ShopAssist AI Bot

Table of Contents

[Project Background 1](#_Toc165826092)

[Problem Statement 1](#_Toc165826093)

[System Design 2](#_Toc165826094)

[Technical Implementation 3](#_Toc165826095)

[Function Calling 3](#_Toc165826096)

[Moderation 5](#_Toc165826097)

[User Experience 5](#_Toc165826098)

[Documentation 7](#_Toc165826099)

# Project Background

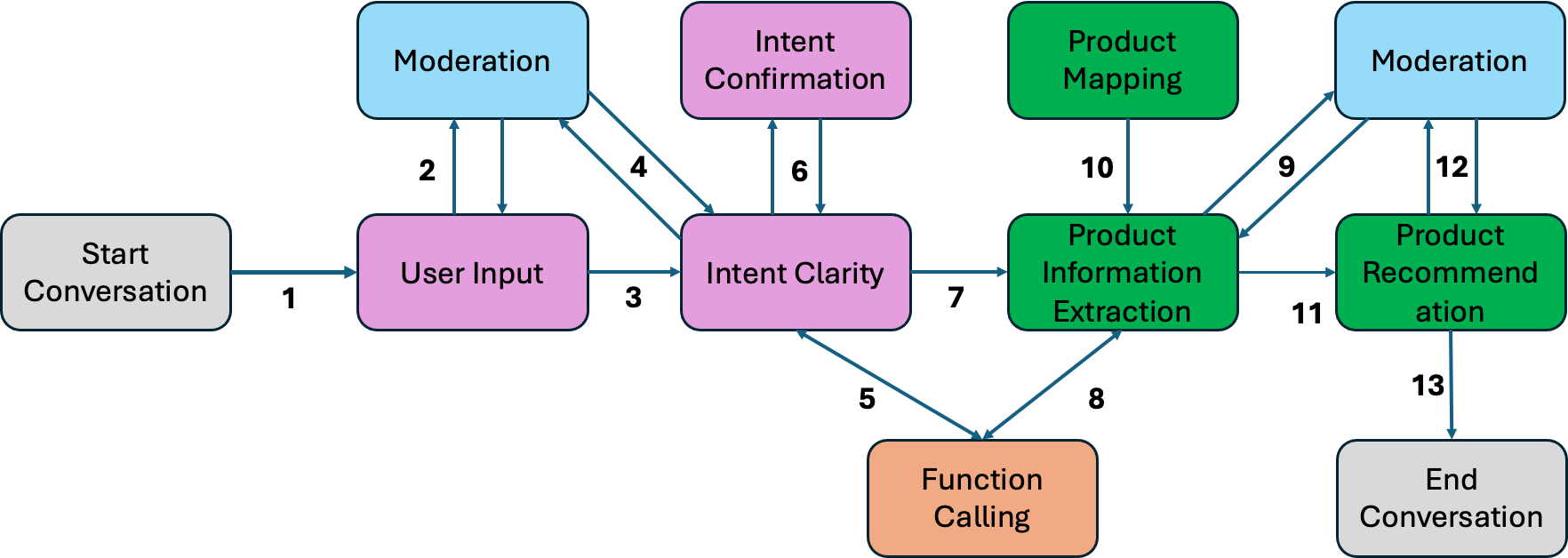
In the current digital era, online shopping has emerged as the preferred choice for numerous consumers. Nevertheless, the extensive array of options and the absence of personalized guidance can render the shopping process intimidating. To alleviate this challenge, we have introduced ShopAssist AI, a chatbot that integrates advanced language models with rule-based functions to ensure precise and dependable information dissemination.

# Problem Statement

Develop a chatbot that leverages a dataset comprising laptop information (including product names, specifications, descriptions, etc.) to deliver precise laptop recommendations tailored to user preferences

# System Design

Shop Assist Chabot Workflow



Grey blocks indicate user initiation and closure

Pink blocks indicate user information extraction

Green blocks indicate product extraction and recommendation

Blue blocks indicate text moderation

Orange block indicate function calling

The enhanced ShopAssist AI Bot system design implements openAI function-calling such that user inputs are correctly classified and appropriate external function is called to classify required user laptop information. This ensures enhanced interaction flow.

