IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151131

InsightAI: An AI-Powered Multi-Modal Document Summarization and Question Answering System Using Large Language Models and NLP Analytics

Bhavesh Kishor Patil[1], Pranav Ravindra Patil[2], Mayur Pankaj Patil[3], Yogesh Dnyaneshwar Patil[4], Prof. Kaminee Subhash Patil[5]

[1]UG student, SES's R. C. Patel Institute of Technology, Shirpur 425405 India[1]

[2]UG student, SES's R. C. Patel Institute of Technology, Shirpur 425405 India[2]

[3]UG student, SES's R. C. Patel Institute of Technology, Shirpur 425405 India[3]

[4]UG student, SES's R. C. Patel Institute of Technology, Shirpur 425405 India[4]

[5]Assistant Professor, SES's R. C. Patel Institute of Technology, Shirpur 425405 India[5]

Abstract: The exponential growth of digital content in multiple formats — text documents, audio recordings, and video lectures — has created significant challenges for efficient information extraction and knowledge synthesis. This paper presents InsightAI, a comprehensive web-based application that leverages Large Language Models (LLMs) and Natural Language Processing (NLP) techniques to provide intelligent multi-modal document summarization, interactive question answering, and analytical insights. The proposed system accepts inputs in seven formats including PDF, DOCX, TXT, MP3, WAV, MP4, and YouTube URLs, and utilizes the Google Gemini 2.5 Flash API for text processing and the Groq Whisper Large-V3 API for audio transcription. InsightAI offers six distinct summarization modes, a Retrieval-Augmented Generation (RAG) based QA chatbot, sentiment analysis, keyword and Named Entity Recognition (NER) extraction, automated quiz generation, flashcard creation, mind map visualization, and multi-language translation across ten languages. Summary quality is scored by a reference-free, LLM-based judge for coherence, consistency, fluency and relevance. A multi-user layer provides email-verified registration with one-time passwords (OTPs) sent through Nodemailer, bcrypt-protected passwords, signed session tokens, password recovery, and a relational database (SQLite for development, PostgreSQL for production) that stores each user's summaries and chat history privately. The system is built using a Python FastAPI backend and a React.js frontend, providing a scalable and user-friendly platform for students, researchers, and professionals.

Keywords: Large Language Models, Natural Language Processing, Document Summarization, Question Answering, Retrieval-Augmented Generation, Multi-Modal Processing, LLM-as-a-Judge Evaluation, User Authentication, One-Time Password, Sentiment Analysis, Google Gemini, Groq Whisper

I. INTRODUCTION

The exponential growth of digital information in the form of documents, research papers, lecture recordings, and meeting videos has created a significant challenge for individuals and organizations seeking to efficiently process and extract meaningful insights. Traditional approaches — reading entire documents or watching full videos — are time-consuming, cognitively demanding, and impractical when dealing with large volumes of data.

Natural Language Processing and Large Language Models have emerged as transformative technologies capable of understanding, summarizing, and generating human language at scale. These technologies enable intelligent systems to process raw text, audio, and video content and produce concise, structured, and meaningful outputs automatically.

InsightAI is developed as a comprehensive multi-modal AI application that goes beyond simple text summarization. The system integrates state-of-the-art LLM capabilities — specifically Google Gemini 2.5 Flash for all text tasks and Groq Whisper Large-V3 for audio and video transcription — with a feature-rich web interface, enabling users to upload diverse

 This work is licensed under a Creative Commons Attribution 4.0 International License



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.151131

document types, receive intelligent summaries in six modes, interact through a QA chatbot, and generate educational materials including quizzes, flashcards, and mind maps.

A. Motivation

Research shows that professionals spend an average of 2.5 hours daily reading documents and reports, while students often struggle to extract key concepts from lengthy academic materials [11]. The emergence of Large Language Models has created unprecedented opportunities to automate intelligent document processing at scale and at low cost.

