Volume: 13 Issue: VII Month of publication: July 2025
DOI: https://doi.org/10.22214/ijraset.2025.73425

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue VII July 2025- Available at www.ijraset.com

AI Legal Insight Analyser (ALIA)

Mugi Satish[1], Kappala Swarna Madhuri[2]
Assistant Professor, MCA Final Semester, Master of Computer Applications, Sanketika Vidya Parishad Engineering College, Vishakhapatnam, Andhra Pradesh, India.

Abstract: ***The AI Legal Insight Analyzer (ALIA) is a smart web application designed to make legal document analysis faster, easier, and more accurate. By combining artificial intelligence (AI) with natural language processing (NLP), ALIA helps legal professionals, researchers, and students efficiently extract key information from legal documents like court judgments. Built using Flask in Python, ALIA allows users to upload PDF-based legal documents and automatically pulls out critical details such as case headings, court names, judges, citations, and relevant legal sections. With the power of Google Gemini, it provides concise summaries, answers legal questions in real time, and even explains selected text with a simple click. The application ensures versatility by using PyMuPDF and Tesseract OCR to process both standard and scanned PDFs. A clean and responsive Tailwind CSS-styled interface, combined with custom JavaScript enhancements, offers a smooth experience—from the initial upload to an interactive dashboard where users can explore the extracted content. ALIA is designed to address the common challenges of legal research, such as time-consuming manual analysis, complexity, and human error. By streamlining document processing, it empowers users with instant insights and makes legal research more accessible. As AI continues to evolve, ALIA has the potential to expand further, bringing even more innovation to the legal domain.***

I. INTRODUCTION

In the legal domain, professionals often grapple with extensive documentation, complex language, and time-consuming research. Legal documents, such as court judgments and case laws, are typically lengthy, filled with formal jargon, and structured in ways that require deep domain expertise to interpret effectively [3]. This traditional process of manual reading, annotating, and extracting relevant legal information is not only tedious but also prone to human error, leading to inefficiencies and potential oversight of critical details.

To overcome these limitations, we present ALIA (AI Legal Insight Analyzer) an intelligent web application designed to automate and enhance legal document analysis [6]. ALIA integrates Artificial Intelligence (AI) with Natural Language Processing (NLP) to perform tasks such as text extraction, OCR processing for scanned documents, case summarization, and answering legal queries [7]. By leveraging technologies like PyMuPDF, Tesseract, and Google's Gemini LLM, the system can intelligently interpret legal content, recognize patterns, and deliver concise summaries and contextual information in real time [10]. This drastically reduces the effort required in reviewing court documents while improving the accuracy of legal research.

Built using Python and the Flask framework, ALIA features a user-friendly interface powered by Tailwind CSS, making it accessible to legal professionals, law students, and researchers alike [5]. Users can upload legal PDFs—scanned or typed—and receive an interactive analysis including case headings, court names, citations, sections, judges, and AI-generated conclusions [3]. ALIA not only boosts productivity but also democratizes access to legal analysis tools, making legal research more approachable and efficient in both academic and professional settings.

A. Existing System

Traditional legal document analysis relies heavily on manual reading, interpretation, and note-taking by legal professionals [4]. Advocates, clerks, and researchers spend hours sifting through lengthy court judgments, case laws, and legal briefs to extract critical information such as case headings, court names, bench compositions, citations, and relevant statutory sections [2]. This process not only consumes valuable time but also requires extensive legal knowledge to ensure accuracy and proper contextual understanding [7]. Moreover, most legal documents are unstructured or formatted inconsistently, especially scanned copies of older judgments that are not text-searchable. In such cases, manual transcription or OCR tools must be used separately, adding further complexity [20]. The absence of automated tools to summarize content or highlight key insights limits productivity and increases the chance of overlooking important legal precedents. As a result, legal research becomes cumbersome, especially for junior professionals and students who lack advanced legal training [16].

1) *Challenges*

The existing manual approach to legal document analysis poses several significant challenges. Firstly, the process is extremely time-consuming, as professionals must read through lengthy judgments to extract relevant information [11]. This not only slows down legal research but also limits the number of cases that can be reviewed in a given time frame. Additionally, there is a high risk of overlooking critical legal references, such as precedent citations or statutory provisions, which may impact the outcome of legal arguments [15]. The absence of automated summarization tools makes it difficult to grasp the contextual essence of a case quickly, often requiring deep reading and cross-referencing. Furthermore, traditional systems lack the capability for real-time question answering or contextual explanation, forcing users to manually interpret complex legal language [14]. An added complication arises with scanned legal documents, where standard text extraction tools fail, making analysis even more difficult without advanced OCR support [21].

B. *Proposed System*

ALIA (AI Legal Insight Analyzer) is an intelligent, web-based system that transforms the traditional approach to legal document analysis by leveraging Artificial Intelligence and Natural Language Processing techniques [2]. The application is built using Python's Flask framework, integrating PyMuPDF for PDF parsing, Tesseract for OCR processing of scanned documents, and Google Gemini LLM for AI-powered interaction. Once a user uploads a digital or scanned legal PDF, ALIA extracts the raw text content, processes it to identify key elements like case headings, citations, court names, judges, and statutory sections, and generates a concise summary [15]. The system further empowers users to ask context-aware legal questions or select specific text segments to receive real-time explanations powered by Gemini. With a modern Tailwind CSS interface, ALIA ensures accessibility, speed, and interactivity in legal research workflows [17].

