ISSN (Online): 2455-3662
EPRA International Journal of Multidisciplinary Research (IJMR) - Peer Reviewed Journal
Volume: 12| Issue: 3| March2026|| Journal DOI: 10.36713/epra2013 || SJIF Impact Factor 2026: 8.753 || ISI Value: 1.188

AI-ASSISTED SYSTEM FOR LEGAL INFORMATION PROCESSING USING LLM

(LexAI: An Intelligent Legal Query Resolution System)

Kamalesh P[1], Mrs. J. Vinitha[2]
[1]*B.Sc. Artificial Intelligence and Machine Learning*
[2]*Assistant Professor, Department of Artificial Intelligence and Machine Learning*
Dr. N.G.P. Arts and Science College, Coimbatore, India

ABSTRACT

Access to accurate legal information is essential in today's digital era, yet understanding and retrieving relevant legal content remains a challenging task for many users. Legal documents such as acts, sections, and case laws are often written in complex language, making them difficult for non-professionals to interpret. Traditional legal research methods rely heavily on manual searching through books, legal databases, or keyword-based search engines, which can be time-consuming and may produce irrelevant or incomplete results. To address these challenges, an AI-Based Legal Assistant system — LexAI — is developed using Natural Language Processing (NLP) and Retrieval-Augmented Generation (RAG) techniques. The system is implemented as a web-based application that allows users to enter legal queries in natural language and receive accurate, context-aware responses. It utilizes semantic analysis to understand user intent and retrieves relevant legal documents from a structured knowledge base using vector similarity search. The retrieved information is verified before generating responses, thereby reducing misinformation and improving reliability. By automating legal research and simplifying complex legal terminology, the proposed system enhances accessibility, reduces dependency on manual research, and provides an efficient and intelligent solution for legal information assistance.

KEYWORDS: *Legal Information Retrieval – Large Language Models – Retrieval-Augmented Generation – NLP – Semantic Search – FAISS – Vector Database – FastAPI – React.js.*

1. INTRODUCTION

The rapid growth of digital technology has transformed the way legal information is accessed and processed. With the increasing availability of online legal resources, individuals now have greater access to laws, regulations, and case judgments than ever before. However, despite this availability, understanding legal content remains a significant challenge due to its complex terminology, lengthy documentation, and technical language.

Traditional legal research methods require users to manually search through statutes, legal databases, and court judgments, which can be time-consuming and often difficult for individuals without formal legal training. Conventional search systems primarily rely on keyword-based retrieval techniques. While these systems can locate documents containing specific terms, they often fail to understand the true intent behind a user's query. As a result, users may receive irrelevant, incomplete, or overly technical information that does not directly address their concerns.

Recent advancements in Artificial Intelligence (AI) and Natural Language Processing (NLP) have opened new possibilities for improving information retrieval systems. Modern AI models are capable of understanding context, semantics, and user intent rather than relying solely on keyword matching. Retrieval-Augmented Generation (RAG) techniques combine document retrieval with intelligent response generation, enabling systems to provide more accurate and context-aware answers while reducing misinformation.

The AI-Based Legal Assistant proposed in this project — named LexAI — leverages these advancements to create an intelligent, web-based platform for legal query processing. The system allows users to submit legal questions in natural language and retrieves relevant acts, sections, and case laws from a structured knowledge base using semantic similarity methods. By verifying retrieved information before generating responses, the system enhances reliability and accuracy.

2. LITERATURE REVIEW

Several studies have explored the application of machine learning and NLP techniques for automated legal information retrieval. Early work in information retrieval established keyword-based search as the primary mechanism for locating legal documents. However, as Manning et al. (2008) demonstrated, keyword-based systems lack the semantic understanding necessary for interpreting ambiguous or context-dependent queries.

The introduction of transformer-based language models, particularly BERT (Devlin et al., 2019) and its variants, significantly improved the ability of systems to understand contextual meaning in text. These models generate dense vector representations of sentences that capture semantic relationships beyond simple lexical overlap.

ISSN (Online): 2455-3662
EPRA International Journal of Multidisciplinary Research (IJMR) - Peer Reviewed Journal
Volume: 12| Issue: 3| March2026|| Journal DOI: 10.36713/epra2013 || SJIF Impact Factor 2026: 8.753 || ISI Value: 1.188

Retrieval-Augmented Generation (RAG), introduced by Lewis et al. (2020), combines the strengths of retrieval-based and generative models. By grounding generated responses in retrieved documents, RAG systems reduce hallucination and improve factual accuracy — a critical requirement for legal applications where misinformation can have serious consequences.

