FoodBot — AI-powered Food Ordering Assistant

Just do it - Prosus Track

Team: Just do it

https://lablab.ai/event/raise-your-hack/just-do-it-prosus-track

Important documentation

You can read a better & complete documentation here: https://github.com/Ashwath-Shetty/raise_hackathon_prosus_track

Problem Statement

- Ordering food is often frustrating and time-consuming.
- Users struggle to discover good restaurants quickly and conversationally.
- Need for a seamless, intelligent, conversational ordering experience.

Our Solution — FoodBot

- An AI-powered chatbot that orders food effortlessly.
  - Location-based suggestions
  - Natural conversation flow
  - Smart memory and recommendations.

Key Features

🔍 Location Normalization (LLM-powered)

🌟 Top restaurant search (SerpAPI)

🧠 LangChain agent orchestration

💬 Gradio UI

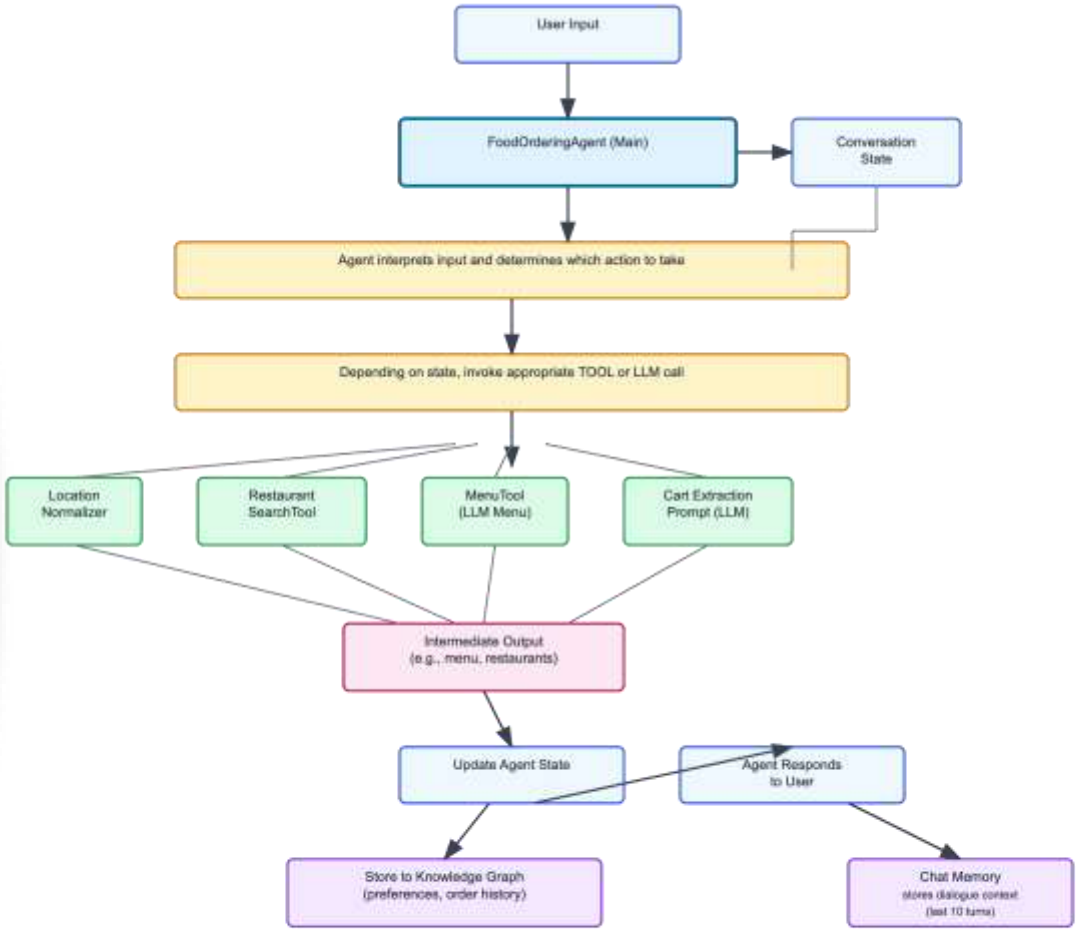
🧠 In-memory knowledge graph

⚡ Groq + LLaMA3 for fast reasoning.

