Project Documentation: RAG Pipeline for Study Material Q&A

Contents

1 Introduction

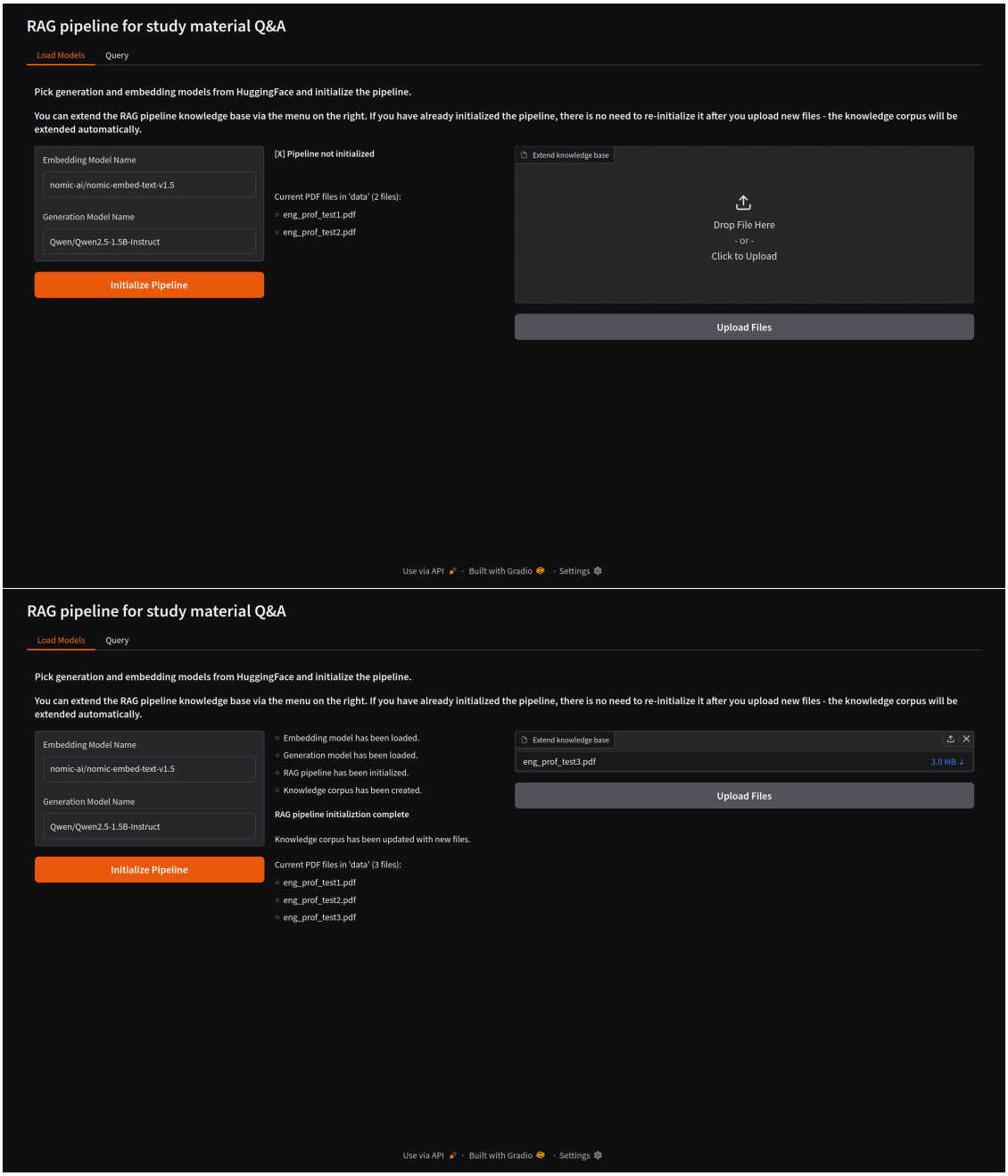
This document details the design and implementation of a Retrieval-Augmented Generation (RAG) pipeline system. The primary goal of this project is to provide users, particularly students, with a tool to ask questions about their study materials (in PDF format) and receive informative, contextually grounded answers. The system is able to leverage many of the modern open-source embedding and generative AI models, coupled with a user-friendly interface built with Gradio, to facilitate an effective Human-AI interaction experience.

