ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26may052

A Hybrid AI- and WebRTC-Based Collaboration Plat-form with Real-Time Speech-to-Text Transcription and an Interactive Chatbot System

Dr. S. Thaiyalnayaki[1]; Kailash T.[2]; K. V. Kishore[3]; Khushi Rani[4]; Kilari Bhavya[5]

[1]Associate Professor; Department of Computer Science Bharath Institute of Higher Education and Research Chennai, TamilNadu
[2]Department of Computer Science Bharath Institute of Higher Education and Research Chennai, TamilNadu
[3]Department of Computer Science Bharath Institute of Higher Education and Research Chennai, TamilNadu
[4]Department of Computer Science Bharath Institute of Higher Education and Research Chennai, TamilNadu
[5]Department of Computer Science Bharath Institute of Higher Education and Research Chennai, TamilNadu

Publication Date: 2026/05/06

Abstract: Online meeting and collaboration tools have become vital for digital communication, but many existing platforms are limited to basic audio and video functional-ity and lack intelligent features that support accessibility and user engagement. To overcome these challenges, our proposed system introduces a real-time collaborative online meeting application that integrates AI-based chat assistance and live speech transcription, aiming to im-prove the efficiency, inclusiveness, and effectiveness of virtual meetings. The system is implemented using a React.js frontend and a Node.js backend with Express and Socket.IO to enable seamless real-time interactions. Firebase Authentication is used to provide secure access control, while Firestore serves as a cloud-based repository for storing chat conversations and meeting transcripts. Real-time audio and video communication is facilitated through WebRTC, with Socket.IO handling signaling be-tween participants. An intelligent chatbot powered by the Groq API offers contextual support during meetings, and a live transcription module utilizing the browser's Web Speech API performs automatic speech recognition to convert spoken content into real-time text and store it with speaker identification. By combining real-time com-munication, cloud technologies, and applied artificial in-telligence, the system delivers a more interactive, accessi-ble, and productive virtual collaboration experience.

Keywords: *WebRTC, AI-Powered Chatbot, Natural Lan-guage Processing (NLP), Speech-to Text Conversion, Auto-matic Speech Recognition (ASR), Quality of Service (QoS).*

How to Cite: Dr. S. Thaiyalnayaki; Kailash T.; K. V. Kishore; Khushi Rani; Kilari Bhavya (2026) A Hybrid AI- and WebRTC-Based Collaboration Plat-form with Real-Time Speech-to-Text Transcription and an Interactive Chatbot System.
International Journal of Innovative Science and Research Technology, 11(5), 1-9.
https://doi.org/10.38124/ijisrt/26may052

I. INTRODUCTION

Remote work, online classes, teams scattered across the map—nowadays, everyone needs sharper virtual meeting tools. Real-time collaboration isn't just nice to have anymore; people expect it. They want to talk, see each other, and bounce ideas around instantly, from whatever device is handy. There are plenty of video conferencing apps out there, but most just stick to the basics: audio, video, and a big, slow, centralized server in the middle. That's a recipe for lag, high costs, and trouble when you try to scale up.

But web tech has come a long way. Now you can launch a meeting straight from your browser—no annoying plugins or clunky installs. Take WebRTC, for example. It connects peo-ple directly, so audio and video flow with barely any lag. The catch? Most WebRTC setups just handle the calls. They don't toss in smart features like AI-powered assistants, chat that ac-tually understands the conversation, or automatic meeting notes.

Meanwhile, AI is finally pulling its weight in collaboration. Large language models and chatbots don't just spit out canned answers anymore—they actually get what you're saying, lend a hand, and keep you moving. Drop one of these assistants into a meeting and suddenly you've got someone who can an-swer questions on the fly, steer the group, or help make calls. But right now, most tools hook up to AI services directly from each user's device. That's not ideal if you care about security, want to control how the AI is used, or need everything tied together inside the meeting.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26may052

Don't forget accessibility and keeping a record. Live speech transcription is a game changer for anyone following along—especially folks who are hard of hearing—and it makes find-ing details later a breeze. Cloud transcription gets things right, but it means shipping audio off to some remote server, which adds delay and costs. Running speech recognition in the browser is snappier and lighter, but it's tricky to plug into se-cure collaboration tools and save everything neatly.

That's what this paper digs into. We're rolling out a real-time meeting platform powered by AI that brings together decen-tralized peer-to-peer calls, secure cloud authentication, smart chat assistance, and live speech transcription—all in one place. The stack? React.js on the frontend, Node.js and Ex-press on the backend, live updates over Socket.IO, Firebase for user management and data, and the Groq API for light-ning-fast, context-aware chat replies. WebRTC keeps the au-dio and video direct, smooth, and efficient.

The platform's built from modules—each one handles its job, whether that's logging people in, running meetings, setting up WebRTC, chatting, working with AI, or transcribing speech. We use token-based authentication to lock down both the APIs and live Socket.IO channels. Every AI request runs through the backend with tight controls to keep things safe. For transcription, the browser's Speech Recognition API jumps in—turning talk into text, tagging who said what, and saving it all to the cloud.

The bottom line? You can blend decentralized web tech with AI and get meetings that are smarter, faster, and actually help-ful. This isn't some bloated, old-school platform—it stays lean, responds fast, and brings intelligent help into every meeting without the mess. It's built for business calls, online classes, group coding sessions, or just keeping teams in sync. Our tests show it's reliable, handles real-world demands, and sets the stage for the next wave of collaborative tools.

II. RELATED WORK

People have put a lot of effort into building platforms where you can talk and meet in real time, especially with low-la-tency audio and video over the web. WebRTC really changed the game here. It lets browsers handle real-time audio, video, and even data channels out of the box, so you don't need any extra plugins. Researchers have shown that WebRTC helps cut down on lag and scales well—at least when you're not dealing with huge meetings—since it skips the big, central-ized media servers. But here's the catch: most WebRTC sys-tems just handle getting the media from point A to point B. They don't come with smart features, like automatic tran-scription or help that actually understands the meeting con-text.

