



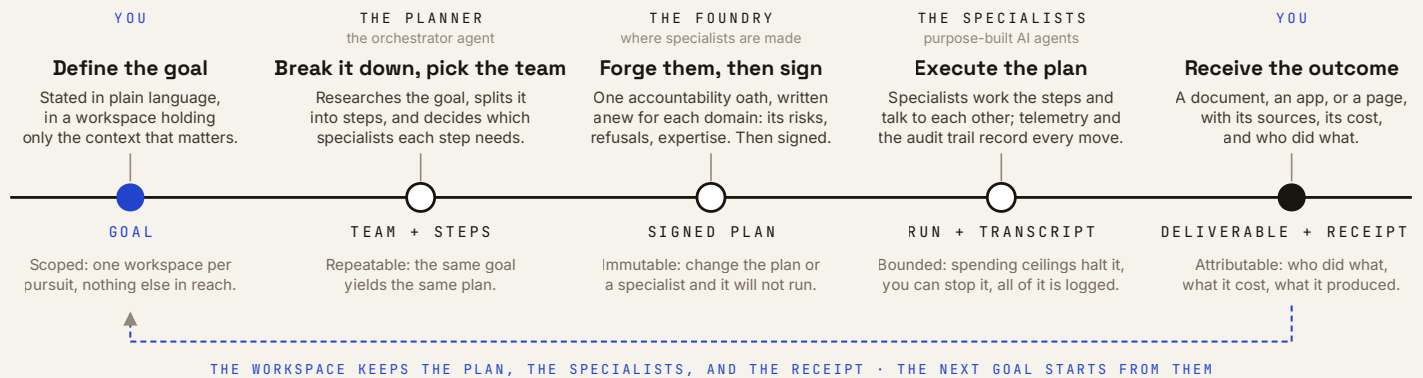
AGENTIC ENSEMBLE · CAPABILITIES

A governed AI workforce. Every action has an author, every plan a signature, every outcome a receipt.

Agentic Ensemble is a self-hosted platform that runs teams of purpose-built AI specialists on signed plans, inside isolated workspaces, with a human in command and a ledger under everything. Built for the operator who wants a workspace that compounds, the builder who wants a governed engine behind their product, and the enterprise whose security team has to sign off.

THE CORE LOOP · WHO ACTS, WHAT EXISTS, WHAT IT GIVES YOU

You describe what you need. The platform plans it, staffs it with purpose-built specialists, runs it under your limits, and returns the result with a record of every step.



Worked example. Goal: read a 40-file data room and write a diligence memo. The planner cast an analyst, a contracts reader, and a writer and signed a six-step plan. The memo came back with every source cited, every step in the transcript, every dollar attributed. Next quarter: same plan, new files.

THREE THINGS THAT ARE TRUE OF EVERY RUN

Work persists and compounds.

Every pursuit gets its own workspace where files, deliverables, decisions, and specialists accumulate across runs, versioned like code. Reset the tree; the specialists survive.

PROOF · CONTENT STORE PER
WORKSPACE

Nothing runs unsigned.

A plan is signed against the exact specialists it names before any agent moves; change either and it is blocked. You can stop or kill any run, ceilings halt it, and packs that declare decisions wait for a human.

PROOF · UNSIGNED GRANTS: 0

Nothing leaves unrecorded.

Keys stay in the vault and are injected at the gateway; the model never holds one. Outbound traffic is allow-listed per vendor. Every action, cost, and decision lands in a hash-chained ledger.

IN PRODUCTION, NOT ON A ROADMAP

Since August 2026	A reference node runs the full stack on AWS, every enforcement toggle published with its live value.
Red-teamed	Specialists forged for our own code hunt its vulnerabilities; because plans are schedulable, that review can run as continuous monitoring.
Signed releases	Releases ship by signed, verified digest; the pipeline's own checks caught two misconfigurations before any user did.

GRANTS TO AN UNSIGNED PLAN

0 by construction

AUDIT EVENTS ATTRIBUTABLE

100 %

SECURITY PILLARS IN CODE

10 of 10

THREE WAYS TO START · MECHANISMS OVERLEAF

Use it. Integrate it. Roll it out.

Use it: a workspace, a specialist, a goal. **Integrate it:** your product on the API with a scoped key. **Roll it out:** a workspace per team, your workflows as signed plans.

Andrew Wright

ASSISTANT@AGENTICENSEMBLE.AI

REQUEST EARLY ACCESS



HOW IT IS BUILT

Nine ways chat-thread AI fails, and the mechanisms we use against each.

One line per mechanism, all shipped and demonstrable on a live node.

