Building an AI System for Interacting with Legal Cases

A Project Report submitted in partial fulfilment of the requirements for the award of the degree of

Bachelor of Technology

In

Computer Science and Engineering

by

Harsh Rajoria (2215500074)

Kartik Chahar (2215500085)

Bhanu Pratap Singh (2215500044)

Garvit Saraswat (2215500069)

Bhuvanesh Sharma (2215500046)

Under the Guidance of

Dr. Rahul Pradhan,
Associate Professor

DepartmentofComputerEngineering&Applications Institute

of Engineering & Technology

GLA University, Mathura – 281406

Date of Submission – 23rd December 2024

Contents

Department of Computer Engineering and Applications GLA University, 17 km. Stone NH#2, Mathura-Delhi Road, Chaumuha, Mathura – 281406 U.P (India)

Declaration

I hereby declare that the work which is being presented in the B.Tech. Mini Project **“Building an AI System for Interacting with Legal Cases”**, in partial fulfillment of the requirements for the award of the ***Bachelor of Technology*** in Computer Science and Engineering and submitted to the Department of Computer Engineering andApplications of GLA University, Mathura, is an authentic record of my own work carried under the supervision of **Dr. Rahul Pradhan, Associate Professor.**

The contents of this project report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree.

Harsh Rajoria
University Roll No.: 2215500074

Garvit Saraswat
University Roll No.: 2215500069

Bhuvanesh Sharma
University Roll No.: 2215500046

Kartik Chahar
University Roll No.: 2215500085

Bhanu Pratap Singh
University Roll No.: 2215500044

Certificate

This is to certify that Harsh Rajoria, Kartik Chahar, Garvit Saraswat, Bhanu Pratap Singh and Bhuvanesh Sharma has successfully completed the project titled – 'Building an AI System for Interacting with Legal Cases' at GLA University under my supervision and guidance in the fulfilment of requirements of Fifth Semester, Bachelor of Technology – Honors (Computer Science & Engineering) of GLA University, Mathura.

Dr. Rahul Pradhan

Acknowledgement

We would like to extend our sincere and heartfelt thanks towards all those who have helped us in making this project. Without their active guidance, help, cooperation and encouragement, we would not have been able to present the project on time.

We extend our sincere gratitude to our trainer Dr. Rahul Pradhan for his moral support and guidance during the tenure of our project.

We also acknowledge with a deep sense of reverence, our gratitude towards our parents and other faculty members of the college for their valuable suggestions given to us in completing the project.

Dr. Rahul Pradhan

Abstract

The legal domain is characterized by the extensive use of textual data, such as case laws, judgments, and legal documents, which can be challenging to navigate and analyze efficiently. This project, *LexPilot*, aims to build an AI-powered system designed to assist legal professionals by streamlining legal research and improving access to critical information. Leveraging advanced technologies, including Large Language Models (LLMs) and vector similarity search, *LexPilot* enables efficient interaction with legal documents, particularly PDFs.

The system extracts and processes text, converting it into vectorized representations stored in FAISS for fast similarity-based retrieval. When users pose queries, *LexPilot* identifies and retrieves the most relevant sections of legal texts and generates precise, contextually relevant responses using pre-trained LLMs. This approach enhances productivity and decision-making in legal research by reducing the time spent manually sifting through documents.

Our methodology involves requirement analysis, legal data acquisition and preprocessing, vectorization using state-of-the-art embedding models, and LLM integration for query response generation. The system is implemented through a user-friendly interface developed in Streamlit, ensuring accessibility for legal practitioners.

This project has significant implications for the legal field, providing a scalable and intelligent solution for managing complex legal queries. Future developments aim to enhance the system's accuracy, expand its capabilities to address more nuanced legal scenarios, and integrate emerging technologies to further refine the user experience. *LexPilot* thus represents a step forward in leveraging AI to innovate legal research and consultancy.