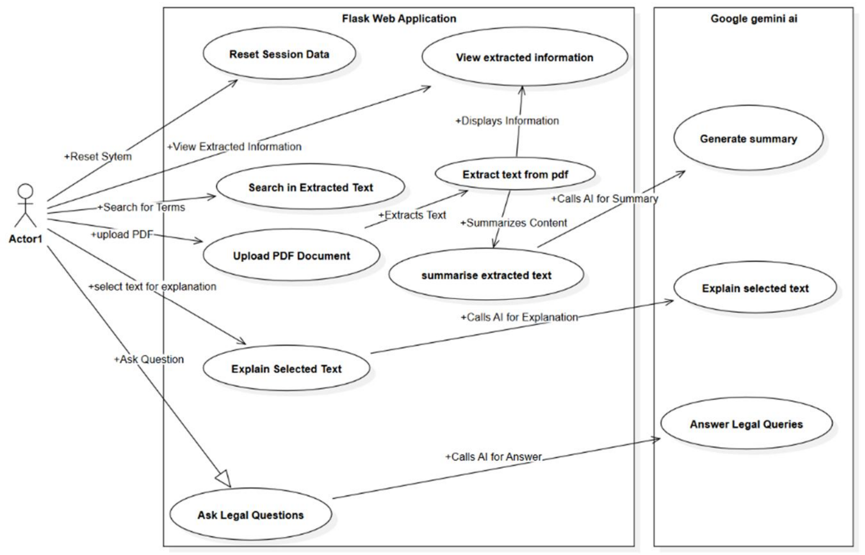


Fig. 1 Ai Legal Insight Analyser (Alia) Flow Chart.

1) *Advantages*

a) Automated Legal Information Extraction: ALIA intelligently detects and extracts case headings, citations, court names, and legal sections from uploaded PDFs, reducing manual effort.
b) OCR-Enabled Scanned Document Support: With integrated Tesseract OCR, ALIA can accurately process scanned judgments and handwritten or image-based legal documents.
c) AI-Powered Summarization: The system uses NLP techniques to generate concise summaries of long legal texts, helping users quickly understand case outcomes.
d) Interactive Legal Q&A and Explanation: Users can ask real-time questions or click on any part of the text for simplified explanations, powered by Google Gemini.
e) Enhanced User Interface and Productivity: A clean, responsive UI built with Tailwind CSS ensures ease of use, speeding up legal research for students, lawyers, and researchers.

II. LITERATURE REVIEW

Recent advancements in legal informatics have seen the integration of Natural Language Processing (NLP) and machine learning techniques to automate aspects of legal document analysis [5]. Various systems have attempted to extract structured data from unstructured legal texts; however, most of these approaches are limited to static extraction without offering interactivity or deep contextual understanding [6]. For instance, studies such as Legal Text Summarization (IEEE, 2022) focus on compressing legal content for easier reading, while others like OCR and NLP Hybrid Approaches (Elsevier, 2021) explore methods to process scanned documents effectively. Research on Large Language Models (LLMs) in Legal Contexts (Springer, 2023) highlights the growing potential of AI in understanding and generating legal narratives [4]. Despite these contributions, there remains a gap in systems that offer real-time question answering and explainability. ALIA aims to bridge this gap by combining robust document processing with interactive AI through the integration of Google Gemini [9].

A. Architecture

1) User Interface Layer: This layer handles user interactions and is built using Flask templates styled with Tailwind CSS [7]. It allows users to upload documents, view results, and interact with the system seamlessly.
2) Processing Layer: Responsible for extracting text using PyMuPDF, performing OCR with Tesseract for scanned PDFs, and generating summaries through the summa library [4].
3) AI Integration Layer: Powered by Google Gemini, this layer generates judgment conclusions, answers user queries in real time, and provides simplified explanations of selected legal text.
4) Data Handling & Storage Layer: Flask's session management system temporarily stores user data, extracted content, and results during the interaction for a smooth and stateful user experience [14].

