IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 7, October 2026
Impact Factor: 8.2

AI-Powered Meeting Summarization with Task Extraction and Smart Scheduling

Dengale Gaurav Baban[1], Bargaje Dnyaneshwar Ambadas[2], Shinde Prasad Pandurang[3], Walunj Siddhi Gauram[4], Jawkar Onkar Pramod[5], Prof. M. N. Pimple[6], Prof. G. R. Sarode[7], Dr. H. B. Jadhav[8]
Student, Department of Computer Engineering[12345]
Faculty, Department of Computer Engineering[678]
Adsul's Technical Campus, Chas, Ahilyanagar, Maharashtra, India

Abstract: *Meetings are where teams make decisions and assign work, yet the knowledge they produce is captured poorly: manual minutes are incomplete, action items are scattered across notes, e-mail and chat, deadlines are forgotten, and a decision taken in one meeting may be silently contradicted in another. This paper presents MeetX, an AI-powered meeting intelligence platform that converts spoken meeting content into structured, verifiable and actionable information. Audio is captured from a Google Meet bot integration or the browser microphone and transcribed with a Whisper-based component. Llama 3 models served by Groq, orchestrated by a Python FastAPI service with LangChain, generate executive summaries, key takeaways, decisions and action items, each task carrying an assignee, due date, priority and confidence score. Low-confidence tasks enter a human-in-the-loop review queue, and approved tasks are dispatched through Gmail, Slack webhooks and Google Calendar. Distinctive modules include a SHA-256 decision ledger with Markdown ADR export, decision-reversal tracking with a proposed Team Decision Stability Index, DAG-based cross-meeting deadline-conflict analysis, a retrieval-augmented "Ask AI" memory and bi-directional status synchronization. The front end uses React 18 and TypeScript; the back end uses Spring Boot 3 with PostgreSQL. This paper reports the design, mathematical model and evaluation plan of the system. It is a preliminary work, and no measured results are claimed.*

Keywords: Meeting Summarization, Action-Item Extraction, Speech Recognition, Large Language Models, Retrieval-Augmented Generation, Decision Tracking, Human-in-the-Loop, SHA-256

I. INTRODUCTION

Meetings are the main mechanism through which teams make decisions and assign work, but the knowledge produced in them is rarely captured reliably. Manual note-taking is incomplete and subjective; action items end up in personal notes, chat threads and e-mails; owners and deadlines stated aloud are forgotten; and a decision taken in one meeting can be contradicted in a later one without any record of the change. Follow-up also forces the same information to be re-entered in separate tools such as e-mail, team chat and calendars. As meetings accumulate, answering "what was decided, by whom and why" becomes a slow manual search.

Recent advances in speech recognition [1] and large language models (LLMs) [2], [3], [4] make it feasible to automate much of this work. However, fully automatic output can be wrong, so teams need a way to review uncertain results before they trigger real notifications. This paper presents **MeetX**, a web-based platform that listens to a meeting, understands it, and then acts on it: every spoken commitment becomes a task with an owner and a deadline, every decision becomes a verifiable record, and every follow-up is sent through the channel the team already uses, while a human reviewer stays in control of low-confidence items.

Copyright to IJARSCT
www.ijarsct.co.in
DOI: 10.48175/IJARSCT-40481

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 7, October 2026
Impact Factor: 8.2

The remainder of the paper is organised as follows. Section II reviews related work. Section III states the problem and objectives. Section IV describes the architecture and methodology, and Section V gives the mathematical model. Sections VI and VII present the technology stack and features. Section VIII describes the evaluation plan and current status. Sections IX to XII discuss applications, limitations, future scope and conclusions.

II. LITERATURE REVIEW

Speech recognition for meetings. Research on meeting understanding relies on corpora such as the ICSI [5] and AMI [6] meeting corpora, which pair multi-party recordings with transcripts and annotations. Meetings are difficult for recognisers because of overlapping speech, informal language and variable acoustics. Whisper [1] is trained on large-scale weakly supervised multilingual audio and is designed to be robust to accents, noise and technical vocabulary. Recognition quality is commonly measured by word error rate, WER = (S + D + I) / N.