B. Main Contributions

The main contributions of this work are:

- A multi-modal document processing pipeline supporting seven input formats: PDF, DOCX, TXT, MP3, WAV, MP4, and YouTube URL
- A dual-API architecture using Google Gemini for all text tasks and Groq Whisper for audio and video transcription
- A lightweight RAG-based QA chatbot without heavy ML dependencies (no FAISS, scipy, or PyTorch required)
- Integration of ten NLP features in a single unified platform accessible through a professional web interface
- Reference-free summary quality scoring with an LLM-based judge (coherence, consistency, fluency, relevance), inspired by G-Eval [16]
- A secure multi-user layer: email-OTP registration through Nodemailer, bcrypt password storage, JWT sessions, password recovery, and per-user persistent storage on SQLite or PostgreSQL
- Support for multi-language translation across ten languages including Hindi, Marathi, Spanish, French, and Japanese

II. RELATED WORK

A. Text Summarization

Early text summarization systems used extractive approaches based on TF-IDF scoring and graph algorithms such as TextRank [1]. The introduction of sequence-to-sequence neural networks [2] enabled abstractive summarization, and transformer-based models like BERT [3] and BART [4] significantly advanced summary quality. Recent surveys [5] [6] demonstrate that LLM-based abstractive summarizers consistently outperform traditional methods in human evaluation while scoring lower on exact-match metrics like BLEU due to paraphrasing.

B. Multi-Modal Summarization

Jangra et al. [7] present a comprehensive survey on multi-modal summarization covering text, image, audio, and video modalities, establishing that unified multi-modal systems outperform single-modality tools. The Whisper model by Radford et al. [8] demonstrated near-human accuracy in automatic speech recognition across 97 languages, enabling high-quality audio and video transcription as input to downstream NLP tasks.

C. Retrieval-Augmented Generation

Lewis et al. [9] introduced RAG as a method combining parametric LLM memory with non-parametric retrieved document context, significantly reducing hallucination in knowledge-intensive tasks. RAG is particularly effective for document Q&A where the model must restrict answers to content from a specific uploaded document.

D. Summary Evaluation Metrics

Lin [10] introduced ROUGE, which scores a summary by its n-gram and longest-common-subsequence overlap with a human-written reference. It needs a reference summary, which does not exist when a user uploads a new file, and it penalises paraphrase, so abstractive summaries from LLMs often score low even when they are good. Liu et al. [16] proposed G-Eval, in which an LLM rates a generated text on dimensions such as coherence, consistency, fluency and relevance without a reference, and reported higher agreement with human judgements than earlier automatic metrics. They also found that LLM-based evaluators can favour text written by LLMs, a bias that matters when the same model family writes and grades a summary.

E. Gap in Existing Systems

Table I compares InsightAI with existing tools. Most tools support only one input format and lack integrated educational features, multi-language output, and quality evaluation metrics.

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151131

TABLE I COMPARISON OF EXISTING SUMMARIZATION SYSTEMS

Tool	Formats	QA	Quiz	Lang	Free
ChatPDF	1	Yes	No	No	Part.
Quillbot	1	No	No	No	Part.
Notion AI	1	No	No	No	No
Adobe AI	1	Yes	No	No	No
InsightAI	7	Yes	Yes	Yes	Yes

III. SYSTEM ARCHITECTURE

InsightAI follows a three-layer client-server architecture: a React.js frontend presentation layer, a Python FastAPI backend API layer, and a cloud-based AI/LLM processing layer.

A. Dual-API Architecture

A key architectural decision is the use of two complementary APIs to balance cost, speed, and capability. Google Gemini 2.5 Flash handles all text-based AI tasks including summarization, QA, NLP feature extraction, quiz generation, mind mapping, and translation. Groq Whisper Large-V3 handles audio transcription for MP3, WAV, MP4, and YouTube inputs. FFmpeg is used to extract and compress audio from video files before transmission to the Whisper API, greatly reducing the size of the file sent to the API.

B. Backend Architecture

The backend is built using Python FastAPI with Uvicorn ASGI server and organized into modular files. Table II summarizes the full technology stack.

TABLE II SYSTEM TECHNOLOGY STACK

Component	Technology
Frontend	React 18, Vite, Tailwind CSS
Backend	Python FastAPI, Uvicorn
Text LLM	Google Gemini 2.5 Flash
ASR	Groq Whisper Large-V3
Audio Proc.	FFmpeg
PDF Parser	pdfplumber
DOCX Parser	python-docx
Export	ReportLab, python-docx
Lang Detection	langdetect
Evaluation	LLM judge (G-Eval style)
Database	SQLite / PostgreSQL (psycopg)
Auth	bcrypt, PyJWT
OTP e-mail	Node.js, Nodemailer

C. Frontend Architecture

The frontend is built with React 18 and styled using Tailwind CSS with a formal professional blue theme. The application provides an eight-tab sidebar navigation: Upload, Summary, QA Chat, Analytics, Quiz, Flashcards, Mind Map, and Translate. All API communication is handled through Axios with a Vite dev-proxy forwarding requests to the FastAPI backend on port 8000.

