JURI AI: AI System for Legal Assistance and Lawyer Appointment Management

Sanika Tikole[1], Aditya Jare[2], Rajendra Margale[3], Faizal Mistry[4]
[1,2,3,4]*P. E S MCOE*

***Abstract*— However, the issue of inaccessible legal aid services still remains one of the major obstacles in India, as the cost of consultation, complicated legal procedures, and lack of professional advice make things challenging. The current research paper introduces a new technology solution known as JURI AI – an AI-based LegalTech tool combining the functionalities of Retrieval-Augmented Generation (RAG), contract analysis, legal risk evaluation, and scheduling lawyer appointments into one ecosystem. In particular, the application utilizes semantic search powered by Pinecone Vector Database, contract clause classification with machine learning algorithms, legal validation rules for contracts according to Indian regulations, and a large language model for answering legal questions. The experimental assessment of contract classification produced 89.42% of accuracy rate, whereas rule-based validation was able to detect risks and missing clauses of the contract.**

***Index Terms*—LegalTech, Retrieval-Augmented Generation, Legal-BERT, NLP, Contract Analysis, Artificial Intelligence, Indian Law**

I. INTRODUCTION

Access to legal services in India remains a significant challenge because of high consultation fees, delayed legal procedures, and limited access to qualified legal professionals. Millions of citizens struggle to obtain legal guidance Getting legal guidance in India can still be difficult for many people due to high costs and delayed legal procedures. Technologies based on Artificial Intelligence now make it possible to support users with legal information, document analysis, and faster access to assistance. JURI AI is developed as an AI-powered legal assistance platform that combines semantic retrieval, transformer-based reasoning, and legal document analysis.

The platform provides:

- AI-based legal question answering
- Contract analysis and risk detection
- Lawyer appointment scheduling
- Legal document assistance
- User-friendly web interface

The primary goal of JURI AI is to make legal support affordable, scalable, and accessible while reducing the workload of legal professionals.

II. RELATED WORK

Rule-based expert systems formed the backbone of traditional legal assistance systems, which were devoid of context and flexibility. The rise of transformer models like BERT and GPT helped enhance semantic comprehension in legal NLP systems. LEGAL-BERT represents a domain-specific language model trained on a legal corpus that includes statues, contractual documents, and judgments. LEGAL-BERT surpasses BERT on many tasks pertaining to the legal domain, such as clause classification and legal entailment. The use of retrieval-augmented generation in NLP enhances factual accuracy by integrating semantic retrieval with generation models. Unlike other models that generate responses based on model memory alone, RAG retrieves context-related data first before proceeding to generate a response. Current solutions, such as LawGeex and Casetext, concentrate mostly on Western legal systems. JURI AI provides an Indian solution via statutory risk validation and expansion to multiple languages.

III. SYSTEM ARCHITECTURE

JURI AI follows a modular three-tier architecture consisting of frontend, backend, AI processing, and database components.

3.1. Frontend Layer

The frontend layer of JURI AI is developed using React.js and TypeScript, providing a responsive and user-friendly interface for different stakeholders, including users, lawyers, and administrators. The frontend serves as the primary interaction point between users and the system and enables access to all major functionalities of the platform.

The user interface supports legal query submission, contract upload and analysis, lawyer profile exploration, appointment scheduling, and legal document generation. A dedicated user dashboard allows users to interact with the AI-powered legal chatbot, upload contracts for review, view risk analysis reports, and book appointments with registered legal professionals.

Lawyers are provided with a dedicated dashboard through which they can manage their professional profiles, view and respond to appointment requests, maintain availability schedules, and interact with clients. This functionality enables efficient appointment management and improves communication between legal professionals and users. Administrators are provided with system management capabilities, including monitoring user activities, managing lawyer registrations, supervising appointments, and maintaining platform resources. Administrative controls help ensure secure and efficient operation of the system.

The frontend communicates with the backend services through REST APIs and is designed to provide a seamless user experience across different devices and screen sizes. The responsive design and intuitive navigation improve accessibility and usability for all categories of users.

