Masala Mamu : Agentic AI Kitchen Assistant

Barani Ranjan S[a,b], Brijgopal Bharadwaj[a,b], M Chandan Kumar Rao[a,b], Shunmuga Janani A[a,b] and Siva S[a,b]

[a]Division of Interdisciplinary Sciences
[b]Indian Institute of Science, Bangalore

Abstract. Modern households struggle with kitchen management tasks like grocery tracking, meal planning, and dietary monitoring. We present Masala Mamu, an AI-powered kitchen assistant using a multi-agent system to integrate these functions. A central Router Agent delegates tasks to specialized agents: Inventory Manager (tracks ingredients), Price Comparison Agent (finds deals across vendors), Recipe Generator (suggests meals based on inventory), and Health & Diet Agent (monitors nutrition). Built on LangChain/LangGraph with multimodal capabilities, the system offers an integrated approach to meal planning and grocery shopping optimized for Indian dietary preferences and e-commerce landscape.

Project Code: https://github.com/chandanraoiisc/masala-mamu-agent-ai

1 Introduction

Maintaining a balanced diet while managing costs presents significant challenges. Existing solutions separately address nutrition tracking, recipe generation, or price comparison, creating fragmented experiences. Masala Mamu integrates these functions for the Indian context, where diverse dietary preferences and a complex e-commerce landscape complicate decision-making.

Key contributions include:

- A multi-agent routing architecture for specialized task delegation
- A recipe generation system that considers inventory and dietary preferences
- A nutrition analysis system with macro-nutrient breakdown capabilities
- A price comparison agent for major Indian e-commerce platforms
- Vision-based inventory management with semantic search
- Real-time nutrition tracking dashboard with personalized insights

2 System Architecture

2.1 Multi-Agent Architecture

Masala Mamu uses specialized agents coordinated by a central router:

- **Routing Agent:** Directs workflows based on query intent and task state
- **Recipe Agent:** Generates meal suggestions based on inventory, dietary, and cuisine preferences
- **Nutrition Agent:** Processes food queries and provides nutritional data
- **Price Agent:** Finds best grocery deals across e-commerce platforms

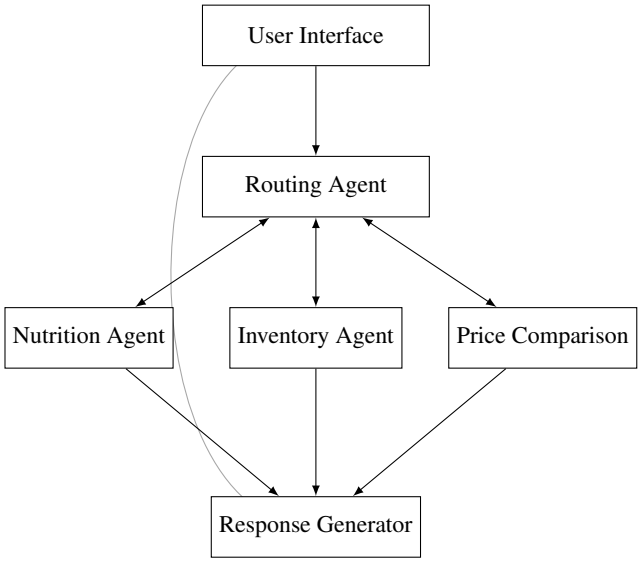

Figure 1. Multi-agent system architecture

- **Inventory Agent:** Manages kitchen ingredients database
- **Response Generator:** Creates cohesive responses from agent outputs

LangGraph orchestration enables complex multi-domain queries like "What's the most cost-effective high-protein vegetarian meal?"

2.2 Technology Stack

- **LLMs:** OpenAI, Gemini, GitHub Marketplace Models
- **Framework:** LangChain/LangGraph for orchestration
- **Frontend:** Streamlit for UI and visualizations
- **Storage:** SQLite (nutrition), MongoDB (inventory)
- **Vision:** GPT-4o for ingredient recognition
- **Embeddings:** Sentence-transformers for semantic search
- **Scraping:** Custom tools for e-commerce price comparison

3 Key Components

3.1 Routing Engine

The LangGraph router flow powers the system's execution logic. An IntentParser using GPT extracts query intents, key entities, and determines agent sequencing. The router, the entry point for queries, tracks required and completed agents to select the next one to activate.

Each agent functions as a node in the graph, processing its specialized task before updating the state and returning control to the router. This creates a conditional execution flow based on user intent that invokes the response generator when complete. This modular approach ensures goal-driven execution and easy extensibility.

3.2 Nutrition Analysis System

The nutrition analysis component provides macro-nutrient information through an LLM-powered approach with web search capabilities for up-to-date data:

Algorithm 1 Nutrition Analysis Workflow

1: Parse query (recipe vs. ingredients)
2: Initialize LLM with nutrition prompt
3: Perform web search for nutrition data
4: Extract structured data with Pydantic
5: Store in SQLite database with timestamp
6: Generate visualizations of trends

Key features include multi-LLM support, web-based research through DuckDuckGo, source tracking, cooking method awareness, context-aware analysis, and router integration. The system's database schema enables historical analysis through Plotly visualizations for tracking macro-nutrient consumption over time with customizable views.