CONTEXT BLOATS

- **Resettable workspaces.** A variant, the platform's isolated workspace, is built from a baseline and can be destroyed and rebuilt from it, or promoted into a new baseline. Persona identity is canonical and signed, and the operator's teaching overlay sits outside the signature, so the tree resets while the agents and what you taught them survive.
- **Rolling memory.** Past 20 messages, a session's oldest half is summarised and archived; later turns load the summary, not the raw thread. A new session starts empty, so nothing bleeds across sessions or workspaces.
- **Per-step context.** A step receives only the handoffs its plan names; numbers travel through structured stores, not prose.
- **Collapsed deliverables.** A rendered document or page is replaced in later turns by a compact reference instead of being re-sent and re-billed.
- **Per-turn tool scoping.** A step loads only the tool servers it is granted.

WORK VANISHES

- **Versioned content.** A git-backed content store snapshots human-authored content per workspace and baseline, excluding agent runtime state; each tenant gets its own repository.
- **Durable runs.** Every run keeps a transcript that survives a browser reload, and its apps, documents, and pages are tied to the run id so delivery can attach them.
- **Persona journals.** Each specialist's contributions are recorded per plan and readable afterwards.

SECRETS REACH THE MODEL

- **Vault and runtime files.** Credentials stay in the vault; a runner reads a per-integration runtime file that is rewritten on rotation, so a changed key applies on the next call without a restart.
- **Scrub at spawn.** A fixed list of platform secrets can never reach a child process.
- **Gateway injection.** The egress gateway allows platform essentials plus the active packs' hosts and denies the rest; provider keys are added to outbound requests at the gateway, never inside the model.
- **Injection defences.** Untrusted content is sanitised before it is inlined into a prompt; shell tool calls are scanned for credential theft, exfiltration, and privilege-escalation patterns. Run bundles are envelope-encrypted at rest.

COST RUNS UNCHECKED

- **Hard ceilings.** A dollar ceiling per run and per tenant and a cap on concurrent workers, all host-side and halting.
- **Priced approvals.** A pre-run cost estimate turns the approval gate into an economic decision.
- **Ledger and invoice.** An append-only token ledger per run, a monthly breakdown by persona, plan, and model, and a per-instance workspace quota.

NO ACCOUNTABILITY

- **Forged specialists.** The Foundry takes a domain brief and writes a full persona: an oath naming that field's real temptations, testable commitments, failure patterns, and competencies each bound to the oath. Its content hash is what the plan signature binds.
- **Named actors.** Every audit entry names the run, the persona, and the human; every approval carries the operator's name.

NO HUMAN IN THE LOOP

- **Stop and kill switch.** An operator stop marks the run halted before anything is torn down, so nothing respawns; spend ceilings halt a run rather than warn.
- **Approval queue for packs.** A plan that declares a decision schema submits cards to the Decision Inbox, and a named human approves or rejects; what an approval may do is a closed vocabulary the agent cannot extend. Tokens are single-use and expire with the proposal's window.

NO AUDIT TRAIL

- **Hash-chained ledger.** Each entry hashes its predecessor; reorder, delete, or edit and verification names the first broken entry. Entries can be HMAC-signed, and a daily append-only file is kept as a second sink.
- **Scrubbed before signing.** Secrets are removed before an entry is signed or stored.
- **Erasable on request.** Run transcripts support a per-run erase for data-subject requests without touching the chain.

RUNS DRIFT

- **Two kinds of drift.** Verification tells plan drift from persona drift and blocks either before a run starts.
- **One pre-flight.** Scheduled runs pass the same check as a manual run, so the two paths cannot diverge.
- **Re-sign, never accept.** A changed persona triggers a re-sign flow; an unsigned plan that requests any capability is refused.

BUILDING A PLATFORM, NOT A PRODUCT

- **A governed API.** Versioned routes with machine keys, per-tenant scoping, rate limits, and scope checks on every call.
- **One image, signed.** The same image serves development and production; CI builds and signs it, the deploy verifies it, and only git reaches a box.
- **Posture you can read.** A Security Posture view lists every enforcement toggle with its live value, so a buyer sees the hardening state instead of taking a deck's word for it.

HOW WE BUILT IT · PLANNED AND REVIEWED IN THE PLATFORM · WRITTEN BY A CODING AGENT

- **Specified here.** Every feature began in the platform as an intent document a human approved, then an engineering document of build, rewire, delete, and acceptance steps. The plan is committed with, or ahead of, the code it describes.
- **Decided here.** Plans were reviewed against the codebase and amended in place; decisions a builder may not reopen sit in a table at the top; risky parts were spiked first.
- **Written by a coding agent.** A dedicated coding agent wrote the source, working from those documents rather than a chat thread. The platform is for knowledge work: it directed the build and did not type it. Security code follows a ten-pillar standard each module cites.
- **Reviewed here.** Forged security hunters audited the code in three cycles, one document per finding with file, line, exploit path, fix, and test. The record stays after shipping: status tags, handoffs, postmortems.