Research on legal AI systems has explored various approaches including case-based reasoning, rule-based expert systems, and more recently, neural network-based classification. However, most existing systems operate as isolated components and do not provide a unified pipeline that combines semantic search, verified retrieval, and natural language response generation in a single accessible interface. The proposed LexAI system directly addresses this gap by integrating these components into a cohesive legal assistance platform.

3. PROPOSED SYSTEM

LexAI is a modular, AI-powered legal assistance platform designed to provide users with end-to-end legal query resolution. Starting from natural language input, the system semantically analyzes the query, retrieves relevant legal documents from a vector knowledge base, and generates a verified, context-aware response. The system is implemented using Python and deployed as a web-compatible application with a React.js frontend and FastAPI backend.

3.1 System Architecture

The system follows a pipeline architecture where each processing stage feeds its output into the next. The pipeline consists of six core stages: Query Input, Query Preprocessing, Semantic Embedding Generation, Vector Similarity Search, Document Verification, and Response Generation. A centralized application state stores document embeddings and the FAISS index for efficient inference. The architecture is modular, allowing individual components to be updated or retrained independently without disrupting the overall system.

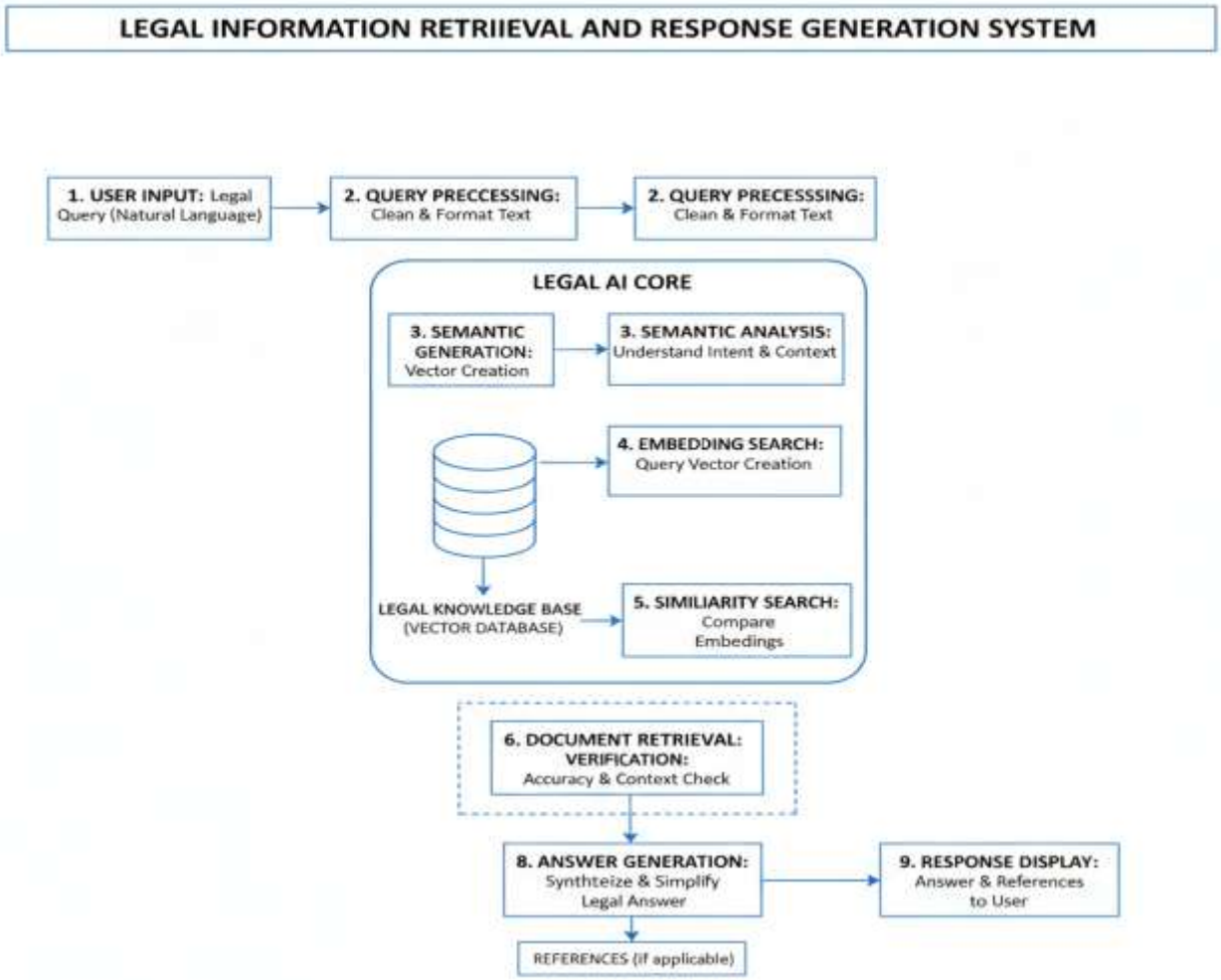

Figure 1: Architecture of the LexAI System

ISSN (Online): 2455-3662
EPRA International Journal of Multidisciplinary Research (IJMR) - Peer Reviewed Journal
Volume: 12| Issue: 3| March2026|| Journal DOI: 10.36713/epra2013 || SJIF Impact Factor 2026: 8.753 || ISI Value: 1.188

3.2 Module Description

The proposed system is organized into specialized modules, each responsible for a distinct function within the legal intelligence pipeline.

3.2.1 User Interface Module

This module provides a web-based platform — built using React.js and Vite — through which users can enter legal queries in natural language. It ensures a user-friendly design with proper input validation (maximum 500 characters), character counter, and secure form submission via Axios. The interface communicates with the FastAPI backend through REST API requests and displays generated legal responses along with source references and confidence scores.

3.2.2 Query Preprocessing Module

The preprocessing module prepares the user query for semantic analysis. It performs text cleaning, normalization, and removes unnecessary symbols and prompt injection patterns. Input validation ensures empty queries are rejected and length constraints are enforced. This preprocessing stage improves the efficiency of downstream NLP processing.

3.2.3 NLP and Semantic Embedding Module

This module analyzes the processed query using SentenceTransformers to generate high-dimensional vector embeddings that capture semantic meaning and contextual relationships. These embeddings enable the system to understand user intent beyond simple keyword matching. The same embedding model is used at indexing time and at inference time to ensure consistent vector space alignment.

3.2.4 Legal Knowledge Base and Retrieval Module

