A REPORT

ON

AI POWERED LEGAL DOCUMENTATION ASSISTANT

Submitted by,

Metta Siva Nanda Reddy	**20211CSE0126**
Dalavai Geetha Sree	**20211CSE0493**
Mohammed Noaman Ahmed	**20211CSE0559**

Under the guidance of,

Ms. AKKAMAHADEVI C

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

At

PRESIDENCY UNIVERSITY

BENGALURU

MAY 2025

PRESIDENCY UNIVERSITY

PRESIDENCY SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Project report **"AI POWERED LEGAL DOCUMENTATION ASSISTANT"** being submitted by "Metta Siva Nanda Reddy, Dalavai Geetha Sree, Mohammed Noaman Ahmed" bearing roll number(s) "20211CSE0126, 20211CSE0493, 20211CSE0559" in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology in Computer Science and Engineering is a bonafide work carried out under my supervision.

Ms. Akkamahadevi C
Assistant Professor
PSCS/PSIS
Presidency University

Dr. Asif Mohamed
Associate Professor & HoD
PSCS
Presidency University

Dr. MYDHILI NAIR
Associate Dean
PSCS
Presidency University

Dr. SAMEERUDDIN KHAN
Pro-Vice Chancellor -Engineering
Dean –PSCS/PSIS
Presidency University

PRESIDENCY UNIVERSITY

PRESIDENCY SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

We hereby declare that the work, which is being presented in the project report entitled **AI POWERED LEGAL DOCUMENTATION ASSISTANT** in partial fulfillment for the award of Degree of **Bachelor of Technology** in **Computer Science and Engineering**, is a record of our own investigations carried under the guidance of **Ms. AKKAMAHADEVI C, Assistant Professor, School of Computer Science Engineering & Information Science, Presidency University, Bengaluru.**

We have not submitted the matter presented in this report anywhere for the award of any other Degree.

Metta Siva Nanda Reddy-20211CSE0126

Dalavai Geetha Sree-20211CSE0493

Mohammed Noaman Ahmed-20211CSE0559

ABSTRACT

Legal documentation plays a critical role in business operations, regulatory compliance, and individual legal matters, yet it remains a complex, time-consuming, and often costly process. Many individuals and small businesses struggle with understanding legal jargon, ensuring compliance, and managing contracts without professional legal assistance, leading to risks of misinterpretation and legal disputes. This report introduces an AI-powered Legal Documentation Assistant that leverages the Gemini API to automate key legal processes, including document analysis, clause extraction, contract summarization, legal Q&A, and AI-driven drafting. By utilizing advanced natural language processing, the system simplifies legal language, enhances accuracy, and ensures consistency in legal documentation. It minimizes human errors, reduces reliance on expensive legal services, and enables faster legal decision-making. The platform integrates secure data handling, intuitive user interfaces, and AI-driven legal insights to provide an accessible, efficient, and reliable solution for individuals, businesses, and legal professionals. By streamlining legal workflows, improving transparency, and reducing costs, this solution democratizes legal access, making legal processes more inclusive and equitable. Its implementation will empower users with the confidence to navigate legal complexities efficiently while ensuring compliance with legal standards.

ACKNOWLEDGEMENT

First of all, we indebted to the **GOD ALMIGHTY** for giving me an opportunity to excel in our efforts to complete this project on time.

We express our sincere thanks to our respected dean **Dr. Md. Sameeruddin Khan**, Pro-VC, School of Engineering and Dean, School of Computer Science Engineering & Information Science, Presidency University for getting us permission to undergo the project.

We express our heartfelt gratitude to our beloved Associate Deans **Dr. Mydhili Nair,** Presidency School of Computer Science Engineering, Presidency University, and **Dr. Asif Mohamed**, Head of the Department, School of Computer Science Engineering , Presidency University, for rendering timely help in completing this project successfully.

We are greatly indebted to our guide **Ms. Akkamahadevi C,** Assistant Professor and Reviewer **Mr. Md Ziaur Rahman**, Assistant Professor, School of Computer Science Engineering , Presidency University for her/his inspirational guidance, and valuable suggestions and for providing us a chance to express our technical capabilities in every respect for the completion of the project work.

We would like to convey our gratitude and heartfelt thanks to the CSE7301 University Project Coordinators **Mr. Md Ziaur Rahman and Dr. Sampath A K,** department Project Coordinators **Dr. Jayanthi. K.** and Git Hub coordinator **Mr. Muthuraj.**

We thank our family and friends for the strong support and inspiration they have provided us in bringing out this project.

Metta Siva Nanda Reddy

Dalavai Geetha Sree

Mohammed Noaman Ahmed

LIST OF TABLES

LIST OF FIGURES

TABLE OF CONTENTS

CHAPTER-1
INTRODUCTION

Legal documentation is essential in managing commercial transactions, forming partnerships, and ensuring regulatory compliance. However, in the Indian context, the legal process is often seen as complex, inaccessible, and difficult to navigate. Legal contracts and agreements are typically filled with highly technical language, making them challenging to interpret for individuals without formal legal education. For small businesses, independent professionals, and startups, this poses a substantial problem—they must either spend large sums on legal services or take risks by interpreting documents themselves. Even a minor misunderstanding of a clause could lead to expensive legal consequences or breached contracts.

Traditional approaches to legal documentation remain heavily dependent on manual processes. Reviewing or modifying legal drafts often involves printing documents, marking them by hand, scanning, and emailing back and forth—steps that are not only inefficient but also prone to human error. Furthermore, the legal environment is constantly changing, with frequent updates to tax rules, labor codes, and data privacy regulations. As a result, a document that was legally compliant in the past may become outdated or non-compliant within a year, further complicating legal workflows and increasing the need for continuous document updates.

Despite the growing digital transformation across various sectors like finance and healthcare, the legal industry continues to lag behind in adopting intelligent automation tools. Most digital legal solutions are limited in functionality and do not offer intuitive interfaces or advanced features like real-time clause detection or semantic simplification. Consequently, many users avoid legal formalities altogether or depend on one-size-fits-all templates, which often lack the necessary specificity to meet legal standards. This results in documents that are either legally insufficient or carry substantial risk. To counter these inefficiencies, an AI-driven legal document assistant, built on the Gemini API, has been developed. This assistant incorporates advanced natural language processing (NLP) capabilities to interpret, analyze, and generate legal documents that are both accurate and easy to understand. It can identify and summarize key clauses such as payment terms, deadlines, and confidentiality provisions,

transforming dense legal text into simple summaries. By providing quick insights into a document's most important components, users are empowered to make informed decisions without needing a legal expert to interpret the fine print.

Beyond merely summarizing content, the assistant actively detects missing elements or risky clauses—such as unbalanced indemnity terms or outdated compliance references—and suggests appropriate alternatives. These features not only improve the reliability of legal documents but also ensure they meet current legal standards. With a few guided inputs through a Q&A interface, users can generate complete templates tailored to their requirements. Whether it's a service agreement, a lease, or a non-disclosure agreement, the platform customizes each document using the latest legal frameworks relevant to Indian law.

This tool eliminates the time delays and financial burdens typically associated with legal consultations. Users can prepare and finalize agreements quickly, giving entrepreneurs and independent professionals the agility to operate at business speed without the legal bottlenecks. Moreover, it democratizes access to legal support, enabling even those without legal training to confidently draft and understand legal documents. As the system evolves, it is expected to offer even more personalized services, setting a new standard for how everyday users engage with legal processes in India.

Another major challenge lies in the lack of accessible, affordable legal services across many regions. Particularly in rural and semi-urban parts of India, legal expertise is either scarce or unaffordable. Even in metropolitan areas, the high cost of hiring lawyers makes routine document review and drafting impractical for startups and independent professionals. This has led many to rely on generic online templates or verbal agreements—both of which leave them vulnerable to legal risks. There is a growing demand for solutions that provide professional-quality legal support in a scalable and user-friendly format.

The manual preparation of legal documents also raises concerns regarding accuracy. Errors such as omitted clauses, vague wording, or legal inconsistencies are common in documents created without professional oversight. These mistakes may not be immediately noticeable but can lead to conflicts, penalties, or invalidated agreements later on. By integrating automation into the drafting process, such issues can be minimized. AI systems offer

consistency, reduce manual workload, and apply legal templates accurately, thus improving the overall quality and reliability of legal documentation.

Moreover, the scalability of AI systems makes them particularly effective in democratizing legal access across different demographics. These tools can serve thousands of users simultaneously without compromising on quality or accuracy. For institutions, this scalability offers a cost-effective way to manage repetitive legal tasks, while for individuals, it removes barriers created by geography, language, and cost. With cloud-based deployment, AI assistants can be accessed from anywhere, providing timely support even in urgent situations. As these platforms evolve, they may also incorporate multilingual support, voice-based interaction, and integration with digital government services—further enhancing accessibility and inclusion.

The ongoing progress in artificial intelligence represents a pivotal moment for innovation within the legal sector. Specialized AI models trained on legal data are capable of analyzing detailed contractual language, flagging potential compliance concerns, and offering real-time recommendations for improvement. Systems like the Gemini-based assistant promote a forward-thinking approach by enabling users to address legal risks proactively rather than responding after complications have emerged. As these technologies become more prevalent, they hold the potential to streamline legal operations, reduce dependency on traditional legal services, and increase public understanding of legal matters—ultimately empowering individuals to handle legal responsibilities with greater confidence and independence.

