Siddhant Thorve, 2026, 14:1
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

Capture and Learning Intelligence Platform (CLIP)

Siddhant Thorve, Ashish Thakur, Harsh Thakur, Ms. Pooja Patil

Abstract- In traditional academic settings, students who miss live lectures often struggle to catch up due to a lack of structured resources and peer interaction. This project presents A-CLIP (Capture and Learning Intelligence Platform), a hybrid intelligent system designed to bridge this gap by transforming static video recordings into a comprehensive, interactive learning environment. Unlike standard video players, A-CLIP creates a "study-like" atmosphere where absent students can engage with material more deeply than in a physical classroom. The platform utilizes a microservices-inspired architecture, combining a Node.js orchestrator for real-time state management with a specialized Python service for high-performance AI tasks. For every uploaded lecture, the system automatically generates detailed, structured study notes using Llama 3.2 and curates supplementary external resources from the web, ensuring students have access to a wealth of context beyond the video itself. To reinforce mastery, an AI-driven adaptive quiz engine generates assessments with difficulty tiers (Beginner and Pro), locking advanced modules until foundational concepts are understood. Furthermore, the platform simulates the social aspect of learning through a "Study Room", where students can watch Siddhant Thorve. 2026, ISSN (Online): 2348-4098 ISSN (Print): 2395-4752 International Journal of Science, Engineering and Technology synchronously, video chat via WebRTC, and collaborate on shared whiteboards. This holistic approach ensures that missing a lecture no longer results in a learning deficit, but rather offers an opportunity for personalized, resource-rich study.

Keywords: Generative AI, Automated Note-Taking, Collaborative Study Environments, Real-Time Synchronization.

I. INTRODUCTION

Background and Motivation

The post-pandemic era has fundamentally accelerated the adoption of digital education, shifting the paradigm from traditional physical classrooms to hybrid and asynchronous learning models. Educational institutions increasingly rely on Learning Management Systems (LMS) and video repositories to deliver content to students who cannot attend live sessions. While these platforms provide accessibility, they often reduce the learning experience to a passive consumption of static media files.

Research indicates that asynchronous learning, while flexible, suffers from a critical "interaction gap." Students watching recorded lectures miss the dynamic social cues, peer discussions, and immediate feedback loops that characterize a physical classroom. This isolation often leads to lower engagement rates and reduced retention of complex technical concepts. As video becomes the dominant medium for remedial education, there is an urgent need to transform it from a passive broadcast into an active, intelligent, and collaborative workspace.

Problem Statement

Current solutions for video-based learning face two distinct limitations: passive consumption and content overload. Passive Consumption: Standard video players (e.g., YouTube, Google Drive) lack synchronous features. A student watching a missed lecture does so alone, without the ability to brainstorm, sketch ideas, or discuss doubts with peers in real-time.

Content Overload: A 60-minute technical lecture contains approximately 7,000–9,000 words of spoken content. Manually reviewing this footage to find specific concepts or creating comprehensive notes is time-consuming and inefficient. Existing platforms typically offer either collaboration (e.g.,

Zoom, Microsoft Teams) or content hosting (e.g., Moodle, Blackboard), but rarely integrate Generative AI and Real-Time Synchronization in a single, cohesive architecture to support remedial students.

Proposed Solution

(A-CLIP) To address these challenges, this paper presents the Capture and Learning Intelligence Platform (A-CLIP). A-CLIP is a hybrid intelligent system designed to bridge the gap between recorded content and active learning. Unlike traditional video players, A-CLIP serves as a dual-engine platform:The Collaboration Engine: Utilizes WebRTC and Socket.io to create a "Study Room" where multiple students can watch recorded lectures in perfect synchronization, video chat via a peer-to-peer mesh network, and collaborate on a shared whiteboard.

The Intelligence Engine: Deploys a specialized Python-based AI pipeline using Llama 3.2 and OpenAI Whisper to automatically generate structured study notes, quiz assessments, and visual knowledge graphs from raw lecture footage.

