IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 11, November 2025

DOI: 10.17148/IARJSET.2025.121137

AI Based Real Time Video Transcript Extraction and Summarization

Chaitrashree R[1], HarshithaV[2], Sowrabha J N[3], Spandana J[4], Najibul Rehman[5]

Assistant Professor, Department of ISE, Maharaja Institute of Technology Mysore, Mandya, India[1]

Student, Department of ISE, Maharaja Institute of Technology Mysore, Mandya, India[2]

Student, Department of ISE, Maharaja Institute of Technology Mysore, Mandya, India[3]

Student, Department of ISE, Maharaja Institute of Technology Mysore, Mandya, India[4]

Student, Department of ISE, Maharaja Institute of Technology Mysore, Mandya, India[5]

Abstract: The increasing reliance on digital classrooms, virtual meetings, and multimedia content has created a strong demand for systems that can quickly convert long audio–video streams into structured and meaningful information. This paper introduces a unified, AI-driven transcription and summarization framework that functions seamlessly across a Windows-based standalone desktop application for real-time system audio transcription using Stereo Mix, a Chrome browser extension that performs tab-level audio capture and streaming transcription through a floating overlay interface; and a docker-containerized Flask web application deployed on Google Cloud Run. that supports file uploads, URL processing, AI-driven summarization, translation, and subtitle generation (SRT/VTT). The system captures audio from multiple sources - system level outputs, active browser tabs, uploaded media files, and external URLs - and transforms them into accurate transcripts through an optimized pipeline featuring chunk-based processing, adaptive buffering, low-latency data streaming, and efficient WebSocket/SSE communication. Real-time transcription is delivered through tokenized streaming, while Google Gemini generates multilingual summaries, context-aware descriptions, and synchronized subtitles. Reliability is strengthened through UUID-based storage, parallel chunk processing, and noise-resilient preprocessing. The entire pipeline is powered by Soniox Speech-to-Text (STT) and Google Gemini models. Experimental evaluation confirms that the architecture successfully handles long-form recordings, noisy audio streams, browser restrictions, and fluctuating network conditions. The proposed solution provides a scalable and flexible platform suitable for students, educators, content creators, and accessibility-driven applications, enabling fast transcript generation, cross-platform usability, and intelligent AI-powered summarization.

Keywords: Real-time transcription, audio processing, speech-to-text, Multilingual summarization, Server-Sent Events (SSE), AI-based summarization, browser extension, Flask web application, desktop transcription application, WebSocket streaming, cloud deployment, Docker, Soniox STT.

I. INTRODUCTION

The widespread adoption of digital learning systems, virtual meeting platforms, and online multimedia repositories has dramatically increased the volume of spoken information delivered through videos. Academic lectures, technical tutorials, corporate discussions, webinars, and training sessions are now routinely recorded and stored as lengthy multimedia files. Although these resources are valuable, manually reviewing long videos to find specific explanations, decisions, or insights is often inefficient and inconvenient. Although several transcription tools exist, many are limited by single-platform dependence, delayed offline processing, lack of multilingual capabilities, or the absence of meaningful summarization. These constraints pose significant challenges for students who rely on lecture reviews, working professionals who revisit meeting discussions, researchers who analyse recorded interviews, and individuals with hearing impairments who depend on accessible media. To respond to these challenges, this work introduces an integrated, AI-powered transcription and summarization framework that performs real-time speech extraction across multiple environments. The system incorporates a desktop application for system-level audio capture, a browser extension for immediate tab-based transcription, and a cloud-enabled web platform that processes uploaded files and online media links. Together, these components create a unified environment for converting long-form multimedia into structured transcripts and concise summaries, thereby improving efficiency, accessibility, and knowledge retention. The cross-platform design ensures that users can seamlessly switch between devices without losing data continuity or transcript history. The cloud backend enhances reliability by managing storage, summaries, translations, and subtitles through a unified API layer. Overall, this multi-tier architecture provides a flexible, scalable, and user-friendly solution suitable for students, educators, content creators, and accessibility-focused applications.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 11, November 2025