CHAPTER-2

LITERATURE SURVEY

S.No.	Authors	Title	Year	Summary
1	V. Gupta and R. Kumar, IEEE Access	A Service of Legal Text Techniques for Indian Legal Documents	2023	Reviewed NLP strategies tailored for Indian legal texts. Their work tackled language inconsistencies and document variations while examining key preprocessing and classification models. The authors emphasized the scarcity of specialized datasets in the legal domain. It identified inefficiencies in current annotation practices.
2	A. Arora and A. Soni, Springer / IAEME	Natural Language Processing for Legal Documentation in Indian Languages	2023	Discussed NLP applications in regional Indian languages, pointing out the challenges of language diversity and proposing frameworks for multilingual document handling. They advocated for building language models trained on Indic corpora. Their future work targets integration with legal aid services in rural regions.
3	P. Kumar and V. Bansal, Information Technology Journal	Natural Language Processing for Automated Legal Document Analysis and Contract Review	2023	Presented approaches for automating contract analysis through NLP, with a focus on extracting clauses and identifying risks. Their system, leveraging machine learning, demonstrated real-world efficiency in law firms and included performance comparisons against manual review. The paper concluded with a call for better contract annotation tools.

4	K. Wallace, The National Magazine	Plain Language Legal Writing: Part I – Writing as a Process	2021	Advocated for the use of clear, accessible legal writing. The author highlighted structured, iterative writing processes aimed at improving comprehension and reducing legalese in public documents.
5	R. Paul, T. Shah, and A. Patel, Springer / ACM	A Comprehensive Analysis of Indian Legal Documents	2023	Reviewed the organization and semantics of Indian legal texts with an emphasis on digitization. They applied machine learning for segmentation and metadata tagging, proposing schema-based structures for improved document consistency and retrieval. The authors also proposed a modular legal archive model for nationwide deployment.
6	P. Kumar, S. Reddy, and V. Sharma, IEEE	Automated Techniques on Indian Legal Documents: A Review	2023	Surveyed recent trends in legal automation in India. Their work emphasized summarization and tagging of legal cases using hybrid systems, noting limitations posed by ambiguous legal terminology.
7	A. Singh and R. Sharma, Springer	Sentiment Analysis for Legal Judgment Text in India's Supreme Court	2023	Applied sentiment analysis to Supreme Court judgments, revealing patterns linked to judicial decision-making. The study raised ethical concerns around sentiment interpretation and promoted model explainability. Their analysis found trends in case outcomes based on sentiment polarity. They acknowledged risks of oversimplifying legal reasoning. Recommendations included transparency and explainability tools for legal AI outputs.

8	S. K. Singh, A.K. Jain, and P.K. Gupta, SN Applied Sciences(Springer)	A Two-Staged NLP-Based Framework for Assessing the Sentiments on Indian Legal Documents	2023	Introduced a two-phase NLP approach for extracting sentiments from Indian legal texts. Their hybrid system improved performance and offered a modular framework suitable for civil and criminal law. Evaluation metrics showed marked improvements over baseline sentiment tools. The authors discussed deployment feasibility in Indian courts.
9	M. Sharma and R. Verma , Artificial Intelligence and Law(Springer)	A Case Study for Automated Attribute Extraction from Legal Documents Using NLP	2024	Presented a case study on extracting legal entities and attributes using named entity recognition and dependency parsing. The research emphasized the benefits of legal ontologies and demonstrated scalability. They highlighted challenges in linking entities to legal statutes. Future enhancements included multi-language support and richer ontologies.
10	A. Patel, B. Kumar, and C. Singh, IEEE AIML / IJNRD	LAWBOT: A Smart User Indian Legal Chatbot Using Machine Learning Framework	2023	Developed *LAWBOT*, a machine learning-based Chatbot to assist users in understanding Indian legal documents. It utilized intent recognition and retrieval techniques, supporting legal literacy and access to justice. The Chatbot responded to real user queries in both Hindi and English. Accuracy in intent detection improved with continuous user feedback. The system aimed to reduce reliance on costly legal consultations for basic queries.

Table.2.1 Literature Survey

CHAPTER-3
RESEARCH GAPS OF EXISTING METHODS

Legal documentation serves as a critical component in formalizing agreements, ensuring regulatory compliance, and safeguarding parties' interests in both business and personal transactions. These documents act as reference points for rights and responsibilities, making them foundational to dispute resolution and risk management. Yet, the traditional methods associated with preparing and managing legal documents remain outdated and inaccessible to a large section of the population, especially in emerging economies like India. A closer look at these practices reveals several long-standing issues that have not been fully addressed by current technologies or workflows.

Complexity of Legal Terminology

One of the most significant obstacles in legal document accessibility is the use of complex and specialized language. Legal texts are often laden with archaic expressions, long compound sentences, and technical terms that are difficult for the average person to comprehend. This disconnect between legal language and layperson understanding leads to frequent misinterpretation and misuse. Many users either skip reading important clauses or sign documents without a clear understanding of their implications. While some efforts have been made to introduce plain language alternatives, these are far from widespread or standardized. For most individuals, legal documents remain an intimidating puzzle, requiring expert assistance for even basic interpretation.

Time-Intensive Procedures

The process of drafting and reviewing legal documents involves numerous repetitive steps, including consultations, revisions, and mutual approvals. These workflows are often slowed down by back-and-forth communication between stakeholders and frequent manual edits. Even minor changes can trigger a full-cycle re-review, adding days or weeks to project timelines. Businesses that rely on rapid execution, especially startups or freelance professionals, may face operational bottlenecks due to these delays. Time-sensitive opportunities may be lost when legal documentation cannot be finalized promptly.

High Legal Costs

Professional legal services come at a premium, particularly in metropolitan areas or niche legal domains. For small enterprises and individuals, the costs associated with document drafting, legal review, or compliance consultation can be prohibitively high. As a result, many users choose to bypass legal services altogether or rely on informal templates found online—often at the cost of legal accuracy and enforceability. This economic disparity reinforces unequal access to justice and legal security, especially in developing or underserved markets.

Susceptibility to Human Error

Manual preparation of legal documents exposes the process to various kinds of human mistakes, such as typographical errors, omitted clauses, or inconsistent formatting. These errors may not be caught until the contract is reviewed in a legal dispute or regulatory audit. In complex documents, the absence of cross-referencing tools or automated compliance checkers increases the likelihood of contradictions and unclear responsibilities. Under pressure or workload constraints, legal professionals may overlook critical inconsistencies. Without automated checks in place, such flaws often go unnoticed until disputes arise, resulting in lengthy resolution processes and avoidable costs

Inadequate Access to Legal Resources

Access to professional legal help remains highly uneven across geographies. Urban areas typically have a concentration of lawyers and legal firms, while rural regions struggle with a lack of trained professionals. This gap forces many individuals to rely on informal advice or outdated sample documents, both of which can lead to substantial legal risk. Furthermore, digital tools that could bridge this gap are either too generic or fail to account for local laws, languages, and document formats. Without localized and affordable legal assistance, the population remains at a disadvantage, unable to secure their rights or enforce obligations effectively. The lack of awareness about legal processes further compounds this issue, especially among underrepresented groups, first-time entrepreneurs, or marginalized communities.

CHAPTER-4
PROPOSED METHODOLOGY

To address the challenges posed by traditional legal documentation processes such as complexity, limited accessibility, high costs, and a high risk of errors, this project proposes a modern, AI-powered system designed to simplify, automate, and enhance legal document handling. By leveraging the capabilities of a Gemini-based natural language processing engine, the system can analyze, summarize, and generate legal content in a way that is both accurate and user-friendly. This approach is particularly suited to individuals, startups, and small businesses lacking constant access to legal professionals.

Core Components of the System

Legal Document Analysis and Keyword Extraction

This module scans uploaded legal documents such as contracts, agreements, or NDAs and identifies key information such as clauses, deadlines, conditions, and obligations. It uses AI to understand the semantic structure of each document, allowing it to highlight sections that demand user attention.

How it Works: Users upload a text file through the frontend interface. The Gemini-based AI processes the document, extracts critical legal terms and clauses using context-aware algorithms, and displays them with visual highlights. This helps users pinpoint essential content without needing to read the entire document line by line.

Technologies Used:

- **Frontend:** React.js, HTML5, CSS3, and JavaScript for interactive user experience.
- **Backend:** Python with Django to process requests and communicate with AI engine.

Document Summarization Using NLP

This module enables the summarization of long legal documents into concise bullet-point summaries. The goal is to enhance readability and support quicker decision-making by reducing cognitive load on the user.

How it Works: After analysis, the AI segments the document into logical parts and condenses each section using natural language generation. The result is a summary that captures the core responsibilities, time frames, and risks presented in the original text.

Technologies Used:

- **AI Model:** Gemini's natural language processing system**.**
- **Processing Logic:** Python-based integration to parse, summarize, and return content

Template-Based Document Generation

In this module, users can create ready-to-use legal documents such as service agreements, NDAs, or lease contracts by answering a guided set of questions. The AI uses the input data to dynamically generate a complete and legally coherent draft.

How it Works: Users are presented with a form that collects basic information about the parties, transaction details, obligations, and timelines. The AI uses this data to populate a legal template, embedding appropriate clauses based on Indian legal standards and industry norms.

Technologies Used:

- **AI Language Generator:** Gemini engine for generating context-sensitive legal language.
- **Data Storage** AWS S3 is used to securely store user templates, preferences, and document history.

Advantages of the Proposed System

The integration of AI into legal documentation offers numerous tangible benefits over traditional approaches:

- **Improved Comprehension:** Legal content is converted into plain language summaries, making documents more accessible to users without legal training.
- **Time Efficiency:** Automated document analysis and generation drastically reduce the time required for legal tasks from hours or days to just minutes.
- **Reduced Human Error:** Automated checks help identify missing clauses and inconsistencies, minimizing risks associated with manual drafting.
- **Greater Accessibility:** Users from any location can create and review legal documents through a web interface without needing in-person legal consultations.
- **Customizability and Scalability:** The AI engine adapts templates based on context, making it useful across sectors and scalable for large-scale deployment.

