AI Legal Consultant

Arshdeep Singh, Soumya Bhate, Yatharth Kumar

Whoa!!

This AI assistant provides general legal information only.
It does NOT constitute legal advice. For specific legal matters, please consult a licensed attorney.

Table of Contents

01

Project Overview

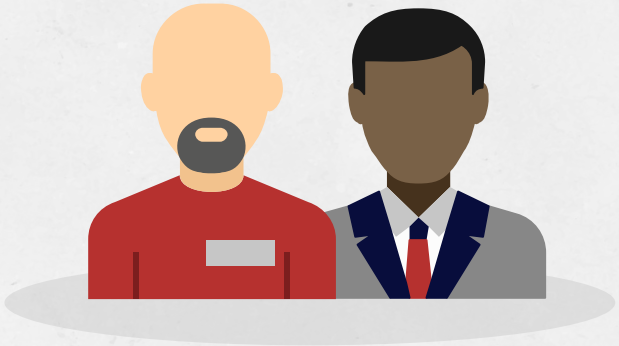
An abstract of the legal AI chatbot

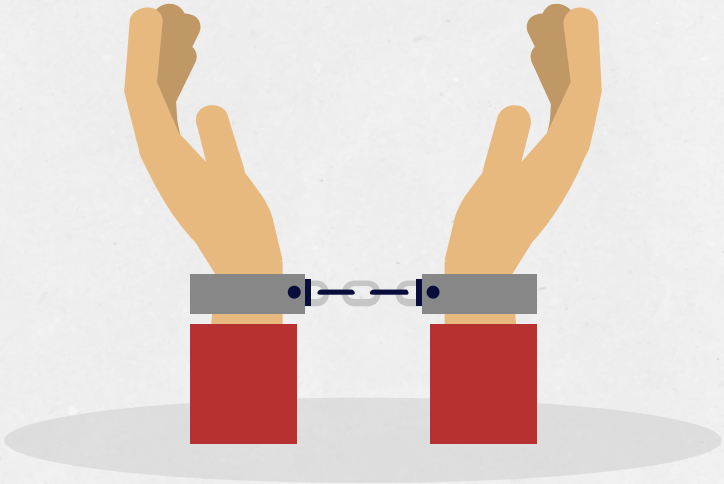
Project Overview

Purpose: Build a chatbot that answers criminal law questions using a RAG pipeline

Key Features-

- Domain-specific knowledge base (Eg- Criminal, Contract, Real estate Law etc)
- MongoDB vector search with embeddings
- Open AI powered LLM response generation
- Text-to-speech (TTS) and Speech to text (STT) via Eleven Labs
- Upload your legal documents and ask away!
- Downloadable consultations and audio device settings





02

Knowledge Base & Chunking

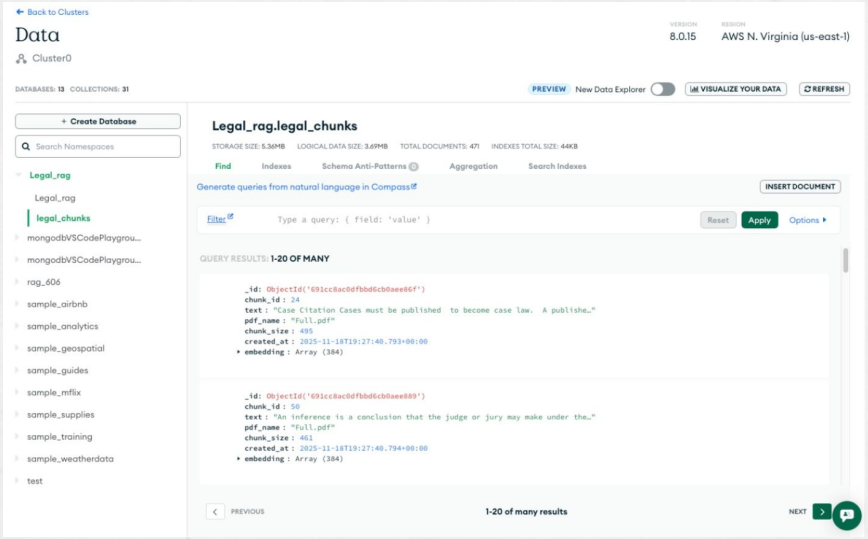
Using MongoDB to create a KB for a RAG style bot