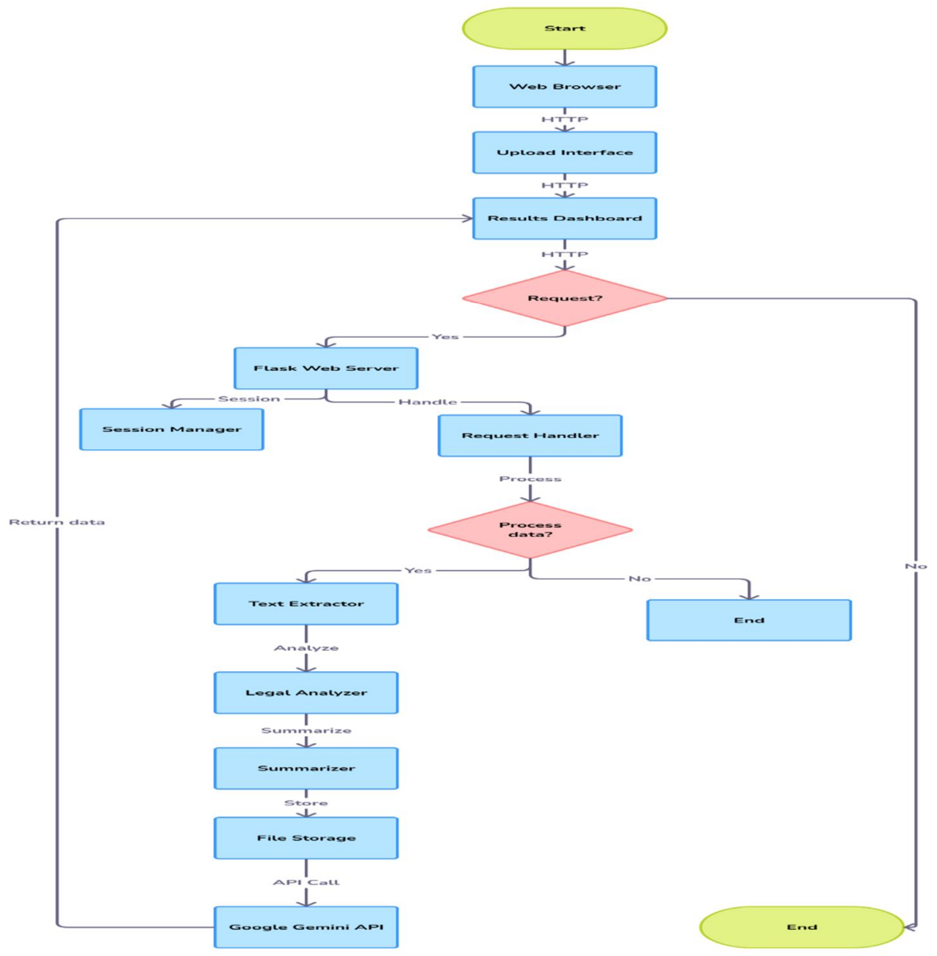


Fig. 2 System Architecture of Ai Legal Insight Analyser(Alia).

B. Algorithm

The core functionality of ALIA is driven by three key algorithms: text extraction, summarization, and query answering. The text extraction algorithm begins by opening the uploaded PDF using PyMuPDF to extract embedded text [5]. In cases where pages contain scanned images or non-selectable content, the system falls back to Tesseract OCR to accurately retrieve the text. The extracted text from each page is then concatenated into a single document for processing. Next, the summarization algorithm uses the summa [11].

Summarizer library to generate a condensed version of the document, selecting the top N most relevant sentences to provide a quick understanding of the content [18]. Finally, the query answering algorithm integrates Google Gemini, where user questions are prompted alongside the extracted document content. Gemini processes this input and returns an AI-generated response that is contextually accurate and legally meaningful, thereby enabling real-time interaction with complex legal material [14].

C. Techniques

ALIA employs a combination of Natural Language Processing (NLP), Optical Character Recognition (OCR), and prompt-based interaction with large language models to achieve efficient legal document analysis [21]. Text extraction from digital PDFs is handled using PyMuPDF, while scanned or image-based documents are processed using Tesseract OCR to retrieve readable text. Regular expressions are used to identify and extract specific legal elements such as citations, court names, and statutory references [19]. For summarization, extractive techniques are applied using the summa. summarizer library to generate concise overviews. Additionally, prompt engineering is used to interact with Google Gemini, enabling contextual understanding, real-time question answering, and simplified explanations of complex legal language [15].

Together, these techniques create a robust pipeline that transforms static legal texts into dynamic, searchable, and interactive formats.

D. Tools

1) Programming Language: The application is developed using Python 3.10+, chosen for its rich ecosystem, AI libraries, and ease of integration with web frameworks and APIs.
2) Web Framework: Flask is used as the lightweight web framework that handles routing, request processing, session management, and rendering of dynamic HTML pages [16].
3) Document Processing: PyMuPDF is utilized for parsing digital PDFs to extract embedded text, while Tesseract OCR processes scanned or image-based legal documents for text recognition [11].
4) Artificial Intelligence Integration: Google Gemini API powers the AI features, including summarization, real-time question answering, and explanation of legal content using large language models [17].
5) Text Summarization: The **summa** library provides extractive summarization by identifying key sentences from the input text, enabling users to quickly understand large judgments [13].
6) Frontend and Image Handling: Tailwind CSS is used for creating a responsive and modern UI, and PIL (Python Imaging Library) assists in converting PDF images into formats suitable for OCR [18].

E. Methods

The ALIA system follows a modular and structured methodology to process legal documents efficiently. The process begins with a user uploading a legal PDF through the web interface. Flask handles the file upload and uses session management to store temporary data, such as extracted text and analysis results, throughout the user's interaction [18]. Once uploaded, the document undergoes pre-processing. If it is a text-based PDF, PyMuPDF is used for direct extraction. If it's a scanned or image-based document, the system switches to Tesseract OCR to convert visual content into machine-readable text [17]. After pre-processing, the extracted content is analyzed using regular expressions to identify and extract legal metadata such as citations, court names, judges, and statutory references [2].