Chapter–1:Introduction

1.1 Overview and Motivation

Lexpilot is a comprehensive software application designed to streamline the process of analyzing and extracting valuable information from various types of documents. Whetherit's text documents or PDF files, this tool offers a powerful set of features to facilitate efficient document processing. Lexpilot is a powerful software application designed to efficiently analyze documents across multiple formats. It seamlessly extracts text from documents while preserving formatting, enabling comprehensive content analysis using advanced natural language processing techniques. With customizable workflows and integration options, Lexpilot increases productivity and facilitates informed decision- making for both individual users and enterprise-level deployments.

The motivation behind the development of the Lexpilot stems from the ever- increasing amount of textual data generated across industries and disciplines. Traditional methods of manual document analysis are time-consuming, error-prone, and often inadequate for handling large volumes of documents. By creating a comprehensive software solution, we aim to streamline and automate the process of obtaining previews from various document formats. Our goal is to provide users with the tools necessary to effectively analyze text, identify patterns, and make informed decisions based on the valuable information contained in their documents. Ultimately, the Lexpilot serves as a catalyst for increasing productivity, fostering innovation, and unlocking the full potential of text data in a variety of fields.

1.2 Objective

The goal of Lexpilot is to offer a comprehensive software solution that simplifies document analysis processes across different formats. Using advanced text extraction algorithms and natural language processing techniques, we aim to simplify text content extraction while preserving formatting and structure. Through customizable workflows and integration options, the application aims to cater to various industries and disciplines, allowing users to tailor analytics to specific needs and workflows. Ultimately, our goal is to increase productivity, facilitate informed decision-making, and drive innovation by enabling users to efficiently extract valuable insights from their text data stores.

1.3 Summary of Similar Applications

Similar document analysis applications include TextHero, which provides sentiment analysis, keyword extraction, and summarization features; DocuSign Analyzer, specializing in legal document review with advanced search and compliance tools; IBM Watson Discovery, offering natural language processing and entity extraction to uncover insights from unstructured data; Academic research-focused Textricator with citation analysis and topic modeling; and PowerBI Document Analyzer, a plugin for Microsoft PowerBI offering data visualization and text mining capabilities. While theseapplications share the goal of aiding document analysis and insight extraction, each offers unique capabilities tailored to specific industries and use cases, from regulatory compliance to academic literature review and business intelligence.

1.4 Organization of the Project

The organization of the project has several phases aimed at ensuring its successful implementation and the fulfillment of goals.

Starting with research and requirements gathering, understanding user needs, researching frameworks and technologies for chatbot development, gathering requirements for menu integration, order tracking and billing calculations.
Following this, the process moves to language learning and learns the necessary frameworks for developing chatbots.

This is followed by system design and prototyping in which we designed the website interface and data flow, prototyped the basic website interface, and planned the integrationof the chatbot with the food ordering website.

Then comes development, implementation of chatbot features by design, integration of chatbot with backend website systems for real-time data exchange, and implementation of frontend website design. Testing is then done to identify and fix any bugs or issues, and feedback from potential users is then collected for improvements. Improving the interfaceand features based on feedback.

Model deployment to the web, ensuring seamless integration and compatibility of the chatbot with the web, launching the chatbot and monitoring its performance.
Future scope implementations consider additional features such as providing food recommendations and continuously monitoring user feedback and making necessary improvements.

.

Chapter–2: Primary Reasons for choosing this Project

1. **Simplifying Legal Research**
   Legal professionals often deal with extensive documentation and precedents. An AI system can streamline the process of retrieving relevant cases, laws, or judgments, saving time and effort.

2. **Improving Access to Justice**
   By providing easy access to legal information, such a system can empower individuals who might not afford legal consultation to understand their rights and legal options better.

3. **Reducing Errors and Bias**
   Humans may overlook key cases or details due to fatigue or cognitive biases. AI systems, when trained well, can offer consistent, accurate, and unbiased analysis.

4. **Advancing Legal Technology**
   Contributing to innovation in the legal field by introducing AI solutions that enhance productivity, workflow efficiency, and accessibility within the legal domain.

5. **Educational Value**
   The project allows for an in-depth understanding of Natural Language Processing (NLP), legal data processing, and AI's real-world applications. Use of libraries like langchain, streamlit to simplify the coding part.

6. **Market Demand**
   With the growing complexity of legal systems and increasing reliance on technology, there's a rising demand for AI-based legal solutions, making it a promising area of development.

