Infosys Internship Program

SwiftVisaAI Based Visa Eligibility Screening Agent

AI-Powered Multi Country Visa Assistant using RAG

Final Internship / Milestone Submission Report

Submitted by: Nishanth M

Roll No/ID: manjnishanth.04@gmail.com

Mentor Name: Siddarth

Project : SwiftVisaAI Based Visa Eligibility Screening

Duration: 3 months

December 26, 2025

Contents

Abstract

This project presents the design and implementation of an AI-powered multi-country visa information assistant using the **Retrieval-Augmented Generation (RAG)** architecture. The system enables users to ask visa-related questions for multiple countries such as the USA, UK, Canada, Australia, and Schengen countries. Instead of relying solely on large language models, the system first retrieves relevant official visa information using similarity search techniques and then generates accurate answers using Google Gemini. This approach significantly reduces hallucinations, improves reliability, and ensures context-aware responses.

1 Introduction

Visa application processes vary across countries and involve complex rules, documentation requirements, and eligibility criteria. Applicants often rely on scattered online resources, which can be outdated or misleading. With the rise of AI chatbots, users expect quick answers; however, generic chatbots may generate incorrect or hallucinated information.

2 Problem Statement

- **Scattered Information:** Visa information is dispersed across multiple websites and documents.
- **Unreliable AI Responses:** Users often receive inconsistent or incorrect answers from generic AI tools.
- **Hallucinations:** Large Language Models may hallucinate when factual grounding is missing.
- **Need for Control:** There is a need for a controlled, country-specific, and reliable visa query system.

3 Objectives

The key objectives of this project are:

1. To build a visa query system supporting multiple countries.
2. To implement Retrieval-Augmented Generation (RAG) for factual grounding.
3. To reduce hallucination by restricting answers to retrieved context.
4. To minimize API usage and cost using local similarity search.
5. To design a modular, scalable, and professional application structure.

4 Scope of the Project

- Covers visa-related queries for USA, UK, Canada, Australia, and Schengen countries.
- Supports document-based question answering.
- Can be extended to additional countries and visa types.
- Future integration with a web interface or mobile application is possible.

5 Tools and Technologies Used

5.1 Programming & Libraries

Python, NumPy, Scikit-learn, dotenv.

5.2 AI&NLPTechnologies

Google Gemini API, Retrieval-Augmented Generation (RAG), TF-IDF Vectorization, Cosine Similarity, Prompt Engineering.

6 System Architecture

The system follows a modular RAG-based architecture:

1. **User Input:** User asks a visa-related question.
2. **Retriever Module:** TF-IDF and cosine similarity identify the most relevant content chunk.
3. **Context Selection:** The best-matching chunk is selected.
4. **Generator Module:** Google Gemini generates the final answer using only the retrieved context.

