RAG-Based LLM for Radiation Oncology Patient Queries

December 6, 2024

Ayush Argawal
Jiayu Hu
Seth Kazarian
Grace Xie

Presentation Outline

i. Business Statement and Proposed Solution
ii. Architecture Overview and Components
iii. Evaluation and Analysis
iv. Deployment and Tools
v. Live Demonstration

i. Problem Context – Oncology Patient Questions

1. Radiation process involves high-energy X-rays, precision treatments, **technical information processes**.
2. Impact on patient well being: body image, sexual health, emotional well being.
    - **Patients lack accessible, empathetic information.**
    - **Clinicians face burnout, repetitive patient queries.**

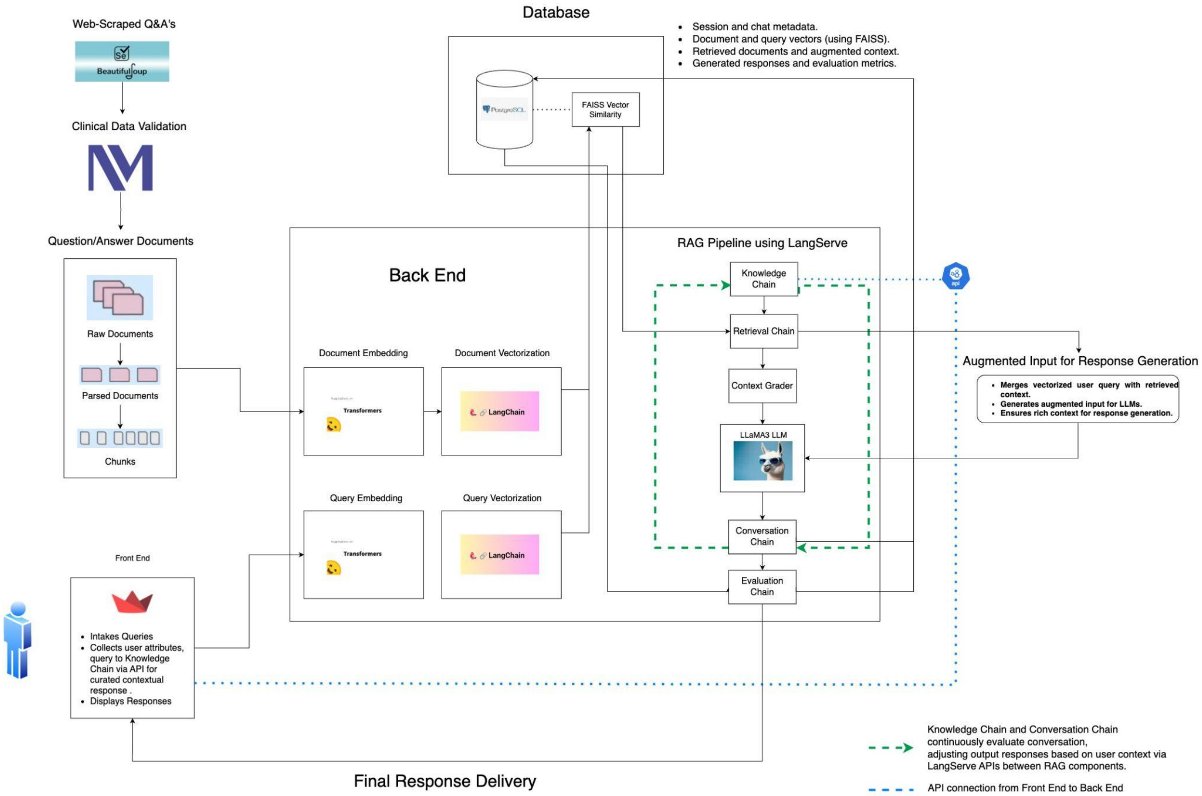
Solution: AI chatbot to complement clinical care, providing real-time, personalized responses.

i. Core Objectives and Features

Build RAG-based LLM chatbot: to provide personalized, accurate, and empathetic responses based on the patient questions.

