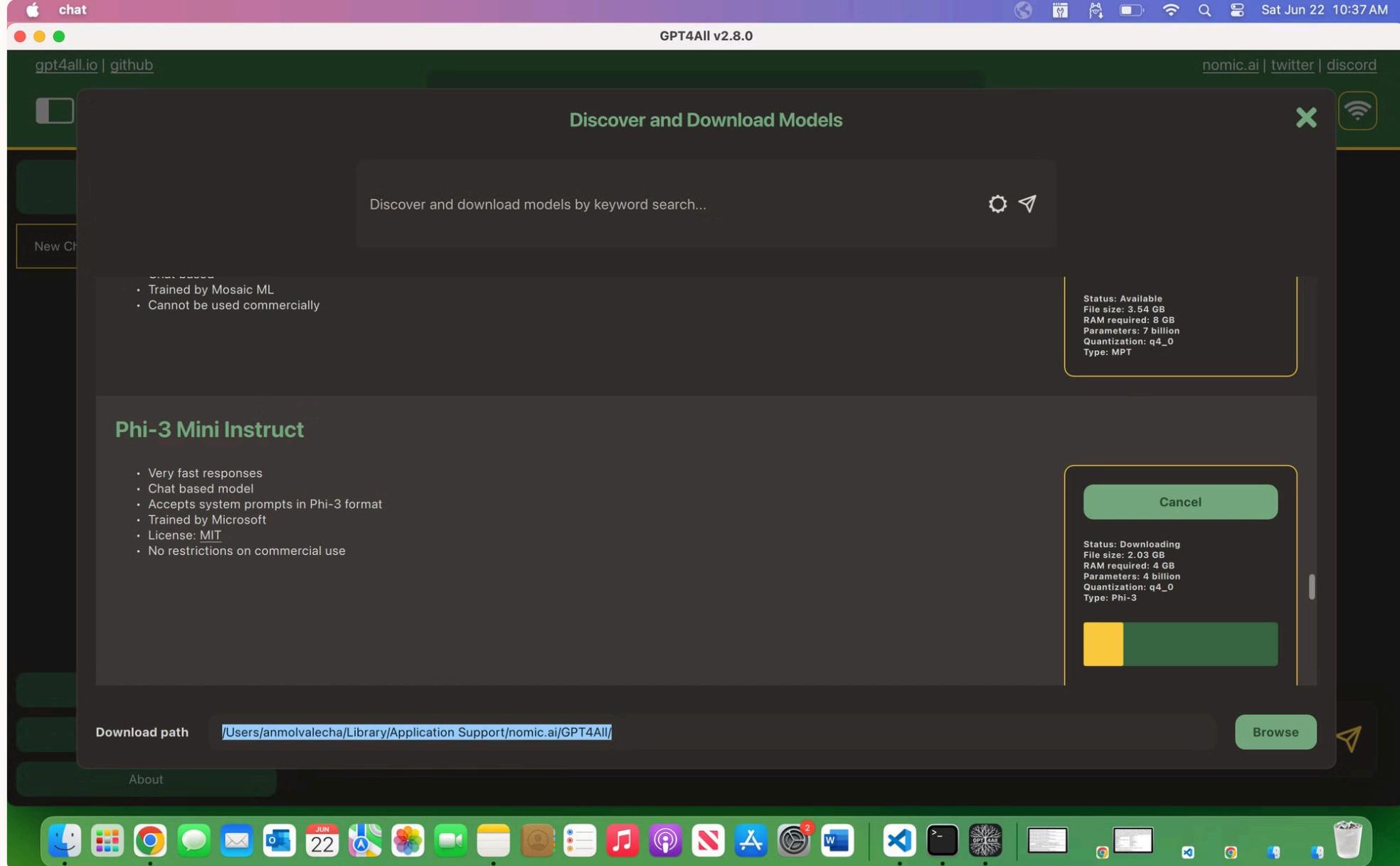
{
  "response": " Hello! How may I assist you today?"
