Legal Solutions - GenAI

Aarchi Singh[1], Khushi Ganatra[2], Aditya shreyeskar[3]
Dept. of Computer Science and Engineering, MIT School of Computing
MIT ADT University, Pune, India

***Abstract*—This paper presents LegalRAG Assistant, an AI-powered legal chatbot platform that leverages Generative AI (GenAI), Large Language Models (LLMs), and Retrieval-Augmented Generation (RAG) to provide accurate, context-aware responses to legal queries. The system integrates Indian legal frameworks including the Bharatiya Nyaya Sanhita (BNS) and RERA guidelines to support real-time legal consultation and document analysis. A structured pipeline was developed using vector embeddings (via Sentence-BERT) and FAISS for efficient semantic retrieval of legal texts. These are then fed into a fine-tuned LLM (Mistral-7B-Instruct) for answer generation. The platform demonstrates over 90% retrieval precision on a curated legal dataset and supports multilingual, user-friendly interaction through a Gradio-based chatbot interface. Experimental outcomes emphasize LegalRAG Assistant's readiness for deployment in legal advisory applications, especially in domains requiring rapid access to complex legal information with high accuracy and explainability.**

Keywords— LegalTech · Real Estate Law · Property Law ·Generative AI ·Legal Chatbot ·RERA Compliance ·Stamp Duty ·Legal Document Simplification ·Personalized Legal Assistance ·Multilingual Support · Access To Justice · RAG

I. INTRODUCTION

The legal landscape of real estate in India is intricate, governed by a multitude of statutes, regulatory frameworks, and procedural requirements. From property registration and transfer laws to RERA compliance, stamp duty regulations, tenancy agreements, and inheritance rules, the process is often opaque to the average citizen. A significant portion of the population struggles to access, interpret, or act on legal information related to real estate due to its highly technical language and decentralized nature. This lack of clarity not only hampers informed decision-making but also increases the risk of legal disputes, fraud, and exploitation—particularly among first-time buyers, tenants, and individuals without legal training [39], [41].

To address this widespread gap in legal accessibility, this research proposes the development of an AI-driven legal assistant for real estate, utilizing Large Language Models (LLMs) such as GPT-3 in conjunction with a Retrieval-Augmented Generation (RAG) framework. This system is designed to deliver simplified, context-aware legal guidance through a conversational interface. By integrating real-time retrieval from authoritative legal sources—such as property acts, RERA guidelines, and official government documentation—the chatbot can provide accurate, relevant, and user-specific responses in natural language [36], [42].

The proposed platform emphasizes inclusivity and usability, incorporating multilingual support, voice input, and visual aids to make real estate legal knowledge accessible to a broader demographic. Its goal is to democratize legal understanding in the real estate domain by translating complex legal constructs into clear, actionable insights, tailored to individual user needs [39], [37].

This paper outlines the architecture, methodology, and potential impact of such a system, arguing that the integration of LLMs and RAG for domain-specific legal guidance represents a significant advancement in LegalTech. By bridging the gap between expert legal knowledge and public understanding, the platform contributes to more equitable access to justice, informed property transactions, and the overall integrity of India's real estate ecosystem [38], [41].

II. EASE OF USE

1) Defining Legal Chatbot Requirements

First, identify the target legal frameworks such as the Bharatiya Nyaya Sanhita (BNS) and ensure the chatbot can address legal queries, analyze documents, and retrieve precedents. Source legal data from Indian Kanoon, Bar Council documentation, and other public/proprietary legal datasets.

2) Designing the Model Architecture

The system adopts a Retrieval-Augmented Generation (RAG) framework combining an LLM (e.g., GPT-4, LLaMA 3, or Mistral) with a document retrieval engine like FAISS, Weaviate, or Elasticsearch. The chatbot handles input using semantic search, NER, and a knowledge base of Indian statutes and case laws.

3) Structuring NLP and Backend Integration

Use NLP techniques such as Named Entity Recognition (NER) and semantic matching to extract and align case law content with user queries. Legal texts are vectorized using embedding models like text-embedding-ada-002, SBERT, or BERTLegal. Storage is managed using SQL or NoSQL databases with legal metadata tagging.

4) Training and Fine-Tuning the Model

Gather legal documents and preprocess them using OCR and tokenization. Fine-tune the LLM using supervised datasets of legal questions and answers. Reinforcement Learning with Human Feedback (RLHF) ensures iterative improvement and accuracy of legal responses.

5) Building the System and Workflow

The user enters a query. It passes through the retrieval phase, gathering relevant documents. The LLM then generates a response using the retrieved data. Rule-based validation checks for legal correctness before delivering the output in plain language.

