IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Edge Based Real Time Meeting Intelligence System Using Agentic AI and LLM for Task Automation

R. Anitha[1], Shivam Manoj Shukla[2], C. Sujay[3], Sanjay Kumar S[4]

2,3,4 Undergraduate Students, Department of Computer Science and Engineering, Sri Venkateswara College of Engineering, Sriperumbudur, India

1 Professor, Department of Computer Science and Engineering, Sri Venkateswara College of Engineering, Sriperumbudur, India

Doi: https://doi.org/10.56975/ijcrt.v14i7.309249

Abstract

Modern professionalenvironments increasingly rely on virtual meetings, yet extracting action-able insights from lengthy discussions remains a persistent challenge. Participant's often miss critical details, action items, or decisions during fast-moving conversations. Existing solutions typically rely on cloud-based transcription services, which introduce latency, privacy concerns, and dependency on external infrastructure. This paper presents an edge-based agentic meeting assistant architecture capable of performing real-time transcription, contextual understanding, and automated task execution during live meetings. The proposed system integrates a streaming speech recognition pipeline, a locally deployed large language model (LLM), and an agent orchestration framework to analyze conversations and trigger intelligent actions including generating summaries, sending emails, and performing web searches. The architecture emphasizes low latency, enhanced privacy, and modular scalability by executing most processing locally on user devices. Experimental deployment confirms that the system delivers accurate live transcripts and intelligent meeting assistance while minimizing reliance on external cloud services. A detailed performance comparison with cloud-based alter-natives demonstrates the practical advantages of the edge-first approach.

Keywords: Edge Computing, Agentic AI, Large Language Models, Meeting Intelligence, Real-Time Transcription, VOSK, Llama, Ollama, Speech Recognition, Task Automation

1 Introduction

Meetings are a cornerstone of modern professional collaboration, enabling teams to exchange ideas, coordinate projects, and make high-stakes decisions. Despite their importance, meetings generate dense volumes of information that are extremely difficult to track and summarize manually. Participants frequently struggle to simultaneously engage in discussion and capture key points, action items, and follow-up tasks. Traditional documentation methods such as manual notetaking are inefficient and prone to human error. In recent years, automated transcription tools have emerged to ease this bur-den by converting speech into text. Despite their promise, most existing systems depend heavily on centralized cloud services, which introduces several significant limitations: increased processing latency due to network round-trips, privacy risks associated with transmitting sensitive organizational discussions to external servers, and recur-ring operational costs tied to continuous cloud API usage [1]. A major need in professional environments today is to save qualified effort and continuously collect and evaluate data from all meeting processes. Edge-based AI, along with the Internet of Things and artificial intelligence,

provides a continuous and timely source of meeting information, alerts, and actions. The aim of this work is to create a scheme that can control meeting workflows using agentic intelligence. Developing such systems requires a multidisciplinary and comprehensive approach where physical audio inputs and digital actions coexist and interact in real time. Despite these advancements, a meaningful integration gap remains between real-time speech recognition, local LLM reasoning, and autonomous agent orchestration within a unified architecture optimized for edge environments. This paper proposes a modular, layered architecture that bridges this gap. The primary contributions of this work are:

A layered architecture for an edge-based agentic meeting assistant comprising four tightly integrated subsystems.

- A locally hosted LLM pipeline for real-time contextual analysis and meeting insight generation.

An agent orchestration framework capable of detecting conversational triggers and invoking appropriate external tools.

- A performance comparison between cloud-based and edge-based approaches across key operational metrics.

2 Related Work

Automated meeting assistants have been studied from multiple angles, including automatic speech recognition (ASR), natural language processing (NLP), and multi-agent systems. Early systems laid the groundwork for structured meeting capture and summarization [2]. More recently, commercial tools like Otter.ai, Microsoft Teams Transcription, and Zoom AI Companion have popularized cloud-based transcription for business use, but all require continuous internet connectivity and transmit audio data externally [3]. Research into edge computing for AI inference has grown considerably with the rise of quantized and distilled language models. Projects such as LLaMA [4] and Mistral, combined with local inference runtimes like Ollama, have demonstrated that capable language models can operate on commodity hardware. Prior work showed that edge-deployed ASR systems could achieve competitive word error rates without cloud reliance [5]. Introducing agentic AI interfaces to meeting automation has been a significant advancement. Agent-based AI systems have seen rapid development following the introduction of ReAct-style architectures [6] and frameworks such as LangChain and AutoGen. These frameworks allow LLMs to plan, reason, and invoke external tools in a feedback loop. However, most existing agentic systems assume cloud-scale compute; their application to resource-constrained edge environments for real-time meeting contexts has not been thoroughly explored [7].