2 User Interface (UI) Design

The UI is built using Gradio framework and is structured into two main tabs for clarity and ease of use.

2.1 "Load Models" Tab

This tab is dedicated to the initial setup of the RAG pipeline and possible extension of the knowledge base (uploading new exam PDFs).

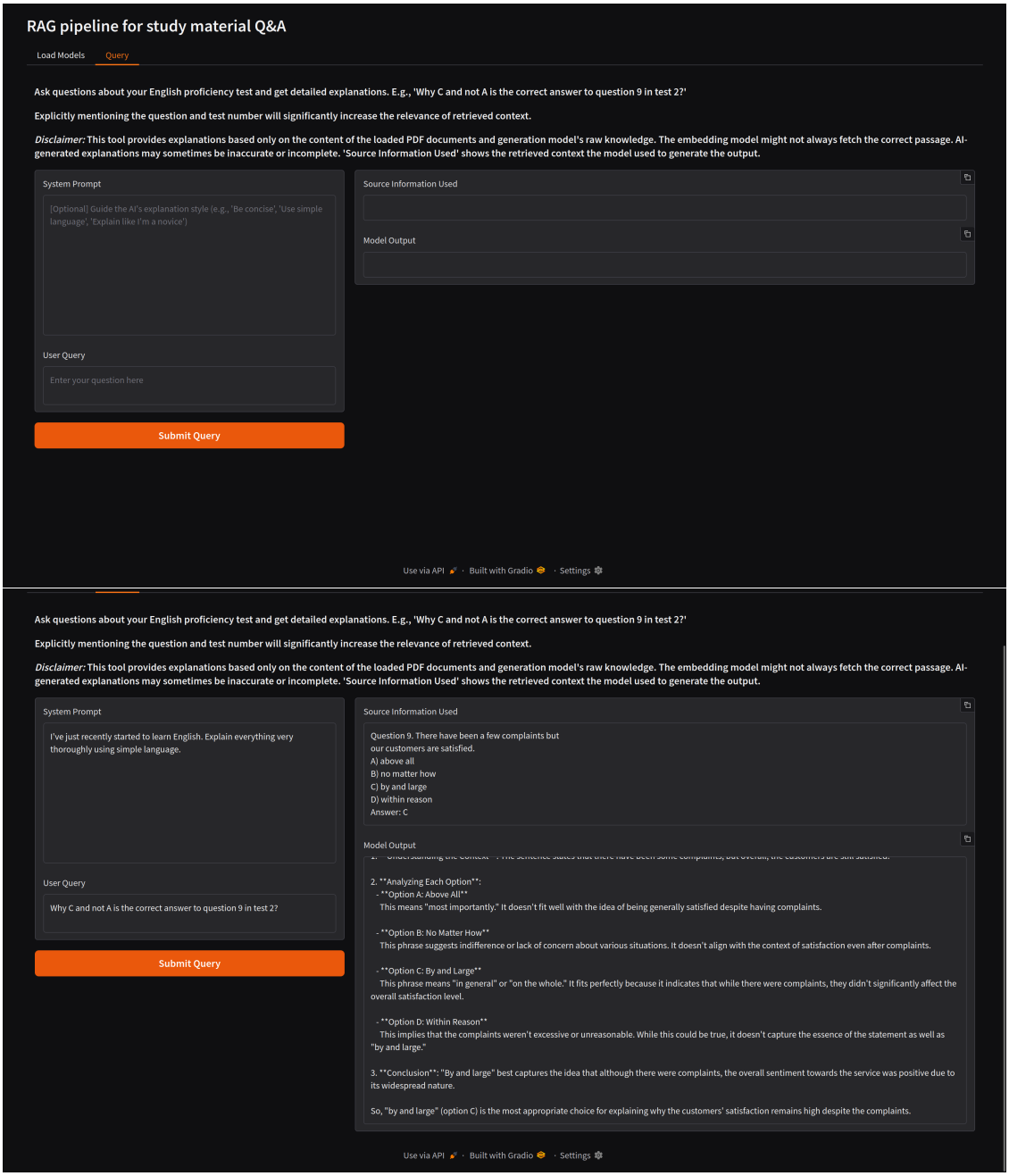


Design Choices & Rationale:

- **Model Selection Textboxes:** Offers users **control and freedom** (Lab 9: Nielsen's Heuristics, Microsoft HAX G17: Provide global controls) to select HuggingFace models. Defaults aid novice users, addressing **directability** (Lab 5: Interactive Attributes of AI).
- **"Initialize Pipeline" Button:** Clear call to action. Provides step-by-step **feedback** via pipeline state markdown (e.g., "Embedding model loaded," "Corpus created"), enhancing **visibility of system status** (Lecture 3: Norman's Principles, Lab 9: Nielsen's Heuristics, Lecture 9: Shneiderman's Prometheus Principles).
- **Pipeline State Markdown:** Delivers real-time **feedback** on initialization, managing user expectations and indicating progress (Lecture 9: Shneiderman's Prometheus Principles).
- **Corpus Files Display:** Provides **transparency** (Lab 5: Trust Enablers) on the RAG system's knowledge base, dynamically updated on file upload.
- **File Upload:** Standard Gradio PDF upload (file type is filtered). Allows users to extend the knowledge base, showing **user control**. "Upload Files" button updates corpus and file display, giving immediate **feedback**.

2.2 "Query" Tab

This tab is where users interact with the initialized RAG pipeline to ask questions.



Design Choices & Rationale:

- **Disclaimer Markdown:** Prominently displayed to **set the right expectations** (Google People+AI P2: Set the right expectations, Microsoft HAX G2: Make clear how well the system can do what it can do) regarding AI capabilities and errors - an ethical consideration against over-trust.

- **System Prompt Textbox:** Offers users **control** (Lab 9: Nielsen's Heuristics, Microsoft HAX G17: Provide global controls) to guide model behavior and tone, enabling personalization and aligning with the **social nature of explanations** (Lab 8: Miller's Properties).
- **User Query Textbox:** Clear input field for the user's question.
- **"Submit Query" Button:** Initiates RAG process. Clear call to action.
- **"Source Information Used" Textbox:** Critical for **interpretability** and **transparency** (Lab 5: Trust Enablers, Lab 8: Goal of XAI, Microsoft HAX G11: Make clear why the system did what it did). Shows AI-used evidence, allowing users to understand *why* an answer was generated and verify grounding. Aligns with **selective nature of explanations** (Lab 8: Miller's Properties).
- **"Model Output" Textbox:** Displays AI-generated answer. **Output streaming** improves perceived performance and offers continuous **feedback** (Lecture 3: Norman's Principles, Lecture 9: Shneiderman's Prometheus Principles), reducing wait time and enhancing engagement.

3 Core Functionality & Design Choices

3.1 Model Selection and Loading

Users can specify HuggingFace model names (embedding, generation), catering to experimentation and varying hardware. Loading includes error handling (`ModelLoadingError`) and UI feedback. Quantization (`BitsAndBytesConfig`) for the generation model manages resources (backend feature).

3.2 Knowledge Corpus Creation and Management

The system processes PDFs from `DATA_DIR`.