3.2. Backend Layer

The backend layer of JURI AI is implemented using Node.js and Express.js and serves as the central communication component between the frontend interface and AI-powered services. It is responsible for processing user requests, managing application workflows, and coordinating interactions among various system modules.

The backend exposes RESTful APIs for user authentication, lawyer management, appointment scheduling, contract processing, and legal document generation. These APIs facilitate secure and efficient communication between the frontend application and backend services.

To ensure secure access control, JWT (JSON Web Token) based authentication is implemented throughout the system.

The platform supports three distinct user roles:

1. User
2. Lawyer
3. Administrator

Role-based authorization mechanisms are used to restrict access to system resources according to the privileges associated with each role. Users can access legal assistance services, upload contracts, generate legal documents, and schedule appointments. Lawyers can manage professional profiles, appointment schedules, and client interactions, while administrators are responsible for monitoring platform activities and managing system resources.

The backend also manages appointment workflows, user accounts, lawyer availability schedules, document storage operations, and communication with the AI backend. MongoDB Atlas is utilized for storing structured application data including user profiles, lawyer profiles, appointment records, uploaded contracts, generated legal templates, and system-related information.

By acting as an integration layer between the frontend, AI services, and database systems, the backend ensures reliable operation, scalability, and efficient data management across the JURI AI platform.

3.3. AI Backend Layer

The AI backend of JURI AI is implemented using FastAPI and serves as the core intelligence layer of the platform. It consists of two primary components: the Legal RAG Chatbot and the Contract Review Engine. Additionally, the platform provides a Legal Document Repository that allows users to access and download commonly used legal templates.

3.3.1.Legal RAG Chatbot

The Legal RAG Chatbot is based on a Retrieval-Augmented Generation (RAG) architecture designed to provide context-aware and reliable responses to legal queries. Legal documents are segmented into chunks using RecursiveCharacterTextSplitter with a chunk size of 800 characters and an overlap of 100 characters. The generated chunks are converted into

semantic embeddings using the sentence-transformers/all-mpnet-base-v2 model and stored in Pinecone Vector Database using cosine similarity indexing.

When a user submits a legal query, the query is converted into an embedding and matched against stored document embeddings. The legal contexts that are most relevant are obtained and fused together with the user's question via LangChain RetrievalQA and then presented to the Groq-hosted LLaMA 3.3 70B model. Such an approach ensures greater accuracy of answers and lower hallucination rates due to grounding in the context.

3.3.2.Contract Review Engine

The Contract Review Engine is an automated tool that performs text analysis of legal contracts that are provided in the formats PDF and DOCX. The text in these documents is extracted and segmented into clauses. Afterward, each clause is categorized through a Legal-BERT model trained on a dataset of legal contract clauses. The categorizer classifies the clauses into different categories, which may include clauses related to confidentiality, liability, jurisdiction, termination, indemnification, and payment among others. After classification, the clauses are checked by an Indian legal rule-based validator for compliance with pre-defined legal rules. These risk scores are determined based on the confidence of the classifier and legal validation results.

3.3.3.Legal Document Repository

The Legal Document Repository provides users with access to commonly used legal templates and reference documents. Users can browse, search, and download legal documents such as Non-Disclosure Agreements (NDAs), Memorandums of Understanding (MoUs), rental agreements, employment agreements, and service agreements.

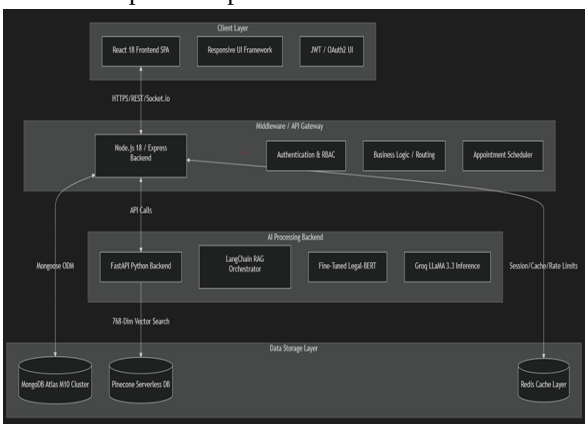
Document metadata is maintained within MongoDB Atlas, allowing efficient categorization, retrieval, and management of templates. Administrators can upload, update, approve, and manage available documents within the repository. This functionality improves access to legal resources while reducing the effort required to locate commonly used legal forms and agreements.