Older video conferencing tools still lean on centralized set-ups, using things like Selective Forwarding Units (SFUs) or Multipoint Control Units (MCUs) to juggle the media streams. That works if you're running big meetings and need something reliable, but it means higher costs and more lag. Lately, researchers have started mixing peer-to-peer WebRTC with signaling servers. This way, you can strike a balance be-tween scaling up and keeping good performance. Usually, these systems use protocols like WebSocket or Socket.IO to set up sessions, but they don't always take care of secure user authentication or keeping data safe and accessible for the long haul.

Authentication and managing sessions are still tough nuts to crack in these collaborative systems. Some projects use OAuth or their own token systems to lock down real-time communication. More recently, cloud-based services like Firebase Authentication have become popular because they're easy to plug in and scale. Still, most of the time, au-thentication just controls who gets in—it doesn't go further, like securing real-time signaling or keeping AI services locked down throughout the whole meeting.

Lately, there's been a lot of buzz about bringing AI into meet-ings. People are trying out chat assistants powered by large language models to help users out—whether that's answering questions, digging up info on the fly, or assisting with meeting tasks. The problem? Most of these AI features run on central-ized services that clients tap directly, which opens the door to security headaches, rate limits, and the risk of people going overboard. Plus, these AI tools usually feel tacked on, not re-ally woven into how people actually work together during meetings.

Transcription is another piece of the puzzle. Cloud tools like Whisper, Deepgram, and Google Speech-to-Text get high marks for accuracy, but they need to keep sending audio to a server, which slows things down and racks up costs. On the other hand, browser-based speech recognition is faster and lighter, but earlier research often skipped things like figuring out who's speaking or keeping a solid record of transcripts. That makes it harder to get value out of these transcriptions after the meeting wraps up.

The system in focus here takes a different approach. It brings together peer-to-peer WebRTC, secure cloud authentication, real-time signaling, AI chat assistance, and live transcrip-tion—all under one modular roof. By routing AI tasks through a backend that controls the flow and storing chat and transcript data in a real-time database, the system tackles a bunch of the pain points from earlier attempts, covering scala-bility, security, and smarter, in-the-moment help during meet-ings.

III. PROPOSED SYSTEM

This system brings AI and real-time meeting tools together to make virtual teamwork smoother and, honestly, way less an-noying. Think of it as a one-stop platform for quick peer-to-peer meetings, smart chat, and live speech-to-text. It runs on WebRTC for crisp audio and video, uses cloud authentication for security, and handles everything behind the scenes with real-time signaling. The AI steps in to keep conversations on track and helps everyone stay focused.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26may052

Everything's built to be flexible, secure, and easy to upgrade. Each part—media, signaling, authentication, storage, and AI—works separately, so you can swap out or update one without messing up the rest.

It's simple. Users log in through their browser, start or join meetings, and connect instantly. They can stream audio and video, chat in real time, get help from the AI assistant, and see live transcripts as they speak. Messages and transcripts get saved to the cloud, so nothing disappears when the meeting ends.

➢ *System Architecture*

- The system breaks down into five key layers: Client, Authen-tication, Signaling & Backend, Peer-to-Peer Communication, and Data & AI Services.

- Client Layer: This is what users see and use, built with Re-act.js. Here, people log in, manage meetings, chat, and read transcripts. Custom React hooks like useAuth and useMeet-ing handle logins and meeting events, and the browser's Me-diaDevices API takes care of audio and video.

- Authentication Layer: Firebase Authentication covers sign-in—email or third-party accounts, whatever works. Once us-ers log in, they get a JWT token. The backend checks this to-ken to confirm who's logging in and keeps sessions locked down. That same token approves API calls and Socket.IO connections.

- Signaling and Backend Layer: The backend runs on Node.js and Express, with Socket.IO for real-time connections. This layer sets up meetings, connects participants, gets WebRTC connections going (like sharing SDP and ICE candidates), checks user sessions, and manages access to AI features. Every AI request runs through the backend, which logs eve-rything and limits requests so it stays fair.

- Peer-to-Peer Communication Layer: WebRTC handles direct audio and video streams between users. The signaling server just helps users find each other—after that, streams flow straight between people, cutting down on lag and server load.

- Data and AI Services Layer: Chat messages and transcripts go to Firebase Firestore as soon as they're made. Each tran-script marks who said what, and when. For AI chat, the backend sends messages to the Groq API, which reads the conversation and replies.