Paper Organization

The remainder of this paper is organized as follows: Section II reviews existing literature on e-learning and generative AI. Section III details the hybrid system architecture, specifically the Microservices interaction between Node.js and Python. Section IV describes the implementation of the real-time synchronization and AI pipelines. Section V presents the experimental results and performance analysis, and Section VI concludes with future research directions.

II. RELATED WORK

Evolution of E-Learning Platforms

Early e-learning systems, such as Moodle and Blackboard, focused primarily on file management and assignment tracking. While effective for administration, these platforms treat video lectures as static files, offering no interactive features. Recent platforms like Coursera and Udemy have improved content delivery but still rely on an asynchronous model where students watch videos in isolation. Research suggests that this lack of synchronous peer interaction leads to lower completion rates compared to physical classrooms.

Generative AI in Education

The rise of Large Language Models (LLMs) has introduced new possibilities for automated tutoring. Tools like ChatGPT and various browser extensions can now summarize text or generate quizzes. However, these tools generally operate as external plugins rather than integrated platform features. Students are often forced to manually copy transcripts into separate AI tools to get summaries, creating friction in the learning process. There is a lack of platforms that automatically trigger these AI workflows the moment a teacher uploads a lecture.

Real-Time Collaborative Systems

Technologies like WebRTC and WebSocket have enabled real-time communication in tools like Zoom and Microsoft Teams. While these are excellent for live meetings, they are not optimized for recorded study sessions. They lack the specific state-synchronization required to let a group of students pause, rewind, and seek through a recorded lecture together while maintaining perfect sync.

Summary of Existing Gaps

Current solutions fall into two separate categories: video hosting sites (YouTube, Google Drive) which lack collaboration, and meeting apps (Zoom) which are not designed for studying past content. There is currently no unified platform that combines the high- quality storage of a video host with the real-time social features of a meeting app and the automated intelligence of generative AI. A-CLIP aims to fill this specific gap.

III. PROPOSED SYSTEM ARCHITECTURE

Architectural Overview

The Capture and Learning Intelligence Platform (A-CLIP) is architected as a Hybrid Microservices System, fundamentally designed to decouple I/O-bound operations from CPU-bound artificial intelligence tasks. This separation of concerns addresses the dual challenge of modern e-learning systems: providing a low-latency, real-time user

experience while simultaneously executing computationally intensive generative models.

The system is composed of four distinct layers: the Client Presentation Layer, the Node.js Orchestration Layer, the Python Intelligence Layer, and the Hybrid Data Infrastructure.

The Orchestrator (Node.js & Express): Serving as the system's central nervous system, this layer functions as an API Gateway and state manager. Built on the non-blocking, event-driven architecture of Node.js, it is optimized for high concurrency. Its primary responsibilities include:

- Real-Time State Synchronization: It hosts the Socket.io server which maintains a persistent WebSocket connection with all active clients in a "Study Room." This allows for the broadcasting of video state events (Play, Pause, Seek) with sub-100ms latency.
- Request Routing: It intercepts all client requests, routing authentication calls to Firebase and complex analysis requests to the Python service.
- Storage Management: It implements a logic tier that intelligently segments data, streaming heavy media to hot storage (Cloudinary) while directing lightweight metadata and JSON logs to cold storage (Google Drive).

The Intelligence Engine (Python & FastAPI): This dedicated microservice acts as the system's computational "brain." By isolating the AI stack in a separate Python environment, the system prevents heavy inference tasks from blocking the main application thread.

