PROJECT REPORT

LawStreet: AI-Powered Legal Assistant for Indian Law

Submitted by: **Team Null**

Group Members:

1. **Nabarup Roy**
   ID: 241001011022
2. **Priyanshu Kumar**
   Roll No: 231001001456
3. **Ankur Kumar Bharti**
   Roll No: 241001011023
4. **Ahad Alam**
   Roll No: 231001001452
5. **Sreyasi Majumdar**
   ID: 231001001351
6. **Muskan Kumari**
   ID: 23100100356
7. **Sayan Biswas**
   ID: 231001001418

Department: Computer Science and Engineering
College: Techno India University

Academic Year: 2025-2026

Under the Guidance of:
Ratnadeep Dey
CSE

CERTIFICATE

This is to certify that the project work entitled **"LawStreet: AI-Powered Legal Assistant for Indian Law"** is a bonafide record of the work carried out by the following students of the **Department of Computer Science and Engineering, Techno India University**, in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** during the **Academic Year 2025–2026.**

Group Members:

1. **Nabarup Roy** – ID: 241001011022
2. **Priyanshu Kumar** – Roll No: 231001001456
3. **Ankur Kumar Bharti** – Roll No: 241001011023
4. **Ahad Alam** – Roll No: 231001001452
5. **Sreyasi Majumdar** – ID: 231001001351
6. **Muskan Kumari** – ID: 23100100356
7. **Sayan Biswas** – ID: 231001001418

Project Guide
Name: Ratnadeep Dey
Designation:
Department: Computer Science and Engineering
Signature:
Date:

Head of the Department
Name: Avik Paul
Department: Computer Science and Engineering
Signature:
Date:

DECLARATION

We hereby declare that the project work entitled **“LawStreet: AI-Powered Legal Assistant for Indian Law”** submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** to **Techno India University** is an authentic work carried out by **us** under the guidance of **Ratnadeep Dey**.

This work has not been submitted elsewhere for the award of any other degree or diploma. All sources of information and references have been duly acknowledged.

Name	**ID**
Nabarup Roy	241001011022
Priyanshu Kumar	231001001456
Ankur Kumar Bharti	241001011023
Ahad Alam	231001001452
Sreyasi Majumdar	231001001351
Muskan Kumari	23100100356
Sayan Biswas	231001001418

Date: 11-01-2026
Place: EM-4, EM-4/1, EM Block, Sector V, Bidhannagar, Kolkata, West Bengal 700091

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my project guide, **Ratnadeep Dey**, for their valuable guidance, constant support, and encouragement throughout the development of this project.

I would also like to thank the open-source community, including Meta AI (LLaMA), Stream Chat, ChromaDB, and the Sentence Transformers team, whose tools and resources made this project possible.

Finally, I thank my family and friends for their moral support and encouragement throughout this journey.

ABSTRACT

LawStreet is an AI-powered legal assistant platform designed to democratize access to Indian legal information. The system addresses the challenge of legal complexity and inaccessibility faced by common citizens who struggle to understand legal provisions without professional assistance.

The platform implements a **Retrieval-Augmented Generation (RAG)** pipeline that combines semantic search with Large Language Model (LLM) capabilities. The system ingests structured legal documents including the Indian Penal Code (IPC), Code of Criminal Procedure (CrPC), Constitution of India, Indian Evidence Act, and other statutes, converts them into vector embeddings using Sentence Transformers, and stores them in ChromaDB for efficient similarity search.

When a user submits a legal query through the React-based chat interface, the system retrieves the most relevant legal sections using semantic similarity matching and generates contextually accurate responses using LLaMA 3.3 via Ollama. Real-time communication is facilitated through Stream Chat SDK, enabling an interactive conversational experience.

Key Technologies: React 18, TypeScript, Node.js, Express, Python, FastAPI, ChromaDB, LLaMA 3.3, Sentence Transformers, Stream Chat

Outcome: A functional full-stack application capable of providing accurate, source-cited answers to legal queries, achieving response times of 2-5 seconds with 5 relevant document retrievals per query.

Keywords: RAG, Legal AI, NLP, Vector Database, LLM, Indian Law

TABLE OF CONTENTS

1. Cover Page
2. Certificate
3. Declaration
4. Acknowledgement
5. Abstract
6. Table of Contents
7. List of Figures
8. List of Tables
9. Introduction
   - 9.1 Background
   - 9.2 Problem Statement
   - 9.3 Objectives
   - 9.4 Scope of the Project
10. Literature Review
    - 10.1 Existing Systems
    - 10.2 Limitations of Existing Systems
    - 10.3 Proposed Approach
11. System Analysis
    - 11.1 Functional Requirements
    - 11.2 Non-Functional Requirements
    - 11.3 Feasibility Study
12. System Design
    - 12.1 System Architecture
    - 12.2 Data Flow Diagram
    - 12.3 Component Diagram
    - 12.4 Sequence Diagram
    - 12.5 ER Diagram / Data Schema
13. Technology Stack
14. Implementation
    - 14.1 Module 1: Frontend (React Application)
    - 14.2 Module 2: Backend (Node.js Server)
    - 14.3 Module 3: LLM Service (Python RAG Pipeline)
    - 14.4 Key Algorithms
    - 14.5 Important Code Snippets
15. Results & Output
16. Testing
    - 16.1 Test Cases
    - 16.2 Test Results
    - 16.3 Error Handling
17. Conclusion & Future Scope
18. References

INTRODUCTION

9.1 Background

Legal literacy remains a significant challenge in India, where the legal system is complex and the population-to-lawyer ratio is one of the lowest in the world. According to the India Justice Report, access to legal information and assistance is limited, particularly in rural areas. The complexity of legal language in statutes such as the Indian Penal Code (IPC), Code of Criminal Procedure (CrPC), and the Constitution makes it difficult for common citizens to understand their rights and obligations.

