OPERATING MANUAL · 2026

AI as a team, not a tool.

Everyone is building an AIOS. We built **The AIOS** — the operating system that turns AI into a team of co-workers: a legal you, an accountant you, a marketing you, an engineering you. Orchestrated by you, governed by your intent, compounding every day.

`declared context` `observed context` `intent` `agents` `skills` `rituals`

01 WHAT IT IS

Your AI co-workers, orchestrated.

The AIOS turns AI into a team. You can run every co-worker yourself and be their orchestrator — or let one AI co-worker run the rest as your chief of staff who never sleeps and absorbs the coordination overhead with the best agentic culture.

Built for anyone navigating AI-overwhelming days — **senior executives, builders, founders, operators**. AI alone multiplies confusion. The AIOS gives you the structure (prompt, context, intent, collaboration, second brain) where clarity emerges — so you make the most of AI without losing what makes you irreplaceable, and without IP/PII risk.

THREE PROGRESSIVE STAGES — EACH RETURNS ~10× THE LEVERAGE

WEEK 1 — AUTOMATE

Gain speed, do faster

Daily plans, drafts, syntheses. 30 minutes becomes 30 seconds.

WEEK 2 — AMPLIFY

Gain bandwidth, do more

Agents draft proposals, write in your voice, research while you sleep. You stop being the bottleneck.

MONTH 1 — AGENCY

Gain autonomy, do agentic

AI co-workers act on your behalf with judgment, within trust boundaries you've defined.

FOUR PRINCIPLES, ALL LOAD-BEARING

Amplify intelligence, not artificial

Human + AI beats human alone or AI alone.

Context, not prompts

Prompts are the artifact most people optimize. Context is the substrate that determines what those prompts can do.

Trust earned over time

Autonomy compounds with judgment — like a good A-player on a real team.

Portable, not proprietary

The AIOS is the layer; the LLM is interchangeable. Plug Claude (recommended), Gemini, or your best model.

02 THE ARCHITECTURE

Context that compounds.

Most setups ship declared context only. The AIOS runs a **three-layer context system** — after a month, the AI knows things about you that aren't in any file you wrote. *An empty ground produces a generic answer no matter how brilliant the reader* — the vault is the enriched ground every agent reads.

LAYER 1 — DECLARED

You write it

Identity, voice, working style, business, role expectations. What you explicitly tell the AI about yourself.

LAYER 2 — OBSERVED

Claude writes it

Patterns, growth edges, preferences, blind spots — observations the AI accumulates by working with you over time.

LAYER 3 — INTENT

The trust contract

`INTENT.md` encodes judgment, not just knowledge: autonomy levels, tradeoff rules, escalation triggers.

WHAT MAKES IT OPERATIONALLY DISTINCT

Governed	The INTENT.md trust contract controls what AI handles autonomously vs. what needs your review.
Multiplayer	Personal × team × company topologies. Mount a company's context in one prompt; collaborate over Drive, GitHub, or local folders.
Substrate-agnostic	Filesystem as context, plain Markdown, no RAG. Storage is a plugin — the same commands work on any substrate.
Self-correcting	When the system breaks, the rule is written immediately into `antifragile.md` — every break upgrades the system.
Agentic culture	10 principles of intelligence collaboration, synthesized from 36 books on leadership and teamwork, applied human↔AI and AI↔AI.

> "There are no bad agents, only bad operators." — the agentic culture, principle in practice

03 THE RHYTHM

The rituals are load-bearing.

The daily commands aren't quick-start sugar you can skip — they're the system's nervous system. They route session insights into observed context (so the AI actually gets smarter), close the carry-loop (so nothing falls through), and surface drift before it becomes invisible. **Logging is not routing** — the framework is built on the routing.

THE DAILY LOOP

MORNING
/aios:today
A grounded daily plan from your full vault context — calendar, tasks, open threads, priorities.

PER SESSION
/aios:close-session
Lightweight capture when a focused work session ends — bridges the work back to the vault.

