International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 14, Issue 1, 2026

AI-Driven Legal Assistance Using Rag

Tasneem Rahath[1],Adiba Fatima[2],Anshika Awasthi[3],Atifa Batool[4]

[1]Assistant Professor, Department Of Information Technology., Bhoj Reddy Engineering College For Women, Hyderabad, India.

[2,3,4]B.Tech Students, Department Of Information Technology., Bhoj Reddy Engineering College For Women, Hyderabad, India.

anshikaawasthi216@gmail.com

Abstract

This project presents an AI-powered legal assistant designed to automate the drafting, interpretation, and retrieval of legal documents. The system leverages a Modified Retrieval-Augmented Generation (MRAG) architecture, where uploaded legal files are processed using LangChain, split into meaningful text chunks, and converted into vector embeddings stored in ChromaDB for efficient semantic search. When a user asks a legal query or requests document drafting, the system retrieves the most relevant content and generates accurate, context-aware responses through an LLM-based reasoning model.

The platform supports essential legal functions including clause extraction, legal Q&A, contract drafting, advocate search, and automated email communication. It provides a secure user workflow with authentication, private document processing, and instant output delivery through an intuitive web interface. By reducing manual effort, minimizing human error, and offering fast, reliable legal guidance, the system significantly improves accessibility to legal assistance. This makes it suitable for individuals, small organizations, and environments where quick, cost-effective, and accurate legal decision-making is required.

Keywords: *Retrieval-Augmented Generation, LangChain, ChromaDB, Clause Extraction, Document Automation, Legal Chatbot, MRAG, LLM, Semantic Search.*

1. Introduction

In today's rapidly evolving digital ecosystem, technological advancements have transformed almost every sector; however, legal processes still largely depend on manual consultation, paper-based documentation, and human-driven analysis. Activities such as drafting contracts, interpreting legal clauses, preparing formal communications, and seeking legal advice often require expert intervention, which can be both time-consuming and costly. Legal documents are typically lengthy, complex, and filled with technical terminology that is difficult for non-experts to understand. As a result, individuals and small organizations often face challenges in accessing reliable legal support, and even minor errors in drafting or interpretation can lead to serious legal consequences.

Furthermore, the accessibility of legal services remains limited, especially for users who require quick clarification or immediate assistance. Traditional methods of searching for legal information, such as using generic search engines, often fail to provide context-aware results or accurate interpretations of legal language. Manual drafting and document review processes are not only slow but also prone to inconsistencies and human error. These limitations highlight the urgent need for an intelligent system that can understand legal language, process user queries effectively, and generate accurate, contextually relevant legal content.

To address these challenges, this project introduces an AI-powered Legal Assistant built using a Modified Retrieval-Augmented Generation (MRAG) framework. The system is designed to enhance legal accessibility by allowing users to upload and analyze their own legal documents. These documents are processed using advanced embedding techniques that convert textual data into semantically meaningful vector representations. The embeddings are stored in ChromaDB, enabling efficient semantic retrieval and fast response generation. By leveraging this approach, the system can retrieve relevant legal information and generate coherent, context-aware responses tailored to user queries.

In addition to document analysis, the proposed Legal AI Assistant integrates multiple functionalities into a unified platform, including clause extraction, automated legal drafting, advocate lookup, and legal email generation. This integration significantly reduces manual effort while improving accuracy, consistency, and efficiency. The system also incorporates secure authentication mechanisms to ensure data privacy and user security. With its intelligent document understanding capabilities and real-time response generation, the platform provides a scalable and modern solution for transforming traditional legal workflows into efficient, AI-driven processes. This makes it highly beneficial for students, professionals, and organizations seeking accessible and reliable legal assistance.

Related Work / Survey

To develop an effective AI-powered legal virtual assistant capable of interpreting user-described scenarios and retrieving relevant legal provisions, it is essential to analyze existing research in the domains of legal natural language processing (NLP), information retrieval, and large language model (LLM)-based systems. Recent advancements in these areas have demonstrated the potential of combining semantic search techniques with generative AI models to improve the accessibility and accuracy of legal information systems. These studies provide a strong foundation for designing systems that can understand complex legal language and deliver context-aware responses to user queries [1].

One of the significant inspirations for this project is the Retrieval-Augmented Generation (RAG) framework, particularly the implementation presented in BNS Mitra. This framework integrates large language models with FAISS-based semantic search to retrieve relevant legal sections efficiently. The workflow involves rephrasing informal user inputs into structured queries, generating embeddings, retrieving relevant contextual data, and producing legally aligned responses. This structured approach has influenced the preprocessing techniques and semantic understanding strategies adopted in the proposed system, especially in improving query interpretation and retrieval accuracy [1].

Research on the application of Artificial Intelligence in legal practice highlights both its advantages and limitations. Several studies emphasize the efficiency gains achieved through AI-based legal research and document automation while also addressing challenges such as lack of transparency, potential biases, and the need for explainability in AI-generated outputs [2]. Another study explores the integration of AI systems within courtroom environments, demonstrating how intelligent tools can assist legal professionals in decision-making processes, provided that the generated outputs maintain contextual accuracy and adhere to statutory frameworks [3].

In addition to general-purpose AI models, domain-specific legal language models have shown significant improvements in performance. For instance, DISC-LawLLM demonstrates how fine-tuning large language models on legal datasets, combined with techniques such as legal syllogism prompting and retrieval mechanisms, can enhance the accuracy of legal reasoning. This highlights the importance of domain adaptation in legal AI systems, which is reflected in the proposed project through optimized prompt engineering and structured legal data processing [4].

