

# ChatBot development using OpenAI API and Streamlit

By: Vaishnavi Patil

## 1. Set Up Virtual Environment

First, ensure you have virtualenv installed. If not, you can install it by running:

```
vaishnavi@DESKTOP-9V8KJG2:/mnt/c/Users/Mohit/Desktop/Gen AI/Week3/chatbot_streamlit$ pip install virtualenv
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: virtualenv in /usr/lib/python3/dist-packages (20.13.0+ds)
Requirement already satisfied: distlib<1,>=0.3.1 in /usr/lib/python3/dist-packages (from virtualenv) (0.3.4)
Requirement already satisfied: six<2,>=1.9.0 in /usr/lib/python3/dist-packages (from virtualenv) (1.16.0)
Requirement already satisfied: platformdirs<3,>=2 in /usr/lib/python3/dist-packages (from virtualenv) (2.5.1)
Requirement already satisfied: filelock<4,>=3.2 in /usr/lib/python3/dist-packages (from virtualenv) (3.6.0)
vaishnavi@DESKTOP-9V8KJG2:/mnt/c/Users/Mohit/Desktop/Gen AI/Week3/chatbot_streamlit$
```

### a. Create a Virtual Environment

Navigate to your project directory and create a virtual environment:

```
vaishnavi@DESKTOP-9V8KJG2:/mnt/c/Users/Mohit/Desktop/Gen AI/Week3/chatbot_streamlit$ pwd
/mnt/c/Users/Mohit/Desktop/Gen AI/Week3/chatbot_streamlit
vaishnavi@DESKTOP-9V8KJG2:/mnt/c/Users/Mohit/Desktop/Gen AI/Week3/chatbot_streamlit$ virtualenv venv
created virtual environment CPython3.10.12.final.0-64 in 3186ms
  creator CPython3Posix(dest=/mnt/c/Users/Mohit/Desktop/Gen AI/Week3/chatbot_streamlit/venv, clear=False,
, no_vcs_ignore=False, global=False)
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy, app_data_dir
=/home/vaishnavi/.local/share/virtualenv)
    added seed packages: pip==22.0.2, setuptools==59.6.0, wheel==0.37.1
  activators BashActivator,CShellActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActi
vator
vaishnavi@DESKTOP-9V8KJG2:/mnt/c/Users/Mohit/Desktop/Gen AI/Week3/chatbot_streamlit$
```

This will create a venv folder inside your project directory.

### b. Activate the Virtual Environment

I am using Ubuntu so run **source venv/bin/activate**

```
vaishnavi@DESKTOP-9V8KJG2:/mnt/c/Users/Mohit/Desktop/Gen AI/Week3/chatbot_streamlit$ source venv/bin/act
ivate
(venv) vaishnavi@DESKTOP-9V8KJG2:/mnt/c/Users/Mohit/Desktop/Gen AI/Week3/chatbot_streamlit$
```

You should now see (venv) in your terminal, indicating that the virtual environment is active.

