RESEARCH ARTICLE

MinuteMaster: An AI-Powered Meeting Transcription and Scheduling System

Barnali Gupta Banik[1,*], **Pampari Yukitha**[1], **Charan Sai Venna**[1] **and Puppala Mangala Tulasi**[1]

[1] Mahatma Gandhi Institute of Technology, Gandipet, Hyderabad, Telangana 500075, India

Abstract

Taking notes during meetings sounds easy, but in reality, people often miss key points—especially when multiple people are talking. Managing follow-ups in separate apps only adds to the hassle. *MinuteMaster* was built to simplify this. It's a web app that brings transcription, speaker identification, summarization, and scheduling into one place. It uses Whisper for multilingual speech-to-text, *pyannote.audio* to identify who's speaking, and BART to turn long transcripts into clear, short summaries. A built-in calendar helps users manage meetings without switching apps. We tested it on 30 recordings across English, Hindi, Telugu, Tamil, and mixed languages. Transcription accuracy ranged from 76% to 88%, and summaries performed better than *TextRank* with a ROUGE-1 score of 0.48. In a study with 42 users, it received an average rating of 4.25/5. The system runs on Flask with MongoDB, making it simple to deploy and scale.

Keywords: AI transcription, BART summarization, meeting automation, natural language processing, speaker diarization.

1 Introduction

Almost every team has regular meetings, but the details often slip through the cracks—what was decided, who's responsible, and where the records are stored. Manual note-taking doesn't help much. It pulls attention away from the discussion, misses key points, and often ends up incomplete. In India, where people switch between languages mid-conversation, it becomes even more challenging. Existing tools like Otter.ai or Teams Recap help to some extent, but they fall short—especially with Indian languages—and still require switching between apps for transcripts, summaries, and scheduling. We built MinuteMaster to address this issue. It's a simple web app where users can record or upload audio, get transcripts with speaker labels, view clear summaries, and manage meetings—all in one place. It supports multiple languages and keeps everything organized. The rest of the paper covers related work, system design, results, comparisons, and future improvements.

Submitted: 11 April 2026
Revised: 11 May 2026
Accepted: 15 June 2026
Published: 25 August 2026

Vol. 2, **No.** 3, 2026.
10.62762/NGCST.2026.350184

***Corresponding author:**
✉ Barnali Gupta Banik
barnali.guptabanik@ieee.org

Citation
Banik, B. G., Yukitha, P., Venna, C. S, & Tulasi, P. M. (2026). MinuteMaster: An AI-Powered Meeting Transcription and Scheduling System. *Next-Generation Computing Systems and Technologies*, 2(3), 70–75.