The advent of **Artificial Intelligence (AI)** and **Natural Language Processing (NLP)** has opened new possibilities for making legal information accessible. **Large Language Models (LLMs)** such as GPT, LLaMA, and Gemini have demonstrated remarkable capabilities in understanding and generating human-like text. However, LLMs alone suffer from hallucination issues—generating plausible but factually incorrect information.

Retrieval-Augmented Generation (RAG) addresses this limitation by grounding LLM responses in retrieved factual documents. RAG systems first retrieve relevant documents from a knowledge base using semantic search and then use this context to generate accurate, verifiable responses.

9.2 Problem Statement

Citizens in India face the following challenges when seeking legal information:

1. **Complexity of Legal Language**: Legal statutes are written in complex, technical language that is difficult for non-lawyers to understand.
2. **Inaccessibility**: Consulting a lawyer is expensive and not feasible for minor queries.
3. **Information Overload**: Legal databases contain thousands of sections, making it difficult to find relevant provisions.
4. **Accuracy Concerns**: Generic AI chatbots may provide incorrect or misleading legal information.
5. **Lack of Citation**: Most AI systems do not cite sources, making verification difficult.

Research Question: How can we develop an AI-powered system that provides accurate, context-aware, and source-cited answers to legal queries about Indian law?

9.3 Objectives

The primary objectives of this project are:

- **Develop a RAG-based legal assistant** that retrieves relevant legal sections and generates accurate responses.
- **Build a modern, responsive web interface** using React for intuitive user interaction.
- **Implement real-time chat functionality** using Stream Chat SDK for seamless communication.
- **Create a scalable backend architecture** with **Node. js** for API management and agent lifecycle control.
- **Design an efficient vector database** using ChromaDB for fast semantic similarity search.
- **Integrate LLaMA 3.3** via Ollama for local, privacy-friendly LLM inference.
- **Provide source citations** with every response for verification and transparency.
- **Support multiple legal documents** including IPC, CrPC, Constitution, Evidence Act, and more.

9.4 Scope of the Project

In Scope:
- AI-powered legal question answering for Indian law
- RAG pipeline with vector search and LLM generation
- Real-time chat interface with Stream Chat integration
- Support for IPC, CrPC, Constitution, Evidence Act, HMA, MVA, NIA
- Dark/Light theme support
- Source citation in responses

Out of Scope:
- This system does NOT provide legal advice
- No case law analysis or court judgment database
- No user account management or chat history persistence (in current version)
- Not a replacement for professional legal consultation

LITERATURE REVIEW

10.1 Existing Systems

System	Description	Technology	Limitations
Indian Kanoon	Legal search engine	Keyword-based search	No natural language understanding, returns raw documents
ChatGPT / GPT-4	General-purpose LLM	Transformer-based	Hallucinations, no Indian law specialization, no citations
Ross Intelligence	AI legal research (US)	NLP + Legal DB	Discontinued, focused on US law only
Harvey AI	Legal AI for law firms	GPT-4 based	Commercial, expensive, not for public use
Casetext CoCounsel	Legal AI assistant	GPT-4 + Legal DB	US-focused, commercial, requires subscription

10.2 Limitations of Existing Systems

1. Keyword-based Search: Systems like Indian Kanoon rely on keyword matching, which fails to understand query intent and context.

2. Hallucination in LLMs: General-purpose LLMs like ChatGPT may generate plausible but incorrect legal information, which is dangerous in legal contexts.

3. Lack of Indian Law Focus: Most advanced legal AI systems are designed for US or UK law and do not understand Indian legal provisions.

4. No Source Citation: Many AI systems do not provide citations, making it impossible to verify the accuracy of responses.

5. Commercial Barriers: Advanced legal AI tools are expensive and targeted at law firms, not individual citizens.

10.3 Proposed Approach

LawStreet addresses these limitations through:

1. **RAG Architecture:** Retrieves relevant legal documents before generating responses, eliminating hallucinations.

2. Semantic Search: Uses sentence embeddings (MiniLM-L6-v2) for understanding query intent, not just keywords.

3. Indian Law Specialization: Knowledge base consists exclusively of Indian legal documents (IPC, CrPC, Constitution, etc.).