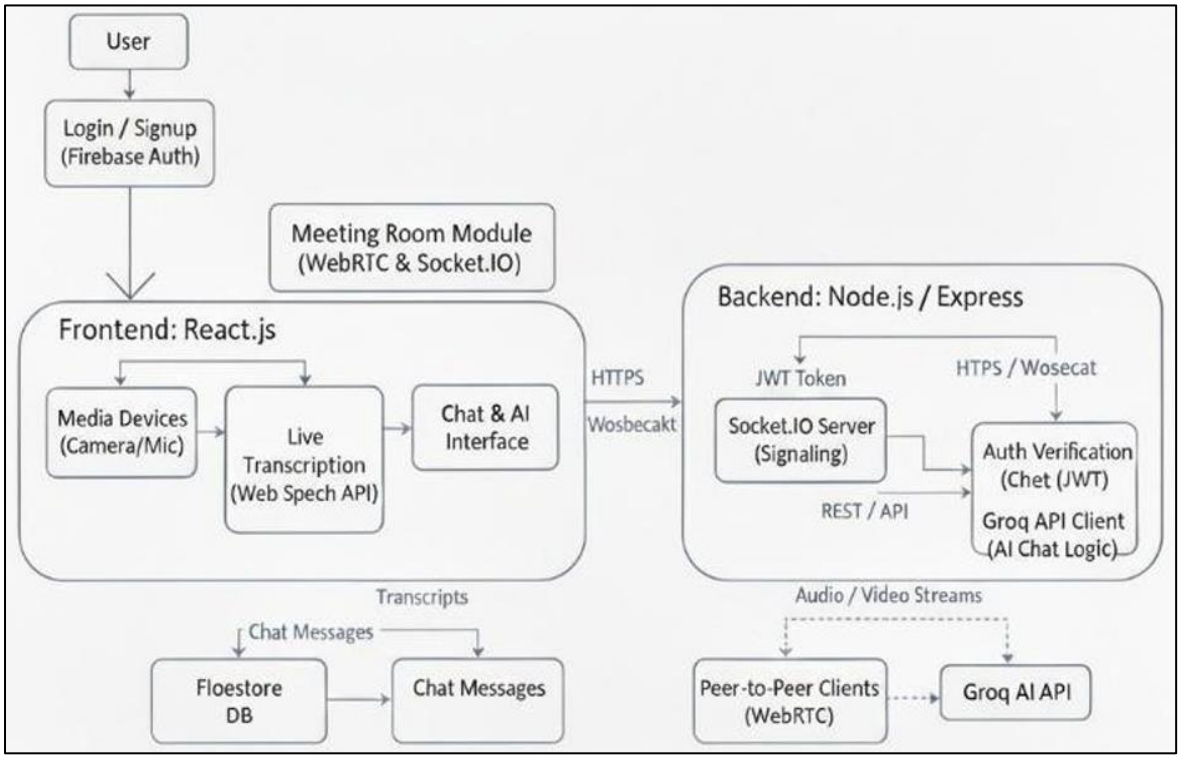

Fig 1 System Architecture

➢ *Algorithm*

It starts simple the app wakes up, checks who you are with Firebase Authentication, and keeps things locked down. Once you're in, you get two choices: start a new meeting or join one that's already rolling. After you pick, the app grabs your cam-era and mic and sets up everything for live chat.Here's where things get interesting. WebRTC handles the actual audio and video calls, making sure you can see and hear everyone in real time. Socket.IO does the behind-the-scenes work, keeping everyone connected and making sure there's barely any delay. The result? Smooth, real-time conversations.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26may052

While you're talking, the app listens and uses the browser's Web Speech API to turn your words into live captions. You can also send real-time chat messages whenever you want. If you throw a question at the AI, it zips over to the backend, checks with the Groq API, and brings back a smart answer. Every chat and transcription gets saved right into Firestore, so you can pull them up later even after the meeting's over.

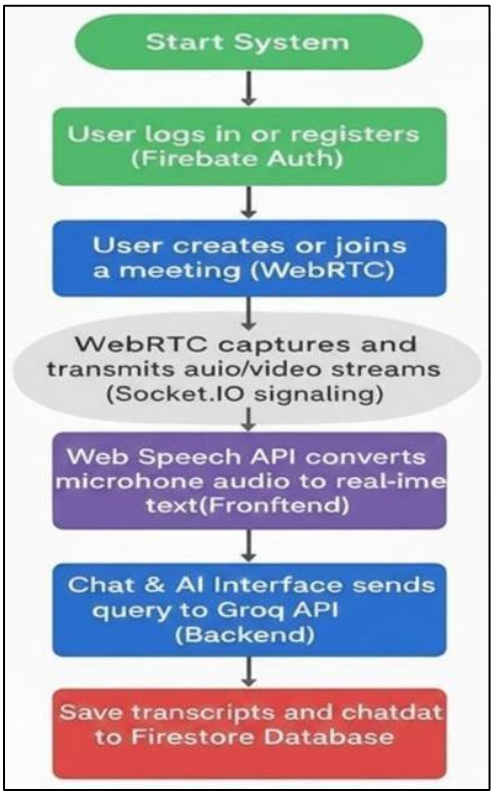

Fig 2 Algorithm

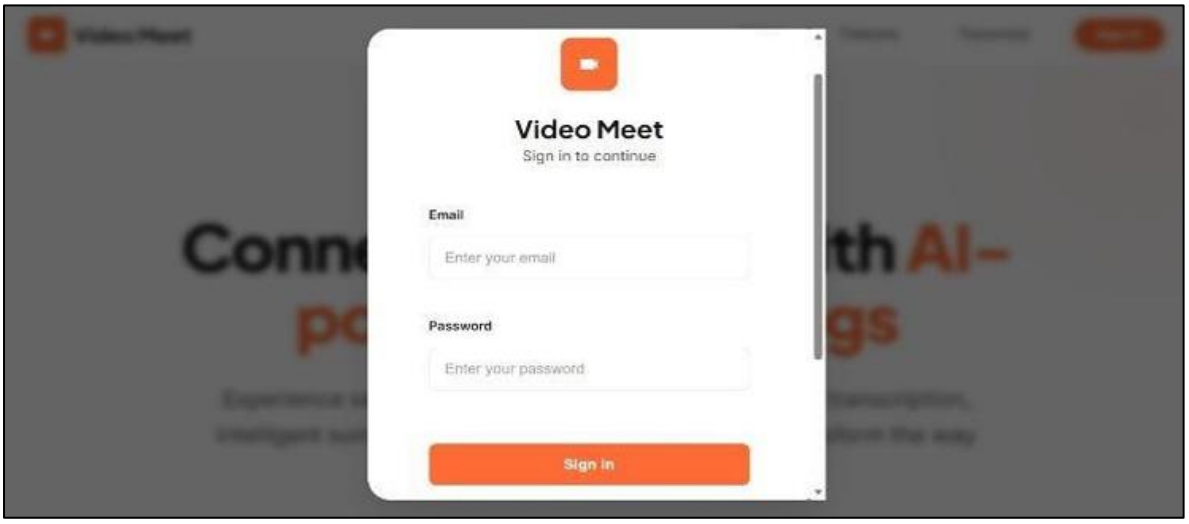

Fig 3 Creating Room

➢ *Modules*

- Authentication Module: Handles sign-up, login, logout, and session checks. Only logged-in users get into meetings—the system always checks JWT tokens before anyone can use the backend or connect with Socket.IO.
- Live Transcription Module: Listens to what users say and turns speech into text in real time with the browser's Speech Recognition API. It tags transcripts with who said what and when, then saves everything in Firestore for later.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26may052