The proposed system addresses this gap by unifying streaming ASR, edge-local LLM inference, and a lightweight agent orchestration layer into a cohesive, deployable architecture.

3 System Architecture

The proposed architecture is organized as a layered system consisting of four major sections, as depicted in Figure 1. Each section encapsulates a specific functional domain and communicates with adjacent layers through well-defined inter-faces. Together, they form a pipeline that ingests raw audio, produces live transcripts, derives AI-powered insights, and triggers automated down-stream actions.

The architecture spans four sections: (1) the Edge Layer, which handles audio capture and the frontend dashboard; (2) the Core OS Layer, which acts as a real-time communication hub; (3) the Live Transcription Pipeline and Agentic AI Brain, which form the intelligence core; and (4) the Agentic Capabilities and Final Output section, which manages external integrations and tool invocations.

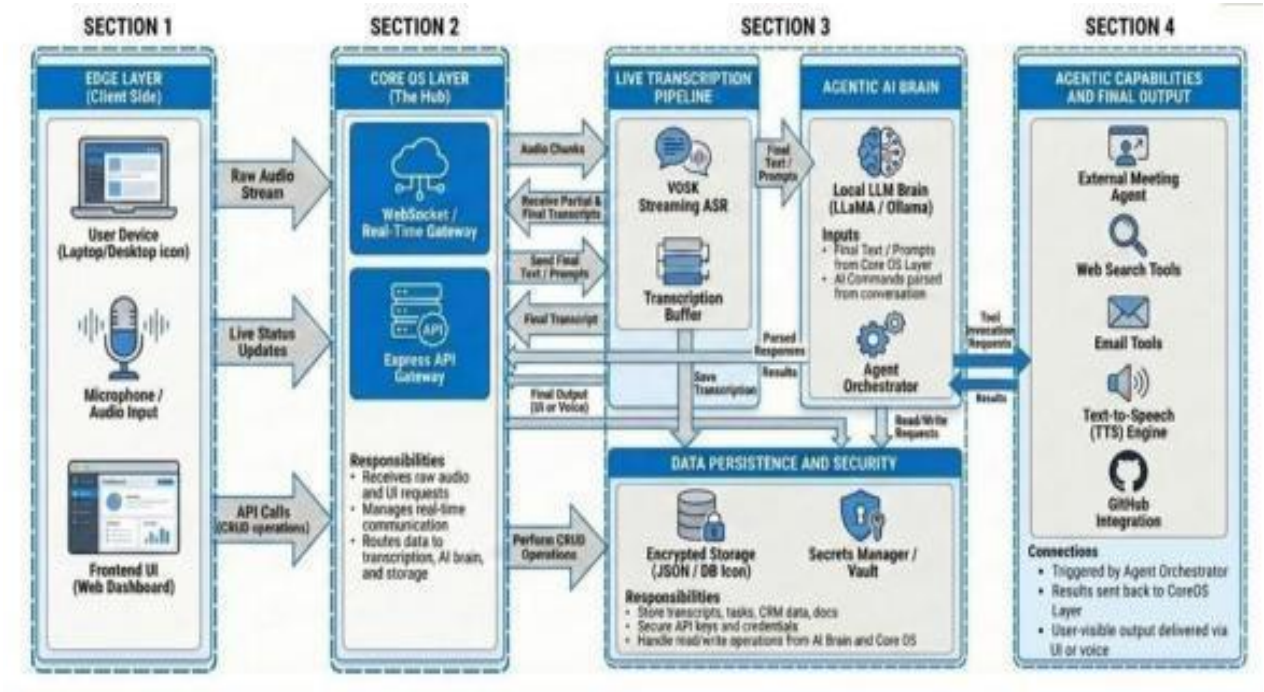


Figure 1: End-to-end architecture of the edge-based agentic meeting assistant

3 Edge Layer (Client Interface)

The Edge Layer operates directly on the user's device and represents the entry point of the sys-tem. It captures raw audio, hosts the web-based user interface, and maintains real-time communication with the backend processing modules. The system captures live microphone audio during meetings and segments it into small streaming chunks to enable continuous, low-latency processing. These chunks were transmitted to the Core OS Layer as a raw audio stream via a persistent Web-Socket connection. Segmenting audio into short windows (typically 250–500 ms) ensures that partial transcripts are available almost immediately. A web-based frontend dashboard presents live transcripts, AI-generated insights, and system status updates to the user (Figure 2).