Language models and summarization. Transformer networks [2] underlie modern LLMs, which can perform tasks from instructions or a few examples without task-specific training [3]; open families such as Llama 3 [4] are available through hosted inference services such as Groq [7]. Meeting summarization is harder than news summarization because transcripts are long and multi-speaker, as shown by the hierarchical model of Zhu et al. [8] and the QMSum benchmark [9] for query-based summaries.

Action-item and decision extraction. Purver et al. [10] formulated action-item detection as an annotation and classification problem covering the task, its owner, timeframe and agreement, and showed that action items are spread over several utterances and are often implicit. LLMs can extract such structure directly from a transcript, but they may omit items, invent details or misattribute owners, which motivates confidence scoring and human review.

Retrieval, integrity and scheduling. Retrieval-Augmented Generation (RAG) [11] combines a retriever with a generator so that answers can be traced to sources; a typical pipeline uses chunking, sentence embeddings [12] or dense retrieval [13], vector search and answer generation [14]. Architecture Decision Records (ADRs) [15] capture the context, decision and consequences of a decision in a lightweight format, and hash-based time-stamping [16] with SHA-256 [17] makes later alteration detectable. Temporal constraint networks [18] and interval algebra [19] provide the theory for deadline-overlap reasoning, and topological sorting [20] checks that task dependencies form a directed acyclic graph (DAG). Finally, the human–AI interaction guidelines of Amershi et al. [21] recommend showing how well the system can do what it does and supporting efficient correction.

Research gap. The surveyed work covers speech recognition, summarization, action-item detection and retrieval individually. MeetX addresses their integration into one workflow that also includes human review, decision-integrity verification, decision-reversal tracking, cross-meeting deadline-conflict analysis and synchronization with e-mail, chat and calendar tools. This statement is based on the surveyed sources and does not claim that no other system exists.

III. PROBLEM STATEMENT AND OBJECTIVES

A. Problem Statement

To design and develop a web-based platform that (i) captures meeting audio from Google Meet and the browser microphone, (ii) transcribes it, (iii) uses LLMs to produce summaries, decisions and action items with assignees, due dates, priorities and confidence scores, (iv) lets a human reviewer correct and approve uncertain items, (v) dispatches approved tasks through Gmail, Slack and Google Calendar, (vi) keeps a tamper-evident record of decisions and tracks reversals, (vii) detects cross-meeting deadline conflicts, (viii) answers natural-language questions over past meetings using RAG, and (ix) synchronizes task status from replies and shows history and analytics.

B. Objectives

- To capture audio from a Google Meet session and the browser microphone and store a text transcript.
- To generate an executive summary, key takeaways and a sentiment indication for every processed meeting.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 7, October 2026
Impact Factor: 8.2

- To extract tasks with assignee (fuzzy-matched to team contacts), due date, priority and confidence, and evaluate extraction with precision, recall and F1-score on a manually labelled sample.
- To dispatch task e-mails, Slack summaries and Calendar follow-ups, recording the outcome of each dispatch.
- To store a SHA-256 hash for every decision, provide integrity verification and Markdown ADR export, and track decision reversals.
- To detect overlapping deadlines and owner over-commitment across meetings and propose adjustments.
- To answer questions about past meetings with grounded, source-referenced answers, and to route low-confidence tasks to human review.

All numerical targets in this paper are proposed design targets and not achieved results.

IV. SYSTEM ARCHITECTURE AND METHODOLOGY

A. Overall Architecture

MeetX follows a three-tier design, shown in Fig. 1: a React single-page application (presentation layer), a Spring Boot REST back end that owns business logic, persistence and all calls to Gmail, Calendar and Slack (application layer), and a separate Python FastAPI service for AI work (AI service layer) [22], [23]. PostgreSQL [24] is the data layer. Separating the AI service lets it use the Python ecosystem (Whisper, LangChain [25]) while the back end uses mature Java tooling for transactions, validation and integrations. Each external dependency (Groq, Whisper hosting, Google APIs, Slack) sits behind a service interface so that it can be replaced and so that failure of one channel does not stop the others.