}
(gpt4all-venv) anmolvalecha@Anmols-MacBook-Air Assignment3 % curl -X POST http://localhost:5050/api/generate \
-H "Content-Type: application/json" \
-d '{"prompt": "Name a few countries in Asia"}'

{
  "response": " Sure, here are some countries in Asia:\n\n1. China\n2. India\n3. Japan\n4. South Korea\n5. Thailand\n6. Vietnam\n7. Indonesia\n8. Malaysia\n9. Philippines\n10. Singapore"
}
(gpt4all-venv) anmolvalecha@Anmols-MacBook-Air Assignment3 % curl -X POST http://localhost:5050/api/generate \
-H "Content-Type: application/json" \
-d '{"prompt": "Tell me something more about India"}'

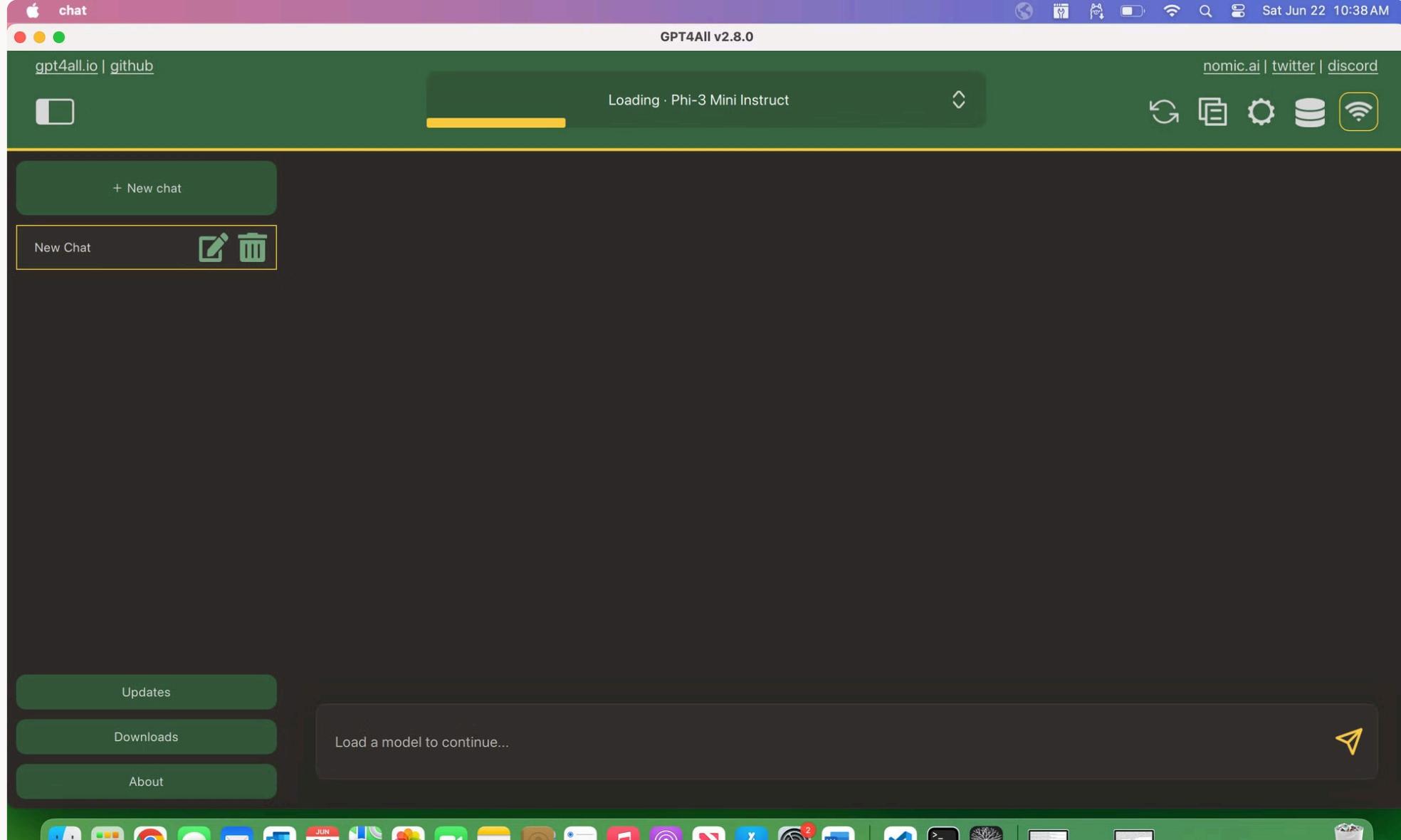
{
  "response": " India is a country located in South Asia and is known for its diverse culture, rich history, and vibrant economy. It is the seventh-largest country in the world by both area and population. The official language of India is Hindi, but there are many other languages spoken throughout the country, including English, Bengali, Tamil, Telugu, Marathi, and more. India is known for its delicious cuisine, which includes dishes such as butter