CHATER-5
OBJECTIVES

The primary goal of this project is to design and implement an intelligent legal document assistant that simplifies legal workflows and makes legal services more accessible to users with limited legal knowledge. The system aims to eliminate barriers commonly found in traditional legal processes by offering automated solutions for document analysis, summarization, and generation. The following are the major objectives of the proposed system:

Increase Accessibility to Legal Services and Tools

The platform is structured to benefit users who may not have easy access to legal professionals due to financial limitations or geographic isolation. It introduces a set of AI-assisted features that enable individuals and small organizations to independently manage legal documents without requiring legal expertise. Through a user-friendly interface and intuitive workflows, the system empowers users to engage with legal content confidently. It also helps users understand legal terminology more clearly by simplifying complex clauses, thereby promoting legal literacy. In doing so, the system narrows the gap between professional legal expertise and layperson usability.

Improve Efficiency in Legal Documentation

Traditional legal documentation is often a time-consuming process that involves numerous rounds of drafting, reviewing, and editing. The proposed AI-based system aims to eliminate unnecessary delays by automating several of these stages. By integrating intelligent summarization and document generation capabilities, users can produce complete legal drafts in a matter of minutes. This efficiency is further enhanced through the ability to reuse previously saved templates and edit them dynamically. The system also supports quick preview and revision features, allowing for faster decision-making and reducing the turnaround time for legal documentation workflows.

Ensuring Accuracy, Consistency, and Legal Compliance

One of the primary goals of the assistant is to improve the reliability and quality of legal documents. Manual errors such as missing clauses, outdated references, or inconsistent formatting are common issues that the system seeks to eliminate. By following structured legal templates and incorporating industry-standard clauses, the platform ensures a consistent format across all documents. It cross-verifies content for completeness and alignment with relevant legal regulations such as the Indian Contract Act and data privacy rules. As a result, users can create legally sound documents with minimized risk of disputes or legal liabilities.

Reduce Legal Costs and Enhance Affordability

Engaging professional legal services for documentation can be costly, particularly for startups, freelancers, and small enterprises. This project introduces a cost-effective alternative that automates basic documentation tasks, reducing the need for regular consultations. Users can manage legal operations internally without expanding legal teams, thereby saving on operational costs. The platform is particularly useful for users operating within limited budgets, as it allows for legally compliant document generation without recurring professional fees. By lowering these financial barriers, the system fosters wider inclusion and accessibility within the legal domain.

Support Customization and Real-Time Interactivity

Since legal requirements vary based on context, the platform is built with flexibility and adaptability at its core. Users can input case-specific details, and the system generates customized legal documents accordingly. It also allows users to modify particular clauses in real time, supported by AI-generated suggestions to enhance accuracy. Collaborative editing is enabled so multiple stakeholders can contribute to a document simultaneously, improving team-based legal workflows. Moreover, the system regularly updates its templates to reflect the latest changes in regulatory requirements, ensuring that all documents are up-to-date and legally relevant.

CHAPTER-6

SYSTEM DESIGN & IMPLEMENTATION

- **System Architecture Diagram**

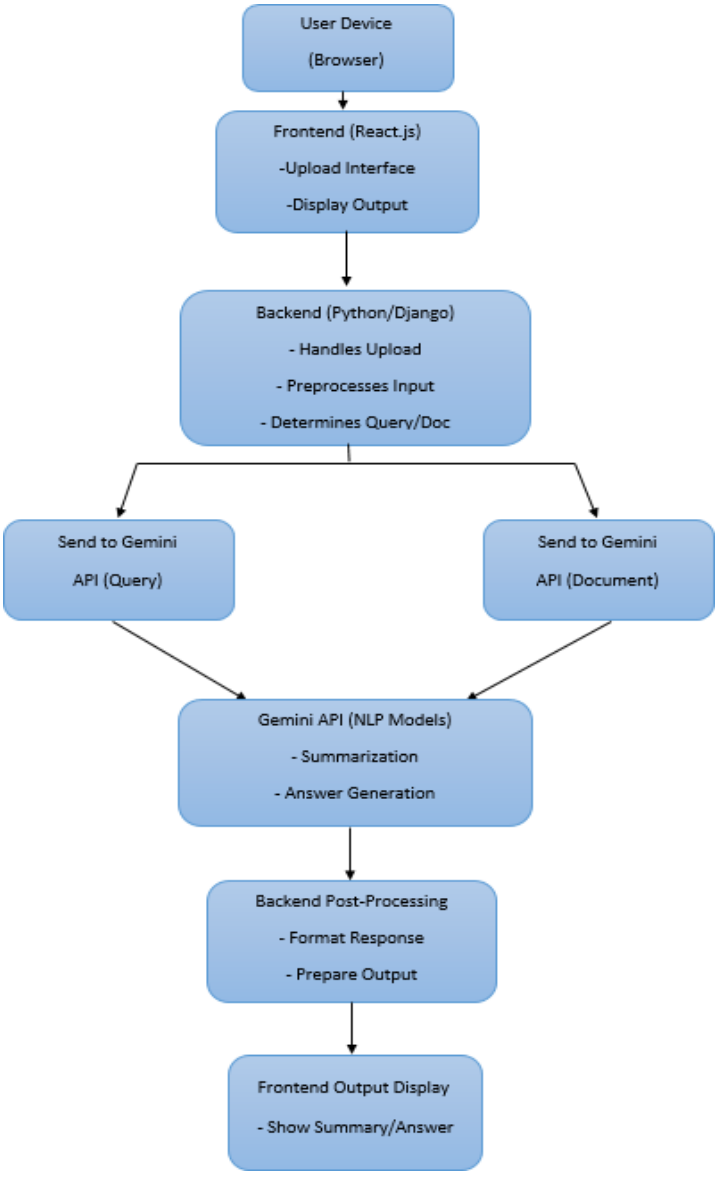

Figure.6.1 System Architecture Diagram(State Transition Diagram)

- **User Flow Diagram**

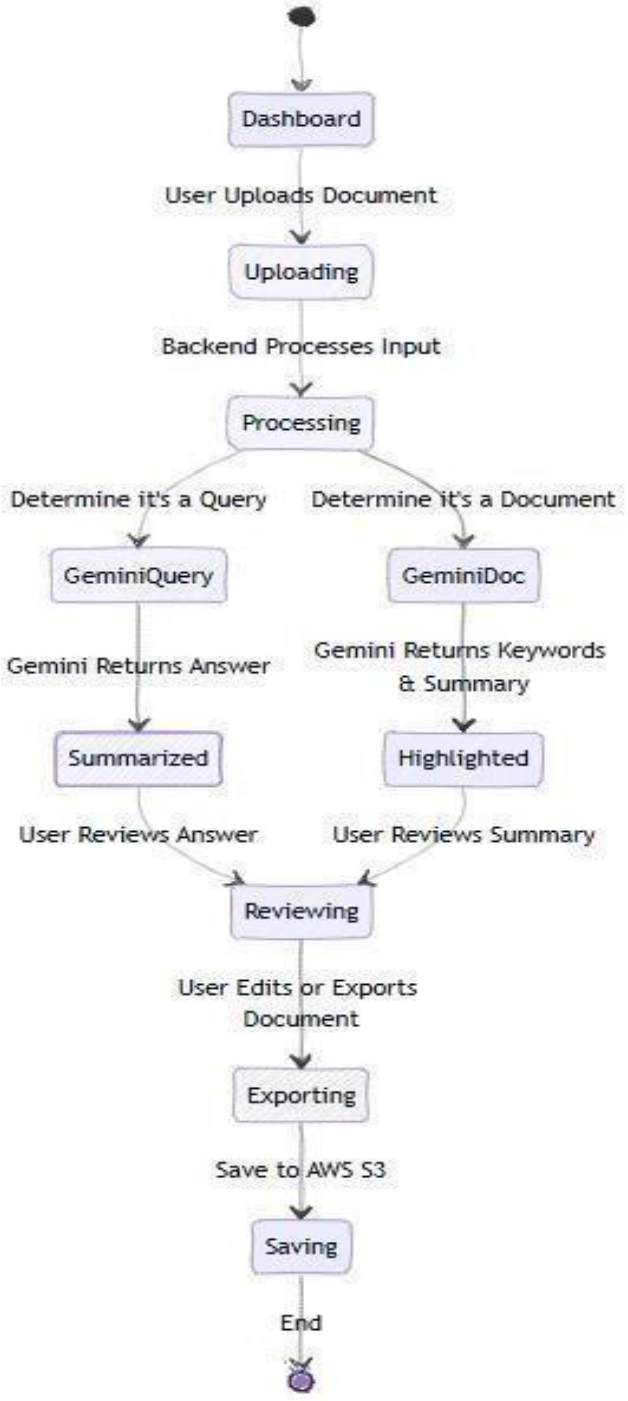

Figure.6.2. User Flow Diagram

CHAPTER-7

TIMELINE FOR EXECUTION OF PROJECT (GANTT CHART)