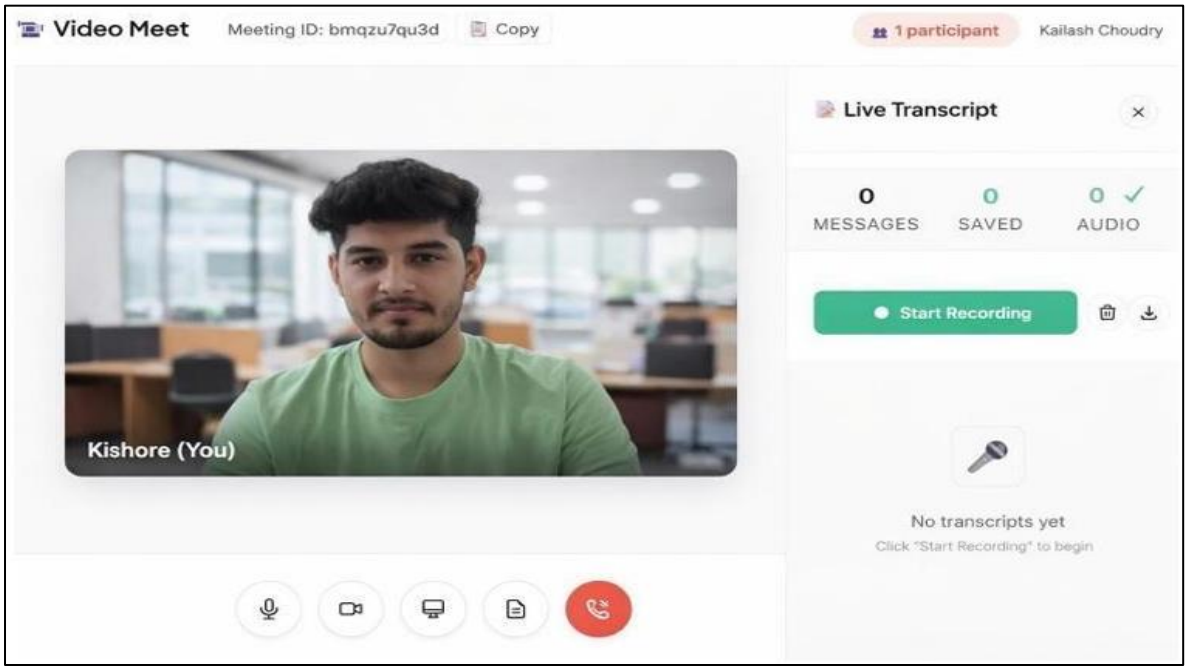

Fig 4 Live Transcription

- AI Interaction Module: Handles questions for the AI assis-tant. It sends messages to the backend, which calls the Groq API for answers. The system limits requests to keep things fair.

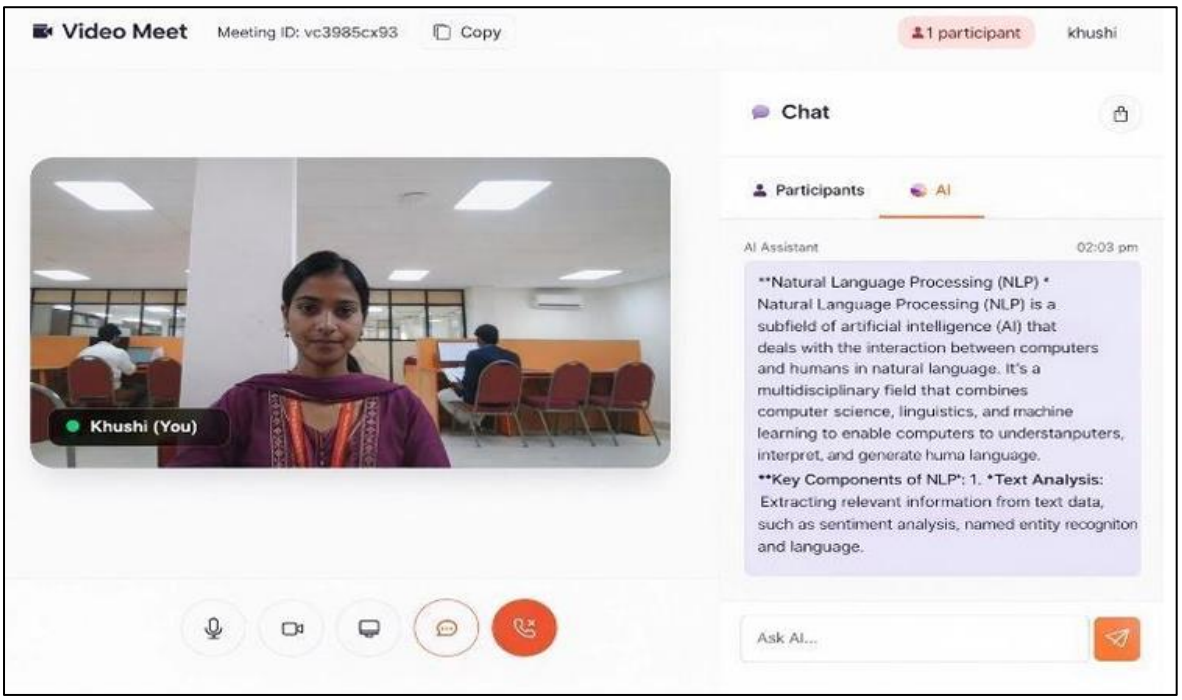

Fig 5 AI Chatbot

IV. RESULTS AND DISCUSSION

We put the new AI-powered meeting system through its paces in a bunch of live scenarios—people joining and leaving, sharing audio and video, chatting, and generating live tran-scriptions. The peer-to-peer setup with WebRTC really kept things snappy. Audio and video lag stayed low, especially in small and medium meetings, since we didn't have to funnel everything through a central server.