- **Depth of RAG database:** Increase from 115 to >500 QA pairs
- **Dynamic Retrieval Pipeline:** Facebook AI Similarity Search for embedding-based similarity search.
- **LLM-Driven Response Generation:** LLaMA3.1 8B
- **Grading and Evaluation:** LangChain Grader Chain
- **User Profile Tailoring:** patient demographics for personalization.
- **Data Management System:** Store user queries, generated responses, evals, vector embeddings.
- **Feedback Loop:** Collection of user feedback to improve model performance.
- **Preprocessing Pipeline:** consistent embeddings, accuracy in retrieval, response.

ii. System Architecture Overview



Real-Time Query Processing: Supports fast vectorized retrieval and response generation within milliseconds.
Dynamic Retrieval with F.AI.S.S.: Top-K similarity search with fallback to general knowledge base for ambiguous queries.
Modular Microservices Architecture: Independent components to ensure scalability through lightweight architecture.
Personalized User Experience: Tailors responses based on user demographics: age, gender, and education level.
Evaluation and Feedback Loop: Scores responses for relevance, trustworthiness, and empathy. Stores clinician feedback for continuous refinement.

ii. Data Collection and Processing

Data Sources: Cancer.org, RTAnswers.org from 115 to >500 QA pairs.

Data Collection Process:

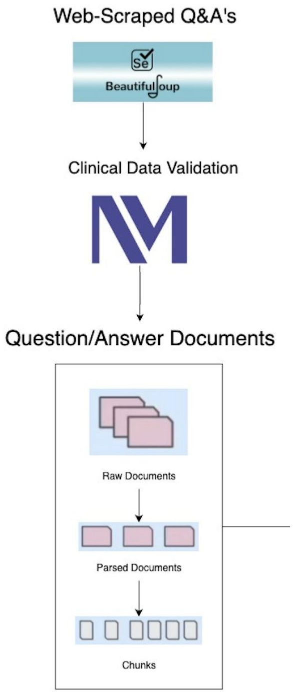
- Web Scraping: *BeautifulSoup* and for Q&A Extraction
- Text Cleaning: HTML tags, special characters, stop words.
- Normalization: lowercase for consistency.

Extraction:

- Use *Llama3.1:8b LLM* to generate directed questions from non-question format sources and validate from NU Medicine Team

Vectorization Pipeline:

- Two Vector Stores: 1. based on Question as Index; 2. based on Answers as Index
- Both Question and Answer are vectorized by *huggingFace*-"transformers/all-mpnet-base-v2"
- The vectorized Embeddings are stored in *Faiss Vector Store* locally



ii. Database

PostgreSQL database stores:

- User metadata and queries,
- Session conversation, chatbot generated responses
- Q&A dataset
- Evaluation scores for analysis.

Scalability for large interaction: Database supports multiple user sessions and real time data retrieval by engaging between backend chains and stored data.

- Embedding vectors stored as an extension to SQL database.
- Supports Facebook AI similarity search for relevance scoring.

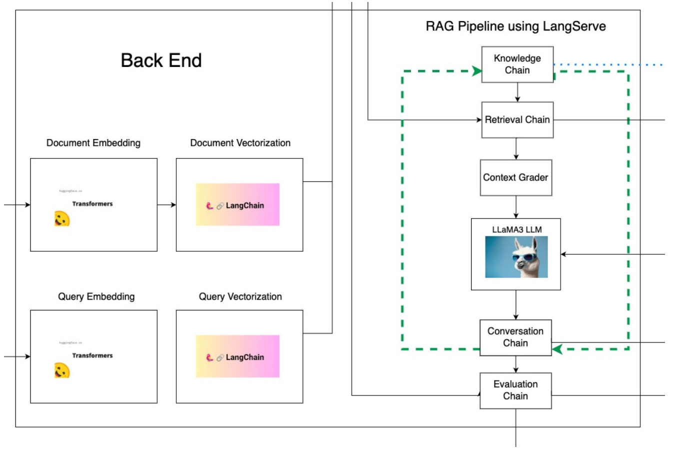
session			
INTEGER	id	PK	Auto-incrementing primary key
VARCHAR	username		Username of the session
INTEGER	age		Age of the user (nullable)
VARCHAR	gender		Gender of the user (nullable)
VARCHAR	disease_site		Disease site (nullable)
VARCHAR	education_level		Education level (nullable)

has many

session_chat			
INTEGER	id	PK	Auto-incrementing primary key
INTEGER	session_id	FK	Foreign key to session
TEXT	user_question		Original user question
TEXT	parsed_question		Parsed representation of the question
TEXT	response		System's response
BOOLEAN	is_verified		Whether the response is verified
FLOAT	retrieval_similarity		Similarity score (nullable)
FLOAT	retrieval_relevancy		Relevancy score (nullable)
JSON	response_eval_scores		Evaluation scores in JSON format (nullable)
JSON	response_analytics		Additional response analytics (nullable)

ii. Backend Design and Components

1. **Embeddings**: HuggingFace Transformers-sentence-transformers/all-mpnet-base-v2- create dense vector representations of text for both document and query inputs. Uses ClinicalBERT for embeddings of technical, domain-specific data to ensure medical accuracy.
2. **RAG Pipeline**: Combines knowledge retrieval, context grading, and LLM input to deliver precise and relevant responses.
3. **Evaluation Chains:** use predefined metrics (e.g., relevance, accuracy, empathy) and LLMs to systematically assess and ensure the quality, reliability, and domain alignment of generated responses.