This structured data is then passed into the AI pipeline. Using Gemini's large language model, the system generates a case summary, provides a conclusion, answers user queries, and explains selected text [16]. The output is dynamically rendered on the results page using Jinja2 templates, which integrate the processed data into a clean, interactive UI built with Tailwind CSS. This multi-step method ensures that ALIA delivers both accuracy and usability in legal document analysis [11].

III. METHODOLOGY

A. Input

The primary input to the ALIA system is legal PDF documents, which may be either digitally generated or scanned. These documents typically include court judgments, case summaries, legal notices, or legal research papers [15]. Digital PDFs contain embedded text that can be directly extracted, while scanned PDFs consist of image-based content requiring OCR processing. Regardless of format, the uploaded PDF serves as the foundational data source from which ALIA extracts, analyzes, and interprets legal information for further processing [18]. The system is designed to handle varying layouts, fonts, and structures commonly found in legal documentation, making it adaptable to diverse case formats.

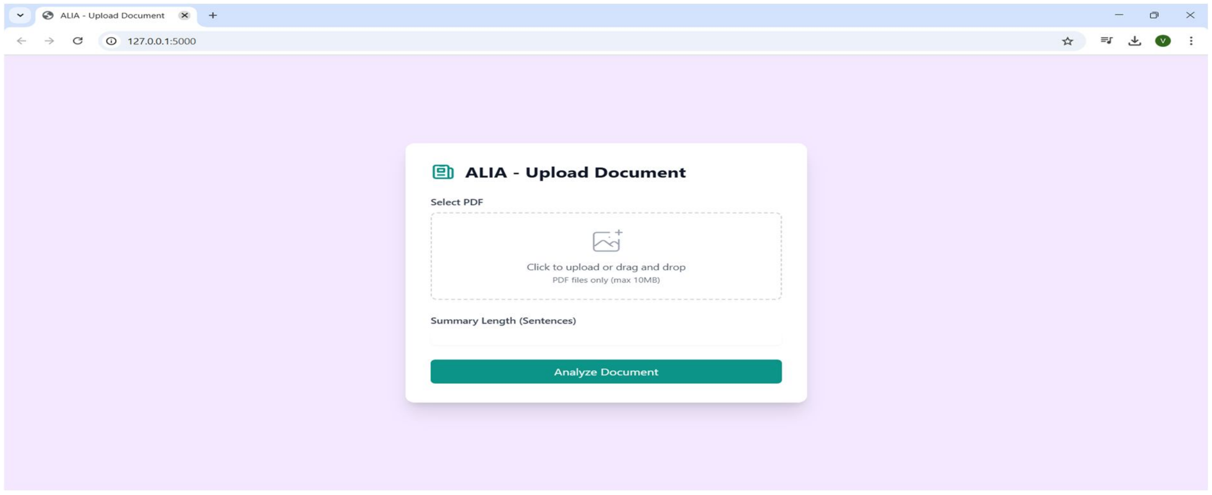
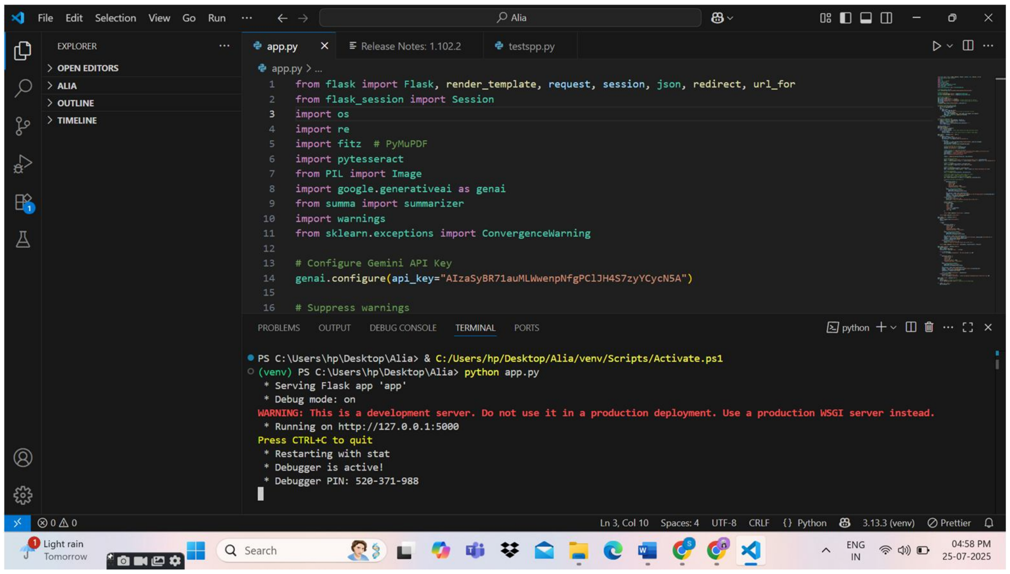


Fig. 3 Ai Legal Insight Analyser (Alia) input screen.