Chapter–3: Software Requirement Analysis

3.1 Technical Feasibility

1. **Frontend technology:** Using React for front-end development ensures a modular and efficient user interface, while Tailwind CSS provides fast styling options, enabling rapid iteration and customization of the user experience.

2. **Authentication integration:** Using Clerk for authentication simplifies user management and authentication processes, provides users with a secure and seamless login while reducing development overhead.

3. **Backend Technologies:** Node.js with Express.js offers a scalable and efficient backend environment that enables asynchronous processing and easy creation of RESTful APIsto handle document parsing and data retrieval requests.

4. **API Integration:** The integration of Rapid API and API Ninjas provides access to a wide range of third-party APIs for additional functions such as language translation, sentiment analysis, and entity recognition, enriching the capabilities of Lexpilot.

5. **Text processing:** Incorporating Lang Chain for text processing tasks enables advancednatural language processing features, including tokenization, stemming, and part-of- speech tagging, increasing the accuracy and depth of document analysis.

Technical Feasibility Summary:

The technical feasibility of the Lexpilot project is robust due to the comprehensive set of frontend and backend technologies used. React and Tailwind CSS make it easy to develop a modern and responsive user interface, while Clerk simplifies authentication and user management. Node.js with Express.js powers the backend and provides scalability and flexibility to handle document analysis requests. Integration with Rapid API, API Ninjas and Lang Chain enriches the application with advanced text processing and analysis capabilities. Overall, the combination of these technologies ensures that the lexpilot is technically feasible and capable of providing a powerful and user-friendly document analysis solution.

3.2 System Requirement Analysis

a) Functional Requirements

- **User Verification:** Users should be able to securely register, log in and log out using Clerkauthentication. Authentication mechanisms should ensure the privacy and security of userdata.

- **Document Upload:** Users should be able to upload documents in various formats including DOCX, PDF, TXT. The system should support uploading multiple documents atthe same time.

- **Text Extraction:** After uploading a document, the system should extract the text content from the documents while preserving the formatting and structure.

- **Content Analysis:** The application should perform content analysis using natural language processing techniques to identify key topics, themes, sentiments and entities in documents.

- **Keyword Detection:** Users should be able to specify keywords or phrases of interest andthe system should highlight occurrences of those keywords in the document.

a) Non-Functional Requirements

- **Performance:** Probe should be highly responsive, with minimal latency when processingdocument uploads, text extraction and analytics tasks, even under heavy user load. Response time for user interactions and generation of analysis results should be within acceptable limits to ensure a smooth user experience.

- **Scalability:** Probe should be designed to scale horizontally and vertically to accommodategrowing numbers of users and documents without degrading performance. Scalability should be achieved through efficient use of resources and load balancing mechanisms.

- **Security:** Probe should follow standard security procedures to protect user data and documents from unauthorized access, manipulation or disclosure. Encryption should be used for data transmission and storage.

- **Reliability:** Document Analyzer should be highly reliable, with minimal downtime and errors. The system should be fault tolerant, with built-in redundancy and failover mechanisms to ensure continuous operation.

Chapter–4: Scope of the Project

Objective
To develop an AI-powered system that simplifies the interaction with legal cases by providing intelligent search, summarization, and recommendation capabilities for legal professionals, students, and the general public.

Features

- **Legal Case Search Engine**:
  Enable users to search for cases using keywords, citations, or natural language queries.
- **Summarization of Cases**:
  Automatically generate concise summaries of lengthy legal cases to highlight key points and judgments.
- **Recommendation System**:
  Suggest relevant cases, statutes, or precedents based on user queries.
- **Question-Answering System**:
  Allow users to ask specific legal questions and receive accurate, context-aware responses.
- **Categorization and Tagging**:
  Classify legal cases into categories such as criminal, civil, corporate law, etc.
- **Document Upload and Parsing**:
  Allow users to upload legal documents for the system to analyze and extract key details.