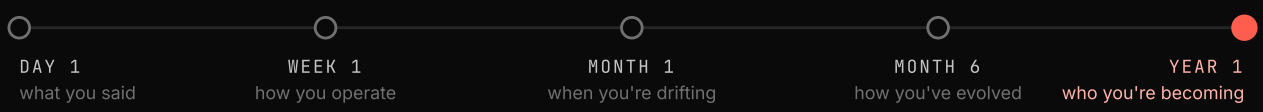
EVENING
/aios:close-day
What shipped, what grew, what carries forward — routed to projects and observed context.

THE FULL CADENCE

CADENCE	COMMANDS	WHAT IT DOES
Weekly	`/aios:7plan` · `/aios:weekly-learnings`	Monday strategy across ventures · Friday compile of the week's insights
Bi-weekly	`/aios:drift` · `/aios:graduate` · `/aios:emerge`	Avoidance audit · promote ideas to permanent notes · surface implied patterns
Monthly	`/aios:compact` · `/aios:housekeeping`	Digest + archive last month · vault hygiene proposals
Quarterly	`/aios:role-report`	Period report against your role pillars, exported as a branded PDF

THE COMPOUND EFFECT

Day one it knows what you said. A year in, it knows who you're becoming.



> Not a productivity hack — compounding self-awareness. You do the thinking. The vault does the remembering.

04 PROJECTS

The vault tells the why. The repo tells the how.

Every piece of work gets one note in `projects/` — a dashboard answering **what is this, what's next, what's blocked**. Say "let's work on X" and Claude zooms in without losing the full picture, because every project note carries a **Current State table**: the router that wires strategy in the vault to execution wherever the work actually lives.

```
› "let's work on acme-ops"
  1 · read projects/acme-ops.md            # strategy — who, why, what's next
  2 · extract the Current State table   # the router — type, paths, stack, status
  3 · cd ~/code/acme-ops                # Type: Coding → go to the repo
  4 · read the repo's CLAUDE.md + .claude/settings.json + README
      # execution rules — architecture, build/test/deploy, permissions
```

TYPE DRIVES THE ROUTING

CODING
Routes to the repo
The repo's own `CLAUDE.md` takes over for execution: architecture, conventions, build/test/deploy. Its `settings.json` scopes permissions per project.

NON-CODING
Routes to the drive
Working files (proposals, docs, sheets) live at the Drive path. The vault note itself IS the deep context — no repo to enter.

HYBRID
Routes to both
Code and documents side by side — the table carries both paths, and both context layers stay loaded.

The result: **two layers active at once** — vault context (strategic: who, why, for whom) and project context (execution: how, where, what's next). And the wiring is self-propagating: every coding project Claude scaffolds is born with its own `CLAUDE.md` + README + settings, so any teammate — human or AI — lands oriented.

NAMED BY CATEGORY, KEPT AS DASHBOARDS

`<venture>—*` `space-*` `advisory-*` `personal-*` `experiment-*` `infra-*` — the prefix tells the kind at a glance; lifecycle lives in frontmatter.

History doesn't pile up in the note: what shipped lives in git log, what was learned routes to observed context, session notes keep only the last five. `/close-day` flags any note growing past dashboard size.

> A project note is a dashboard, not a history book — it answers what's next, never "what happened on March 28."

05 THE TOOLBOX

Describe what you need. The system matches.

You don't memorize commands or file paths. Everything below is available the moment setup finishes — say what you want and your session picks the right tool.

24 COMMANDS

31 AGENTS · 6 BUNDLES

42 BUNDLED SKILLS

10 BUNDLED MCPS

WHAT THE SYSTEM IS MADE OF

THE SHARED SPINE

VAULT	*the second brain*
AGENTS	*a team of specialists*
SKILLS	*capabilities on demand*
COMMANDS	*the daily rituals*
MCPS	*hands into your stack*
HOOKS	*reflexes that fire on their own*