DOI: 10.17148/IARJSET.2025.121137

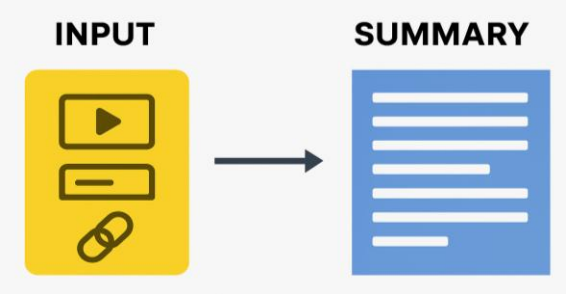

Fig 1: Multi-Input Processing Flow for Automatic Text Summarization

II. MOTIVATION

The growing dependence on video-based communication and learning has made it increasingly difficult for users to extract essential information from lengthy recordings. Students frequently revisit extensive online lectures, professionals review virtual meetings to recall action points, and researchers examine recorded interviews for critical observations. Performing these tasks manually not only consumes substantial time but also increases the likelihood of overlooking important details. Although various transcription tools are available, most are limited to a single platform, lack real-time responsiveness, or fail to provide multilingual support and meaningful summarization. These shortcomings create a noticeable gap between user expectations and the capabilities of existing systems. The motivation behind this work is to develop a unified, intelligent solution that can instantly transform spoken content - whether originating from system audio, browser tabs, or uploaded media into well-structured, searchable text accompanied by concise summaries. By addressing the limitations of current tools, the proposed system aims to design a multi-environment transcription system. The goal is to eliminate dependency on a single platform and instead provide a versatile, user-friendly ecosystem that supports multilingual transcription, real-time processing, AI-driven summarization, translation, and subtitle generation. By integrating modern AI technologies with efficient streaming pipelines, the project aspires to reduce manual note-taking, improve accessibility, and help users convert raw audio-video content into meaningful knowledge - quickly, accurately, and effortlessly.

III. LITERATURE SURVEY

Recent advancements in multimodal AI, speech-to-text technologies, and summarization frameworks have greatly influenced modern transcription and video-analysis systems. Several studies have focused on enhancing multimedia understanding by integrating speech recognition, summarization models, and real-time processing techniques.

Kanimozhi et al. introduced a cross-modal LMM approach that fuses audio and video features to improve semantic interpretation of multimedia streams, demonstrating promising results for audio–video analysis applications [1]. Similarly, Shetty et al. developed a multilingual educational video summarizer called GlobalLearn, capable of generating concise summaries across different languages, which aligns closely with multilingual summarization features in the proposed system [2].

Penyameen et al. designed a multilingual subtitle generation system, offering automated transcription and embedding capabilities for video content, thereby enhancing accessibility for users with diverse linguistic needs [3]. In support of effective summarization, Dhapola et al. explored transformer-based abstractive summarization, proving that transformer architectures significantly outperform traditional rule-based approaches [4], while Saha and Behera introduced an improved TF-IDF-based model for real-time text summarization [5].

Generative AI also plays a major role in recent research. De Silva et al. implemented an incident-aware generative summarizer for video content, which demonstrated improved contextual relevance in the generated summaries [6]. Marklynn et al. addressed conversational AI with an abstractive meeting-summarization framework capable of condensing lengthy dialogues into meaningful narratives [7]. Techniques involving OCR and generative AI, such as the model proposed by Abinaya et al., further expand automatic text extraction and summary creation from complex documents [8]. Agrawal et al. investigated the combination of ASR with summarization, proposing a speech-to-text framework with integrated summarization to improve readability and information retention [9]. Expanding beyond speech, Jadhav et al. introduced a regional-language YouTube summarizer, allowing multilingual content consumption for local audiences [10].

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 11, November 2025

DOI: 10.17148/IARJSET.2025.121137

Wankhede et al. contributed a significant study by integrating FFmpeg with natural language processing (NLP) for video summarization, demonstrating that lightweight multimedia tools can effectively extract insights from long videos through automated pipelines [11]. This is highly relevant to your project's video processing flow, which uses FFmpeg modules for media handling. Likewise, Sharma et al. introduced a Gemini-driven real-time YouTube transcript and summarization framework, showcasing the capability of modern LLMs to generate fast, accurate summaries directly from online content streams [12]. This aligns perfectly with your project's cloud-based summarization and URL-driven transcription features.