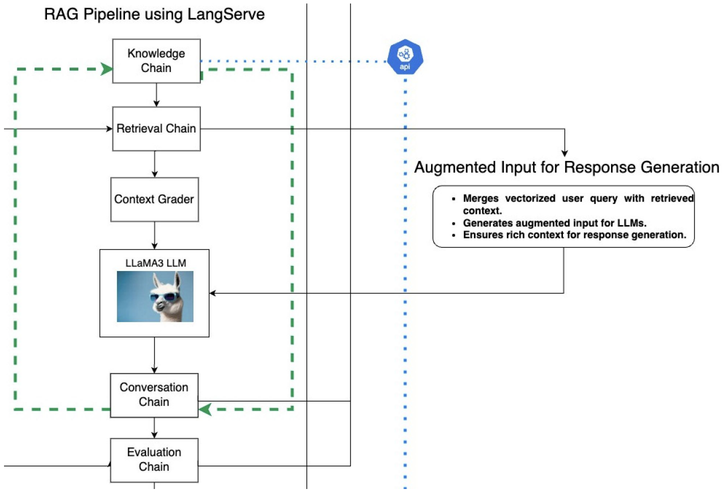


Lightweight architecture:

- Connecting the backend directly to the database.
- Connecting frontend to the backend (and not database) via API to ensure a scalable system.

ii. RAG Pipeline

Knowledge Chain: Learns about the user and integrates conversational context to ensure consistency and relevance in ongoing dialogue.
Retrieval Chain: Handles document retrieval by identifying and re-ranking the most relevant documents from the vector store based on relevance thresholds.
Augmentation: Merges user queries and retrieved documents to prepare rich contextual inputs for the LLMs.
Generation: Converts augmented inputs into fluent, informative responses using LLaMA3 LLM.
Context Grader: Evaluates the quality and trustworthiness of retrieved documents to refine the input for response generation.
Conversation Chain: Manages interactive dialogues, ensuring logical flow and leveraging user-specific insights for a tailored experience by working with Knowledge Chain.
Evaluation Chain: Scores responses for empathy, readability, completeness, and overall quality assurance.



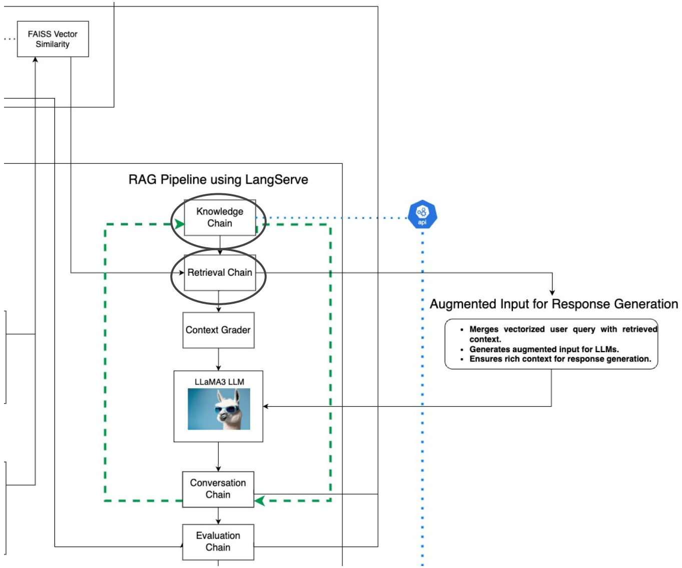
ii. Knowledge and Retrieval Chain

Knowledge Chain:

- **User Information Management:** Stores essential user details such as name, age, gender, education level, and cancer location.
- **Personalized Query Framing:** Generates personalized user queries by incorporating details like disease site and conversation history.
- **Dynamic Summary Creation:** Continuously updates a running summary of the conversation, capturing user inputs and chatbot responses.
- **Contextual Adaptation:** Ensures tailored responses by using stored user data (e.g., age, education level) to guide the LLM in crafting appropriate, context-sensitive replies.
-

Retrieval Chain:

- **Vector-Based Matching:** Retrieves the most relevant documents from the vector store using advanced vector similarity algorithms.
- **Dynamic Re-Ranking:** Ranks retrieved results based on relevance thresholds to prioritize the most applicable responses.
- **Pre-Processing for Context:** Prepares retrieved documents for downstream components like augmentation and response generation.

