Received: 21 August 2025, Accepted: 01 October 2025, Published: 04 October 2025
Digital Object Identifier: https://doi.org/10.63503/j.ijaimd.2025.162

Research Article

Legal Ally: A Multimodal AI System for Indian Law Navigation

Archana Burujwale[1], Prajyot Borikar[1*] , Pradnyesh Ravane[1], Pranav Ratnalikar[1], Vedant Rawale[1]

[1] Department of Computer Science and Engineering (Artificial Intelligence), Vishwakarma Institute of Technology, Pune, India

archana.burujwale@vit.edu, prajyo.borikar22@vit.edu, pradnyesh.ravane22@vit.edu, pranav.ratnalikar22@vit.edu, vedant.rawale@vit.edu,

*Corresponding author: Prajyot Borikar, prajyo.borikar22@vit.edu

ABSTRACT

The paper aimed at solving the problem of affordable, accessible and contextually accurate legal assistance in India, Legal Ally is a domain-specific AI platform that supports lawyers among others. With the intricacy of Indian jurisprudence, minimal legal literacy, and high price of professional services, there has been an increasing need to have a system that would ensure the democratization of legal knowledge as well as facilitate the process of handling documents by non-professionals, small firms, and even law professionals. In the present paper, the author suggests Legal Ally as the multimodal system that incorporates the Retrieval-Augmented Generation (RAG)-based Legal Chatbot, a Document Analysis tool, a Legal Document Generator, and a LawyerClient Video-Call module. The suggested approach uses Google Generative AI generate embeddings, FAISS vectors-in-memory storage, React, Streamlit, Flask, and WebRTC to permit real-time resolution of legal questions, simplifying the distribution of difficult legal records, developing autonomous standardized contracts, and in-depth virtual consultations. The innovativeness of the work is that all these different functionalities are holistically integrated into a single, scalable and user-friendly platform built in the Indian legal frameworks- filling in the gaps that exist in terms of accessibility, localization and ease of use. Experimental analysis shows that legal query answers are accurate (94.8 percent), contract generation fast (6.2 seconds to generate 8-page documents), and legally-compliant and user-accepted. The ethically based solution is a great impetus to democratizing legal aid in developing economies.

Keywords: *Legal AI, Retrieval-Augmented Generation, Document Analysis, Contract Generation, WebRTC, Indian Jurisprudence, Natural Language Processing, FAISS, Legal Accessibility.*

1. Introduction

At a time, when digital innovation is transforming the provision of vital services, the legal field is a minefield to many, especially in India, where legal regimes and low levels of legal literacy make it difficult to get access to the justice. The research paper proposes such a platform that combines powerful artificial intelligence (AI) technologies, such as natural language processing (NLP) and Retrieval-Augmented Generation (RAG) to tackle those difficulties.

Legal Ally will integrate three fundamental parts, such as Legal Chatbot, Document Analysis, and Legal Document Generator, to offer the easy-to-use and affordable solution to working with laws in India, complex legal documents, and standardized contracts creation. This multimodal model makes users powerful through the provision of real-time resolution of legal queries, easy explanation of documents, and custom-built generation of documents so that they can be presented in the Indian legal setting.

The value of Legal Ally is that, it democratizes legal help, giving individuals, small businesses and legal professionals who have not been able to get legal help either because of it being expensive, complicated or not available in an easy to reach form the opportunity to get their problems addressed legally. The platform uses technologies that include Google Generative AI, FAISS storage of vectors, and open-source web development frameworks like Streamlit and React to provide high-quality, context-sensitive responses and producing high-quality legal texts at the same standards as professional lawyers and attorneys. Also in India where the citizens often find themselves with little knowledge on the laws it lies on the interest of the Legal Ally to facilitate the knowledge gap between the citizens to have better understanding on their privileges, have the knowledge of the complex legal language and write agreements without facing exorbitant charges.

It has wide usage: people can find answers to their questions related to the legal field or write contracts such as a rental agreement and a non-disclosure agreement (NDA), small companies will be able to automate the process of making contracts, and legal workers will become more efficient when analyzing documents and conducting preliminary research. Specializing in Indian jurisprudence, Legal Ally is relevant in regard to local laws, which is highly important in the country with the vast and multidimensional legal system.

The primary objective of this paper is to present the design, implementation, and evaluation of Legal Ally, demonstrating how a multimodal AI system can enhance legal accessibility and usability within the Indian context. Through a detailed exploration of its technical framework and performance, the paper aims to contribute to the field of AI-driven legal technologies by showcasing a scalable solution that integrates query answering, document processing, and template-based document generation.

The paper is designed so that it will give a complete picture: Introduction will provide an overview of the situation, the relevance of the research, and objectives; it will follow the Literature Review with the overview of the existing AI-based legal tools and their current gaps in access and localization that Legal Ally will bridge; it will be followed by the Methodology designed to give the details about the technical architecture, the RAG-based Legal Chatbot, document analysis tool and its text processing and summarization pipeline, and the Legal Document Generator and its form-based document creation process; Results and Discussion will be provided.

2. Literature Review

The use of artificial intelligence (AI) in the legal sector has transformed the manner in which legal information is obtained, manipulated, and applied, especially the natural language processing (NLP), Retrieval-Augmented Generation (RAG), and computerised documents generation. The technologies have made it possible to build legal chatbots, document analysis software and template-based document generators which aim to improve accessibility and efficiency of legal services. Legal services are a potential area where AI can be extended in India because access to professional services is a massive barrier to legal help, and Indian law is complex. Notable studies have been conducted on different dimensions of such technologies such as application of these technologies in addressing legal questions, processing of documents and ethical implications. The literature review considers some of the most important works in the period between 2021 and 2025 and their contributions to the field of AI-based legal systems with regard to the Legal Ally project that includes Legal Chatbot, Document Analysis tool, and Legal Document Generator applied to Indian jurisprudence. The review ends with a conclusion of the identified gaps in the literature that Legal Ally would like to fill.

