**ShopAssist AI 2.0 – Intelligent Laptop Recommendation Chatbot**

**Overview**

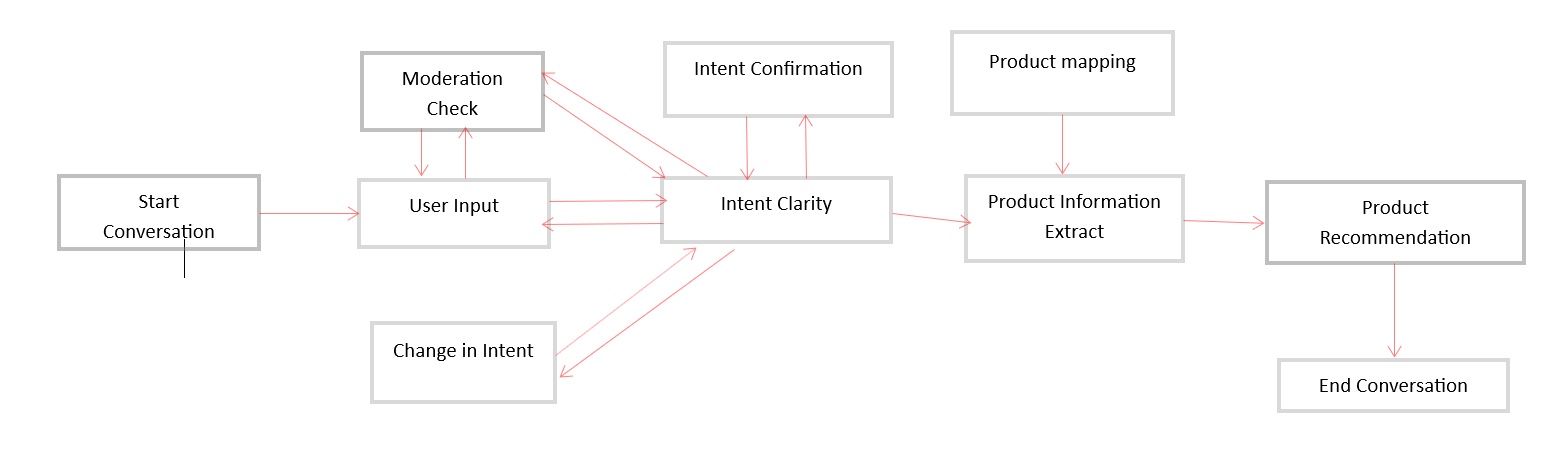
**ShopAssist AI 2.0** is a conversational assistant built using OpenAI's Function Calling capability to provide personalized laptop recommendations. It collects detailed user requirements and maps them to a dataset of laptops using domain-specific classification rules. The assistant presents the top 3 matching products using natural language responses.

**Model Used**: gpt-3.5-turbo-1106

**Open AI API’s**: Chat Completions and Moderation

**Dataset**: laptop\_data.csv

**System Design:**

****

Note: Used the same stages and System design of given ShopAssist AI 1.0 and made below enhancements in 2.0

**New Enhancements:**

**Mid-Conversation Intent Reset**

* **What it does**: Detects if the user changes the **primary purpose** or **key preferences** mid-conversation (e.g., from gaming to programming).
* **How**: Compares new user input to previous confirmed intents. If there's a shift in purpose or a major change in specs, it **resets the conversation context** and restarts the recommendation flow.
* **Benefit**: Prevents irrelevant results and confusion; enables adaptive and natural conversation handling.

