Car Chat Bot: Car Buy AI - Shantanu Saxena

1. Introduction

1.1 Project Goals

Objective: Create a smart assistant bot that suggests cars tailored to users' preferences and budget.

Target Audience: Individuals seeking cars with particular specifications.

Key Features:

- Personalized recommendations
- Simplified car selection process based on the following keys:
  - Car category
  - Kmpl (Average of 4-wheeler)
  - Seater
  - Transmission
  - Fuel type
  - Price

1.2 Scope

In-Scope: Utilizing a terminal for static chat while working with data in a Jupyter notebook in VSCode.

2. Car chat bot AI Model

2.1 Model Selection

Chosen Model: GPT-3.5-turbo by OpenAI.

2.2 Model Training

Initial dataset: We used pre-trained car data collected from a GitHub link or found online.

Fine-Tuning Data: We’ve added extra features for dealer details and GPS information. I'm writing comprehensive prompts that describe every step and scenario in detail.

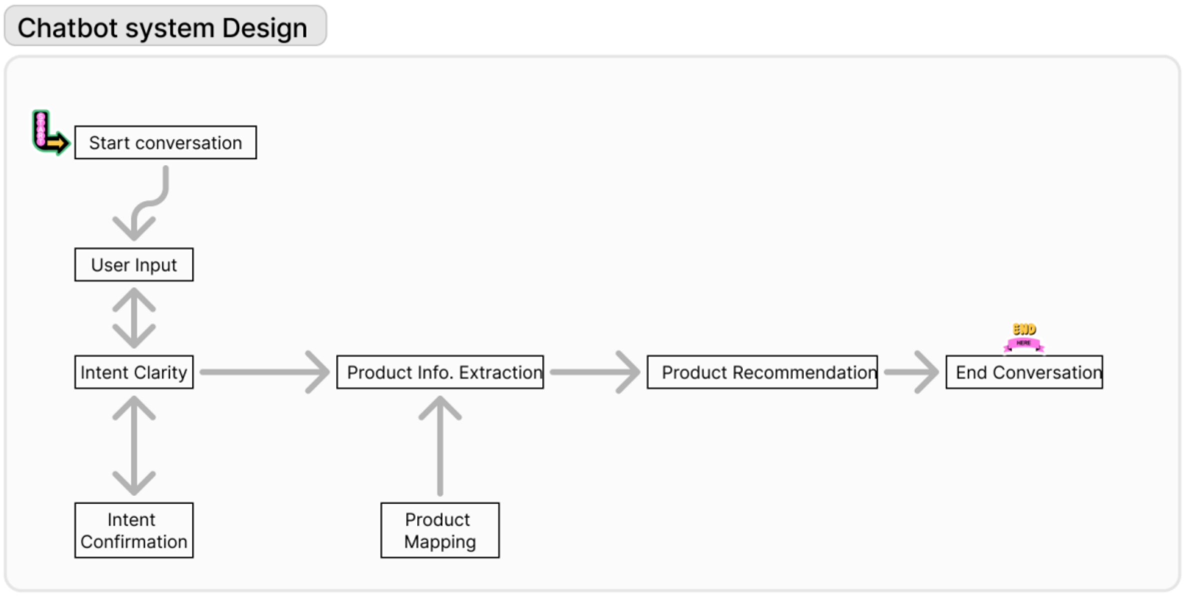
2.3 Model Performance

Metrics: comparing the data gathered by the bot during the conversation with the expected data.

3. Data Details

Column	Description
car model	The specific model name of the car
car category	Category of the car (e.g., Compact SUV, Sedan)
brand name	The manufacturer or brand name of the car
price	Price of the car in INR
engine	Engine details of the car
transmission	Type of transmission (e.g., Manual, Automatic)
fuel type	Type of fuel used (e.g., Diesel, Petrol, Electric)
kmpl	Kilometers per liter (fuel efficiency)
color	Color of the car
year	Year of manufacture
seater	Number of seats in the car
carfullfeature	Full feature description of the car
dealer ID	A unique identifier for each dealer
dealer name	Name of the dealer
dealer GPS	GPS location of the dealer