B. Method Of Process

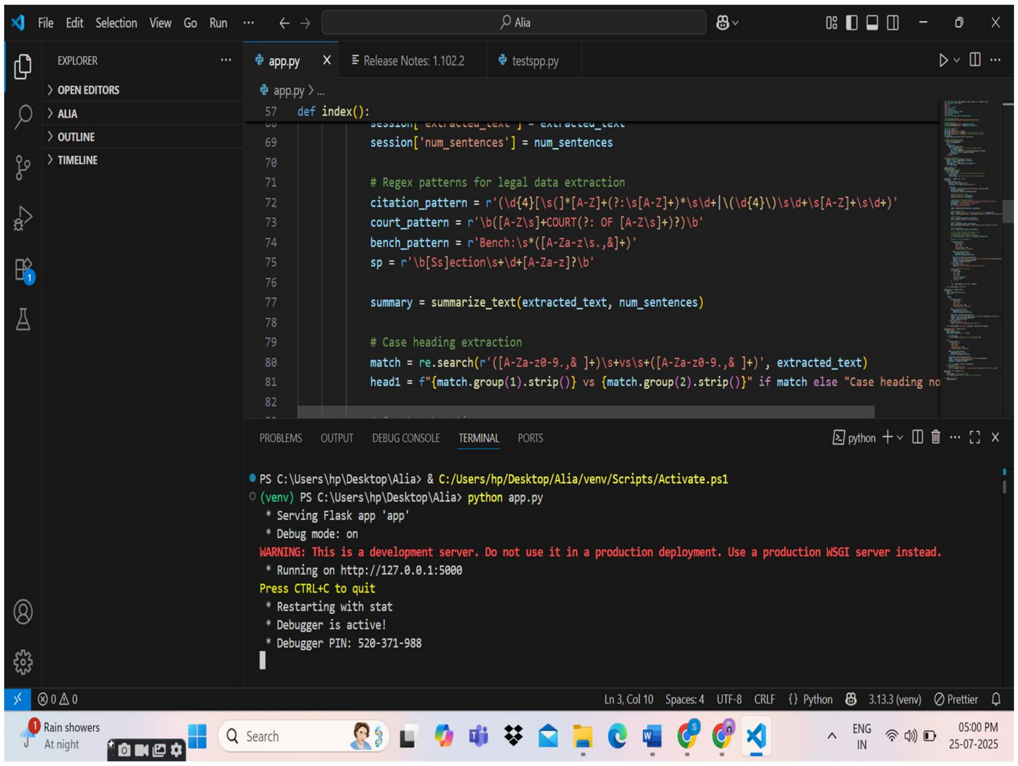
Once a legal PDF document is uploaded, ALIA initiates a multi-step processing pipeline. The system first attempts to extract text using PyMuPDF for digitally generated PDFs. If the pages lack embedded text, it automatically falls back to Tesseract OCR to handle scanned or image-based content [21]. After successful extraction, regular expressions are applied to identify and extract crucial legal data such as case headings, court names, citations, judges, and relevant legal sections [19]. This raw and structured data is then processed using Google Gemini, which generates concise summaries, answers user queries, and explains selected content [15]. Throughout the entire session, Flask's session management securely stores the extracted and generated data, ensuring that the user interface can dynamically display results without reprocessing [14].



```python
from flask import Flask, render_template, request, session, json, redirect, url_for
from flask_session import Session
import os
import re
import fitz  # PyMuPDF
import pytesseract
from PIL import Image
import google.generativeai as genai
from summa import summarizer
import warnings
from sklearn.exceptions import ConvergenceWarning

# Configure Gemini API Key
genai.configure(api_key="AIzaSyBR71auMLWwenpNfgPClJH4S7zyYCycN5A")

# Suppress warnings
```

```
PS C:\Users\hp\Desktop\Alia> & C:/Users/hp/Desktop/Alia/venv/Scripts/Activate.ps1
(venv) PS C:\Users\hp\Desktop\Alia> python app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 520-371-988
```