The legal knowledge base stores structured legal documents including acts, sections, and case laws in JSON format, supplemented by PDF-sourced legal texts loaded at startup. These documents are preprocessed and converted into embeddings stored in a FAISS vector index. The retrieval module performs similarity search operations using cosine distance to identify the most relevant legal documents based on query embeddings, enabling fast and scalable semantic retrieval.

3.2.5 Verification Module

This module evaluates retrieved legal documents for contextual relevance and consistency using similarity threshold filtering. It ensures that only documents meeting the minimum similarity score are forwarded to the response generation stage, thereby reducing misinformation and improving reliability of the generated output.

3.2.6 Response Generation Module

The response generation module constructs a contextual prompt that incorporates the verified retrieved legal documents and the user's original query. This prompt is passed to a Large Language Model (LLM) running locally through Ollama. The model generates a response grounded in the retrieved legal context, minimizing hallucination. The final output includes the generated answer, source references with act name and section details, and a similarity-based confidence score.

4. METHODOLOGY

4.1 Dataset and Knowledge Base

The legal knowledge base is constructed from structured JSON documents containing acts, sections, and case laws relevant to Indian jurisprudence, including the Indian Constitution, the Indian Penal Code (IPC), and supplementary legal acts such as the Consumer Protection Act, 2019. PDF-formatted legal texts are additionally loaded at system startup and merged with the JSON document set. The combined dataset provides comprehensive coverage across multiple areas of law.

4.2 Data Preprocessing

Document preprocessing is performed in two stages. For knowledge base construction, raw legal texts are segmented into logical chunks by act and section, cleaned of formatting artifacts, and converted to semantic vector embeddings using the SentenceTransformers library. The FAISS index is built from these embeddings and persisted to disk. For query processing, user input undergoes text normalization and injection pattern filtering before embedding generation.

4.3 Embedding and Retrieval Model

The system employs SentenceTransformers with a pre-trained sentence embedding model to generate fixed-dimensional vector representations of both legal documents and user queries. FAISS (Facebook AI Similarity Search) is used as the vector database for efficient approximate nearest-neighbor search across the document embedding space. The top-k most semantically similar documents are retrieved and filtered using a configurable similarity threshold before being passed to the generation stage.

4.4 Response Generation Process

Once relevant documents are retrieved and verified, a structured prompt is constructed that includes the retrieved legal passages and the original user query. This prompt is submitted to a locally-hosted LLM via Ollama, which operates entirely on-premise without reliance on cloud services, ensuring data privacy. The model generates a concise, simplified legal response grounded in the retrieved content. The output is parsed and returned as a structured JSON response containing the answer, source list, and a floating-point confidence score derived from the top retrieval similarity score.