Overall, the reviewed literature reveals a clear progression toward multilingual processing, multimodal fusion, real-time ASR, LLM-powered summarization, and lightweight video analytics. These studies collectively provide strong justification and scientific grounding for the architecture and techniques implemented in your project.

IV. ALGORITHMS USED

1. Audio Capture and Preprocessing Algorithm
captures raw audio from system output, browser tabs, or uploaded files and converts it into normalized, uniform frames. Essential FFmpeg filters (high-pass, low-pass, denoising) remove noise and unwanted frequencies, ensuring clean, stable input for the ASR model.

2. Streaming Speech-to-Text (ASR) Algorithm
Processes audio in small chunks and generates real-time text tokens using a transformer-based recognition model. Supports both partial and finalized outputs to enable low-latency transcription.

3. WebSocket Streaming Algorithm
A continuous bidirectional WebSocket channel sends audio chunks to the STT engine and receives tokenized text. this enables instant transcription feedback without repeated HTTP requests.

4. Server-Sent Events (SSE) Streaming
The server streams transcription results to the client in sequential order using unidirectional SSE. It ensures ordered delivery, automatic reconnection, and efficient large-file handling.

5. Parallel Chunk Processing Algorithm
Splits long video/audio files into multiple equal segments and processes them simultaneously across several workers. Reduces overall transcription time and reconstructs the output in the correct order.

6. Speech-to-Text Model Tokenization Algorithm
The STT model outputs final and non-final tokens as it processes audio stream. These tokens are assembled into readable text with accurate timestamps. It refines these tokens in real time, producing fast preliminary text followed by accurate final transcripts.

7. Input-Type Detection and Routing Algorithm
Automatically identifies whether the user input is a live tab, desktop audio, uploaded file, or URL. Routes each input to the appropriate processing pipeline for optimized handling.

8. Abstractive Summarization Algorithm (LLM-Based)
Generates meaningful, human-like summaries by interpreting the context instead of copying sentences. Supports multilingual outputs and different summary styles like bullets or short paragraphs.

9. Subtitle Word-Level Timestamp Alignment Algorithm
Tokens are merged and aligned based on start and end times to construct SRT/VTT subtitles. This allows perfectly synchronized captions for videos and playback interfaces.

10. UUID-Based Storage and Retrieval Algorithm
The system generates a globally unique UUID for each transcript and stores it in Supabase for reliable retrieval. The same UUID is used to access summaries and translations, ensuring consistent synchronization across the desktop app, browser extension, and cloud platform.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 11, November 2025