Architecture Overview

- Modular design using LangChain
  - Agent orchestrates tools and logic
  - Location normalization, restaurant search, and menu tools
  - Gradio UI for user interaction

Agents Workflow

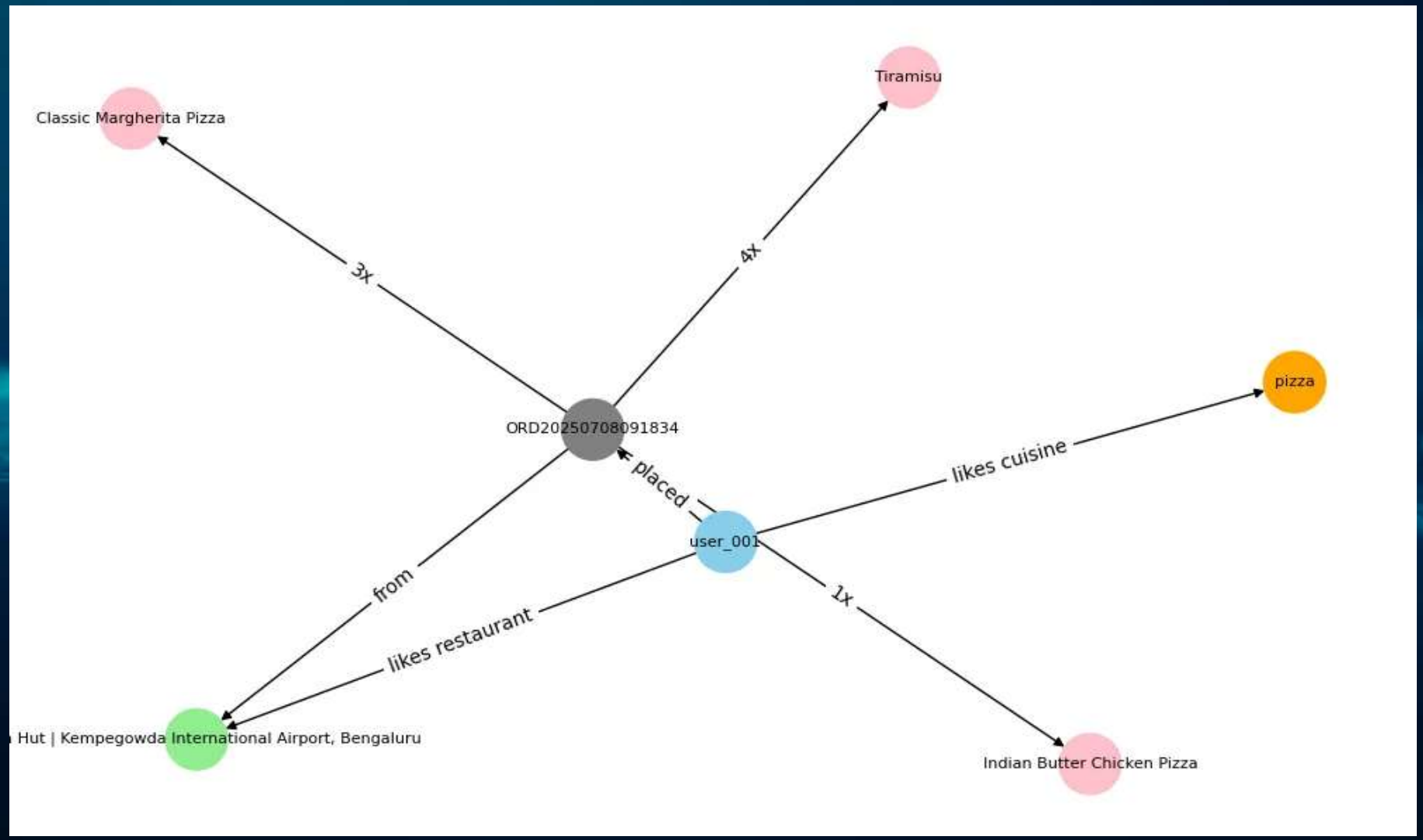


Tools / Agents

🛠️ Tool Descriptions

Tool Name	Purpose	Input Format	Output Format
LocationNormalizerTool(LLM)	Normalize user's text input location (e.g., "near Jyothi Nivas") uses LLM to understand the location and get the coordinates	`user_message: str`	`{ "location": "Koramangala, Bengaluru", "ll": "..." }`
RestaurantSearchTool	Fetch top 3 restaurants using SerpAPI (Google Maps)	`location: str`, `food_type: str`	Formatted string or JSON of top restaurants
MenuTool	menu for selected restaurant	`restaurant_name: str`, `cuisine_type: str`	`formatted_menu: str`, `structured_items: JSON`
Cart Extraction (LLM)	Extract items and quantities from natural text (user says "2 pizzas"), helpful for cart update, menu lookup etc.	Prompt includes `menu` + `user message`	`[{"item": "Pizza", "quantity": 2}]`
Knowledge Graph	Store and query user's preferences and order history	Accessed via user_id	JSON-like structure with past orders & locations

Knowledge Graph



Conversation Flow

- Smooth natural language flow:
  Greet user → Ask location → Suggest restaurants → Show menu → Update cart → Checkout and confirm order.

🍕 Food Ordering Chatbot

Welcome to our AI-powered food ordering service! I'll help you find restaurants and place orders.

hi

Send

New Order

Show Knowledge Graph

Knowledge Graph

Live Demo

- Try it live!
  (https://huggingface.co/spaces/Ashwath-Shetty/food-ordering-bot)

Future Improvements to the food ordering App

- WhatsApp/Twilio integration for confirmations
  - Advanced menu retrieval and structured scraping
  - Asynchronous execution
  - Persistent user memory
  - Personalized recommendations based on history.

