PoliBot - AI-Powered Policy Document Q&A Assistant (POC)

**Overview**

**PoliBot** is a Proof-of-Concept (POC) application designed to provide intelligent question-answering over large and complex policy documents. It uses the power of **Large Language Models (LLMs)**, **semantic search**, and **chunked document embeddings** to provide precise answers with references from uploaded documents.

This solution is built using **Python**, **LangChain**, **FastAPI**, **Milvus Vector DB**, and a **React Material UI** frontend.

**Problem Statement**

Organizations such as school boards, government bodies, or public communities manage numerous lengthy policy documents. Searching and retrieving relevant information during meetings or decision-making processes becomes challenging and time-consuming.

**Solution Approach**

* Load PDF/DOCX policy documents.
* Chunk documents into meaningful segments.
* Generate embeddings using LangChain + Google Generative AI.
* Store embeddings in **Milvus Vector DB**.
* Use semantic similarity search to fetch relevant chunks.
* Generate answers using Gemini or similar LLMs.
* Display user-friendly output in a modern React UI with document viewer support.

**Tech Stack**

**Backend (Python + FastAPI)**

* **FastAPI**: Web framework for REST APIs.
* **Pydantic**: For data validation and schema definition.
* **Uvicorn**: ASGI server for FastAPI.
* **python-dotenv**: Manage environment variables.
* **LangChain**: Framework for building LLM-powered applications.
* **langchain-community**: Community components and tools for LangChain.
* **langchain-google-genai**: Gemini model integration for LLM responses.
* **pymilvus**: Python SDK for Milvus Vector DB.
* **unstructured**: For extracting content from PDFs and DOCX files.
* **python-docx**: Handling and reading Microsoft Word files.
* **python-multipart**: File upload handling in FastAPI.

**Frontend (React + Material UI)**

"dependencies": {

"@emotion/react": "^11.14.0",

"@emotion/styled": "^11.14.0",

"@mui/icons-material": "^7.0.1",

"@mui/material": "^7.0.1",

"framer-motion": "^12.6.3",

"pdfjs-dist": "^5.1.91",

"react": "^19.1.0",

"react-dom": "^19.1.0",

"react-doc-viewer": "^0.1.14",

"react-scripts": "5.0.1"

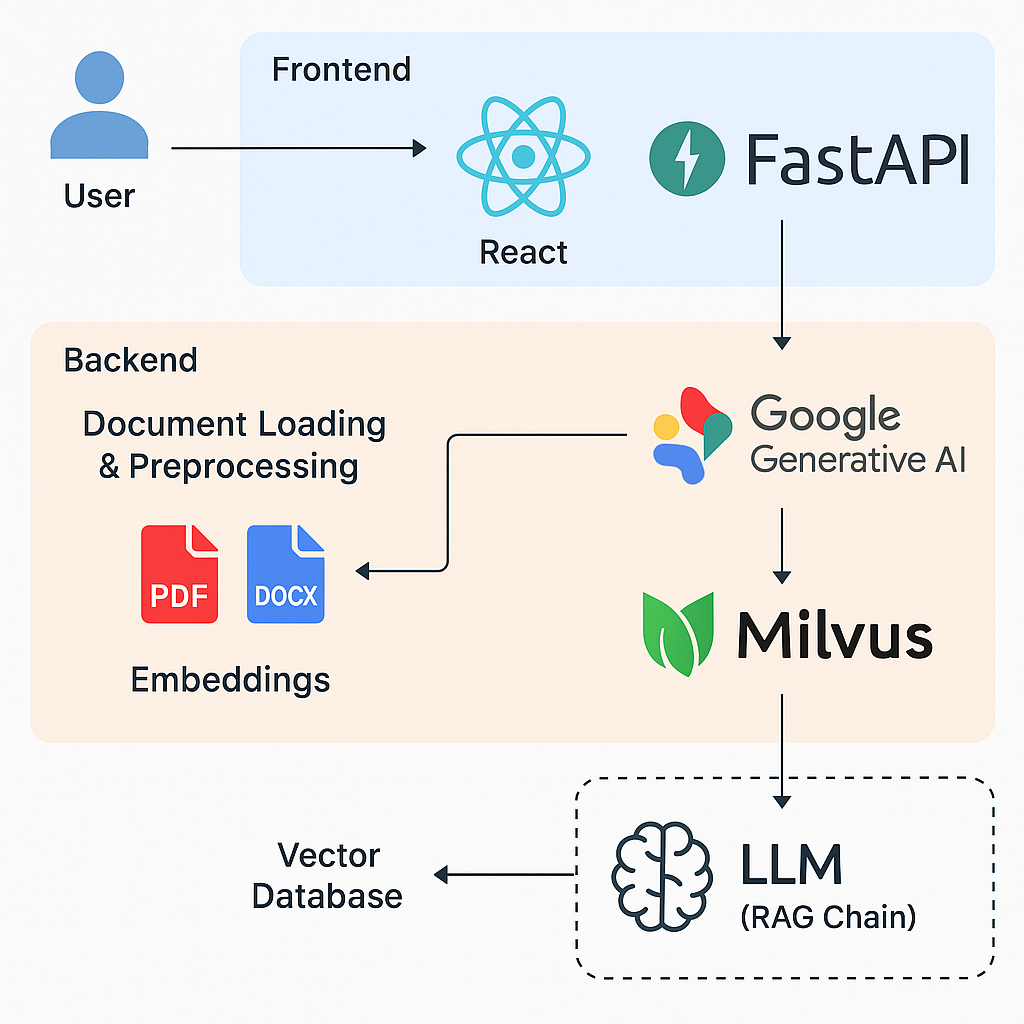
}

* **Material UI**: For modern, responsive UI components.
* **React Doc Viewer**: To preview uploaded documents.
* **Framer Motion**: For animations.
* **PDF.js**: To handle and render PDFs.

**Features**

* Upload and parse PDF/DOCX files.
* Ask natural language questions related to uploaded policies.
* AI-powered answer generation with references.
* Control access to public/private documents.
* Real-time document preview and Q&A.

**Architecture**

****

1. **Document Loader**: Uploads PDF/DOCX files, parses into text.
2. **Chunking Engine**: Splits large content into overlapping chunks.
3. **Embedding Generator**: Converts chunks into vector format using Gemini embeddings.
4. **Vector Store (Milvus)**: Stores all embeddings for future semantic search.
5. **Query Processor**:
   * Converts user query into embedding.
   * Searches for top relevant chunks.
   * Sends context + query to Gemini LLM.
   * Returns final answer.
6. **Frontend**: React UI with file viewer, Q&A interface, and styled answers.

**Deployment Notes**

* Ensure Milvus is running (local or cloud instance).
* Set up environment variables for API keys (Google Gemini).
* Use Uvicorn to run FastAPI server.
* Run React frontend on default dev server or integrate into a single deployment (optional).

**Future Enhancements**

* Role-based access to policies (Board Member/Admin/Public).
* Audit logs of questions asked.
* Multi-lingual question-answer support.
* Document tagging and auto-categorization.
* Summarization & highlights for faster reading.
* Integration with existing community management portals.

**Conclusion**

PoliBot delivers an intelligent and user-friendly interface for understanding complex policy documents using the latest LLM and vector search capabilities. It reduces manual effort and provides real-time, reliable answers within organizational boundaries.

Developed by: Huzefa Khan  
Stack: Python, FastAPI, LangChain, Google GenAI, Milvus, ReactJS