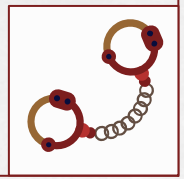
KB and Chunking

- PDF Upload & Extraction using PyPDF2 to extract text from Law Books and legal documents.
- Chunking sentences of this extracted text with overlap (size=500, overlap=100) to provide context across chunks
- Metadata Stored in MongoDB with chunk_id, pdf_name, chunk_size, created_at
- KB in MongoDB decreases the risk of hallucinations faced by most transformer models

Why do we need a KB?

- We know that laws are getting updated almost every day and it is not possible for LLMs to keep up to it.
- Building a rag style chatbot makes it feasible for LLM to get richer context and help you make better decisions!
- The screenshot shows chunks of a criminal law pdf. The LLM can now answer your questions from your uploaded text with this Criminal Law Pdf as context.





03

Embedding & Vector Search

Discussing about how we are embedding the chunk of texts and using vector search

Embedding & Vector Search

Embedding Model

Via SentenceTransformers

all-miniLM-L6 -v2

MongoDB

Storage

Embeddings stored in a NoSQL DB using bulk write

Vector Search

Retrieves top-k relevant chunks based on query embedding

$vectorSearch

Relevant Information from the KB

Output

User query will be the input and vector search will make sure it provides relevant information from the selected KB (streamlit Legal Area option)

04

RAG Answer Generation

Explains how KB is being used to answer questions

RAG Answer Generation

Search Function:

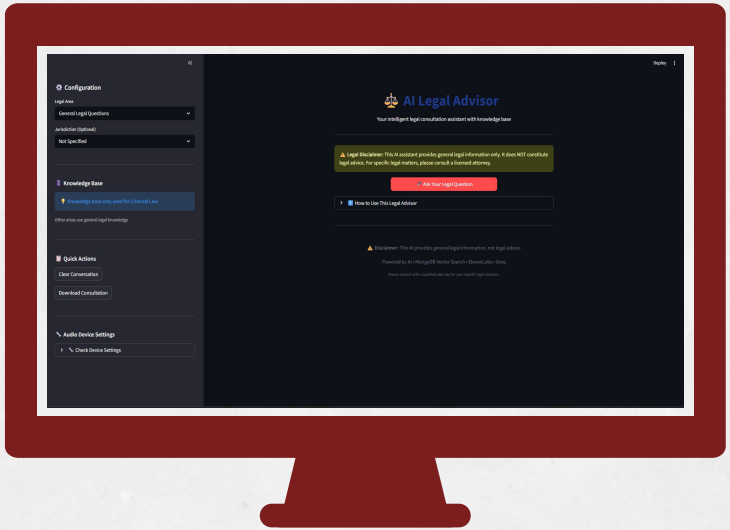
- Retrieves top 3 (customizable) relevant chunks using semantic similarity

Prompt Construction:

- Injects retrieved context into Groq prompt
- Uses openai/gpt-oss-120b for response generation

Guardrails:

- Bot only responds to legal queries
- Prompt role curated for “legal advisor”
- Disclaimer embedded in prompt and UI



05

User Interface and Experience

Talks about the Streamlit UI with python and the interaction flow

UI and Interaction Flow

Streamlit UI Features:

- Select your Legal Area that you have questions about
- Criminal Law KB toggle (MongoDB-backed, will upload more documents)
- Audio device settings and TTS playback (TTS not added yet)
- Downloadable consultation as .txt
- Conversation history maintained

Interaction Flow:

- “Ask Your Legal Question” button / Upload your document
- Input field with placeholder guidance
- Legal disclaimer prominently displayed

06

Future Enhancements

Milestones to Complete

Features to add

- Multi-domain KB support (Civil, Corporate, etc.)
- Upload PDF/ document option for the users with personal details redacted
- Real-time audio playback (TTS)
- Model fine-tuning for legal tone and accuracy and improve UI

Thanks!

Does anyone have any questions?