**Project Overview**

The goal of this project is to develop a Retrieval-Augmented Generation (RAG)-based system to guide users through the U.S. visa application process. The system simplifies complex visa-related queries by retrieving relevant information from a corpus of official U.S. visa documents and generating clear, user-friendly responses using advanced language models.

**Key Objective:**

* Assist users in understanding various visa types, application steps, required documentation, and interview procedures for U.S. visa applications.

**Data Sources**

The system utilizes the document **“United States Visa System: Information for Experts from the People’s Republic of China,”** published by the American National Standards Institute (ANSI).

* **Corpus Size:** 2 PDF document related to US Visa System.
* **Content Summary:**
  + Visa types and classifications (e.g., B1, B2).
  + Visa application procedures and documentation requirements.
  + Interview preparation guidelines.
  + Initiatives for expedited visa processing.
  + U.S. consular district details.
  + Visa classification summary tables and appendices.
* **Storage and Access:**
  + The content is processed using **PyPDF2** and stored in a **FAISS** index for efficient retrieval.

**Chatbot Functionalities**

1. **Understanding Visa Requirements:**
   * Detailed explanations of U.S. visa types, eligibility criteria, and their purpose (e.g., B1 for business, B2 for tourist).
   * Information on requirements like proof of ties to the home country and financial stability.
2. **Application Process Guidance:**
   * Provides step-by-step instructions for completing the application, paying fees, scheduling interviews, and attending interviews.
   * Covers essential documentation, common errors, and best practices to avoid mistakes.
3. **Document Preparation:**
   * Lists required documents and highlights common mistakes (e.g., mismatched names, missing biometric details).
   * Provides advice for ensuring complete and accurate documentation.
4. **Business Travel Facilitation:**
   * Answers questions about expedited visa programs for businesses, including formal facilitation and group appointments.

**System Functionalities and Capabilities**

1. **Retrieval and Generation:**
   * The system uses **FAISS-based semantic search** to retrieve relevant content from the U.S. visa corpus.
   * Retrieved content is processed by the **Meta-Llama-3-8B-Instruct** model, generating concise, user-friendly answers based on the query.
2. **Semantic Search:**
   * **FAISS** computes cosine similarity to retrieve contextually relevant document chunks, while **BM25** re-ranking ensures keyword relevance, combining semantic and lexical matching.
3. **Multimodal Question Answering:**
   * The **Meta-Llama-3-8B-Instruct** model generates accurate responses for complex queries by synthesizing relevant content from retrieved documents.
4. **Customization:**
   * Users can configure the granularity of query results (e.g., kkk), balancing the level of detail and relevance in the retrieved content.

**Models Used**

1. **Embedding Models:**
   * **Model:**

**Sentence-Transformers' all-MiniLM-L6-v2** is used to generate dense vector embeddings for text chunks.

* + **Storage:**

Embeddings are stored in a **FAISS** index for fast retrieval based on cosine similarity.

1. **Language Models:**
   * **Model:**

**Meta-Llama-3-8B-Instruct** (14.96 GB model) is used to generate structured, coherent responses based on the retrieved context.

* + **Quantization:**

The model is quantized to 8-bit, reducing memory usage without sacrificing performance, ensuring efficient execution on standard GPUs.

**Retrieval Methods**

1. **FAISS (Facebook AI Similarity Search):**

**FAISS** retrieves the top k most relevant document chunks based on semantic similarity, enhancing system responsiveness even with large datasets.

1. **BM25:**

**BM25** is applied after **FAISS** retrieval to re-rank results by keyword relevance, ensuring precise matches for specific visa-related queries.

1. **MMR (Maximal Marginal Relevance):**

**MMR** ensures diversity in the retrieval by penalizing redundant information and selecting the most relevant, diverse chunks for query answering.

**Key Parameters**

1. **Chunk Size:**
   * Documents are split into 1000-character chunks with a 50-character overlap, preserving semantic coherence and context for more accurate retrieval.
2. **Retrieval Depth (kkk):**
   * The system allows fine-tuning of retrieval depth (typically k=5), ensuring an optimal balance between relevance and detail in the retrieved content.
3. **Quantization:**
   * To optimize memory usage, the language model is quantized to 8-bit precision, reducing the memory footprint while maintaining high response quality.
4. **Model Configuration:**
   * The language model is configured to run efficiently on available hardware using **device\_map="auto"** for dynamic resource allocation.

**Preprocessing Steps**

1. **Text Extraction:**

**PyPDF2** is used to extract text from the 27-page PDF document. The content is concatenated into a single string for further processing.

1. **Chunking:**

The extracted text is split into overlapping chunks of 1000 characters with a 50-character overlap to maintain context across chunks.

1. **Embedding Generation:**

**Sentence-Transformers' all-MiniLM-L6-v2** model generates dense vector embeddings for each text chunk, which are stored in a **FAISS** index for efficient retrieval.

**Prompt Template**

1. **System Prompt:**

The system prompt ensures that the model only responds based on the given context: "You are an expert in U.S. visa systems. Provide accurate and relevant information only from the given context."

1. **Initial Reply:**

The initial reply prompts the user to input a query: "Please provide the question and extracted text for which I need to produce structured answers."

1. **Dynamic Prompt Formatting:**

The prompt is dynamically formatted with relevant retrieved context, ensuring consistency across different retrieval methods (MMR or BM25) and query types.

It prepares the prompt by appending retrieved documents for context. It creates a formatted string by combining the user's query with the retrieved document snippets, ensuring the model receives relevant and focused information to generate an accurate response.

**Retrieval Settings**

1. **K-Top Retrieval:**
   * The top k most relevant results are retrieved using **FAISS**, with the parameter **λ = 0.5** balancing relevance and diversity in the **MMR** method.
2. **BM25 Re-ranking:**
   * After **FAISS** retrieves the top k results, **BM25** refines them based on keyword relevance, ensuring precise results for detailed queries.
3. **MMR (Maximal Marginal Relevance):**
   * **MMR** adjusts retrieval by penalizing redundant information and selecting diverse, relevant chunks using cosine similarity scoring.

**Advanced Techniques**

1. **Contextual Compression:**
   * Recursive text splitting creates overlapping chunks, preserving context and semantic integrity across text segments.
2. **Multi-stage Retrieval:**
   * **FAISS** is used for initial retrieval, followed by **BM25** re-ranking to enhance accuracy by combining semantic and keyword-based relevance.
3. **Dynamic Prompt Adjustment:**
   * The system adjusts prompts based on query type and retrieval method, ensuring the LLM receives the most relevant context for optimal performance.

**Performance Metrics**

1. **Retrieval Efficiency:**

**FAISS** retrieves relevant chunks in approximately 0.7 seconds, and **BM25** re-ranking takes 0.9 seconds, ensuring efficient and precise retrieval.

1. **Model Performance:**

The model generates responses within 300 tokens and avoids hallucination when provided with clear context.

1. **Memory Usage:**

The quantized 14.96 GB model operates efficiently on a 16 GB GPU, ensuring smooth processing of large datasets and complex queries.

**Reproducibility**

1. **Platform:**

The system runs on **Google Colab**, offering an accessible and GPU-accelerated environment for the entire pipeline.

1. **Preprocessing Steps:**

Text extraction with **PyPDF2** and chunking with **NLTK** are fully documented, ensuring reproducibility across different platforms.

1. **Quantization Details:**

The model is optimized for deployment on low-memory systems, including those with 16 GB GPUs, ensuring efficient fine-tuning and inference.