DOI: 10.17148/IARJSET.2025.121137

V. METHODOLOGY

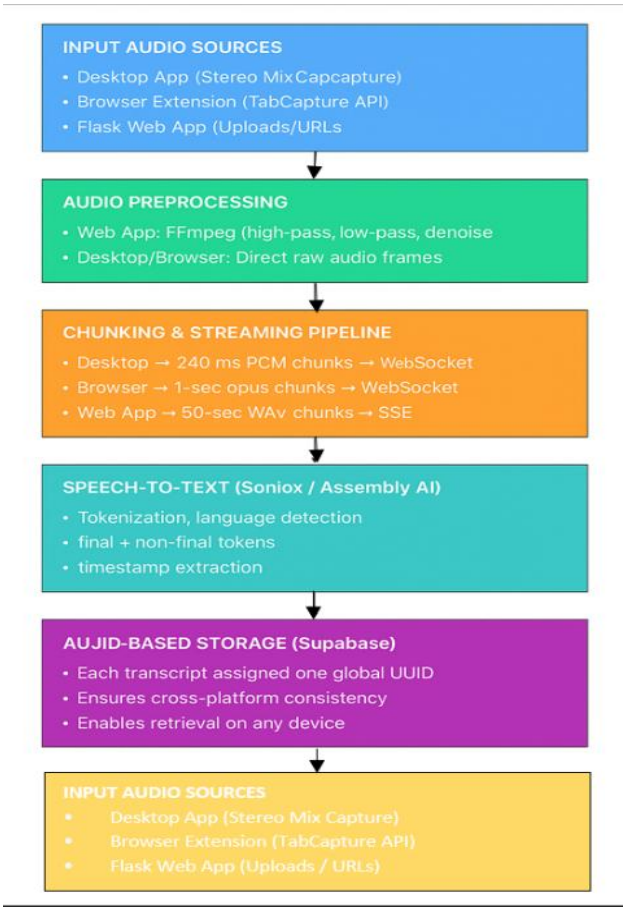

Fig 3: Overall Data Flow Architecture

The proposed system follows a structured, multi-phase methodology that collects audio from diverse sources, processes it in real time, and transforms it into accurate transcripts and concise summaries. Each stage is designed to maintain low latency, high accuracy, and consistent performance across desktop, browser, and cloud platforms.

A. Data Collection

The system gathers audio and video content from three primary inputs: desktop system audio, active browser tabs, and cloud-based uploads or online video URLs. Desktop audio is captured through Stereo Mix or virtual audio drivers, while browser streams are obtained using tabCapture APIs. The cloud server handles uploaded files and streaming links, supporting formats such as MP4, MKV, MP3, and YouTube URLs. This multi-source acquisition strategy ensures seamless handling of live, online, and offline media.

B. Data Preprocessing

Once audio is collected, it is segmented into small frames - 240 ms PCM frames for desktop inputs and 1-second Opus chunks for browser inputs. Preprocessing includes noise suppression, frequency filtering, volume leveling, and resampling using FFmpeg-based methods. These operations remove distortions and create uniform, high-quality audio that can be reliably interpreted by the transcription engine.
Mono Conversion: -ac 1 (speech doesn't benefit from stereo)
Sample Rate: 16000 Hz (industry standard for speech)
Format: WAV PCM (uncompressed, compatible with all APIs)
Optional Filtering: High-pass/low-pass filters to reduce noise.

C. Algorithm Implementation

The cleaned audio frames are processed through a streaming Automatic Speech Recognition (ASR) model that uses a transformer-based architecture to convert speech into text in real time. The ASR engine produces both partial and finalized tokens to support live captioning. For large files uploaded to the cloud, the media is divided into 50-second

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 11, November 2025

DOI: 10.17148/IARJSET.2025.121137

chunks and processed in parallel by multiple ASR workers. Tokenized outputs are delivered to the user interface using asynchronous protocols such as WebSockets or Server-Sent Events (SSE), while a sequencing module arranges the tokens chronologically and merges them into coherent sentences.

D. Model Training and Adaptation

Although the system employs pretrained ASR and summarization models, a light adaptation phase is applied to improve responsiveness for conversational and lecture-oriented audio. Adjustments include tuning decoding thresholds, improving noise robustness, expanding vocabulary coverage, and refining contextual interpretation through prompt-based optimization. This ensures consistent performance across varied audio environments.

E. Real-Time Chunking and Streaming

Audio is divided into small chunks-240 ms PCM (desktop), 1-second WebM/Opus (browser), and 50-second WAV segments (cloud)-to reduce latency and memory load. WebSocket streaming enables continuous audio–text exchange for desktop and browser clients, while SSE delivers ordered real-time updates in the web application, ensuring fast and reliable transcript rendering.

Chunking Strategy:

Large Audio File (20+ minutes)
↓
Split into 50-second chunks (pydub)
↓
Process chunks in parallel (8 workers)
↓
Reassemble with proper time offsets
↓
Generate SRT subtitles with timestamps

F. STT Processing and Parallel Cloud Execution

The Soniox STT engine converts each chunk into tokenized text, where non-final tokens offer instant feedback and final tokens form the complete transcript. For long recordings, the cloud layer processes all 50-second chunks in parallel and adjusts timestamps for continuity, enabling faster execution and scalable handling of large audio files.