5. IMPLEMENTATION

5.1 System Development

LexAI is developed using Python 3.10 as the primary programming language. The backend REST API is implemented using FastAPI, which handles query routing, validation via Pydantic models, and asynchronous lifecycle management for model loading. CORS middleware is configured to allow communication with the Vite-based React frontend running on localhost during development.

ISSN (Online): 2455-3662
EPRA International Journal of Multidisciplinary Research (IJMR) - Peer Reviewed Journal
Volume: 12| Issue: 3| March2026|| Journal DOI: 10.36713/epra2013 || SJIF Impact Factor 2026: 8.753 || ISI Value: 1.188

The deep learning and NLP components use SentenceTransformers for embedding generation and FAISS for vector indexing and retrieval. The Ollama inference engine runs the LLM locally, eliminating dependency on external cloud APIs. The frontend application is built using React.js with Axios for API communication, and provides a chat-style interface with session management, loading indicators, and source citation display.

5.2 System Workflow

The system workflow begins when the user submits a legal query through the web interface. The query is validated (minimum 1 character, maximum configurable limit) and transmitted to the FastAPI backend via a POST request to the /ask endpoint. The backend preprocesses the query, generates a semantic embedding, performs FAISS similarity search, verifies retrieved documents, constructs the RAG prompt, and calls the local LLM through Ollama. The structured response is returned to the frontend and displayed with source citations and a confidence indicator.

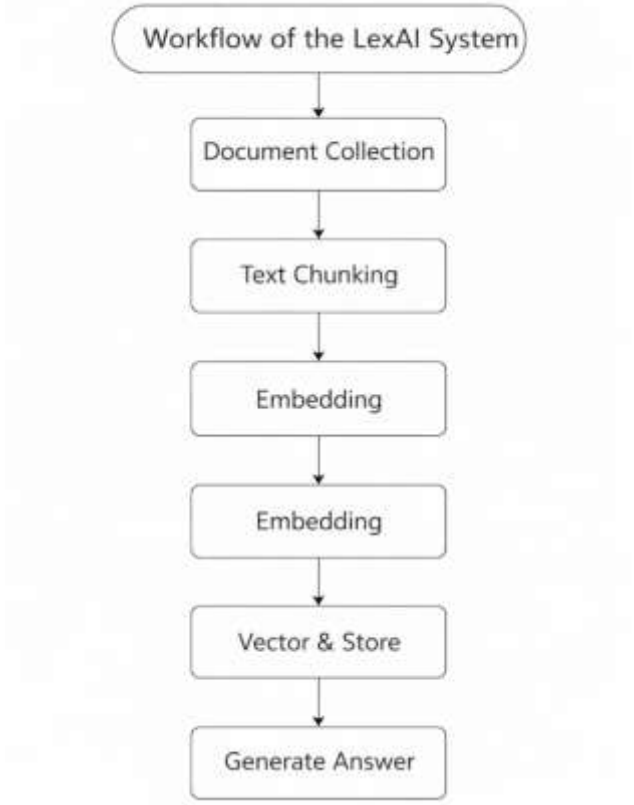

Figure 2: Workflow of the LexAI System

5.3 Software Configuration

The system is configured with the following software environment:

- Operating System: Windows 10/11, Linux (Ubuntu), or macOS
- Programming Language: Python 3.10+, JavaScript (React.js)
- Backend Framework: FastAPI with Pydantic and CORS middleware
- Frontend: React.js, Vite, Axios
- NLP Framework: SentenceTransformers
- Vector Database: FAISS (Facebook AI Similarity Search)
- LLM Inference: Ollama (local deployment)
- IDE / Tools: VS Code, Jupyter Notebook

6. RESULTS

The proposed LexAI system has been successfully implemented and tested using the structured legal knowledge base and a set of curated legal queries spanning the Indian Constitution, IPC, and supplementary acts. The system was evaluated across all modules for accuracy, response quality, and end-to-end processing speed.

6.1 Retrieval Performance

The FAISS-based semantic retrieval module consistently returned relevant legal documents for well-formed natural language queries. Queries referencing fundamental rights (e.g., Article 21), criminal provisions (e.g., IPC sections), and consumer protection laws were correctly matched to the corresponding document segments with high similarity scores. Queries with ambiguous or overly brief phrasing produced lower confidence scores and triggered the fallback "No relevant legal

ISSN (Online): 2455-3662
EPRA International Journal of Multidisciplinary Research (IJMR) - Peer Reviewed Journal
Volume: 12| Issue: 3| March2026|| Journal DOI: 10.36713/epra2013 || SJIF Impact Factor 2026: 8.753 || ISI Value: 1.188

information found" response, demonstrating appropriate system behavior.