**Function Calling API Integration**

OpenAI Function Calling is used for:

| **Purpose** | **Function Used** | **Description** |
| --- | --- | --- |
| Extracting user intent | intent\_confirmation\_layer() | Extracts structured user preferences (e.g., high GPU, low weight) from conversation. |
| Classifying laptop features | product\_map\_layer() | Converts free-form product descriptions to structured feature-level specs. |
| Finding top matches | compare\_laptops\_with\_user() | Filters and ranks top 3 laptops from dataset using classified features. |
| Recommendation | initialize\_conv\_reco() | List of recommended laptops with name, specs, and price. |

**Conversation Workflow (dialogue\_mgmt\_system)**

**Step-by-Step Flow:**

1. **Initialize Conversation**  
   Starts with a system prompt defining chatbot behavior and waits for user input.
2. **User Provides Input**  
   The assistant receives and processes user needs (e.g., "I want a laptop for gaming").
3. **Moderation check**

User input and Assistant response is monitored by Open AI moderation and if any Unethical input is passed then this will be flagged and ends the conversation.

1. **Function Call Detection**  
   If GPT detects the need to extract structured intent, it triggers intent\_confirmation\_layer().
2. **Mid-Conversation Reset Check** ✅ *(New)*
   * Compares the newly extracted intent to previously stored one.
   * If the purpose has changed (e.g., from "gaming" to "business use"), the bot:
     + Resets the full conversation.
     + Reinitializes all context.
     + Asks the user to start again.
3. **Recommendation Pipeline**
   * Extracts specs from the laptop dataset using product\_map\_layer().
   * Filters and compares using compare\_laptops\_with\_user().
   * Validates the result set.
4. **Natural Language Response**
   * GPT generates final recommendations using structured info as context.

**Benefits of This Architecture**

| **Feature** | **Advantage** |
| --- | --- |
| 🔁 Mid-Convo Reset | Makes interaction more natural and error-free when users change their mind. |
| 🧩 Modular Function Calling | Ensures clarity and maintainability; each task is clearly separated. |
| 🧠 Feature-Level Matching | Higher accuracy of product matches thanks to structured classification. |
| 🔍 Top-3 Result Selection | Prioritizes and surfaces only the most relevant laptops to avoid information overload. |
| 🤖 GPT Response with Context Memory | Keeps conversation coherent while supporting follow-up questions. |

**Code Structure Summary**

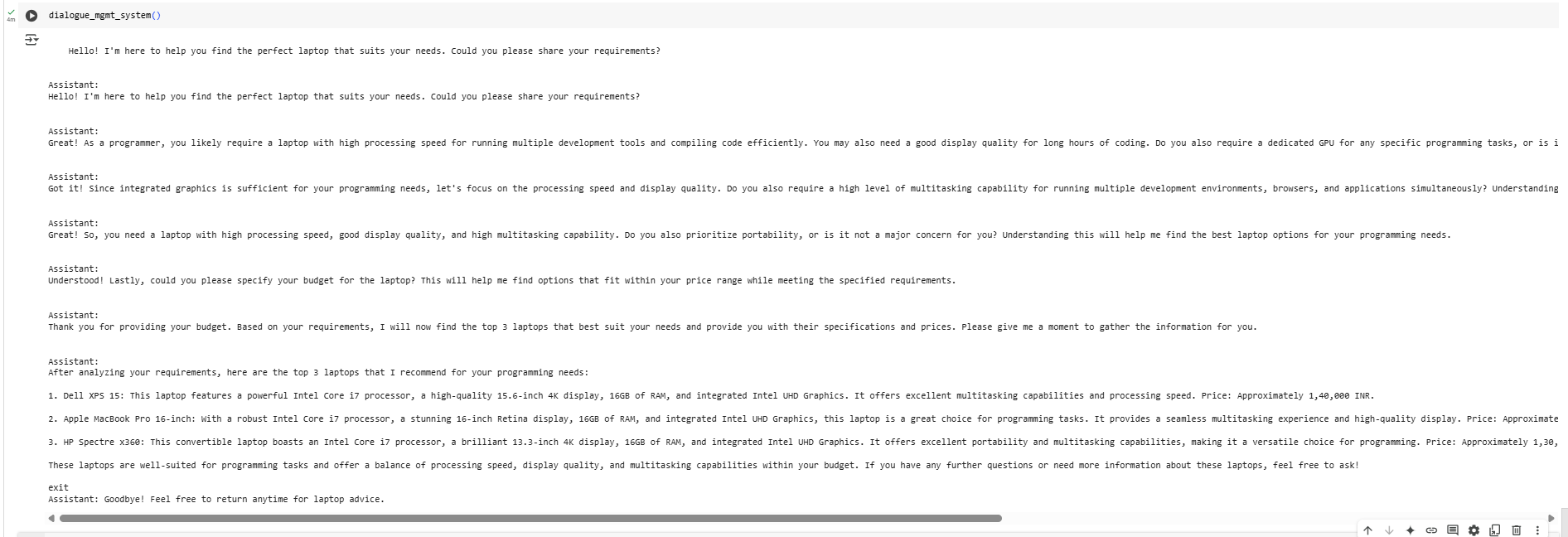
* dialogue\_mgmt\_system() – Main loop handling interaction flow.
* initialize\_conversation() – Loads system prompt and sets initial conversation context.
* get\_chat\_model\_completions() – Makes API calls to GPT with tools for function-calling and normal calls.
* intent\_confirmation\_layer() – Extracts structured intent JSON from GPT.
* is\_intent\_changed()- Compares new user input to previous confirmed intents. If there's a shift in purpose or a major change in specs, it resets the conversation context and restarts the recommendation flow.
* compare\_laptops\_with\_user() – Compares user preferences with classified specs in dataset.
* product\_map\_layer() – Converts free-text descriptions to 5-dimension classification (GPU, RAM, etc.).
* recommendation\_validation() – Ensures top 3 results are returned.
* initialize\_conv\_reco()- Top 3 laptops are recommend to User with LLM response.