3.4. Database Layer

JURI AI employs a hybrid database architecture consisting of MongoDB Atlas and Pinecone Vector Database to support both transactional operations and semantic retrieval tasks.

MongoDB Atlas serves as the primary database for storing structured application data, including user profiles, lawyer profiles, appointment records, uploaded contract metadata, legal document repository information, chatbot interaction history, authentication details, and administrative records. In addition, the database stores predefined legal validation rules, risk assessment configurations, clause categories, and compliance conditions used by the contract review engine for evaluating legal agreements against Indian legal requirements.

Pinecone Vector Database is utilized for storing semantic embeddings generated from legal documents processed by the Retrieval-Augmented Generation (RAG) pipeline. Legal document chunks are converted into 768-dimensional vector representations using the sentence-transformers/all-mpnet-base-v2 embedding model and indexed using cosine similarity. Such embeddings make semantic search possible and provide an efficient method for fetching related information in legal matters. The use of MongoDB Atlas and Pinecone provides an efficient means for handling structured application workflow logic, validation logic, and unstructured legal data. MongoDB handles transactional data, appointment records, user information, and legal compliance rules, while Pinecone supports semantic retrieval operations required for legal question answering. This hybrid architecture improves retrieval accuracy, scalability, and overall platform performance.



Structural Architecture Diagram of JURI AI

IV. METHODOLOGY

This paper proposes an innovative JURI AI platform with the help of RAG, Legal BERT model for legal document analysis, role-based user management system, and appointment management system for lawyers. Methodology includes document ingesting, semantic search, legal questions and answers, contract analysis, legal rule validity check, and lawyer appointment booking.

4.1. Document Ingestion

The document ingestion module is responsible for processing legal documents before they are indexed and utilized by the AI components. The system supports PDF and DOCX document formats, allowing users to upload contracts, agreements, legal notices, and reference documents.

The documents that have been uploaded go through several pre-processing procedures, such as extraction of text, cleaning, normalization, and segmentation. The PDF documents are preprocessed by using PyMuPDF, whereas the DOCX documents are preprocessed by using python-docx. This will prepare the extracted text for indexing and retrieval.

The extracted text is segmented using RecursiveCharacterTextSplitter with a chunk size of 800 characters and an overlap of 100 characters. The overlap mechanism preserves contextual continuity between adjacent chunks and improves retrieval quality. The generated chunks are subsequently converted into semantic embeddings and stored in Pinecone Vector Database for future retrieval operations.

4.2. Retrieval-Augmented Question Answering

The legal chatbot employs a Retrieval-Augmented Generation (RAG) framework to improve factual accuracy and reduce hallucination during response generation.

Legal document chunks are converted into 768-dimensional semantic embeddings using the sentence-transformers/all-mpnet-base-v2 model. These embeddings are indexed within Pinecone Vector Database using cosine similarity.

When a user submits a legal query, the query is transformed into an embedding using the same model and compared against stored document vectors.

$$\text{Similarity}(Q, D) = \frac{Q \cdot D}{\| Q \| \| D \|}$$

where Q represents the query embedding and D represents the document embedding.

The top-k relevant document chunks are retrieved using cosine similarity search and combined with the user query through LangChain RetrievalQA. The retrieved context is then forwarded to the Groq-hosted LLaMA 3.3 70B language model for response generation. This approach ensures that generated responses are grounded in retrieved legal information rather than relying solely on the language model's internal knowledge.

4.3. Clause Classification and Risk Detection

The contract review module automates the analysis of legal contracts uploaded in PDF and DOCX formats. The uploaded document undergoes text extraction and clause segmentation to identify individual contractual provisions.