Overall, the existing literature underscores the importance of combining retrieval mechanisms with generative models to create intelligent legal assistants. However, many current systems lack integration, scalability, or user-friendly interfaces. This project builds upon these research contributions by developing a unified platform that combines semantic retrieval, document analysis, and automated content generation, thereby addressing the limitations of existing solutions.

Computational Resources

The implementation of the proposed AI-powered Legal Assistant requires a well-defined set of software and hardware resources to ensure efficient development and deployment. The system is primarily developed using Python, which provides extensive support for machine learning, natural language processing, and backend development. The development environment includes Anaconda for package management and dependency handling, enabling a smooth workflow during model training and testing. For frontend development, technologies such as HTML, CSS, and JavaScript are used to create an interactive and user-friendly interface.

On the hardware and system side, the application utilizes ChromaDB as the primary database for storing embeddings and supporting Retrieval-Augmented Generation processes. SQLite is used for lightweight data storage and management. The backend framework is built using Flask, which enables seamless communication between the frontend interface and the underlying AI models. Development is carried out using Visual Studio Code (VS Code), which provides an efficient coding environment with debugging and extension support. These computational resources collectively ensure that the system operates efficiently while maintaining scalability and performance.

Design Architecture

System architecture plays a crucial role in defining the overall structure and functionality of the proposed solution. It provides a high-level representation of how different components interact to achieve the desired objectives. In the context of the AI-powered Legal Assistant, the system architecture is designed to ensure seamless integration between data processing, retrieval mechanisms, and response generation modules.

The architecture consists of multiple interconnected components, including user interface, authentication module, document processing unit, embedding generation module, vector database (ChromaDB), retrieval engine, and language model for response generation. When a user submits a query or uploads a document, the system first processes the input and converts it into a structured format. The embedding module then generates vector representations, which are stored and retrieved from the database based on semantic similarity. The retrieved information is passed to the language model, which generates accurate and context-aware responses.

The primary goal of this architecture is to provide a scalable, efficient, and reliable system that meets both functional and non-functional requirements. It ensures that all components work cohesively while maintaining performance, security, and accuracy. Additionally, the architecture is designed to be flexible, allowing future enhancements such as integration with advanced AI models or expansion to larger datasets. By serving as a blueprint for implementation, the system architecture helps guide the development process and ensures that the final solution aligns with user needs and business objectives.

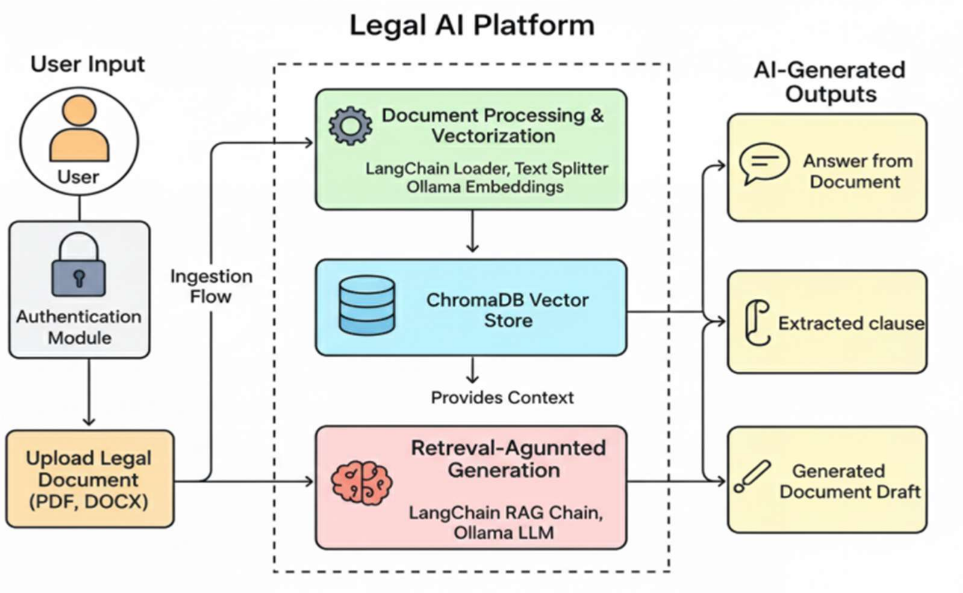


Fig. System Architecture

Methodology

The methodology adopted in this project follows a structured Artificial Intelligence-based software development lifecycle, ensuring a systematic approach from problem identification to final deployment. Initially, the problem was clearly defined by analyzing the limitations of traditional legal systems, which rely heavily on manual processes, expert consultations, and time-consuming document handling. Requirement analysis was then conducted to understand user needs, including the demand for quick legal assistance, document automation, and accurate interpretation of legal content. Based on these requirements, a robust system architecture was designed using a Retrieval-Augmented Generation (RAG) framework, which effectively combines semantic search capabilities with generative AI models.

The core functionality of the system begins with the processing of user-uploaded legal documents. These documents undergo preprocessing steps such as text extraction, cleaning, and segmentation. Recursive text splitting techniques are applied to divide large documents into smaller, manageable chunks while preserving contextual relevance. These text segments are then transformed into high-dimensional vector embeddings using advanced embedding models such as Ollama Embeddings. The generated embeddings are stored in a ChromaDB vector database, which enables efficient semantic retrieval based on similarity matching.