IV. AUTHENTICATION AND DATA PERSISTENCE

The first prototype served one anonymous visitor at a time and kept every result in server memory. To support many users, InsightAI adds email-verified accounts, protected sessions, password recovery and a relational database that stores each user's results privately.

A. Architecture

Four parts cooperate. The React frontend provides the Register, Verify-OTP, Login, Forgot-Password and My-Documents pages. FastAPI hosts the authentication and database modules. A separate Node.js service uses Nodemailer to create, e-mail and check one-time passwords; the API reaches it through an internal HTTP call and it is not exposed

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151131

to the internet. SQLite is used in development and PostgreSQL in production, selected by one environment variable; a thin wrapper keeps the SQL identical.

B. Registration and Email Verification

1) The user submits a name, email and password. The server hashes the password with bcrypt [18] and stores the account as unverified.
2) The OTP service draws a random six-digit code, stores it with a ten-minute expiry and sends it by email through Nodemailer.
3) The user enters the code. The service accepts it once and deletes it; the server marks the account verified and issues a JSON Web Token (JWT) [17].

Login is refused until verification succeeds. A user who registers again before verifying receives a new code instead of creating a duplicate account.

C. Sessions and Access Control

Tokens are signed with HS256, carry the user id and a seven-day expiry, and travel in an Authorization header. Every endpoint except registration, verification, login and recovery decodes the token, loads the user, checks that the account is verified, and filters all database queries by the user id taken from the token, so one user cannot list, open or delete another user's documents. The signing secret is read from an environment variable, and the server refuses to start on the hosting platform without it.

D. Account Recovery

Forgot-password sends a reset OTP to a verified account. The server returns the same message whether or not the email is registered, as recommended in OWASP guidance [19]. The OTP and the new password are submitted together; the OTP is consumed during verification and the new bcrypt hash replaces the old one.

E. Database Design

Table III lists the three tables. One user owns many documents and each document has many chat messages; deleting a document deletes its messages. Only derived results are stored, not the uploaded file. Structured NLP outputs (keywords, sentiment, quality scores) are kept as JSON text so their format can change without altering the schema.

TABLE III DATABASE SCHEMA

Table	Main columns
users	id, name, email (unique), password (bcrypt hash), is_verified, created_at, last_login
documents	id, user_id (FK), filename, file_type, word and character counts, summary, mode, language, keywords, sentiment, evaluation (JSON text), created_at
chats	id, user_id (FK), doc_id (FK), role, content, created_at

F. Security Controls

Table IV lists the controls and the threat each one addresses. The OTP is cancelled after five wrong guesses, which limits guessing of a six-digit code, and the account itself is never locked, so an attacker cannot lock out a victim.

TABLE IV SECURITY CONTROLS

Control	Mechanism	Threat
Password storage	bcrypt, salt per password	Database leak
Email ownership	Six-digit OTP, 10 min expiry	Fake accounts
OTP replay	Deleted after use	Code reuse
Guessing	Cancelled after 5 wrong tries	Brute force
Email flooding	30 s between OTPs per address (HTTP 429)	Inbox flooding
Enumeration	Same reply for unknown emails	Account discovery
Data access	Queries filtered by token user id	Cross-user leaks
Browser access	CORS origin allow-list	Other websites

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151131

V. METHODOLOGY

A. Document Parsing Pipeline

The parser module handles all file type detection and extraction. PDFs are processed page-by-page using pdfplumber. DOCX files are parsed paragraph-by-paragraph using python-docx. Audio files are sent to Groq Whisper. Video files are first processed through FFmpeg with parameters: 8 kHz sample rate, mono channel, 32 kbps bitrate, then transcribed by Whisper.

B. Summarization Algorithm

The summarization module supports six output modes:

- **Bullet Points:** Key ideas as a structured bullet list
- **Detailed Prose:** Comprehensive paragraph summary
- **Executive Brief:** Three sentences: topic, findings, conclusion
- **One-Liner:** Single informative sentence
- **Full Detailed:** Sections with Overview, Key Points, Conclusion
- **Academic Style:** Abstract format with argument, findings, implications

For documents under 12,000 characters, a single Gemini API call generates the summary. For longer documents, a hierarchical chunking strategy splits text into 10,000-character segments, generates bullet summaries per chunk, then combines partial summaries into the user's chosen output mode and language.