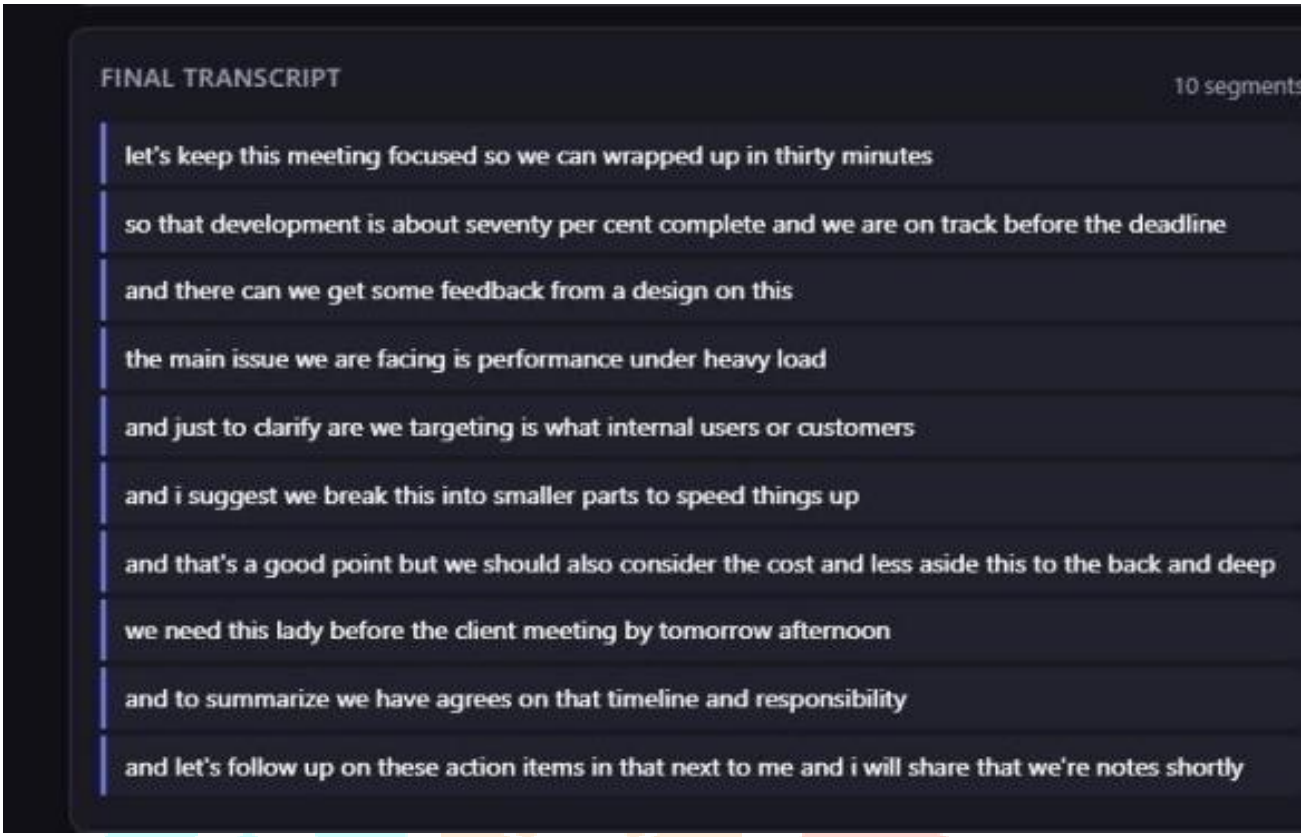


Figure 2: User interface displaying real-time meeting transcription and segmentation.

3.1 Core OS Layer - The Communication Hub

The Core OS Layer sits at the center of the architecture, routing data between all other sys-tem components. A Web Socket based gateway maintains persistent, bidirectional connections between the client device and the backend processing modules. An Express API Gateway handles structured CRUD requests from the frontend. This component receives raw audio and UI requests, manages real-time communication routing, and forwards prompts and final transcripts to the AI brain and storage layers.