For getting people connected, Socket.IO handled the signal-ing side smoothly. Even when a bunch of folks joined at once, sessions kicked off fast with hardly any wait.

On the security front, the system used Firebase Authentica-tion with JWT checks on the backend. This locked down ac-cess right from the start, but without slowing things down when meetings began. Tokens kept the session secure throughout, working reliably for both REST APIs and real-time chat.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26may052

We stored chat messages and live transcriptions in Firebase Firestore. This kept everyone in sync and made it easy to re-view meeting data later. Storing all that didn't slow anything down for users during the meeting.

The AI chat assistant did its job well—quick, context-aware replies came back fast, even when routed through the Groq API on the backend. Rate limits kept things from getting over-loaded during busy moments, so performance stayed solid. Having instant AI help in the chat made the whole meeting feel a lot more interactive, without getting in the way of the main conversation.

Live transcription worked well too. The browser's Speech Recognition API delivered accurate, real-time text as long as the network and sound conditions were decent. Speaker names and timestamps got added automatically, which made the transcripts easy to follow. Sure, browser-based recogni-tion varied a bit depending on which browser or device people used, but since the heavy lifting happened on the client side, we saved a lot of server resources and kept things light and fast.

In short, the system pulled everything together—decentral-ized chat and media, smart AI help, and real-time transcrip-tion—in one flexible package. The modular design means it can scale or add features down the road, like better transcrip-tion engines or meeting summaries, without losing its speed, security, or reliability. All in all, this approach works well for remote teams, online classes, and any group that needs real-time collaboration.

Fig 6 User Interface

- User Interface: The user interface makes it easy to start a new meeting or join an existing one using a secure link. It shows who is currently in the meeting and takes care of starting au-dio and video for each participant. Behind the scenes, it man-ages the connection setup and smoothly handles people join-ing or leaving at any time. It also gives users quick access to chat, AI help, and live transcription during the meeting.

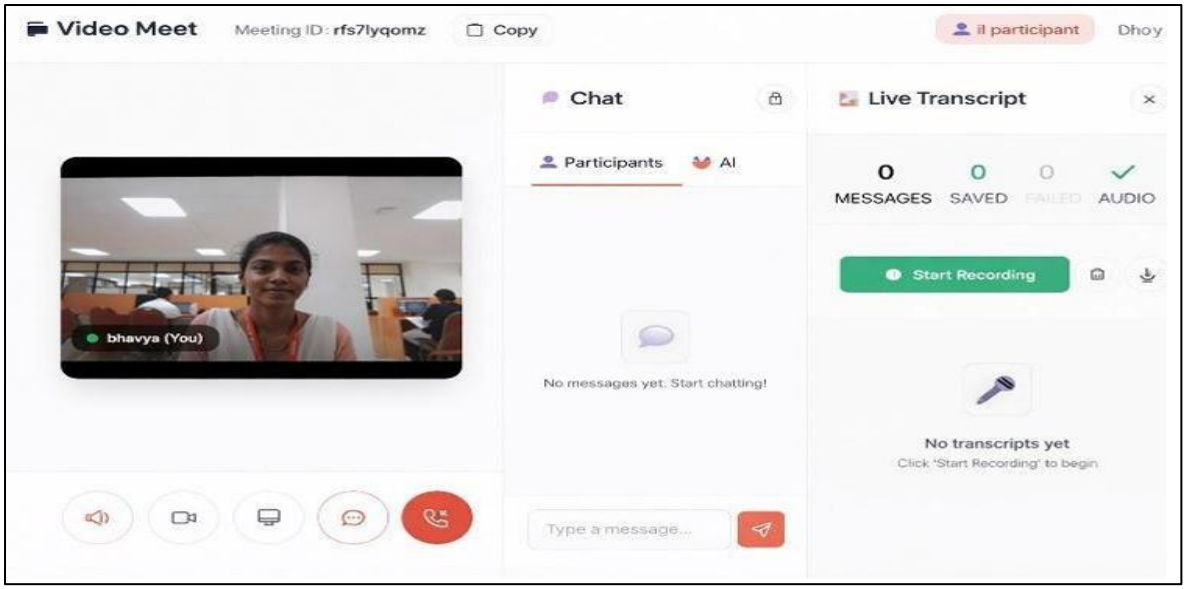

Fig 7 Admin Interface

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26may052

- Admin interface: a web-based video conferencing app where you see everyone's live video, manage who's in the meeting, chat in real time, and get instant transcription and recording—all right in your browser.