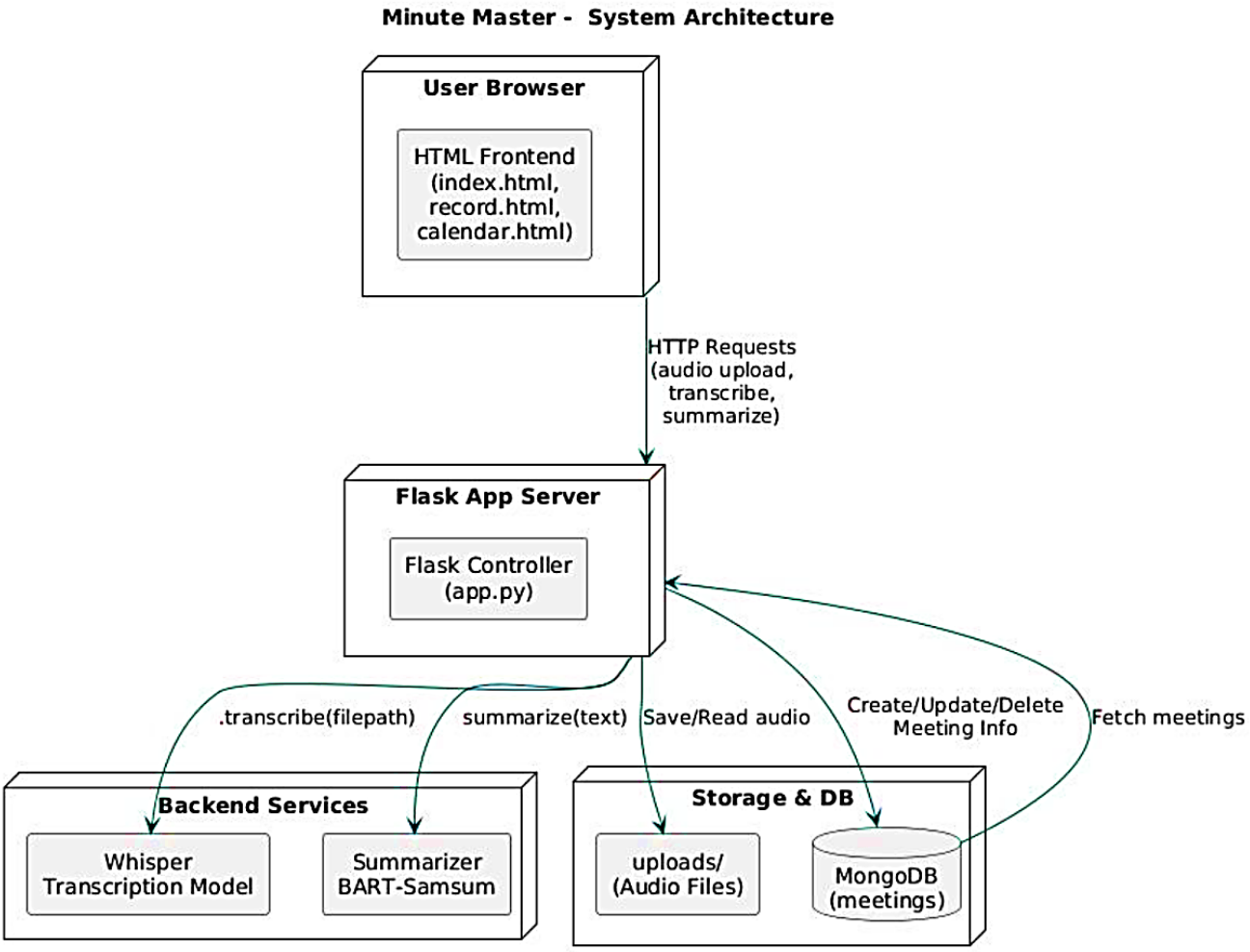

Figure 1. Architecture of the system.

2 Related Work

Research on meeting analysis has been evolving for years. Early work by Murray and Renals [1] showed that it was possible to detect action items using simple cues like word patterns and timing, though these methods struggled with noisy or varied meeting settings. Over time, more advanced approaches improved summarization. Models like HMNet by Zhu et al. [2] looked at both sentence meaning and overall context, including speaker roles, and performed well on datasets such as AMI and ICSI. Narayan et al. [3] used reinforcement learning to rank and select the most informative sentences for extractive summarization, offering a more principled alternative to heuristic sentence selection methods. Similarly, QMSum by Zhong et al. [4] approached summarization as a question-answering task, making outputs more focused and useful. MeetingBank [5] further highlighted the importance of domain-specific data for real-world performance. On the speech side, Radford et al. [11] introduced Whisper, marking a significant step forward in automatic speech recognition with strong multilingual support and robustness to accents and mixed-language conversations. For identifying speakers, pyannote.audio by Bredin et al. [7] has become a reliable tool. Shang et al. [8] proposed an unsupervised abstractive meeting summarization approach combining multi-sentence compression with budgeted submodular maximization to select and compress salient content, demonstrating competitive performance without requiring labeled training data.

In terms of generating summaries, Liu et al. [6] proposed topic-aware pointer-generator networks for summarizing spoken conversations, demonstrating that incorporating topic structure into summarization models leads to more coherent and contextually grounded outputs. Building on such sequence-to-sequence advances, BART by Lewis et al. [9] produces more natural and readable outputs compared to older extractive methods, and has also been shown by Cohen et al. [10] to improve the clarity

of action items—both of which directly inform the summarization design of MinuteMaster.

3 Methodology

MinuteMaster has three main parts, as shown in Figure 1. The first is a browser-based frontend for users to interact with. The second is a Flask-based Python backend that runs all the AI processing. The third is a MongoDB database for storing transcripts, summaries, and meeting records. Each part is kept separate. This makes it easy to update or replace one part without affecting the others.

