ShopAssist AI Project Report with Function Calling

GIT Repo URL: https://github.com/avinashtadavarthy/shopassist-ai

1. Introduction

1.1 Project Objectives

In today's digital age, online shopping has become the go-to option for many consumers. However, the overwhelming number of choices and the lack of personalised assistance can make the shopping experience daunting. To address this, we have developed ShopAssist AI, a chatbot that combines the power of large language models and rule-based functions to ensure accurate and reliable information delivery.

The main objectives of the project were:
- To utilise a database of laptop information.
- Use GPT models to gather user requirements properly into a structured JSON format.
- Utilise both inputs to suggest the best laptops from the database for the user.
- The latest GPT-4o-mini model was used, which is more effective than the GPT-3.5-turbo and GPT-4-turbo models. The effectiveness of GPT-4o-mini lies in its improved accuracy, faster response times, and better understanding of user requirements compared to the other models.

1.2 Approach

- **Conversation and Information Gathering**: The chatbot utilizes language models to understand and generate natural responses. Through a conversational flow, it asks relevant questions to gather information about the user's requirements.
- **Information Extraction**: Once the essential information is collected, rule-based functions extract the top 3 laptops that best match the user's needs.
- **Personalized Recommendation**: Leveraging this extracted information, the chatbot engages in further dialogue with the user, efficiently addressing their queries and aiding them in finding the perfect laptop solution.

The chatbot should ask a series of questions to:

- Determine the user's requirements using six features: GPU intensity, Display quality, Portability, Multitasking, Processing speed, and Budget.
- Confirm if the user's requirements have been correctly captured at the end.
- After that, the chatbot lists the top 3 products that are most relevant and engages in further conversation to help the user find the best one.

2. Project Design

2.1 High-level Architecture

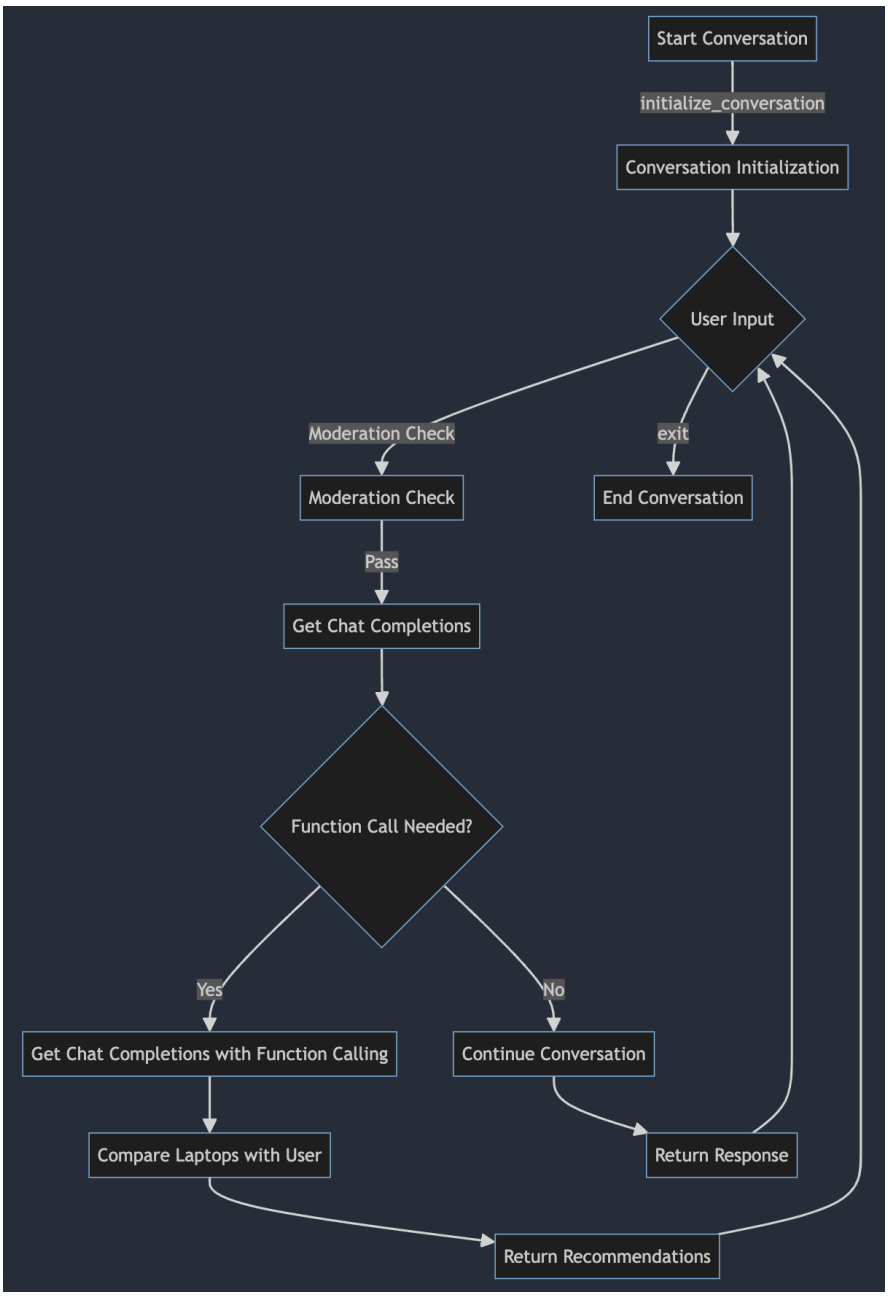
The high-level architecture of ShopAssist AI consists of the following main components:

- Conversation Initialization:
  - initialize_conversation(): Initializes the variable conversation with the system message.
- Chat Completions:
  - get_chat_completions(): Takes the ongoing conversation as input and returns the response by the assistant.
  - get_chat_completions_function_calling(): Similar to get_chat_completions(), but includes the compare_laptops_with_user function defined as a function call. This ensures that if a function

call is needed, it generates the proper JSON and calls the function to get a second response with the function response included.

- Moderation Check:
  - moderation_check(): Checks if the user's or the assistant's message is inappropriate and ends the conversation if necessary.
- Laptop Comparison:
  - compare_laptops_with_user(): Compares the user's profile with different laptops and returns the top 3 recommendations.
- Recommendation Initialization:
  - initialize_conv_reco(): Initializes the recommendations conversation.

Here is a diagram to visualize the high-level architecture:



3. Implementation

The implementation involved the following steps:

- **Conversation Initialization**: Setting up the initial conversation with system messages to establish context.
- **User Interaction**: Gathering user requirements through a series of questions.
- **Moderation**: Ensuring appropriate conversation flow by checking for inappropriate content.
- **Function Calls**: Utilizing function calls to process the user's requirements and fetch the top 3 laptop recommendations.
- **Recommendation and Follow-up**: Providing personalized recommendations and engaging in further dialogue to help the user finalize their choice.

4. Challenges

- **Prompt Optimization**: Ensuring correct responses by adding more information to the prompts.
- **User Interaction Flow**: Avoiding overwhelming the user with questions by improving the system prompt. It was a challenge to make the AI assistant ask questions about all the parameters at once.
- **Function Calling Complexity**: Navigating the complexity of response handling with function calls. Since it was being done for the first time, the task was challenging.

5. Lessons Learned

- **Completions API**: Gained in-depth knowledge of using the completions API.
- **Function Calling**: Understood the merits and complexities of function calling.
- **Pandas Usage**: Improved skills in using pandas for file handling.

6. Future Work

- **Dataset Improvement**: Enhance the dataset with a larger set of laptops.
- **Real-time Data Integration**: Connect with online databases to provide up-to-date information on prices and availability. Some suggestions would be the open product database or other similar databases to get data from the latest data from the internet to be able to accommodate questions from the users such as "where to buy", "latest price" etc which needs real time, real world information.
- **UI Integration**: Develop a user interface and host the chatbot for broader usability.
- **Rigorous Testing**: Conduct extensive testing, including edge cases, and leverage AI for identifying potential failure points during testing.