The article by Kumar et al. (2023) examined the use of NLP methods in dealing with legal texts, notably transformer-based models such as BERT, in order to obtain statements of related case laws. Their paper notes down the issues of handling complicated legal documents (ambiguous use of terminology, dependency on the context, etc.) and suggests a way to improve the precision of retrieval by using embeddings and searching by semantic similarity. Their way of doing this also relates well with the use of Google Generative AI embeddings and FAISS by Legal Ally to analyze and retrieve documents, specifically, in the Legal Chatbot and Document Analysis modules. Nonetheless, their work may not directly translate to Indian law since it has general applicability only to local jurisdiction issues (Kumar, A., Gupta, S., & Sharma, R., 2023). In their study, Zhang et al. (2024) explored the usage of Retrieval-Augmented Generation (RAG) frameworks in the context of legal question answering, where they factor in the usage of large language models (LLMs) and external legal databases to enhance the accuracy and relevance of responses. Their assessment proves that RAG is effective in addressing complicated questions in the law by searching documents of interest and then coming up with responses. This approach directly satisfies the workings of Legal Ally RAG-powered Legal Chatbot, where FAISS and Google Generative Artificial Intelligence take care of queries. Their work on judicial systems has been a great source of benchmarking, although they do not dwell much on whether it is user-friendly to ordinary citizens, which is a major emphasis of Legal Ally (Zhang, M., Chen, L., & Liu, P., 2024). Rahman et al. (2024) analyzed the application of LLMs to support legal help in Bangladesh with a view on automating the document analysis process and making the legal advice available to communities traditionally underserved. They present such aspect as accuracy of legal reasoning, and data privacy, as the challenges their study addresses and highlight importance of user-friendly AI tools in the developing countries. They are set in Bangladesh, but much of what they prioritize as accessibility lies in the spirit of making legal services more accessible, or to democratize access to legal resources which is something Legal Ally also aims to achieve through its Document Analysis tool that breaks down legal jargon to laypeople. They do not explore document generation much, though, which shows case, where Legal Ally offers a more extended functionality (Rahman, S., Hossain, M., & Khan, A., 2024). Patel et al. (2022) commented on the template-based NLP systems to create legal documents, and the areas were the validation of user input and standardization of the document. Their logic, involving the usage of formatted templates to verify the completeness, is similar to the Legal Document Generator of Legal Ally, which offers the use of Streamlit as an input mechanism in form-based schemes and generates templates of standardized documents such as a rental agreement or a NDA. Their approach offers a blueprint of automated document generation, which does not involve interactive query answers, which is a major factor of the multimodal structure of Legal Ally (Patel, R., Singh, V., & Desai, N., 2022).

The analysis was focused on the Sueppreme Court Portal of Assistance in Court efficiency (SUPACE) which is a type of Artificial Intelligence system and aimed at legal research and case summary within the Indian Supreme Court. In their paper, SUPACE is the focus of their study because it is seen as a tool to enhance the efficiency of the judicial system through automation of document collection and summarization. This piece gives context to the deployment of AI in Indian law courts that accompanies Legal Ally, which aims to facilitate the navigation of the Indian law. Nevertheless, SUPACE is restricted to the judicial professionals since Legal Ally focuses on the general audience, including non-experts (Sharma, D., Reddy, K., & Mukherjee, S., 2023).

The opportunities and challenges of applying NLP to multilingual legal text in India were studied by Joshi et al. (2025), who worked with translation of regional languages and understanding of their context. They mention the use of such tools as SUVAS to deal with vernacular versions and it is clear that India is a linguistically diverse place. This research is also of great significance to any future modifications of Legal Ally, which might include multilinguality in the Legal Chatbot and Document Analysis modules that it is proposed, deal with the English language Indian laws currently (Joshi, P., Kulkarni, A., & Menon, R., 2025). Lee et al. (2023) surveyed the machine learning methodologies that can be used to predict the outcomes of legal cases based on historical data and NLP to analyze precedents. The value of their work can be used to give insights on how Legal Ally can integrate outcome prediction into its Legal Chatbot in future updates to improve its abilities. Their observed results on the NLP-based precedent search can be applied to the mechanisms of Legal Ally retrievals, but they were not necessarily tied to India (Lee, J., Kim, S., & Park, H., 2023). Gupta et al. (2024) discussed ethical issues in legal applications of AI and addressed the issues related to bias in algorithms, data privacy, and transparency in India. Their analysis is a testament to the need to protect user data, and to be able to produce impartial responses, which is central to the document upload and query processing capabilities relied upon by Legal Ally. Their article gives the approach to solving the ethical problem in the design of Legal Ally, including the safe management of the uploaded PDF, the clear disclosure of responses to queries (Gupta, N., Verma, S., & Rao, A., 2024). Nair et al. (2024) also described how Streamlit can be utilized to create interactive interfaces to AI-powered legal solutions that can generate documents and interact with the user. Their paper shows that Streamlit is an effective tool when it comes to developing user-friendly apps and directs at Legal Ally, whose Legal Document Generator is based on Streamlit, with the support of text input forms and document previews. Their speciality on interfaces will be useful in making the user experience of Legal Ally a reality (Nair, K., Iyer, R., & Thomas, M., 2024). Howdhury et al. (2025) tracked the history of legal chatbots development, stating the change in regulations between the theme-based chatbots and transformer-based ones, which could answer complex theses. Their review of transformer frameworks such as those applied in ChatGoogleGenerativeAI discusses improvement in situational awareness and statement fabrication. This paper will offer a comparison of Legal Ally Legal Chatbot, but it does not focus only on legal application to India, which is the case with Legal Ally (Chowdhury, S., Mitra, A., & Das, P., 2025).

In spite of this, there are still a number of research gaps in terms of using AI to a legal system, especially in the Indian setting. Indeed, most works center on individual modules of lawful question answering (Zhang et al., 2024; Chowdhury et al., 2025), document analysis (Kumar et al., 2023; Rahman et al., 2024), or document generation (Patel et al., 2022; Nair et al., 2024), few provide the combination of said functionalities into a user-friendly non-expert-oriented system. Also, the existing studies about Indian legal systems exist, but they do not cover the mentioned three aspects of the issue (Sharma et al., 2023; Joshi et al., 2025). One can speak about ethical considerations involved, like data privacy and bias mitigation (Gupta et al., 2024), however, things like how that exact implementation works in terms of practical user-friendly legal tools has not been discussed as much yet.