Each extracted clause is classified using a Legal-BERT model. The classifier identifies various legal clause categories including confidentiality clauses, liability clauses, termination clauses, jurisdiction clauses, indemnity clauses, and payment-related clauses.

Following classification, the clauses are evaluated using a rule-based legal validation engine. Risk scores are assigned based on classification confidence and legal validation outcomes. Clauses identified as potentially risky are included in the generated contract review report along with risk scores, detected issues, and compliance recommendations, enabling users to quickly identify sections requiring legal attention.

4.4. Legal-BERT Fine-Tuning

To improve contract clause classification performance, the Legal-BERT model (nlpaueb/legal-bert-base-uncased) was fine-tuned using a legal contract clause dataset obtained from Kaggle.

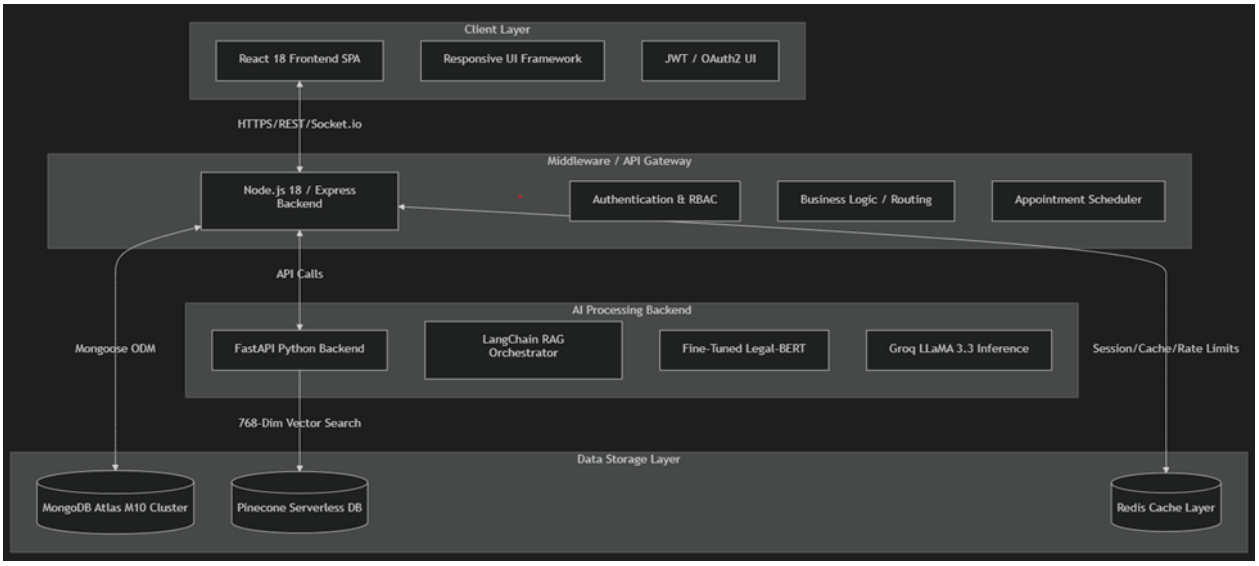
The dataset contains annotated legal clauses belonging to multiple categories such as Confidentiality, Liability, Jurisdiction, Termination, Payment Terms, and Indemnity. Prior to training, the dataset was preprocessed by removing null values, normalizing text, and encoding clause labels into numerical categories.

The dataset was divided into training and testing subsets using an 80:20 split ratio. Fine-tuning was

performed using the AdamW optimizer with a learning rate of 2×10^{-5}, batch size of 8, and three training epochs. Model performance was evaluated using Accuracy, Precision, Recall, and F1-Score metrics.
The fine-tuned model was subsequently integrated into the contract review engine to improve clause classification and risk assessment performance.