- Model Pipeline Management: It orchestrates a multi-stage pipeline where raw media is first processed by FFmpeg, transcribed by OpenAI Whisper, and then synthesized by Llama 3.2 for high-level cognitive tasks such as summarization, quiz generation, and diagrammatic reasoning.
- Multilingual Support: It integrates the NLLB-200 model to provide real-time translation capabilities, ensuring accessibility across language barriers.
- The Client Presentation Layer (React.js): The frontend is a Single Page Application (SPA) built with React.js, designed to handle dynamic content rendering without full page reloads. It integrates a Peer-to-Peer (P2P) Mesh Network using WebRTC, allowing client browsers to exchange video and audio streams directly for video conferencing, bypassing the central server to reduce bandwidth costs.
- Inter-Service Communication: Communication between the Node.js Orchestrator and the Python Intelligence Engine is established via internal RESTful API calls. The Node.js server sends a "trigger" payload containing the media reference, and the Python service responds asynchronously with the generated knowledge artifacts (JSON notes, Mermaid syntax, Quiz objects), which are then persisted to the database.

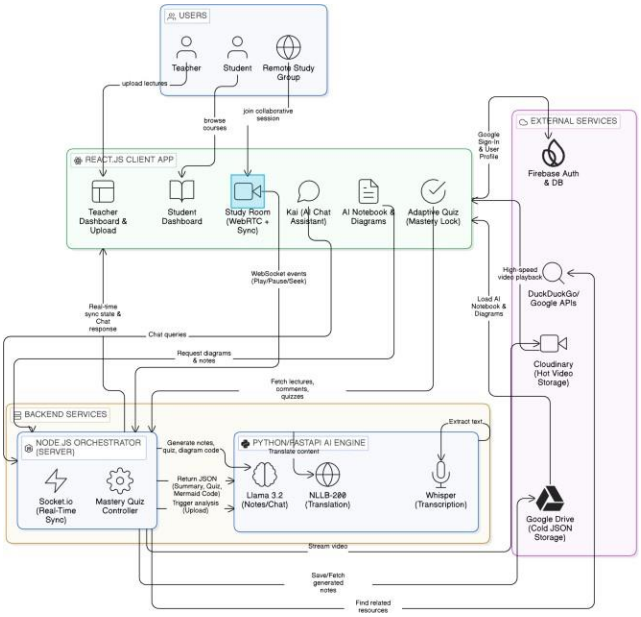

Figure. 3.1 System Architecture

The Hybrid Storage Model

To optimize costs without sacrificing performance, the platform implements a dual-storage strategy:

- Hot Storage (Cloudinary): Used exclusively for video streaming. It provides adaptive bitrate streaming (HLS) to ensure smooth playback on different bandwidths.

- Cold Storage (Google Drive): Used for storing generated metadata, JSON logs, and AI summaries. By utilizing institutional Google Drive storage for these lightweight files, the platform significantly reduces the hosting costs associated with traditional cloud databases.

Data Flow Pipeline
The system processes data in three stages:

- **Ingestion:** A lecture is uploaded via the client. The Node.js server immediately streams the video to Cloudinary and saves a raw reference.

- **Processing:** The Node.js server triggers the Python service. The system extracts audio, transcribes it, and runs it through Llama 3.2 for summarization and NLLB for translation.

- **Delivery:** The processed JSON metadata is stored in Drive. When a student opens the "Study Room," the frontend fetches the video from Cloudinary and the notes from Drive simultaneously.

IV. METHODOLOGY AND IMPLEMENTATION

Intelligent Content Generation Pipeline
This pipeline transforms raw video into structured knowledge:

- **Audio Extraction:** The audioProcessing.py module uses FFmpeg to strip audio tracks from uploaded video files efficiently.
- **Transcription & Translation:** The audio is processed by OpenAI Whisper to generate timestamped transcripts. If a target language is selected, the NLLB-200 model translates the text while preserving educational context.
- **Visual Generation:** The text is analyzed to identify relationships between concepts. The system generates Mermaid.js syntax, which is rendered on the client side using ELK algorithms to produce auto-arranged flowcharts and diagrams.

Real-Time Synchronization Engine
To simulate a physical classroom, the platform ensures all users see the same content at the exact same time:

- **State Sync:** A Socket.io namespace manages the "Video State" (Play, Pause, Seek). When a host pauses the video, the event is broadcast to all peers within milliseconds, ensuring synchronized viewing.