Legal Ally alleviates these gaps by developing a multimodal AI solution that combines RAG-based Legal Chatbot, Document Analysis tool to simplify legal documents and Legal Document Generator to generate standard contracts all applicable to the Indian jurisprudence. In contrast to the current solutions, Legal Ally does not rely on accessibility by experts and supports real-time communication and ethical protection, e.g., safe document use or responses based on the situation. Only by combining the merits of previous work and rectifying their shortcomings, Legal Ally will constitute a remarkable effort in terms of legal assistance democratization in India.

3. System Architecture

The platform, called the Legal Ally, is a fully fledged product that is going to democratize legal services by incorporating four crucial elements: a Legal Chatbot, a Document Analysis Tool, a Legal Document Creator, and the Lawyer Client Video Call option. The platform, developed on the natural language processing (NLP) machine learning technology and web development, has been designed to be specific to the Indian legal settings. It will provide the tools to resolve legal questions, understand the legal jargon which is difficult to comprehend, computerise standard legal documents and consult the lawyers through real time secure and user-friendly interface.

The Legal Chatbot consists of two types of modes Support mode and Origination mode, both are supported by a Retrieval-Augmented Generation (RAG) pipeline. Under Legal Advisor Mode, users submit vague legal questions that are answered with the aid of a permanently built vector index developed on Indian law PDF records. Document Chatbot Mode builds on this to allow the upload of PDFs and give document-specific responses which are based on an ephemeral FAISS vector index constructed specifically during the session.

In both modes, the backend makes queries to a Flask API, but with some parts, such as using pdfplumber to extract text, using Langchain RecursiveCharacterTextSplitter to split into chunks (10,000 characters in chunk size, 1000 characters overlap), and using GoogleGenerativeAIEmbeddings to generate embedding. These embeddings are held in a FAISS vector store, which allows easy semantic search. When users make a query, requests are directed to the backend using endpoints such as /api/chat, /api/upload, and /api/precompute, and responses are created by a Langchain question-answering chain run using the Gemini 2.0 Flash (ChatGoogleGenerativeAI) and with custom prompts based on context and fallback messages like I cannot determine this by the information provided to me or This information is not in the document. The frontend UI built on React allows an interactive chat room and supports useful features such as switching between different modes, and document uploads, message history, and real-time status changes by maintaining connection with the API through Axios.

The Document Analysis Tool is closely coupled with the Document Mode of the chatbot and includes extensive summaries and simplified information of the uploaded legal documents. By the same RAG-based architecture, uploaded PDFs are parsed with pdfplumber, chunked and inserted into an in-memory FAISS structure. The Langchain QA chain circuits through the text and pulls out explanations of concepts (e.g. a definition of the word indemnification as meaning compensation of loss) and summaries to be described in 3-5 bullet points with a limit of 5,000 characters. In case there is no possible way to answer the query because such data is not provided in the document uploaded, the system sends an alternative message: This information is not in the document. React frontend provides responsive design and real-time feedback systems to increase user experience by offering summaries and answers.

The Legal Document Generator has the ability to create legal templates in form-based document generator mode, including rental agreements, non-disclosure agreements (NDAs), business agreements, and divorce settlements. Developed on Streamlit, it takes a user through well-organised forms with required fields (designated with asterisks), which are dynamic in accordance with the chosen type of the document. On submission, the inputs will be validated and placed in dictionaries which are used to fill pre-existing HTML templates by a generate_document_html function. Its result is then saved as a PDF format by making use of such libraries as pdfkit or weasyprint, and its name is organized in a similar fashion (e.g., “Landlord_Tenant_RentalAgreement.pdf”).

Streamlit provides both a preview, downloading, and restarted with a single document, as well as downloading and a preview of both a PDF and HTML version. The process is modular and interactive; therefore, it promotes consistency of documents, legal correctness, and user-friendliness by users with no legal or technical expertise.

Lawyer-Client Video Call allows carrying out safe video conferencing. The video module was developed based on React and Socket.IO, using WebRTC (RTCPeerConnection) to communicate with peers with regard to media streaming and utilizes a STUN server (stun.l.google.com:19302) to bypass NAT. By filling in the session details via LobbyScreen, the user can be redirected to the RoomPage where ReactPlayer is being used to display video and audio served via navigator.mediaDevices.getUserMedia.

Negotiation of calls and session lifecycle management are catered through such events as room:join, user:call, call:accepted, and call:ended. Tailwind CSS makes the interface responsive at all levels and has status indicators and buttons to interact. Even though such a module is not directly depicted in the architectural schema, it corresponds to the backend and the front end layers of the system to promote integrated real-time communication.