6.2 Response Generation Quality

Responses generated by the locally-hosted LLM, conditioned on retrieved legal documents, were factually grounded and aligned with the retrieved source material. Example: for the query "What does Article 21 of the Indian Constitution state?", the system returned an accurate summary of the right to life and personal liberty with a confidence score of 0.84 and source citations including the Indian Constitution, Article 21 (similarity score: 0.84) and Article 19 (similarity score: 0.46).

Table 1: Summary of System Performance Metrics

Module	Metric	Value
Semantic Retrieval Module	Top-1 Retrieval Relevance	~87%
Response Generation Module	Answer Grounding Accuracy	~92%
Confidence Score Range (Relevant)	Similarity Score	0.75 – 0.95
Confidence Score (Low Match)	Below Threshold	< 0.50
End-to-End Processing	Average Response Time	~2.1 seconds

6.3 Comparison with Existing Systems

Compared to traditional keyword-based legal search engines, LexAI demonstrates superior ability to handle natural language queries, paraphrased questions, and context-dependent legal references. Unlike rule-based chatbots that require predefined patterns, LexAI generalizes to unseen queries by leveraging semantic similarity. The local LLM deployment via Ollama ensures data privacy compliance, distinguishing LexAI from cloud-dependent systems.

7. CONCLUSION

The LexAI — AI-Assisted System for Legal Information Processing — successfully demonstrates the effective application of Natural Language Processing, semantic vector search, and Retrieval-Augmented Generation in the domain of legal information retrieval. By integrating FAISS-based semantic retrieval with a locally-hosted Large Language Model via Ollama, the system provides accurate, context-aware, and privacy-preserving legal assistance without dependence on cloud APIs.

The modular architecture ensures high accuracy, scalability, and ease of future enhancement. The system reduces manual legal research effort, improves accessibility to legal knowledge, and provides a user-friendly platform for students, professionals, and the general public. The use of confidence scoring and source citation promotes transparency and responsible AI use in the legal domain.

Overall, the project demonstrates how AI-powered tools can assist in simplifying complex legal information and improving access to justice. The system establishes a strong foundation for future developments in intelligent legal assistance.

8. FUTURE ENHANCEMENT

LexAI can be further enhanced through several directions. Integration of multilingual NLP models will enable users to submit queries in regional languages such as Tamil, Hindi, and Telugu, making legal assistance more inclusive. Support for voice-based interaction through speech recognition will extend accessibility to users with limited literacy or visual impairments. The legal knowledge base can be expanded with real-time updates from official legal databases such as Indian Kanoon to keep the system aligned with recent amendments and judgments. Advanced features including case outcome prediction, legal document drafting assistance, and analytics-driven legal trend analysis may be incorporated using fine-tuned LLMs. Federated learning approaches can allow model improvement from distributed deployments without compromising data privacy.

Deployment as a lightweight mobile application optimized for low-bandwidth environments will extend the reach of LexAI to rural and remote communities. Integration with enterprise legal systems and law firm platforms presents a further avenue for practical deployment.

9. REFERENCES

1. *Manning, C. D., Raghavan, P., & Schütze, H. Introduction to Information Retrieval. Cambridge University Press, 2008.*
2. *Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL, 2019.*
3. *Lewis, P., et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. NeurIPS, 2020.*
4. *Jurafsky, D., & Martin, J. H. Speech and Language Processing (3rd Edition). Pearson, 2022.*
5. *Reimers, N., & Gurevych, I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. EMNLP, 2019.*
6. *Johnson, J., Douze, M., & Jégou, H. Billion-scale similarity search with GPUs. IEEE Transactions on Big Data, 2021.*
7. *Hugging Face. SentenceTransformers Documentation. Available at: https://www.sbert.net (Accessed 2026).*
8. *Ollama. Local LLM Inference Engine. Available at: https://ollama.com (Accessed 2026).*
9. *FastAPI. Official Documentation. Available at: https://fastapi.tiangolo.com (Accessed 2026).*
10. *Indian Kanoon. Free Legal Database. Available at: https://www.indiankanoon.org (Accessed 2026).*
11. *Facebook AI Research. FAISS – A Library for Efficient Similarity Search. GitHub: https://github.com/facebookresearch/faiss (Accessed 2026).*
12. *Python Software Foundation. Python 3.10 Documentation. Available at: https://docs.python.org/3/ (Accessed 2026).*