This workflow also ensures that moderation is applied at appropriates steps on wherever openAI API is called.

Below are the steps followed in ShopAssist AI Bot.

Step1: User initiates conversation with Shop Assist Bot

Step2: User inputs are moderated for inappropriate content.

Step3: Bot seeks further information on user requirements.

Step4: Bot responses are moderated for inappropriate content.

Step5: Bot checks for appropriate function call

Step6: All user inputs are validated.

Step7: Bot provides information for information extraction.

Step8: Product extraction checks for appropriate function call.

Step9: Product information is validated for inappropriate content.

Step10: Existing products are mapped according to user requirements.

Step11: Shortlisted Products are extracted.

Step12: Short listed products are checked for inappropriate content.

Step13: User ends the conversation after accepting the recommendation.

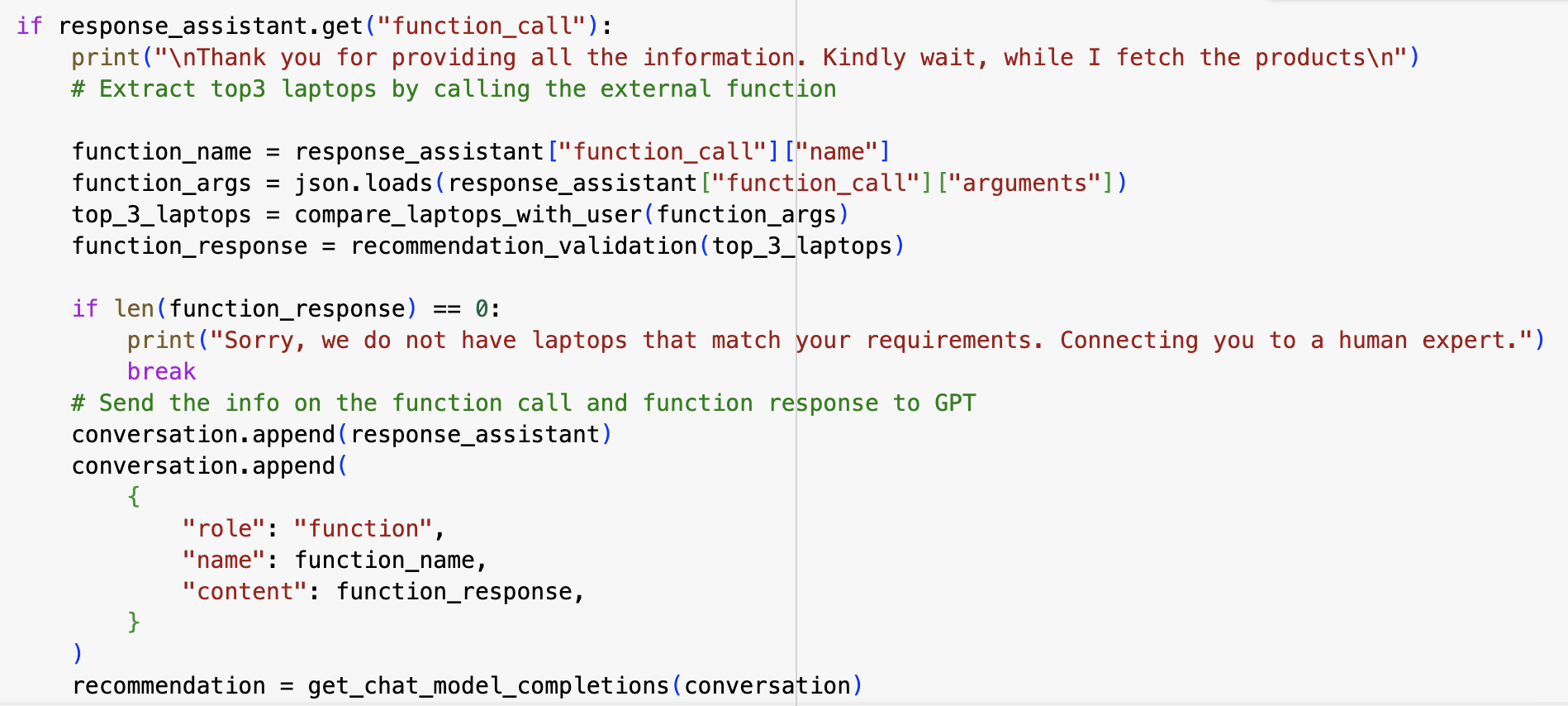
# Technical Implementation

Below are list of all functions defined in the chatbot. In order to follow the 13 steps mentioned in system design, below functions have been defined.