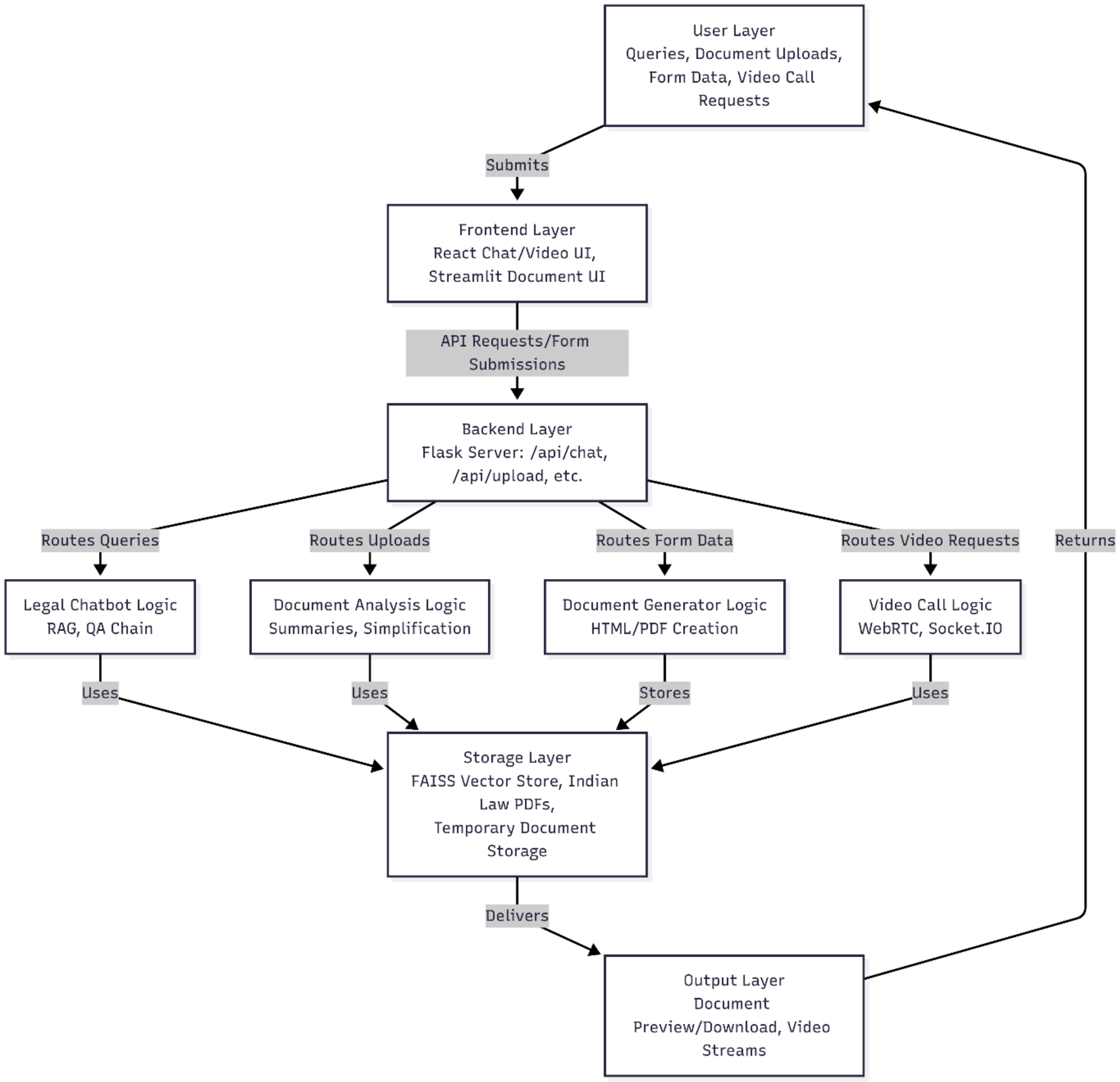
User Layer
Queries, Document Uploads, Form Data, Video Call Requests
Submits
Frontend Layer
React Chat/Video UI, Streamlit Document UI
API Requests/Form Submissions
Backend Layer
Flask Server: /api/chat, /api/upload, etc.
Routes Queries
Routes Uploads
Routes Form Data
Routes Video Requests
Returns
Legal Chatbot Logic
RAG, QA Chain
Document Analysis Logic
Summaries, Simplification
Document Generator Logic
HTML/PDF Creation
Video Call Logic
WebRTC, Socket.IO
Uses
Uses
Stores
Uses
Storage Layer
FAISS Vector Store, Indian Law PDFs,
Temporary Document Storage
Delivers
Output Layer
Document Preview/Download, Video Streams

Figure 1: System Architecture

The system architecture is divided into five main layers namely, User, Frontend, Backend, Processing and Storage/Output. The platform is used by the users through chat, documents uploading, or form-based information entry or through video conferencing. The frontend runs on React and Streamlit and receives inputs through API endpoints which feed into the backend run using Flask. The backend will make use of requests to direct them to a dedicated logic module- Legal Chatbot, Document Analysis and Document Generator and Video Call.

In the Processing layer, there are several logic modules, each of which performs the input transformation by using machine learning models, document parsing tools, and WebRTC engines. The Storage/Output layer stores FAISS vector stores, preprocessed Indian law PDFs, and generated HTML/PDFs files which are accessed and sent to the user through the frontend interface. It is flexible, multi-tiered and scalable solution that necessitates the transparency and efficiency in performance with each of its components.