Target Audience

- Legal professionals (lawyers, judges, paralegals)
- Law students and educators
- General public seeking legal insights

Limitations

- The system will focus on a specific jurisdiction or country's legal system (e.g., U.S. law, Indian law) to ensure depth and accuracy.
- It will provide legal information but not serve as a substitute for professional legal advice.
- Initial deployment will include limited datasets, with scope for expansion over time.

Technologies to be Used

- **Natural Language Processing (NLP)**: For understanding legal queries and summarizing case details.
- **Machine Learning Models**: To train and refine the search and recommendation systems.
- **Legal Databases and APIs**: Utilize publicly available legal datasets or APIs to provide accurate and comprehensive case law information.

Chapter–5: Working Methodology

- Task 1: (Data Acquistion: PDF_to_text) We have taken a sample of pdf provided in the dataset and converted into text. On running the python code in colab a file named output_text.txt is been generated where it transforms completely the pdf into the text.

- Task 2: (Chunking) Then we have installed spacy library and import it, further doing semantic chunking. Splitting texts into chunks or sentences. Further installed scikit-learn sentence-transformers after that KMeans chunking to convert text into more robust form. Then applying ML models such K-Means Clustering to group sentences into clusters, Created chunks by joining clustered sentences.

- Task 3: (Vectorization) There we choose an embedding model and convert each chunk into an embedding(vector). Then load a pre-trained model lightweight for handling large datasets. Then storing Embeddings in a Vector Database using ChromaDB to give unique id to chunks.

- Task 4: (Similarity Search) There we have converted user query into vectors. Using the query vector to search through the vector database to find most similar text embedding.

- Task 5: (Connect with LLM) In this we have finally created an Legal AI Assistant to handle legal cases by connecting to large language models. We have use libraries like langchain, streamlit, google-generativeai, langchain_google_genai.

Chapter – 6: System and Hardware Requirements

1. Hardware Requirements

- **Development Environment**:
  - Processor: Quad-core or higher (e.g., Intel i5/i7, AMD Ryzen 5/7)
  - RAM: Minimum 16 GB (32 GB recommended for handling large datasets)
  - Storage:
    - Minimum: 512 GB SSD for faster data access and model training
    - Additional space for storing datasets (e.g., 1–2 TB HDD for legal data archives)
  - GPU:
    - Required for training and deploying AI models (e.g., NVIDIA RTX 3060 or higher, 8 GB VRAM or more)
    - For basic model inference, a CPU-only system may suffice, but GPUs significantly accelerate performance.
- **Deployment Server (if hosted)**:
  - Processor: Octa-core server-grade processor (e.g., Intel Xeon, AMD EPYC)
  - RAM: 32–64 GB (depending on concurrent users)
  - Storage: SSD for fast I/O operations (minimum 1 TB for initial deployment)
  - GPU: High-performance GPUs for cloud or on-premise (e.g., NVIDIA A100, Tesla V100 for large-scale deployments).

2. Software Requirements

- **Development Tools**:
  - Programming Language: Python (for AI/ML/NLP).
  - Frameworks and Libraries:
    - NLP: SpaCy, NLTK, langchain, Streamlit.
    - Machine Learning: For applying K-Means Clustering, Chunking.
    - Frontend: HTML/CSS (for a web-based interface)
- Integrated Development Environment (IDE): VS Code, Google Colab

Chapter – 7: Listing Out Testing Technology

1. Unit Testing

   To test individual components like API calls, message history management, and text extraction.

   - **Tools**:
     - **pytest**: For writing and running Python test cases

2. Integration Testing

   To ensure proper interaction between components, such as the AI model, Streamlit UI, and the message history.

   - **Tools**:
     - **Postman**: For testing the backend API calls (if you create an external API wrapper for the legal assistant).
     - **pytest-mock**: For mocking responses from the AI model to test integration