* initialize\_conversation(): This initializes the variable conversation with the system message.
* get\_chat\_completions(): This takes the ongoing conversation as the input and returns the response by the assistant
* moderation\_check(): This checks if the user's or the assistant's message is inappropriate. If any of these is inappropriate, it ends the conversation.
* intent\_confirmation\_layer(): This function takes the assistant's response and evaluates if the chatbot has captured the user's profile clearly. Specifically, this checks if the following properties for the user has been captured or not GPU intensity, Display quality, Portability, Multitasking, Processing speed, Budget
* dictionary\_present(): This function checks if the final understanding of user's profile is returned by the chatbot as a python dictionary or not. If there is a dictionary, it extracts the information as a Python dictionary.
* compare\_laptops\_with\_user(): This function compares the user's profile with the different laptops and come back with the top 3 recommendations.
* initialize\_conv\_reco(): Initializes the recommendations conversation
* product\_map\_layer(): This function is responsible for extracting key features and criteria from laptop descriptions
* recommendation\_validation(): This function verifies that the laptop recommendations are good enough, has score greater than 2, and matches the user's requirements
* function\_description for “compare\_laptops\_with\_user() is definition of all the parameters needed for fulfilling user requirement. This is in dictionary format.

## Function Calling

In this particular ShopAssist AI Bot, function calling is integrated. Function calling is used for extracting structured insights from unstructured text. This unstructured text is the output from chat completion API which has relevant laptop data instructions like portability, budget, GPU, graphics, etc from the user and from chat completion API.



The above code snippet shows how function calling was integrated into ShopAssist AI Bot.

The user is prompted till all the required information is confirmed by the intent\_confirmation function. When all information is available, function calling is invoked. Once again the unstructured data is converted to structured data in dictionary format by function call. Also the mandatory fields are ensured to be filled and in the format of “high”, “medium”, “low” with the budget in numeric format.

In the code, we see that the corresponding function call name is stored in variable “function\_name”. The output from function call is then provided to function “compare\_laptops\_with\_users” to get the top3 recommended laptops.

Further the code also checks if there is a function response based on user requirements. If there is a case where all information is provided but still no laptops meet the requirement then the bot responds with - No laptops available, message.

The structured information is again fed to the chat completion API to further check for appropriate recommendations.

Below is an example output of chat completion when function call is invoked.