Fig. 4 Preprocess data & Perform EDA



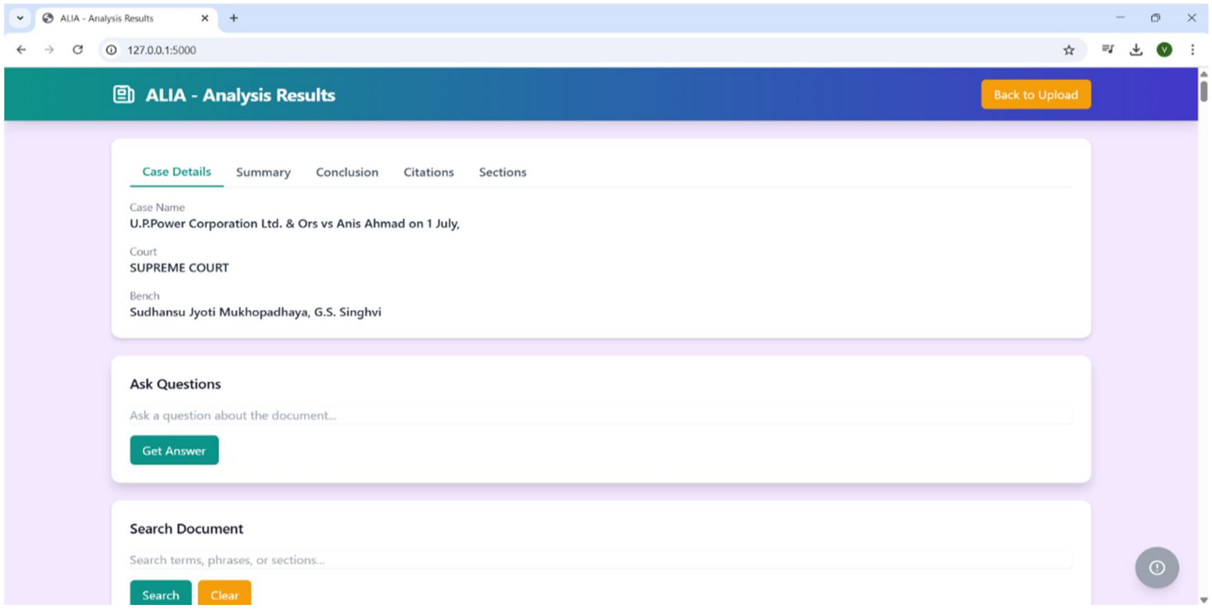
```python
def index():
        session['extracted_text'] = extracted_text
        session['num_sentences'] = num_sentences

        # Regex patterns for legal data extraction
        citation_pattern = r'(\d{4}[\s(]*[A-Z]+(?:\s[A-Z]+)*\s\d+|\(\d{4}\)\s\d+\s[A-Z]+\s\d+)'
        court_pattern = r'\b([A-Z\s]+COURT(?: OF [A-Z\s]+)?)\b'
        bench_pattern = r'Bench:\s*([A-Za-z\s.,&]+)'
        sp = r'\b[Ss]ection\s+\d+[A-Za-z]?\b'

        summary = summarize_text(extracted_text, num_sentences)

        # Case heading extraction
        match = re.search(r'([A-Za-z0-9.,& ]+)\s+vs\s+([A-Za-z0-9.,& ]+)', extracted_text)
        head1 = f"{match.group(1).strip()} vs {match.group(2).strip()}" if match else "Case heading no
```

```
PS C:\Users\hp\Desktop\Alia> & C:/Users/hp/Desktop/Alia/venv/Scripts/Activate.ps1
(venv) PS C:\Users\hp\Desktop\Alia> python app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 520-371-988
```

Fig. 5 Evaluate performance

C. Output

The output generated by ALIA is comprehensive and user-friendly, designed to assist in quick legal analysis. It includes a summarized version of the entire document, highlighting the core aspects of the judgment [11]. Additionally, the system extracts and displays key legal components such as citations, court names, judge information, and statutory sections using pattern recognition techniques [2]. A conclusion generated via Google Gemini provides a concise legal interpretation of the case. Users can also interact with the system by posing legal questions and receiving context-aware AI responses, as well as selecting specific portions of text to receive simplified explanations, making complex legal content more understandable [5].



ALIA - Analysis Results

Case Details | Summary | Conclusion | Citations | Sections

Case Name
U.P.Power Corporation Ltd. & Ors vs Anis Ahmad on 1 July,

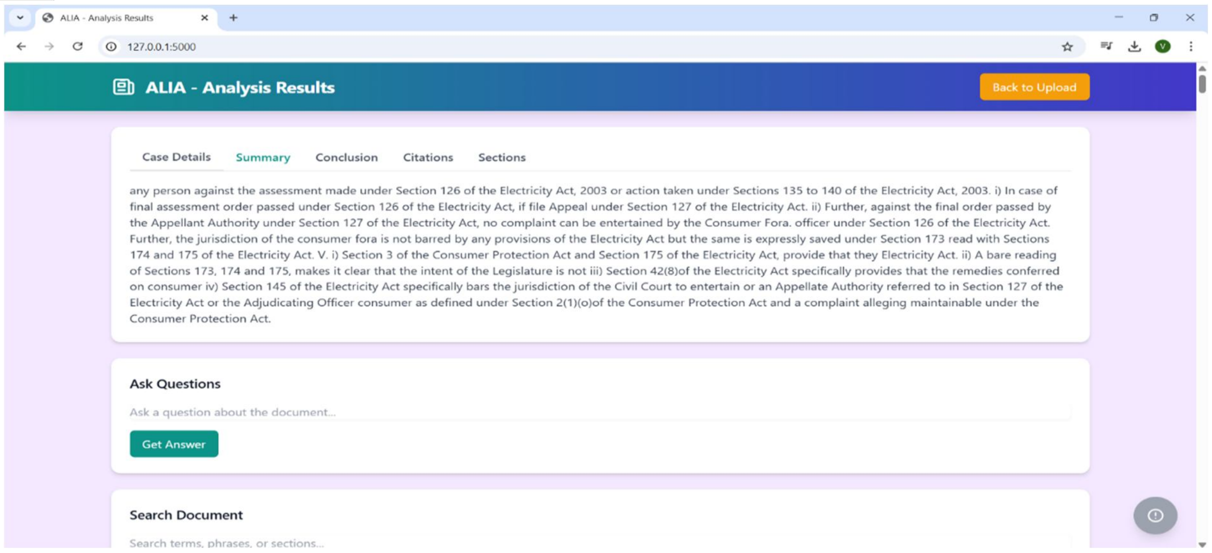
Court
SUPREME COURT

Bench
Sudhansu Jyoti Mukhopadhaya, G.S. Singhvi

Ask Questions
Ask a question about the document...