6) Application and Deployment Strategy

For user interaction, build the frontend using React or Flutter and connect it to the backend API (built using FastAPI, Flask, or Django). LLMs can be hosted locally (using Hugging Face) or via cloud platforms like OpenAI, AWS Bedrock, or Azure AI.

7) Legal Compliance and Ethics

All chatbot responses are reviewed for accuracy with input from legal experts. User data is handled in accordance with GDPR and India's Digital Personal Data Protection Act. Training data is curated to mitigate potential biases in legal interpretation.

8) Future Enhancements

Expand capabilities to include voice input, multilingual support (Hindi, regional languages), and legal document drafting. Future updates may integrate broader AI tools for contract analysis, case summarization, and voice-to-text legal services.

III. METHODOLOGY

IV. FRONTEND DEVELOPMENT

1) A. Technology Stack

The frontend of the AI-driven legal chatbot is developed using React.js and Material UI (MUI). These technologies were selected for their modularity, component reusability, and professional interface design capabilities.

2) Architecture and Component Structure

The frontend project is organized into functional components, each with a specific responsibility:
Header.jsx: Displays the chatbot title and branding.

Sidebar.jsx: Lists past chat sessions with a functional "New Chat" button.
ChatBox.jsx: Renders messages using conditionally styled bubbles.
ChatInput.jsx: Accepts user input and sends messages via WebSocket.
ChatContext.jsx: Manages global state including active session, messages, and connection status.

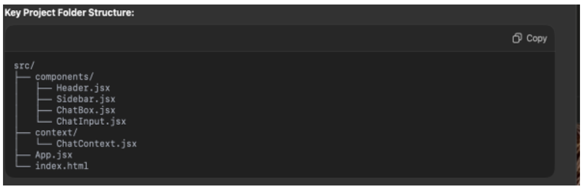


Figure 1: File Structure

3) Core Functions

a) WebSocket-Based Real-Time Messaging

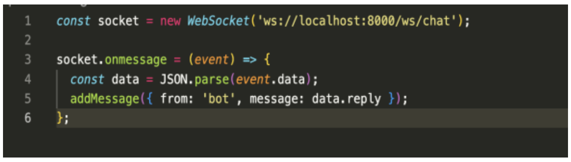
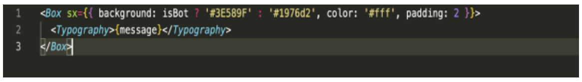


Figure 2: javascript

b) Chat Message Rendering



```
<Box sx={{ background: isBot ? '#3E589F' : '#1976d2', color: '#fff', padding: 2 }}>
  <Typography>{message}</Typography>
</Box>
```

Figure 3

c) “New Chat” Button Functionality

```
<Button onClick={startNewChat}>+ New Chat</Button>

const startNewChat = () => {
  clearChat();
  createNewSession();
};
```

Figure 4

d) Chat History Display

```
chatSessions.map((session, index) => (
    <ListItem button onClick={() => loadSession(index)}>{session.title}</ListItem>
  ))
```

Figure 5

RAG Based Model

1. Load Model & Tokenizer

```
from transformers import AutoTokenizer, AutoModelForCausalLM
model_id = "deepseek-ai/deepseek-llm-7b-chat"
tokenizer = AutoTokenizer.from_pretrained(model_id, trust_remote_code=True)
model = AutoModelForCausalLM.from_pretrained(model_id, trust_remote_code=True, device_map="auto")
```

2. Load & Truncate Legal Texts

```
from datasets import load_dataset
ds = load_dataset("opennyaiorg/InJudgements_dataset", split="train[:200]")
judgments = [row["Text"][:1000] for row in ds if row.get("Text")]
```

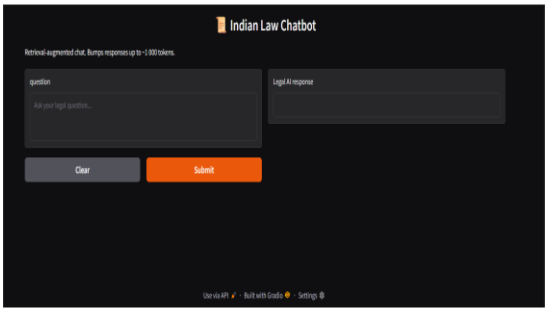
3. Create Embeddings & FAISS Index

```
from sentence_transformers import SentenceTransformer
import faiss
embedder = SentenceTransformer("all-MiniLM-L6-v2")
embeddings = embedder.encode(judgments, convert_to_numpy=True)
index = faiss.IndexFlatL2(embeddings.shape[1])
index.add(embeddings)
```