chicken, biryani, and naan. It is also famous for its traditional textiles, jewelry, and art. India has a rich history that dates back thousands of years, with many ancient ruins and monuments still visible today. Some of the most well-known landmarks in India include the Taj Mahal, Agra Fort, and the Red Fort in Delhi. India is also known for its diverse culture, which includes traditions such as yoga, meditation, and spiritual practices like Hinduism, Buddhism, and Sikhism"
}
(gpt4all-venv) anmolvalecha@Anmols-MacBook-Air Assignment3 %
```

# GPT4all Interaction via UI

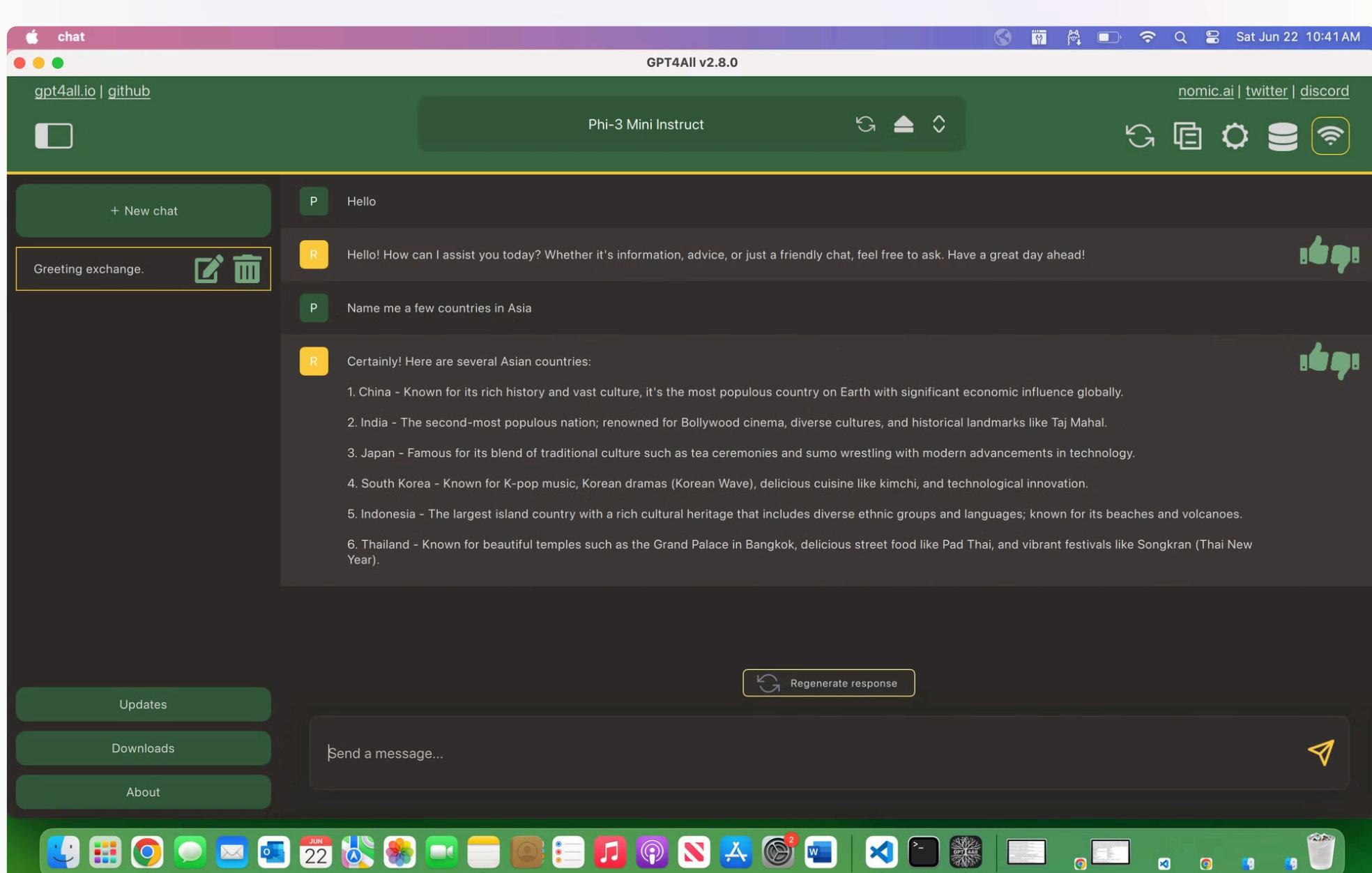
- Open tool and select desired model



- Open a new chat



- Conversation with LLM Model



# References

## LLama3:

- **GitHub Repository:**
  - [LLama3 GitHub](#)
- **Installation via pip:**
  - pip install streamlit
- **User Interface:**
- Download LLama phi3 UI: [LLama3 UI](#)

## GPT4all:

- **GitHub Repository:**
  - [GPT4all GitHub](#)
- **Installation via pip:**
  - pip install gpt4all
  - pip install flask
- **User Interface:**
  - Download GPT4all UI: [GPT4all UI](#)