ChatMaestro — Domain & Technical Glossary

Revision: 2026-09-17

This page defines in plain language the **technical vocabulary** used across the ChatMaestro docs, covering language-model terms, embeddings and search, database and Postgres, and the auth and runtime platform. Every acronym is spelled out, and where it helps a line says *how the term is used in ChatMaestro*.

This is the companion to `JARGON-AUDIT.md`, which covers the informal idioms (*footgun*, *load-bearing*, and friends). **Entity and table names** — `experiment`, `experiment_run`, `cohort`, `chat_round`, and the rest — are defined in the **module ERDs** and in `schema.dbml` itself, not here; this page is the shared concepts, not the schema.

CONTENTS

Language Models & Prompting

Term	Meaning
attendee / attending	A *derived* status, not a stored column: an enrollee who is taking part in a run right now — `run_enrollment` that is not withdrawn and whose `last_activity_at` is within a recency window. The live monitor shows two counts per run — everyone **enrolled**, and the subset **attending**. Distinct from "enrolled," which is fixed at launch; attending drifts live.
blinding / blinded	Keeping enrollees (and sometimes human raters) unable to tell which experimental condition they are in, so their behavior and ratings are not biased. Enrollee handles are anonymized for this reason.
chat memory	The earlier turns of a conversation replayed to the model on later rounds, so it can answer a follow-up such as "shorter" that means nothing on its own. Switched on per experiment (`experiment.enable_chat_history`); the depth is a platform constant bounded by the **smallest** context window among the model set-up's members, so every arm of a comparison gets the same.
chat session	One continuous conversation between an enrollee and the assistant, made of numbered rounds. Clearing the chat starts a new session and the assistant remembers nothing of the previous one, while a reconnect continues the same session.
completion / generation	The model's output text for a given prompt.
condition	A ChatMaestro term for the fixed, shared set-up of an experiment — the prompts, scenario, opening message, context documents, and scoring configuration that every run of that experiment holds constant. Runs vary the four per-run variables (nudge, model configuration, cohort, Socratic mode) *on top of* one shared condition, which is what makes them comparable within the experiment.

Term	Meaning
context window	The maximum number of tokens a model can consider at once (prompt plus response combined).
enrollee	A ChatMaestro term (and the `profile.role` value) for a person who takes part in studies — the one who does the chatting and whose answers are collected. An enrollee is enrolled into a run through a `run_enrollment` record, and into a reusable group through `cohort_member`.
inference	Running a trained model to get an answer — as opposed to *training* the model in the first place. Everything ChatMaestro does at runtime is inference.
LiteLLM	The open-source library the middle tier uses to call every model provider through one uniform interface, so chat, judging, and embeddings all route the same way regardless of vendor — hosted (Anthropic, OpenAI, Google) or self-hosted (Ollama, vLLM).
LLM (large language model)	The model that generates text — the "chatbot" engine. It is, in effect, a very large statistical function trained on huge amounts of text: given some text, it predicts likely continuations.
medoid	The member of a group that sits closest to all the others — a real item, never an average. `consensus` returns the medoid of the winning cluster, so the answer shown is one a model actually produced.
nudge	A ChatMaestro term: a short, enrollee-safe steering message shown to enrollees, and one of a run's four per-run independent variables (alongside the model configuration, the cohort and Socratic mode — the things a study deliberately changes between runs).
prompt	The input text sent to the model. ChatMaestro distinguishes the **system prompt** (standing instructions, hidden from the enrollee) from the **user prompt** (the enrollee's message).
prompt cascade	ChatMaestro's layered resolution of the system prompt: the experiment base, then the per-model override — the override choosing `inherit`, `replace`, or `append`. The fully resolved prompt is not stored; because the experiment and model configuration are immutable while a run references them, it re-resolves to the same value every time.
prompt enhancement	Rewriting an enrollee's message into a better-engineered prompt before the models answer it. Both versions are kept — the rewrite the models saw and the enrollee's own words — and scoring measures the answer against the enrollee's words, so a rewrite that drifted from the request scores worse rather than better.
provider / model-id	The vendor (`anthropic`, `openai`, `google`) and the exact model string (`claude-opus-4-8`, `gpt-5`). One API key per provider unlocks all of that provider's models.
reasoning model	A model that performs internal step-by-step "thinking" before answering (for example the OpenAI o-series and GPT-5-class). These often reject a custom `temperature`.
rewind	Discarding the tail of a chat session from the assistant's memory while keeping the earlier part, as against clearing the chat, which discards the whole conversation. Neither deletes anything: the rounds stay in the transcript and still count in the run's statistics.