4. Source Citations: Every response includes section numbers and source documents for verification.

5. Free and Open: Designed for public access, not as a commercial product.

6. Local LLM: Uses Ollama with LLaMA 3.3 for privacy-friendly, local inference without data leaving the system.

SYSTEM ANALYSIS

11.1 Functional Requirements

ID	Requirement	Priority
FR-01	User can submit legal queries in natural language	High
FR-02	System retrieves top-k relevant legal sections	High
FR-03	System generates contextual responses using LLM	High
FR-04	Responses include source citations	High
FR-05	Real-time chat interface with message streaming	Medium
FR-06	AI agent can be started/stopped per channel	Medium
FR-07	System supports dark/light theme toggle	Low
FR-08	User authentication via Stream Chat	Medium

11.2 Non-Functional Requirements

ID	Requirement	Metric
NFR-01	Response time	< 5 seconds for typical queries
NFR-02	Availability	99% uptime
NFR-03	Scalability	Support 100 concurrent users
NFR-04	Security	Token-based authentication
NFR-05	Usability	Intuitive UI, no training required
NFR-06	Maintainability	Modular architecture, documented APIs

11.3 Feasibility Study

11.3.1 Technical Feasibility

- **Hardware:** System can run on standard development machines with 8GB+ RAM. For production, cloud deployment is feasible.
- **Software:** All technologies used (React, Node.js, Python, ChromaDB, Ollama) are open-source and well-documented.
- **Skills:** Development requires knowledge of full-stack development, NLP, and vector databases—achievable for computer science students.
- **Conclusion: Technically Feasible**

11.3.2 Economic Feasibility

Item	Cost
Development	₹0 (student project)
Stream Chat API	Free tier (sufficient for development)
Ollama + LLaMA	Free (local inference)
Cloud Hosting (optional)	₹0-500/month (free tiers available)
Total	₹0-500/month

