Standard Operating Procedure (SOP) for Building a ChatGPT Clone with Ollama

This SOP **(**Standard Operating Procedure) is a 10-step guide for building a ChatGPT clone utilizing Ollama from installation to customization and testing.

Pre-requisite: Basic understanding of command line interfaces and python programming.

1. Download and Install Ollama:

- Visit the Ollama homepage at https://ollama.ai/
- Click on "Download Now" to download the application.
- Open the downloaded file and install Ollama.
- Note: Ollama is currently available for Mac OS and Linux, with a Windows version in development.

2. Explore Available Models:

- After installing Ollama, locate the icon in the taskbar.
- Click on the icon to access the Ollama homepage, where available models are listed.
- Familiarize yourself with models such as Code Llama, Llama 2, Mistol, Zephyr, Falcon, and Dolphin 2.2.

3. Run Models in Parallel:

- Open the terminal and type the command **ollama run [model_name]** to run a specific model (e.g., **ollama run mistol**).
- Demonstrate running multiple models simultaneously using commands like **ollama run mistol** and **ollama run llama2**.
- Observe the seamless transition between models.

4. Customize Model Parameters:

- Create a model file using a text editor (e.g., Visual Studio Code).
- Specify model details in the model file, such as temperature and system prompts.
- Save the model file and use the command **ollama create [model_profile] [path_to_model_file]** to create a model profile.
- Run the customized model using **ollama run [model_profile]**.

5. Build a ChatGPT Clone:

- Create a new project folder for the ChatGPT clone (e.g., "open_chat").
- Use a Python script (e.g., "main.py") to build the ChatGPT application.
- Import necessary libraries (requests, json).
- Define a function to generate responses (**generate_response**).
- Utilize Gradio to create a user interface for interaction.

- Implement a conversation history mechanism within the script.
- Test the application locally using the terminal.

6. Integration with Gradio:

- Integrate Gradio into the Python script to create a user-friendly interface.
- Define input and output functions for Gradio, linking them to the **generate_response** function.
- Launch the application and access it through the provided local URL.

7. Conversation History Implementation:

- Enhance the script to store and retrieve conversation history.
- Modify the **generate_response** function to append and retrieve conversation history.
- Ensure the model maintains context from previous interactions.

8. Testing and Interaction:

- Test the ChatGPT clone by interacting with it through Gradio.
- Verify that the conversation history is accurately retained for context-aware responses.
- Make adjustments to the script as needed for improved functionality.

9. Further Customizations and Exploration:

- Explore additional Ollama integrations, such as web and desktop interfaces, terminal integrations, and various libraries.
- Experiment with different models and extensions available through Ollama.

10. Documentation and Feedback:

- Document any modifications or improvements made to the script.
- Provide feedback to the Ollama community or developers.
- Share insights or suggestions for further improvements.