Term	Meaning
Socratic mode	A run setting that decides how the sufficiency gate replies to a request it has stopped: with it on, the trusted judge asks for one missing piece at a time and the request is assembled through dialogue; with it off, the judge names everything missing and asks the enrollee to rewrite the request and resend it. It is switched on per run (`experiment_run.socratic_enabled`) while both phrasings are frozen on the experiment, so a series can compare the two ways of repairing a request without anybody editing prompt text. The models under study never ask anything themselves.
system prompt	The standing instruction that sets the model's role and behavior, hidden from the end user. In ChatMaestro it is the base of the *prompt cascade*.
temperature	A randomness dial for generation, usually 0–2. 0 is deterministic and repeatable; higher is more varied and "creative." Some reasoning models reject any value other than 1, which is why ChatMaestro keeps a per-model capability skip-list.
token / tokenizer	A token is the unit a model reads and writes — roughly ¾ of a word (a word-piece). The tokenizer is the component that splits text into tokens. Billing, limits, and cost are all counted per token, and different models tokenize the same text differently, which is why ChatMaestro tracks tokens per model call, not per round.

Ensembles, Judging & Scoring

Term	Meaning
capability text / expertise	The one curated sentence describing what a catalog model is good at (`model_catalog.expertise`), written as prose because the MoE router matches it against the prompt *by meaning*. It is distinct from the model's `aptitudes` tags, which are short labels for filtering a picker and are too thin to route on.
combine method	How several generator models become one answer: `single` (the one member answers, skipping the ensemble — required for a one-member config), `synthesize` (the highest-weight member merges all answers), `vote` (the generators peer-score on the rubric; the highest weighted tally wins), `consensus` (the answers are grouped by meaning and the largest weighted cluster's most central answer wins), `judge_best` (the experiment's trusted judge scores all and the top wins), `moe` (a router picks one member to answer and the others are not called). The first five fan out and then reduce; `moe` routes and calls one.
declined verdict	The judge's record that an answer refused the request rather than attempting it, marked *appropriate* or *unwarranted*. It exists because refusing and failing look identical to a scorer otherwise: an appropriate refusal takes relevance to not-applicable instead of scoring it badly, while an unwarranted one stays scored.
ensemble	Running several models on the same prompt and fusing their answers into one, rather than relying on a single model. *How* they are fused is the combine method; the members and the method are named in a `model_config` .
escalation	Falling back to a more expensive but better-informed decision only when the cheap one is inconclusive. ChatMaestro escalates routing from embeddings to the judge on a tie, so the LLM call is paid for on the ambiguous minority of prompts rather than on all of them.