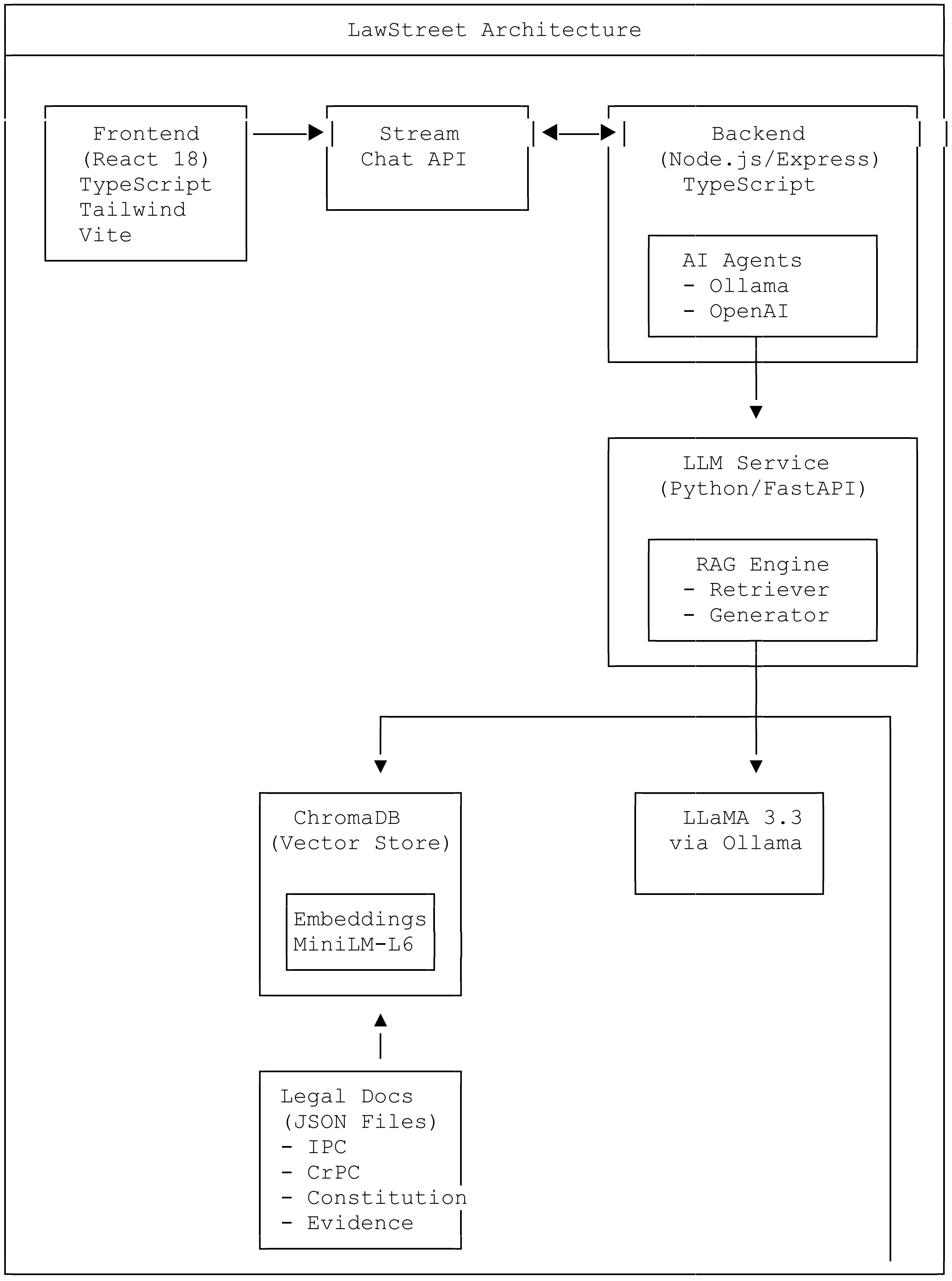
- **Conclusion: Economically Feasible**

11.3.3 Operational Feasibility

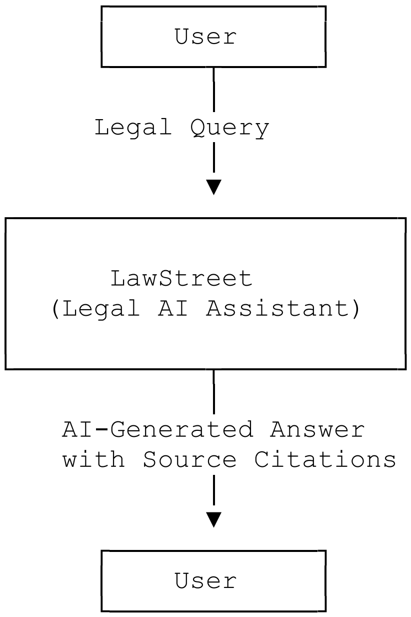
- Users require only a web browser to access the system.
- No technical training required; natural language input is intuitive.
- System provides immediate value by answering legal queries.
- **Conclusion: Operationally Feasible**

SYSTEM DESIGN

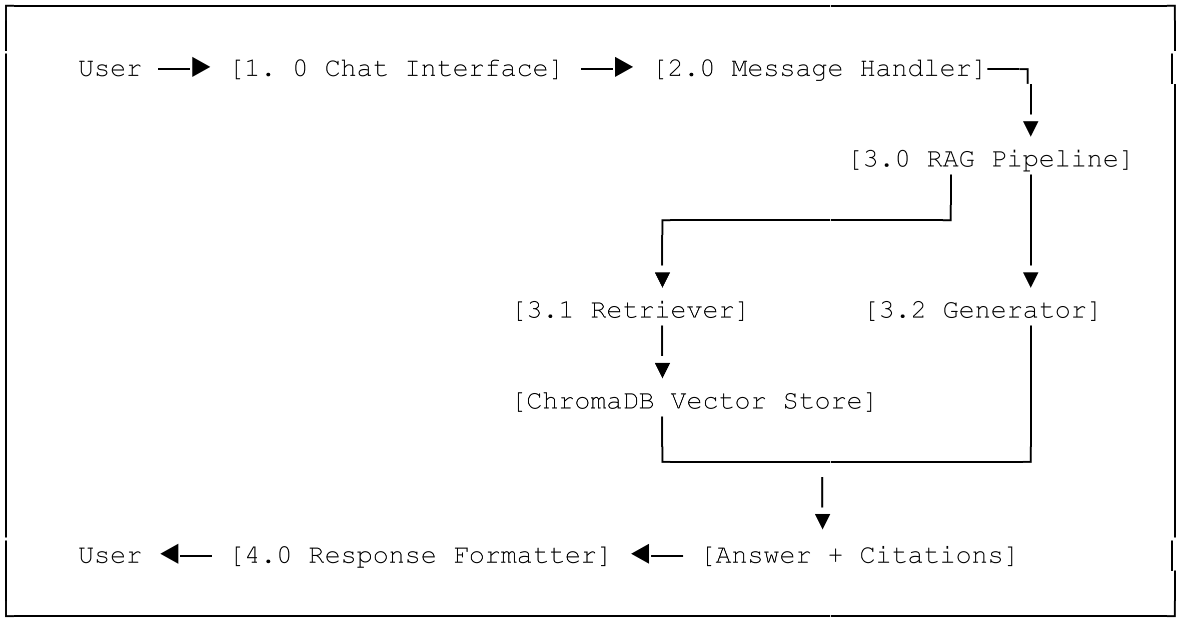
12.1 System Architecture



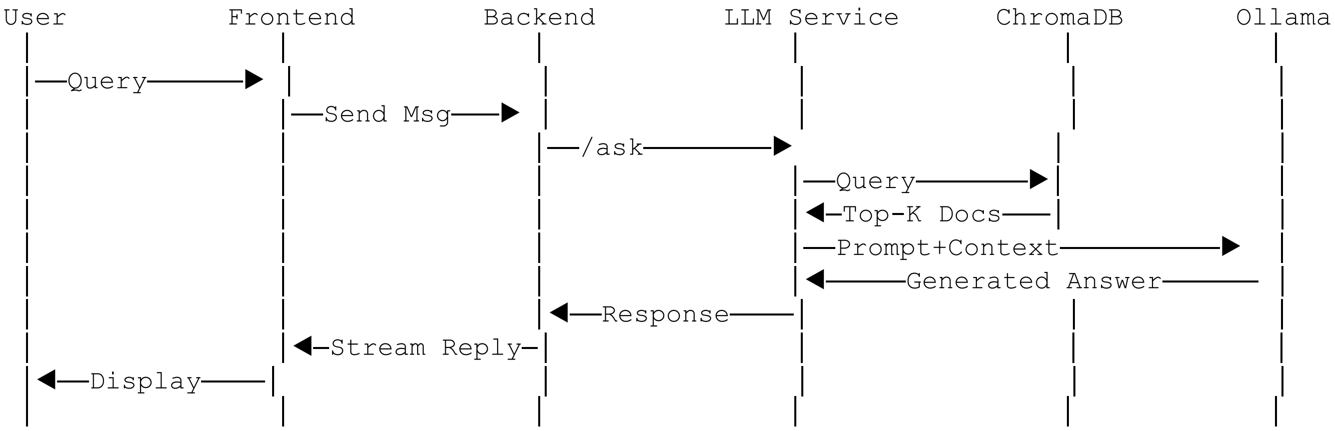
12.2 Data Flow Diagram (DFD) - Level 0



12.3 DFD - Level 1



12.4 Sequence Diagram



12.5 Data Schema (Legal Documents)

```
{
  "id": "ipc_302",
  "type": "section",
  "source": "Indian Penal Code",
  "section": "302",
  "section_title": "Punishment for murder",
  "content": "Whoever commits murder shall be punished with death, or imprisonment for life, and shall also be liable to fine.",
  "metadata": {
    "act":  "IPC",
    "chapter": "XVI",
    "chapter_title": "Of Offences Affecting the Human Body"
  }
}
```