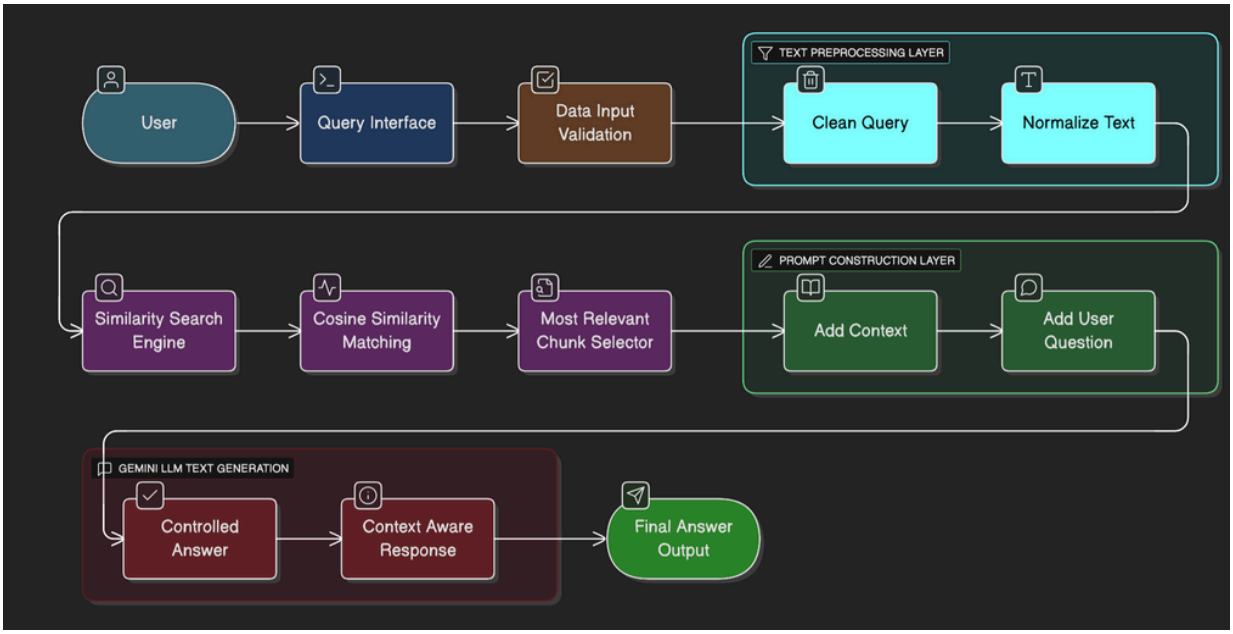


Figure 1: High-Level System Architecture

7 Implementation Details

7.1 DataPreparationandChunking

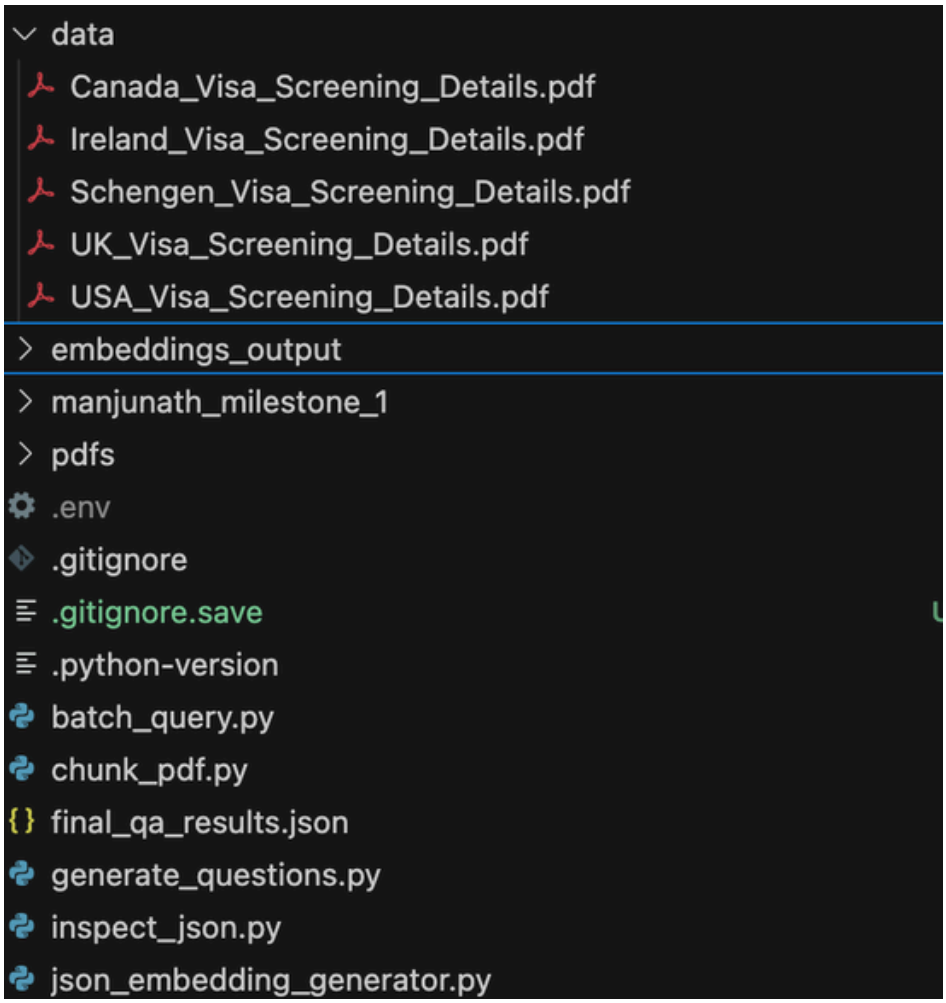
Visa documents were collected and split into smaller, meaningful text chunks. Chunking helps in retrieving only relevant information instead of entire documents.

8. Project Folder Structure

The application is organized into a clean and professional structure to ensure maintainability and scalability.

Directory Breakdown:

- visa-rag-application/: Root directory.
- app/: Contains source code modules (config, loader, retriever, generator).
- data/: Stores country-specific text chunks.
- run.py: Entry point for the application.



9 Embeddings and Similarity Search

Instead of using online embedding APIs, TF-IDF vectorization was used locally to compute text similarity. Cosine similarity identifies the most relevant chunk for a given query.

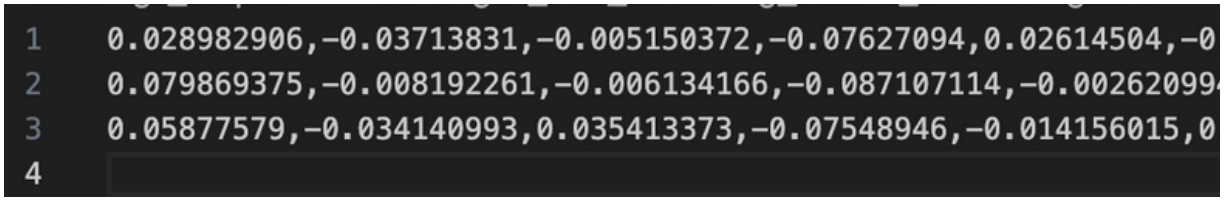


Figure 2: Embeddings

10 Multi-Country Support

Each country has its own data directory containing chunks and embeddings. The user selects the country at runtime, making the system flexible and scalable.