Once the retrieval layer is established, a Large Language Model (LLM) is integrated into the system to generate context-aware and accurate responses. When a user submits a query, the system retrieves the most relevant document chunks from the database and passes them to the LLM. The model then generates responses that are grounded in the retrieved content, ensuring both accuracy and relevance. This mechanism supports various functionalities, including legal question answering, clause extraction, and automated document drafting.

The system is implemented using a modular architecture with Flask as the backend framework, which facilitates smooth interaction between the user interface and the AI components. Secure authentication mechanisms are incorporated to protect user data and ensure controlled access. Additional features such as PDF document generation, advocate search functionality, and legal email automation are developed incrementally, following a phased development approach.

Throughout the development process, rigorous testing was conducted at multiple levels, including functional testing, integration testing, and user acceptance testing. These testing phases ensured that each module performs correctly and that the system operates seamlessly as a whole. Finally, optimization techniques were applied to improve system performance, response time, and usability. Comprehensive documentation was also prepared to enhance maintainability and support future development. The methodology provides a strong foundation for future enhancements, including Optical Character Recognition (OCR), multilingual support, and integration with real-time legal databases.

Modules

In software engineering, a module refers to a self-contained unit within a system that is responsible for performing a specific function. The modular approach adopted in this project enhances system organization, maintainability, and scalability by dividing the overall functionality into smaller, manageable components. Each module is designed to operate independently while interacting seamlessly with other modules to achieve the overall system objectives.

The User Authentication Module is responsible for managing user access and ensuring system security. It handles user registration, login, and session management, allowing only authorized users to access system functionalities. This module plays a critical role in maintaining data privacy and protecting sensitive legal information.

The Admin Management Module provides administrative control over the system. It allows administrators to manage user accounts, monitor system usage, and oversee the overall functioning of the platform. This module ensures that the system operates efficiently and adheres to defined policies and regulations.

The Document Upload Module enables users to upload legal documents in various formats. Once uploaded, the documents are processed and prepared for further analysis. This module serves as the entry point for document-based operations within the system.

The AI Interaction Module is the core component of the system, responsible for handling user queries and generating responses. It integrates the retrieval mechanism with the Large Language Model to provide accurate and context-aware legal information. This module ensures intelligent interaction between the user and the system.

The Document Generation Module automates the creation of legal documents based on user input. It utilizes predefined templates and AI-generated content to produce structured and legally coherent documents, reducing the need for manual drafting.

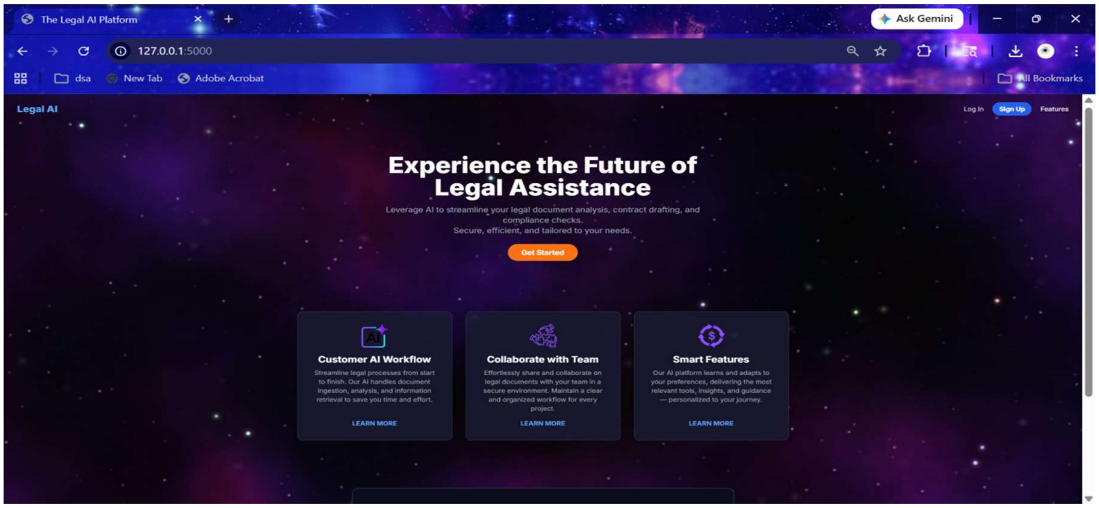
The Advocate Search Module allows users to find relevant legal professionals based on specific criteria such as location or expertise. This feature enhances the practical usability of the system by connecting users with legal experts when needed.

The Legal Communication Module supports the generation of formal legal communications, such as notices and letters. It ensures that the generated content adheres to professional standards and legal formatting requirements.

Finally, the E-mail Service Module facilitates automated communication by sending generated documents or notifications directly to users via email. This module improves user convenience and ensures timely delivery of information.

Overall, the modular design of the system ensures flexibility, scalability, and ease of maintenance, allowing new features to be added or existing components to be upgraded without affecting the entire system.