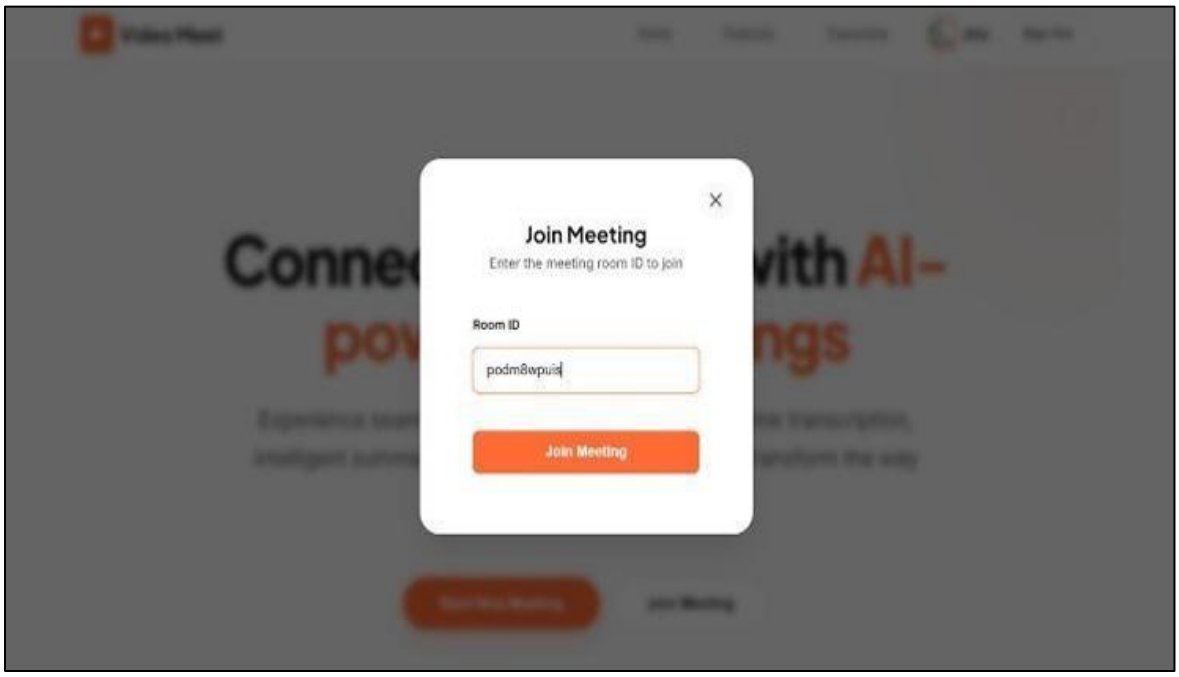

Fig 8 Join Meeting

- Join Meeting: There's also a shot of the secure entry screen. Before anyone gets in, they have to punch in a unique room ID. It's a simple gate, but it keeps random people out. Just another layer of security baked into this AI-powered meeting tool.

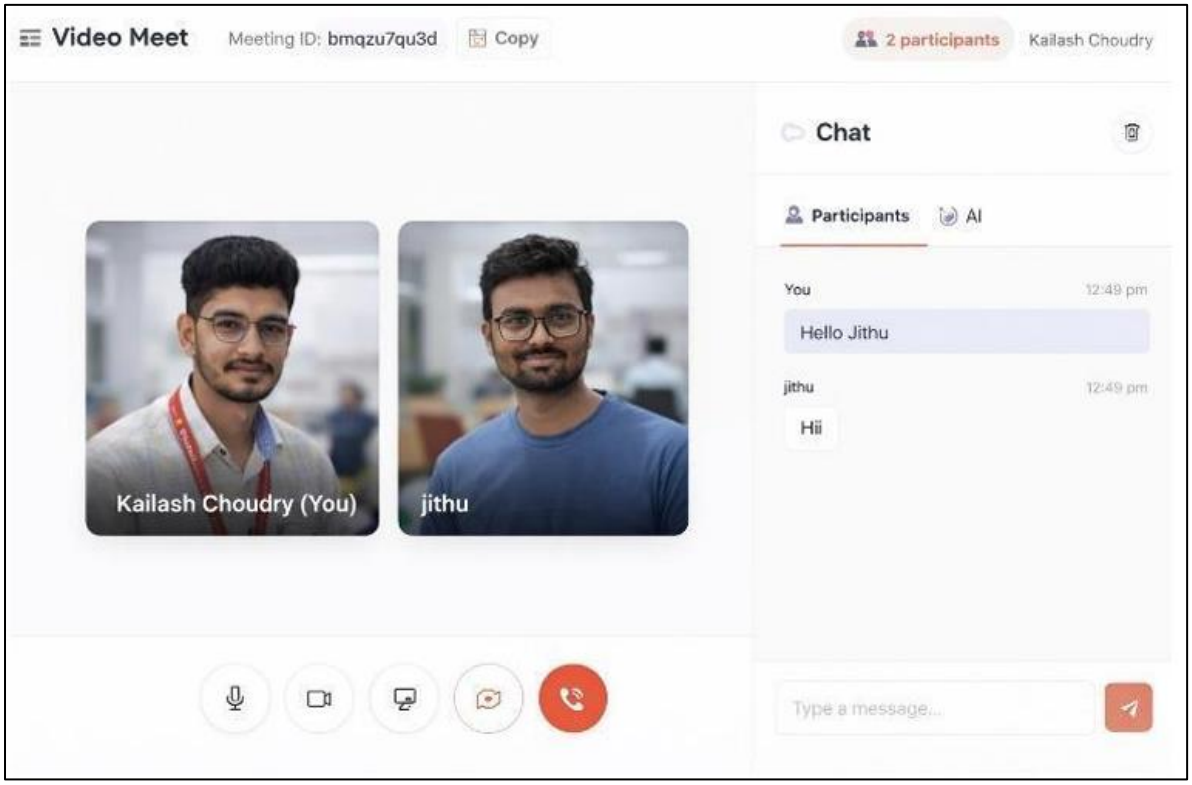

Fig 9 Real-Time Chat Box

- Real-Time Chat box: Lets people send and receive chat mes-sages instantly during meetings. Messages go through Socket.IO and get saved in Firestore to keep chats in sync.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26may052

