Final Project Proposal

Project Title: Ask KSA

Submission Track: NLP/LLM

1. Project Overview

Idea: A RAG-powered assistant that helps expatriates navigate Saudi government procedures (visas, iqama renewal, labor laws) by answering questions from official documents, with voice support for accessibility.

Key Features:

Document Q&A: Query PDFs from sources like Absher or Ministry of Labor.

Multilingual Support: Arabic/English answers (NLLB/Google Translate API).

Voice Interface: Speech-to-text + text-to-speech.

Source Citations: Answers reference official documents (e.g., "Ministry Circular No. XYZ, Page 12").

2. Rationale & Market Relevance

Problem:

1. KSA's 10M+ expats struggle with complex procedures and language barriers.
2. Existing solutions usually lack accuracy for legal and regulatory procedures.

Why This Project?

1. **High Impact**: Directly addresses Vision 2030 goal of improving expat experiences.
2. **Technical Innovation**: Combines RAG with LLMs for Arabic/English for precision and fluency, respectively.
3. **Gap in Market**: No open-source tools tailored to KSA's legal documents with voice support.

Existing Solutions:

1. Government portals (Information is scattered, limited FAQs).
2. Generic chatbots.

3. Workflow & Methodology

Overall Approach:

1. Ingest PDF → 2. Extract text/sections → 3. Embed chunks → 4. Store in Vector → 5. Retrieve Info + Text generation → 6. Display the answer with references

Tech Stack:

PDF Processing → PyMuPDF (text + metadata extraction)

Embeddings → BAAI/bge-small (optimized for Arabic/English)

Voice → WhisperSpeech (Speech to text) and Chat TTS (Test to Speech)

LLM → Not Decided

Key Steps:

Data Pipeline:

Scrape/upload official PDFs → PyMuPDF extraction.

RAG Setup:

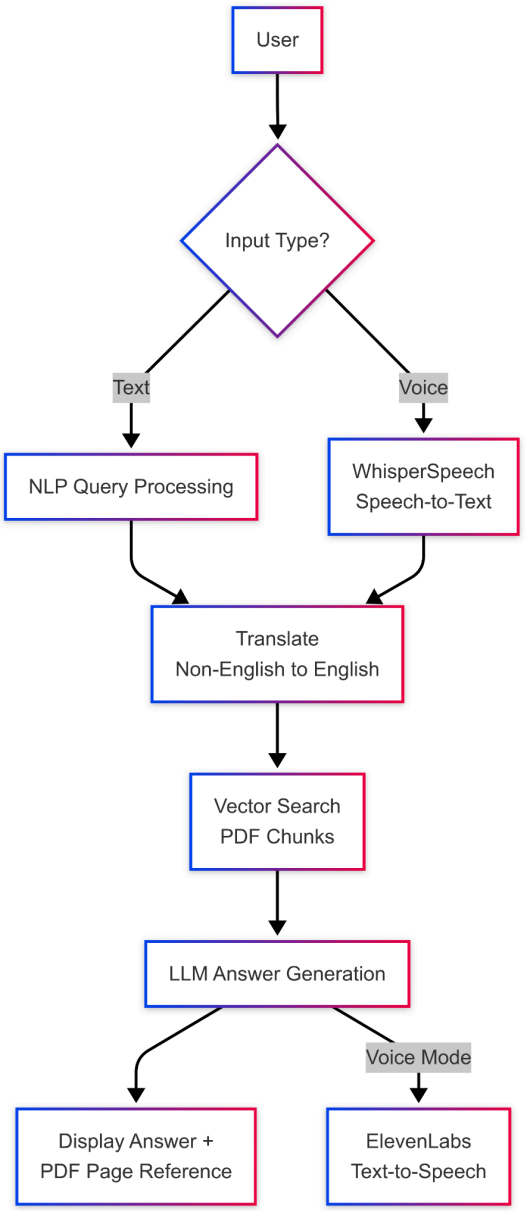
Generate embeddings.

Query Handling:

Multilingual input → Translate → Retrieve → Generate answer → Display + cite sources.

4. ERD & Workflow

System Diagram



Step-by-Step Workflow

User Input:

A user interacts with the system via text (typed question) or voice (spoken question).

Input Routing:

The system checks the input type:

1. Text: Directly sends to the NLP pipeline.
2. Voice: Converts speech to text using Whisper (Speech-to-Text/STT).

Translation (If Needed)

Non-English queries, like Arabic, are translated to English using Google Translate/NLLB for consistent processing.

Retrieval (RAG Core)

The translated text is converted to embeddings (numeric representations) using BAAI/bge-small.

Searches the vector database (pre-loaded with PDF chunks) to find the most relevant document sections.

Answer Generation

The retrieved chunks + user query are fed to an LLM to generate a human-like answer.

The LLM cites specific PDF pages/sections

Output Delivery

Text Mode: Displays the answer + references in the UI (Streamlit/Gradio).

Voice Mode: Uses ElevenLabs (Text-to-Speech/TTS) to read the answer aloud.

5. Stretch Goals

1. **Telegram Bot**: Deploy for beta testing in expat communities.
2. **Proactive Alerts**: Notify users of procedure changes (e.g., "New iqama fee announced!").
3. **OCR for Scanned PDFs**: Tesseract integration.

Why This Proposal Stands Out

Niche Focus: Targets a critical, underserved audience (KSA expats).

Technical Rigor: Uses RAG, multilingual NLP, and voice—all covered in bootcamp.

Deployment-Ready: Docker + FastAPI aligns with MLOps modules.

Measurable Impact: Metrics include precision@3 and user feedback.

Submission Tip: Use **visuals** (like the flowchart above) to save space in the 2-page limit.

Next Steps:

1. Draft proposal in PDF (use Canva for visuals).
2. **Start small**: Test PyMuPDF on one Absher PDF.
3. Submit early for feedback!

This proposal balances **innovation, feasibility, and social impact**—perfect for standing out. Let me know if you'd like help refining the visuals! 🚀

P.S. Need a template? Here's a Google Docs link to adapt this structure.