Post Discharge Medical AI Assistant (POC)

Submitted by: ***Sahim Kazi***

Date: 01-09-2025

Tools & Technologies: LangChain, Gemini-2.5-Flash, Streamlit, FAISS, Faker, Sentence-Transformers (all-MiniLM-L6-v2), JSON, Python

1. Introduction

The **Post Discharge Medical AI Assistant** is a multi-agent generative AI system designed to assist patients after hospital discharge. It provides personalized guidance, answers medical queries, and retrieves patient records using intelligent agent collaboration.
This Proof of Concept (POC) demonstrates practical implementation of **multi-agent orchestration, Retrieval-Augmented Generation (RAG)**, and medical data management with a simple yet effective web interface.

The system focuses on **nephrology (kidney care)** and uses dummy patient data for demonstration. It ensures seamless coordination between the *Receptionist Agent* and *Clinical Agent* while maintaining structured logging and secure data retrieval.

2. System Architecture

The system architecture consists of multiple modular components built using **LangChain** and **Gemini-2.5-Flash** as the central reasoning engine.

2.1 Agent Design

1. **Receptionist Agent**
    - Welcomes patients and collects their name.
    - Uses a **patient lookup tool** to fetch discharge reports from the JSON database.
    - Engages the patient with follow-up questions about medication, diet, and recovery.
    - Routes all medical-related questions to the Clinical Agent.
2. **Clinical Agent**
    - Handles all medical and clinical queries.
    - Uses **RAG (Retrieval-Augmented Generation)** to find accurate responses from the nephrology reference book.
    - Integrates with **FAISS** for vector similarity search.
    - Provides citations or context from retrieved documents for transparency.
    - Uses Gemini-2.5-Flash for reasoning and answer generation.

3. Implementation Details

3.1 Data Setup

- **Dummy Data Generation:**
  25+ post-discharge patient records were generated using the **Faker** library.
  Each record includes patient name, diagnosis, medications, dietary instructions, follow-up schedule, and warning signs.
- **Storage:**
  Patient data and logs are stored in structured **JSON** format for easy retrieval and updates.

3.2 Reference Materials

- A nephrology reference book (PDF) was processed, cleaned, and chunked.
- Text chunks were embedded using **all-MiniLM-L6-v2** embeddings and stored in a **FAISS vector store**.
- This enables semantic search to fetch relevant knowledge during query answering.

3.3 RAG Pipeline

1. Patient query is received by the Clinical Agent.
2. The query is embedded using sentence-transformers.
3. Similar chunks are retrieved from FAISS.
4. Retrieved context is passed to Gemini-2.5-Flash via LangChain for answer synthesis.
5. The response includes reasoning and citations from reference material.

3.4 Logging System

- A detailed logging mechanism captures:
  - Patient inputs and responses
  - Agent decisions and handoffs
  - Data retrieval attempts
  - Timestamped session logs

All logs are saved in JSON for transparency and debugging.