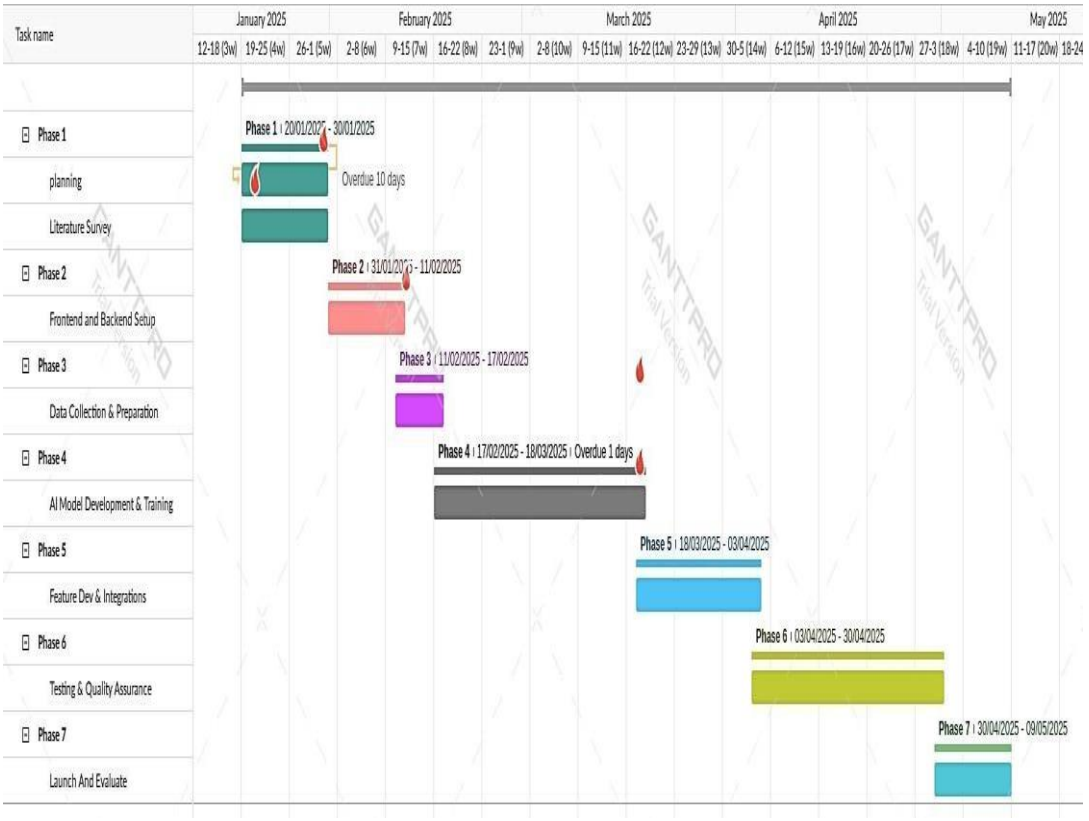

Figure 7.1. Gnatt chart

CHAPTER-8
OUTCOMES

The integration of an AI-powered legal document processing assistant promises transformative changes in the legal services landscape. This innovation is expected to benefit individuals, small enterprises, and legal practitioners by enhancing service accessibility, lowering costs, increasing accuracy, and simplifying the drafting and management of legal documents.

Expanding Access to Legal Support

Simplifying Legal Workflows for Non-Experts: The platform is designed with a strong focus on accessibility and ease of use, making it suitable for individuals without formal legal education—including entrepreneurs, small business operators, and first-time users. Tasks such as drafting contracts, non-disclosure agreements (NDAs), and memorandums of understanding (MoUs) can be carried out with minimal effort, thanks to an intuitive interface that demystifies complex legal procedures.

Multilingual and Inclusive Document Creation: A key feature enhancing accessibility is the platform's support for multiple languages. Users can interact with the assistant and generate legal documents in their preferred language, which is particularly beneficial in a multilingual country like India. This functionality breaks down linguistic barriers that often prevent individuals from engaging with legal systems. Whether a user prefers Hindi, Tamil, Bengali, or another regional language, the platform ensures that legal content is both understandable and accurately translated.

Self-Guided Legal Document Generation: By offering automated drafting and summarization features, the system enables users to create legally relevant documents on their own. Its design caters to a wide audience by incorporating multilingual support, thereby ensuring inclusivity across linguistic and regional boundaries. This helps broaden the user base, making legal resources available to those who were previously underserved due to language or complexity barriers.

Reducing Legal Costs and Saving Time

Lowering the Cost of Legal Services: A significant advantage of this AI platform is its ability to drastically reduce the expenses traditionally associated with legal assistance. By automating document preparation, the need for professional legal review in standard cases is minimized. This cost-saving feature is particularly valuable for small businesses, freelancers, and startups operating on tight budgets.

Accelerating Document Delivery: The assistant can generate complete, legally compliant documents within minutes—drastically reducing the time spent waiting for lawyer reviews or document revisions. This speed is critical for time-sensitive matters such as closing deals, onboarding employees, or formalizing vendor agreements. Users can quickly respond to legal needs without procedural delays.

Minimizing Dependence on Legal Consultations: Under traditional systems, individuals often consult legal professionals for even basic documentation, which can be both time-consuming and costly. The AI assistant eliminates this barrier by enabling users to produce documents independently. This not only reduces the frequency of paid consultations but also makes legal resources more broadly accessible.

Improving Precision and Legal Compliance

Minimizing Human Error: Manual drafting is susceptible to oversight whether through omitted clauses, inconsistent phrasing, or incorrect legal terminology. The AI system mitigates these risks by referencing predefined legal templates and industry standards. This ensures each document is generated with structural integrity and legal soundness, reducing the chance of disputes or non-compliance.

AI Validation and Legal Standard Alignment: Backed by the Gemini framework, the assistant cross-checks every document against a database of current laws, regulatory norms, and standard practices. It detects inconsistencies, flags missing components, and proposes corrections in real time. This validation mechanism significantly enhances both the reliability and enforceability of the final document.

Ensuring Uniform Language and Formatting: Consistency is essential in legal writing, especially when businesses produce multiple agreements across departments. The assistant ensures that every document maintains a standardized tone, structure, and terminology helping companies preserve uniformity in communications with clients, partners, and employees.

Building Trust Through Clarity and Security

Plain Language Legal Content: Legal terminology can often confuse non-experts, leading to misinterpretation of key clauses. This AI assistant addresses the issue by translating complex legal jargon into straightforward, readable language. Users gain a clearer understanding of their rights and responsibilities, allowing them to make informed decisions with greater confidence.

Real-Time Monitoring of Document Workflow: To foster transparency, the platform provides users with live updates throughout the document lifecycle from initial drafting to final approval. Users can track progress, request changes, or approve versions as needed, ensuring full control and visibility during every stage of development.

Securing Sensitive Legal Information: Given the confidential nature of legal content, data protection is integral. The platform employs robust encryption protocols, role-based access, and compliance with major data privacy frameworks such as GDPR and India's PDPB. These safeguards assure users that their information is protected from unauthorized access or data breaches throughout their interaction with the system.

CHAPTER-9
RESULTS AND DISCUSSIONS

Results

The AI-powered legal document assistant developed in this project has demonstrated strong performance in enhancing the efficiency and reliability of legal documentation processes. The system focuses on automation, user accessibility, and language simplification, which together contribute to faster document handling, cost savings, and improved user confidence. The main results are outlined below.

Fast and Accurate Document Generation

The platform enables users to generate professional legal documents such as contracts, NDAs, and agreements within minutes.

- **Structured Templates**: The system provides ready-made templates tailored to various legal scenarios. Users can input information through a step-by-step form.
- **Real-Time Output**: Documents are created instantly based on user input, significantly reducing the time spent on drafting.
- **Legal Formatting and Consistency**: Each document follows standard legal formatting and terminology, ensuring clarity and uniformity.

Simplified Legal Language and Summaries

The assistant improves user comprehension by translating complex legal language into simpler terms and summarizing key sections.

- **Plain-Language Translation**: Legal jargon is converted into easy-to-understand phrases, helping users interpret contracts more effectively.
- **Keyword Emphasis**: The system highlights important clauses, names, and terms to draw attention to critical sections.
- **Section Summaries**: Lengthy sections are reduced into digestible summaries, improving overall document navigation.

Enhanced Accessibility and Language Support

Designed with inclusivity in mind, the platform users from different backgrounds and regions.

- **Multilingual Options**: Legal content can be generated and reviewed in multiple languages, eliminating language barriers.
- **User-Friendly Design**: The interface is intuitive, making it accessible even to those with limited digital or legal experience.
- **Guided Interaction**: The system assists users with prompts and explanations throughout the creation process.

Workflow Integration and Collaboration

The assistant can be embedded into broader systems to support team-based legal work and business operations.

- **Multi-User Collaboration**: Team members can co-edit, comment, and finalize documents collaboratively.
- **Compatibility with Tools**: Integration with software like HR systems or contract management platforms enhances workflow.
- **Custom Workflow Settings**: Users can set document approval steps, notifications, and review processes tailored to their needs.

Efficiency and Cost Optimization

The AI-driven model substantially cuts both the time and expense of legal documentation.

- **Reduced Turnaround Time**: Tasks that previously took days can now be done in a fraction of the time.
- **Decreased Legal Consultation Costs**: Standard legal services can be replaced with automated solutions for routine tasks.
- **Scalability**: The system is suitable for individuals, small teams, and organizations that need to generate high document volumes efficiently.

Discussions

The system developed shows significant potential in changing how legal documentation is approached especially for users lacking legal expertise or access to legal services. By eliminating the need for continuous legal oversight in routine documentation, the platform fosters greater independence and confidence among non-expert users. It bridges the gap between complex legal content and everyday understanding through intuitive design and real-time guidance.

Furthermore, the integration of multilingual support ensures that language is no longer a limiting factor for accessing legal tools, especially in linguistically diverse regions. The assistant empowers individuals to draft, review, and manage legal agreements in a way that is both efficient and legally reliable. This is particularly impactful in underserved communities, where legal aid is limited or unaffordable.

User Empowerment Through Technology

By simplifying legal processes, the assistant allows users to take control of their legal responsibilities without outside help.

- **Legal Confidence**: Users feel more comfortable managing legal tasks independently with AI validation and guidance.
- **Educational Value**: Explanatory prompts help users understand legal terms and document structure during creation.
- **Reduced Legal Risk**: Real-time suggestions and compliance checks minimize the chance of errors or omissions.