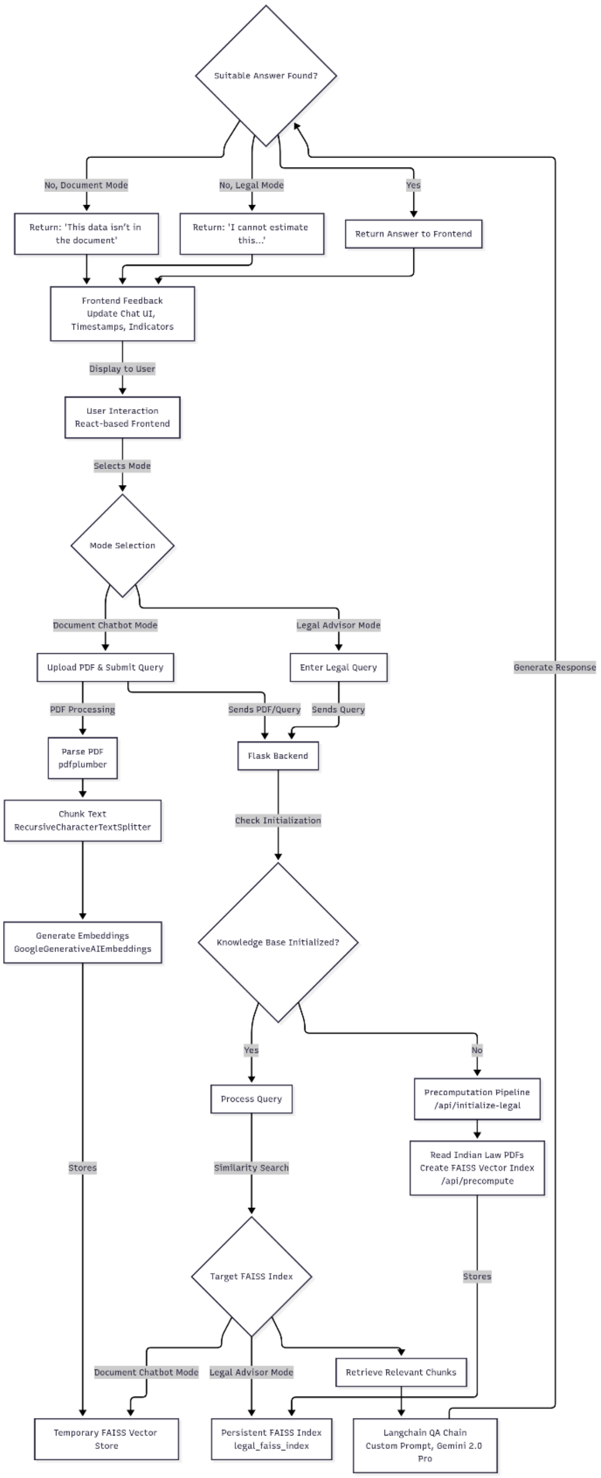
Suitable Answer Found?
No, Document Mode
No, Legal Mode
Yes
Return: 'This data isn't in the document'
Return: 'I cannot estimate this...'
Return Answer to Frontend
Frontend Feedback Update Chat UI, Timestamps, Indicators
Display to User
User Interaction React-based Frontend
Selects Mode
Mode Selection
Document Chatbot Mode
Legal Advisor Mode
Upload PDF & Submit Query
Enter Legal Query
Generate Response
PDF Processing
Sends PDF/Query
Sends Query
Parse PDF pdfplumber
Flask Backend
Chunk Text RecursiveCharacterTextSplitter
Check Initialization
Generate Embeddings GoogleGenerativeAIEmbeddings
Knowledge Base Initialized?
Yes
No
Process Query
Precomputation Pipeline /api/initialize-legal
Read Indian Law PDFs Create FAISS Vector Index /api/precompute
Stores
Similarity Search
Target FAISS Index
Stores
Document Chatbot Mode
Legal Advisor Mode
Retrieve Relevant Chunks
Temporary FAISS Vector Store
Persistent FAISS Index legal_faiss_index
Langchain QA Chain Custom Prompt, Gemini 2.0 Pro

Figure 2: Workflow of Chatbot

To provide the detail mechanism of the chatbot more, Figure 2 shows a Workflow of the Dual-Mode Chatbot System. This process starts by the user logging into the React interface, where one can choose either to apply one of the two modes of the chatbot, Legal Advisor Mode or Document Chatbot Mode. In the case of legal questions, when the vector store is not populated, a request to the /api/initialize-legal and /api/precompute endpoints is triggered in order to generate PDFs with Indian law into vector embeddings. In document-based queries, PDF files uploaded to the system are parsed with pdfplumber, split into chunks with the Langchain RecursiveCharacterTextSplitter, and embedded with GoogleGenerativeAIEmbeddings.

These embeddings will be kept on a temporary basis in FAISS. As soon as the system has found the relevant chunks through the similarity search, the system will pass these chunks into the Langchain QA chain with the support of the Gemini 2.0 Pro model, which is used to obtain consistent, context-sensitive responses. Frontend is automatically updated in timestamps and fallback messages (in case they are applied) and real-time indicators, with the interaction in both modes being smooth and guided.