4.5. Legal Rule Validation Engine
In addition to machine learning-based classification, the contract review module incorporates a rule-based legal validation engine designed to evaluate contractual provisions against Indian legal principles.
The validation engine contains predefined legal rules related to confidentiality clauses, indemnity clauses, arbitration provisions, jurisdiction clauses, liability limitations, termination conditions, non-compete agreements, and data privacy requirements. These rules are derived from Indian legal frameworks including the Indian Contract Act, 1872 and other applicable regulations.
During contract analysis, classified clauses are evaluated against these predefined legal rules. Any detected violations, ambiguities, or potentially unfavorable provisions are flagged and included in the final risk assessment report. The predefined legal validation rules and compliance conditions are maintained within the system repository and are utilized during contract analysis to support consistent legal risk assessment.

4.6. User and Appointment Management
The platform supports role-based access control for Users, Lawyers, and Administrators. JWT-based authentication is implemented to ensure secure access to system resources and functionalities.
Users can access legal assistance services, upload contracts, access and download legal document templates from the template repository, and schedule appointments with lawyers. Lawyers can manage professional profiles, maintain availability schedules, and respond to appointment requests. Administrators are responsible for monitoring platform activities, managing legal document templates, and supervising system resources.
Appointment records, user profiles, lawyer information, and system metadata are stored within MongoDB Atlas. The appointment management system prevents scheduling conflicts and supports efficient coordination between users and legal professionals.



RAG Query Processing Workflow of JURI AI

V. EXPERIMENTAL RESULTS

5.1. Evaluation Setup
The contract classification module was evaluated using the Kaggle Legal Contract Dataset (changethan/legal-contract-dataset). The dataset contains 8,628 annotated legal clauses belonging to 47 clause categories. An 85:15 stratified train-test split was used for model training and evaluation.

The implemented classifier utilizes TF-IDF feature extraction combined with an SGDClassifier. Model performance was evaluated using Accuracy, Precision, Recall, and F1-Score metrics.

The rule-based legal validation engine was evaluated separately using a synthetic contract evaluation dataset consisting of 120 contract scenarios containing predefined risk levels, expected clauses, and missing clause annotations.

5.2. Contract Clause Classification Results

Table 1 presents the performance of the trained contract clause classification model.

Contract Clause Classification Results

Metric	Value
Accuracy	89.42%
Precision (Macro)	81.14%
Recall (Macro)	81.03%
F1-Score (Macro)	81.03%

The results indicate that the classifier effectively categorizes legal clauses while maintaining balanced performance across multiple clause categories.

5.3. Risk Detection Evaluation

The rule-based legal validation engine was evaluated on 120 contract scenarios. The obtained results are shown in Table 2.

Risk Detection Evaluation Results

Metric	Value
Risk Level Accuracy	100%
Missing Clause Precision	100%
Missing Clause Recall	87.5%
Key Clause Precision	86.11%
Key Clause Recall	100%

The evaluation demonstrates the effectiveness of the rule-based validation engine in identifying contractual risks, detecting missing clauses, and generating compliance recommendations aligned with Indian legal principles.

5.4. Contract Review Output

For each uploaded contract, the system generates:

- Overall risk level (Low, Medium, High)
- Numerical risk score
- Detected key clauses
- Missing clauses
- Legal observations
- Compliance recommendations

These outputs assist users in understanding contractual risks and identifying provisions that may require legal review.

5.5. Discussion

The experimental results demonstrate that combining machine learning-based clause classification with rule-based legal validation provides effective support for contract review and legal risk assessment. The current evaluation focuses on contract analysis functionality. Future work will include comprehensive quantitative evaluation of the Retrieval-Augmented Generation (RAG) chatbot using annotated legal question-answer datasets and benchmark evaluation frameworks.

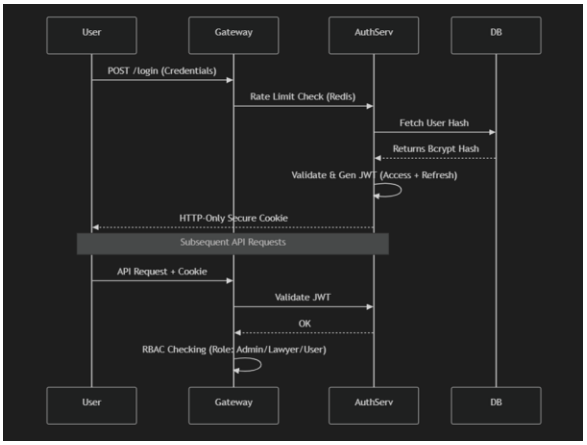
VI. SECURITY AND COMPLIANCE

The platform implements multiple security mechanisms:

- JWT Authentication
- AES-256 Encryption
- TLS 1.3 Security
- Rate Limiting
- Secure Cookies
- GDPR and DPDPA Compliance

Passwords are hashed using bcrypt with a cost factor of 12.