- **Peer-to-Peer Mesh:** Video chat features are implemented using WebRTC (PeerJS). This creates a direct mesh network between students' browsers, allowing for low-latency video calls without routing heavy media traffic through the central server.

Adaptive Assessment Algorithm
The quiz module uses a "Mastery Learning" approach. The system generates two sets of questions: Foundations (basic recall) and Pro (application-based). The application logic enforces a lock on the Pro module, which is only unlocked after the student scores above a threshold (e.g., 60%) on the Foundations quiz.

V. RESULTS AND ANALYSIS

System Implementation
The proposed system was successfully implemented with a fully functional "Study Room" supporting concurrent users. The interface integrates the video player, collaborative whiteboard, and AI- generated notebook into a single cohesive dashboard.

Figure. 5.1 Homepage

The A-CLIP homepage serves as a secure, role-based gateway for students and teachers to access their personalized learning dashboards. Its clean interface positions the platform as a "smart lecture assistant," ensuring intuitive navigation to all study tools.

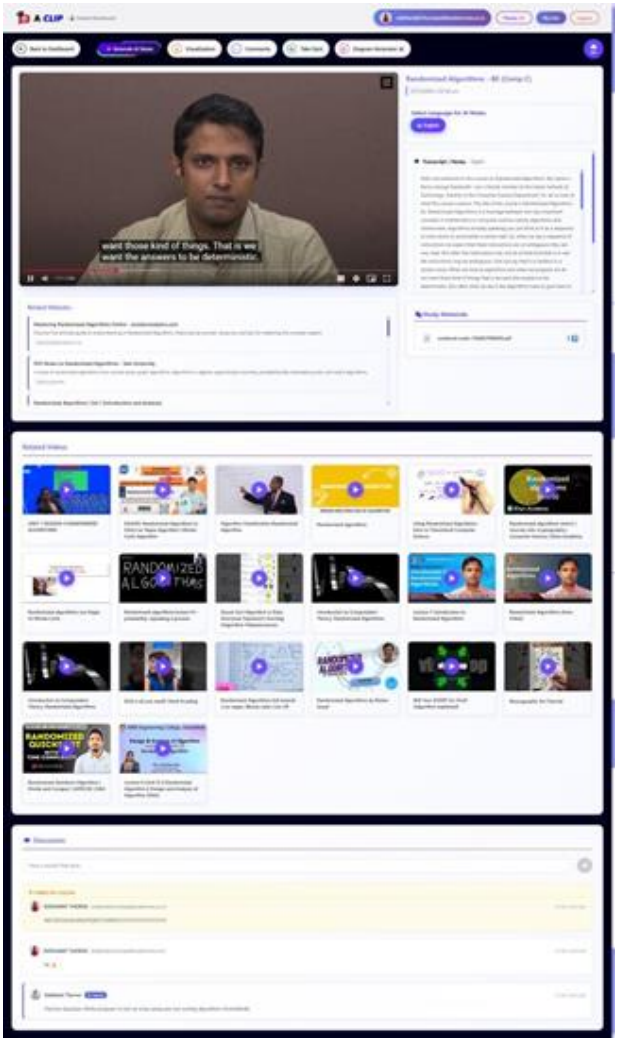

Figure. 5.2 Student Dashboard

The Student Dashboard integrates a video player with synchronized AI transcripts, automated study notes, and curated external resources to create a comprehensive learning environment. It enhances engagement through a built-in discussion forum and personalized recommendations, ensuring students have all necessary materials in a single view.