### c. Install Dependencies

Once the virtual environment is activated, install the dependencies from the requirements.txt file:

```
(venv) vaishnavi@DESKTOP-9V8KJG2:/mnt/c/Users/Mohit/Desktop/Gen AI/Week3/chatbot_streamlit$ pip install -r requirements.txt
Collecting streamlit==1.22.0
  Downloading streamlit-1.22.0-py2.py3-none-any.whl (8.9 MB)
    8.9/8.9 MB 24.7 MB/s eta 0:00:00
Collecting openai==0.27.0
  Downloading openai-0.27.0-py3-none-any.whl (70 kB)
    70.1/70.1 KB 7.1 MB/s eta 0:00:00
Collecting python-dotenv==1.0.0
  Downloading python_dotenv-1.0.0-py3-none-any.whl (19 kB)
Collecting python-dateutil
  Using cached python_dateutil-2.9.0.post0-py2.py3-none-any.whl (229 kB)
Collecting requests>=2.4
  Using cached requests-2.32.3-py3-none-any.whl (64 kB)
Collecting click>=7.0
  Using cached click-8.1.7-py3-none-any.whl (97 kB)
Collecting blinker>=1.0.0
  Downloading blinker-1.8.2-py3-none-any.whl (9.5 kB)
Collecting pandas<3,>=0.25
  Using cached pandas-2.2.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (13.1 MB)
Collecting importlib-metadata>=1.4
  Downloading importlib_metadata-8.5.0-py3-none-any.whl (26 kB)
Collecting tornado>=6.0.3
  Using cached tornado-6.4.1-cp38-abi3-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux2014_x86_64.whl (436 kB)
Collecting tzlocal>=1.1
  Downloading tzlocal-5.2-py3-none-any.whl (17 kB)
Collecting altair<5,>=3.2.0
  Downloading altair-4.2.2-py3-none-any.whl (813 kB)
    813.6/813.6 KB 16.5 MB/s eta 0:00:00
Collecting numpy
  Downloading numpy-2.1.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (16.3 MB)
    16.3/16.3 MB 27.2 MB/s eta 0:00:00
    208.9/208.9 KB 12.9 MB/s eta 0:00:00
Installing collected packages: pytz, zipp, watchdog, validators, urllib3, tzlocal, tzdata, typing-extensions, tqdm, tornado, toolz, toml, tenacity, smmap, six, rpds-py, python-dotenv, pympler, pygments, protobuf, propcache, pillow, packaging, numpy, mdurl, MarkupSafe, idna, frozenlist, entrypoints, click, charset-normalizer, certifi, cachetools, blinker, attrs, asynctimeout, aiohappyeyeballs, requests, referencing, python-dateutil, pyarrow, multidict, markdown-it-py, Jinja2, importlib-metadata, gitdb, aiosignal, yarl, rich, pydeck, pandas, jsonschema-specifications, gitpython, jsonschema, aiohttp, openai, altair, streamlit
Successfully installed MarkupSafe-3.0.1 aiohappyeyeballs-2.4.3 aiohttp-3.10.9 aiosignal-1.3.1 altair-4.2.2 async-timeout-4.0.3 attrs-24.2.0 blinker-1.8.2 cachetools-5.5.0 certifi-2024.8.30 charset-normalizer-3.3.2 click-8.1.7 entrypoints-0.4 frozenlist-1.4.1 gitdb-4.0.11 gitpython-3.1.43 idna-3.10 importlib-metadata-8.5.0 Jinja2-3.1.4 jsonschema-4.23.0 jsonschema-specifications-2024.10.1 markdown-it-py-3.0.0 mdurl-0.1.2 multidict-6.1.0 numpy-2.1.2 openai-0.27.0 packaging-24.1 pandas-2.2.3 pillow-10.4.0 propcache-0.2.0 protobuf-3.20.3 pyarrow-17.0.0 pydeck-0.9.1 pygments-2.18.0 pympler-1.1 python-dateutil-2.9.0.post0 python-dotenv-1.0.0 pytz-2024.2 referencing-0.35.1 requests-2.32.3 rich-13.9.2 rpds-py-0.20.0 six-1.16.0 smmap-5.0.1 streamlit-1.22.0 tenacity-8.5.0 toml-0.10.2 toolz-1.0.2 tornado-6.4.1 tqdm-4.66.5 typing-extensions-4.12.2 tzdata-2024.2 tzlocal-5.2 urllib3-2.2.3 validators-0.34.0 watchdog-5.0.3 yarl-1.14.0 zipp-3.20.2
(venv) vaishnavi@DESKTOP-9V8KJG2:/mnt/c/Users/Mohit/Desktop/Gen AI/Week3/chatbot_streamlit$
```

d. Set up environment variables:

```
# Define your OpenAI API key
OPENAI_API_KEY = "sk-..."
# Replace with your actual API key
```

