Python Chatbot Final Project

Overview

In this final project, you will create a personality-driven chatbot that can remember conversations and interact with users in different styles. Your chatbot will use a provided AI function while maintaining its own unique personality. This project brings together the Python concepts we've learned throughout the course, with a focus on object-oriented programming.

Learning Objectives

After completing this project, you will be able to: - Create and work with Python classes - Understand inheritance and how to override methods - Use lists and dictionaries to store information - Handle user input and maintain program state - Write clear, well-documented code

Project Requirements

1. Memory System

Your chatbot must include a basic memory system that can: - Keep track of the last 3 messages in the conversation - Use this memory to make conversations more natural

2. Personality System

Your chatbot must include at least two different personalities: - A FriendlyBot that's casual and warm - A TeacherBot that's more formal and educational - Each personality should have its own unique way of talking

3. Basic Conversation Features

Your chatbot must: - Take user input and generate appropriate responses - Remember what was said earlier in the conversation - Stay in character for its personality

Grading Criteria

Basic Functionality (50%)

- Memory system works correctly (20%)
- Personalities are distinct and consistent (15%)
- Responses are appropriate and make sense (15%)

Code Quality (30%)

- Clear variable and function names (10%)

- Good comments explaining your code (10%)
- Proper indentation and organization (10%)

Creativity (20%)

- Unique personality traits (20%)
- Interesting additional features (+5%)

Extra Credit Opportunities (+5%)

You can earn extra credit by adding: - A third personality type - A way to switch between personalities - More types of information to remember - Special commands (like "/help")

Getting Started Tips

1. Obtain your API key from Groq
2. Use the query_llm function in generate_response to see how to access the AI
3. Start with the Memory class
4. Test each method as you write it
5. Add one personality at a time
6. Use the debugger and print statements to debug
7. Ask for help if you get stuck!

Submission Requirements

Required Files

1. Your Python code files (.py)
2. A brief write-up explaining:
    - How your chatbot works
    - What makes each personality unique
    - Any extra features you added

Format

- Submit all files in a zip folder

Timeline and Deadlines

- Project Assigned: July 30
- Final Submission Due: August 6
- No late submissions accepted

Setup Instructions

Obtaining Your API key

You will need to create a Groq account and obtain your own API key to complete the project. Feel free to use any other service as well, just be sure to update `utils.py` accordingly.

1. Navigate to Groq cloud (https://console.groq.com/login)
2. Input your email and click login with email. It will send a link to your email address to register your account.
3. Click the link in the email and sign in to Groq cloud.
4. After logging in, on the left hand side, select API keys
5. Click create API key, enter a name for your key, and click submit
6. Your key will be generated. **Be sure to copy it somewhere otherwise you will need to create a new one.**
7. After creating your API key, you will need to update your `utils.py` with the following code. **Be sure to replace "ADD YOUR GROQ API KEY HERE" with the API key you obtained in step 8.**
8. Additionally, feel free to change the model parameter to any of the LLMs listed (https://console.groq.com/docs/rate-limits) in the free tier. You may notice that some LLMs are better than others.

Creating a Virtual Environment

Using Command Line

1. Open your terminal/command prompt
2. Navigate to your project directory:

   ```
   cd path/to/your/project
   ```

3. Create a virtual environment:
   - Windows: `python -m venv venv`
   - Mac/Linux: `python3 -m venv venv`
4. Activate the virtual environment:
   - Windows: `venv\Scripts\activate`
   - Mac/Linux: `source venv/bin/activate`

Using VSCode

1. Open your project folder in VSCode
2. Press `Ctrl+Shift+P` (Windows) or `Cmd+Shift+P` (Mac)
3. Type "Python: Select Interpreter"
4. Click "+ Enter interpreter path"
5. Create a new virtual environment
6. Select the new virtual environment

Using PyCharm

1. Open your project in PyCharm
2. Go to File → Settings → Project → Python Interpreter
3. Click the gear icon → Add
4. Choose "New environment" and click OK
5. PyCharm will create and configure the virtual environment

Installing Required Packages

After activating your virtual environment:

```
pip install openai streamlit
```

Running Your Code

From Command Line

1. Make sure your virtual environment is activated (you should see `(venv)` in your terminal)
2. Navigate to your project directory
3. Run your chatbot without GUI (useful for testing):

   ```
   python chatbot_logic.py
   ```

4. Run your chatbot with GUI (after you've implemented everything):

   ```
   streamlit run app.py
   ```

From VSCode

1. Open your chatbot_logic.py file
2. Make sure the correct interpreter is selected (bottom left corner)
3. Click the play button or press `F5`
4. To run your full app, use the terminal command from above.

From PyCharm

1. Open your Python file
2. Right-click in the editor
3. Choose "Run 'chatbot_logic.py' "
4. To run your full app, use the terminal command from above.

Troubleshooting

- Make sure you have set your API key in the utils.py file.
- If you get a "module not found" error, make sure:
  1. Your virtual environment is activated
  2. You've installed the required packages

3. You're using the correct Python interpreter