C. RAG-Based QA System

The QA module implements a lightweight RAG approach without heavy ML dependencies. Document text is split into paragraph-aware chunks of about 800 characters stored server-side by session ID. On each user question, a keyword intersection scoring function computes:

$$\text{score}(C_i, Q) = |W(C_i) \cap W(Q)| \tag{1}$$

where $W(C_i)$ is the word set of chunk i and $W(Q)$ is the word set of the question. The top-4 highest-scoring chunks are concatenated as context and passed to Gemini with a strict system prompt instructing the model to answer only from the provided context.

D. LLM-Based Quality Evaluation

Because a reference summary is not available at run time, quality is scored without one. A single prompt gives the judge the first 3,000 characters of the source document and the generated summary and asks it to rate four dimensions from 1 to 10: coherence (C), consistency with the source (K), fluency (F) and relevance (R). The overall score is

$$S = 2C + 3K + 2F + 3R \quad \text{(maximum 100)}, \tag{2}$$

and is labelled Excellent (≥80), Good (≥65), Fair (≥50) or Poor. The weights favour consistency and relevance. The judge is the same Gemini model that writes the summaries. The method is inspired by G-Eval [16] but simpler: it does not use G-Eval's generated evaluation steps or probability-weighted scoring.

VI. EXPERIMENTAL RESULTS

A. Functional and Security Verification

Two automated scripts test the multi-user layer. The first makes 25 requests against the API with the OTP service replaced by a fake one; it runs on a temporary copy of the backend, so real data is never touched. The second makes 14 checks on the Nodemailer service using a mock SMTP server and a mock HTTPS mail interface. Table V groups the checks. All 39 passed. The API suite passed on SQLite and, separately, on a local PostgreSQL 16 server. These tests do not send real email.

TABLE V VERIFICATION OF THE MULTI-USER LAYER

Area	Checks	Passed
Access control and CORS	3	3
Registration and sessions	5	5
Document storage	4	4
Chat history	2	2

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151131

Area	Checks	Passed
User isolation	4	4
Account recovery	7	7
OTP service: SMTP delivery, templates	6	6
OTP service: validation	3	3
OTP service: resend limit	1	1
OTP service: HTTPS mail interface	4	4
Total	**39**	**39**

B. Summary Quality and Processing Time

To be completed with measured data. Describe the test set (documents per input type, sizes) and the computer and network used. Table VI then reports, per input type, the mean processing time and the mean judge scores over n runs. Add a small human rating study to check the judge against people.

TABLE VI MEASURED TIME AND JUDGE SCORES (TO BE FILLED)

Input type	n	Time (s)	Score S
PDF	–	–	–
DOCX	–	–	–
TXT	–	–	–
MP3	–	–	–
MP4	–	–	–

C. System Screenshots

Figs. 1–6 show the implemented InsightAI web interface. All screens were captured while processing one test input, an MP4 lecture video on Docker whose transcript contains 3,889 words (19,891 characters).

Fig. 1 shows the login page, the entry point of the authentication layer described in Section IV. Fig. 2 shows the Upload page, where the user chooses a file or a YouTube URL, one of the six summary modes and one of ten output languages.

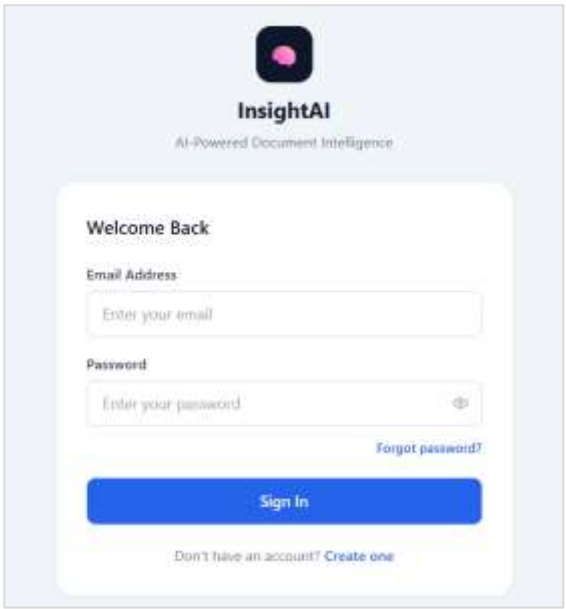


Fig. 1 Login page of InsightAI with email and password sign-in, password recovery and account creation links.

