Web UI Copilots: Multi-Agentic RAG Implementation

CMPE 280 Hackathon Project

Executive Summary

This project presents an innovative multi-agent Retrieval Augmented Generation (RAG) web application developed for the CMPE 280 Web UI Copilots Hackathon at San Jose State University. The solution addresses NewWebCo's need for a productivity-enhancing web interface that leverages specialized AI agents to help employees access domain-specific knowledge efficiently. The implementation combines FastAPI orchestration, specialized knowledge retrieval, and interactive data visualization to create a seamless user experience that routes queries to the most appropriate AI agent automatically.

Problem Definition

NewWebCo, a leading provider of AI services to Fortune 500 companies, required a web-based solution that would allow employees to:

1. Access specialized knowledge across multiple domains (food security, clinical research, general information)
2. Retrieve accurate, relevant information without manually selecting the appropriate knowledge base
3. Visualize complex data for better understanding and decision-making
4. Save and reuse valuable information through a widget system

The challenge involved creating an intelligent orchestration layer that could:

- Analyze user queries to determine their domain
- Route queries to the most appropriate specialized agent
- Process responses for potential visualization
- Provide a clean, intuitive user interface

Technical Implementation

Multi-Agent Architecture

The core of the solution is a three-agent system with domain specialization:

1. **Food Security RAG Agent**:
    - Specialized in agricultural and food security knowledge
    - Uses food security PDF documents as knowledge base

    - Optimized for responding to queries about global nutrition, agricultural trends, and food accessibility
2. **Clinical RAG Agent**:
    - Specialized in medical and clinical research
    - Uses clinical PDF documents as knowledge base
    - Equipped to handle healthcare inquiries, medical research, and patient care information
3. **General Agent**:
    - Handles queries outside specialized domains
    - Supports data visualization and formatting
    - Accesses web search when necessary

Intelligent Query Routing

The system implements a sophisticated classification algorithm that:

- Analyzes keyword density in user queries
- Recognizes domain-specific terminology
- Identifies subject matter patterns
- Routes queries to the most appropriate agent without user intervention

Knowledge Retrieval System

The RAG implementation leverages:

- Vector embeddings of domain-specific documents
- Hybrid search combining semantic and keyword-based approaches
- LanceDB for efficient vector storage and retrieval
- OpenAI embedding models for high-quality text representation

FastAPI Backend

The FastAPI implementation provides:

- Central orchestration endpoint for processing all queries
- Pre-configured data widgets accessing World Bank APIs
- Widget management for saving and retrieving user information
- Visualization processing endpoints

Interactive User Interface

The web interface features:

- Clean, responsive design optimized for productivity
- Query input with automatic agent selection
- Response display with JSON detection and visualization

- Widget management for saving valuable responses
- Feedback mechanisms for continual improvement

Data Visualization Capabilities

The application offers multiple visualization options:

- Automatic detection of JSON data in responses
- Dynamic chart generation (line, bar, pie) based on data structure
- ASCII chart generation for text-based visualization
- Pre-configured widgets for global economic and environmental data

Implementation Highlights

1. **Agent Specialization**:

```
food_agent = Agent(
    name="food_agent",
    role="Food Security RAG specialist",
    model=OpenAIChat(id="gpt-4o"),
    description="Handles Food Security queries using knowledge from FAO reports...",
    knowledge=PDFUrlKnowledgeBase(
        urls=[FOOD_PDF_PATH],
        vector_db=LanceDb(...)
    ),
)
```

2. **Intelligent Query Classification**:

```
def classify_query(query: str) -> str:
    q = query.lower()
    food_matches = sum(1 for word in food_keywords if word in q)
    clinical_matches = sum(1 for word in clinical_keywords if word in q)

    if food_matches > clinical_matches:
        return "food"
    elif clinical_matches > food_matches:
        return "clinical"
    else:
        return "general"
```

3. **User Interface Implementation**:

```
<div class="query-panel">
  <label for="queryInput">Enter Your Query:</label>
  <textarea id="queryInput" rows="3" placeholder="e.g., Show global food security trends..."></textarea>
  <div class="btn-group">
    <button id="runBtn"><i class="fas fa-play"></i> Run</button>
    <button id="clearBtn"><i class="fas fa-trash-alt"></i> Clear</button>
  </div>
</div>
```

4. **Visualization Processing**:

```
function renderChart(chartData) {
  const ctx = document.getElementById('chartCanvas').getContext('2d');
```

```
    if (chartInstance) {
      chartInstance.destroy();
    }

    // Configure chart based on data type
    chartInstance = new Chart(ctx, {
      type: chartData.type || 'bar',
      data: {
        labels: chartData.labels,
        datasets: [{
          label: chartData.label || 'Values',
          data: chartData.values,
          backgroundColor: backgroundColor,
          borderColor: borderColor,
          borderWidth: 1
        }]
      },
      options: { /* Chart configuration */ }
    });
  }
```

Project Outcomes

The Web UI Copilots implementation successfully delivers:

1. **Seamless Knowledge Access**:
   - Users can ask questions in natural language without specifying domains
   - Automatic routing to specialized agents ensures accurate information retrieval
   - Domain-specific knowledge is augmented with web search when needed
2. **Enhanced Data Understanding**:
   - Automatic visualization of appropriate data
   - Multiple visualization formats for different data types
   - Quick access to common datasets through fixed widgets
3. **Improved Productivity**:
   - Widget system allows saving and reusing valuable information
   - Consistent interface across knowledge domains
   - Feedback mechanisms for continuous improvement
4. **Technical Excellence**:
   - Modern architecture using FastAPI, vector databases, and modern UI technologies
   - Responsive design that works across devices
   - Extensible framework that can incorporate additional specialized agents

Conclusion

The Web UI Copilots project demonstrates how intelligent agent orchestration can transform knowledge access in enterprise environments. By combining specialized RAG agents with automatic query routing and interactive visualization, the solution enables NewWebCo employees to efficiently access domain knowledge without needing to understand the underlying complexity.

The implementation fulfills all requirements of the CMPE 280 Hackathon, showcasing practical applications of cutting-edge AI technologies in a productivity-enhancing web interface. The architecture provides a foundation that can be extended with additional specialized agents and visualization capabilities as NewWebCo's needs evolve.