**Smart Judicial Research Assistant Leveraging Retrieval-Augmented Generation and Graph-Aware Instruction Tuning**

**1. Executive Summary**

Judges in India face a massive challenge: over 45 million pending cases and thousands of pages of legal material to analyse for each matter. Civil and criminal cases alike require careful review of statutes, prior judgments, and factual records before a decision can be made. This process is time-consuming and contributes to significant case backlogs.

The Judicial Assistant aims to reduce the research burden on judges by providing an AI-powered tool that:

1. Retrieves relevant statutes and precedents given a case description.
2. Highlights factual and legal similarities between the current case and past judgments.
3. Generates draft judgment structures *(Facts → Issues → Law → Analysis → Order)* to streamline judicial writing.

The system leverages a domain-adapted legal language model trained on Indian judgments, combined with retrieval-augmented generation (RAG) for explainable outputs. Unlike judgment prediction systems, this assistant does not replace judicial reasoning—it supports judges with faster, structured research and drafting tools.

**2. Project Background**

India’s judiciary faces unprecedented caseloads. According to the National Judicial Data Grid (2025), pendency exceeds **4.5 crore cases**, and High Court judges may handle hundreds of matters per month. Much of their time is spent on:  
• Locating relevant statutes across overlapping laws.  
• Searching for similar precedents from decades of judgments.  
• Drafting lengthy orders that follow consistent legal structure.

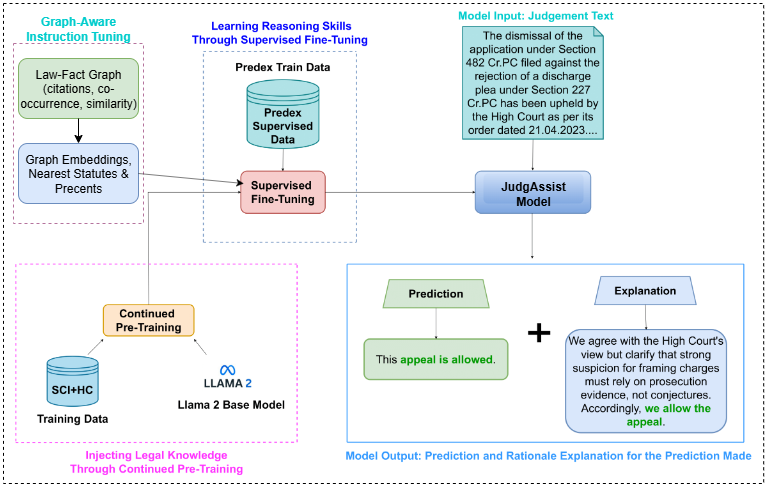
Past research includes:  
• Information retrieval systems (keyword-based): Limited in handling legal semantics.  
• Transformer models like LegalBERT: Good at legal text understanding, but not tuned to Indian contexts.  
• CJPE/ILDC (Court Judgment Prediction with Explanation) — compiled an ~35k-case Indian Supreme Court corpus and formalized CJPE with baselines and occlusion-based rationales; yet a clear gap remains: judge-centric assistants that prioritize statute/precedent retrieval and structured draft orders rather than predicting verdicts.

While recent domain-adapted legal language models have improved outcome classification and rationale generation, the need persists for tools explicitly designed to accelerate legal research and produce well-structured draft orders—all while keeping the final decision firmly with the judge.

**3. Proposed Solution**

**Vision Statement**Equip High Court judges with an AI research assistant that retrieves the most relevant statutes and precedents, explains their connection to the facts, and assembles court-ready draft structures—while leaving adjudication entirely to the bench.

**Architecture**



**Approach & Methodology  
• Data Processing:** - Use large-scale Indian judgments corpus (1950–2025).  
 - Extract structured fields: Facts, Statutes cited, Reliefs/Orders, Citations of Precedents.

**• Embeddings:** - Use InLegalBERT / InLegalLlama embeddings for legal semantics.

**• Retrieval Module:** - Build FAISS indexes of statutes and precedent case summaries.  
 - Given new case facts, retrieve top-k statutes and precedents.

**• Draft Structuring:** - Use JudgAssist in RAG mode to organize retrieved material into a draft judgment structure: **▪ Facts → Issues → Applicable Law → Findings → Order.**

**• Explainability:** - Highlight retrieved statutes and precedents with similarity scores.  
 - Display attention-based heatmaps to show fact–law relevance.  
 - Graph-Aware Instruction Tuning: append a short context footer with 3–5 graph-nearest statutes or precedents to ground explanations in authoritative sources.

**Project Schedule & Milestones**

• Phase 1 (Weeks 1–2): Dataset preparation (civil & criminal judgments).  
• Phase 2 (Weeks 3–4): Embedding generation & retrieval index construction.  
• Phase 3 (Weeks 5–8): RAG pipeline development (statutes + precedent retrieval).  
• Phase 4 (Weeks 9–10): Draft judgment structuring module.  
• Phase 5 (Weeks 11–12): Evaluation, demo app, and final documentation.

**4. Deliverables**

1. Civil & Criminal Case Dataset (structured).
2. Judicial Assistant Model combining retrieval + InLegalLlama (JudgAssist).
3. Precedent Retrieval Tool (FAISS-based).
4. Statute Finder that maps facts → legal provisions.
5. Draft Judgment Structures for judge review.
6. Demo Application (Streamlit/Flask) for user interaction.
7. Final Report & Research Paper Submission.

**5. Resources Needed**

**Project Budget (Estimated)**  
• **Computational Resources:** GPU compute (Colab Pro+/AWS EC2).  
• **Data Storage:** Cloud storage for large corpus.  
• **Software Tools:** Open-source stack (PyTorch, HuggingFace, FAISS).  
• **Miscellaneous:** Hosting, demo app deployment.  
**Estimated Total:**

**Resource Allocation Plan**  
• 60% – Compute for training and inference.  
• 20% – Data storage & preprocessing.  
• 10% – Hosting and demo app deployment.  
• 10% – Documentation and reporting.

**6. Conclusion**

The Judicial Assistant for Legal Research is designed not to predict judicial outcomes, but to support judges in their work. By combining JudgAssisst Model with retrieval-augmented generation, the system will:  
• Provide relevant statutes and precedents in seconds.  
• Highlight connections between case facts and applicable laws.  
• Generate structured draft judgments to save writing time.

This project has the potential to significantly reduce research burdens, enhance transparency, and help address the pendency crisis, while keeping the final decision firmly in the hands of the judge.