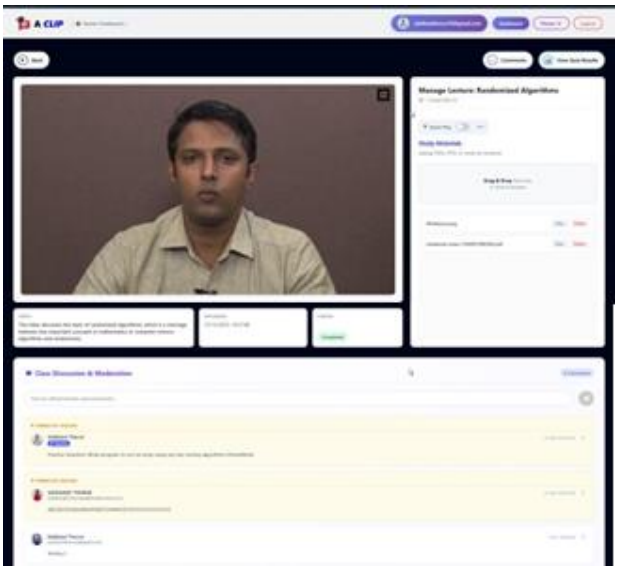

Figure. 5.3 Teacher Dashboard

The Teacher Dashboard serves as a centralized command center where educators can manage lecture content, upload supplementary study materials (like PDFs), and configure playback settings. It also features a robust moderation suite that allows teachers to pin announcements and facilitate class discussions, ensuring a structured and responsive learning environment.

Quantitative Performance Metrics

We analyzed the system's performance across three key metrics:

- **Synchronization Latency:** The average delay between a host triggering a "Pause" event and a client receiving it was measured at
- <85ms over a standard 4G network, providing a near-instantaneous experience.

- **AI Processing Speed:** On a standard T4 GPU environment, the processing ratio was

approximately 1:5 (i.e., a 10-minute video is processed in ~2 minutes), making it viable for near-real-time lesson generation.

- **Resource Utilization:** The Node.js Orchestrator maintained low memory usage (<500MB) even under load, while the Python service efficiently managed VRAM by loading quantized (Int8) versions of the Llama models.

VI. DISCUSSION

Impact on Learning Outcomes

Traditional online learning often suffers because students simply watch videos passively, which leads to boredom and poor information retention. A-CLIP solves this by turning video watching into an active, social experience. By allowing students to watch lectures together in the "Study Room," the platform encourages them to pause the video and discuss difficult concepts the moment they appear. This immediate peer interaction helps clarify doubts instantly rather than waiting for a teacher. Additionally, because the AI automatically generates notes and summaries, students no longer need to split their attention between listening and writing. This lowers their cognitive load, allowing them to focus entirely on understanding the material and participating in the discussion, which naturally leads to better memory retention and deeper understanding.

Scalability

The system is designed to handle growth without slowing down. By separating the application into two distinct parts, we ensure that heavy tasks do not block simple ones. The Node.js server acts like a traffic controller, efficiently managing lightweight tasks such as keeping videos in sync and delivering chat messages for hundreds of users at once. Meanwhile, the heavy computational work, like transcribing audio and generating quizzes, happens entirely in the separate Python engine. This separation means that even if the AI is processing a massive one-hour lecture, the "Study Room" remains instant and lag-free for students. This architecture allows the platform to support many simultaneous classrooms without crashing or freezing.

Limitations

The primary constraint is the hardware requirement for the Python service. Running Llama
3.2 and NLLB locally requires a machine with at least 8GB of VRAM. Future iterations may require cloud-based GPU clusters for larger-scale deployments.

VII. CONCLUSION AND FUTURE SCOPE

Conclusion

The development of A-CLIP successfully proves that advanced Artificial Intelligence can be combined with real-time collaboration tools to fix the problem of passive online learning. Instead of students just watching a video alone, the platform creates an active environment where they can watch together, discuss doubts instantly, and rely on AI to handle the note- taking. This project demonstrated that using a "hybrid" approach—letting Node.js handle the traffic and Python handle the thinking—is a practical way to build powerful educational tools without high server costs. Ultimately, A-CLIP transforms a standard video player into a smart, interactive study companion that helps students learn faster and retain more information.

Future Scope

Moving forward, we plan to expand the platform to make it even more accessible and immersive. Future enhancements will focus on:

- **AI-Driven Personalized Learning Paths:** Instead of a "one-size-fits-all" approach, we plan to upgrade the system to analyze each student's quiz scores and viewing habits. The platform will then automatically recommend a unique study schedule, suggesting specific videos or revision notes to strengthen their weak areas, acting like a personal academic counselor.