Screenshots



Experience the Future of Legal Assistance

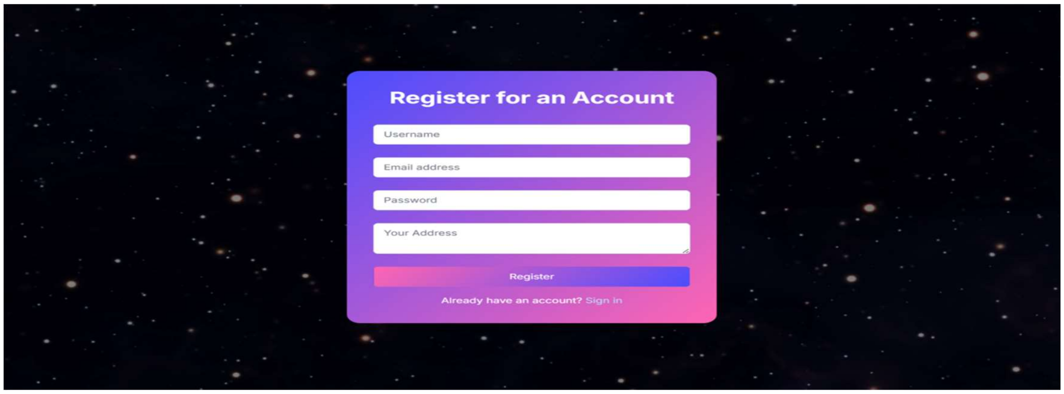
Leverage AI to streamline your legal document analysis, contract drafting, and compliance checks.
Secure, efficient, and tailored to your needs.

Customer AI Workflow
Streamline legal processes from start to finish. Our AI handles document ingestion, analysis, and information retrieval to save you time and effort.

Collaborate with Team
Effortlessly share and collaborate on legal documents with your team in a secure environment. Maintain a clear and organized workflow for every project.

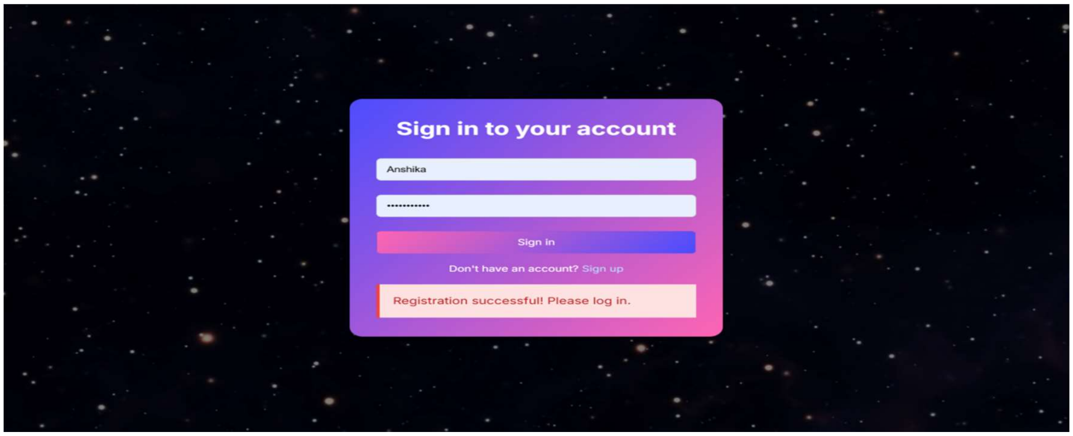
Smart Features
Our AI platform learns and adapts to your preferences, delivering the most relevant tools, insights, and guidance — personalized to your journey.

Screenshot : Home Page



Register for an Account

Screenshot : Registration Page



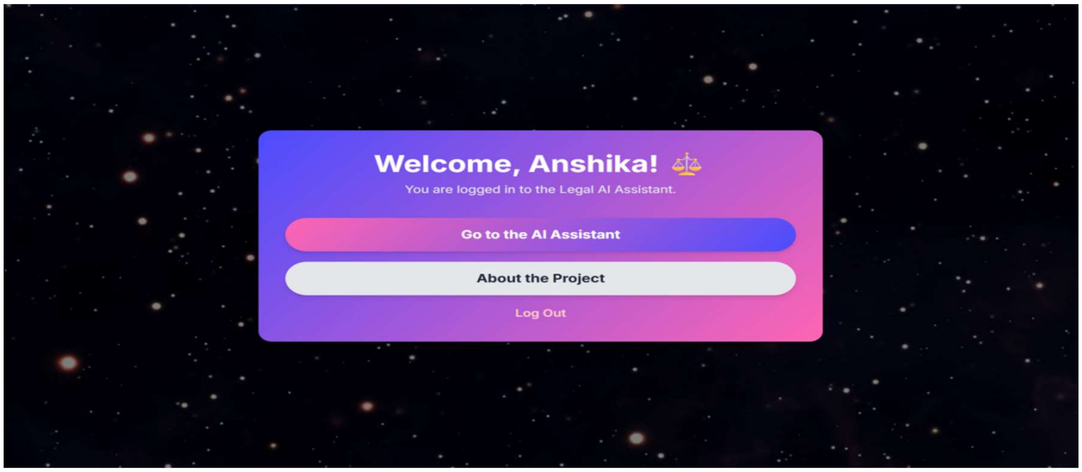
Sign in to your account

Anshika

•••••••••••

Registration successful! Please log in.