Insights from Document Usage

The platform not only helps users generate but also captures patterns and trends in usage.

- **Quality Control**: The assistant ensures that generated documents meet established legal standards by validating clause structure, consistency, and terminology. This reduces the chance of oversight and ensures every document is ready for formal use.
- **Analytics Potential**: The system captures metadata from user behavior, such as frequently selected clauses, modification patterns, and error corrections. This data can be analyzed to refine document templates and suggest process optimizations.

Technology-Driven Legal Evolution

This AI-based solution represents a major shift in how legal services can be delivered, especially in environments where traditional processes are slow, costly, or overly complex. By leveraging artificial intelligence, particularly natural language processing (NLP), the system simplifies legal workflows that would otherwise require extensive human involvement.

- **Natural Language Capabilities**: NLP allows the assistant to analyze, rephrase, and generate legal content in real time. It ensures that documents remain grammatically correct, contextually accurate, and legally sound without human intervention.
- **AI Learning Capabilities**: Over time, the system improves through machine learning, refining its clause recommendations and response accuracy based on user behavior and evolving legal frameworks.
- **Dynamic Legal Suggestions**: Based on input, the system recommends relevant clauses and edits for compliance.

Broader Reach and Future Expansion

The assistant is designed to be scalable and adaptable, with potential applications in various industries, regions, and legal systems. Its architecture allows for easy customization, making it suitable for a wide range of documentation needs beyond the initial use case.

- **Language Inclusivity**: The system continues to expand its multilingual support, making it more accessible to users in non-English-speaking regions. By supporting Indian regional languages and other global languages, the platform ensures that linguistic diversity does not become a barrier to legal engagement.
- **Adaptable Use Cases:** Whether for HR agreements, sales contracts, or freelance engagements, the system can be customized to suit diverse legal scenarios.
- **Future Capabilities**: With further development, the platform may incorporate voice-based input, real-time legal updates, and integrations with government portals for e-filing and contract registration, broadening its scope even further.

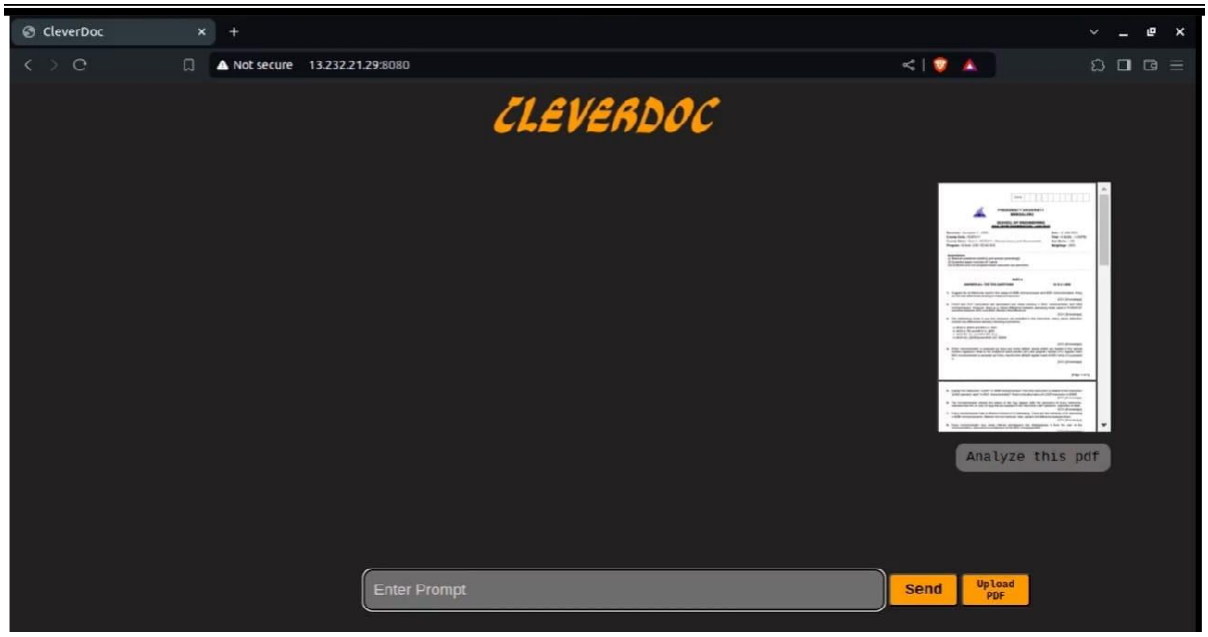

Figure 9.1: Main Page with pdf uploaded

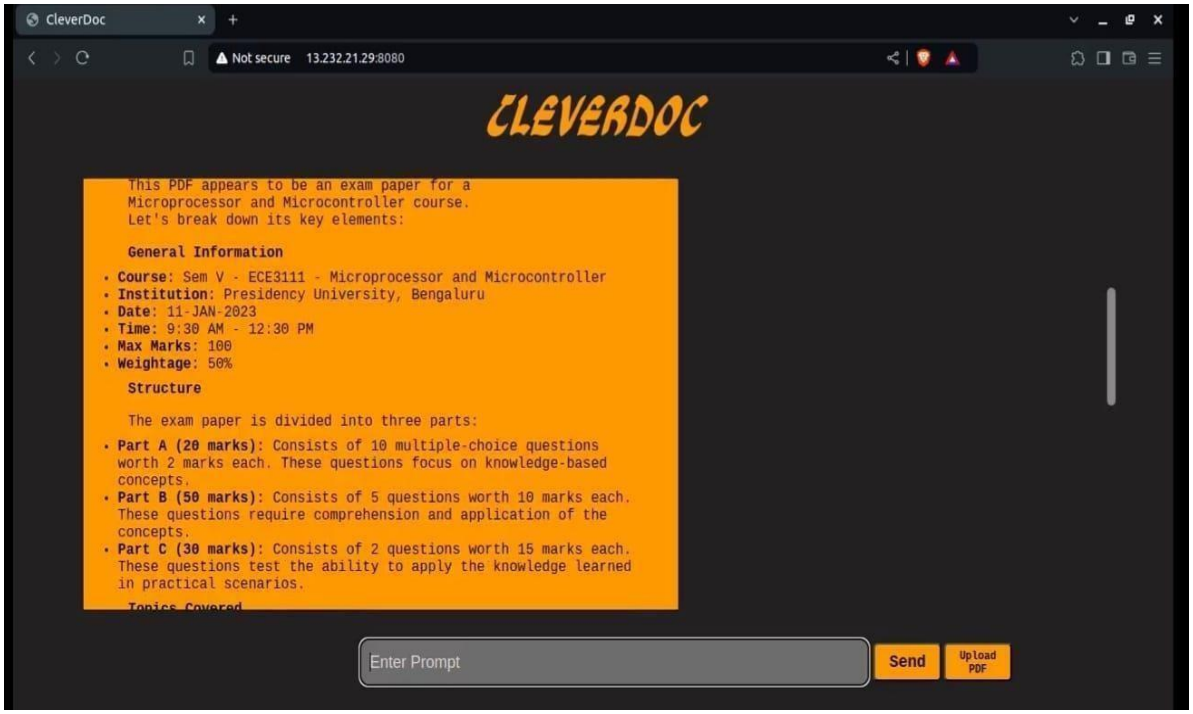

Figure 9.2: Main Page with response

CHAPTER-10
CONCLUSION

The project highlights the effectiveness of using AI to make legal documentation more accessible, faster, and easier to understand. By automating the drafting process and simplifying legal language, the assistant reduces the need for expert intervention in routine tasks. Its user-friendly design and multilingual support make it suitable for a wide range of users. Overall, the system presents a cost-efficient and scalable solution that supports legal self-service and encourages wider legal participation.

REFERENCES

[1] V. Gupta and R. Kumar, "A survey of legal text analysis techniques for Indian legal documents," IEEE Access, vol. 11, pp. 18485-18503, 2023.

[2] A. Arora and A. Soni, "Natural language processing for legal documentation in Indian languages," Springer, 2023.

[3] P. Kumar and V. Bansal, "Natural language processing for automated legal document analysis and contract review," Information Technology Journal, vol. 24, no. 2, pp. 142-158, 2023.

[4] K. Wallace, "Plain language legal writing: Part I – Writing as a process," The National Magazine, 2021.

[5] R. Paul, T. Shah, and A. Patel, "A comprehensive analysis of Indian legal documents," Springer, 2023.

[6] P. Kumar, S. Reddy, and V. Sharma, "Automated techniques on Indian legal documents: A review," IEEE, 2023.

[7] A. Singh and R. Sharma, "Sentiment analysis for legal judgment text in India's Supreme Court," Springer, 2023.

[8] S. K. Singh, A. K. Jain, and P. K. Gupta, "A Two-Staged NLP-Based Framework for Assessing the Sentiments on Indian Legal Documents," SN Applied Sciences, Springer, vol. 5, no. 7, 2023.

[9] M. Sharma and R. Verma, "A Case Study for Automated Attribute Extraction from Legal Documents Using NLP," Artificial Intelligence and Law, Springer, vol. 32, no. 1, 2024.

[10] A. Patel, B. Kumar, and C. Singh, "LAWBOT: A Smart User Indian Legal Chatbot Using Machine Learning Framework," in Proceedings of the 2023 IEEE International Conference on Artificial Intelligence and Machine Learning (AIML), IEEE, 2023

APPENDIX-A

PSUEDOCODE

Backend:

Manage.py

#!/usr/bin/env python

"""Django's command-line utility for administrative tasks."""

import os import sys

def main():

"""Run administrative tasks.""" os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'ridiv.settings') try:

from django.core.management import execute_from_command_line except ImportError as exc:

raise ImportError(

"Couldn't import Django. Are you sure it's installed and "

"available on your PYTHONPATH environment variable? Did you "

"forget to activate a virtual environment?"