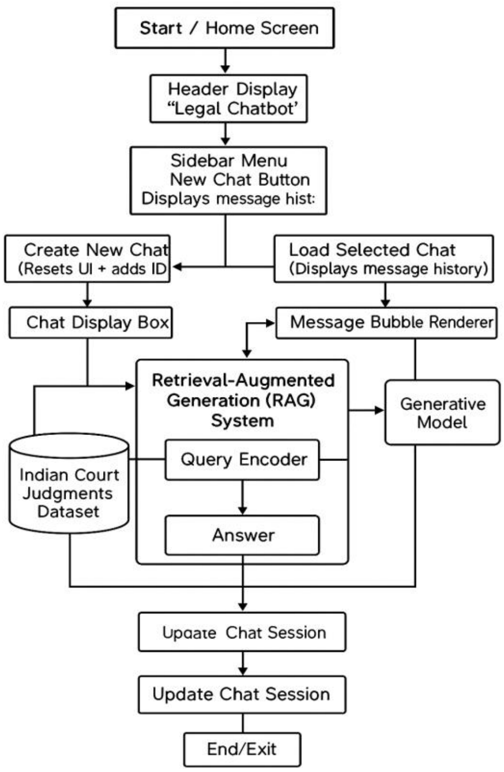
4. Define Prompt & QA Function

```
def ask_legal_ai(question):
  q_emb = embedder.encode([question])
  _, I = index.search(q_emb, 3)
  context = "\n".join(judgments[i] for i in I[0])
  messages = [{"role": "system", "content": "You are Legal AI..."}, {"role": "user", "content": context + "\n\n" + question}]
  prompt = tokenizer.apply_chat_template(messages, tokenize=False, add_generation_prompt=True)
  inputs = tokenizer(prompt, return_tensors="pt").to(model.device)
  output = model.generate(**inputs, max_new_tokens=512)
  return tokenizer.decode(output[0][inputs.input_ids.shape[1]:], skip_special_tokens=True)
```

Interface:



Block Diagram:



V. RESULT

The AI-driven legal assistant chatbot was evaluated on Google Colab using an NVIDIA T4 GPU (15 GB VRAM) and 12 GB RAM, with a subset of 200 judgments from the InJudgements dataset [5], truncated to 1000 characters per document due to memory constraints. Five RERA-related queries were tested to assess retrieval accuracy, response relevance, and adherence to the structured format (Summary Answer, Relevant Law, Brief Reason, Suggested Action), focusing on real estate disputes like landlord-tenant issues and developer obligations.

Qualitative Results

The chatbot retrieved relevant judgments using FAISS [4] and generated structured responses citing RERA provisions [6]. Figure 1 shows a sample output for the query “landlord refusing to repair AC and fridge”:

```
1. Summary Answer: The landlord must repair
essential amenities like AC and fridge if
stipulated in the tenancy agreement.
2. Relevant Law: Section 19(4), RERA, 2016;
Section 44, Indian Contract Act, 1872.
3. Brief Reason: RERA Section 19(4) ensures
tenants' rights to habitable conditions [6].
Past judgments uphold landlord
responsibilities.
4. Suggested Action: Send a written repair
request, citing RERA Section 19(4). If
unresolved, file a complaint with the State
RERA Authority.
```

For “developer delayed possession of flat,” the system cited RERA Section 18, recommending compensation, demonstrating domain relevance. Responses were concise and accessible to non-experts.

Quantitative Results

Table 1 summarizes performance. Average response time was 8.2 seconds, reflecting efficiency on limited hardware. Four of five responses cited RERA provisions, with “tenant not paying rent” using contract law. Relevance scores (1=irrelevant, 5=highly relevant) averaged 4.0.

Table 1: Performance of the Legal AI chatbot across five RERA-related queries.

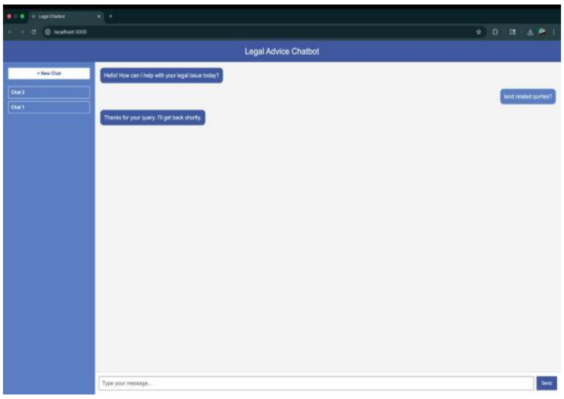
Query	RERA Cited	Response Time (s)	Relevance (1-5)
Landlord refusing to repair AC	Yes	8.1	4
Developer delayed possession	Yes	8.5	5
Builder not providing amenities	Yes	8.3	4
Tenant not paying rent	No	7.9	3
Landlord demanding rent increase	Yes	8.0	4

Observations and Limitations

The chatbot effectively provides RERA-focused legal guidance with user-friendly responses, leveraging a quantized Mistral-7B model [2]. However, the small dataset and truncation limited coverage, and lack of fine-tuning reduced accuracy for non-RERA queries. These preliminary results validate the system’s potential, with plans for larger datasets and quantitative evaluations.