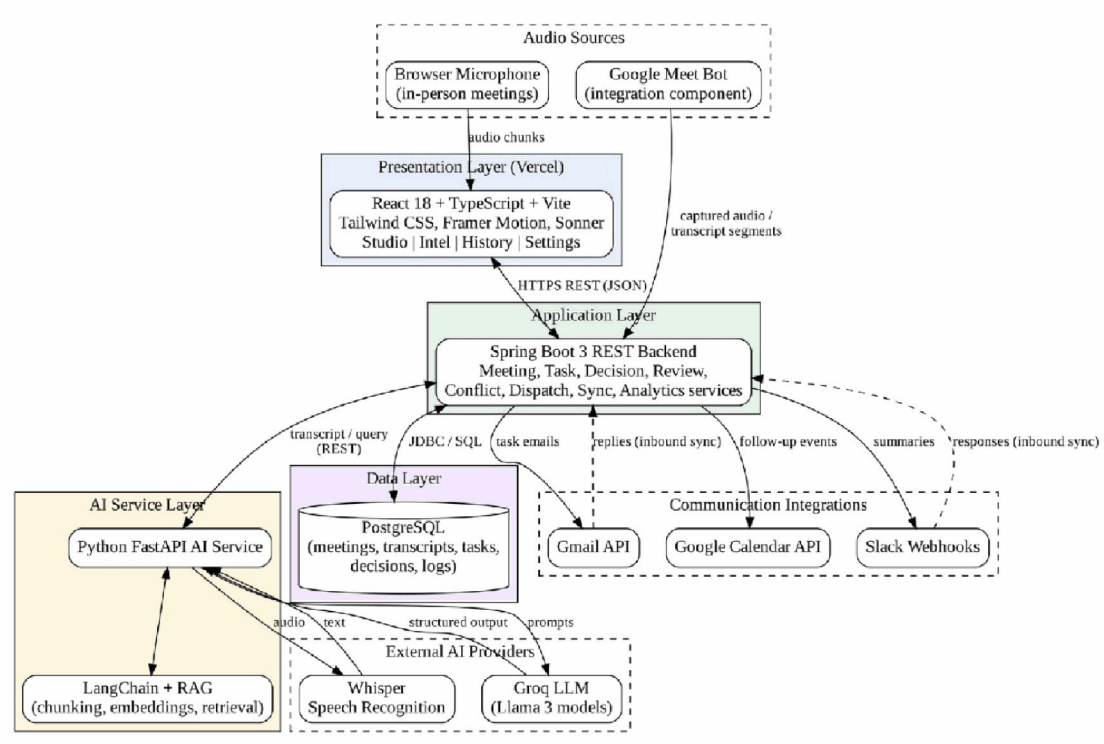


Fig. 1. System architecture of MeetX

B. Process Flow

The main workflow (Fig. 2) has seven steps. **Capture:** the user pastes a Google Meet URL, so that a bot joins and captures audio, or starts the browser microphone for an in-person meeting. **Transcribe:** audio chunks are sent to the AI

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 7, October 2026
Impact Factor: 8.2

service where a Whisper-based component produces text, which is stored in PostgreSQL. **Understand:** structured prompts to Llama 3 models on Groq return a summary, takeaways, decisions, tasks and a sentiment label. **Review:** tasks below the confidence threshold are placed in a review queue. **Act:** approved tasks are dispatched through Gmail, Slack and Calendar. **Remember and verify:** decisions are hashed and checked against earlier ones, and open tasks are analysed for conflicts. **Synchronize:** supported replies update task status.