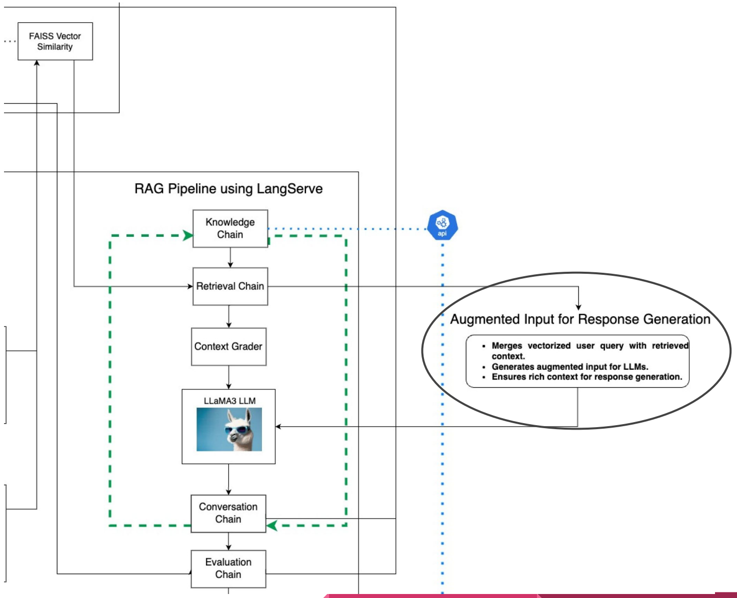


ii. Augmentation: Enriching Input for Precision

Context Integration: Combines vectorized user queries with retrieved documents to create a unified and contextually rich input.

Alignment Refinement: Tailors the input prompt to align with oncology-specific or general-purpose objectives, ensuring accurate and relevant responses.

Optimized Input for LLMs: Enriches retrieved context for seamless processing by next-stage LLMs, enhancing the precision and reliability of generated responses.



ii. Generation and Large Language Model Integration

LLaMa 3.1 (Primary LLM)

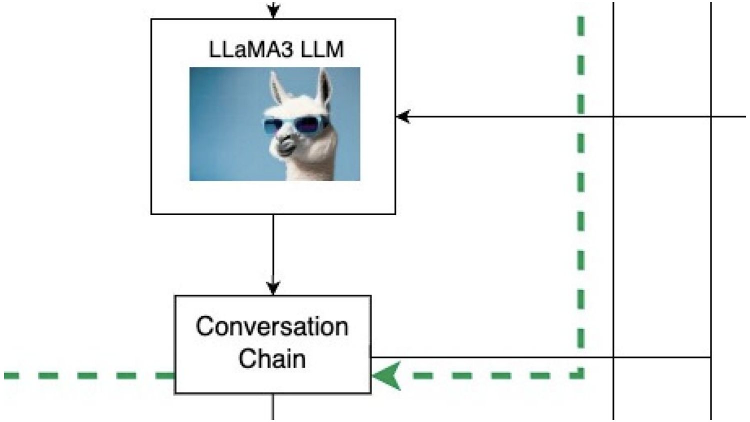
- **Purpose**: Handles all patient inquiries, ranging from general to highly specific oncology-related questions.
- **Capabilities**:
    - Ensures **multi-turn conversational continuity**, maintaining context across complex queries.
    - Balances technical depth and accessibility for diverse patient education levels.
    - **Adapts responses** to align with the system's contextual inputs from the Knowledge Chain and Retrieval Chain.

Augmented Input

- **Enhanced Contextualization**: Combines patient query, retrieved documents, and system-generated context to provide nuanced responses.
- **Optimized Precision**: Tailors outputs to address both technical and personal aspects of oncology queries.