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151131

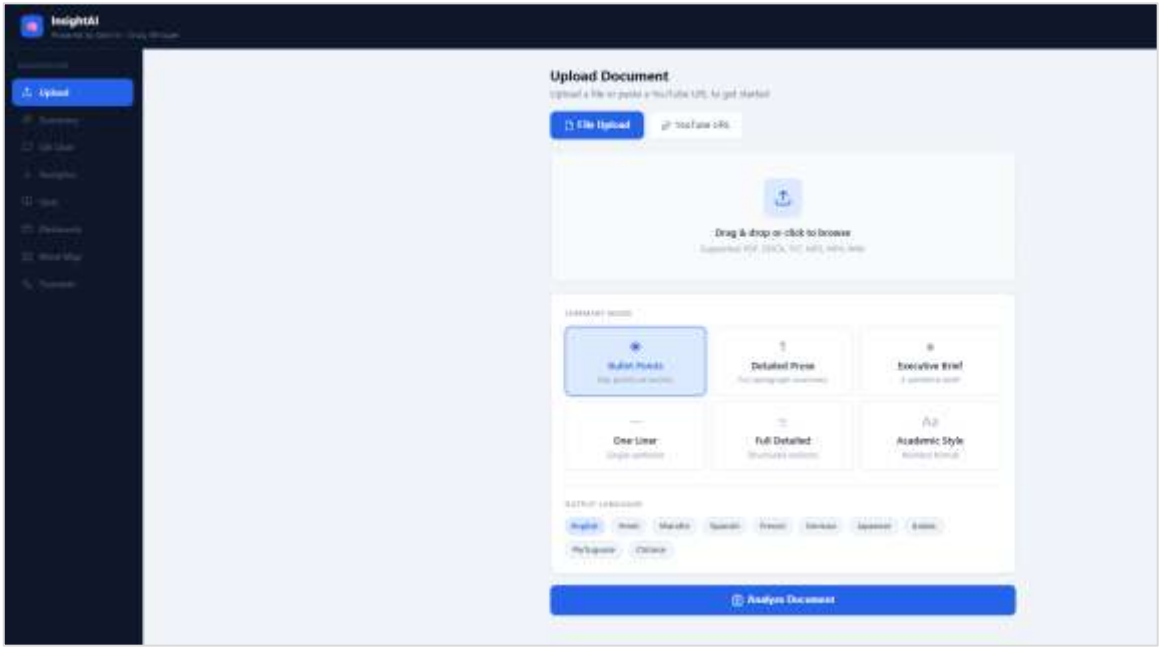


Fig. 2 Upload page with file and YouTube URL input, six summary modes and ten output languages.

After processing, the summary page reports the quality score of the LLM judge (Fig. 3). For this document the judge gave coherence 80%, consistency 90%, fluency 90% and relevance 80%, so the overall score of Eq. (2) is S = 2(8) + 3(9) + 2(9) + 3(8) = 85, which is labelled Excellent. The same page lists the extracted keywords, main topics and named entities.

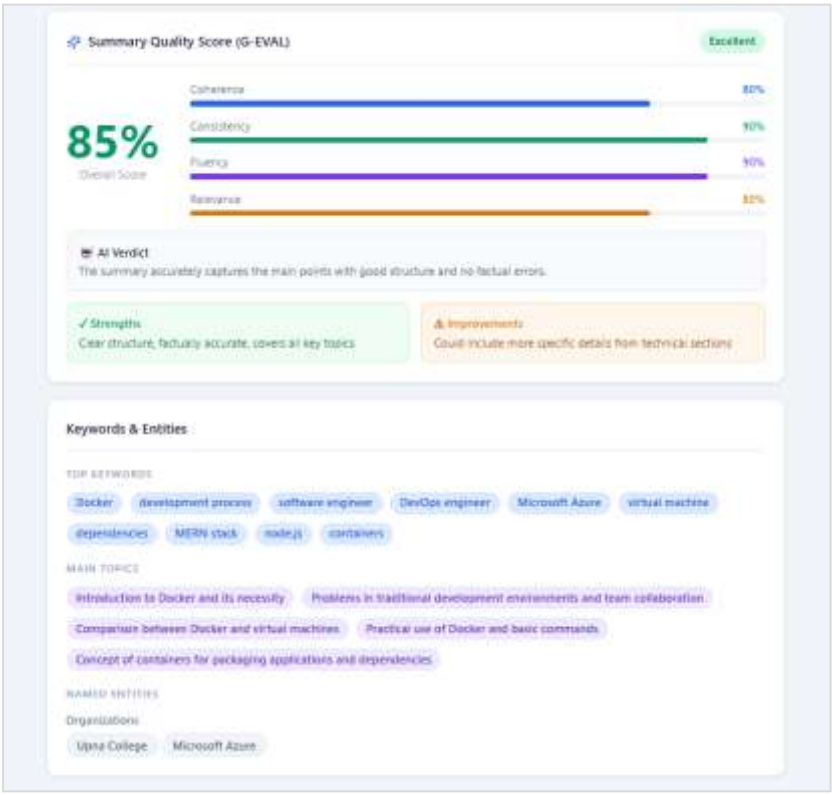


Fig. 3 Summary quality score (G-Eval style) with keywords, main topics and named entities.