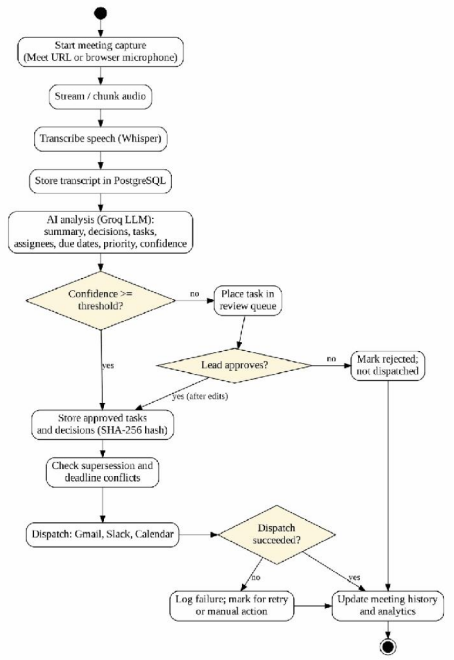


Fig. 2. Activity diagram of the main MeetX workflow

C. Capture and Transcription

Browser-recorded audio (for example WebM/Opus) is split into chunks of a few seconds to tens of seconds and sent to the AI service; the transcript is assembled in time order. The Google Meet bot is an integration component whose success depends on meeting settings and host admission, so browser-microphone capture is kept as an independent fallback. Because background noise, accents and crosstalk lower accuracy, transcripts are stored and remain inspectable by users.

D. AI Analysis and Confidence-Based Routing

Long transcripts are split into segments to respect the context window and rate limits of the LLM. The FastAPI service prompts the model for a summary, decisions and tasks as JSON, validates the output against a schema and retries invalid output. Assignee names are matched to the team contact list by normalized edit distance (Section V-B). Each task receives a confidence score; tasks below the threshold are routed to review instead of automatic dispatch.

E. Human Review and Dispatch

In the review queue a team lead can edit the assignee, priority or due date, and approve or reject each task, following the guidance that confidence should be visible and correction should be efficient [21]. Unapproved tasks are held back from dispatch when review is required. Approved tasks are sent as e-mails through Gmail, summaries and action items are posted to Slack through webhooks, and follow-up meetings are created in Google Calendar, with OAuth 2.0 consent where required [26]. Every dispatch attempt is logged per channel with its outcome and error message, so that failed dispatches can be retried or handled manually.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 7, October 2026 Impact Factor: 8.2

F. Decision Ledger and Supersession Tracking

Each decision is serialized in a canonical form and hashed with SHA-256 when it is saved. A verification action recomputes the hash of the stored record and reports a match or mismatch, and decisions can be exported as Markdown ADRs [15]. When a new decision is saved, it is compared with earlier decisions, and a supersession link (reverses, updates or contradicts) is recorded and the team lead is alerted. Hashing detects but does not prevent modification, and protects only the hashed fields, so the reference hash must itself be protected.

G. Deadline-Conflict Analysis

Open tasks across all meetings are analysed to find overlapping deadlines for the same owner, owner over-commitment, and temporal conflicts along task dependencies. The analysis reports the affected tasks and proposes a timeline adjustment that the user confirms with one click; it detects and explains conflicts but does not by itself optimize workloads.

H. RAG-Based "Ask AI" Memory and Inbound Synchronization

Transcripts, summaries, tasks and decisions are chunked and embedded, and a question is answered from the top-ranked chunks with references to the sources. In the other direction, replies such as "Done" or "Need two more days" received by e-mail or Slack are matched to a task and either update its status or propose a new deadline; only a fixed, configurable set of phrases is supported and unmatched replies are flagged.

V. MATHEMATICAL MODEL

The system is modelled as a function Φ that maps the audio stream A_m of a meeting m to its transcript T_m, summary S_m, task set K_m and decision set D_m, where $\Phi = \text{DSP} \circ \text{HR} \circ \text{LLM} \circ \text{ASR}$ (dispatch, human review, language-model analysis and speech recognition). Human review HR is applied only to tasks whose confidence is below a threshold τ. The formulations below are proposed design definitions and have not been experimentally validated.