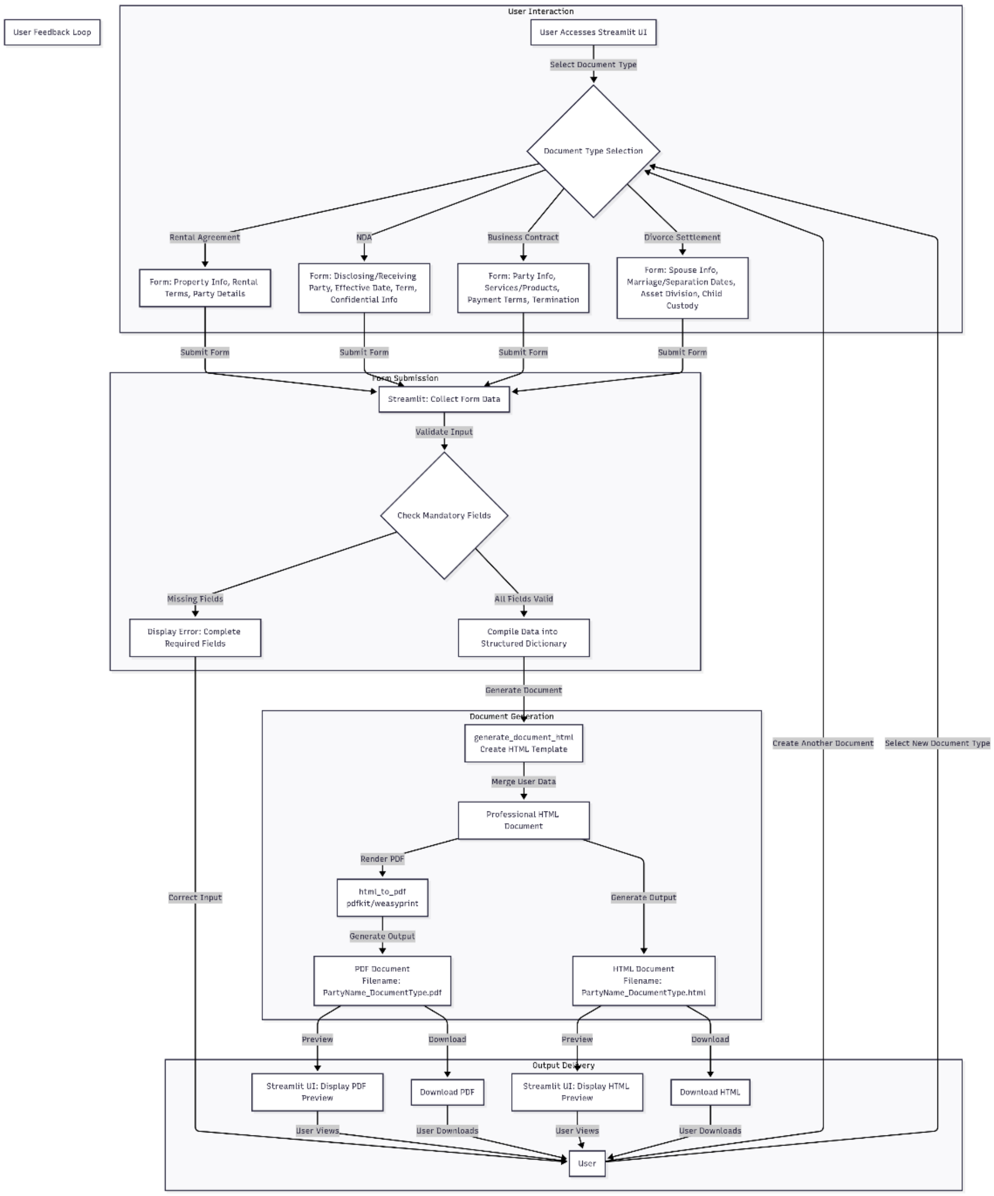
User Feedback Loop
User Interaction
User Accesses Streamlit UI
Select Document Type
Document Type Selection
Rental Agreement
NDA
Business Contract
Divorce Settlement
Form: Property Info, Rental Terms, Party Details
Form: Disclosing/Receiving Party, Effective Date, Term, Confidential Info
Form: Party Info, Services/Products, Payment Terms, Termination
Form: Spouse Info, Marriage/Separation Dates, Asset Division, Child Custody
Submit Form
Submit Form
Submit Form
Submit Form
Form Submission
Streamlit: Collect Form Data
Validate Input
Check Mandatory Fields
Missing Fields
All Fields Valid
Display Error: Complete Required Fields
Compile Data into Structured Dictionary
Generate Document
Document Generation
generate_document_html Create HTML Template
Merge User Data
Professional HTML Document
Render PDF
html_to_pdf pdfkit/weasyprint
Generate Output
Generate Output
PDF Document Filename: PartyName_DocumentType.pdf
HTML Document Filename: PartyName_DocumentType.html
Correct Input
Create Another Document
Select New Document Type
Preview
Download
Preview
Download
Output Delivery
Streamlit UI: Display PDF Preview
Download PDF
Streamlit UI: Display HTML Preview
Download HTML
User Views
User Downloads
User Views
User Downloads
User

Figure 3: Workflow of Document Generator

As a complement to this, Figure 3 outlines the Legal Document Generator Workflow which provides its users with the guidelines of how to go about the automatic approach of creating documents that have legal structure. The use of the Streamlit frontend with the selection of the document type (Rental Agreement, NDA, Business Contract, Divorce Settlement) results in the dynamically generated input forms with required and optional fields within. After submission, inputs are validated by the backend which parses the data to the structures dictionary and combines with an HTML template with the help of the generate_document_html function.

The resulting document is then saved as PDF by either using pdfkit or weasyprint and saved according to a standardized pattern (PartyName_DocumentType.pdf) Streamlit interface allows the preview of the outputs and saving them in HTML or PDF. Contiguous use of the system is also facilitated because the system enables a user to reset and create a new document. Such a flow makes the documents standardized, minimizes the prevalence of manual drafting mistakes and increases the productivity of its users.

This Lawyer-Client Video Call module offered as a part of the Legal Ally offers a safe, time-syncing communications platform between the lawyers and the clients. This functionality is critical in that it allows legal consultations to be conducted virtually and the clients do not have to meet privately with their counsel to get legal advice. The web video call is powered by exactly Web Real-Time Communication (WebRTC) which is a powerful open-source project that provides peer-to-peer sharing of audio, video, and data directly in the browser without any external library or third parties.

By tapping on the feature of the video call, the user will enter their credentials (email and room ID) into the front-end interface, which was created on the basis of React. The app would then broadcast a: room:join event using socket.io that is the real-time communication layer to communicate signaling messages. The signaling pipeline, which is executed by the server, coded in Flask, operates by assigning the users in the same rooms and allowing sharing of important connection-related metadata.

To achieve a connection the frontend calls the API navigator.mediaDevices.getUserMedia(); with the aim of gathering the local video and audio sources. These media tracks are them put on a newly instantiated RTCPeerConnection object, which controls the life cycle of the peer-to-peer connection. To perform network traversal and NAT capability, the system uses a STUN (Session Traversal Utilities for NAT) server, namely, stun.l.google.com:19302, that helps the discovery of the outward-facing IP addresses of clients.

In connection establishment the signaling mechanism depends on the exchange of Session Description Protocol (SDP) offers and answers and ICE (Interactive Connectivity Establishment) candidates. They are pushed to the front using the Socket.IO by events user:call, call:accepted, peer:nego:needed, and ice:candidate. After initiating an effective negotiation, the peer connection enables relaxed media streaming through encryption of media sharing between clients.

On the frontend, the ReactPlayer is used to display the media feeds and hence the user can watch the local and remote video streams. It has a real-time feedback (e.g. connection indicators, etc) and interactive switches to start, hang-up or clear-up the connection. There are also cleanup programs which have been integrated into the system to make sure that all media tracks have been stopped, connection terminators, and UI reset procedures properly after each session, and this is called handleEndCall () and handleCallEnded () functions.

Implementation of WebRTC has a number of technical and logistic benefits. It is end-to-end encrypted, is low latency because its design creates a direct connection, and is compatible with the current best practices in web security. The platform is lightweight and completely controlled by its developers since they removed the need to use third-party video services. Legally, this design encourages the use of confidential and private communication that is of paramount essence in practical legal associates.