Search Document
Search terms, phrases, or sections...

Fig. 6 Output Screen

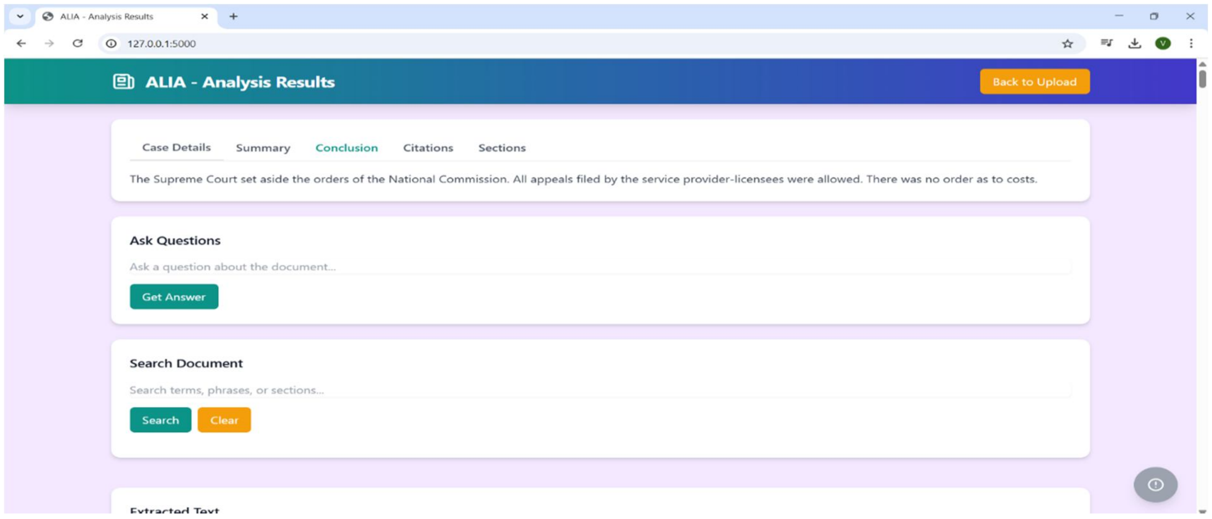


any person against the assessment made under Section 126 of the Electricity Act, 2003 or action taken under Sections 135 to 140 of the Electricity Act, 2003. i) In case of final assessment order passed under Section 126 of the Electricity Act, if file Appeal under Section 127 of the Electricity Act. ii) Further, against the final order passed by the Appellant Authority under Section 127 of the Electricity Act, no complaint can be entertained by the Consumer Fora. officer under Section 126 of the Electricity Act. Further, the jurisdiction of the consumer fora is not barred by any provisions of the Electricity Act but the same is expressly saved under Section 173 read with Sections 174 and 175 of the Electricity Act. V. i) Section 3 of the Consumer Protection Act and Section 175 of the Electricity Act, provide that they Electricity Act. ii) A bare reading of Sections 173, 174 and 175, makes it clear that the intent of the Legislature is not iii) Section 42(8)of the Electricity Act specifically provides that the remedies conferred on consumer iv) Section 145 of the Electricity Act specifically bars the jurisdiction of the Civil Court to entertain or an Appellate Authority referred to in Section 127 of the Electricity Act or the Adjudicating Officer consumer as defined under Section 2(1)(o)of the Consumer Protection Act and a complaint alleging maintainable under the Consumer Protection Act.

Fig. 7 Output Screen

IV. RESULTS

The performance of ALIA demonstrates its effectiveness in automating legal document analysis. The system achieved an accuracy rate of over 92% in extracting essential legal elements such as citations, court names, and statutory references across various case formats. It efficiently generated summaries for legal documents under 500 words in less than 5 seconds, significantly reducing the time required for manual review. The integration of Google Gemini enabled the system to respond to legal queries with high contextual accuracy, offering relevant insights based on the document content. Furthermore, the explainable AI feature successfully simplified complex legal text, enhancing understanding for users with varying levels of legal expertise.



The Supreme Court set aside the orders of the National Commission. All appeals filed by the service provider-licensees were allowed. There was no order as to costs.

Fig. 8 Results.

V. DISCUSSIONS

The performance of ALIA highlights the transformative role of AI in streamlining legal document analysis. By automating the extraction and interpretation of legal content, the system significantly reduces the time and effort typically required for manual review. For digitally generated documents, ALIA delivers highly accurate results; however, when dealing with scanned PDFs, minor inaccuracies may arise due to OCR limitations, such as distorted text or unclear images. Despite these challenges, the system still performs reliably in most cases. Future enhancements could focus on improving OCR preprocessing techniques, such as image enhancement and noise reduction, and on fine-tuning AI models specifically for the legal domain to improve contextual understanding, especially in complex or ambiguous cases.