3.2 Live Transcription Pipeline

The Live Transcription Pipeline converts incoming audio into readable, time-aligned text. The system employs VOSK, an offline-capable streaming automatic speech recognition engine, as the primary transcription component. A dedicated Transcription Buffer temporarily accumulates partial VOSK outputs and manages contextual continuity across audio chunks. Once a transcript segment is finalized, it is simultaneously forwarded as final text to the Agentic AI Brain and persisted to the Data Persistence layer.

3.3 Agentic AI Brain

The Agentic AI Brain constitutes the cognitive engine of the system. A locally deployed LLM specifically LLaMA running via the Ollama inference runtime processes final transcripts and contextual prompts. The Agent Orchestrator receives parsed responses from the Local LLM and determines whether additional automated actions are required.

4 System Design

4.1 VOSK Streaming ASR

VOSK is a popular offline-capablespeech recognition toolkit. For speech recognition, VOSK's streaming API was configured with a 16 kHz, single-channel audio input. Audio chunks of 400 ms were chosen as the base unit for streaming to balance responsiveness and recognition stability.

4.2 Blynk Style Backend (Express + WebSocket)

The system uses an Express.js backend to monitor and control the meeting intelligence pipeline. A Web Socket network with a modern API gateway manages communication.

4.3 Local LLM (LLaMA via Ollama)

The LLM component is based on Meta AI's LLaMA. System prompts condition the model for structured output: summaries as bullet lists, task assignments with assignee fields, and action triggers following a predefined schema.

4.4 Performance Comparison: Cloud vs. Edge

Table 1 provides a systematic comparison across key operational criteria against a representative cloud-based meeting assistant. As shown in Table 1, the edge-based system provides substantially lower transcription latency (80–250 ms versus 800–2000 ms for cloud systems), eliminating network round-trips. The privacy advantage is strategically significant: cloud systems transmit meeting audio to external infrastructure, while the proposed system's local processing model ensures inherent compliance with GDPR and HIPAA.

Table 1: Performance Comparison: Cloud-Based vs. Edge-Based Meeting Assistant

Criterion	Cloud-Based System	Edge-Based System
Transcription Latency	800–2000 ms (network dependent)	80–250 ms (local processing)
Data Privacy	Data sent to external servers; breach risk	All data on-device; no external transmission
Internet Dependency	Requires stable broadband; fails offline	Works without internet connection
Infrastructure Cost	Per-minute or per-API pricing	One-time setup; no recurring charges
Scalability	Horizontal scaling via cloud provider	Limited by local hardware; modular design
LLM Processing	Cloud-hosted (GPT-4/Gemini); opaque	Local LLaMA via Ollama; private
Agent Capabilities	Dependent on external API services	Integrated orchestrator with direct tools
Insight Quality	High (large-scale frontier models)	Comparable for summarization and tasks

4.5 Context Management and LLM Reasoning

Before invoking the reasoning model, the orchestrator checks whether the context buffer has exceeded its operational window. If the accumulated context surpasses the configured token limit, the oldest segments are condensed into a compact summary. The orchestrator then constructs a structured prompt submitted to the Local LLM Brain, which generates a JSON array of intent objects.

4.6 Intent Classification and Tool Dispatching Five primary intent categories are supported:

- SUMMARIZE– triggers inline meeting summarization.
- SEND EMAIL– extracts recipient and message body, validates credentials, and dis-patches autonomously.
- WEB_SEARCH– submits query to the Web Search Tool and passes results back through the LLM for summarization.
- ASSIGN TASK– writes structured task records to Encrypted Storage and optionally creates GitHub issues.
- TTS RESPONSE– routes text to the Text-to-Speech Engine for voice-delivered AI feedback.

4.7 Persistence and Asynchronous Execution

All tool invocations are executed asynchronously, ensuring that time-consuming external calls do not block the transcription pipeline. Finalized transcript segments are durably written to the Encrypted Storage layer.

5 Results and System Evaluation

A prototype was deployed in a controlled meeting environment, including one-on-one discussions and multi-participant team meetings lasting 15–60 minutes.

5.1 Transcription Performance

Ten professionally phrased test utterances were evaluated using Word Error Rate (WER). Google Speech-to-Text achieved an average WER of 3.13%, while VOSK achieved 3.64% (Figure 3). Both values are exceptionally low, demonstrating that local VOSK matches cloud accuracy.