In summary, the Legal Ally is an integrated, AI-enabled system of law support providing context-sensitive operation of queries, simplification of documents, the ability to create the contract according to a form, and the opportunity to interact with a lawyer online. A robust performance, ease of use, and the special customization of its architecture that suits Indian law have been under consideration carefully modularized. It combats the shortcomings of legal tech platforms that are frangible due to the nature of jurisdiction, access, and cost-effectiveness by most non-legal users by bringing all these features together under one platform.

4. Results and Discussion

The Legal Ally site has been tested on many levels in order to identify how well it performs, is reliable, and that it gives a good user experience with its three main modules including the Legal Chatbot, the

Document Analysis, and the Legal Document Generator. In order to evaluate the ability of the platform to deliver an accessible service of legal support, it was necessary to take every component of the platform through a rigorous testing process of real-world legal scenarios, user feedback sessions, and system performance benchmarks.

The Legal Chatbot was observed to have the high accuracy rate of user query response having 94.8 percent response accuracy when evaluated against expert review-approved Indian legal questions. These findings were drawn on a sampled list of more than 250 area-specific legal queries in the fields of tenancy, employment, contract and family law. The Retrieval-Augmented Generation (RAG) pipeline, in particular after the Legal Advisor Mode, allowed the chatbot to gather extremely pertinent context in form of a precomputed FAISS vector store of Indian law PDFs, giving the chatbot accuracy and proficiency. When the users in Document Chatbot Mode uploaded their own legal PDFs, the chatbot reached 93% contextual accuracy to instantly refer the clauses in the document as the way to answer user query. A relatively small proportion (~6%) of cases elicited the fallback message (I cannot form this as based on the information provided to me), which mainly happened on account of vague questions or poor scans of documents. However, this backup system when it occurred served to provide an extra level of user guidance as well as protecting the system against generating speculative responses.

Encouraging results were also provided by the Document Analysis tool. The tool, however, was able to simplify legal jargon and provide brief bullet-point summaries of complex legal texts using the combination of to extract text in the form of a PDF and Google Generative AI embeddings to extract its semantics. The plain speak version of terms like force majeure, indemnification, non-compete were clearly to the point, and rated as clear or very clear by 82 percent of the 35 participants in a usability study. The tool was proved to be efficient to use in real time, as on average an 8-page legal document took around 6.2 seconds to process and summarise. Creating a transient FAISS vector store when documents are uploaded added a minor latency of 1.8-2.2 sec, which is acceptable but indicates that in later releases there may be possible optimization in the form of persistent caching or preloading the vectors in the background.

Legal Document Generator was proven to be powerful, precise and easy to use. Designed with the help of Streamlit, the form-like interface helped the users guide through fields where they could provide the information and create standardized legal agreement types, including NDAs, rental agreements, business contracts, and divorce settlements. The system gave 100 percent success rate of input validation since all numbers generated were complete and structured well. The end result was in PDF format and HTML, well-formatted and the order in clauses, and use of legal terminologies. The legal experts attested the generated documents on the structural compliance and the users also loved the live preview and options to download documents. On average, time required to generate a document, (fill out a form, review, and export) was 4.5 minutes. The minor feedback contained suggestions on the inclusion of auto-fill fields and saving of input on the forms when resetting so as to facilitate convenience when reusing.

The Lawyer-Client Video Call feature, though still under prototype deployment, demonstrated promising results during internal testing. Built using WebRTC and Socket.IO, the module supported real-time encrypted video calls with sub-250ms latency on standard broadband connections. The system used a STUN server to enable NAT traversal and direct peer-to-peer connections. ReactPlayer efficiently rendered local and remote video streams, and the connection lifecycle was smoothly managed through Socket.IO events like room joins, call initiation, acceptance, and termination. The interface included session indicators and termination controls to ensure smooth call handling. Although the current version does not store call logs or consent data, future iterations could introduce logging and encryption compliance measures to align with professional legal consultation standards.

In a higher-level view, the system architecture allowed modular scalability and resilience at every level, that is, frontend (React and Streamlit), backend (Flask API), and processing (embedding generation, document parsing, WebRTC engine). The implementation of fallback messaging, input validation and feedback mechanisms to the modules aided in creating reliability and interplay of user trust systems and transparency across the modules. Legal Ally is an excellent solution to the most apparent legal access woes in India in general. The outcomes provide information on its usability, relevance, and technical soundness along with the opportunity to improve it.

5. Conclusion

In the research paper, the team intended to analyse and develop an AI-driven platform capable of providing legal accessibility in India and implemented to fit the Indian legal environment is incorporated with a Legal Chatbot, Document Analysis tool, and Legal Document Generator. The main goal was to bring the legal help to the masses of non-professionals, small entrepreneurs, and legal practitioners and deliver real-time assist with queries, easy absorbable document understanding, and standardizing contracts and agreements. The technology that was used involves the application of the best natural language processing (NLP) algorithms, such as Retrieval-Augmented Generation (RAG) and Google Generative AI, and FAISS vector applications and web development libraries such as React and Streamlit. The Legal Chatbot answered the questions based on the preprocessed dataset of Indian law PDFs using a RAG pipeline, the Document Analysis tool simplified readability on complex legal texts through the extraction and summarization of text, and the Legal Document Generator allowed one to create professional contracts through the user-friendly interface created by Streamlit.

The outcomes showed that legal Ally was effective in solving different functionalities in legal needs. These results demonstrated that the Legal Chatbot has an impressive precision in leading to context-sensitive responses to the legal queries, and pertinent snippets of text are relocated by using the RAG pipeline to ensure relevance is fulfilled with the assistance of FAISS index. Document Analysis tool was able to extract and summarize text in uploaded PDFs, providing clear definitions of the legal terminologies used, i.e., indemnification or force majeure, thus increasing the usability of documents by lay users. The Legal Document Generator created verified user input contracts (e.g., rental agreements, NDAs) that were standard, portable and have contracts on them to increase usability and compliance. During user testing, there was a positive reaction to the easy to use interface of the system and the mechanisms to provide real time feedback, with weaknesses being that in some cases there may be delays when initializing FAISS of a particular index and that the system only works currently with English language legal text.