3.1 Frontend and User Interface

The frontend is built with HTML5, CSS3, and plain JavaScript, keeping it simple and lightweight. It mainly has two key screens. The Transcription screen allows users to upload audio or record directly in the browser using the MediaRecorder API. Once processed, it displays the raw transcript, a cleaned version, speaker-wise text, and a short summary, along with an option to translate the summary. The Scheduling screen provides a calendar view where users can easily add, edit, or delete meetings. Everything is connected through API calls, so updates happen instantly without needing to reload the page.

3.2 Speech Recognition with Whisper

Before transcription, the audio is first cleaned using FFmpeg and pydub. This helps balance volume, remove long silences, and break long recordings into smaller chunks. Each chunk is then sent to Whisper [11] for transcription, which automatically detects the language—so there's no need for manual selection. The system supports common formats like WAV, MP3, M4A, and OGG, as well as live recordings from the browser.

3.3 Speaker Diarization

Once the transcript is ready, pyannote.audio [7] is used to figure out who is speaking and when. It breaks the audio into segments where each segment has a single speaker. These segments are then aligned with the transcript, and each sentence is tagged with labels like Speaker 1 or Speaker 2. This makes the conversation much easier to follow. The speaker-wise version is saved and displayed along with the full transcript and the summary.

3.4 Summarization with BART

The full transcript is passed to the BART model [9] through the Hugging Face Transformers library. Since BART can't handle very long text in one go, the transcript is split into smaller chunks. Each chunk is summarized first, and then those summaries are combined and summarized again to produce a final, clear version. This approach works well even for long meetings. The final summary, along with the transcript, is stored in MongoDB for future access.

4 Experiments

We evaluated MinuteMaster across four areas: transcription accuracy, summary quality, speaker detection, and processing speed. We also ran a user study with 42 participants from MGIT Hyderabad, including students, faculty, and professionals.

4.1 Transcription Accuracy

The system was evaluated on 30 audio clips (2–15 minutes) across five language groups. English performed best at 88.4%, followed by Hindi (83.7%) and Telugu (81.2%), both above the 80% usability mark. Tamil was close at 79.6%, while mixed-language audio was the most challenging at 76.3%, though still usable with minor edits. Figure 2 shows the word-level accuracy for each language group.

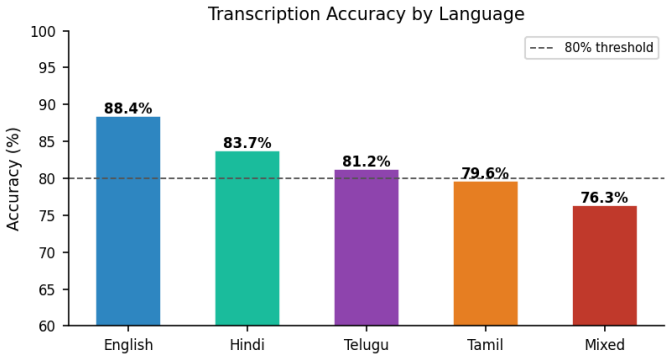

Figure 2. Whisper transcription accuracy by language group (n=30 clips). The dashed line shows the 80% usability threshold.

4.2 Summarization Quality

BART clearly outperformed the TextRank baseline. ROUGE-1 improved from 0.38 to 0.48, and BLEU-4 nearly doubled from 0.11 (TextRank) to 0.18 (BART). The summaries were also more natural and closer to real meeting notes, as confirmed by human evaluation results shown in Figure 3.

4.3 Processing Time

The system scales predictably with audio length. A one-minute clip takes about 2.5 seconds of

Table 1. Performance summary across all evaluation areas.

Metric	Min	Max	Mean	Std Dev
Transcription Accuracy (%)	76.3	88.4	82.6	±4.1
ROUGE-1 (BART)	0.42	0.53	0.48	±0.03
ROUGE-L (BART)	0.39	0.50	0.44	±0.03
BLEU-4 (BART)	0.15	0.21	0.18	±0.02
Diarization Error Rate (%)	6.2	18.7	11.7	±4.5
Pipeline Time / min audio (s)	2.5	64.0	12.1	±8.2
User Satisfaction (1–5)	3.8	4.7	4.25	±0.22

Table 2. MinuteMaster vs. existing commercial tools – feature comparison.