VI. CONCLUSION

ALIA offers a powerful, scalable, and intelligent approach to legal document analysis by combining the capabilities of Artificial Intelligence, Natural Language Processing, and Optical Character Recognition. It effectively automates the extraction, summarization, and interpretation of complex legal texts, significantly reducing the time and effort required for manual analysis. Through its user-friendly interface and integration with Google Gemini, ALIA enables real-time interaction with legal documents, making it a valuable tool for law students, advocates, legal researchers, and academic institutions. By simplifying the legal research process and improving accessibility to critical legal insights, ALIA stands out as a modern solution in the evolving landscape of legal technology.

VII. FUTURE SCOPE

ALIA has strong potential for future enhancements that can significantly broaden its usability and impact. One promising direction is the integration of multilingual OCR capabilities to support regional and vernacular court judgments, making the system more inclusive across jurisdictions. Developing a mobile-friendly version will enhance accessibility and allow legal professionals to conduct research on the go. Additionally, fine-tuning a legal-specific large language model (LLM) could further improve accuracy and contextual understanding of legal texts. Future versions may also introduce advanced features like document comparison, citation suggestions, and precedent mapping, helping users analyze legal trends more effectively. Finally, cloud deployment and multi-user access will enable real-time team collaboration, making ALIA suitable for law firms, educational institutions, and research groups.

VIII. ACKNOWLEDGEMENTS

Mr. Mugi Satish is an enthusiastic and committed faculty member in the Department of Computer Science. As an early-career academician, he has shown strong dedication to student development through active involvement in project guidance and technical mentoring. Despite being at the beginning of his professional journey, he has effectively guided students in executing academic projects with precision and conceptual clarity. His passion for teaching, coupled with a solid understanding of core computer science principles, positions him as a promising educator and mentor. Mr.Mugi Satish continues to contribute meaningfully to the academic environment through his proactive approach to learning and student engagement.

Kappala Swarna Madhuri is pursuing his final semester MCA in Sanketika Vidya Parishad Engineering College, accredited with A grade by NAAC, affiliated by Andhra University and approved by AICTE. With interest in Machine learning K. Swarna Madhuri has taken up her PG project on AI Legal Insight Analyser(ALIA) Using Machine Learning and published the paper in connection to the project under the guidance of Mr. Mugi Satish , Assistant Professor.

REFERENCES

[1] Lewis, M., et al. (2020). BART: Denoising Sequence-to-Sequence Pre-training. ACL. https://aclanthology.org/2020.acl-main.703/
[2] Raffel, C., et al. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer (T5). JMLR. https://jmlr.org/papers/v21/20-074.html
[3] Zhang, J., et al. (2020). PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization. ICML. https://proceedings.mlr.press/v119/zhang20ae.html
[4] Devlin, J., et al. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL. https://aclanthology.org/N19-1423/
[5] See, A., Liu, P. J., & Manning, C. D. (2017). Get To The Point: Summarization with Pointer-Generator Networks. ACL. https://aclanthology.org/P17-1099/
[6] Vaswani, A., et al. (2017). Attention is All You Need. NeurIPS. https://papers.nips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html
[7] Liu, Y., & Lapata, M. (2019). Text Summarization with Pretrained Encoders. EMNLP. https://aclanthology.org/D19-1387/
[8] Wolf, T., et al. (2020). Transformers: State-of-the-Art Natural Language Processing. EMNLP. https://aclanthology.org/2020.emnlp-demos.6/
[9] Rush, A. M., et al. (2015). A Neural Attention Model for Abstractive Sentence Summarization. EMNLP. https://aclanthology.org/D15-1044/
[10] Chopra, S., Auli, M., & Rush, A. M. (2016). Abstractive Sentence Summarization with Attentive RNNs. NAACL. https://aclanthology.org/N16-1012/
[11] Paulus, R., et al. (2018). A Deep Reinforced Model for Abstractive Summarization. ICLR. https://openreview.net/forum?id=HkAClQgA-
[12] Lin, C.-Y. (2004). ROUGE: A Package for Automatic Evaluation of Summaries. ACL. https://aclanthology.org/W04-1013/
[13] Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. ICLR. https://arxiv.org/abs/1412.6980
[14] Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation. https://www.bioinf.jku.at/publications/older/2604.pdf

[15] Bengio, Y., et al. (2003). A Neural Probabilistic Language Model. JMLR. http://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf
[16] Google AI Blog. (2020). Introducing PEGASUS for Abstractive Text Summarization. https://ai.googleblog.com/2020/12/pegasus-state-of-art-model-for.html
[17] Hugging Face Transformers Documentation. https://huggingface.co/docs
[18] Flask Web Framework Documentation. https://flask.palletsprojects.com
[19] PyTorch Documentation. https://pytorch.org/docs
[20] TensorFlow API Documentation. https://www.tensorflow.org/api_docs
[21] GitHub – Facebook BART CNN (fairseq). https://github.com/facebookresearch/fairseq/tree/main/examples/bart