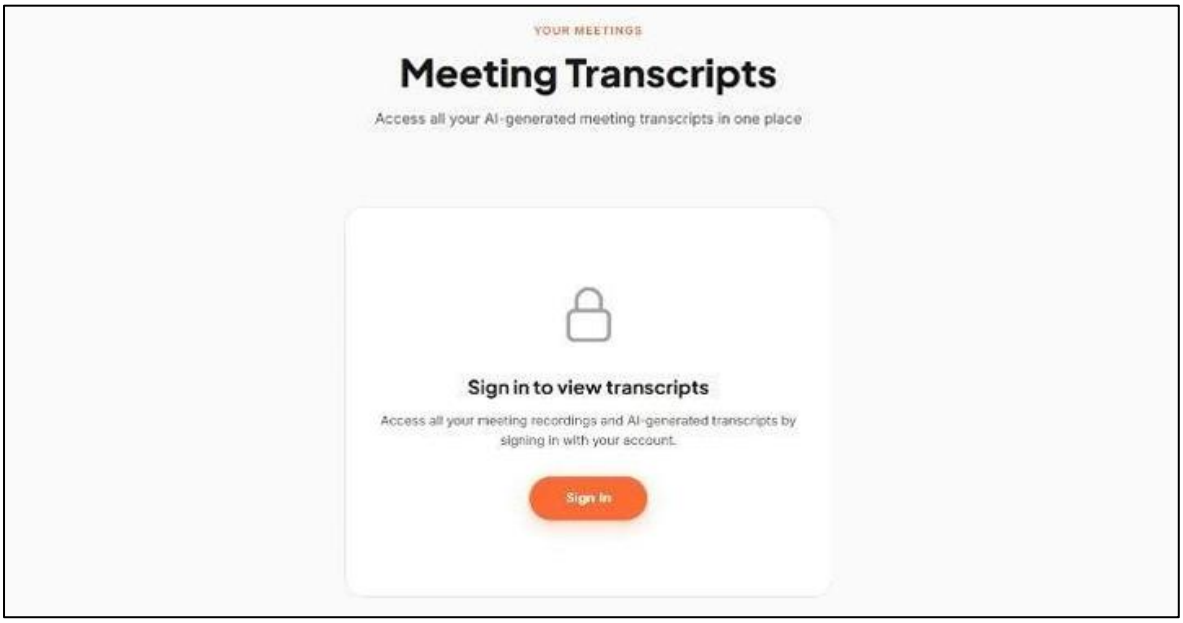

Fig 10 Meeting Transcription

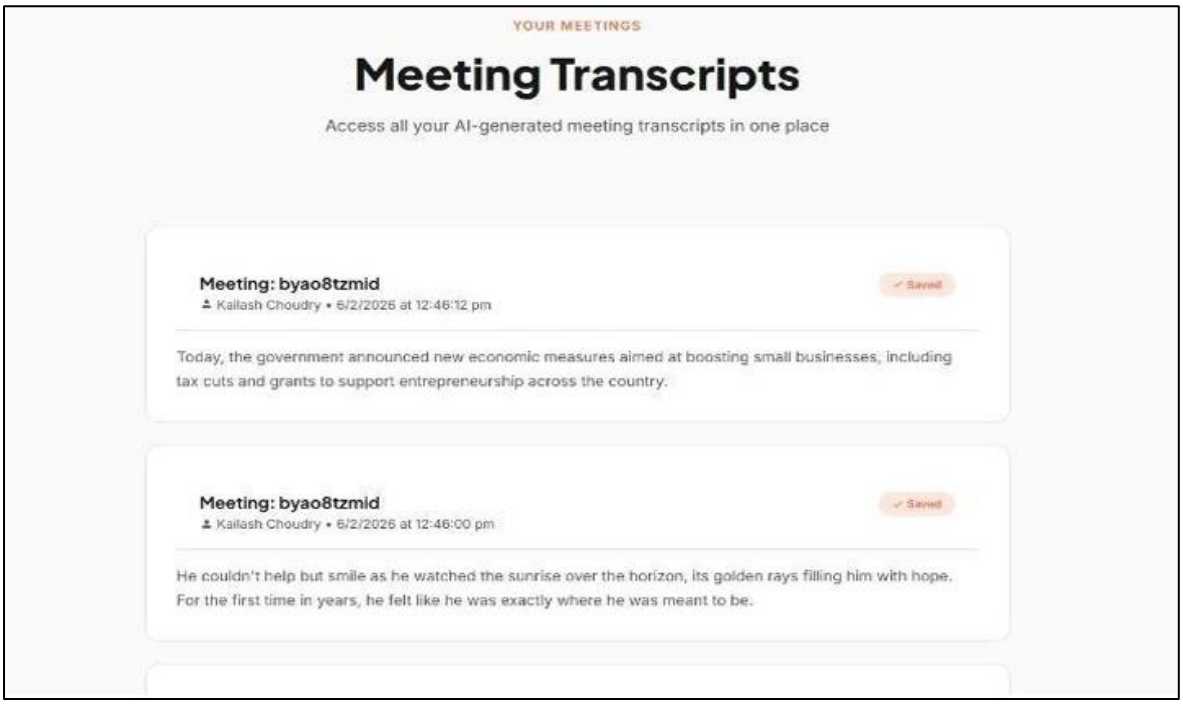

Fig 11 Meeting Transcription

- Meeting Transcription: This screen shows a protected tran-script page where only logged-in users can view meeting transcripts. If a user is not authenticated, access is blocked and they are asked to sign in securely before proceeding.

V. CONCLUSION

This research lays out a real-time meeting platform powered by AI, blending decentralized peer-to-peer communication, smart AI-driven features, and live speech transcription—all in one easy-to-use web system. It steps up to meet the real de-mand for fast, intelligent, and scalable tools, whether you're working remotely, learning online, or managing a team scat-tered across the globe. Modern web tech and cloud-native ser-vices make it all possible.

One of the big wins here is using WebRTC for peer-to-peer audio and video streaming. That choice cuts out a lot of server lag and avoids the bottlenecks you get with traditional, cen-tralized meeting setups. For signaling, Socket.IO keeps things lightweight and scales well as more users join. Security isn't an afterthought, either—Firebase Authentication with JWT checks on the backend locks things down without slowing you down.

Adding an AI-driven chat module takes collaboration up a notch, giving people smart, real-time replies by tapping

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26may052

into large language model APIs on the backend. Live speech tran-scription, handled right in the browser with the Speech Recognition API, makes meetings more accessible and keeps records without piling extra work on servers. Sure, browser-based transcription can struggle with heavy accents or lots of background noise, but for most real-time meetings, it does the job well.