Term	Meaning
final-answer measure	The score distribution over each session's last round rather than over every round (`run_result.final_answer_stats`). It says where a session ended up, which a long one can drift away from; read it beside the all-rounds figure, never instead of it.
groundedness / relevance / coherence / instruction-following	The **default rubric's** four scoring factors. Groundedness = supported by a designated source (context or reference in the prompt), not fabricated, and scored only when a source exists; relevance = actually addresses the prompt; coherence = internally consistent and well-structured; instruction-following = did what was asked (format, length, constraints).
LLM-as-a-judge (also written *LLM-as-judge*)	The industry-standard, **reference-free** evaluation method ChatMaestro uses: a large language model scores each answer against the rubric's criteria, with no human-written gold answer needed, instead of or alongside human raters. The default rubric's four factors, listed under *groundedness / relevance / coherence / instruction-following*, are standard LLM-as-a-judge criteria.
MoA (mixture of agents)	Combining several models' answers and optionally feeding the best or aggregate answer back for another improvement pass, dropping ("pruning") weak performers. This is the mechanism behind ChatMaestro's self-improvement loop.
MoE (mixture of experts)	A routing method where a *router* picks one expert model to answer a given prompt, instead of running them all. ChatMaestro's `moe` combine method routes in two stages: **embedding similarity** against each member's capability text decides when one expert is a clear fit (free, no model call), and a **tie escalates** to the experiment's trusted judge, run at temperature 0, to predict which tied contender will answer best. One generation per round, whatever the member count. Note the name collision: *architectural* MoE (expert sub-networks inside a single model, as in Mixtral) is a different thing the platform never touches.
MoJ (mixture of judges)	A deferred, future concept, not used today: an ensemble of judge models whose scores are combined, to reduce any single judge's bias. ChatMaestro currently uses one trusted judge per experiment; a judge panel is a possible later enhancement (the prompt-eval / ensemble phase).
RAG (retrieval-augmented generation)	Feed the model relevant text retrieved from your own documents so its answer is grounded in that material, not only in its training. ChatMaestro uses RAG to ground the enrollee chat in the experiment's context documents.
router / routing decision	The step that chooses **which** expert answers, before any answer exists — as opposed to a combine step, which chooses **among** answers that already exist. ChatMaestro's router is the experiment's own trusted judge, so no extra model is configured, and every decision is recorded per round (`chat_round.moe_routing`) with the stage that made it. Recording it is what makes the router a measurable experimental variable rather than a black box.
rubric	The scoring criteria (factor definitions + anchored 0–100 scales) the judge applies, an immutable, versioned artifact. The rubric — and its factor set — is **configurable per experiment**: the built-in default has four factors (groundedness, relevance, coherence, instruction-following) plus a weighted composite, but an experiment may pick a rubric whose criteria suit its study (say funniness, originality, timeliness for a joke experiment). Rubrics are defined as **versioned configuration files bundled with the middle tier** (the *rubric registry*), not stored in the database; `experiment.score_rubric_version` names which one an experiment uses, chosen from a drop-down. Frozen per experiment, so all its runs are scored the same way and stay comparable within it.

Term	Meaning
self-improvement loop	Repeatedly refine-then-score an answer until a stop condition is hit: a good-enough score (`score_target`), a too-small gain (`min_score_gain`), or a pass cap (`max_iterations`).
self-preference bias	A judge model's tendency to favor answers from its own family or provider. The launch form warns when the trusted judge shares a provider with, or is the same model as, a generator it will score — a warning rather than a block, since that overlap is sometimes exactly what a study means to observe.
sufficiency gate	A check the trusted judge runs before any model answers, deciding whether a request can be answered at all and replying when it cannot — with a question, or with a request to rewrite, according to the run's Socratic mode. A turned-back attempt is deliberately not a round, so a run's round count, cost per round and score distribution describe answered questions only.
turned-back attempt	A message the sufficiency gate stopped before any model saw it. It writes no round and is scored on nothing; the attempt, why it was stopped and the reply it drew are kept on the round it eventually led to, and the count of them (`chat_round.retry_count`) is how a series measures which style of reply gets an enrollee to an answerable request faster.
swept variable	One of the four things a run is allowed to vary: the `nudge` , the `model_config` , the `cohort` , and the `socratic_enabled` switch. Everything else is frozen on the experiment. The term comes from experimental design — you *sweep* one parameter through its values while holding the rest fixed, so any difference in the results traces to that parameter. A comparison holds three of the four constant and varies the fourth.
trusted judge	The **one** judge model named on each experiment (`score_judge_*`) that does *all* the scoring: it scores candidate answers to pick a round's winner and to drive the self-improvement loop, and the winner's factor scores are the round's analytics scores. Because the same judge and rubric grade every round of every run, the runs are comparable. One judge, one scale, no second scoring pass.

Embeddings & Semantic Search

Term	Meaning
chunk / chunking	Splitting a long document into smaller pieces before embedding, so a search returns the relevant slice rather than the whole file. One source row can map to many embedding rows, one per chunk.
cosine (distance/similarity)	The measure used to compare two vectors, based on the angle between them. It is fixed on the vector index and not stored per row.
embedding	A list of numbers (a *vector*) that represents the *meaning* of a piece of text, positioned so that similar meanings sit near each other. That nearness is what makes "find similar text" possible: close vectors mean similar meaning.
embedding model	The model that turns text into an embedding (for example `openai/text-embedding-3-large`). Vectors from *different* embedding models are not comparable, so the producing model is recorded on every row.