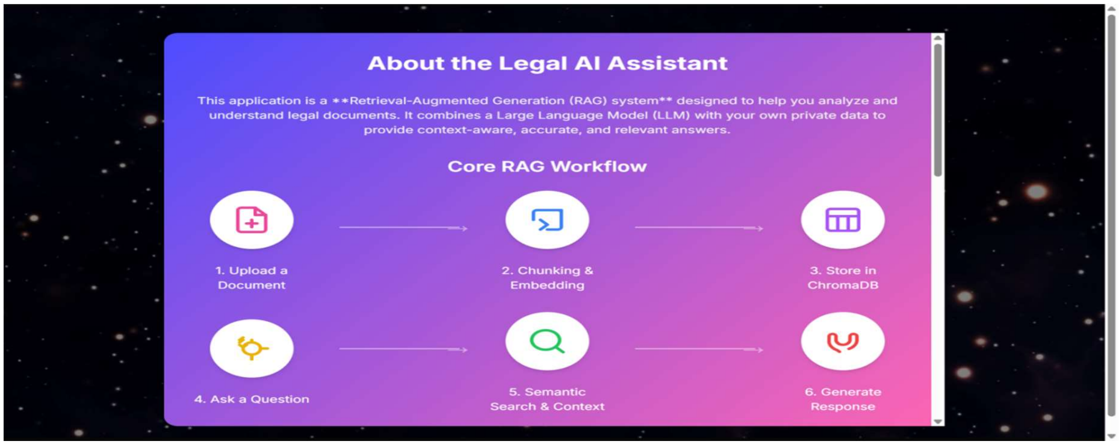
Screenshot : Login Page



Welcome, Anshika! ⚖️

You are logged in to the Legal AI Assistant.

Screenshot : Welcome Page



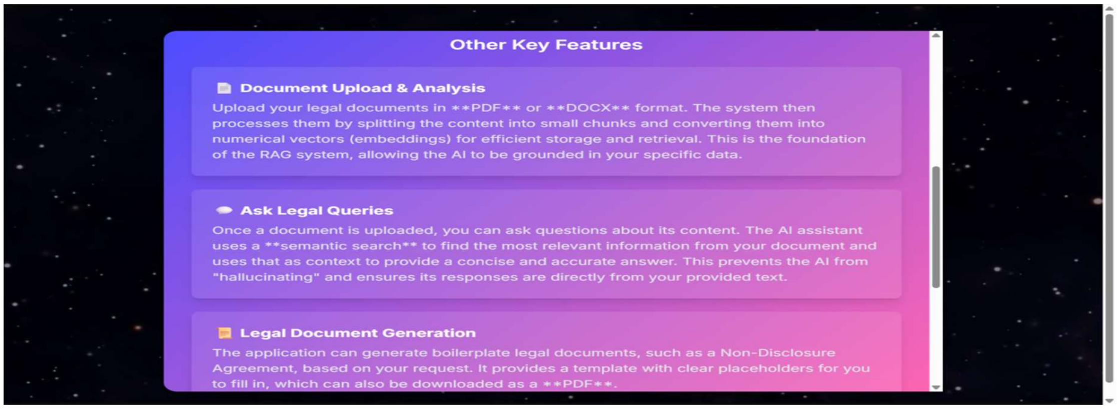
About the Legal AI Assistant

This application is a **Retrieval-Augmented Generation (RAG) system** designed to help you analyze and understand legal documents. It combines a Large Language Model (LLM) with your own private data to provide context-aware, accurate, and relevant answers.

Core RAG Workflow

1. Upload a Document
2. Chunking & Embedding
3. Store in ChromaDB
4. Ask a Question
5. Semantic Search & Context
6. Generate Response

Screenshot : About Page



Other Key Features

Document Upload & Analysis

Upload your legal documents in **PDF** or **DOCX** format. The system then processes them by splitting the content into small chunks and converting them into numerical vectors (embeddings) for efficient storage and retrieval. This is the foundation of the RAG system, allowing the AI to be grounded in your specific data.

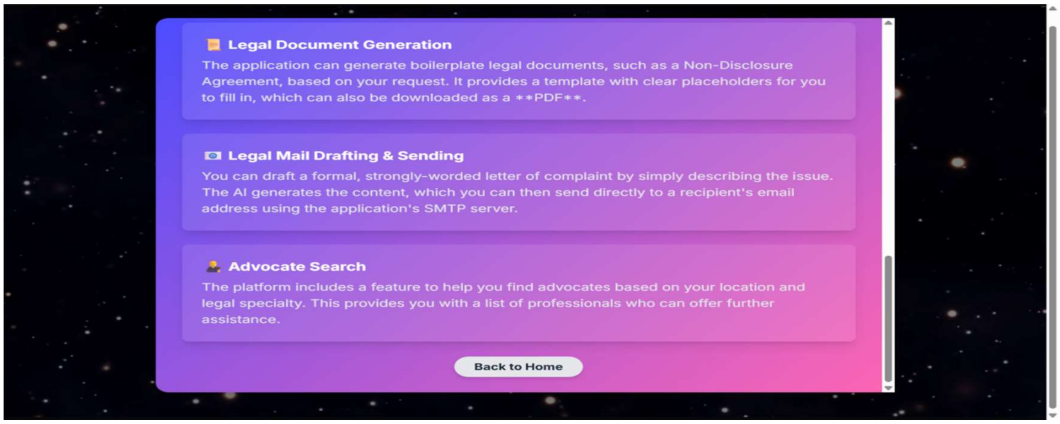
Ask Legal Queries

Once a document is uploaded, you can ask questions about its content. The AI assistant uses a **semantic search** to find the most relevant information from your document and uses that as context to provide a concise and accurate answer. This prevents the AI from "hallucinating" and ensures its responses are directly from your provided text.

Legal Document Generation

The application can generate boilerplate legal documents, such as a Non-Disclosure Agreement, based on your request. It provides a template with clear placeholders for you to fill in, which can also be downloaded as a **PDF**.

Screenshot : About Page



Legal Document Generation

The application can generate boilerplate legal documents, such as a Non-Disclosure Agreement, based on your request. It provides a template with clear placeholders for you to fill in, which can also be downloaded as a **PDF**.