Tool	Multilingual	Speaker ID	AI Summary	Scheduler	Free/Open
Otter.ai	Partial	Yes	Yes	No	No
MS Teams Recap	Limited	Yes	Yes	Yes	No
Zoom AI	Limited	Partial	Yes	Yes	No
Google Meet + Gemini	Yes	Partial	Yes	Partial	No
MinuteMaster (Ours)	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**

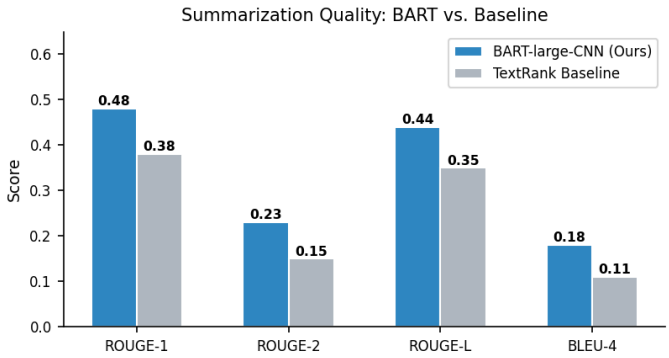

Figure 3. BART vs. TextRank baseline on ROUGE and BLEU-4 scores. Higher values mean better performance.

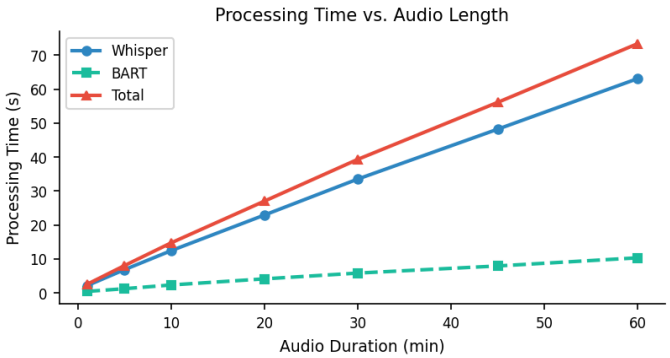

Figure 4. Total pipeline processing time vs. audio duration. BART adds only a small extra time on top of Whisper.

processing time, while longer recordings scale proportionally—fast enough for practical use right after meetings. Figure 4 shows how processing time grows as audio length increases. Table 1 shows the full set of performance results across all evaluation areas.

4.4 Comparison

Compared to popular tools, MinuteMaster stands out by offering multilingual transcription, speaker detection, AI summaries, and built-in scheduling in one place—and for free. It can also be self-hosted, giving organizations full control over their data, which is especially valuable in sectors like healthcare, education, and government. Table 2 compares MinuteMaster with four popular commercial tools.

5 Conclusion

MinuteMaster was created to fix a simple but common issue—important points from meetings often get missed because it's hard to capture everything in real time. The app combines transcription, speaker identification, summarization, and scheduling in one easy-to-use, free platform that supports multiple languages. In testing, it performed well. Transcription accuracy crossed 80% for Indian languages, and the summaries felt more natural compared to basic methods. It was also quick enough to be used right after meetings. Users responded positively, with an

average rating of 4.25/5. Going forward, we plan to improve speaker detection in noisy settings, add real-time transcription, and introduce features like user login and automatic action item extraction. Integration with tools like Google Calendar is also on the roadmap. Since the system is modular, these updates can be added smoothly as it grows.

Data Availability Statement

Data will be made available on request.

Funding

This work was supported without any funding.

Conflicts of Interest

The authors declare no conflicts of interest.

AI Use Statement

The authors declare that Perplexity AI was used to assist with language polishing and grammar checking of the manuscript. The authors have carefully reviewed, revised, and verified all AI-assisted output and take full responsibility for the accuracy and integrity of the content.

Ethical Approval and Consent to Participate

This study involved human participants in a software usability evaluation. As the study posed no greater than minimal risk and involved only voluntary interaction with a software prototype, it was determined to be exempt from full ethics review under the institutional guidelines of Mahatma Gandhi Institute of Technology, Hyderabad. All participants were informed of the study's purpose and provided verbal informed consent prior to participation. Participation was voluntary, and all response data were anonymized before analysis.