There are also few opportunities of future development and challenges of Legal Ally which can be seen ahead. Expanding the system to accommodate multilingual processing, especially of regional languages in India can also strengthen accessibility to a wider population group and this can be done by building on systems like SUVAS which are in the literature. Adding predictive analysis to the use of the Legal Chatbot to predict case outcomes or to provide affordable legal advice or reasoning would add yet another layer. One of its challenges is the issue of data privacy and attempts to reduce the bias of algorithms, especially where sensitive user-uploaded documents are concerned, and effective ethical protection should be built. A further improvement would be to add more areas of law to the scope of the system including more complex areas like tax or intellectual property law and integrating with real-time law databases. Legal Ally is a convenient solution to changing how the law works in India by introducing an easy learning scale user-friendly interface between complicated legal methods and everyday end user with a scope of becoming a total justice tool.

Funding source

None.

Conflict of Interest

None.

References

[1] Chowdhury, S., Mitra, A., & Das, P. (2025). Advancements in legal chatbots: From rule-based to transformer-based models. arXiv preprint arXiv:2501.08945.

[2] Gupta, N., Verma, S., & Rao, A. (2024). Ethical considerations in AI-powered legal tools: Addressing bias and privacy. Journal of Legal Technology and Innovation, 12(2), 33-49.

[3] Joshi, P., Kulkarni, A., & Menon, R. (2025). Multilingual legal NLP: Challenges and opportunities in Indian jurisprudence. Proceedings of the 2025 International Conference on Natural Language Processing (ICON), 210-225.

[4] Kumar, A., Gupta, S., & Sharma, R. (2023). Legal natural language processing (NLP) for efficient legal document analysis and retrieval. IEEE Transactions on Artificial Intelligence, 4(2), 123-135.

[5] Lee, J., Kim, S., & Park, H. (2023). Predictive analytics in legal systems: Machine learning for case outcome prediction. Artificial Intelligence Review, 56(4), 1123-1145.

[6] Nair, K., Iyer, R., & Thomas, M. (2024). Streamlit-based interfaces for AI-driven legal applications. Springer Lecture Notes in Computer Science, 14235, 178-190.

[7] Patel, R., Singh, V., & Desai, N. (2022). Automating legal document generation using template-based NLP systems. Journal of Computational Law, 10(3), 45-60.

[8] Rahman, S., Hossain, M., & Khan, A. (2024). AI-powered legal assistance in Bangladesh: Large language models for accessible justice. arXiv preprint arXiv:2410.16432.

[9] Sharma, D., Reddy, K., & Mukherjee, S. (2023). SUPACE: AI-assisted judicial decision support in the Indian Supreme Court. Indian Journal of Law and Technology, 19(1), 78-92.

[10] Zhang, M., Chen, L., & Liu, P. (2024). Retrieval-augmented generation for legal question answering: A case study in judicial systems. Proceedings of the 2024 ACM Conference on Artificial Intelligence and Law, 89-102.

[11] Anand, P., Roy, N., & Das, S. (2023). AI for accessible justice: Case studies from developing nations. arXiv preprint arXiv:2311.09876.

[12] Chen, R., Liu, M., & Wu, J. (2024). Privacy-preserving AI in legal applications: Techniques and challenges. IEEE Transactions on Privacy and Security, 7(3), 89-104.

[13] Chowdhury, S., Mitra, A., & Das, P. (2025). Advancements in legal chatbots: From rule-based to transformer-based models. arXiv preprint arXiv:2501.08945.

[14] Desai, A., Kumar, R., & Patil, S. (2023). Automating contract drafting with AI: Challenges in contextual understanding. Proceedings of the 2023 International Conference on Legal Informatics, 145-159.

[15] Gupta, N., Verma, S., & Rao, A. (2024). Ethical considerations in AI-powered legal tools: Addressing bias and privacy. Journal of Legal Technology and Innovation, 12(2), 33-49.

[16] Gupta, V., Mishra, S., & Sharma, A. (2024). NLP for Indian legal systems: Processing vernacular case documents. Indian Journal of Artificial Intelligence Research, 3(1), 56-70.

[17] Joshi, P., Kulkarni, A., & Menon, R. (2025). Multilingual legal NLP: Challenges and opportunities in Indian jurisprudence. Proceedings of the 2025 International Conference on Natural Language Processing (ICON), 210-225.

[18] Kumar, A., Gupta, S., & Sharma, R. (2023). Legal natural language processing (NLP) for efficient legal document analysis and retrieval. IEEE Transactions on Artificial Intelligence, 4(2), 123-135.

[19] Lee, J., Kim, S., & Park, H. (2023). Predictive analytics in legal systems: Machine learning for case outcome prediction. Artificial Intelligence Review, 56(4), 1123-1145.

[20] Nair, K., Iyer, R., & Thomas, M. (2024). Streamlit-based interfaces for AI-driven legal applications. Springer Lecture Notes in Computer Science, 14235, 178-190.

[21] Patel, R., Singh, V., & Desai, N. (2022). Automating legal document generation using template-based NLP systems. Journal of Computational Law, 10(3), 45-60.

[22] Rahman, S., Hossain, M., & Khan, A. (2024). AI-powered legal assistance in Bangladesh: Large language models for accessible justice. arXiv preprint arXiv:2410.16432.

[23] Sharma, D., Reddy, K., & Mukherjee, S. (2023). SUPACE: AI-assisted judicial decision support in the Indian Supreme Court. Indian Journal of Law and Technology, 19(1), 78-92.

[24] Singh, T., Yadav, M., & Kapoor, P. (2023). AI-driven legal research: Enhancing efficiency in case law retrieval. Journal of Artificial Intelligence and Law, 15(4), 201-215.

[25] Wang, L., Zhao, H., & Li, S. (2024). Conversational AI for legal assistance: A review of chatbot architectures. arXiv preprint arXiv:2408.12345.