A. Task Confidence and Routing

The confidence $c_i \in [0, 1]$ of an extracted task k_i is a weighted combination of the model-reported certainty s_{llm} and evidence scores for the assignee match (s_{asg}), the due date (s_{due}) and the explicitness of the commitment (s_{ctx}):

$$c_i = w_1 s_{llm} + w_2 s_{asg} + w_3 s_{due} + w_4 s_{ctx}, \quad \Sigma w_j = 1, \; w_j \geq 0 \tag{1}$$

$$\text{route}(k_i) = \text{REVIEW if } c_i < \tau, \text{ otherwise AUTO} \tag{2}$$

The weights and τ are configuration parameters to be tuned on labelled data; a confidence score is an estimate, not a guarantee of correctness.

B. Extraction Metrics and Assignee Matching

With TP extracted tasks matching a labelled task, FP extracted tasks without a match and FN missed labelled tasks:

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP}), \quad \text{Recall} = \text{TP} / (\text{TP} + \text{FN}) \tag{3}$$

$$\text{F1} = 2 \cdot \text{Precision} \cdot \text{Recall} / (\text{Precision} + \text{Recall}) \tag{4}$$

A metric whose denominator is zero is reported as undefined. An extracted name a is matched to a contact b by $\text{sim}(a, b) = 1 - \text{Lev}(a, b) / \max(|a|, |b|)$, where Lev is the Levenshtein distance; the best contact above a threshold is chosen, otherwise the task is left unassigned or reviewed.

C. Decision Integrity

For a decision d with canonical serialization ser(d) (fixed field order, normalized whitespace, UTF-8), the stored hash is $h = \text{SHA-256}(\text{ser}(d))$ and verification recomputes the hash of the stored record d':

$$\text{verify}(d', h) = \text{TRUE} \text{ if and only if } \text{SHA-256}(\text{ser}(d')) = h \tag{5}$$

As an optional extension in the style of hash chaining [16], each record may include the previous hash: $h_k = \text{SHA-256}(\text{ser}(d_k) \,\|\, h_{k-1})$.

D. Deadline Overlap and Dependency Conflicts

Each open task k_i with owner o is an interval $I_i = [s_i, e_i]$, where e_i is its due date and $s_i = e_i - \delta_i$ is the latest start implied by its estimated effort δ_i. Two tasks of one owner overlap, and the owner is over-committed at time t, when:

$$s_i < e_j \text{ and } s_j < e_i \tag{6}$$

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 7, October 2026 Impact Factor: 8.2

$$L_o(t) = \Sigma\ load_i \text{ over tasks active at } t \text{ with owner } o; \quad \text{flag if } L_o(t) > C_o \tag{7}$$

For dependencies, the graph $G = (V, E)$, with edge $(k_a \rightarrow k_b)$ meaning that k_b cannot start before k_a finishes, must be acyclic; topological sorting [20] verifies this in $O(V + E)$ time. The earliest start of a task is $ES_b = \max_a(ES_a + \delta_a)$ over its predecessors, and a due date earlier than $ES_b + \delta_b$ indicates a temporal conflict [18], [19].

E. Retrieval Evaluation

Chunks and queries are embedded as vectors and ranked by cosine similarity. For a question set Q with relevant-chunk sets R_q, retrieval is evaluated by Recall@k and mean reciprocal rank (MRR):

$$\text{Recall@}k = (1 / |Q|)\ \Sigma\ |R_q \cap \text{TopK}(q)| / |R_q|, \quad \text{MRR} = (1 / |Q|)\ \Sigma\ 1 / \text{rank}_q \tag{8}$$

Answer groundedness is assessed manually as the fraction of answer statements supported by the cited chunks.