Fig. 4 shows the document-grounded QA chat, in which the answer to the question “what is docker” is generated from the uploaded document. Fig. 5 shows the analytics page with word count, reading time, character count and the sentiment distribution (72% positive, 18% neutral, 10% negative). Fig. 6 shows the mind map generated for the same document.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151131

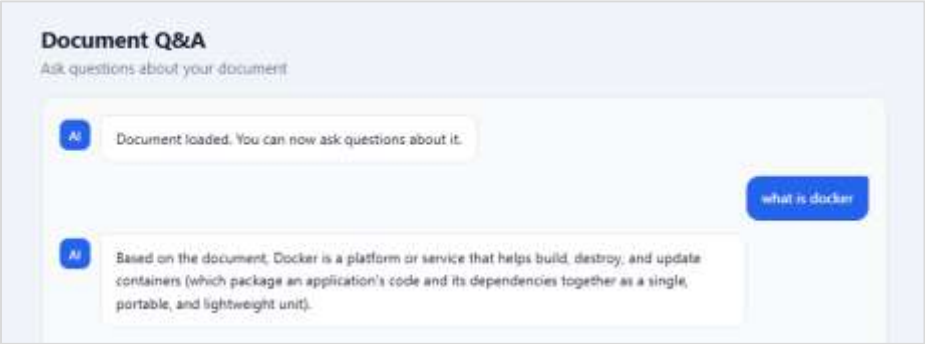


Fig. 4 Document Q&A chat answering a question from the uploaded document.

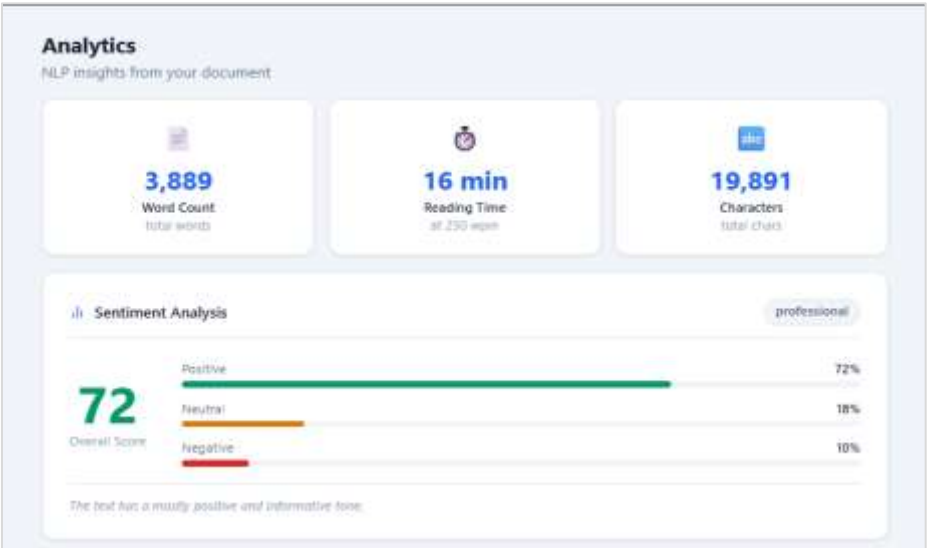


Fig. 5 Analytics page with word count, reading time, character count and sentiment analysis.