4. Bot Design Steps

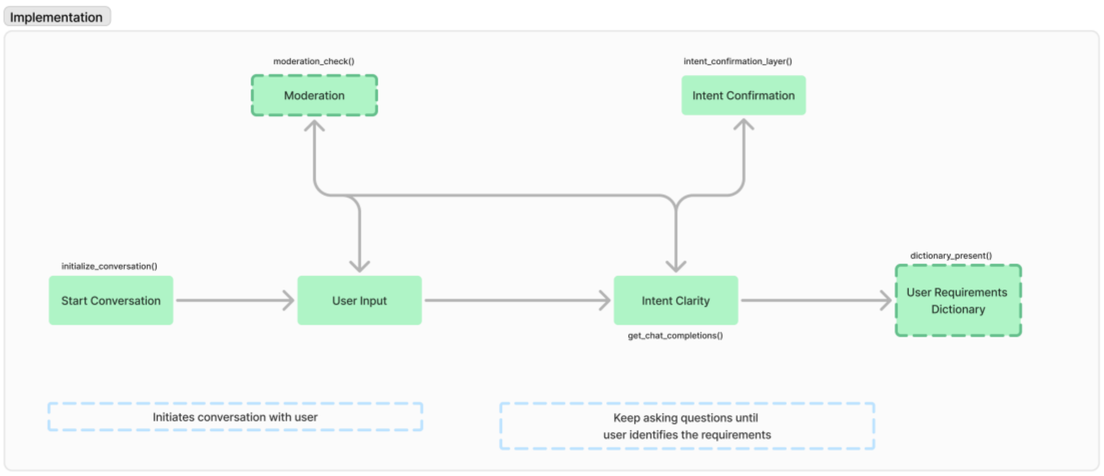


5. Bot Innovation and Creativity

1. Whenever the intent wasn't clear, I added code to display the intent dictionary alongside the messages.
2. In some instances, the results were only showing 2 cars instead of the top 3. Adding the intent helped resolve this issue.
3. Tested the dialogue_mgmt_system() function across various car categories.
4. Explored different strategies to enhance the accuracy of the recommendations.

5.1 User Preferences

The first step is to engage with the user to understand their preferences and needs. This happens through a series of questions and interactions where the bot gathers information like user interests, past interactions, specific requirements, and context. By using natural language processing (NLP), the bot can grasp and pull-out important details from the user's responses, ensuring it collects accurate and thorough data. This information serves as the basis for personalized car recommendations.


Implementation
moderation_check()
Moderation
intent_confirmation_layer()
Intent Confirmation
initialize_conversation()
Start Conversation
User Input
Intent Clarity
get_chat_completions()
dictionary_present()
User Requirements Dictionary
Initiates conversation with user
Keep asking questions until user identifies the requirements

Below is the description of methods used:-

- **initialize_conversation():** This initializes the variable conversation with the system message.

- **get_chat_completions()**: Receives ongoing conversation input and generates the assistant's reply.
- **moderation_check()**: Monitors for inappropriate content from both user and assistant, halting the conversation if detected.
- **intent_confirmation_layer()**: Assesses if the chatbot has accurately captured the user's preferences in terms of kmpl, car category, seater, transmission, fuel type, and price.
- **dictionary_present()**: Verifies if the user's profile is correctly encapsulated in a Python dictionary, extracting relevant data.

5.2 Recommendation’s Prepared

After gathering the user’s preferences, the system processes this information to filter initial car recommendations from the database. The suggested cars should meet or exceed the user's specifications and be priced up to 20% lower than their preferred options. Additionally, the mileage (kmpl) and the number of seats can be equal to or greater than what the user prefers.

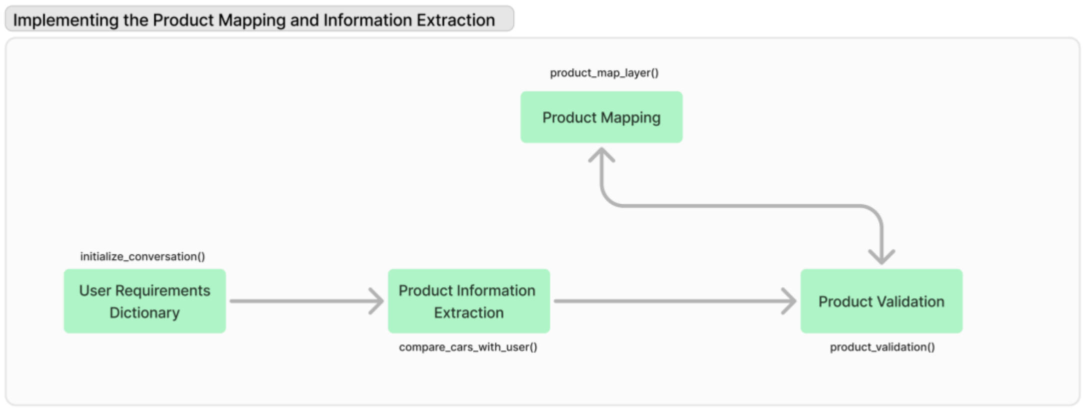
Below is the description of methods used:

- **compare_cars_with_user():** It matches the user's profile with different cars to identify the top three recommendations.
- **product_map_layer() & dictionary_present ()**

  - It serves as a Car Specifications Classifier, designed to extract important features and categorize them based on car descriptions. It follows step-by-step instructions for extracting car features from descriptions.