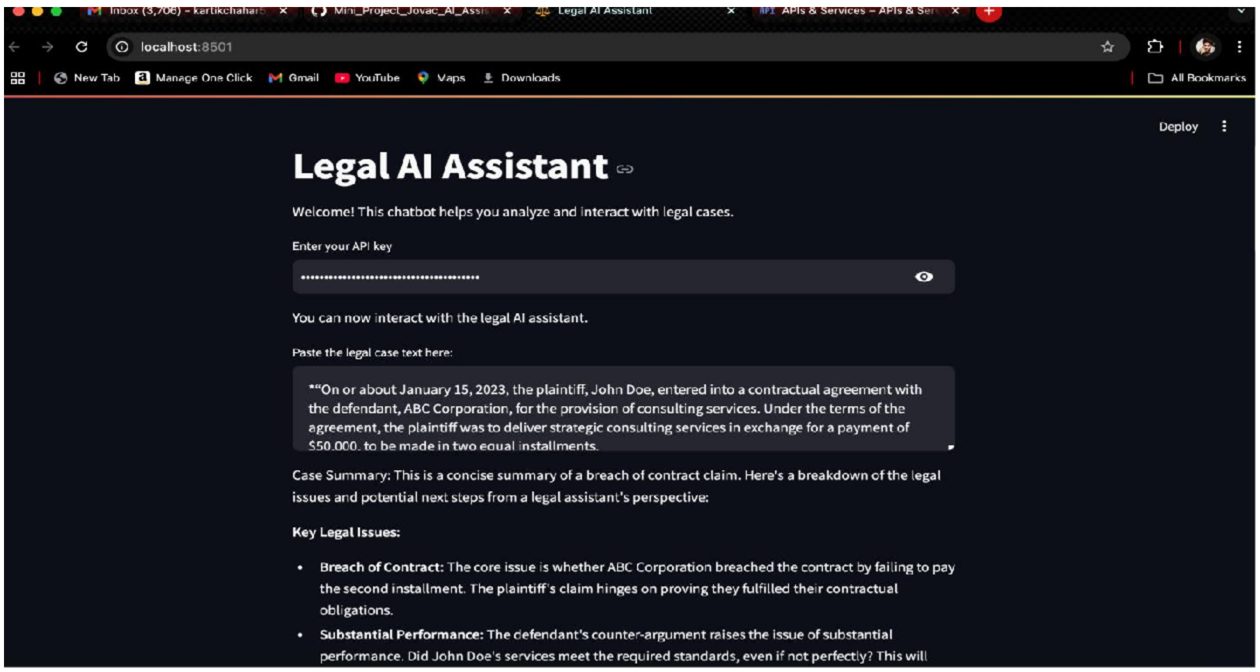
3. Performance Testing

   To ensure the application can handle real-time AI calls and user interactions without lag.

   - **Tools**:
     - **Apache JMeter**: For load testing the application and simulating multiple users.
     - **Locust**: To simulate concurrent users interacting with the Streamlit app.
     - **Streamlit Profiler**: Use built-in Python profiling libraries like cProfile to monitor performance bottlenecks.

Chapter – 8: Snapshots of Output

Snapshot 1:



Legal AI Assistant

Welcome! This chatbot helps you analyze and interact with legal cases.

Enter your API key

••••••••••••••••••••••••••••••••••••••

You can now interact with the legal AI assistant.

Paste the legal case text here:

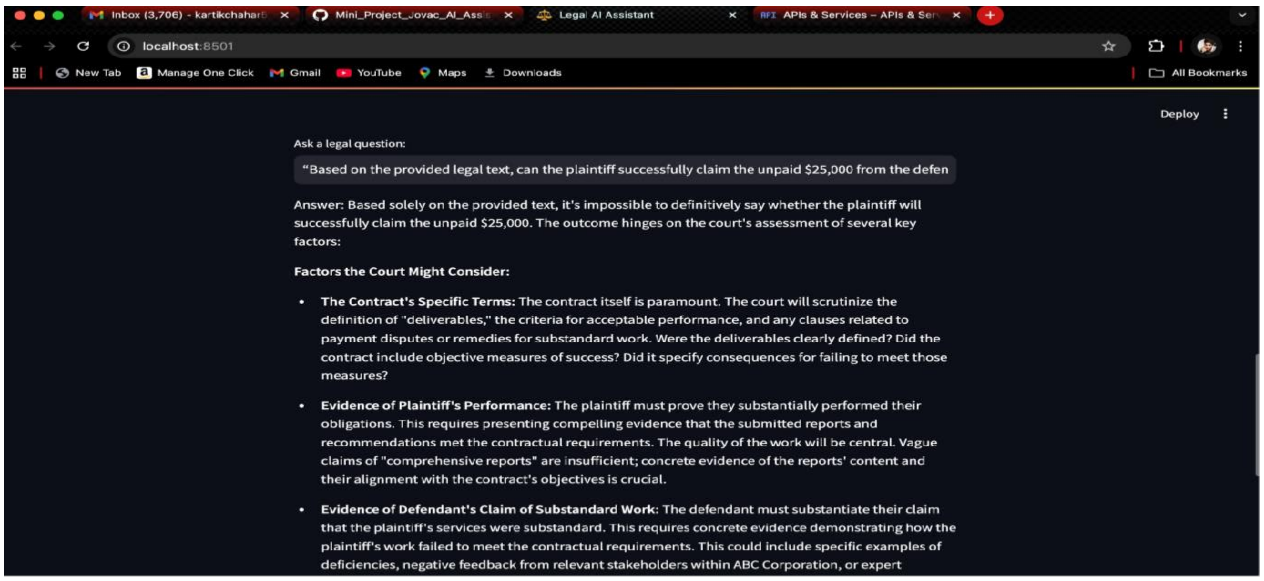
*"On or about January 15, 2023, the plaintiff, John Doe, entered into a contractual agreement with the defendant, ABC Corporation, for the provision of consulting services. Under the terms of the agreement, the plaintiff was to deliver strategic consulting services in exchange for a payment of $50,000, to be made in two equal installments.

Case Summary: This is a concise summary of a breach of contract claim. Here's a breakdown of the legal issues and potential next steps from a legal assistant's perspective:

Key Legal Issues:

- **Breach of Contract:** The core issue is whether ABC Corporation breached the contract by failing to pay the second installment. The plaintiff's claim hinges on proving they fulfilled their contractual obligations.
- **Substantial Performance:** The defendant's counter-argument raises the issue of substantial performance. Did John Doe's services meet the required standards, even if not perfectly? This will

Snapshot 2:



Ask a legal question:

"Based on the provided legal text, can the plaintiff successfully claim the unpaid $25,000 from the defen

Answer: Based solely on the provided text, it's impossible to definitively say whether the plaintiff will successfully claim the unpaid $25,000. The outcome hinges on the court's assessment of several key factors:

Factors the Court Might Consider:

- **The Contract's Specific Terms:** The contract itself is paramount. The court will scrutinize the definition of "deliverables," the criteria for acceptable performance, and any clauses related to payment disputes or remedies for substandard work. Were the deliverables clearly defined? Did the contract include objective measures of success? Did it specify consequences for failing to meet those measures?
- **Evidence of Plaintiff's Performance:** The plaintiff must prove they substantially performed their obligations. This requires presenting compelling evidence that the submitted reports and recommendations met the contractual requirements. The quality of the work will be central. Vague claims of "comprehensive reports" are insufficient; concrete evidence of the reports' content and their alignment with the contract's objectives is crucial.
- **Evidence of Defendant's Claim of Substandard Work:** The defendant must substantiate their claim that the plaintiff's services were substandard. This requires concrete evidence demonstrating how the plaintiff's work failed to meet the contractual requirements. This could include specific examples of deficiencies, negative feedback from relevant stakeholders within ABC Corporation, or expert

Chapter – 9: References

- https://numpy.org/doc/stable/
- https://seaborn.pydata.org/
- https://scikit-learn.org/stable/
- Smith, J., & Johnson, A. (2020). Leveraging APIs for Document Analysis inthe Digital Age. Journal of Information Technology, 30(2), 123-140
- Patel, S., et al. (2021). Enhancing Decision-Making with Automated Document Analysis: A Case Study. Journal of Business Analytics, 12(3),210-225
- https://www.geeksforgeeks.org/machine-learning-with-python/
- https://www.youtube.com/@campusx-official
- https://github.com/kacks77/Mini_Project_Jovac_AI_Assistant
- https://github.com/Harsh-Rajoria/Building-an-AI-System-for-Interacting-with-Legal-Cases..git