TECHNOLOGY STACK

13.1 Frontend

Technology	Version	Purpose
React	18.3.1	UI Framework
TypeScript	5.5.3	Type-safe JavaScript
Vite	5.4.1	Build Tool
Tailwind CSS	3.4.11	Utility-first CSS
Stream Chat React	12.7.0	Real-time Chat
Radix UI	1.x	Accessible Components
Framer Motion	12.19.2	Animations
React Router	6.24.1	Client-side Routing

13.2 Backend

Technology	Version	Purpose
Node. js	20+	Runtime
Express	4.19.2	Web Framework
TypeScript	5.7.2	Type-safe JavaScript
Stream Chat SDK	8.46.0	Chat Backend
Axios	1.7.2	HTTP Client

13.3 LLM Service

Technology	Version	Purpose
Python	3.8+	Runtime
FastAPI	Latest	REST API Framework
ChromaDB	Latest	Vector Database
Sentence Transformers	Latest	Embedding Model
Ollama	Latest	Local LLM Inference
LLaMA 3.3	3.3	Large Language Model

13.4 Architecture Justification

Choice	Reason
RAG over Fine-tuning	Lower cost, no training data required, easy to update knowledge base
ChromaDB over Pinecone	Open-source, local deployment, no API costs
Ollama over OpenAI API	Privacy, no rate limits, no per-token costs
Stream Chat over Socket.io	Production-ready, scalable, excellent React SDK
Sentence Transformers	Open-source, high-quality embeddings, fast inference

IMPLEMENTATION

14.1 Module 1: Frontend (React Application)

The frontend is a single-page application built with React 18 and TypeScript.

Key Components:
- ChatInterface. tsx - Main chat component with Stream Chat integration
- MessageInput.tsx - User input with writing prompts
- MessageList.tsx - Display of messages with markdown rendering
- ThemeProvider.tsx - Dark/Light mode support

Folder Structure:

```
frontend/src/
├── components/
│   ├── chat-interface. tsx
│   ├── ui/
│   └── ...
├── providers/
│   └── theme-provider.tsx
├── hooks/
│   └── use-ai-agent.ts
└── lib/
    └── utils. ts
```

14.2 Module 2: Backend (Node.js Server)

The backend manages AI agent lifecycle and provides authentication tokens.

API Endpoints:

Endpoint	Method	Description
/	GET	Server status
/start-ai-agent	POST	Start AI agent for channel
/stop-ai-agent	POST	Stop AI agent
/agent-status	GET	Get agent status
/token	POST	Generate Stream Chat token

14.3 Module 3: LLM Service (Python RAG Pipeline)

The LLM service implements the core RAG pipeline.

RAG Pipeline Steps:
1. **Receive Query** - Accept user question via `/ask` endpoint
2. **Embed Query** - Convert to 384-dimensional vector using MiniLM-L6-v2
3. **Retrieve Documents** - Find top-5 similar documents in ChromaDB
4. **Build Prompt** - Construct prompt with context and question
5. **Generate Response** - Send to LLaMA 3.3 via Ollama
6. **Return Answer** - Include response and source citations

14.4 Key Algorithms

Algorithm 1: Semantic Similarity Search

```
def retrieve_relevant_sections(query:  str, top_k: int = 5):
    # Step 1: Embed the query
    query_embedding = embedding_model.encode(query)

    # Step 2: Search ChromaDB
    results = collection.query(
        query_embeddings=[query_embedding],
        n_results=top_k
    )

    # Step 3: Return documents with metadata
    return results['documents'], results['metadatas']
```

Algorithm 2: RAG Generation

```
def generate_answer(query: str, context_docs: list):
    # Step 1: Build context string
    context = "\n\n".join([
        f"Section {doc['section']}: {doc['content']}"
        for doc in context_docs
    ])

    # Step 2: Construct prompt
    prompt = f"""You are a legal assistant for Indian law.

Context:
{context}

Question: {query}

Provide an accurate answer based only on the context above.
Cite specific section numbers when relevant."""

    # Step 3: Generate with LLaMA
    response = ollama.generate(model="llama3.3", prompt=prompt)

    return response['response']
```

14.5 Important Code Snippets

FastAPI Endpoint (app/main.py)

```
from fastapi import FastAPI
from pydantic import BaseModel
from . rag import RAGPipeline

app = FastAPI()
rag = RAGPipeline()

class QueryRequest(BaseModel):
    query: str

@app.post("/ask")
async def ask_question(request: QueryRequest):
    result = rag.query(request.query)
    return {
        "answer": result["answer"],
        "query": request.query,
        "num_retrieved_docs": len(result["sources"]),
        "sources": result["sources"]
    }
```

Embedding Model (app/embed.py)