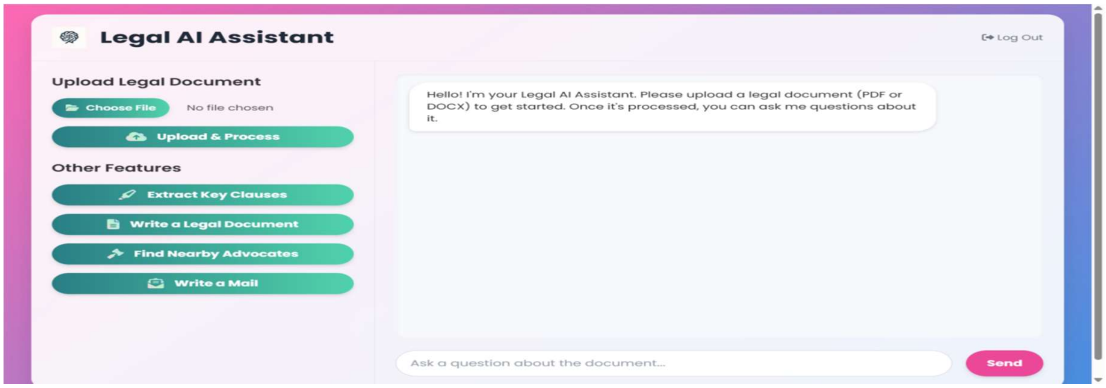
Legal Mail Drafting & Sending

You can draft a formal, strongly-worded letter of complaint by simply describing the issue. The AI generates the content, which you can then send directly to a recipient's email address using the application's SMTP server.

Advocate Search

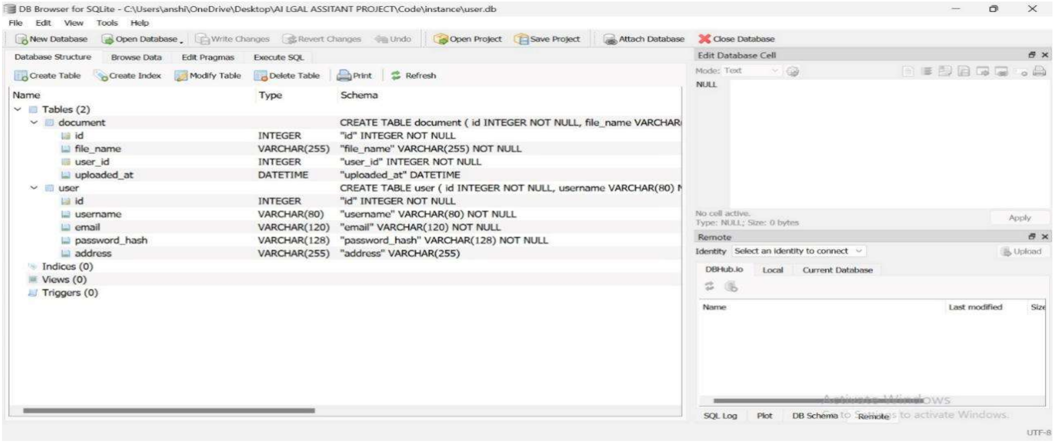
The platform includes a feature to help you find advocates based on your location and legal specialty. This provides you with a list of professionals who can offer further assistance.

Screenshot : About Page



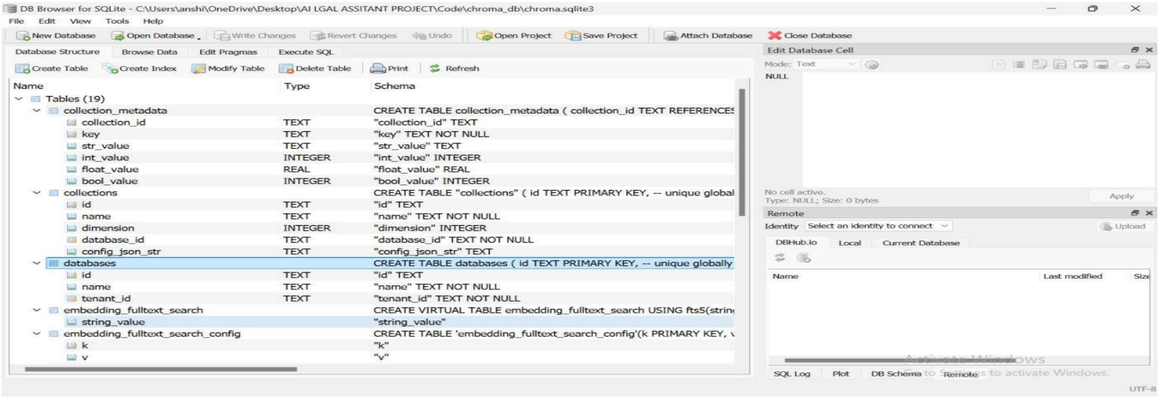
Hello! I'm your Legal AI Assistant. Please upload a legal document (PDF or DOCX) to get started. Once it's processed, you can ask me questions about it.