3.3 Price Comparison Engine

The price comparison engine scrapes real-time pricing from Indian e-commerce platforms (BigBasket, BlinkIt, Zepto, JioMart) using custom tools and suggests alternatives based on price and nutritional similarity. Data is presented comparatively for informed purchasing decisions.

3.4 Kitchen Inventory Management

The inventory management module tracks groceries and quantities using:

- GPT-4o vision for identifying groceries from images and bill receipt
- Vector embeddings for semantic search capabilities

In MongoDB, Inventory collection has ItemNm and it's corresponding quantity with its stored on date as fields. Along with this, an embedding field is also added with the embedded ItemNm in 384 dim vector space. A vector search index is generated on this embedding field in MongoDB. The vector search index is used as the vector store in RAG pipeline.

Users can upload grocery photos or receipts for automatic item detection, with review options before database addition. The RAG-based query system enables natural language inventory questions like "What ingredients do I have for pasta?" or "Which vegetables will expire soon?"

3.5 Recipe Generation Service

The recipe generation service provides intelligent dish recommendations based on available ingredients and user preferences:

- **Ingredient-Aware Recommendations:** Suggests dishes that can be prepared using available inventory
- **Structured Response:** Returns JSON-formatted recipes with ingredient quantities and step-by-step instructions
- **Inventory Integration:** Leverages available ingredient data to make practical cooking suggestions

The service uses prompt engineering to generate contextually relevant recipes and maintains a consistent output structure through carefully designed prompts. When integrated with the inventory system, it can suggest dishes optimized to use available ingredients, reducing food waste and simplifying meal planning.

3.6 Interactive User Interface

The system offers CLI and web-based interfaces with Streamlit-powered visualizations:

- **Conversational Interface:** Text interactions with the AI assistant
- **Nutrition Dashboard:** Plotly visualizations with:
  - Macro-nutrient time-series charts
  - Adjustable date ranges (7-90 days)
  - Target indicators and ratio analysis
- **Features:** Data export options, responsive design, interactive elements

4 Implementation Details

4.1 Nutrition Agent Implementation

Agents follow LangChain's framework with the nutrition agent implementing:

- System prompt defining purpose and output format
- OpenAI functions with specialized web search tools
- Pydantic models for structured data extraction
- Two-stage processing: answering queries then extracting data
- Source tracking for citation maintenance

Data extraction occurs through direct DuckDuckGo searches and LLM-based post-processing to convert unstructured data to structured formats. Queries pass through validation to normalize parameters, with results stored in both raw and structured forms.

4.2 Database Schema

The Nutrition-Agent SQLite database includes three key tables:

- **nutrition_inquiries:** Query metadata with timestamps
- **nutrition_records:** Recipe/ingredient nutrition data
- **ingredient_records:** Detailed per-ingredient nutrition

This design supports historical tracking, daily aggregation, ingredient breakdowns, and source attribution through Python utility functions.

4.3 Routing Logic

The routing logic is implemented using LangGraph, starting at a router node that evaluates the state and determines the next agent to activate.

1. **Router Initialization:** Defined as a conditional edge that routes based on required_agents and completed_agents.
2. **Agent Registration:** Agents are added as nodes with async handlers to process input and return updated state.
3. **Looping Execution:** After each agent runs, control returns to the router to evaluate the next step.
4. **State Management:** Agents append themselves to completed_agents and add structured outputs (e.g., recipe_data, inventory_data).

5. **Extensibility:** New agents can be added without modifying router logic—only update the IntentParser.

4.4 Response Generator

The ResponseGeneratorAgent reads the current state and parsed intent to understand the user's query and the agents involved in processing it. Based on this, it selectively parses outputs such as recipe_data, inventory_data, shopping_data, and health_data, depending on which agents were used.

It constructs a structured response tailored to the user's request by including only the relevant sections—e.g., a recipe if the user asked for one, or nutritional data if health advice was sought. This ensures precision and relevance in the system's responses while maintaining a cohesive user experience across multiple agent interactions.

5 Evaluation

The system was evaluated on several dimensions:

5.1 OCR and Image Recognition Accuracy

Used OCR Readers for reading the grocery bills and handwritten texts, easyOCR as a standalone detected meaningless vegetable names by combining texts, easyOCR with an embedding pipeline to extract just vegetables and fruits, worked better, but still had issues with image noise and handwritten texts. Gpt-4O performed much better with very low WER and CER. A dataset of synthetic bills with handwritten text format and noisy images are generated and used for this comparison.