The system is designed as an assistive technology and does not replace professional legal counsel. Human oversight is maintained for critical legal decisions to ensure ethical and responsible AI usage.



Security Role

VII. LIMITATIONS

Although JURI AI demonstrates strong performance, several limitations remain:

- English-only support
- Dependency on third-party APIs
- Limited Indian case-law exposure
- Context window limitations for long documents

VIII. FUTURE WORK

Future improvements include:

- IndicBERT integration for Hindi, Marathi, and Tamil support
- Mobile application deployment
- Supreme Court and High Court judgment database integration
- Explainable AI modules
- Personalized legal assistant features

IX. CONCLUSION

In this paper, the concept of JURI AI was introduced as an integrated AI-powered legal aid system that involves Retrieval-Augmented Generation (RAG), contract analysis, legal risks identification, and lawyer appointments management. The solution incorporates semantic search powered by Pinecone vector database, large language models for legal question answering, machine learning for contract clause categorization, and a rules-based system for legal compliance according to Indian laws.

The experimental test of the contract classifier module showed satisfactory results, and rule-based validation engine efficiently detected risks in contracts as well as missed clauses. The architecture, implemented on React.js, Node.js, FastAPI, MongoDB Atlas, and Pinecone, provides an efficient integration environment for implementing legal services.

JURI AI aims at helping users and lawyers through increasing the accessibility of legal resources, contract analysis, and appointment management. Further improvements should include multilingual support, expanding of knowledge bases, and testing on a larger number of legal samples.

ACKNOWLEDGMENT

We sincerely acknowledge the guidance and support provided by Mrs. Bhavna Chaudhari and Ms. Gauri Mathad, Department of Artificial Intelligence and Machine Learning, PES Modern College of Engineering, Pune during the completion of this project.

REFERENCES

[1] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” *arXiv preprint arXiv:1810.04805*, 2018.

[2] I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos, “LEGAL-BERT: The Muppets Straight Out of Law School,” arXiv preprint arXiv:2010.02559, 2020.

[3] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” arXiv preprint arXiv:2005.11401, 2020.

[4] Pinecone, “Vector Database for AI,” 2024. [Online]. Available: https://www.pinecone.io/

[5] Groq, “LLaMA 3.3 Model Overview,” 2025. [Online]. Available: https://www.groq.com/

[6] S. Agarwal, “AI in Legal Tech: Trends and Applications,” IEEE Access, 2023.

[7] Meta AI, “Llama 3: Open Foundation and Instruction Models,” 2024. [Online]. Available: https://ai.meta.com/llama/

[8] A. Kakwani, A. Kunchukuttan, S. Golla, N. N. Bhattacharyya, M. Khapra, and P. Kumar, “IndicNLPSuite: Multilingual Language Models and Tools for Indian Languages,” Findings of the Association for Computational Linguistics: EMNLP 2020, pp. 4948–4961, 2020.

[9] LangChain, “Building Applications with LLMs through Composability,” 2024. [Online]. Available: https://www.langchain.com/

[10] Government of India, Digital Personal Data Protection Act, 2023. New Delhi, India: Government of India, 2023.

[11] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M.

Funtowicz, and J. Davison, "Transformers: State-of-the-Art Natural Language Processing," in Proc. 2020 Conf. on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP-Demos), 2020, pp. 38–45.

[12] "Justice delayed is *justice denied. With JURI AI, justice is just a conversation away."* (Tagline/Slogan — not a bibliographic reference, so it should not be included in the reference list.)