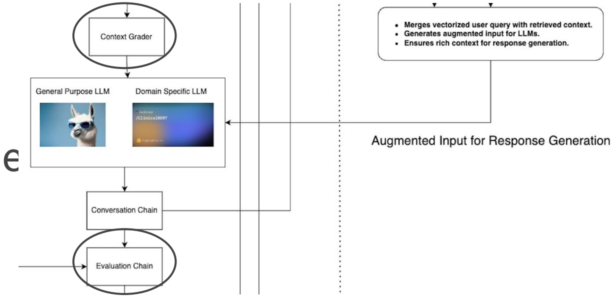
Storage

- LLaMa-generated responses pass through the **Conversation Chain** for consistency and are stored in **PostgreSQL**, enabling review, auditing, and iterative improvements.



ii. Context Grader and Evaluation Chain

Context Grader evaluates and ranks **relevance, accuracy and trustworthiness of retrieved documents**. Ensuring appropriate and high-quality content is selected for the response generation pipeline
Occurs along side Augmentation Phase



Evaluation Chain framework includes both **automated** and **human validation** processes to ensure that responses from the system are **highly reliable**.

- Relevance, trustworthiness, and empathy of responses by llama3.1:8b.
- Provides actionable feedback for continuous system improvement.
- Comparability of LLM-generated responses to authoritative data and experts for reliability.

1. Factual Correctness
2. Completeness
3. Conciseness
4. Coherence
5. Relevance
6. Potential Harm

ii. Streamlit User Interface

Query Input: Users can input queries related to radiation oncology. Interface validates input to avoid incomplete/irrelevant queries.

Collects Demographic Data: age, gender, and education level used to tailor responses for better understanding.

Displayed response includes retrieved context, LLM-generated answer, and evaluation scores: trustworthiness, relevance, empathy.

Feedback Mechanism allows the user to rate the quality and helpfulness of the response and stored for iterative improvement by researchers/developers.

iii. Robustness Analysis

Robustness testing covers diverse oncology-related queries derived from patient needs and clinical contexts, spanning 10 cancer types.

Focused on assessing the system's ability to handle patient-centric and technical questions.

Core Inquiry Domains:

1. **Technical Understanding:** Delves into treatment mechanisms and therapeutic interventions.
2. **Personal Impacts:** Covers quality of life concerns, side effects, and daily disruptions.
3. **Adaptation Needs:** Evaluates interest in recurrence possibilities and emerging therapies.

Strategic Insight: Patient questions indicate a deep need for nuanced, reliable, and empathetic information on their treatment journey.

Patient-Centered Information Dynamics - Robustness in Response Evaluation

Depth of Testing:

- Evaluated generated responses against metrics of **semantic similarity**, **key point alignment**, and **message consistency.**
- Utilizes a weighted scoring model (40% semantic, 30% alignment, 30% consistency).

Adaptive Query Handling processes responses to:

- Lexical and grammatical variations (e.g., typos).
- Contextual nuances specific to patient history and disease site.

Automated Evaluation Pipeline:

- Integrates **LangChain** and **Ollama LLM** with structured prompts for generating reliable and detailed evaluations.
- Ensures scalability and accuracy for large datasets of patient queries, enabling comprehensive robustness analysis.

iv. Backend Implementation and other connections

APIs and Modular Architecture

- Built with **FastAPI** to establish seamless and modular connections between components.
- Serves as the backbone for managing **sessions, chat interactions, and feedback loops.**

Key Connections

- **Front-End ↔ Back-End:**
  The front-end interacts with the RAG pipeline using **API calls for query submissions and response retrieval.**
- **Back-End ↔ Database:**
  The back end communicates with the PostgreSQL database via Facebook AI Similarity Search extension, to store and retrieve document and query embeddings efficiently.

LangServe and RAG Pipeline

- LangServe powers the RAG system by wrapping **LangChain chains as APIs,** ensuring dynamic communication across components.
- Facilitates **real-time updates of Q&A pairs** and context sharing between the Knowledge Chain, Retrieval Chain, and downstream models.

iv. Dockers Role in the RAG System

Backend Dockerization:

- Backend services, including the RAG pipeline, are containerized for scalability and portability.
- Configured with **Ollama Server to run LLMs efficiently within Docker**.

Database Integration:

- Database **mounted dynamically in Docker**, avoiding the need to copy database files during container build.
- Ensures faster build times and simplifies updates.

Optimized Backend Performance:

- Docker Compose manages multi-container applications, linking the backend, database, and APIs.

iv. Model Deployment and Configuration

Deployment Strategy:

- Modular deployment pipeline separates frontend, backend, and database components.
- Designed for both **on-premise and cloud-based scalability.**
- Hosted on **Northwestern Medicine's on-premise servers.**

Environment Configuration:

- Utilizes Docker containers to ensure **consistent and isolated environments** across development, testing, and production.
- Optimized deployment instructions created to align with institutional infrastructure.

Live Demonstration