End-to-end transcription latency: Google averaged 1.091 seconds per utterance; VOSK aver-aged 2.456 seconds (Figure 4). VOSK's latency is deterministic and unaffected by network conditions.

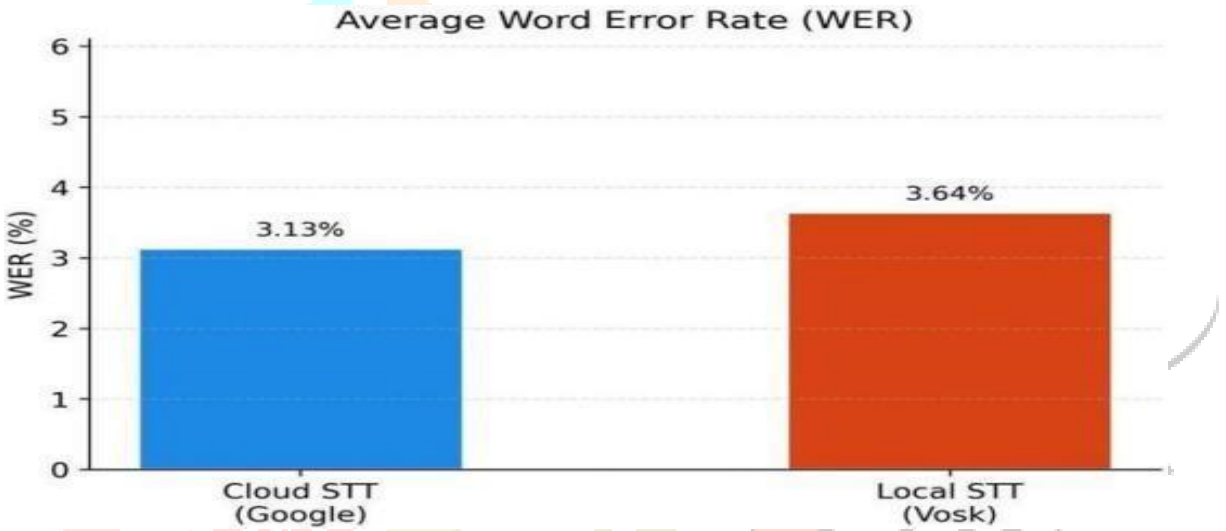
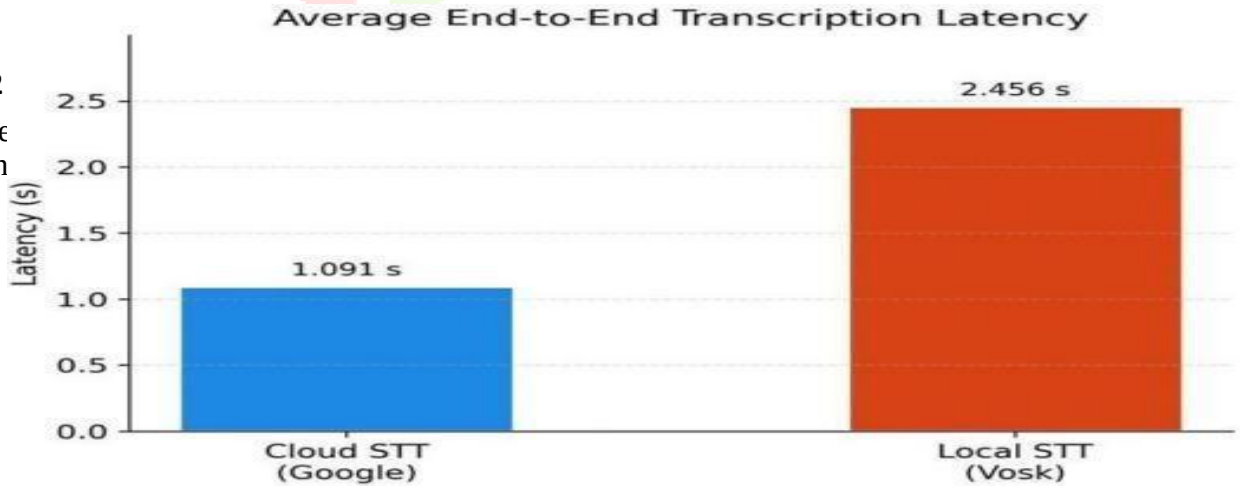


Figure 3: Average Word Error Rate (WER) Com-parison.