- **Virtual Reality (VR) Classrooms:** Currently, students interact via standard video chat. We plan to upgrade this into a 3D virtual environment where students can join the "Study Room" as avatars. This would simulate a real physical classroom, allowing students to "sit" next to friends and interact in a 3D space, making remote learning feel much less lonely.

- **Offline Mode (PWA):**
  Since internet access can be unstable, we want to turn the platform into a Progressive Web App (PWA). This would allow students to download their lectures, AI-generated notes, and quizzes while they have Wi-Fi, and then study smoothly later without needing an active internet connection.

- **Voice-Controlled AI Assistant:**
  We plan to expand the "Kai" chat assistant to support voice commands. This would allow students to ask questions verbally during a lecture—like "Hey Kai, explain that last concept again"—and receive spoken answers, making the platform fully accessible for visually impaired students or those who prefer auditory learning.

REFERENCES

1. M. Guettala, S. Bourekkache, O. Kazar, and S. Harous, "Generative artificial intelligence in education: Advancing adaptive and personalized learning," Acta Informatica Pragensia, vol. 13, no. 3, pp. 460–489, 2024.

2. Z. Kolagar and A. Zarcone, "HumSum: A personalized lecture summarization tool for humanities students using LLMs," in Proc. 1st Workshop on Personalization of Generative AI Systems (PERSONALIZE 2024), St. Julians, Malta, 2024, pp. 36–70.

3. M. Canal-Esteve and Y. Gutierrez, "Educational material to knowledge graph conversion: A methodology to enhance digital education," in Proc. 1st Workshop on Knowledge Graphs and Large Language Models (KaLLM 2024), Bangkok, Thailand, 2024, pp. 85–91.

4. Y. Zhai et al., "Generative AI in personalized learning: Development trajectory, educational applications, and future education," Frontiers in Education, vol. 9, p. 1424386, 2024.

5. Technavio, "Web real-time communication (WebRTC) market growth analysis - size and forecast 2025-2029," Technavio Research Report, London, UK, Jan. 2025.

6. C. K. Lo, "Impact of AI-based personalized learning on student achievement and engagement: A meta-analysis," Educational Research Review, vol. 41, p. 100626, 2024.

7. S. Pan et al., "Unifying large language models and knowledge graphs: A roadmap," IEEE Transactions on Knowledge and Data Engineering, vol. 36, no. 7, pp. 2976–2995, 2024.

8. P. C. H. Chan and K. K. W. Lee, "The era of generative AI in higher education: Automated summarization and assessment," Open Education Studies, vol. 5, no. 1, 2023.

9. D. Baidoo-Anu and L. O. Ansah, "Education in the era of generative artificial intelligence (AI): Understanding the potential benefits of ChatGPT in promoting teaching and learning," Journal of AI, vol. 7, no. 1, pp. 52–62, 2023.

10. M. K. Hassan and R. Ali, "Performance analysis of WebRTC for real-time video conferencing in e-learning platforms," IEEE Access, vol. 11, pp. 15430–15442, 2023.

Author's details

- Siddhant Thorve, Student, Computer Engineering, Pillai HOC College of Engineering and Technology, (Autonomous),Maharashtra, India,siddhantjt23hcompe@student.mes.ac.in

- Ashish Thakur, Student, Computer Engineering, Pillai HOC College of Engineering and Technology, (Autonomous), Maharashtra,India, ashishvt22hcompe@student.mes.ac.in

- Harsh Thakur, Student, Computer Engineering, Pillai HOC College of Engineering and Technology,(Autonomous),Maharashtra,India,harshhath22hcompe@student.mes.ac.in

- Ms. Pooja Patil, Assistant Professor, Computer Engineering, Pillai HOC College of Engineering and Technology, (Autonomous), Maharashtra, India, poojapatil@mes.ac.in