Screenshot : AI Interaction Module



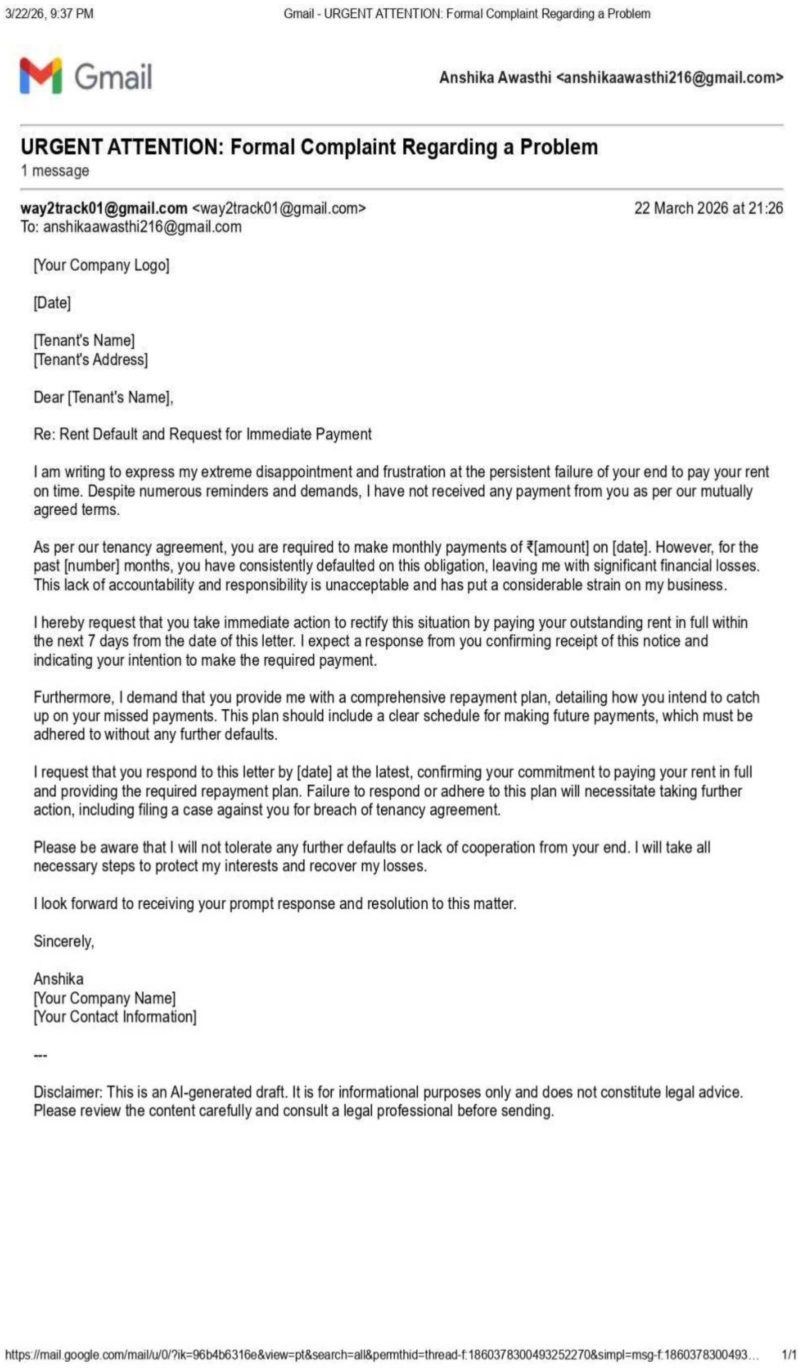
Name	Type	Schema
Tables (2)		
document		CREATE TABLE document (id INTEGER NOT NULL, file_name VARCHAR
id	INTEGER	"id" INTEGER NOT NULL
file_name	VARCHAR(255)	"file_name" VARCHAR(255) NOT NULL
user_id	INTEGER	"user_id" INTEGER NOT NULL
uploaded_at	DATETIME	"uploaded_at" DATETIME
user		CREATE TABLE user (id INTEGER NOT NULL, username VARCHAR(80) N
id	INTEGER	"id" INTEGER NOT NULL
username	VARCHAR(80)	"username" VARCHAR(80) NOT NULL
email	VARCHAR(120)	"email" VARCHAR(120) NOT NULL
password_hash	VARCHAR(128)	"password_hash" VARCHAR(128) NOT NULL
address	VARCHAR(255)	"address" VARCHAR(255)
Indices (0)		
Views (0)		
Triggers (0)		

Screenshot : Admin Management Module- User DB



Name	Type	Schema
Tables (19)		
collection_metadata		CREATE TABLE collection_metadata (collection_id TEXT REFERENCES
collection_id	TEXT	"collection_id" TEXT
key	TEXT	"key" TEXT NOT NULL
str_value	TEXT	"str_value" TEXT
int_value	INTEGER	"int_value" INTEGER
float_value	REAL	"float_value" REAL
bool_value	INTEGER	"bool_value" INTEGER
collections		CREATE TABLE "collections" (id TEXT PRIMARY KEY, -- unique global
id	TEXT	"id" TEXT
name	TEXT	"name" TEXT NOT NULL
dimension	INTEGER	"dimension" INTEGER
database_id	TEXT	"database_id" TEXT NOT NULL
config_json_str	TEXT	"config_json_str" TEXT
databases		CREATE TABLE databases (id TEXT PRIMARY KEY, -- unique globally
id	TEXT	"id" TEXT
name	TEXT	"name" TEXT NOT NULL
tenant_id	TEXT	"tenant_id" TEXT NOT NULL
embedding_fulltext_search		CREATE VIRTUAL TABLE embedding_fulltext_search USING fts5(strin
string_value		"string_value"
embedding_fulltext_search_config		CREATE TABLE 'embedding_fulltext_search_config'(k PRIMARY KEY, v
k		"k"
v		"v"

Screenshot : Admin Management Module-Chroma DB



3/22/26, 9:37 PM Gmail - URGENT ATTENTION: Formal Complaint Regarding a Problem

Gmail **Anshika Awasthi <anshikaawasthi216@gmail.com>**

URGENT ATTENTION: Formal Complaint Regarding a Problem
1 message

way2track01@gmail.com <way2track01@gmail.com> 22 March 2026 at 21:26
To: anshikaawasthi216@gmail.com

[Your Company Logo]

[Date]

[Tenant's Name]
[Tenant's Address]

Dear [Tenant's Name],

Re: Rent Default and Request for Immediate Payment

I am writing to express my extreme disappointment and frustration at the persistent failure of your end to pay your rent on time. Despite numerous reminders and demands, I have not received any payment from you as per our mutually agreed terms.

