**Use Case Documentation FoodieSpot Conversational AI**

**Goal**

Create an AI-powered chatbot to streamline FoodieSpot's restaurant reservation system, allowing users to find, explore, and book restaurants efficiently via a conversational interface.

**Long-Term Goal**

* Enhance customer experience by automating restaurant discovery and reservations.
* Reduce the workload on customer service representatives.
* Improve table occupancy rates across multiple locations.

**Success Criteria**

* The bot should be able to:
  + Accurately recommend restaurants based on user preferences.
  + Handle user queries without human intervention.
  + Complete a reservation quickly.
  + Maintain a user satisfaction rating based on feedback.

**Use Case (100 Words)**

FoodieSpot, a restaurant chain with multiple locations, wants to enhance its reservation system using a conversational AI chatbot. Customers can interact via a chat-based interface to discover restaurants based on cuisine, location, and seating preferences. The bot will utilize an LLM (Llama-3.1-8B) with function calling to dynamically determine user intent and fetch real-time restaurant availability. Users can finalize their selection and book tables seamlessly. The chatbot will provide instant confirmations and allow modifications to bookings. This system aims to improve customer experience, increase reservation efficiency, and optimize restaurant occupancy rates.

**Key Steps (Bot Flow)**

1. User starts a conversation and specifies preferences (cuisine, location, seating, timing, etc.).
2. The chatbot detects intent and fetches restaurant options.
3. User selects a restaurant from the provided list.
4. The chatbot checks table availability and presents time slots.
5. User confirms a suitable time, and the bot books the table.
6. Confirmation is sent, with options to modify or cancel the reservation.
7. Post-reservation, the bot can gather feedback.

**State Transition Diagram**

A diagram of a flowchart

Description automatically generated

**Bot Features**

|  |  |  |
| --- | --- | --- |
| **Feature** | **Details** | **Difficulty** |
| Natural Language Processing | LLM-based intent recognition (Llama-3.1-8B) | 🟢 Easy |
| Restaurant Discovery | Fetching restaurant options dynamically | 🟡 Medium |
| Table Reservation | Checking availability & confirming booking | 🟡 Medium |
| API Integrations | Restaurant database, reservation system | 🟡 Medium |

**Knowledge Bases (KBs)**

* Restaurant database (locations, cuisine, seating capacity, availability)
* User reservation history
* FAQs (common user queries on reservations, modifications, etc.)

**Required Integrations**

* Restaurant management API (for real-time availability)

**Scale-up / Rollout Strategy**

1. **Phase 1 (Pilot)** – Deploy chatbot for a small set of restaurants to test user interactions and refine intent recognition.
2. **Phase 2 (Expansion)** – Scale up to all locations with real-time integrations.
3. **Phase 3 (Enhancements)** – Introduce multilingual support, advanced AI personalization, and payment integrations.

**Key Challenges**

* **Intent Recognition Accuracy** – Ensuring the LLM correctly identifies user intent without hardcoded rules.
* **Tool Calling Optimization** – Managing function execution dynamically within the LLM’s responses.
* **Scalability** – Handling high concurrent user interactions during peak hours.
* **Real-Time Data Sync** – Keeping restaurant availability updated in real time.
* **Fallback Handling** – Seamless handoff to human agents for unresolved queries.

**Prompt Engineering Approach**

**Key Techniques Used:**

1. **Zero-shot & Few-shot Prompting:**
   * Used structured prompts to guide Llama-3.1-8B without pretraining specific queries.
   * Few-shot examples included to improve response accuracy.
2. **Function Calling Implementation:**
   * Designed system prompts to enable the LLM to call functions dynamically.
   * Included JSON-based expected outputs to enhance tool calling reliability.
3. **Context Retention:**
   * Maintained conversation state using history management.
   * Reinforced previous user inputs within subsequent queries.

**Example Conversations:**

1. Finding Restaurant

User: Find italian restraunt in downtown for 4 people at 7 pm

Bot: 1. Pasta Paradise (italian) in Downtown .....

1. Making reservation

User: Book a table at Pasta Paradise for 4 people tonight

Bot: Reservation Confirmed at Pasta Paradise

1. General Query

User: What cuisine do you offer?

Bot: We offer Itlian, Mexican, Spanish ...

**Assumptions & Limitations**

**Assumptions:**

* Users will provide clear preferences in a structured manner.
* The restaurant database will always be up-to-date with accurate availability.
* Internet connectivity is reliable for real-time API calls.

**Limitations:**

* Does not handle complex, multi-turn negotiations for group bookings.
* Limited to English (multilingual support planned for later phases).
* Cannot handle special dietary restrictions dynamically.
* External API failures may cause temporary unavailability of some features.