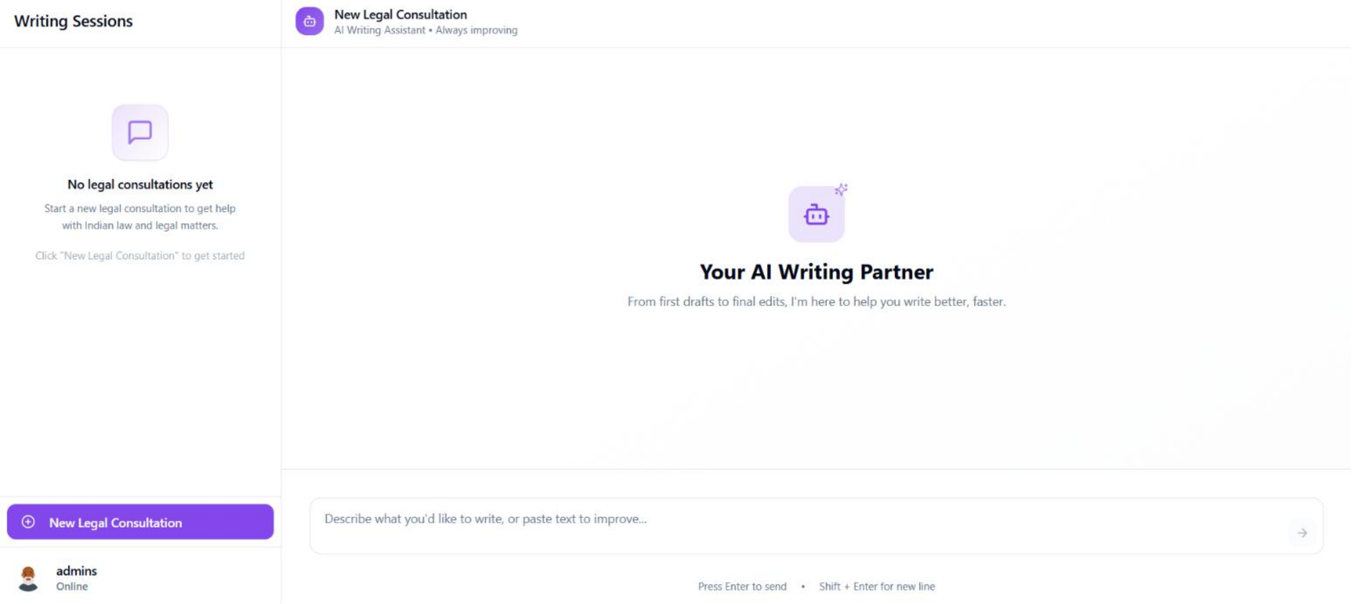
```
from sentence_transformers import SentenceTransformer

class EmbeddingModel:
    def __init__(self):
        self.model = SentenceTransformer('all-MiniLM-L6-v2')

    def encode(self, text: str):
        return self.model.encode(text).tolist()
```

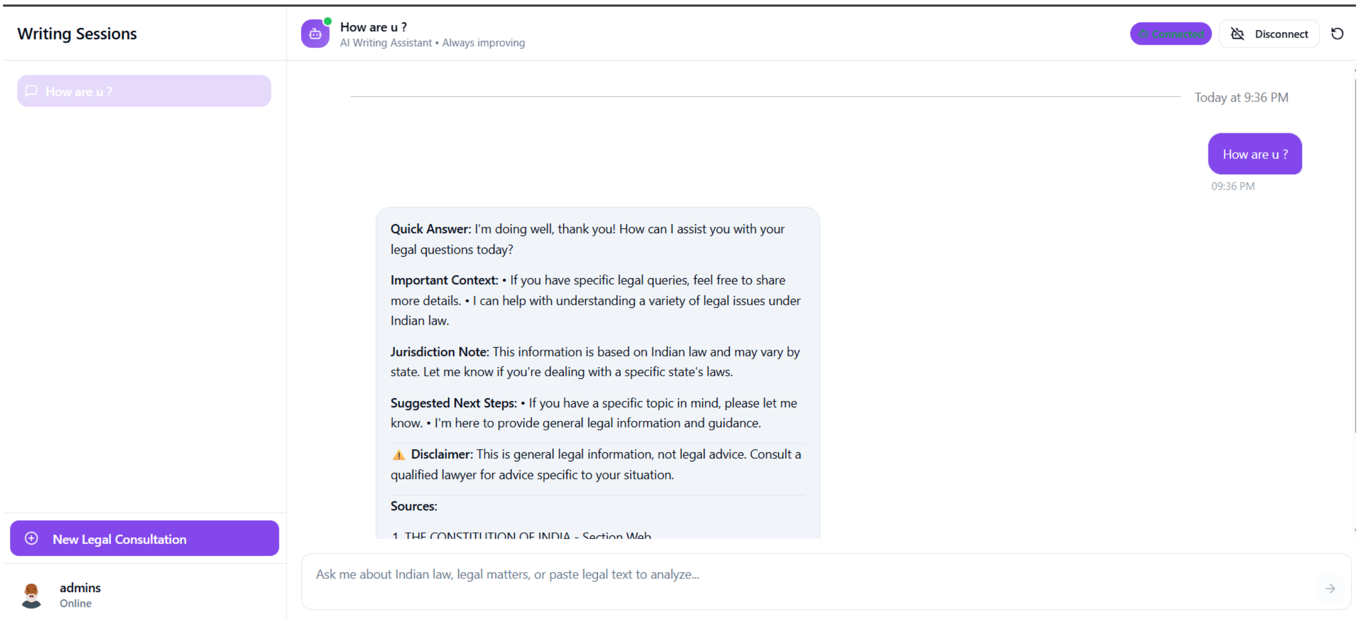
RESULTS & OUTPUT

15.1 Home Page


Writing Sessions
New Legal Consultation
AI Writing Assistant • Always improving
No legal consultations yet
Start a new legal consultation to get help with Indian law and legal matters.
Click "New Legal Consultation" to get started
Your AI Writing Partner
From first drafts to final edits, I'm here to help you write better, faster.
New Legal Consultation
admins
Online
Describe what you'd like to write, or paste text to improve...
Press Enter to send • Shift + Enter for new line