Chatbot Validation:

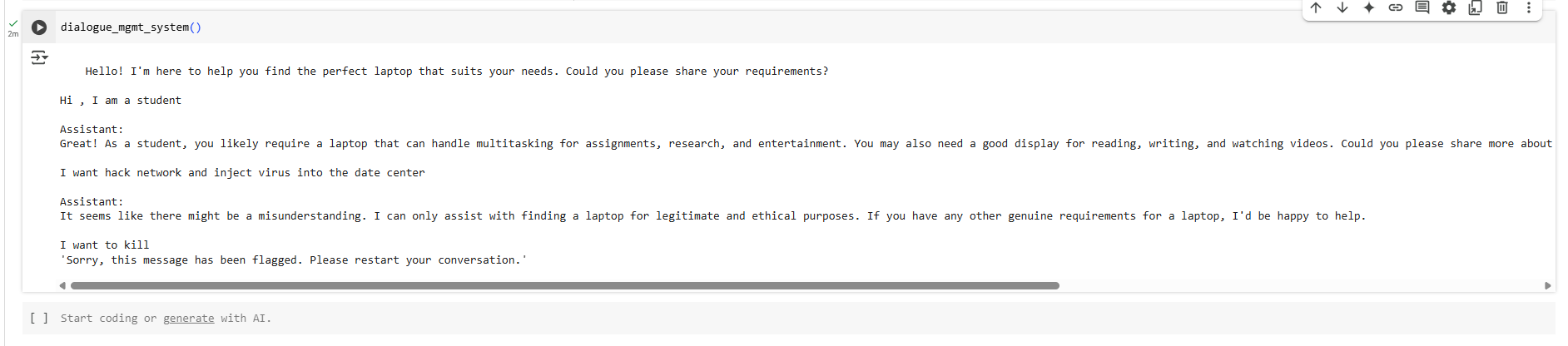
Test Scenarios:

Scenario 1: User and Assistant interaction for a Laptop recommendation

Conversation screenshot output



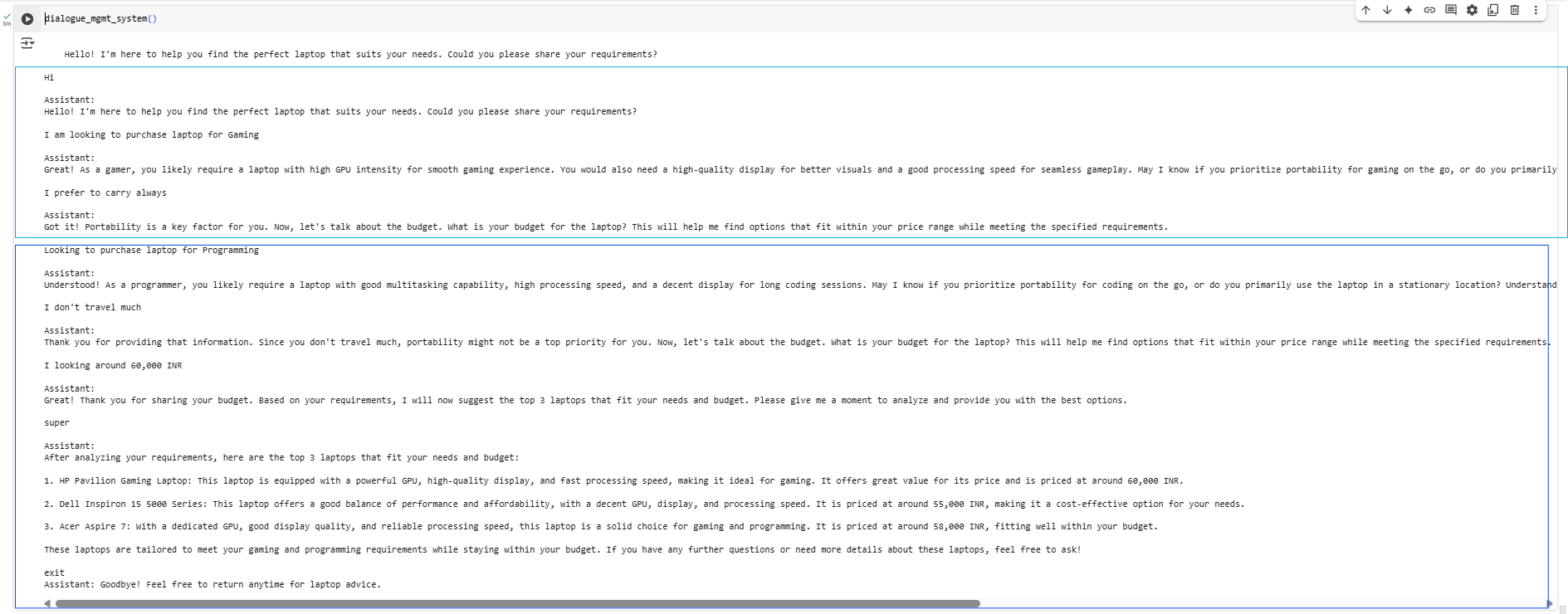
Scenario 2: User asked about unethical question during the conversation so this will be flagged and ends conversation



Scenario 3: User Intent change during the conversation

In the initial conversation – User intended to purchase laptop for Gaming

Later changed the intention to purchase laptop for Learning



**Challenges Encountered**

During implementation, the following challenges were addressed:

* Seamless integration between traditional methods and function calls.
* Managing both function-enabled and standard chat completions effectively.
* Designing a robust approach to handle shifts in user intent or purpose mid-conversation.
* Encountered multiple hurdles while integrating the function call flow into the dialogue\_mgmt\_system().

**Key Learnings**

This project provided several valuable takeaways:

* Gained hands-on experience with OpenAI's Function Calling API.
* Experimented with different OpenAI models to evaluate text generation accuracy and response quality.
* Developed new techniques to dynamically handle changes in user intent regarding laptop preferences.
* Enhanced skills in debugging complex method flows.
* Explored various model parameters (e.g., temperature, top\_p) to control randomness and ensure consistent, context-aware responses.