F. Dashboard Metrics and Team Decision Stability Index

$$\text{Task Completion Rate} = (\text{completed tasks} / \text{tasks due in period}) \times 100 \tag{9}$$

$$\text{Dispatch Success Rate} = (\text{successful attempts} / \text{total attempts}) \times 100 \tag{10}$$

The dashboard shows "N/A" when a denominator is zero. The Team Decision Stability Index (TDSI) is a project-defined metric proposed in this work; for a period with $N_{dec} > 0$ decisions, of which N_{rev} were later reversed and N_{upd} later updated or contradicted:

$$\text{TDSI} = 100 \times \max\{0,\ 1 - (N_{rev} + 0.5\ N_{upd}) / N_{dec}\} \tag{11}$$

A value of 100 means no later reversal or revision. The weights 1 and 0.5 are adjustable assumptions, and the formula has not been verified in an implementation.

VI. TECHNOLOGY STACK

MeetX is built with the technologies summarised in Table I. Provider-specific code (Groq, Gmail, Slack) is kept behind interfaces, and configuration is externalized so that services can be replaced or containerized.

TABLE I: TECHNOLOGIES USED IN MEETX

Component	Technology	Purpose
Front end	React 18, TypeScript, Vite, Tailwind CSS	Responsive single-page UI (Studio, Intel, History, Settings)
Back end	Java Spring Boot 3, REST APIs	Business logic, validation, transactions, integrations
AI service	Python FastAPI, LangChain	Orchestration of transcription, analysis and RAG
Speech-to-text	Whisper-based component	Transcription of audio chunks
Language model	Llama 3 on Groq	Summaries, decisions, tasks, confidence
Database	PostgreSQL (pgvector optional)	Meetings, tasks, decisions, logs, embeddings
Integrations	Gmail API, Google Calendar API, Slack webhooks	Task e-mails, follow-up events, summaries
Deployment	Vercel (front end), cloud-hosted back end	Hosting; back-end provider not yet fixed

VII. FEATURES AND FUNCTIONAL REQUIREMENTS

The system is specified by 30 functional requirements (FR-01 to FR-30) covering authentication, capture, analysis, review, dispatch, the decision ledger, conflict analysis, RAG memory, synchronization, history and a responsive interface. The principal features are listed in Table II. Role-based access control (team member, organizer, team lead/reviewer, administrator) is a stated requirement and is not assumed to be implemented.

TABLE II: KEY FEATURES OF MEETX

Feature	Description	Requirements
Capture	Google Meet bot capture or browser-microphone capture with live waveform and chunking	FR-02, FR-03

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 7, October 2026
Impact Factor: 8.2

Feature	Description	Requirements
Transcription	Time-ordered transcript of audio chunks, stored and inspectable	FR-04
Summarization	Executive summary, key takeaways, sentiment label	FR-05
Task extraction	Title, description, assignee, due date, priority, confidence score	FR-06, FR-08, FR-09
Human review	Edit, approve or reject low-confidence tasks; hold unapproved tasks back	FR-10 to FR-12
Dispatch	Gmail, Slack webhook and Calendar dispatch with per-attempt logs	FR-13 to FR-16
Decision ledger	SHA-256 hash on save, verification, Markdown ADR export	FR-17 to FR-19
Reversal tracking	Supersession links, lead alert, TDSI (proposed)	FR-20, FR-21
Conflict analysis	Deadline overlap, over-commitment, DAG check, one-click proposals	FR-22 to FR-24
Ask AI (RAG)	Natural-language questions answered with source references	FR-25, FR-26
Inbound sync	Status or deadline updates from supported e-mail/Slack replies	FR-27
History and analytics	Searchable history; completion, workload and dispatch KPIs	FR-28, FR-29
Responsive UI	Command palette, mobile drawer, bottom navigation, onboarding tour	FR-30

VIII. IMPLEMENTATION PLAN AND EVALUATION

The project follows an iterative Agile (Scrum-style) model [27] over a planned period of about 22 weeks, in six iterations: (1) capture, transcription and storage; (2) summary, task and decision extraction; (3) review queue and dispatch; (4) decision ledger, supersession and conflict analysis; (5) RAG memory, inbound synchronization, history and analytics; and (6) interface polish, hardening and deployment. This schedule is a plan and not a record of completed work.