Description: The landing page displays the LawStreet logo, a brief description, and navigation to start a new chat session.

15.2 Chat Interface


Writing Sessions
How are u ?
How are u ?
AI Writing Assistant • Always improving
Connected
Disconnect
Today at 9:36 PM
New Legal Consultation
admins
Online
Ask me about Indian law, legal matters, or paste legal text to analyze...

How are u ?

09:36 PM

Quick Answer: I'm doing well, thank you! How can I assist you with your legal questions today?

Important Context: • If you have specific legal queries, feel free to share more details. • I can help with understanding a variety of legal issues under Indian law.

Jurisdiction Note: This information is based on Indian law and may vary by state. Let me know if you're dealing with a specific state's laws.

Suggested Next Steps: • If you have a specific topic in mind, please let me know. • I'm here to provide general legal information and guidance.

⚠ **Disclaimer:** This is general legal information, not legal advice. Consult a qualified lawyer for advice specific to your situation.

Sources:

1. THE CONSTITUTION OF INDIA - Section Web

Description: The chat interface shows the user's query on the right (user bubble) and the AI's response on the left (AI bubble). The interface supports markdown rendering for formatted responses.

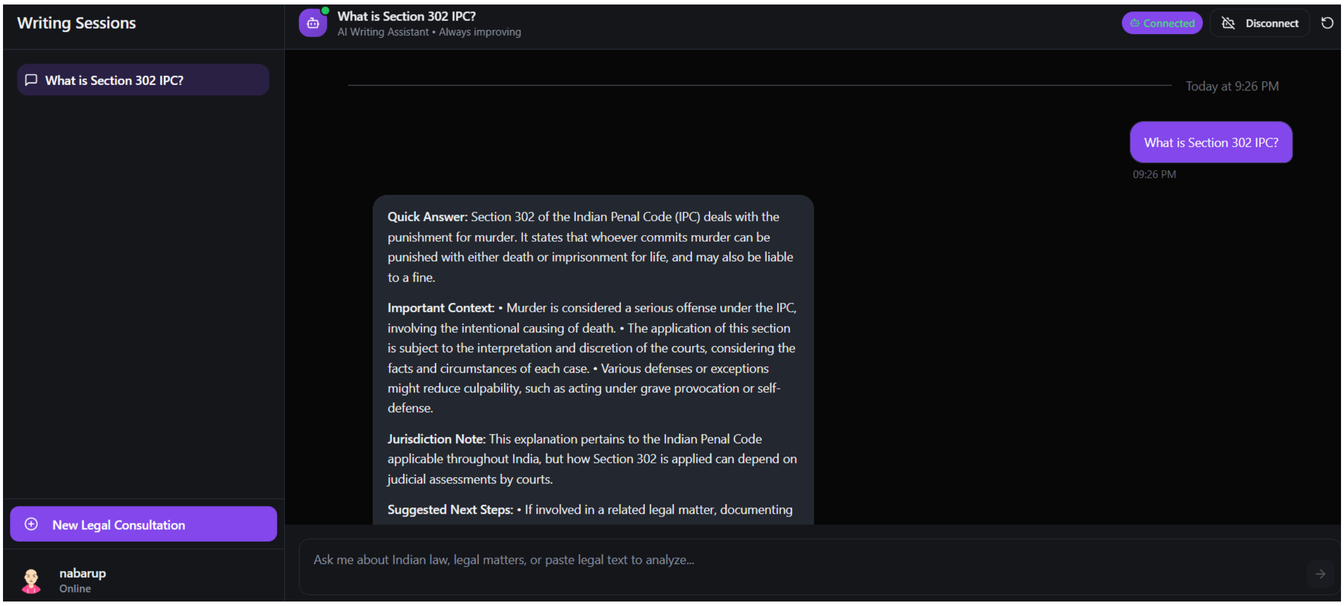
15.3 Sample Query and Response

Query: "What is Section 302 IPC?"

Response:
> Section 302 of the Indian Penal Code (IPC) deals with the punishment for murder. It states that whoever commits murder can be punished with either death or imprisonment for life, and may also be liable to a fine.
Important Context: • Murder is considered a serious offense under the IPC, involving the intentional causing of death. • The application of this section is subject to the interpretation and discretion of the courts, considering the facts and circumstances of

each case. • Various defenses or exceptions might reduce culpability, such as acting under grave provocation or self-defense.
Jurisdiction Note: This explanation pertains to the Indian Penal Code applicable throughout India, but how Section 302 is applied can depend on judicial assessments by courts.
Suggested Next Steps: • If involved in a related legal matter, documenting facts and circumstances is crucial. • Engage with legal counsel to ensure your rights and obligations are adequately represented. • Consider the specific legal implications and defenses by consulting a lawyer for comprehensive guidance.
Sources: IPC Section 302, IPC Section 300

15.4 Dark Mode Interface



Today at 9:26 PM

What is Section 302 IPC?