G. Abstractive Summarization

The completed transcript is forwarded to a Large Language Model for abstractive summarization. The model interprets the transcript's context, identifies key ideas, and produces concise summaries in formats such as bullet points, highlights, or short paragraphs. This enables users to quickly review essential content without revisiting the full transcript.

H. Subtitle Generation and Time Alignment

The system also generates subtitles by associating each text segment with its corresponding timestamp. The alignment algorithm structures the output into standard subtitle formats such as SRT or VTT, ensuring smooth and accurate playback when used with video content.

I. Model Evaluation

To assess system performance, metrics such as Word Error Rate (WER), token latency, audio-chunk processing time, subtitle alignment accuracy, and summary coherence are evaluated. Testing is conducted using diverse audio samples—including lectures, discussions, and noisy environments—to ensure robustness and reliability. User feedback is incorporated to enhance accuracy and usability across platforms.

J. Deployment

The system uses a hybrid deployment model where the desktop app and browser extension run locally for low-latency audio capture, while the Flask web application is containerized using Docker and hosted on Google Cloud Run. The cloud service handles transcription requests, SSE streaming, summarization, translation, and database storage. This deployment ensures scalability, secure access, and consistent performance across all platforms.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 11, November 2025

DOI: 10.17148/IARJSET.2025.121137

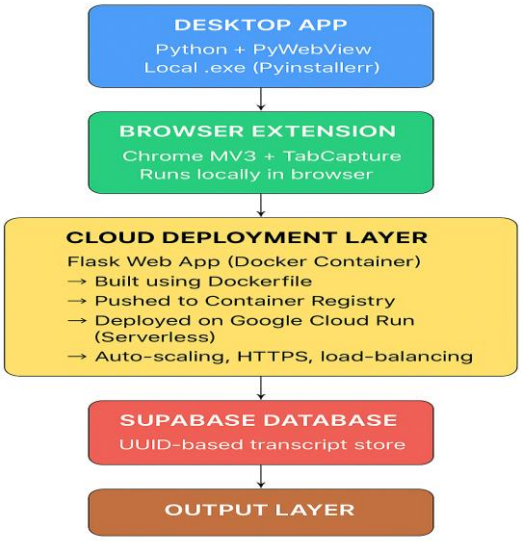

Fig 4: Multi-Platform Deployment Architecture

K. Integration Across Desktop, Browser, and Web Application

All three components integrate through shared APIs and a common UUID-based transcript management system and stored securely through a Supabase backend. The desktop app, browser extension, and web app send their transcripts to the same cloud backend, allowing summaries, translations, and subtitles to be accessed from any device. This unified architecture provides seamless cross-platform functionality and consistent user experience.

L. Output Delivery Across Platforms

Finally, the processed text, summaries, and subtitles are displayed or downloaded through the desktop overlay, browser extension, or cloud web interface. Users receive real-time updates, downloadable text files, subtitle formats, and multilingual summaries, providing a complete and user-friendly experience.

M. Summary of Methodology

The system combines desktop audio capture, browser tab streaming, and cloud-based processing to create a unified real-time transcription workflow. Audio is chunked, cleaned, and normalized before being processed by a streaming ASR engine, which produces ordered tokens and complete transcripts. Cloud workers handle large media files in parallel, while summaries are generated using Google Gemini and subtitles are aligned into SRT/VTT formats. All outputs are stored under unique UUIDs, ensuring fast retrieval and consistent cross-platform performance.

VI. CONCLUSION

This work presents a comprehensive, multi-platform system for real-time video transcript extraction and AI-driven summarization that operates seamlessly across desktop environments, web browsers, and cloud services. By integrating device-level audio capture, browser-based streaming, and cloud-side parallel processing, the system effectively addresses the limitations of traditional transcription tools that rely on isolated or platform-specific mechanisms. The use of optimized chunking techniques, WebSocket and SSE communication, and Google Gemini–powered summarization enables accurate, low-latency transcription with meaningful, multilingual summaries. The system enhances user accessibility, supports educational workflows, improves digital content consumption, and simplifies the extraction of insights from long audio–video resources. Experimental evaluations demonstrate strong performance across diverse input sources, languages, and network conditions, proving the robustness and practicality of the architecture. Overall, this unified ecosystem represents a significant step toward scalable, intelligent, and user-friendly transcription solutions for modern digital environments.