{

"role": "assistant",

"content": null,

"function\_call": {

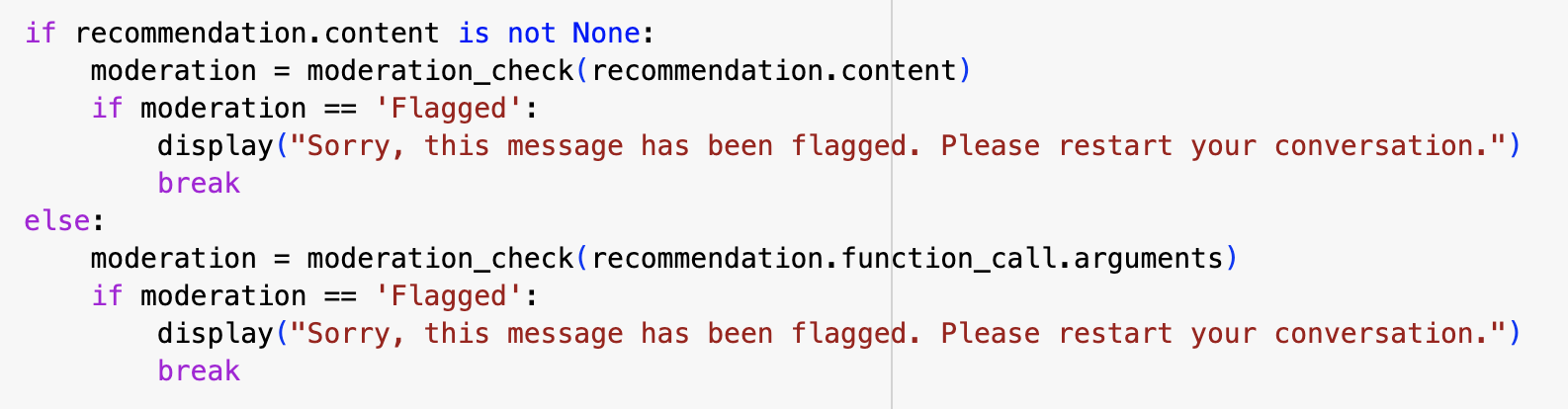
"name": "compare\_laptops\_with\_user",

"arguments": "{\n\"gpu intensity\": \"high\",\n\"display quality\": \"high\",\n\"portability\": \"low\",\n\"multitasking\": \"high\",\n\"processing speed\": \"high\",\n\"budget\": 300000\n}"

}

}

## Moderation



The above code snippet is exercised for moderation check. Moderation ensures that inappropriate text is not entertained in the ShopAssist AI Bot. Moderation check is called every time there is a user input or a response from chat completion API.

There is if, else statement defined for moderation check. The reason is because the content field of assistant response is not null when function call is not invoked. Also, the “arguments” field is populated when function call is exercised.

Below is an example output of chat completion where “content” is null and “arguments” is populated.

{

"role": "assistant",

"content": null,

"function\_call": {

"name": "compare\_laptops\_with\_user",

"arguments": "{\n\"gpu intensity\": \"high\",\n\"display quality\": \"high\",\n\"portability\": \"low\",\n\"multitasking\": \"high\",\n\"processing speed\": \"high\",\n\"budget\": 300000\n}"

}

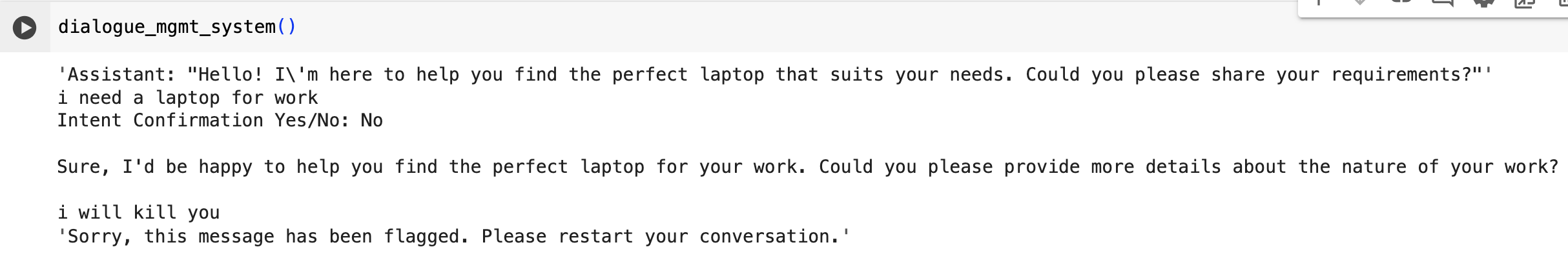
}

# User Experience

User experience has improved because of moderation check and function calling.