Term	Meaning
GIN (generalized inverted index)	The Postgres index type used for full-text search and for JSONB containment queries (`@>`).
HNSW (hierarchical navigable small world)	The graph-based index that makes nearest-vector search fast (approximate but very quick). ChatMaestro has one shared HNSW index over all content vectors, plus one on the saved-question vectors.
keyword / full-text search	Matching literal words, backed by a Postgres GIN full-text index. It stays available on every text column alongside semantic search, so a short note is reachable either way.
Matryoshka / MRL (Matryoshka representation learning)	A training technique that lets an embedding be *truncated* to fewer dimensions and still work. It is how a larger model's output is shortened to ChatMaestro's fixed 1024 dimensions.
OCR (optical character recognition)	Reading machine-text out of an image or a scanned PDF, so a document that is only pixels becomes searchable, embeddable text. ChatMaestro runs OCR during document ingestion for scanned files.
pgvector	The Postgres extension that adds the `vector` column type and the vector indexes (HNSW). On Supabase it installs into the `extensions` schema.
semantic / similarity search	Finding text by *meaning* — the nearest vectors to a query vector — rather than by exact keywords. Complementary to keyword search, never a replacement for it.
top-K	Return the K nearest matches (for example the top 5 most similar document chunks).
vector	The array of numbers that is the embedding. ChatMaestro fixes every vector at **1024 dimensions** so they are all comparable on one index.

Database & Postgres

Term	Meaning
canonicalization / canonical prompt	The normalized wording a saved natural-language question is stored under, so a differently phrased version of the same question still matches it. It is produced by substituting the values the agent bound and then validating one rewrite against mechanical checks, falling back to the substituted form when any check fails.
CDC (change-data-capture)	Streaming row-level changes out of a database as they happen, used to keep an external index in sync with a remote source.
CHECK constraint	A rule on a column or row that the database enforces on every write (for example: a cohort target is set exactly when the message kind is `cohort`).
constraint trigger (deferred)	A trigger that enforces a rule and can be checked at the *end* of a transaction, so a whole set of related rows can be written before the rule is validated.

Term	Meaning
`count(*) OVER()`	A window-function trick that returns the *total* row count alongside each row of a single page, so a paged query does not need a second query just to count.
DDL (data definition language)	The `CREATE` / `ALTER` statements that define the database structure, generated here from `schema.dbml`.
DENORM (denormalization)	Deliberately storing a duplicate of some data (for example a copied email in `created_by_email`, or `audit_log.actor_label`) for performance or to preserve provenance. ChatMaestro marks each one "DENORM (safe: …)" to note it can never drift, because it copies a value that is either immutable or intentionally captured at a point in time.
immutable-while-referenced	ChatMaestro's fidelity rule: while any run references a definition (an `experiment`, `nudge`, `model_config` and its models, or `cohort` and its membership), that definition's substantive fields are read-only and it cannot be deleted (`ON DELETE RESTRICT`), with only `notes` and `run_defaults` left editable. This keeps a finished run's exact set-up recoverable by reading the definitions directly, so no per-run copy (snapshot) is needed. To change a locked definition, you clone it.
jsonb	Postgres's binary JSON column type, for flexible, schemaless data (preferences, audit before/after states, per-factor score breakdowns).
keyset pagination	Paging through results by "everything after this key value" instead of `OFFSET N`, so that deep pages stay fast no matter how far in you are. Also called cursor pagination. ChatMaestro uses this — over a `(created_at, id)` index — as the sole deep-paging mechanism on its high-volume tables; range-partitioning them is a deferred scaling option, not part of the current schema.
NL→SQL (natural language to SQL)	Turning a plain-language question into a database SQL query with an LLM. This is how ChatMaestro's *Ask* engine answers analytics questions: it compiles the question to read-only SQL, checks it (with `EXPLAIN`, without running it), then runs it under the asker's own permissions.
OCC (optimistic concurrency control)	A scheme that stops two users from silently overwriting each other's edits: each row has a `version` counter, and an update only succeeds if the version still matches the one that was read. It is applied deliberately, only to rows two people can co-edit through a form.
ON DELETE: cascade / set null / restrict / no-action	What happens to a child row when its parent is deleted: also delete it (cascade), null out the link (set null), block the delete (restrict), or the default (no-action).
partial (unique) index	An index — often a uniqueness rule — that applies *only* to rows matching a `WHERE` predicate (for example "exactly one admin," or "one saved-question label per owner among live rows").
PK / FK / UK	Primary key (the row's unique identifier), foreign key (a reference to another table's row), unique key (a column or set that must be unique).
RLS (row-level security)	Per-row access rules enforced by the database itself, based on who is asking: the database narrows every query to just the rows the current user may see. ChatMaestro's admin > experimenter > enrollee ranking is enforced here, not only in app code.
schema (namespace)	A Postgres namespace that groups tables under a name (ChatMaestro uses `chat_maestro`). Note the collision: "schema" also loosely means "table design" — this glossary means the *namespace* sense.