References

[1] Murray, G., & Renals, S. (2008). Detecting Action Items in Meetings. In A. Popescu-Belis & R. Stiefelhagen (Eds.), *Machine Learning for Multimodal Interaction (MLMI 2008)*, LNCS vol. 5237, pp. 208–213. Springer. [CrossRef]

[2] Zhu, C., Xu, R., Zeng, M., & Huang, X. (2020, November). A hierarchical network for abstractive meeting summarization with cross-domain pretraining. In *Findings of the association for computational linguistics: EMNLP 2020* (pp. 194-203). [CrossRef]

[3] Narayan, S., Cohen, S. B., & Lapata, M. (2018, June). Ranking sentences for extractive summarization with reinforcement learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)* (pp. 1747-1759). [CrossRef]

[4] Zhong, M., Yin, D., Yu, T., Zaidi, A., Mutuma, M., Jha, R., ... & Radev, D. (2021, June). QMSum: A new benchmark for query-based multi-domain meeting summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 5905-5921). [CrossRef]

[5] Hu, Y., Ganter, T., Deilamsalehy, H., Dernoncourt, F., Foroosh, H., & Liu, F. (2023, July). Meetingbank: A benchmark dataset for meeting summarization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 16409-16423). [CrossRef]

[6] Liu, Z., Ng, A., Lee, S., Aw, A. T., & Chen, N. F. (2019, December). Topic-aware pointer-generator networks for summarizing spoken conversations. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)* (pp. 814-821). IEEE. [CrossRef]

[7] Bredin, H., Yin, R., Coria, J. M., Gelly, G., Korshunov, P., Lavechin, M., ... & Gill, M. P. (2020, May). Pyannote.audio: neural building blocks for speaker diarization. In *ICASSP 2020-2020 IEEE International conference on acoustics, speech and signal processing (ICASSP)* (pp. 7124-7128). IEEE. [CrossRef]

[8] Shang, G., Ding, W., Zhang, Z., Tixier, A., Meladianos, P., Vazirgiannis, M., & Lorré, J. P. (2018, July). Unsupervised abstractive meeting summarization with multi-sentence compression and budgeted submodular maximization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 664-674). [CrossRef]

[9] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., ... & Zettlemoyer, L. (2020, July). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 7871-7880). [CrossRef]

[10] Cohen, A., Kantor, A., Hilleli, S., & Kolman, E. (2021, August). Automatic rephrasing of transcripts-based action items. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021* (pp. 2862-2873). https://aclanthology.org/2021.findings-acl.253.pdf

[11] Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., & Sutskever, I. (2023, July). Robust speech recognition via large-scale weak supervision. In *International conference on machine learning* (pp. 28492-28518). PMLR.

Barnali Gupta Banik is an Associate Professor with over 20 years of diverse academic and professional experience. She holds a doctorate and has authored more than 50 research publications. She is a Senior Member of IEEE and actively contributes to the academic community through guest lectures, conference session chair roles, and as a jury member for various hackathons. Her research interests include Artificial Intelligence and Machine Learning, Blockchain, and Cryptography. (Email: barnali.guptabanik@ieee.org)

Pampari Yukitha Final-year B.Tech student in Computer Science and Engineering (Data Science) at MGIT Hyderabad. Her work focuses on natural language processing, web development, and HCI. She designed the frontend, led user testing, and handled performance evaluation for this project. (Email: pyukitha_csd226747@mgit.ac.in)

Charan Sai Venna Final-year B.Tech student in Computer Science and Engineering (Data Science) at MGIT Hyderabad. His interests are in backend systems, AI integration, and full-stack development. He built the Flask backend, integrated Whisper and BART, and designed the MongoDB data layer for this project. (Email: csaivenna_csd226711@mgit.ac.in)

Puppala Mangala Tulasi is an academician with over 16 years of diverse teaching experience. She is currently pursuing her Ph.D. at National Institute of Technology Tiruchirappalli. Her research interests include Deep Learning and Image Processing. She has presented and published three papers in international conferences and has actively participated in several faculty development programs and workshops, reflecting her commitment to continuous learning and professional growth.(Email: pmangalatulasi_cse@mgit.ac.in)