) from exc

execute_from_command_line(sys.argv)

if _name___== '_main_':

main()

Settings.py

"""

Django settings for ridiv project.

Generated by 'django-admin startproject' using Django 5.0.

For more information on this file, see https://docs.djangoproject.com/en/5.0/topics/settings/

```
For the full list of settings and their values, see
https://docs.djangoproject.com/en/5.0/ref/settings/
"""
import os from pathlib
import Path

# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent



# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/5.0/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY =
'django-insecure-11#k)0oi=n)broh=)c$c8y5)js&x7yicz^57w1%_=0xt8o)7%v'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []

# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'rest_framework',
    'corsheaders',
```

```
    'app',
]

MIDDLEWARE = [
    'corsheaders.middleware.CorsMiddleware',
   'django.middleware.security.SecurityMiddleware',
   'django.contrib.sessions.middleware.SessionMiddleware',
   'django.middleware.common.CommonMiddleware',
'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'ridiv.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]
WSGI_APPLICATION = 'ridiv.wsgi.application'
```

```
# Database
# https://docs.djangoproject.com/en/5.0/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}



# Password validation
# https://docs.djangoproject.com/en/5.0/ref/settings/#auth-password-validators
AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME':
'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
        'NAME':
'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME':
'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME':
'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]



# Internationalization
```

```
# https://docs.djangoproject.com/en/5.0/topics/i18n/

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_TZ = True

CORS_ALLOW_ALL_ORIGINS = True
CORS_ALLOWED_ORIGINS = [
    'http://127.0.0.1:5173',
    'http://localhost:5173',
]
# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/5.0/howto/static-files/
STATIC_URL = 'static/'
# Define media settings
MEDIA_URL = '/media/'
MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
# Default primary key field type
# https://docs.djangoproject.com/en/5.0/ref/settings/#default-auto-field

DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'
```

Urls.py

```
from django.contrib import admin from
django.urls import path,include from
django.conf import settings from
django.conf.urls.static import static urlpatterns
= [     path('admin/', admin.site.urls),     path('',
include('app.urls')),
```

```
]+ static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

Models.py

```
from django.db import models

class UploadedPDF(models.Model):
    pdf_file = models.FileField(upload_to='uploads/')      uploaded_at =
models.DateTimeField(auto_now_add=True)
```

Utils.py

```
import os import
time

import google.generativeai as genai # /media/uploads/resumedraft.pdf
genai.configure(api_key='AIzaSyDBP8i-K_pmk-1mi168uwTFD1VnlsLzSSg') def
prompt(file,message="Analyse the given pdf",history=[]):     def upload_to_gemini(path,
mime_type=None):
       file = genai.upload_file(path, mime_type=mime_type)
        print(f"Uploaded file '{file.display_name}' as: {file.uri}")
return file      def wait_for_files_active(files):          for name in
(file.name for file in files):
           file = genai.get_file(name)              while
file.state.name == "PROCESSING":
               print(".", end="", flush=True)
time.sleep(10)                 file =
genai.get_file(name)               if file.state.name
!= "ACTIVE":
               raise Exception(f"File {file.name} failed to process")
print("...all files ready")          print()
    generation_config = {
    "temperature": 1,
```

```
    "top_p": 0.95,
    "top_k": 64,
    "max_output_tokens": 8192,
    "response_mime_type": "text/plain",
    }
    model = genai.GenerativeModel(     model_name="gemini-1.5-flash",
generation_config=generation_config,
    )
    # TODO Make these files available on the local file system
    # You may need to update the file paths     current_directory =
os.path.dirname(os.path.abspath(_file_))        backend_directory =
os.path.dirname(current_directory)      file = backend_directory+file      files =
[
    upload_to_gemini(file, mime_type="application/pdf"),
    ]
    # Some files have a processing delay. Wait for them to be ready.
    wait_for_files_active(files)      history=[
          {
             "role": "user",
             "parts": [
 files[0],
            ],
          },
       ]
    history=history+history      chat_session =
model.start_chat(          history=history
    )
    response = chat_session.send_message(message)      return
response.text
```

Views.py

```
from django.shortcuts import get_object_or_404 from
django.http import JsonResponse from
```

```
django.views.decorators.csrf import csrf_exempt from .models
import UploadedPDF from .serializers import
UploadedPDFSerializer from .utils import prompt import json
@csrf_exempt def
upload_pdf(request):
    if request.method == 'POST':
        serializer = UploadedPDFSerializer(data=request.FILES)         if
serializer.is_valid():              serializer.save()              return
JsonResponse(serializer.data, status=201)          return
JsonResponse(serializer.errors, status=400)     return JsonResponse({'error': 'POST
method required'}, status=400)

@csrf_exempt
def retrieve_pdf(request, pk):
    pdf = get_object_or_404(UploadedPDF, pk=pk)     if
request.method == 'POST':
        serializer = UploadedPDFSerializer(pdf)          data = json.loads(request.body)          data
= prompt(message=data.get('message'),file=serializer.data['pdf_file'],history=dat
a.get('history'))          return JsonResponse({"response":data}, status=200)
     return JsonResponse({'error': 'GET method required'}, status=400) @csrf_exempt def
   delete_pdf(request, pk):
    pdf = get_object_or_404(UploadedPDF, pk=pk)     if request.method ==
'DELETE':          pdf.delete()          return JsonResponse({'message': 'PDF deleted
successfully'}, status=204)

    return JsonResponse({'error': 'DELETE method required'}, status=400)
```

App/urls.py

```
from django.urls import path from .views import upload_pdf,
retrieve_pdf, delete_pdf

urlpatterns = [      path('pdf/', upload_pdf, name='pdf-upload'),
path('pdf/<int:pk>/', retrieve_pdf, name='pdf-retrieve'),
```

```
path('pdf/<int:pk>/delete/', delete_pdf, name='pdf-delete'),
]
```

Frontend:

Main.tsx

```
import ReactDOM from "react-dom/client";
import App from "./App.tsx"; import
"./index.css";

ReactDOM.createRoot(document.getElementById("root")!).render(<App  />);
```

App.tsx

```
import React from "react"; import "./App.css";
import Home from "./components/Home"; const
App: React.FC = () => {  return (
   <>
     <Home />
   </>
 );
};

export default App;
```

Home.tsx

```
import React, { useState, useRef, useEffect } from "react"; import
"./Home.css"; import axiosInstance from "../axiosInstance"; import
ReactMarkdown from "react-markdown"; import remarkGfm from
"remark-gfm"; interface Message {  type: "text" | "pdf" | "response";
```

```
content: string;
}
interface UploadResponse {
  id: string;  pdf_file:
string;
}
interface MessageResponse {  response: string;
}
interface HistoryEntry {
role: string;  parts:
string[];
}
const Home: React.FC = () => {   const [messages, setMessages] =
useState<Message[]>([]);  const [id, setId] = useState<string>("");
const fileInputRef = useRef<HTMLInputElement>(null);   const
inputRef = useRef<HTMLInputElement>(null);   const chatRef =
useRef<HTMLDivElement>(null);  const idRef = useRef<string>(id);
useEffect(() => {      idRef.current = id;
  }, [id]);
  const handleUpload = () => {      if
(fileInputRef.current) {
fileInputRef.current.click();
    }
  };
  const uploadPDF = async (file: File): Promise<UploadResponse> => {      try {
     const formData = new FormData();       formData.append("pdf_file", file);
const response = await axiosInstance.post<UploadResponse>(
       "/pdf/",
formData,
       {
         headers: {
           "Content-Type":  "multipart/form-data",
         },
       }
```

```
    );
    return response.data;    } catch (error) {
console.error("Error uploading PDF:", error);      throw new
Error(`Error uploading PDF: ${error}`);
   }
  };
  const sendMessage = async (
    msg: string,
 history: HistoryEntry[]
 ): Promise<MessageResponse> => {    try {
const payload = {        message: msg,
history: history,
    };
    const response = await axiosInstance.post<MessageResponse>(
      `/pdf/${id}/`,
payload,
      {
        headers: {
          "Content-Type": "application/json",
        },
      }
    );
    return response.data;    }
catch (error) {
    console.error("Error uploading PDF:", error);      throw new
Error(`Error uploading PDF: ${error}`);
   }
  };
  const handleFileSelect = async (     event:
React.ChangeEvent<HTMLInputElement>
  ) => {     const file =
event.target.files?.[0];
   if (file) {
try {
```

```
      const result = await uploadPDF(file);
setId(result.id);       } catch (error) {
console.error("Upload failed:", error);
     }
     processPDF(file);
   }
  };
  const processPDF = (file: File) => {
    const fileReader = new FileReader(); fileReader.onload = (e) =>
    {
      if (e.target?.result && typeof e.target.result === "string") {
    const pdfMessage: Message = {        type:
"pdf",
       content: e.target.result,
      };
       setMessages((prevMessages) => [...prevMessages, pdfMessage]);         if
(chatRef.current) {          chatRef.current.scrollTop = chatRef.current.scrollHeight;
chatRef.current.scrollIntoView({             behavior: "smooth",            block: "end",
inline: "nearest",
        });
       }
     }
   };
   fileReader.readAsDataURL(file);
  };

  const handleMessageSubmit = (event: React.FormEvent) => {
event.preventDefault();      if (inputRef.current &&
inputRef.current.value.trim() !== "") {       const newMessage: Message = {
      type: "text",        content:
inputRef.current.value.trim(),
     };
     setMessages((prevMessages) => [...prevMessages, newMessage]);
inputRef.current.value = "";       if (chatRef.current) {         chatRef.current.scrollTop
```