11. Challenges Faced

- API Quotas: Gemini API quota limitations during initial embedding generation attempts.
- Compatibility: Model compatibility issues with deprecated Gemini versions.
- Data Structure: JSON structure mismatches while loading embeddings.
- Configuration: Environment variable and API key configuration errors.

12. Solutions Implemented

- Local Embeddings: Switched from online embeddings to TF-IDF similarity to bypass quota limits.
- Security: Centralized API configuration using dotenv for secure key management.
- Refactoring: Modularized codebase for easier debugging and maintenance.
- Validation: Verified supported Gemini models using model listing before deployment.

13. Results and Output

The system successfully answers visa queries for 5 countries. It generates context-aware and accurate responses while preventing hallucinations by restricting answer generation to the retrieved context.

14. Visa streamlit agent

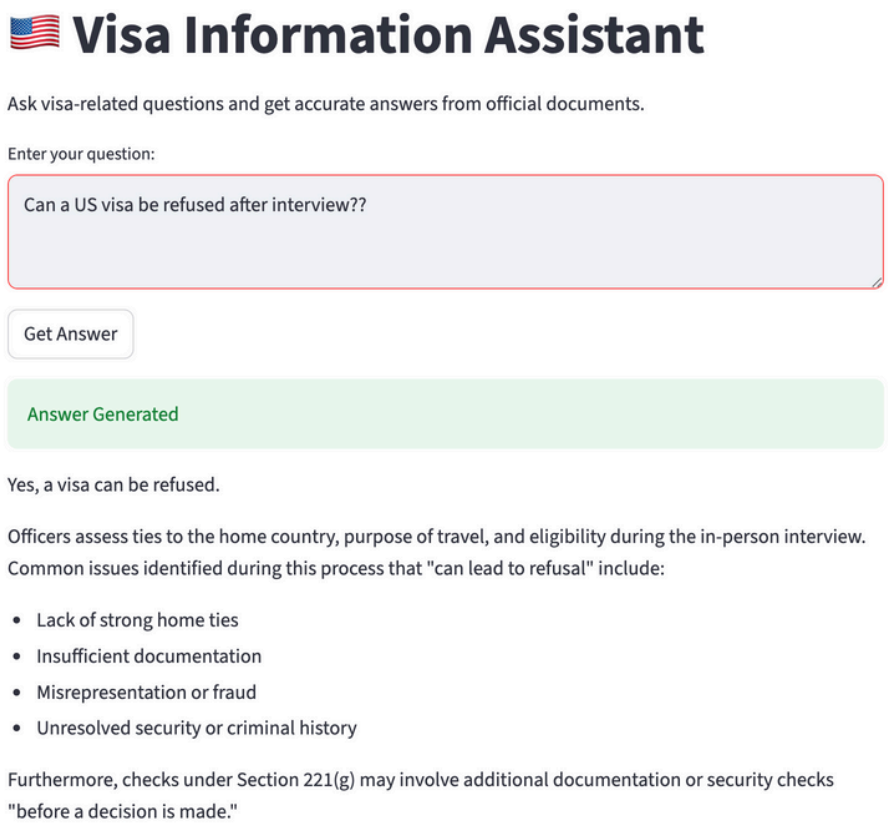
Key Features & Functionality:

- **Conversational AI Interface:** Uses a natural language chat system, allowing users to ask questions about specific visa types or requirements and receive context-aware responses.

- **Requirement Checker:** Helps travelers instantly identify necessary documentation, such as passport validity, proof of funds, or specific forms, based on their destination and nationality.

- **Tailored Recommendations:** Analyzes user-provided details to suggest the most appropriate visa route, whether for business, tourism, or long-term stay.
- **Real-Time Updates:** Continuously references the latest official guidelines to ensure users receive accurate and currently valid information.

- **User-Friendly Design:** Built on the Streamlit framework, providing a responsive and aesthetic layout that eliminates the need for navigating complex government portals.



8 Conclusion

This project successfully demonstrates how Retrieval-Augmented Generation (RAG) can be used to build reliable AI applications for real-world use cases. By combining traditional NLP techniques with modern generative AI, the system ensures accuracy, efficiency, and scalability.

Declaration

I hereby declare that this project is my original work carried out under the guidance of my mentor and has not been submitted elsewhere for evaluation.

Signature: Nishanth M **Date:** 06 - 01- 2026