e. Deactivate the virtual environment (when done) using **deactivate**

## 2. Main Application Code

After all the requirements are installed and virtual environment is set, let's move with the creation of chatbot using gpt-4o-mini model and streamlit.

```
◆ chatbot.py X
chatbot streamlit > ◆ chatbot.py > ...
1  import streamlit as st
2  import openai, os
3  from dotenv import load_dotenv
4
5  # Load environment variables for OpenAI API key
6  load_dotenv()
7  openai.api_key = os.getenv("OPENAI_API_KEY") # Set the API key
8
9  # Title of the chatbot
10 st.title("My GPT-4o-mini Chatbot 🤖")
11
12 # CSS for full-page shaded blue gradient background
13 st.markdown(
14     """
15     <style>
16     /* Apply the gradient to the whole page */
17     html, body, .stApp {
18         height: 100%;
19         background: linear-gradient(135deg, #804e92, #000428); /* Shaded blue gradient */
20         color: white; /* Text color for readability */
21     }
22     </style>
23     """,
24     unsafe_allow_html=True
25 )
26
27 # Initialize messages in the session state
28 if "messages" not in st.session_state:
29     st.session_state.messages = []
30
31 # Display messages
32 for message in st.session_state["messages"]:
33     with st.chat_message(message["role"]):
34         st.markdown(message["content"])
35
36 # Handle user input and OpenAI response
37 if user_prompt := st.chat_input("Your Prompt:"):
38     st.session_state.messages.append({"role": "user", "content": user_prompt})
39
40     # Display user message
41     with st.chat_message("user"):
42         st.markdown(user_prompt)
43
44     # Assistant response
45     with st.chat_message("assistant"):
46         chatbot_msg = st.empty()
47         full_response = ""
48         stream = openai.chat.completions.create(
49             model="gpt-4o-mini",
50             messages=[
51                 {"role": msg["role"], "content": msg["content"]}
52                 for msg in st.session_state["messages"]
53             ],
54             stream=True,
55         )
56
57         # Stream the response
58         for chunk in stream:
59             token = chunk.choices[0].delta.content
60             if token is not None:
61                 full_response = full_response + token
62                 chatbot_msg.markdown(full_response)
63
64         chatbot_msg.markdown(full_response)
65
66 # Store assistant's response in session
67 st.session_state.messages.append({"role": "assistant", "content": full_response})
68
```

Running the Application

To run the Streamlit app, use the command: **streamlit run chatbot.py**



```

vaishnavi@DESKTOP-9V8KJG2:/mnt/c/Users/Mohit/Desktop/Gen AI/Week3/chatbot_streamlit$ source venv/bin/activate
(vaishnavi@DESKTOP-9V8KJG2:/mnt/c/Users/Mohit/Desktop/Gen AI/Week3/chatbot_streamlit$ streamlit run chatbot.py
/mnt/c/Users/Mohit/Desktop/Gen AI/Week3/chatbot_streamlit/venv/lib/python3.10/site-packages/numpy/_core/getlimits.py:548: UserWarning: Signature b'\x00\xd0\xcc\xcc\xcc\xcc\xcc\xcc\xfb\xbf\x00\x00\x00\x00\x00\x00' for <class 'numpy.longdouble'> does not match any known type: falling back to type probe function.
This warnings indicates broken support for the dtype!
    machar = _get_machar(dtype)

Collecting usage statistics. To deactivate, set browser.gatherUsageStats to False.

You can now view your Streamlit app in your browser.

Network URL: http://192.168.1.76:8501
External URL: http://107.193.140.97:8501

```

Then follow the link shown



# My GPT-4o-mini Chatbot 🤖

How can I help you?

give me a function to find the even numbers

give me a function to find the even numbers

Certainly! Below is a simple function in Python that takes a list of numbers as input and returns a list of even numbers from that list.

```
def find_even_numbers(numbers):  
    """Return a list of even numbers from the given list."""  
    even_numbers = [num for num in numbers if num % 2 == 0]  
    return even_numbers  
  
# Example usage:  
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
even_numbers = find_even_numbers(numbers)  
print(even_numbers) # Output: [2, 4, 6, 8, 10]
```

You can call the `find_even_numbers` function with any list of integers, and it will return a new list containing only the even numbers.