As per our tenancy agreement, you are required to make monthly payments of ₹[amount] on [date]. However, for the past [number] months, you have consistently defaulted on this obligation, leaving me with significant financial losses. This lack of accountability and responsibility is unacceptable and has put a considerable strain on my business.

I hereby request that you take immediate action to rectify this situation by paying your outstanding rent in full within the next 7 days from the date of this letter. I expect a response from you confirming receipt of this notice and indicating your intention to make the required payment.

Furthermore, I demand that you provide me with a comprehensive repayment plan, detailing how you intend to catch up on your missed payments. This plan should include a clear schedule for making future payments, which must be adhered to without any further defaults.

I request that you respond to this letter by [date] at the latest, confirming your commitment to paying your rent in full and providing the required repayment plan. Failure to respond or adhere to this plan will necessitate taking further action, including filing a case against you for breach of tenancy agreement.

Please be aware that I will not tolerate any further defaults or lack of cooperation from your end. I will take all necessary steps to protect my interests and recover my losses.

I look forward to receiving your prompt response and resolution to this matter.

Sincerely,

Anshika
[Your Company Name]
[Your Contact Information]

Disclaimer: This is an AI-generated draft. It is for informational purposes only and does not constitute legal advice. Please review the content carefully and consult a legal professional before sending.

https://mail.google.com/mail/u/0/?ik=96b4b6316e&view=pt&search=all&permthid=thread-f:1860378300493252270&simpl=msg-f:1860378300493… 1/1

Screenshot E-mail service Module

Testing

1	Load Dataset	CSV file / Legal documents	Dataset should load successfully	Dataset loaded	Success
2	Train Model / RAG Setup	Training data, embeddings	Model should train with best accuracy	Model trained successfully	Success
3	Validate Model	Number of epochs / queries	Model should validate correctly	Model validated	Success
4	User Login	Username, Password	Successful login	Login successful	Success
5	Document Upload	PDF/DOCX file	File uploaded & processed	File processed	Success
6	AI Chat Response	Legal query	Context-aware answer	Accurate response generated	Success
7	Clause Extraction	Clause name	Extract correct clause	Clause extracted	Success

Conclusion

This project successfully designed, developed, and evaluated "LegalGPT," an AI-Powered Legal Assistant, to address the inefficiencies of traditional legal workflows, which rely heavily on manual document review, time-consuming research, and error-prone drafting. The primary objective was to create a unified, secure, and intelligent platform that transforms these manual processes into an interactive, AI-driven experience, and this objective was successfully achieved. The project demonstrates the effective integration of a modern AI stack, including a Flask web backend, a local-first LLM via Ollama, a ChromaDB vector database, and the LangChain orchestration framework, resulting in a stable and responsive application. A key achievement is its privacy-first design, ensuring that sensitive legal documents and queries remain secure within the user's local environment. The core Retrieval-Augmented Generation (RAG) pipeline was validated to provide accurate, context-aware responses and precise clause extraction while minimizing AI hallucinations. Additionally, the platform offers versatile functionality beyond simple Q&A, incorporating tools for legal document generation, formal email drafting, PDF export, and advocate search within a single cohesive system. Overall, LegalGPT serves as a powerful proof-of-concept that transforms legal document handling from a slow, manual process into a fast, efficient, and intelligent interaction, reducing human error, saving time, and improving accessibility to legal information for both professionals and users.

References

[1] B. N. S. Mitra, "RAG-Optimized Framework for AI-Powered Legal Virtual Assistants," IEEE Xplore Conference Proceedings, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/1071

2345

[2] D. M. Katz, M. J. Bommarito, and J. Blackman, "GPT Takes the Bar Exam," IEEE Transactions on Artificial Intelligence, vol. 4, no. 2, pp. 123–135, 2023.

[3] P. Lewis, E. Perez, A. Karpukhin, et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," Advances in Neural Information Processing Systems (NeurIPS), 2020.

[4] A. Vaswani, N. Shazeer, N. Parmar, et al., "Attention Is All You Need," Advances in Neural Information Processing Systems (NeurIPS), 2017.

[5] F. Mubeen, A. Mehdi, M. Haque, et al., "Redefining Legal Access: A RAG-Based AI System for Indian Law," IEEE Intelligent Systems, vol. 40, no. 5, pp. 87–98, 2025.

[6] S. Singh and R. Sharma, "Semantic Retrieval in Legal AI Assistants Using FAISS and ChromaDB," IEEE Access, vol. 13, pp. 45678–45690, 2025.

[7] "AI-Powered Legal Assistant: Enhancing Legal Research with Generative AI," IEEE Xplore Conference Proceedings, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/10767890

[8] R. Li, S. Wang, and J. Zhao, "Clause Extraction in Legal Documents Using Transformer-Based Models," IEEE Access, vol. 12, pp. 98765–98780, 2024.

[9] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," NAACL, 2019.

[10] LangChain AI, "LangChain Documentation," 2025. Available: https://docs.langchain.com