CognitX Pilot Chatbot — User Stories & Acceptance Criteria

Access rule (applies globally): A User can only receive information sourced from meetings where they are the organizer or an attendee. An Admin can query across all meetings regardless of attendance. This rule is enforced at retrieval time, not just at the UI level (i.e., excluded meetings must not leak into context sent to the LLM).

USER STORIES

U1 — Ask a natural-language question

As a user, **I want to** ask a natural-language question about past meetings, **so that** I get an answer without manually searching transcripts or summaries.

Acceptance Criteria:
- Given a user submits a free-text question, when the system processes it, then it retrieves relevant chunks only from meetings where the user is organizer or attendee.
- Given relevant chunks are found, when the answer is generated, then it must be grounded only in retrieved content (no fabrication beyond what's in the summaries/transcripts).
- Given no relevant chunks exist within the user's authorized meetings, when the query is processed, then the system returns a clear "no information found" response rather than a guess.
- Given the same question is asked twice, when data hasn't changed, then the answer should be consistent (no random variation in facts, only phrasing).
- Response time: answer returned within an agreed SLA (e.g., <5s for retrieval + generation) under normal load.

U2 — See source attribution

As a user, **I want to** see which meeting(s) an answer was derived from (title, date, organizer), **so that** I can verify the information or dig deeper.

Acceptance Criteria:
- Given an answer is generated, when it's displayed, then it includes a list of source meetings with title, date, and organizer for each chunk used.
- Given multiple meetings contributed to one answer, when sources are shown, then they are deduplicated and ordered by relevance or recency.
- Given a user clicks/expands a source, when triggered, then they see the relevant excerpt (summary or transcript snippet) that supported the answer — not the full raw transcript unless they have access.
- Given a source meeting is one the user is not authorized to see, when generating the answer, then that source must never appear, even indirectly through paraphrased content.

U3 — Filter/scope by project, department, or date range

As a user, **I want to** scope my question by project, department, or date range, **so that** I get relevant results instead of noise across unrelated meetings.

Acceptance Criteria:
- Given a user applies a filter (project/department/date range), when a query is submitted, then retrieval is constrained to metadata matching the filter AND the user's access rights (both conditions apply, not either/or).
- Given a filter returns zero eligible meetings, when the query runs, then the system informs the user no matching meetings were found, rather than falling back to unfiltered results silently.
- Given no filter is applied, when a query runs, then the system searches across all meetings the user has access to.
- Filters must be combinable (e.g., project AND date range together).

U4 — Ask "who is the go-to person for X"

As a user, **I want to** ask who the go-to person is for a given topic, **so that** I can find the right contact without digging through meeting history.

Acceptance Criteria:
- Given multiple meetings mention a topic, when the system identifies a "go-to person," then it should be based on explicit signals (e.g., stated ownership, assigned action items, frequency of being named as responsible) — not just attendance frequency.
- Given the system cannot confidently determine a go-to person from available data, when queried,

then it states this explicitly rather than guessing based on weak signals (e.g., most frequent attendee).
- Given an answer is given, when displayed, then it cites the specific meeting(s)/moment(s) where that person was identified as responsible.
- Given ownership appears to have changed over time (e.g., reassigned), when answering, then the system should prioritize the most recent signal and can optionally note the change.

U5 — Get an honest "I don't know"

As a user, **I want to** be told when there isn't enough information to answer confidently, **so that** I don't act on a hallucinated or guessed response.

Acceptance Criteria:
- Given retrieval returns low-relevance or no matching chunks, when generating a response, then the system must default to a clear "insufficient information" message.
- Given the system is uncertain but has partial information, when answering, then it should present what it found and explicitly flag the gaps (e.g., "found status update from March, nothing more recent").
- A confidence/relevance threshold must be defined and tunable (e.g., via retrieval similarity score) to trigger this fallback — this is not left to the LLM's own judgment alone.

U6 — Multi-turn follow-up questions

As a user, **I want to** ask follow-up questions in the same conversation, **so that** context carries over and I don't have to repeat myself.

Acceptance Criteria:
- Given a user asks a follow-up question, when it's processed, then prior conversation turns (question + answer + sources) are included as context for retrieval and generation.
- Given a follow-up references something ambiguous ("what about last month?"), when resolved, then the system uses conversation history to disambiguate before retrieving.
- Given a session is inactive beyond a defined timeout, when the user returns, then context is reset (define the timeout, e.g., 30 min).
- Access control (organizer/attendee-only) must be re-validated on every turn, not just the first one, in case authorization context changes mid-session.

U7 — Flag an answer as wrong/unhelpful

As a user, **I want to** flag an incorrect or unhelpful answer, **so that** the retrieval/generation quality can be reviewed and improved.