- **Document Processing:** `PDFLoader` extracts text, `DocumentTaskSplitter` uses regex for structured passage segmentation. Metadata (source, page, test/question numbers, indices) is stored.
- **Deterministic IDs:** SHA256 hashes of passage content/source prevent duplicates (Google People+AI P5: Invest early in good data practices).
- **Corpus Update:** UI file uploads re-invoke `create_knowledge_corpus`, extending ChromaDB. This provides **user control** and ensures AI learns from new data (Microsoft HAX G13: Learn from user behavior, in terms of knowledge corpus).

3.3 Query Processing and Context Retrieval

- **Metadata Filtering:** Extracts test/question numbers from queries to filter ChromaDB, improving retrieval relevance and **efficiency** (Microsoft HAX G4: Show contextually relevant information).
- **Context Window:** Backend parameters (`context_window_backward`, `context_window_forward`) in `RAGPipeline.query` allow `DocumentProcessor.retrieve_task` to fetch passages with surrounding context for the LLM.

3.4 Text Generation and Output Streaming

The generation model uses the query concatenated with retrieved `passage_context`.

- **System Prompt Integration:** User-defined system prompt (with defaults) guides LLM response style.
- **Streaming Output:** `TextIteratorStreamer` streams generated text to UI, providing immediate **feedback** and improving UX with incremental results.

3.5 Error Handling and User Feedback

- **Custom Exceptions:** Specific error handling and logging via custom exceptions (e.g., `ModelLoadingError`, `GenerationError`).
- **UI Feedback:** User-facing errors use non-technical `gr.Error` messages. Status updates manage expectations during long operations, aligning with **visibility of system status** and **helping users recover from errors** (Lab 9: Nielsen's Heuristics).

4 Application of Human-AI Interaction Design (HAIID) Principles

This project consciously incorporates several HAIID principles discussed throughout the course:

- **Transparency and Interpretability (Lab 5: Trust Enablers, Lab 8: Goal of XAI, Microsoft HAX G11: Make clear why the system did what it did:**
  - The most significant feature supporting this is the **"Source Information Used"** display. It allows users to see the exact text passage(s) the AI used for generation, making the AI's reasoning (retrieval part) transparent and the output interpretable.
  - Displaying the names of the loaded models and the list of corpus files also contributes to system transparency.
  - The default chat template in `TransformersGenerationModel` ensures a structured interaction even if the chosen model lacks one.

- **User Control and Freedom (Lab 9: Nielsen's Heuristics, Microsoft HAX G17: Provide global controls):**
  - Users control the choice of embedding and generation models.
  - Users control the knowledge corpus by uploading PDF documents.
  - Users formulate their own queries and can provide system prompts to guide the AI's tone and style.
  - The ability to dismiss or ignore AI suggestions (though not explicitly "dismissing" an answer, users can simply ask a new query) is partially supported.
- **Feedback and Visibility of System Status (Lecture 3: Norman's Principles, Lab 9: Nielsen's Heuristics, Lecture 9: Shneiderman's Prometheus Principles):**
  - The UI provides ongoing feedback:
    - Status messages during model initialization and corpus creation.
    - Streaming of the AI's generated output.
    - Clear display of retrieved context and the final answer.
    - Error messages through `gr.Error` when issues occur.
  - This keeps the user informed about "what is happening" and "what just happened".
- **Setting Expectations and Accountability (Microsoft HAX G1: Make clear what the system can do, G2: Make clear how well the system can do what it can do, Google People+AI P2: Set the right expectations, P4: Be accountable for errors):**
  - The disclaimer in the "Query" tab explicitly states that the AI can make mistakes and its output may not always be accurate. This helps **"Make clear how well the system can do what it can do"** (HAX G2) and makes the system **"accountable for errors"** (Google P4).
  - By showing the source context, the system provides a basis for users to hold the AI **"accountable"** to its retrieved information.
- **Human-Centered Design (Lab 8, Lecture 3):**
  - The system is designed to solve a specific user need (Q&A for study materials).
  - The UI aims to be intuitive for this target task.
  - The focus is on augmenting the user's ability to find information rather than full automation without oversight. The system fits the **"Human-centric AI system"** variant (Lecture 2), where the user is in control and the machine provides context-specific information.
- **Ethical Considerations (Lab 5: Trust Enablers):**