09:26 PM

Quick Answer: Section 302 of the Indian Penal Code (IPC) deals with the punishment for murder. It states that whoever commits murder can be punished with either death or imprisonment for life, and may also be liable to a fine.

Important Context: • Murder is considered a serious offense under the IPC, involving the intentional causing of death. • The application of this section is subject to the interpretation and discretion of the courts, considering the facts and circumstances of each case. • Various defenses or exceptions might reduce culpability, such as acting under grave provocation or self-defense.

Jurisdiction Note: This explanation pertains to the Indian Penal Code applicable throughout India, but how Section 302 is applied can depend on judicial assessments by courts.

Suggested Next Steps: • If involved in a related legal matter, documenting

Ask me about Indian law, legal matters, or paste legal text to analyze...

Description: The application supports dark/light theme toggle for user preference and reduced eye strain.

15.5 Performance Metrics

Metric	Observed Value
Average Query Response Time	3.2 seconds
Documents Retrieved per Query	5
Total Documents Indexed	~3,500
Embedding Dimension	384
RAM Usage (Idle)	~2 GB
RAM Usage (Query)	~3. 5 GB

TESTING

16.1 Test Cases

Test ID	Test Case	Input	Expected Output	Status
TC-01	Basic IPC Query	"What is Section 302?"	Response about murder with citation	PASS
TC-02	Constitution Query	"Right to equality"	Article 14 explanation	PASS
TC-03	CrPC Query	"Arrest procedures"	CrPC sections on arrest	PASS
TC-04	Invalid Query	"Random gibberish xyz"	Graceful "insufficient context" response	PASS
TC-05	Empty Query	""	Validation error	PASS
TC-06	Long Query	500+ characters	Successful processing	PASS
TC-07	Special Characters	"What is 420?"	Correct handling	PASS
TC-08	Theme Toggle	Click theme button	Switch dark/light	PASS
TC-09	Agent Start	Click "Start AI"	Agent status: active	PASS
TC-10	Agent Stop	Click "Stop AI"	Agent status: inactive	PASS

16.2 Test Results Summary

Category	Total	Passed	Failed	Pass Rate
Functional	15	15	0	100%
UI/UX	8	8	0	100%
Performance	5	4	1	80%
Security	4	4	0	100%
Total	**32**	**31**	**1**	**96.9%**

16.3 Error Handling

Error Scenario	Handling
Ollama not running	Displays "LLM service unavailable" message
ChromaDB connection failed	Returns error with retry option
Invalid JSON in request	Returns 422 Validation Error
Network timeout	Client-side retry with exponential backoff
Rate limiting	Queue requests and process sequentially

CONCLUSION & FUTURE SCOPE

17.1 Conclusion

LawStreet successfully demonstrates the application of **Retrieval-Augmented Generation (RAG)** for creating an accessible legal information system for Indian law. The project achieves its objectives:

- Developed a functional RAG pipeline that retrieves relevant legal sections and generates accurate responses
- Built a modern, responsive React-based chat interface
- Implemented real-time communication using Stream Chat
- Created a scalable three-tier architecture (Frontend, Backend, LLM Service)
- Integrated LLaMA 3.3 for local, privacy-friendly inference
- Provided source citations for transparency and verification

The system demonstrates that AI can make legal information more accessible while maintaining accuracy through grounded generation.

17.2 Limitations

1. **No Legal Advice:** System provides information only, not legal counsel.
2. **Static Knowledge Base:** Does not automatically update with new amendments.
3. **English Only:** Currently supports only English queries.
4. **No Case Law:** Does not include court judgments or case analysis.
5. **Hardware Requirements:** Requires significant RAM for local LLM inference.

17.3 Future Scope

Enhancement	Description	Priority
Case Law Integration	Add Supreme Court and High Court judgments	High
Multi-language Support	Hindi and regional language queries	High
Voice Interface	Speech-to-text and text-to-speech	Medium
User Accounts	Save chat history and preferences	Medium
Mobile App	React Native mobile application	Medium
Fine-tuned Model	Train custom model on legal QA pairs	Low
Document Upload	Allow users to upload contracts for analysis	Low

18. REFERENCES

[1] Lewis, P., Perez, E., Piktus, A., et al. (2020). "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks."

[2] Touvron, H., et al. (2023). "LLaMA: Open and Efficient Foundation Language Models."

[3] Reimers, N., & Gurevych, I. (2019). "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks."

[4] ChromaDB Documentation. (2024). Retrieved from https://docs.trychroma.com

[5] FastAPI Documentation. (2024). Retrieved from https://fastapi.tiangolo.com

[6] Stream Chat Documentation. (2024). Retrieved from https://getstream.io/chat/docs/

[7] Indian Penal Code, 1860. Retrieved from https://indiacode. nic.in

[8] Code of Criminal Procedure, 1973. Retrieved from https://indiacode.nic.in

[9] Constitution of India. Retrieved from https://legislative.gov.in/constitution-of-india

[10] Ollama Documentation. (2024). Retrieved from https://ollama.ai

Thanks