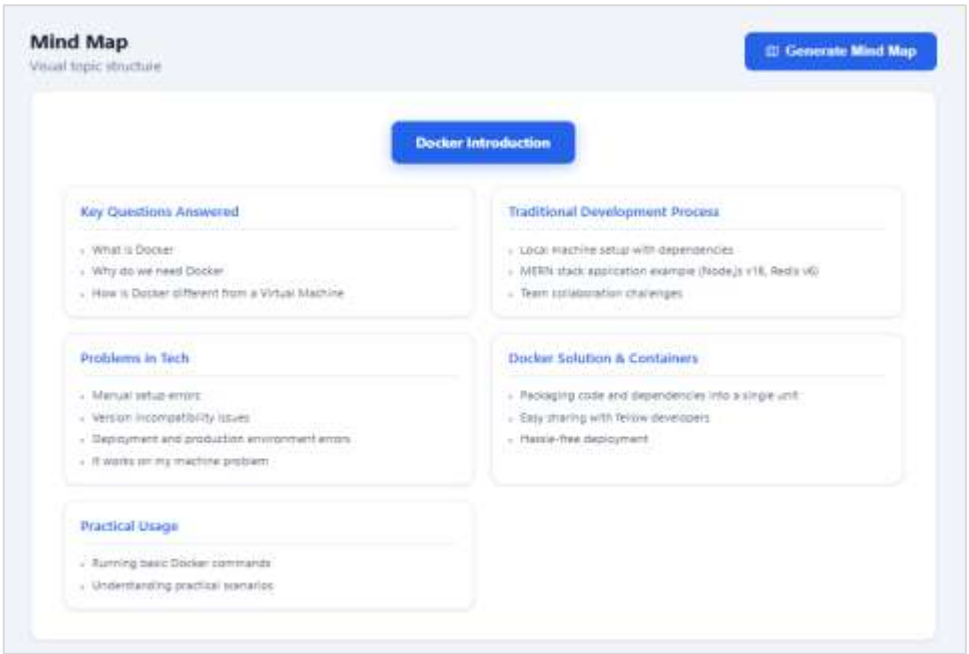


Fig. 6 Mind map of the document, grouped into topics and sub-points.

 This work is licensed under a Creative Commons Attribution 4.0 International License



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.151131

VII. DISCUSSION

A. Evaluation Approach

A reference-free judge fits a tool whose users upload new files, but its scores are indicative, not accuracy figures. The same model family writes and grades the summary, and Liu et al. [16] report that LLM judges can favour LLM-written text. The judge sees only the first 3,000 characters of the source, so for long documents consistency is checked against the opening only. The scores have not yet been compared with human ratings.

B. Dual-API Architecture

Gemini handles text and Groq Whisper handles audio. This split was adopted after the free tier of the Groq language model limited output to 1,000 tokens per minute, which was too low for long summaries, while Whisper transcription was unaffected. Whisper supports 97 languages [8].

C. Lightweight RAG Design

Keyword-overlap retrieval avoids sentence-transformers, FAISS, scipy and PyTorch, which together need downloads of over two gigabytes, so the system installs quickly on modest machines. It matches words, not meaning, so questions that share few words with the relevant passage may retrieve poor context. Embedding-based retrieval is future work.

D. Design Choices for Accounts and Hosting

Free hosting tiers erase local files when a server restarts and block the usual SMTP ports 25, 465 and 587 [20]. A file-based database would therefore lose all accounts, and Nodemailer could not reach a Gmail server. The code selects PostgreSQL and an SMTP relay on port 2525 through environment variables, and keeps SQLite and Gmail for local development. Email sending is kept in a separate Node.js service so that Nodemailer can be used while the rest of the system stays in Python.

E. Limitations

- The text and search chunks behind the quiz, flashcards, mind map and question answering are held in server memory and are lost on restart; saved summaries and chats are not.
- Documents longer than 50,000 characters are summarised from their first 50,000 characters only.
- Login tokens cannot be revoked before they expire, and there are no refresh tokens.
- Pending OTPs are kept in memory, so restarting the OTP service cancels them.
- The minimum password length is six characters, which is short and should be raised.
- The forgot-password reply is identical for all emails, but the server contacts the mail service only for registered accounts, so response time differs; guidance [19] advises uniform timing.
- Quality scores come from an unvalidated LLM judge, and no human evaluation has been done.
- Retrieval for question answering is keyword-based.

VIII. FUTURE WORK

- **Evaluation:** a human rating study, BERTScore, and a judge from a different model family to reduce self-preference
- **Account security:** refresh tokens with server-side revocation, a stronger password policy, uniform response timing, optional Google sign-in and two-factor authentication
- **Persistence:** store extracted text and original files (for example in object storage) so every feature works on an old document after a restart
- **Retrieval:** embedding-based RAG and questions across several documents
- **Long documents:** summarise the whole text instead of the first 50,000 characters
- **Reach:** a mobile application and shared workspaces
- **Domains:** specialised summarisation for legal, medical and financial documents

IX. CONCLUSION

This paper presented InsightAI, a multi-modal document intelligence platform that combines large language models, speech transcription and NLP analytics in one web application. It accepts seven input formats and provides six-mode summarization, a document-grounded question answering chatbot, sentiment analysis, keyword and entity extraction, quiz and flashcard generation, mind maps, translation and a reference-free quality score.