  - The disclaimer and transparency features contribute to ethical use by managing expectations and allowing verification.
  - The system (as a local prototype) inherently respects data privacy for uploaded documents. If deployed, explicit privacy measures would be paramount (Google People+AI P7: Be transparent about privacy and data settings).
  - While not actively mitigating bias in the LLMs themselves, the RAG approach of grounding answers in user-provided documents can reduce reliance on the LLM's general, potentially biased, knowledge. The quality and neutrality of the uploaded corpus become critical.

- **Managing AI's Uncertainty and Complexity (Lab 7):**
  - **Capability Uncertainty:** Addressed by allowing users to select specific models and by the RAG architecture itself, which constrains the LLM's task. The disclaimer also acknowledges performance unpredictability.
  - **Output Complexity:** Handled by the RAG framework, which aims to produce relevant and grounded (less complexly "creative") outputs based on retrieved text.

- **Match Between System and Real-World (Lab 9: Nielsen's Heuristics):**
  - The Q&A format is a natural way for users to seek information.
  - UI labels and instructions use generally understandable language.

- **Accessibility (General HCI Principle, HAIID:**
  - The project leverages the Gradio framework, which aims to adhere to fundamental web accessibility standards for its core UI components.
  - The UI provides inherent support for screen readers, basic keyboard navigation, and color themes.

5 Information Visualization Techniques

While the project does not employ complex charts or graphs in the traditional InfoViz sense, it uses several techniques to visualize information for the user:

- **Textual Display of Information:** The primary method.
  - `Source Information Used`: Presents the retrieved context as a block of text, allowing users to read and understand the basis of the AI's answer.
  - `Model Output`: Displays the AI-generated answer, also as text. Streaming makes this a dynamic textual visualization.
  - `Pipeline State Markdown`: Uses text to convey the status of system processes.
  - `Current PDF files`: A list format to show the contents of the knowledge base.

- **Layout and Grouping (Gradio UI):**
  - Tabs ("Load Models," "Query") visually separate distinct stages of interaction.
  - Within tabs, related elements are grouped (e.g., model inputs, query inputs, outputs), aiding **recognition rather than recall** (Lab 9: Nielsen's Heuristics).
- **Implicit Visualization of Process:** The sequence of interactions and feedback (e.g., uploading a file and seeing the "Current PDF files" list update) visually communicates the system's internal processes to some extent.

6 Lessons Learned

Developing this RAG-based Q&A system underscored several key lessons in Human-AI Interaction Design:

1. **Explainability is the Cornerstone of Trust in RAG:** Simply providing an answer is insufficient. The ability to display the `Source Information Used` proved fundamental, transforming the AI from a "black box" into a verifiable assistant and directly addressing core XAI needs. Users can only truly trust and learn from explanations they can trace back to source material.
2. **Proactive Expectation Management is Crucial:** LLMs are constrained by their knowledge. While developing and testing different embedders and retrieval pipelines, the system sometimes provided very inaccurate results, which had to be iteratively worked on. The upfront disclaimer managing expectations about accuracy and knowledge boundaries was vital in fostering appropriate reliance on the system.
3. **Clear Visibility and User Control Mitigate AI Opacity:** Providing continuous feedback on system status (model loading, corpus updates, errors via `gr.Error`) and ensuring users control key inputs (models, documents, queries) significantly improves usability and reduces the feeling of interacting with an unpredictable or uncontrollable agent.
4. **Iterative Design is Key for Effective HAIID Implementation:** Translating abstract HAIID principles into concrete, user-friendly UI elements required iterative refinement. Initial ideas for displaying status or context often needed adjustments to truly enhance clarity and ease of use from the end-user's perspective.