Challenges & Learnings

- Menu data inconsistencies
  - API rate limits
  - Fine-tuning conversational flows
  - Balancing personalization with privacy.

How to Extend this solution to Other services

- https://github.com/Ashwath-Shetty/raise_hackathon_prosus_track?tab=readme-ov-file#-how-to-extend-this-solution-to-other-services

How to Extend this solution to Other services – Travel Booking

💼 1. Travel Booking Assistant

🎯 Use Case

Allow users to:

- Search for flights and hotels
- Plan itineraries
- Book travel arrangements

💡 Flow Breakdown

1. **Greeting**
2. **Ask for travel type**: flight, hotel, or full itinerary
3. **Collect necessary details**: dates, destinations, number of travelers
4. **Use travel APIs** (like Skyscanner, Amadeus) to search flights/hotels
5. **Present top options**
6. **Add to travel itinerary/cart**
7. **Confirm & book**
8. **Store preferences (destinations, airlines, budgets)** in the Knowledge Graph

🔧 Tools to Add

- `FlightSearchTool` : Integrates with flight APIs
- `HotelSearchTool` : For hotel options based on destination and dates
- `ItineraryPlannerTool` : LLM + rule-based trip planner
- `TravelBookingTool` : Final booking or summary

How to Extend this solution to Other services – Product Market Place

🛍️ 2. Product Marketplace Assistant

🎯 Use Case

Support buying and selling of new or second-hand products.

💡 Flow Breakdown

1. **Greeting**
2. **Ask if user wants to Buy or Sell**
3. **For Buying:**
   - Ask for category (electronics, books, clothing)
   - Ask for filters (price range, condition, location)
   - Search using APIs (Flipkart, OLX, Amazon)
4. **For Selling:**
   - Collect product name, condition, price, and location
   - Generate a draft listing via LLM
   - Ask for confirmation
5. **Summarize cart/listing**
6. **Confirm or restart**

🔧 Tools to Add

- `ProductSearchTool` : Scrape or call APIs to search for products
- `ListingBuilderTool` : For LLM-based generation of seller posts
- `MarketplaceAPITool` : Integration layer to interact with platforms like OLX or Flipkart

Agent Orchestration workflow Extended

🔄 3. Shared Orchestration Strategy (Multi-Agent Router)

To manage multiple services like **food**, **travel**, and **marketplace**, a central **router agent** can classify the user's intent and forward the request to the appropriate domain agent.

🔗 📌 Intent Routing Logic

```
User Message
      |
      ▼
[ Intent Classifier ]
      |
  ┌──────────────────────────┐
  ▼                          ▼                          ▼
FoodOrdering          Travel                  Marketplace
   Agent               Agent                    Agent
```

```python
def route_intent(user_input: str) -> str:
    user_input = user_input.lower()
    if any(word in user_input for word in ["flight", "hotel", "travel", "trip", "itinerary"]):
        return "travel"
    elif any(word in user_input for word in ["buy", "sell", "product", "item", "market"]):
        return "marketplace"
    else:
        return "food"
```

🧠 Unified Multi-Agent Launcher

```python
intent = route_intent(user_message)

if intent == "food":
    response = food_ordering_agent.run(user_message)
elif intent == "travel":
    response = travel_booking_agent.run(user_message)
elif intent == "marketplace":
    response = marketplace_agent.run(user_message)
```

Knowledge Graph sharing Across Services

- 🎁 **Knowledge Sharing Across Domains**
- All domain agents share:
- 🔁 **Conversation Memory** – lets the assistant reference recent steps, across all services
- 🧠 **Knowledge Graph** – stores structured user data like:
  - preferred cuisines or restaurants (FoodBot)
  - favorite travel destinations or airlines (TravelBot)
  - product preferences or sell history (MarketplaceBot)
- This enables **cross-domain intelligence**, like:
- “Would you like to rebook your last trip to Goa and also reorder that Margherita Pizza from Pasta Street?”

Extending the workflow to other services

🧩 Modular Agent Structure

Each new service uses a similar modular structure:

```
agent/
  ├── food_agent.py
  ├── travel_agent.py
  └── marketplace_agent.py

tools/
  ├── location_tool.py
  ├── restaurant_search_tool.py
  ├── flight_search_tool.py
  ├── product_search_tool.py
  └── etc...
```

You only need to:

- Plug in service-specific tools
- Add state transitions for that service
- Reuse memory, conversation, and KG logic

✅ Benefits of This Design

Advantage	Description
🧠 Centralized Reasoning	Core logic (LLM prompts, memory) is shared across all agents
🧩 Plug-and-play Tools	Easy to register tools like `FlightSearchTool`, `ProductSearchTool`, etc
📈 Scalable	Add more domains (e.g., tutoring, finance) by registering new agents
🧠 Context Awareness	Cross-service suggestions and memory retention
🧪 Fast Prototyping	Each agent can be tested and deployed independently

Thanks