```
= chatRef.current.scrollHeight;
    }
   }
  };
  const prephistory = (msgs: Message[]): HistoryEntry[] => {    const
history: HistoryEntry[] = [];    msgs.map((msg) => {      if (msg.type
=== "text") {        history.push({ role: "user", parts: [msg.content] });
    } else if (msg.type === "response") {  history.push({ role:
    "model", parts: [msg.content] });
    }
 });
 return history;
 };
 //use effect- everytime a message is added to the array if the type is text send the
request, if the type  is response serve it  useEffect(() => {    const handleMessage =
async () => {      if (messages.length > 0) {        const lastMessage =
messages[messages.length - 1];        if (lastMessage.type === "text") {
       // Prepare history          const history: HistoryEntry[] =
prephistory(messages);          history.pop(); // Remove the last message as it was
just added          try {
          const res: MessageResponse = await sendMessage(
lastMessage.content,              history
          );
          const msg: Message = {
type: "response",
            content: String(res.response),
          };
          setMessages((prevMessages) => [...prevMessages, msg]);
        } catch (error) {            const msg: Message
= {              type: "response",              content:
"Failed to send message",
          };
          setMessages((prevMessages) => [...prevMessages, msg]);
        }
```

```
      }
    }
  };

  handleMessage(); },
[messages]); useEffect(()
=> {
   const handleBeforeUnload = async () => { // Perform the request to
     your server await
     axiosInstance.delete(`/pdf/${idRef.current}/delete/`); };
     window.addEventListener("beforeunload", handleBeforeUnload);
     return () => {     window.removeEventListener("beforeunload",
     handleBeforeUnload);
  };
 }, []);
 return (
  <div className="container">
   <header>
    <h1 className="head">Legal Document Analysis</h1>
   </header>
   <main className="parent">
    <input          type="file"
ref={fileInputRef} style={{ display:
"none" }}
onChange={handleFileSelect}
accept=".pdf"
     />
     <section className="chat" ref={chatRef}>
      {messages.map((message, index) => (
       <div key={index} className={`message ${message.type}`}>
        {message.type === "text" ? (
         <p>{message.content}</p>
        ) : message.type === "pdf" ? (
         <embed
```

```
src={message.content}
type="application/pdf"
width="200px"                height="300px"
            />
          ) : message.type === "response" ? (
            <div className="response">
              <ReactMarkdown remarkPlugins={[remarkGfm]}>
                {message.content}
              </ReactMarkdown>
</div>
          ) : (
            <p>Unsupported message type</p>
          )}
        </div>
      ))}
      </section>
      <button className="but" onClick={handleUpload}>
        Upload PDF
       </button>
       <form onSubmit={handleMessageSubmit} className="inp">
        <input type="text" ref={inputRef} placeholder="Enter Prompt" />
        <button type="submit" className="send">
          Send
        </button>
       </form>
     </main>
   </div>
  );
};

