SwiftVisa: AI-Based Visa Eligibility Screening Agent

1. Objective

The objective of this project is to build a Large Language Model (LLM)-powered visa eligibility screening agent that evaluates whether a user may qualify for a given visa type, based entirely on structured user inputs and a pre-curated policy knowledge base. All reasoning is performed by a generative AI model grounded in immigration policy documents using Retrieval-Augmented Generation (RAG).

This system is intended to assist users in understanding their visa eligibility across different countries and visa types without relying on rule-based systems, translation APIs, OCR, or scraped content.

2. Data Sources

2.1 Visa Policy Knowledge Base (Vector Store)

- Official visa eligibility guidelines collected manually from government sources (e.g., USCIS, IRCC, gov.uk)
- Stored in a vector database (e.g., FAISS or Chroma) after text extraction and embedding
- Structured by visa type and country for targeted retrieval during inference

2.2 User Inputs

- Structured personal and professional details submitted through a web form: ○ Age, Nationality, Education, Employment, Income, Visa Type

3. Project Outcome

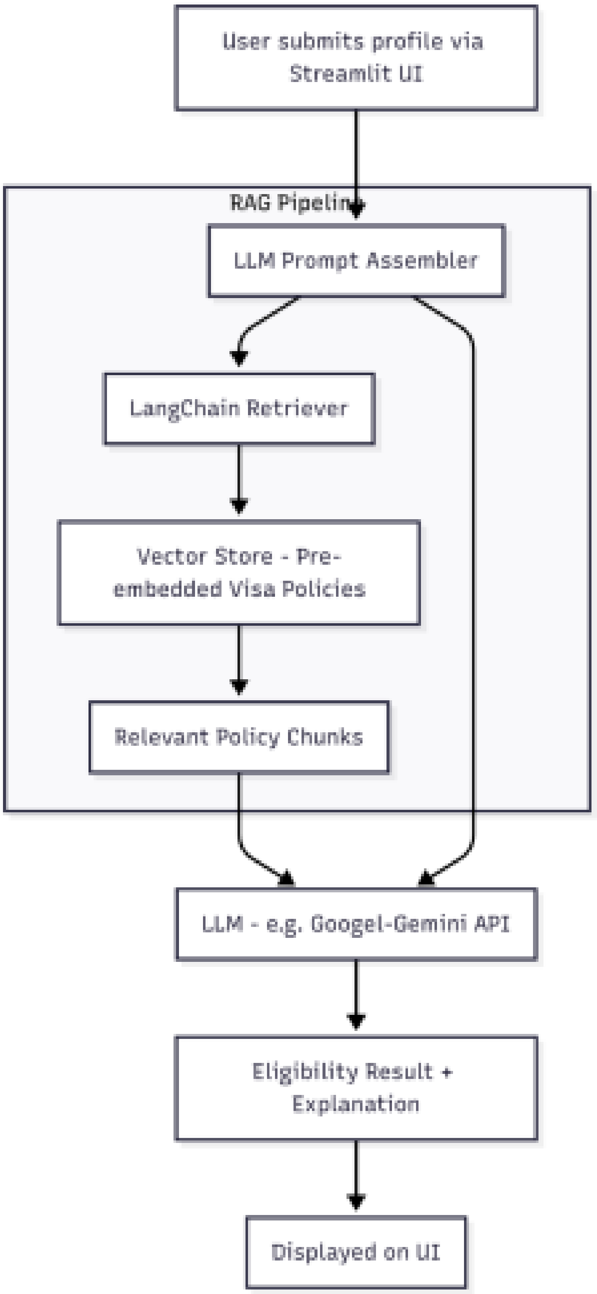
- An AI-based system that simulates a visa eligibility officer using natural language reasoning.
- No rule-based filters — all decisions are generated dynamically by the LLM. • A pre-built, searchable vector store of immigration policies used to ground model responses.
- A user-friendly interface deployed online.

4. Project Workflow

• Define target visa types and manually collect relevant policy documents • Process and embed these documents into a vector database • Design and prompt the LLM to evaluate eligibility using retrieved context • Build a web UI for user data input

• Deploy and document the application

5. System Architecture


User submits profile via Streamlit UI
RAG Pipeline
LLM Prompt Assembler
LangChain Retriever
Vector Store - Pre-embedded Visa Policies
Relevant Policy Chunks
LLM - e.g. Googel-Gemini API
Eligibility Result + Explanation
Displayed on UI

6. Week-wise Module Implementation and Milestones

Milestone 1: Week 1–2

Module: Research, Design & Policy Corpus Preparation

• Finalize supported visa categories and eligibility-related fields •
Manually collect immigration documents for each visa type
• Extract text, clean, and chunk documents
• Create embeddings using SentenceTransformer or OpenAI
• Store in FAISS or Chroma

Deliverables:

- Cleaned, embedded policy document store
- Document index organized by visa type and country
- Initial LLM prompt for eligibility screening

Milestone 2: Week 3–4

Module: RAG + LLM Pipeline

• Implement retrieval chain using LangChain or custom RAG logic •
For each query, retrieve top-K policy chunks from the vector store •
Construct prompt using user profile + retrieved content
• Generate eligibility response and explanation via LLM
• Add confidence score and document citations if applicable

Deliverables:

- Working RAG+LLM pipeline
- Eligibility outputs with explanations grounded in policy
- Logged decision history with response quality tracking

Milestone 3: Week 5–6

Module: User Input Flow

• Build Streamlit form for structured input (age, visa type, country, etc.) •
Set up session state and pass user input to the backend for inference

Deliverables:

- Fully functional frontend with dynamic form fields
- User input successfully integrated with the RAG+LLM backend

Milestone 4: Week 7–8

Module: Deployment

• Deploy to Streamlit Cloud or Hugging Face Spaces •
Write documentation, final report, and record demo

Deliverables:

- Fully deployed application

- GitHub repository with all code and README
- PDF final report and video walkthrough

7. Evaluation Criteria

Week	Evaluation Metrics
2	Quality and completeness of vector store (visa policies)
4	Functional RAG + LLM pipeline
6	Complete UI and integration with backend
8	Successful deployment and presentation clarity

8. Tech Stack

Component	Tools / Libraries
Frontend	Streamlit
Backend	LangChain / FastAPI
LLM	GPT-4, Qwen2, Mistral via LM Studio
Vector Store	FAISS / Chroma
Embeddings	SentenceTransformers / OpenAI Embeddings
Deployment	Streamlit Cloud / Hugging Face / Render