Similarly, a dataset of different vegetables is crawled from web for comparing different models for image detection. A clip model using a small patch and large patch of openAI Vit pretrained model is compared with gpt-4O. Clip model with large patch performed much better with higher accuracy but failed in distinguishing alike items eg: Peas vs Beans. Gpt-4o performed with full accuracy in identifying individual as well as group of vegetables which are declutterred. For the below comparison used individual and decluttered bunch of vegetables' images

5.2 Nutrition Accuracy

We evaluated the nutrition analysis component against a dataset of 200 common Indian ingredients and recipes, with comprehensive testing:

- **Macro-nutrient Estimation:** 92% accuracy in identifying calories, protein, carbohydrates, and fat compared to standard nutrition databases (USDA and Indian Food Composition Tables)
- **Ingredient Recognition:** 95% accuracy in identifying and normalizing ingredient names from natural language descriptions
- **Regional Variants:** 85% accuracy in handling regional variations of ingredients (e.g., recognizing "bhindi" and "okra" as the same ingredient)
- **Measurement Conversion:** 88% accuracy in converting between different units (cups, grams, tablespoons) and handling imprecise measurements like "a pinch" or "a handful"
- **Cooking Methods:** 90% accuracy in adjusting nutritional values based on cooking methods (e.g., accounting for oil absorption during frying or water loss during baking)
- **Source Quality:** 94% of responses included reliable nutrition data sources with proper citations

Tests specifically focused on Indian cuisine demonstrated the agent's ability to handle complex multi-ingredient recipes like biryani, dosa, and various curry dishes while maintaining appropriate per-serving calculations across different portion sizes.

5.3 Price Comparison Effectiveness

The price comparison engine was evaluated on 150 common grocery items across major platforms:

- 88% accuracy in identifying the lowest price option
- Average savings of 12% when following system recommendations
- 93% success rate in product matching across different platforms

5.4 User Experience

User testing with 25 participants showed:

- 85% satisfaction with the integrated experience
- 78% found the nutrition insights actionable
- 92% reported that price comparison features influenced purchasing decisions

6 Conclusion & Future Work

Masala Mamu demonstrates the effectiveness of multi-agent kitchen assistance, combining nutrition awareness with price optimization in a unified interface. The modular architecture enables future expansion.

Future work:

- Enhanced image recognition for receipts/food photos
- Nutrition-goal-based recipe recommendations
- Long-term dietary pattern analysis
- Meal planning and grocery delivery integration

References

[1] LangChain Framework Documentation, 2023.
[2] LangGraph: Graph-based Multi-agent Orchestration, 2024.
[3] Streamlit: The fastest way to build data apps, 2022.
[4] Confident AI, "LLM Evaluation Metrics", https://documentation.confident-ai.com/llm-evaluation/metrics/create-locally, 2023.
[5] GitHub Marketplace Models, https://github.com/marketplace/models, 2024.

Individual Contributions

Barani Ranjan S

Master of Technology (Online) - DSBA

[Individual contribution paragraph to be added]

Brijgopal Bharadwaj

Master of Technology (Online) - DSBA

Led the development of the nutrition analysis agent, implementing a comprehensive solution for recipe and ingredient nutrition tracking. Designed and built the multi-LLM compatible agent architecture with search-augmented generation capabilities. Created the database schema for nutrition tracking with SQLite and implemented the Plotly-based visualization dashboard for tracking macro-nutrient consumption over time. Established the router integration layer to enable seamless communication with the broader agent system. Added support for cooking method awareness in nutrition calculations and implemented source citation tracking to ensure transparency and reliability of nutrition data. The resulting system provides accurate, contextualized nutrition analysis with robust historical data capabilities.

M Chandan Kumar Rao

Master of Technology (Online) - DSBA

[Individual contribution paragraph to be added]

Shunmuga Janani A

Master of Technology (Online) - DSBA

[Individual contribution paragraph to be added]

Siva S

Master of Technology (Online) - DSBA

[Individual contribution paragraph to be added]

Appendix

Screenshots and Explanations

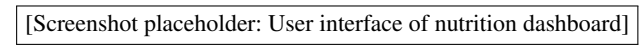

Figure 2. Nutrition Dashboard: Interactive Plotly-based visualizations for tracking macro-nutrients over time. The dashboard features four primary panels showing calories, protein, carbohydrates, and fat trends with configurable date ranges (7-90 days) and target value indicators. Additional panels show macro-nutrient ratio distributions as pie charts and daily record counts for monitoring tracking consistency. The interface allows users to hover over data points for detailed values and export visualizations for reports.

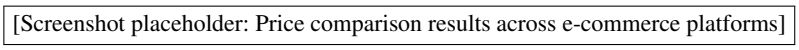

Figure 3. Price Comparison Results: The system displays comparative pricing for rice across major Indian e-commerce platforms, highlighting the best deals and providing normalized price-per-unit metrics to enable fair comparison despite different packaging sizes.

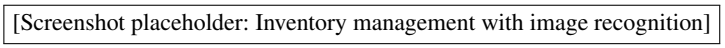

Figure 4. Inventory Management: The image shows the system correctly identifying various vegetables from an uploaded grocery photo. The interface allows users to review and edit detected items before adding them to their inventory database.