A secure multi-user layer turns the prototype into a platform for many users: email-verified registration with Nodemailer OTPs, bcrypt password storage, JWT sessions, password recovery, and a relational database that keeps each user's

 This work is licensed under a Creative Commons Attribution 4.0 International License



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.151131

summaries and chats private and permanent. Automated tests of this layer, 39 checks in total, all passed on SQLite and PostgreSQL. The dual-API design (Gemini for text, Groq Whisper for audio) runs on free tiers and needs no GPU. Measured processing times and a human validation of the quality judge remain to be reported, and the limitations listed above define the next steps.

REFERENCES

[1]. R. Mihalcea and P. Tarau, "TextRank: Bringing Order into Texts," in *Proc. EMNLP*, 2004, pp. 404–411.
[2]. I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," *NeurIPS*, 2014.
[3]. J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *NAACL*, 2019. [Online]. Available: https://arxiv.org/abs/1810.04805
[4]. M. Lewis et al., "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation," in *ACL*, 2020. [Online]. Available: https://arxiv.org/abs/1910.13461
[5]. G. Wang and W. Wu, "Surveying the Landscape of Text Summarization with Deep Learning: A Comprehensive Review," *arXiv:2310.09411*, Oct. 2023. [Online]. Available: https://arxiv.org/abs/2310.09411
[6]. Y. Zhang, H. Jin, D. Meng, J. Wang, and J. Tan, "A Comprehensive Survey on Automatic Text Summarization with Exploration of LLM-Based Methods," *arXiv:2403.02901*, 2024. [Online]. Available: https://arxiv.org/abs/2403.02901
[7]. A. Jangra et al., "A Survey on Multi-modal Summarization," *arXiv:2109.05199*, 2021. [Online]. Available: https://arxiv.org/abs/2109.05199
[8]. A. Radford et al., "Robust Speech Recognition via Large-Scale Weak Supervision," in *ICML*, 2023. [Online]. Available: https://arxiv.org/abs/2212.04356
[9]. P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *NeurIPS*, 2020. [Online]. Available: https://arxiv.org/abs/2005.11401
[10]. C.-Y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," in *Proc. ACL Workshop on Text Summarization*, 2004. [Online]. Available: https://aclanthology.org/W04-1013.pdf
[11]. W. X. Zhao et al., "A Survey of Large Language Models," *arXiv:2303.18223*, 2023. [Online]. Available: https://arxiv.org/abs/2303.18223
[12]. H. Zhang et al., "Extractive Summarization via ChatGPT for Faithful Summary Generation," *arXiv:2304.04193*, Apr. 2023. [Online]. Available: https://arxiv.org/abs/2304.04193
[13]. I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The Long-Document Transformer," *arXiv:2004.05150*, 2020. [Online]. Available: https://arxiv.org/abs/2004.05150
[14]. N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," in *EMNLP*, 2019. [Online]. Available: https://arxiv.org/abs/1908.10084
[15]. A. Vaswani et al., "Attention Is All You Need," in *NeurIPS*, 2017. [Online]. Available: https://arxiv.org/abs/1706.03762
[16]. Y. Liu, D. Iter, Y. Xu, S. Wang, R. Xu, and C. Zhu, "G-Eval: NLG Evaluation using GPT-4 with Better Human Alignment," in *Proc. EMNLP*, 2023, pp. 2511–2522. [Online]. Available: https://arxiv.org/abs/2303.16634
[17]. M. Jones, J. Bradley, and N. Sakimura, "JSON Web Token (JWT)," *RFC 7519*, IETF, May 2015. [Online]. Available: https://www.rfc-editor.org/rfc/rfc7519
[18]. N. Provos and D. Mazières, "A Future-Adaptable Password Scheme," in *Proc. USENIX Annual Technical Conference (FREENIX Track)*, 1999.
[19]. OWASP Foundation, "Forgot Password Cheat Sheet," *OWASP Cheat Sheet Series*. [Online]. Available: https://cheatsheetseries.owasp.org/cheatsheets/Forgot_Password_Cheat_Sheet.html
[20]. Render, "Deploy for Free," *Render documentation*. [Online]. Available: https://render.com/docs/free

 This work is licensed under a Creative Commons Attribution 4.0 International License