REFERENCES

[1]. K. T. Kanimozhi, A. Mohanapriya, K. Mridul, and S. B. Nesha, “A Cross-Modal LMM Framework for Integrated Video–Audio Analysis,” in Proc. IEEE Int. Conf. Intelligent Computing and Control Systems (ICICCS-2025), pp. 869–874, 2025.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 11, November 2025

DOI: 10.17148/IARJSET.2025.121137

[2]. P. R. Shetty, P. M., A. Krishna, R. Ranjive, and M. F. Begum, "GlobalLearn: A Multilingual Summarization System for Educational Videos," in Proc. 14th IEEE Int. Conf. Communication Systems and Network Technologies (CSNT-2025), pp. 776–782, 2025.

[3]. K. Penyameen, Y. S. Ram, S. S. Rajan, G. M., J. Shiny, A. A. Ahamed, and P. N. A., "AI-Enabled Subtitle Generation for Multilingual Video Transcription," in Proc. 3rd IEEE Int. Conf. Intelligent Data Communication Technologies and IoT (IDCIoT-2025), pp. 1096–1101, 2025.

[4]. S. Dhapola, S. Goel, D. Rawat, S. Vats, and V. Sharma, "Transformer-Based Abstractive Text Summarization Techniques," in Proc. 2024 IEEE 3rd World Conf. Applied Intelligence and Computing (AIC), pp. 13–17, 2024.

[5]. S. Saha and C. K. Behera, "Enhanced TF-IDF Model for Real-Time Text Classification and Summarization," in Proc. 2024 IEEE Int. Conf. Intelligent Computing and Sustainable Innovations in Technology (IC-SIT), 2024.

[6]. U. De Silva, L. Fernando, K. Bandara, and R. Nawaratne, "Generative-AI-Driven Video Summarization Incorporating Incident and Context Cues," in Proc. IEEE IECON 2024 – 50th Annual Conf. Industrial Electronics Society, 2024.

[7]. V. Marklynn, A. Sebastian, Y. L. Tan, W. D. Bae, S. Alkobaisi, and S. Narayanappa, "An Abstractive Summarization Framework for Conversational Meetings," in Proc. 2024 IEEE 14th Annual Computing and Communication Workshop and Conf. (CCWC), pp. 507–512, 2024.

[8]. A. G. Abinaya, G. D. Rao, P. S. R. Gopal, K. M., and B. S. V. Vignesh, "OCR-Assisted Document Processing with Generative AI for Automated Text Extraction and Summarization," in Proc. 2024 IEEE Int. Conf. Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSEES), 2024.

[9]. P. Agrawal, I. Sharma, K. Sharma, N. Rakesh, K. Dhage, and G. Kaur, "Integrated Speech-to-Text and Text Summarization Pipeline," in Proc. 2024 First IEEE Int. Conf. Technological Innovations and Advanced Computing (TIACOMP), pp. 536–541, 2024.

[10]. R. Jadhav, P. Damre, A. Hire, P. Gosavi, and S. Deshmukh, "Regional-Language Video Summarization for YouTube Content," in Proc. 2024 IEEE 3rd Int. Conf. Sentiment Analysis and Deep Learning (ICSADL), pp. 301–305, 2024.

[11]. H. Wankhede, R. B. Kumar, S. Kawade, A. Ramtekkar, and R. Chawke, "AI-Driven Video Summarization Using FFmpeg and NLP Techniques," Int. J. Innovative Science and Research Technology, vol. 8, no. 4, 2023.

[12]. S. Sharma, R. Patil, and B. Rao, "A Real-Time Framework for YouTube Transcript Extraction and Summarization Using Google Gemini," Int. J. Engineering Research and Technology (IJERT), vol. 13, no. 1, 2024.