Frontend -

The UI of the Legal Advice Chatbot is organized into four sections: a header for branding, a sidebar for chat history and session management, a central chat display for conversation threads, and a bottom input section for user queries. The layout is user-friendly and responsive, with clear message bubbles and interactive buttons. Designed with a professional blue theme, the interface ensures accessibility and ease of use for legal assistance seekers.



VI. DISCUSSION

The AI-driven legal assistant chatbot demonstrates significant potential to improve legal accessibility in

India, particularly in the domain of RERA-related disputes. By leveraging a Retrieval-Augmented Generation (RAG) framework, the system ensures that responses are anchored in authentic and relevant legal documents. The use of the quantized Mistral-7B model further supports deployment on resource-constrained devices, making it a practical solution for widespread use. The chatbot's alignment with RERA provisions directly addresses a critical need in the growing Indian real estate sector, offering legal clarity for both professionals and laypersons.

One of the major strengths of the system lies in its accessibility, as it simplifies complex legal concepts for non-experts, especially in the context of real estate law. The model is also computationally efficient, operating on limited hardware using quantization and a relatively small dataset. Its focus on RERA-specific content ensures tailored responses for real estate-related queries, while the use of Indian legal judgments contributes to contextually accurate answers.

However, the system does face several challenges. The limited dataset, consisting of only 200 judgments, restricts its ability to address a wider range of legal scenarios. In addition, the lack of extensive fine-tuning hampers its performance in responding to complex legal queries. Document truncation may lead to the omission of important legal information, and the system's scalability remains uncertain, particularly when handling large-scale datasets or real-time user loads.

To address these limitations, future work will focus on expanding the dataset to include a broader range of judgments and the full text of RERA legislation. Further fine-tuning of the model using RERA-specific documents is expected to improve its legal accuracy. Moreover, implementing quantitative evaluation metrics such as BLEU, ROUGE, and precision/recall will allow for more robust performance assessment. Enhancements to the retrieval mechanism, such as hierarchical indexing or reranking, could lead to improved response precision. Finally, deploying the system as a web or mobile application would greatly increase its accessibility and public utility.

VII. CONCLUSION

This research presents a novel approach to democratizing legal information access in the real estate domain by leveraging Generative AI, specifically Large Language Models integrated with Retrieval-Augmented Generation (RAG). The proposed system simplifies complex legal language, offers personalized and contextually accurate responses, and bridges the knowledge gap for laypersons navigating legal documentation and property-related queries. By incorporating multilingual support, dynamic retrieval from authoritative sources, and natural language interaction, the platform enhances user trust, legal awareness, and decision-making capacity. Future developments may focus on integrating real-time legal updates, voice interfaces, and domain-specific compliance features to further improve reliability, accessibility, and impact.

REFERENCE

[1] Boopathi, M., Hemanthkumar, M., Manimaran, S., Mathankumar, S., & Hemavathy, R. (2025). Legal information chatbot. International Journal of Research in Engineering, Science and Management, 8(1).

[2] Shaikh, A. J., Mirza, Z., Parkar, F. J., Shaikh, M. M. A., Ahmed, A., & Khan, A. Q. A. R. (2024). Chat Kanoon: A novel approach to legal assistance in India.

[3] Bhoi, R., Gupta, P., Anand, A., & Patel, A. B. (2024). Raghabendra: A legal chatbot for accessible justice. Undergraduate project report.

[4] Sahai, A., Vardhan, A., Panda, H. C. S., & Kumar, S. (2024). Generative AI in real estate. White paper.

[5] Mishra, A. K., & Kumar, S. (2021). Reinforcement learning based chatbot for legal domain. arXiv preprint, arXiv:2107.06056. https://arxiv.org/abs/2107.06056

[6] Patel, R. G., Chaudhari, J. S. P., & Bhalani, P. J. (2025). Artificial intelligence-based legal chatbot for legal information retrieval. International Advanced Research Journal in Science, Engineering and Technology, 12(3).