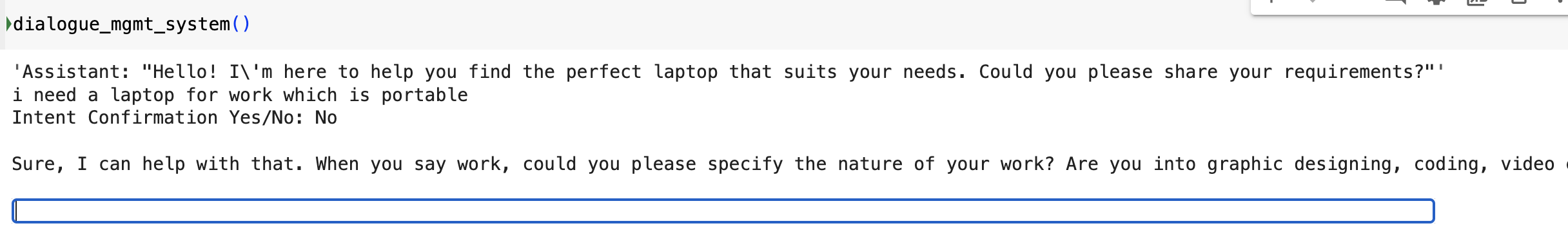
Below are some examples.

1. Inappropriate text

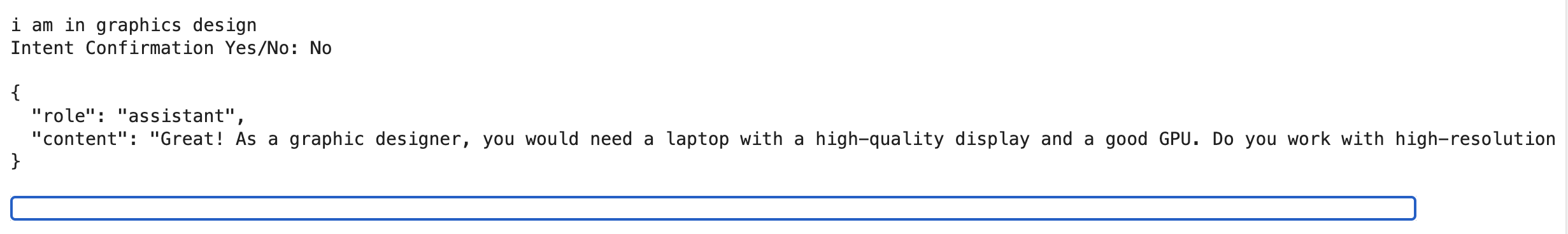


The above example shows how moderation kicks in when user enters inappropriate text.