Acceptance Criteria:
- Given a user flags an answer, when submitted, then the system logs the question, generated answer, retrieved sources, and user ID/timestamp for review.
- Given a flag is submitted, when logged, then it is visible to admins in a review queue (see A3).
- Optional: user can specify a reason category (e.g., "wrong person," "outdated info," "irrelevant," "hallucinated").
- Flagging must not require the user to expose more meeting content than their own permissions already allow.

ADMIN STORIES

A1 — Control which meetings are indexed

As an admin, **I want to** control which meetings/summaries are included in the chatbot's index, **so that** sensitive meetings remain excluded from retrieval entirely.

Acceptance Criteria:
- Given an admin marks a meeting as excluded, when indexing runs, then that meeting's chunks are removed/never added to the vector store, not just hidden at query time.
- Given a meeting is excluded, when any user (including admin queries not explicitly scoped to "all") asks a related question, then that meeting must not surface unless the admin explicitly includes it back.
- Exclusion actions must be logged (who excluded what, when).
- Given an already-indexed meeting is excluded after the fact, when the exclusion is applied, then existing embeddings for that meeting are purged, not just flagged.

A2 — View usage/query logs

As an admin, **I want to** see usage and query logs, **so that** I understand what people ask and where the bot underperforms.

Acceptance Criteria:
- Given queries are submitted, when logged, then each entry includes: user (or role), question text, timestamp, retrieved sources, answer given, and confidence/relevance score.
- Given an admin views logs, when filtering, then they can filter by date range, user, department, or "no answer found" outcomes.
- Logs must respect data minimization — admin can see query logs across all users (since admin has full visibility), but this access should itself be audited.
- Aggregate metrics should be available: query volume over time, % of "no answer" responses,

most frequent topics.

A3 — Review flagged answers

As an admin, **I want to** review answers flagged by users, **so that** I can identify and fix retrieval/generation issues.

Acceptance Criteria:
- Given a flagged answer, when viewed by admin, then it shows full context: question, answer, sources retrieved, flag reason, user, timestamp.
- Given an admin resolves a flag, when marked resolved, then they can add a note (e.g., "fixed via re-indexing," "false flag, answer was correct").
- Flag queue should be sortable/filterable by status (open/resolved) and reason category.

A4 — Re-index / refresh data on demand

As an admin, **I want to** trigger re-indexing of new or updated summaries, **so that** newly added meeting data becomes searchable without waiting for a scheduled batch job.

Acceptance Criteria:
- Given new summaries are generated, when an admin triggers re-index, then only new/changed content is processed (incremental, not full re-embed) unless a full rebuild is explicitly requested.
- Given re-indexing is running, when a user submits a query, then the system either serves from the last stable index or clearly indicates data may be incomplete — no partial/corrupted index states should be queryable.
- Re-index completion (success/failure, item counts) should be logged and visible to the admin who triggered it.

A5 — Configure access control by attendee/department

As an admin, **I want to** configure and audit access rules (organizer/attendee-only for users, full access for admins), **so that** users only ever receive information from meetings they were part of.

Acceptance Criteria:
- Given a user queries the chatbot, when retrieval executes, then the metadata filter (attendee list OR organizer match on user ID) is applied at the retrieval layer, before any content reaches the LLM context window.

- Given a user is added/removed from a meeting's attendee list after the meeting occurred, when access is recalculated, then their retrieval access updates accordingly (define whether access is based on live attendee list or a snapshot at meeting time — this needs an explicit decision).
- Given an admin queries, when no filters are applied, then they retrieve across all meetings without attendee restriction.
- Access control logic must be centrally enforced (e.g., in the retrieval service), not duplicated/reimplemented per client, to avoid inconsistent enforcement.

A6 — Monitor hallucination / low-confidence rate

As an admin, **I want to** monitor the rate of low-confidence or potentially hallucinated answers, **so that** I can catch retrieval quality degradation early.

Acceptance Criteria:
- Given each query is processed, when logged, then a confidence/relevance score is stored alongside the answer.
- Given a dashboard view, when accessed, then admin sees trends: % low-confidence answers over time, broken down by department/project if possible.
- Given the low-confidence rate crosses a defined threshold, when detected, then an alert/notification is triggered to the admin (define channel: email, dashboard badge, etc.).

Open Decisions Needed

1. **Attendee snapshot vs. live list** — does access follow who attended at meeting time, or current org/team membership?
2. **Confidence threshold** — what similarity/relevance score triggers "insufficient information" fallback?
3. **Session timeout** for multi-turn context.
4. **Full re-index vs incremental** — default behavior and who can trigger a full rebuild (cost/performance implications).

Revision #2
Created 29 July 2026 07:47:17 by N8N
Updated 29 July 2026 07:47:43 by EMB