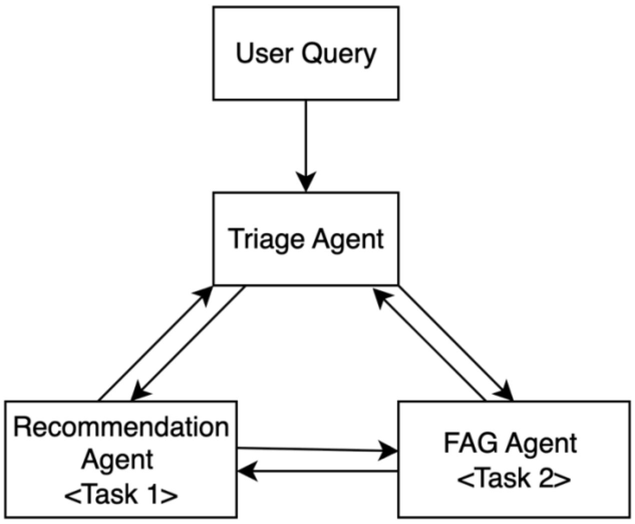
Overview:

Our customer service system uses a Triage Agent to direct queries to specialized Recommendation and RAG-powered FAQ Agents. Seamless handoff between these agents ensures continuous, appropriate support as user intents change during the conversation.

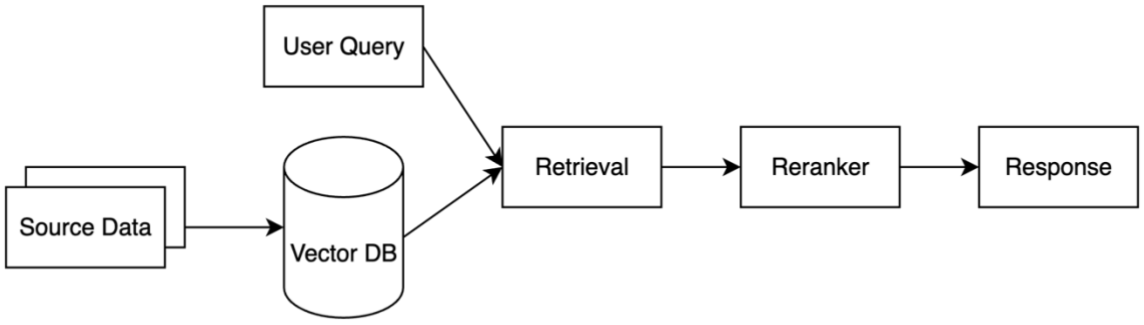


Detailed Task Implementations:

1. Task 1 : Custom Agent and Tool (Product Recommendation)

Tool : Telco Product/Roaming Plan Recommendation Engine

This tool aims to deliver personalized product and roaming plan suggestions, significantly saving customer time by simplifying complex offerings. For the business, it reduces reliance on human agents for routine queries, leading to operational cost savings, while improving customer satisfaction and retention through tailored experiences.



<u>Approach Implemented:</u>

1. Source Data: Mocked data representing a range of telco products, including prepaid and postpaid mobile data, SIM-only plans, and home broadband plans, was generated for initial implementation.
2. Vector Database: ChromaDB is utilized to store product names and descriptions, each assigned a unique ID for efficient retrieval
3. Retrieval: Given a user query, ChromaDB performs a similarity search against the vector database, returning the top_n most relevant products.

4. Re-ranking: A Large Language Model (LLM) is employed for a re-ranking step. This involves prompting the LLM to eliminate irrelevant products from the initial retrieval set and order the remaining products from most to least relevant.

Agent Integration: The product_reco tool is integrated into the recommendation_agent by defining it as a Python function, decorated with @function_tool to expose it to the agent's LLM with a clear name and description. This tool is then included in the recommendation_agent's list of available tools. This configuration empowers the LLM to intelligently determine when to invoke product_reco for specific recommendations, thereby enhancing its ability to fulfill customer requests effectively.

Possible Improvements:

- Personalization: Leverage historical customer usage data to provide a more personalized recommendations.
- Enhanced Query Retrieval: Implement more fine-grained retrieval through advanced intent extraction (e.g., identifying specific needs like "price," "data amount"). Applying metadata filters and structured queries can further refine retrieval results.
- Advanced Re-ranking: Integrate a dedicated cross-encoder re-ranker model after ChromaDB retrieval to achieve more precise and consistent relevance scoring, allowing the LLM to focus primarily on final response formatting.

2. Task 2 : Retrieval Augmented Generation (RAG)

The current implementation follows a traditional RAG approach for the FAQ agent.
Possible Improvement: Implement a feedback loop RAG where user feedback on agent responses is stored. For relevant historical queries, this method would boost the scores of previously well-rated documents and penalize poorly-rated responses, leading to continuous improvement in answer quality.

3. Task 3: Integration Strategy and Deployment Approach

Deployment Method: The primary deployment target is a web portal, ideally integrated as a chatbot within the existing website.

High-Level Deployment Architecture:

1. Frontend (Client-Side Web Application): The user-facing component, executed within the customer's web browser.
2. Backend (Server-Side Agent Application): Python-based scripts exposed as an API, responsible for orchestrating the core AI logic and agent interactions. This layer also manages the inter-agent handoff logic.
3. Data & External Services: Comprises data storage solutions (e.g., ChromaDB) and integrations with external APIs (LLM providers, Singtel's internal systems).

Challenges and Mitigation Strategies:

1. Data Security, Privacy, and Compliance:

    - Mitigation: Implement robust encryption for data both in transit and at rest to prevent unauthorized interception and ensure compliance with data protection regulations.
2. Scalability:
    - Mitigation: Employ load balancers to effectively distribute workloads across multiple servers. Utilize asynchronous processing to concurrently handle numerous requests. For long-term scalability, consider a cloud-native solution designed for elastic scaling.
3. Latency:
    - Mitigation: Given the use of LLMs with billions of parameters, sequential LLM calls, and vector searches, latency is a critical concern. To enhance efficiency, consider techniques like LLM quantization or utilizing lower precision data types during inference to reduce model size and accelerate response times.

<u>**Evaluation Metrics**</u>

1. Agent Performance

- Query Completion Rate: Percentage of user queries fully resolved by the agent bot.
- Intent Accuracy: Percentage of cases where the user's query intent is correctly identified.
- Tool Completion Rate: Percentage of successful responses generated by custom tools.
- Response Latency: Time taken for the agent to return a response.
- Cost per Conversation: Cost associated with LLM API calls per conversation.

2. Business Value

- Conversion Rate: Percentage of users who proceed to take a desired action (e.g., clicking on a recommended product, making a purchase) after interacting with the recommendation agent.
- Customer Satisfaction: Aggregate rating from post-conversation customer surveys.
- First Contact Resolution Rate: Percentage of user inquiries resolved during the initial interaction with the agent.