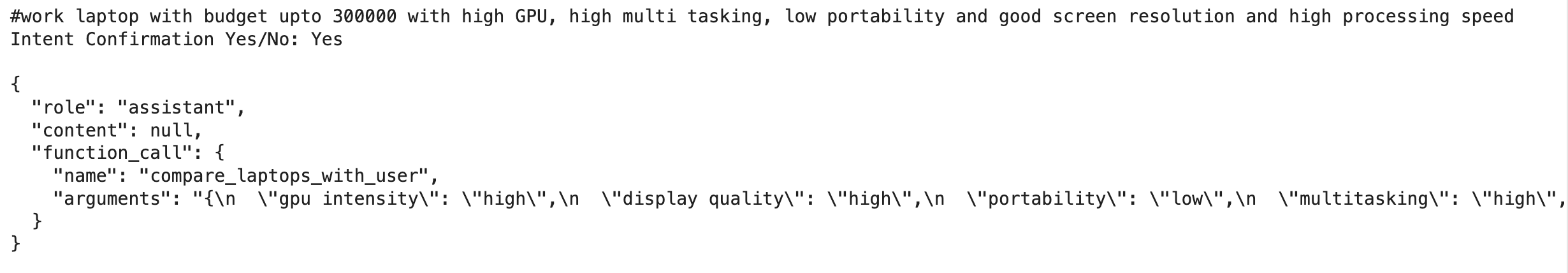
1. Function calling



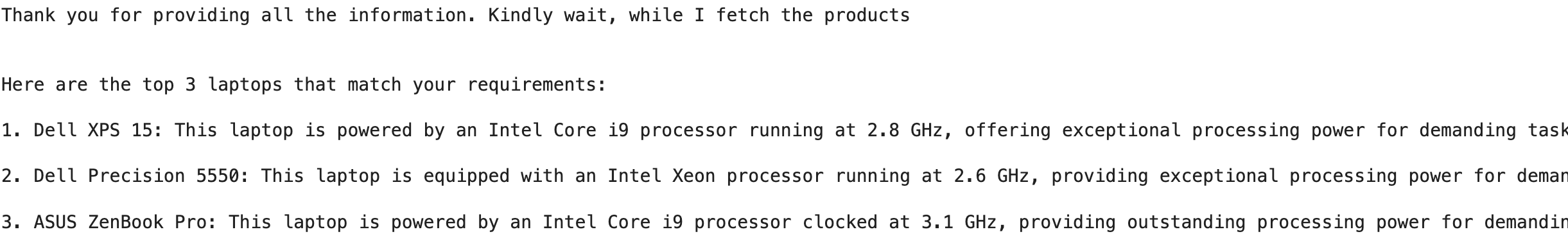
The above example shows how the bot continues the conversation and prompts for required mandatory information.



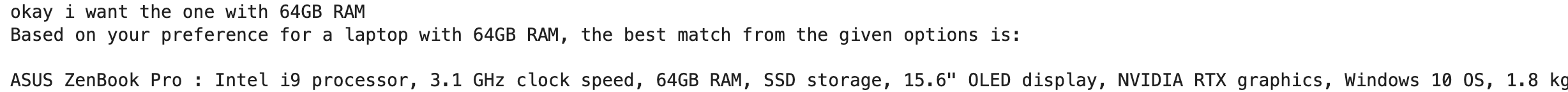
The above example shows how further information is extracted.



The above example shows how function calling is invoked and all information is available and given values – high, medium and low. This ensures that the data is then parsed through available laptop dataset to recommend appropriate laptops.



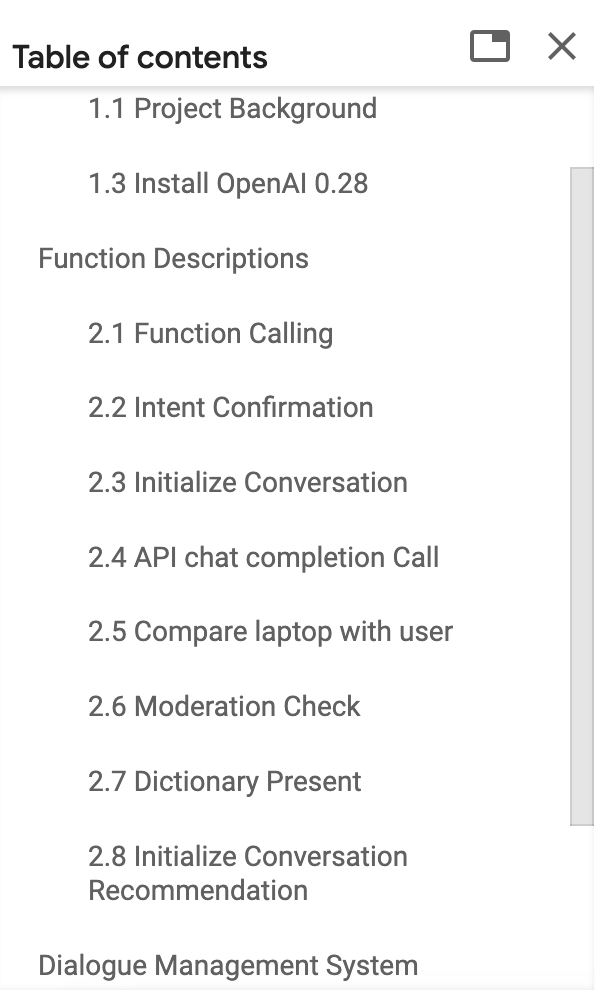
The above example shows how the user requirements are met and top 3 laptops are recommended.



The above example shows how the bot further narrows down the recommended laptop based on user request.

# Documentation

Appropriate documentation can be found in the code notebook attached in the assignment. Here is snapshot from notebook.



Instructions to Run the code.

1. Install openAI 0.28
2. Change your OS directory. Currently set to

os.chdir('/content/drive/MyDrive/upgrad/GenAI\_Course\_Master/Course\_1\_ShopAssistAI/Week\_2/Session\_1/')

1. Sample string user input.

#work laptop with budget upto 300000 with high GPU, high multi tasking, low portability and good screen resolution and high processing speed