export default Home;
```

APPENDIX-B

SCREENSHOTS

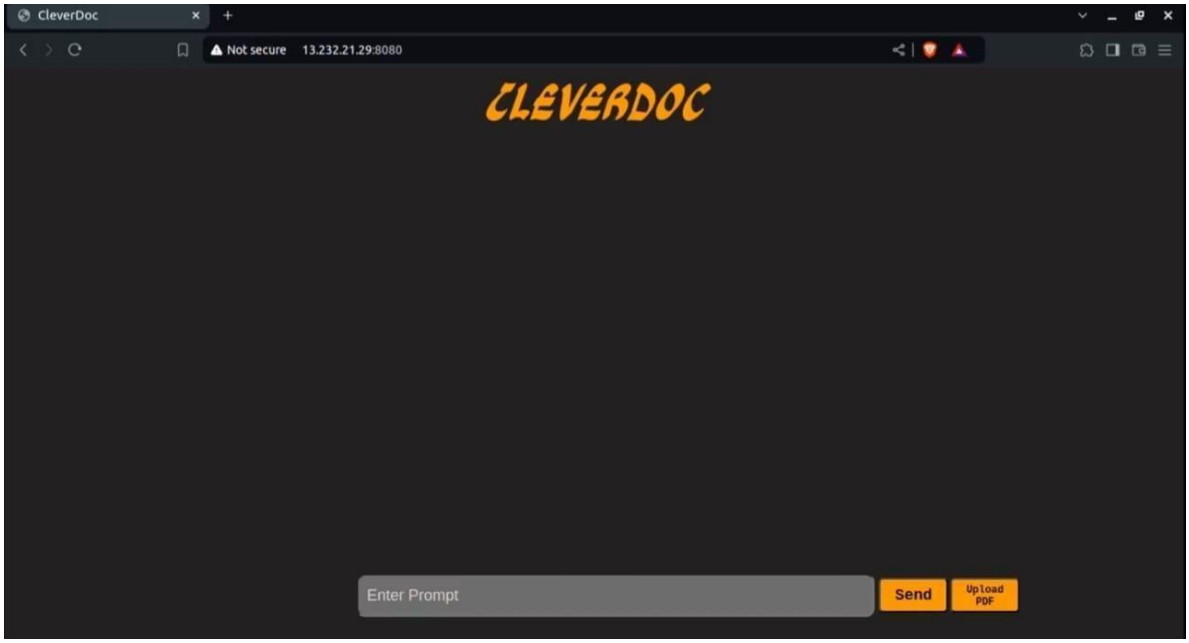

Figure 1: Main Page

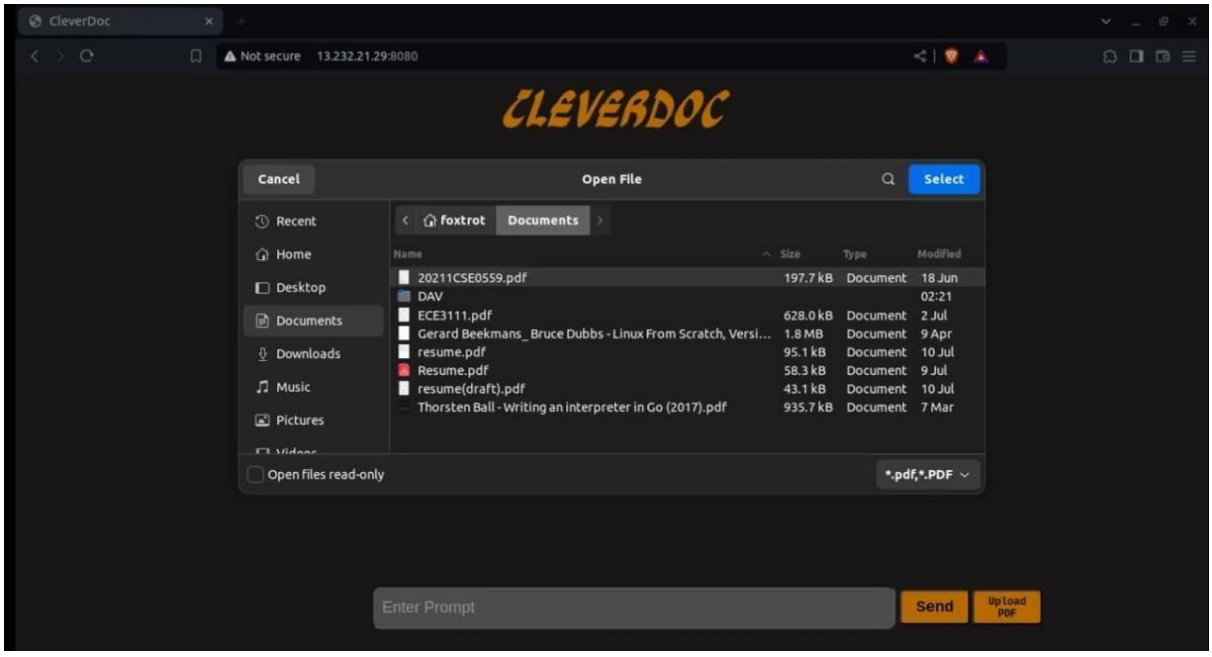

Figure 2: Main Page while uploading pdf

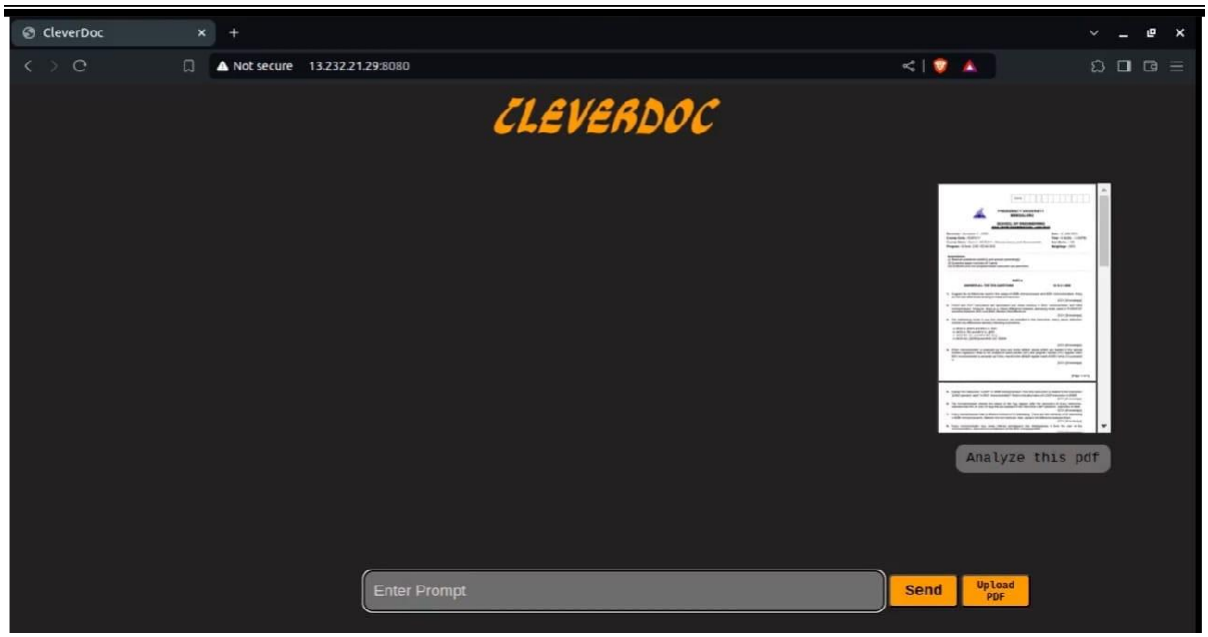

Figure 3: Pdf uploaded

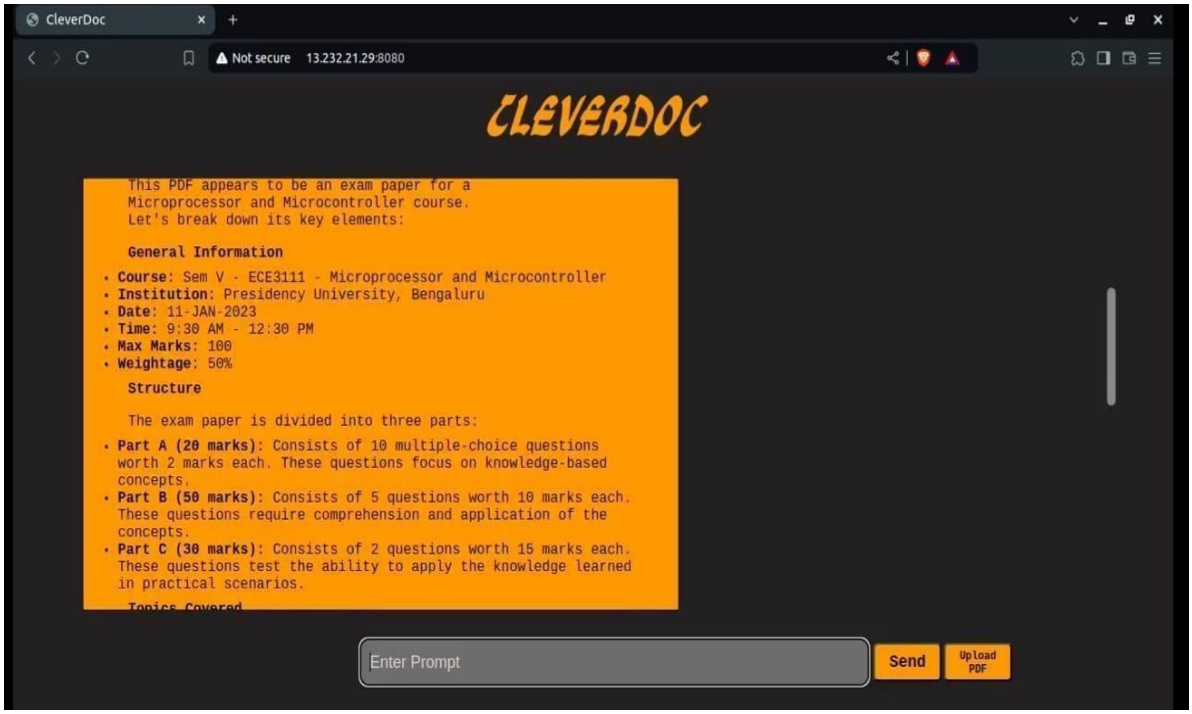

Figure 4: Pdf analysed

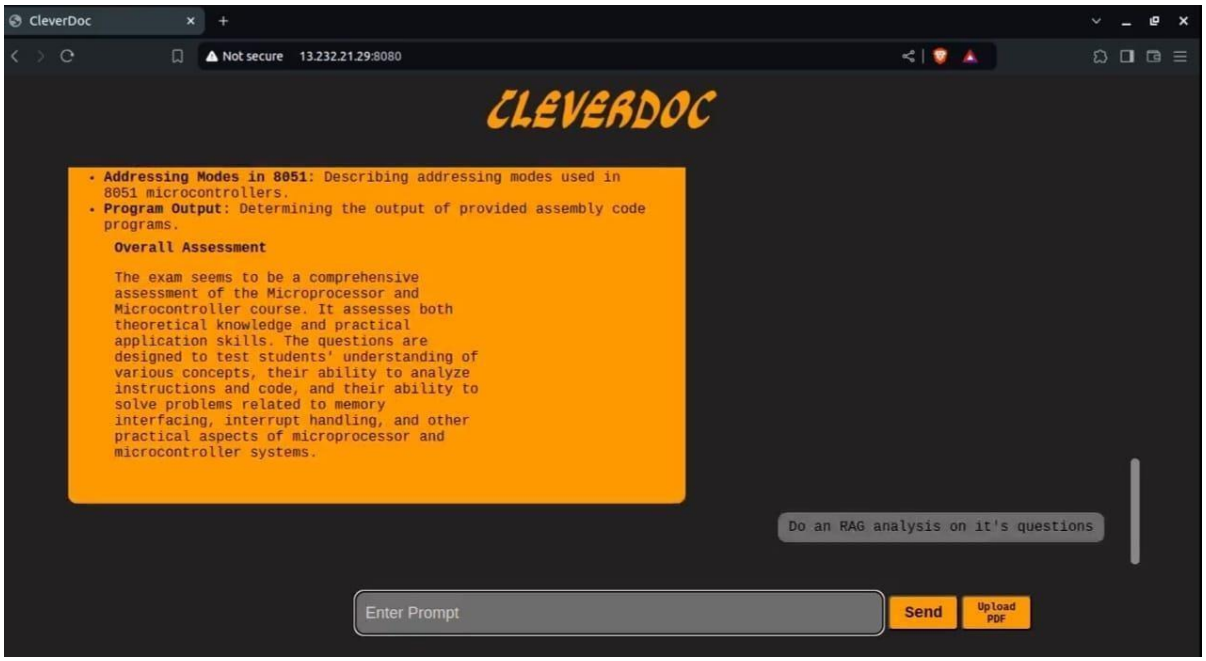

Figure 5: Query Given

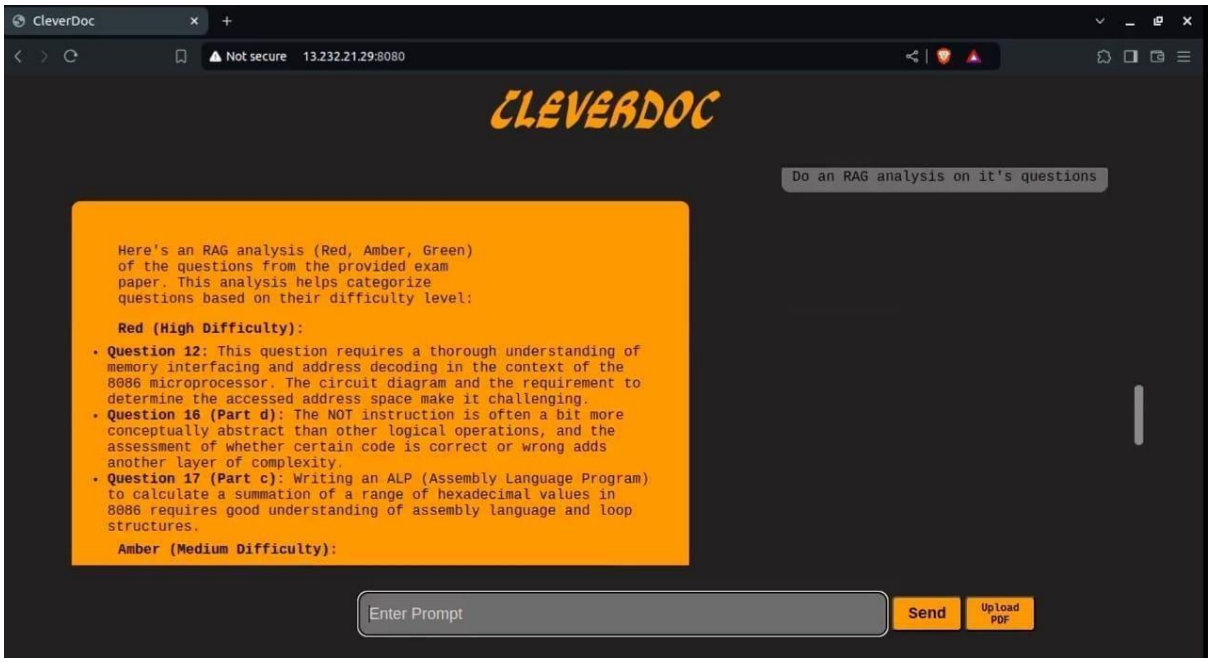

Figure 6: Response for the Query

APPENDIX-C

ENCLOSURES

Publication Certificate

Figure 7: Certificate

Figure 8: Certificate 2

Figure 9: Certificate 3

Figure 10: Certificate 4

Figure 11: Certificate 5

SIMILARITY INDEX

Figure 12: Plagiarism Check Report

SUSTAINABLE DEVELOPMENT GOALS

This project aligns with SDG 16: Peace, Justice and Strong Institutions

Figure 13: SDG

By promoting accessible, transparent, and efficient legal support through AI-driven automation. By simplifying complex legal language and providing tools for clause analysis, summarization, and template generation, the assistant empowers individuals and small organizations to engage with legal processes confidently. It enhances legal literacy, reduces reliance on costly legal services, and improves institutional transparency. In doing so, the system contributes to more inclusive justice systems and helps strengthen accountable, rights-based institutions.