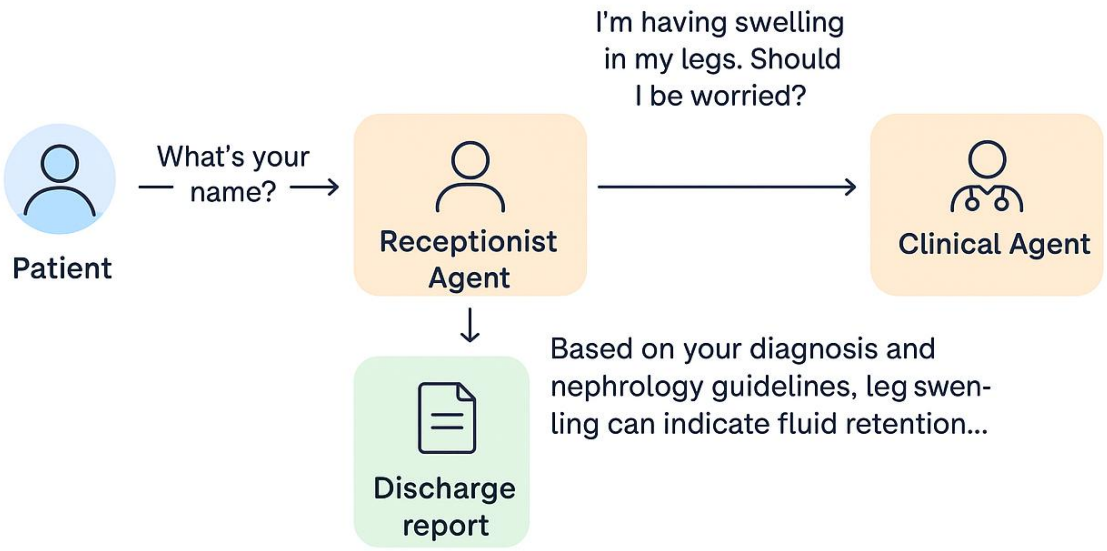
4. Frontend Interface

The frontend was developed using **Streamlit**, chosen for its simplicity and flexibility.
Features include:

- User-friendly chat interface.
- Input field for patient name and queries.
- Real-time conversation display with color-coded agent responses.
- Viewable logs of past interactions.

5. Workflow Summary

1. **Initial Interaction:**
   The system greets the patient and requests their name.
   Example: *"Hello! I'm your post-discharge care assistant. What's your name?"*
2. **Patient Lookup:**
   The Receptionist Agent retrieves the patient's discharge report using the name.
3. **Follow-up Questions:**
   The agent discusses medication, diet, or follow-up appointments.
4. **Medical Query Routing:**
   For health concerns, the query is forwarded to the Clinical Agent.
5. **Clinical Response:**
   The Clinical Agent performs RAG-based reasoning and provides an evidence-backed response.
6. **Logging:**
   Every step is logged for traceability and audit.



6. Key Features

- ✅ Multi-agent coordination (Receptionist + Clinical Agent)
- ✅ RAG-based knowledge retrieval using FAISS
- ✅ Semantic search with sentence-transformers
- ✅ Patient data management in JSON format
- ✅ Real-time chat via Streamlit
- ✅ Timestamped logging system
- ✅ Dummy data generation using Faker
- ✅ Integration with Gemini-2.5-Flash for intelligent responses

7. Results and Outcomes

The final POC successfully demonstrates:

- Contextual understanding of patient data and clinical conditions.
- Smooth handoff between agents based on query intent.
- Accurate retrieval of nephrology reference material for knowledge grounding.
- Functional web UI that allows real-time multi-agent interaction.

The project fulfills all required deliverables, including multi-agent architecture, RAG implementation, patient data retrieval, logging, and UI integration.

8. Architecture Justification

Component	**Choice**	**Justification**
LLM	Gemini-2.5-Flash	Fast, reasoning-capable model suitable for multi-agent dialogues.
Vector DB	FAISS	Lightweight, efficient vector store for semantic retrieval.
Embeddings	all-MiniLM-L6-v2	Compact yet semantically strong model for clinical text.
Framework	LangChain	Streamlines tool usage, agent orchestration, and RAG.
Frontend	Streamlit	Quick deployment and easy visualization.
Data Storage	JSON	Simple, flexible for small-scale POC.

9. Conclusion

This project successfully showcases a **Post Discharge Medical AI Assistant** powered by **multi-agent architecture and RAG**. It automates post-hospital follow-ups by combining **structured patient data**, **clinical knowledge retrieval**, and **intelligent reasoning**.

The system demonstrates practical applications of **LangChain, FAISS, and Gemini-2.5-Flash** in healthcare AI — emphasizing explainability, modularity, and educational safety.

Disclaimer: This system is for educational and research purposes only. It is not a substitute for professional medical advice.

Final Output