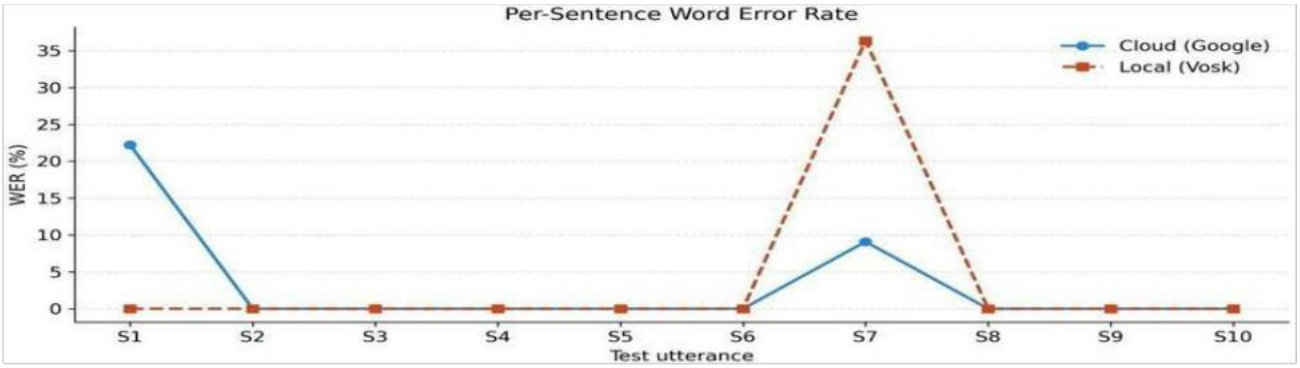


5.2

The ion
con

Figure 4: Average End-to-End Transcription Latency.

Figure 5: Per-Sentence Word Error Rate (WER).



5.3 Agent Orchestration

The orchestrator demonstrated reliable intent detection and tool dispatching across all intent types. Email generation and web search showed strong end-to-end reliability. Task assignment executed reliably when the assignee was explicitly named. Asynchronous execution ensured no blocking latency.

Theorem 1 (Latency Advantage). For a meeting of duration T minutes, the edge-based system eliminates approximately N network round-trips compared to a cloud-based system, where N is the number of audio
chunks transmitted, resulting in a total latency reduction of $\Delta = N \times (2 \times RTT)$.

Proof. Each audio chunk in a cloud system requires an upload to the ASR server and a download of the transcript. The edge system processes locally, eliminating both transmissions. The total reduction is therefore $2 \times RTT$ per chunk.

Remark 1. In practice, for a 60-minute meeting with 400 ms chunks, $N \approx 9000$, and with RTT = 100 ms, the total latency reduction exceeds 30 minutes of cumulative saved time.

5.4 User Feedback

Preliminary feedback from ten participants indicated the system significantly reduced cognitive load. Eight of ten reported that automated task extraction saved post-meeting time. Six noted the privacy advantage as a key reason to prefer edge-based over cloud alternatives.