The modular design means it's easy to maintain and ready for whatever comes next—think hybrid WebRTC setups, deeper AI-driven meeting insights, multilingual transcription, or au-tomatic summaries. Tests showed the platform holds up under real meeting pressure, delivering stable performance. All in all, bringing together decentralized communication, smart AI features, and live transcription really upgrades the virtual meeting experience and sets a solid base for future AI-pow-ered collaboration tools.

REFERENCES

[1]. A systematic review on WebRTC for potential applica-tions and challenges beyond audio video streaming — H. Mahmoud & R. Abozariba, Multimedia Tools & Applica-tions, Feb 2025. DOI: 10.1007/s11042-024-20448-9.

[2]. GK Shwetha, PS Shetty, SN Baliga, JKA Rathod, C Divya, "Transforming Education with Innovative Virtual Learning Environments", IEEE Conference Publication, Feb-ruary 2025, DOI: 10.1109/ICRASET63057.2024.10894905

[3]. Vuyisa Baza, Nomusa Dlodlo, Alfredo Terzoli, "Building a Peer-to-Peer Learning Platform for University Students Us-ing WebRTC and Mobile Technology", IEEE Conference Publication, April 2025, DOI:10.1109/ZCICT63770.2024.1095830.

[4]. Real Time Speech-to-Text on Edge: A Prototype System for Ultra-Low Latency Communication with AI-Powered NLP — S. di Leo, L. De Cicco, S. Mascolo; Information, 2025. DOI: 10.3390/info16080685.

[5]. S. Di Leo, L. De Cicco, and S. Mascolo, "Edge-Based Real-Time Speech Recognition for Low-Latency Collabora-tive Applications," IEEE Transactions on Network and Ser-vice Management, vol. 21, no. 2, pp. 1345–1357, Apr. 2024, doi: 10.1109/TNSM.2024.3361028

[6]. H. Mahmoud and R. Abozariba, "Challenges and Oppor-tunities of WebRTC-Based Collaboration Platforms Beyond Video Conferencing," IEEE MultiMedia, vol. 31, no. 1, pp. 64–73, Jan.–Mar. 2024, doi: 10.1109/MMUL.2024.3342197.

[7]. Javlon Tursunov, Gregor Rozinaj, Vivek Dwivedi, Ivan Minárik, "A Customizable WebRTC-based Video Conferenc-ing System For Real-time Communication", IEEE Confer-ence Publication, 19 August 2024, DOI: 10.1109/IWS-SIP62407.2024.10634026

[8]. WebRTC over 5G: A Study of Remote Collaboration QoS in Mobile Environment — Journal of Network and Systems Management, 2024 (article number). DOI available via Springer.

[9]. J. Park and K. Lee, "AI-Assisted Real-Time Collaboration Systems with Context-Aware Conversational Agents," in Proc. IEEE Int. Conf. on Artificial Intelligence and Virtual Reality (AIVR), 2024, pp. 98–105, doi: 10.1109/AIVR60935.2024.00022.

[10]. L. De Cicco, S. Mascolo, and V. Palmisano, "Perfor-mance Analysis of WebRTC Signaling Using WebSockets and Socket.IO," IEEE Communications Letters, vol. 27, no. 5, pp. 1290–1294, May 2023, doi: 10.1109/LCOMM.2023.3257782.

[11]. P. Singh and R. Kumar, "AI-Driven Chatbots for Real-Time Collaborative Applications: Design and Performance Evaluation," IEEE Internet Computing, vol. 27, no. 2, pp. 45–53, Mar.–Apr. 2023, doi: 10.1109/MIC.2023.3241187.

[12]. Y. Zhang, X. Li, and H. Wang, "Real-Time Speech-to-Text Transcription Using Browser-Based Speech Recognition APIs," in Proc. IEEE Int. Conf. on Web Intelligence (WI), Thessaloniki, Greece, 2023, pp. 212–219, doi: 10.1109/WI57407.2023.00041.

[13]. A. R. Khan and S. Mehta, "Cloud-Integrated Authentica-tion and Authorization for Real-Time Web Applications Us-ing JWT," IEEE Access, vol. 11, pp. 45678–45690, 2023, doi: 10.1109/ACCESS.2023.3290041.

[14]. Safiqul Islam, Michael Welzl, Tobias Fladby, "Real-Life Implementation and Evaluation of Coupled Congestion Con-trol for WebRTC Media and Data Flows", IEEE Conference Publication, September 2022, DOI: 10.1109/AC- CESS.2022.3206041

[15]. S. Petrangeli, J. De Cock, and R. Van de Walle, "Design and Evaluation of Low-Latency WebRTC-Based Real-Time Communication Systems," IEEE Access, vol. 10, pp. 112345–112358, 2022, doi: 10.1109/AC-CESS.2022.3187465.

[16]. M. Al-Shabi, A. Al-Dweik, and M. Al-Masri, "Secure WebRTC Architecture for Real-Time Multimedia Collabora-tion Applications," in Proc. IEEE Int. Conf. on Communica-tions (ICC), Seoul, South Korea, 2022, pp. 1–6, doi: 10.1109/ICC45855.2022.9838894.

[17]. Jovana Marašević, Ana Gavrovska, "Virtual Reality and WebRTC implementation for Web educational application de-velopment", IEEE Conference Publication, January 2021, DOI: 10.1109/TELFOR51502.2020.9306513

[18]. Julius Flohr, Erwin P. Rathgeb, "Reducing End-to-End Delays in WebRTC using the FSE-NAlgorithm for SCReAM Congestion Control", IEEE Conference Publication, March 2021, DOI: 10.1109/CCNC49032.2021.9369574

[19]. Soft Real-Time Communication with WebSocket and WebRTC Protocols Performance Analysis for Web-based Control Loops — IEEE Conference Publication (via IEEE Xplore) DOI: 10.1109/8864680.