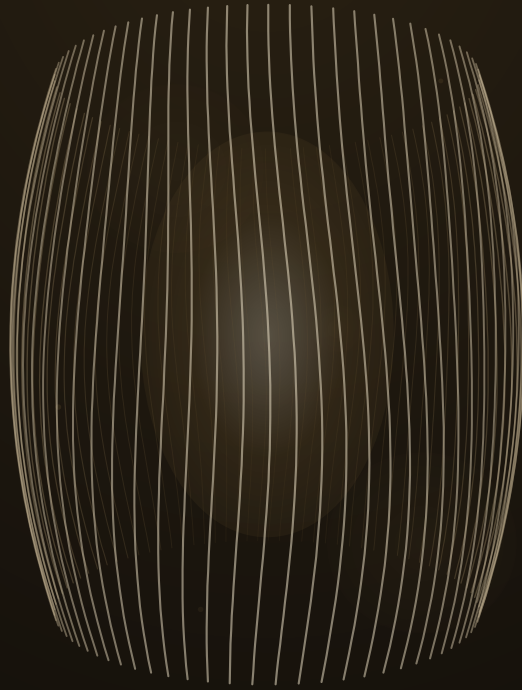


UNDERSTANDING WEAVE



How it *really* works.

For the reader who suspects this is a language model in a trench coat. The architecture at the depth needed to judge it, honest throughout about what runs today and what is built and staged for tomorrow.

WEAVEDEEP.COM

THE OBJECTION

“Isn’t this just a language model with a search index?”

It is exactly the right suspicion, because most of what is sold as enterprise AI is precisely that, and you should not take a denial on faith. A language model is a component Weave uses, the way a car uses an engine; it is not the car. The difference is the whole subject here: how knowledge is held, how an answer is grounded, what the model is and is not permitted to see, how its output is checked, and, over time, how the arrangement is designed to get better on its own without leaving the boundaries it is allowed.

One discipline runs through what follows. What is live today is said in the present tense. What is built but deliberately not yet switched on is called staged. What is decided and designed but not yet built is called designed. A system that cannot tell you, to the part, which of these each of its capabilities is, is not one to trust; so this is written to tell you.

MEMORY

A structured record, not a pile of text.

Weave does retrieve, so be exact about the difference. An ordinary retrieval system keeps your documents as searchable text and hands a model the fragments that look relevant. Weave instead reads the documents once and writes what they establish into a structured record: organised facts in databases the organization owns, with the relations between them kept. Dates are held in three states and never flattened: certain, ambiguous (every reading the source allows is preserved, not one silently chosen), or absent. Neither a structured record nor checking a claim against it is new in the research literature; what Weave claims is not novelty of the parts, but that both run over a store the organization owns and can correct, and from which it could walk away with everything intact.

THE MIND

It answers, then checks the answer against the record.

A question does not simply go to the model. Weave retrieves the records that bear on it against a fixed budget, pinning coverage, genuine disagreement and anything the question named so nothing crowds them out; it builds the prompt from that; the model answers; and then the answer is read back, claim by claim, against the record. Rule-based checks cover anchors like dates, money and percentages, ownership, disagreement, undisclosed conflict, denial, ordering and derived attribution; a paid semantic pass runs on top for the figures, commitments and negative or completeness claims a fixed rule cannot safely judge. A failing claim is sent back once; whatever still fails is cut rather than shipped. Its honest boundary: a fluent claim that contradicts nothing on file

can still pass, and the loop checks the answer against what was extracted into Memory, not against the original page — a gap a separate fidelity instrument exists to measure.

WHAT MAKES IT AN ARCHITECTURE

No single model is allowed to be load-bearing. The constitution forbids pinning Weave to one model: no model is named in a prompt, a policy maps each task to models that have passed Weave's own evaluation, and failover never reaches one that failed. That is the design. Today, production calls a single provider with no failover; the module that asks for a job rather than a model, the policy, and a second provider's adapter are built and staged on a branch, held until after the first live pilot. Staged, not yet running.

Provenance is a property of the pipeline, by design. Every output is meant to carry how it was made, stamped from outside the model, because a model's report of its own identity is worthless as evidence. What runs today is narrow and real: a record, kept beside the brain, of which model answered each question. The fuller stamp — provider, the model and policy and prompt versions, a timestamp — is built and staged; an integrity hash over the output, so a later edit no longer matches its record, is designed, not yet built. The honest limit is stated rather than hidden: such a hash proves integrity within Weave's own trust boundary, not that a named provider served it.

The sensitive is meant to be sealed before the model sees it. The design seals sensitive substance out of the material before the model reads it, so under enforcement there is nothing for a clever prompt to coax back. The honest default today is report mode: the seal detects what it would withhold and names it, but leaves it in place until a deployment turns enforcement on. So by default the sensitive is flagged, not yet removed — and Weave measures what its own sealing would miss, including a sealed figure reconstructable from open ones.

The decisions that matter are reserved for a person. Five acts are held permanently human: the review of what is sensitive, the promotion of any method or model into production, staging, accept-or-reject, and any unapproved write to a real brain's memory. Today only one — the promotion of a made artifact — is enforced in code, and even that authenticates a brain's key rather than a named person; the other four are held by operator discipline and credential separation. The constitution names them permanent; turning each into an enforced gate is work in progress, named, not papered over.

A demo shows you its best moment. An engineered system can show you its edges — and a paper that marks what is staged and what is merely designed is doing exactly that.

BUILT TO GET BETTER

Designed to improve itself, and standing still on purpose.

First, plainly: today Weave does not improve itself autonomously at all. It checks with a person on essentially everything. What its constitution makes the default is continuous improvement — any process, prompt or routing rule is a standing candidate to be made better — and two mechanisms are designed to carry that: a lesson learned once and inherited everywhere it applies, and a model substrate that evaluates new candidates and drafts changes to its own routing. Both are designed, not built. What is live is only the narrow, real thing: a correction you make lands in Memory, and the next answer reasons over the corrected record. The capacity to improve is constitutional and partly built; the permission to exercise it unattended has not been granted, by design, and sits behind a fence of permanent human gates, reversibility, audit and a capped budget.

WHAT IS PROVEN, AND WHAT IS NOT

It can be wrong. The difference is what happens next.

Weave holds a rule unusual in this field: no capability is called proven without a named artifact that proves it, and that rule cuts against its own story. Weave publishes a computed account of itself that marks, for each of its own claims, whether an instrument is watching it or whether nothing yet is, and names the instrument each uninstrumented claim still needs — a count kept in that live account rather than frozen onto a page. It keeps two kinds of evidence apart: that it put no wrong thing in, checkable on live data without an answer key; and that it left no right thing out, measurable only against a known answer key in a lab. The first is never allowed to stand in for the second. Weave can still be wrong, as any system can; the difference is that the failure is auditable, fixable, and then relied on — more than can be said for a person's memory or a model's confident guess.

You do not have to believe any of this from a description. The point of the architecture is that it is inspectable: every claim here is one you can be shown, and every label — live, staged, designed — is one you can check against the running system. That is the only kind of trust worth having in something you would hand your real work.