Term	Meaning
surrogate key / UUID	A system-generated identifier used as the primary key instead of real data. ChatMaestro uses UUIDs (universally unique identifiers) throughout, so joins never depend on mutable data like email.
trigger / trigger function	Database code that runs automatically on insert, update, or delete. ChatMaestro uses triggers only for the embedding lifecycle and one cross-row check; auditing is deliberately app-driven, not trigger-driven.

Auth, Platform & Runtime

Term	Meaning
CDN (content delivery network)	A globally distributed set of servers that deliver static web files (HTML, JavaScript, images) from a location near each user, for speed. ChatMaestro's front end is served from the Cloudflare Pages CDN.
connection pool	A set of reused database connections, so each request does not pay the cost of opening a new one.
Data API (PostgREST)	Supabase's auto-generated REST interface that *would* let a browser query Postgres directly. ChatMaestro turns it **off**, so the middle tier is the only path to the database and every access passes through its checks.
JWT (JSON Web Token)	A signed token, issued by Supabase Auth at sign-in, that proves who is logged in. The browser uses it three ways: as the bearer token on its HTTPS calls to the middle tier; to open the Realtime WebSocket, where it authorizes *which live channels* the browser may subscribe to (keyed to the enrollee and run); and — forwarded by the middle tier to Postgres via `request.jwt.claims` — to make RLS apply to that user. It is an identity token, not database-specific authorization.
middle tier	The persistent Python service (on fly.io) that holds the LLM API keys, is the *only* path to the database, and runs the chat orchestration. Browsers never talk to the model providers or the database directly.
OAuth / OTP	Sign-in via a third-party provider (Google/Microsoft/Apple) / a one-time passcode emailed to the user. ChatMaestro's sign-in is OAuth-first with an email-OTP fallback — no passwords.
opting out	An enrollee's own withdrawal from the study pool, which ends every study they are currently in, stops further invitations and keeps them out of future ones. It is pool-wide because joining was, it is theirs alone to trigger, and it is distinct from an account being switched off by staff.
R2	Cloudflare's object storage, which exposes the same **S3-compatible API** as Amazon S3 (so standard S3 tools and libraries work against it). ChatMaestro keeps uploaded files, extracted-text snapshots, and database snapshots here, referenced from the database by key rather than stored as blobs in it.
Realtime (Supabase Realtime)	Supabase's service for pushing live messages to the browser over a WebSocket. ChatMaestro uses it for the live run stream an experimenter watches and the operator messages an enrollee receives; each connection's JWT authorizes which channels it may subscribe to. Realtime is the one Supabase-specific piece a future migration off Supabase would have to replace.

Term	Meaning
service-role key	A privileged Supabase key that *bypasses* RLS. ChatMaestro uses it only for admin operations, never on high-volume user paths, because bypassing RLS there would defeat access control.
SPA (single-page application)	The browser front end (a Cloudflare Pages app) that talks only to the middle tier. It is called *single-page* because the browser loads one HTML page once and JavaScript rewrites its content in place as you navigate, instead of fetching a whole new page from the server for each screen.
Supabase	The managed platform providing Postgres, authentication, and realtime. ChatMaestro uses it for those managed pieces only; its Data API is off, since the middle tier is the sole database path.
Supavisor	Supabase's connection pooler. Because the middle tier holds its own pool, Supavisor can run in session mode and prepared statements stay on.
WebSocket	A persistent, two-way network connection kept open between browser and server so the server can *push* data the instant it happens, with no repeated polling. In ChatMaestro the browser holds one such connection — to Supabase Realtime — and only *receives* on it (the live run stream and operator messages); the middle tier is what publishes to it, and everything the browser sends travels to the middle tier over HTTPS.

This page is hand-kept, like the jargon reference. Add a row when a new technical term enters the docs, and keep entity and table definitions in the schema and its ERDs rather than here.