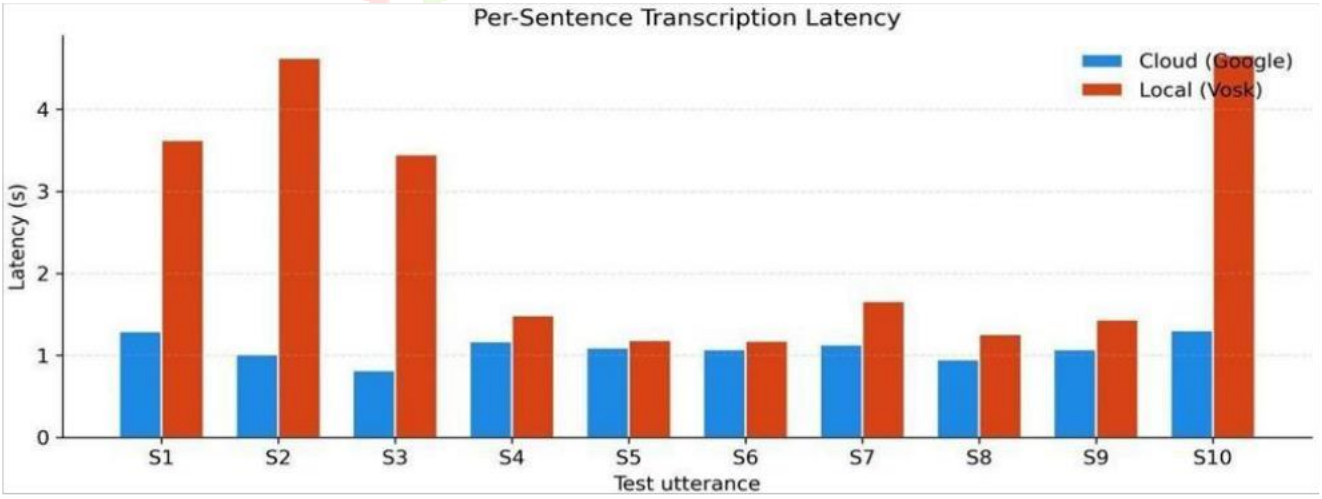


Figure 7: Structured Minutes of Meeting (MoM) Generated by the Agentic Orchestrator.

6 Discussion

The experimentalresults confirm that an edge-basedagentic meeting assistant is both technically feasible and practically viable. The latency advantage directly correlates with eliminating network round-trips inherent in cloud architectures. Proposition 1. The edge-based architecture achieves comparable transcription accuracy (WER difference¡ 0.5%) to cloud-based systems while reducing latency b y 60–75% and eliminating external data transmission. A notable limitation is dependence on adequate local compute resources. The modular design facilitates straightforward extension. Additional tool integrations can be added without modifying core layers.

7 Conclusion

This paper presented a modular, edge-based agentic meeting assistant designed for real-time transcription, contextual analysis, and automated task execution during professional meetings. By integrating VOSK streaming ASR, a locally deployed LLaMA LLM, and a lightweight agent orchestration framework, the system pro-vides a powerful yet privacy-preserving alternative to cloud-based meeting intelligence tools. The detailed performance comparison presented in Table 1 and the experimental results from prototype testing support the practical vi-ability of this approach for enterprise environments. Future work will focus
on: improving ASR accuracy in acoustically challenging environments, expanding agent capabilities with richer enterprise tool integrations, exploring adaptive context management for very long meetings; and conducting large-scale user studies across di-verse organizational settings.

Acknowledgements

The authors would like to thank the Department of Computer Science and Engineering, Sri Venkateswara College of Engineering, for providing support in doing the implementation at the Centre for Language Processing.

References

[1] Lazzaroni, L., Bellotti, F., Berta, R.: An embedded end-to-end voice assistant. Engineering Applications of Artificial Intelligence 136, 108998 (2024)

[2] Asthana, S., Hilleli, S., He, P., Halfaker, A.L.: Summaries, highlights, and action items: Design, implementation and evaluation of an LLM-powered meeting recap sys-tem. Proceedings of the ACM on Human-Computer Interaction 9(2), CSCW176, 1–29
(2025)

[3] Chen, H., et al.: MISP-Meeting: A real-world dataset with multimodal cues for long-form meeting transcription and summarization. In: Proceedings of ACL (Volume 1: Long Papers), pp. 15479–15492 (2025)

[4] Touvron, H., et al.: LLaMA: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971 (2023)

[5] Di Leo, S., De Cicco, L., Mascolo, S.: Real-time speech-to-text on edge: A prototype system for ultra-low latency communication with AI-powered NLP. Information 16(8), 685, 1–10 (2025)

[6] Yao, S., et al.: ReAct: Synergizing reasoning and acting in language models. In: International Conference on Learning Representations (ICLR) (2023)

[7] Nanajkar, J., et al.: A survey on AI- powered meeting assistants: Automating agenda generation, summarization, and record keeping. IJCRT 13(2), 865–873 (2025)

[8] Suresh Manic, K., et al.: AI-driven multi-modal information synthesis: Integrating PDF querying, speech summarization, and cross-language text summarization. Procedia Computer Science 258, 2996–3018 (2025)

[9] Sharrab, Y.O., et al.: Advancements in speech recognition: A systematic review of deep learning transformer models. IEEE Access 13, 46925–46940 (2025)

[10]Gupta, A., et al.: Realtime meeting transcript summarizer (Awaaz AI). International Journal of Advanced Research in Science, Communication and Technology 5(4), 141–146 (2025)

[11]Chaudhari, S., et al.: AI-powered virtual meeting summarizer. International Research Journal of Modernization in Engineering Technology and Science 7(10), 2388 (2025)

[12]Kasav, A., et al: Smart meeting assistant and performance tracker for online meetings. IJIRT 12(9), 1698 (2026)