Evaluation will use the metrics defined in Section V together with the proposed tests in Table III. These tests have not yet been executed, and **no measured results are reported in this preliminary paper**.

TABLE III: PROPOSED EVALUATION AND TEST CASES (NOT YET EXECUTED)

Aspect	Proposed test	Expected result / metric
Transcription	Noisy and clean audio samples	Word error rate (WER) per condition
Task extraction	Labelled sample transcripts	Precision, recall, F1-score
Review gate	Task with confidence below τ	Appears in review queue; not dispatched until approved
Decision integrity	Modify a stored decision, then verify	Hash mismatch reported
Supersession	Add a decision contradicting an earlier one	Link created and lead alerted
Conflict analysis	Two overlapping tasks for one owner	Conflict reported with a proposal
Ask AI	Questions present in stored meetings	Recall@k, MRR, groundedness
Inbound sync	Reply “Done” to a task e-mail	Task status becomes Done
Failure isolation	Invalid Slack webhook	Failure logged; other channels unaffected

IX. APPLICATIONS

- Software-development teams: stand-ups, sprint planning and design meetings, with ADR export for technical decisions.
- Corporate and consulting meetings: management and client meetings with action tracking and documented commitments.
- Project management and distributed teams: converting discussions into tracked tasks and follow-up events, with a shared meeting memory across time zones.
- Educational project groups: recording guide meetings and tracking agreed actions.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 7, October 2026
Impact Factor: 8.2

X. ADVANTAGES AND LIMITATIONS

A. Advantages

- Integrates capture, summarization, task extraction, review and dispatch in one workflow, so that information is not re-entered across tools.
- Keeps a human in control of uncertain AI output through confidence scores and a review gate.
- Makes decisions traceable through hashed records, ADR export and reversal tracking.
- Modular design: each external service is isolated behind an interface and can be replaced.

B. Limitations

- Requires an internet connection; transcription and LLM inference run in cloud services, and offline operation is not supported.
- Transcription accuracy drops with noise, accents and overlapping speakers, and LLM output can be incorrect or inconsistent.
- The Google Meet bot depends on meeting settings, host admission and platform terms.
- Hash verification detects but does not prevent modification, and graph-based analysis does not by itself optimize workloads.
- Third-party quotas and outages affect dispatch. Recording and AI processing require participant consent, and transcripts may contain personal data processed by third parties.

XI. FUTURE SCOPE

- Speaker diarization and multilingual meeting support.
- Finer-grained role-based access control and integration with project-management tools.
- Learned tuning of the confidence threshold and weights from reviewer feedback.
- External anchoring of decision hashes for stronger tamper evidence.
- Larger-scale evaluation with real teams, including usability and accessibility testing.

XII. CONCLUSION

This paper presented MeetX, an AI-powered platform that converts meeting audio into summaries, decisions and tasks with owners, deadlines, priorities and confidence scores, and coordinates follow-up through Gmail, Slack and Google Calendar. Its distinguishing elements are a human-in-the-loop review gate, a SHA-256 decision ledger with supersession tracking, DAG-based deadline-conflict analysis and a RAG-based meeting memory. The design is deliberately cautious about AI limitations: confidence scores, review and logged, verifiable records keep people responsible for outcomes. The expected benefits are less manual documentation and follow-up effort and better traceability of decisions and responsibilities; these are expected benefits, not measured results. Implementation and evaluation using the plan above will show how far the goals are achieved.

ACKNOWLEDGMENT

The authors thank the Department of Computer Engineering, Adsul's Technical Campus, Ahilyanagar, for providing the facilities and guidance that made this project possible.

REFERENCES

[1]. A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *Proc. 40th Int. Conf. Machine Learning (ICML)*, PMLR vol. 202, 2023.