  - Uses specific rules for each feature—like mileage (kmpl), car category, seating capacity, transmission type, fuel type, and price—and links them to the corresponding classification value.

- Includes Few Shot Prompting, which features a sample conversation between the user and the assistant to illustrate the expected results of the feature extraction and classification process.

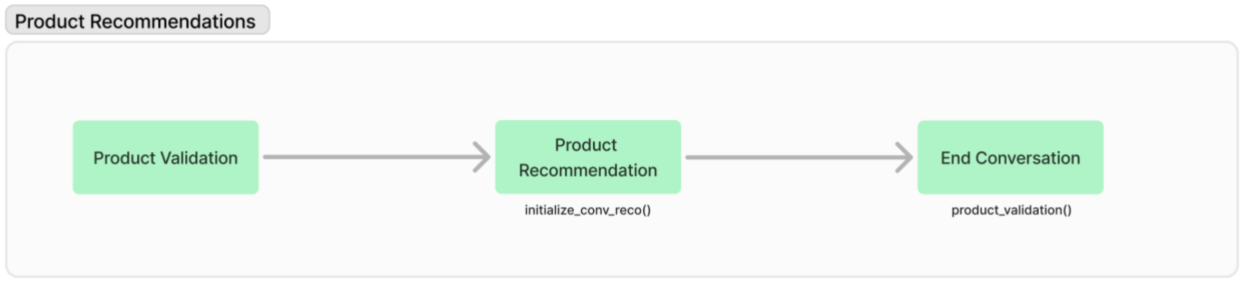

Implementing the Product Mapping and Information Extraction
product_map_layer()
Product Mapping
initialize_conversation()
User Requirements Dictionary
Product Information Extraction
compare_cars_with_user()
Product Validation
product_validation()

5.3 Final Recommendation

After showing the initial car recommendations, the bot stays interactive, allowing the user to give feedback or ask follow-up questions. It refines the suggestions by incorporating the user's input, clarifying their preferences, and addressing any uncertainties. This ongoing process ensures that the final recommendations are tailored to the user's needs, creating a more precise and satisfying experience.

Below is the description of methods used :-

- **initialize_conv_reco():** Initializes the recommendations conversation, to have further discussion about the recommended car and suggest the best option.



6. Challenges Encountered

Discussion of challenges encountered during the project and their solutions.

Case 1:

- **Issue:** When the intent was 'no', the bot was not providing reasons even if all 6 key values were filled. The bot should provide whatever data was collected and not ask follow-up questions. It was simply saving the intent as 'no'.
- **Solution:**
  - Added prompts to encourage the bot to ask questions unless all dictionary values are filled.
  - Started showing the dictionary values.
  - Added more sample conversations in prompts with a different number of questions.

Case 2:

- **Issue:** While engaging with the user for QA in the function **dialogue_mgmt_system()**, in a few cases, the chatbot was replying with 2 recommendations instead of 3.
- **Solution:** Specified in the prompt that it has to show 3 recommendations. However, if only matching data of 2 cars, then it will show 2 cars. This depends on current matching car in DF we have used in this example.

Case 3:

- **Issue:** In case of a spelling mistake, the chatbot was not able to automatically understand the intent. Example:

```
{"car category": "SUV",
 "kmpl": 15,
 "seater": 5,
 "transmission": "autometic",
 "fuel type": "diesel",
 "price": "low"}
```

- **Solution:** I asked for the name of the wrong key, which the chatbot answered, and corrected the same.

7. User Experience and Evaluation

While executing the function **dialogue_mgmt_system()**, I asked for the following preferences:

The function **evaluate_model_response()** also performed well when checked for student, family, and business preferences.

The function **evaluate_model_response()** also performed well when checked for student, family, & business preferences. The key values matched with user and hence high score.

8. Conclusion

8.1 Summary

- Successfully developed a car recommendation bot with advanced personalization features.
- Leveraged GPT-4 for robust natural language understanding and interaction.

8.2 Future Work

ENHANCEMENTS FUNCTIONAL:

- Once the products are extracted, the dialogue flow doesn't allow recalling of product extraction if there is any intent change.
- The output format of each layer is inconsistent. The function API capability of GPT to instruct the output format as per the input request can be used to fix it.
- User satisfaction rating could be added.
- Add another feature so that chat can be continued starting fresh after an intent change.
- ***Function Calling*** using the GPS date link can be enhanced. For now, the chat bot responds with GPS link. It will be great user experience in case those links come as clickable (using function calling to Google's MAP using Google's API) and then user can reach directly to dealer location.

ENHANCEMENTS TECHNICAL:

- Web application development
- UI development for better user interaction
- In case of a spelling mistake, the chatbot is not able to solve it.