[2]. A. Vaswani et al., "Attention is all you need," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 30, 2017, pp. 5998–6008.

[3]. T. B. Brown et al., "Language models are few-shot learners," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 1877–1901.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 7, October 2026
Impact Factor: 8.2

[4]. A. Dubey et al., "The Llama 3 herd of models," *arXiv preprint* arXiv:2407.21783, 2024.

[5]. A. Janin et al., "The ICSI meeting corpus," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 1, 2003, pp. I-364–I-367.

[6]. I. McCowan et al., "The AMI meeting corpus," in *Proc. 5th Int. Conf. Methods and Techniques in Behavioral Research (Measuring Behavior)*, 2005.

[7]. Groq, Inc., "GroqCloud developer documentation." [Online]. Available: https://console.groq.com/docs

[8]. C. Zhu, R. Xu, M. Zeng, and X. Huang, "A hierarchical network for abstractive meeting summarization with cross-domain pretraining," in *Findings of the ACL: EMNLP 2020*, 2020.

[9]. M. Zhong et al., "QMSum: A new benchmark for query-based multi-domain meeting summarization," in *Proc. NAACL-HLT*, 2021, pp. 5905–5921.

[10]. M. Purver, P. Ehlen, and J. Niekrasz, "Detecting action items in multi-party meetings: Annotation and initial experiments," in *Machine Learning for Multimodal Interaction (MLMI 2006)*, LNCS vol. 4299. Berlin, Germany: Springer, 2006, pp. 200–211.

[11]. P. Lewis et al., "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 9459–9474.

[12]. N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in *Proc. EMNLP-IJCNLP*, 2019, pp. 3982–3992.

[13]. V. Karpukhin et al., "Dense passage retrieval for open-domain question answering," in *Proc. EMNLP*, 2020, pp. 6769–6781.

[14]. Y. Gao et al., "Retrieval-augmented generation for large language models: A survey," *arXiv preprint* arXiv:2312.10997, 2023.

[15]. M. Nygard, "Documenting architecture decisions," Cognitect blog, Nov. 2011. [Online]. Available: https://www.cognitect.com/blog/2011/11/15/documenting-architecture-decisions

[16]. S. Haber and W. S. Stornetta, "How to time-stamp a digital document," *J. Cryptology*, vol. 3, no. 2, pp. 99–111, 1991.

[17]. National Institute of Standards and Technology, "Secure Hash Standard (SHS)," FIPS PUB 180-4, Aug. 2015.

[18]. R. Dechter, I. Meiri, and J. Pearl, "Temporal constraint networks," *Artificial Intelligence*, vol. 49, no. 1–3, pp. 61–95, 1991.

[19]. J. F. Allen, "Maintaining knowledge about temporal intervals," *Commun. ACM*, vol. 26, no. 11, pp. 832–843, 1983.

[20]. A. B. Kahn, "Topological sorting of large networks," *Commun. ACM*, vol. 5, no. 11, pp. 558–562, 1962.

[21]. S. Amershi et al., "Guidelines for human-AI interaction," in *Proc. 2019 CHI Conf. Human Factors in Computing Systems*, 2019, Paper 3, pp. 1–13.

[22]. S. Ramírez, "FastAPI documentation." [Online]. Available: https://fastapi.tiangolo.com

[23] VMware/Broadcom, "Spring Boot reference documentation." [Online]. Available: https://spring.io/projects/spring-boot

[24]. PostgreSQL Global Development Group, "PostgreSQL documentation." [Online]. Available: https://www.postgresql.org/docs/

[25]. LangChain, Inc., "LangChain documentation." [Online]. Available: https://python.langchain.com

[26]. D. Hardt, Ed., "The OAuth 2.0 authorization framework," IETF RFC 6749, Oct. 2012.

[27]. K. Beck et al., "Manifesto for Agile Software Development," 2001. [Online]. Available: https://agilemanifesto.org