THE EXTENSION LAYER

`aios/` — *shipped to everyone*

`custom/` — *only yours · survives updates*

`<company>/` — *mounted · inherited by your team*

SELECTED COMMANDS BEYOND THE RITUALS

`/aios:ingest` — any URL, PDF, or transcript into the vault

`/aios:connect` — bridges between unrelated domains

`/aios:company` — mount a venture's shared context

`/aios:trace` — how your thinking on a topic evolved

`/aios:ghost` — answer or draft in your actual voice

`/aios:challenge` — steel-man the opposite of your thinking

`/aios:collaborate` — shared spaces on Drive / GitHub / local

`/aios:update` — pull the latest framework, auto-applied

DEEP DIVES — THE REST OF THIS MANUAL

06 **Skills** — capabilities that load themselves

07 **The Fleet** — 31 co-workers in 6 bundles

08 **Connectors** — 10 MCPs + the hooks

09 **Glass** — the graphical front door

10 **Shared workspaces** — company & collaborate

11 **Templates** — start from a shape

12 **Personalizations** — USER.md is yours

13 **Operating Claude** — practices that compound

14 **Containment** — the Fortress

15 **Getting started** — 30 seconds in, current forever

16 **Troubleshooting** — diagnose the system

17 **The horizon** — the orchestrator's flip

06 SKILLS

Capabilities that load themselves.

Skills are reusable expertise packs that auto-load at session start — you never invoke one by name. Describe the work and the right skill wakes: ask for a landing page and `frontend-design` engages; ask to debug and `systematic-debugging` takes over. **42 bundled skills** ship across four source folders, each following Anthropic's open Skills spec.

FOUR SOURCE FOLDERS

SKILLS/AIOS — 17
AIOS-built
Engineering patterns, Obsidian-native tooling, infographics, data presentation, compliance.

ANTHROPIC — 11
Vendored, Apache-2.0
From `anthropics/skills`: frontend design, canvas art, doc co-authoring, MCP & skill builders.

SUPERPOWERS — 14
Vendored, MIT
From `obra/superpowers`: TDD, systematic debugging, brainstorming, plan writing & execution.

SKILLS/CUSTOM
Yours
Operator-built skills. Survive every `/aios:update`, override bundled on collision.

WHAT'S IN THE BOX

Coding & engineering	API design · architecture patterns · Karpathy-style coding · TDD · systematic debugging · React/Next.js · Python · Tailwind · error handling · migrations
Content & design	Frontend design · canvas & algorithmic art · theme factory · infographic builder · data presentation · internal comms · doc co-authoring
Documents & files	Word, PDF, PowerPoint, Excel — create, edit, merge, fill (via Anthropic's document-skills plugin)
Obsidian-native	Wikilinks & callouts · JSON Canvas mind-maps · Bases database views · vault ops from the CLI
Planning & process	Brainstorm before building · write plans · execute with checkpoints · verify before claiming done
Meta	skill-creator builds new skills · mcp-builder builds new connectors — the system extends itself

> You don't pick the tool. You describe the work — the right capability wakes.

07 THE FLEET

Thirty-one co-workers, six bundles.

`spawn {name}` opens a named tab with that co-worker's full identity pre-loaded; `/aios:agent {name}` wears the hat in your current session. Every agent follows the same agentic culture as you — and its work bridges back to your vault via `/close-session`, so nothing a co-worker does is lost.

SALES

Explore and qualify leads, draft proposals from your catalog, keep the CRM honest, watch the market's mentions.

`sales-lead-hunter` · `sales-proposal-writer` · `sales-crm-updater` · `brand-monitor`

STRATEGY

Advisory frameworks, market intel, company deep dives, open-protocol governance.

`consultant` · `market-researcher` · `company-analyst` · `protocol-steward`

FINANCE & LEGAL

Bookkeeping to contract review — a back office that never queues.

`accountant` · `lawyer` · `compliance-checker` · `invoice-tracker`

ENGINEERING

Discovery to shipping — plus the builder that extends the AIOS itself.

`technical-cofounder` · `code-reviewer` · `security-engineer` · `bug-triager` · `code-documenter` · `aios-builder`

COMMUNICATION

Your voice, multiplied across every channel and every deck.

`content-writer` · `deck-builder` · `email-drafter` · `meeting-prepper` · `report-drafter` · `content-scheduler` · `design-md-author`

PERSONAL

The bundle that works on you, not just for you.

`growth-companion` · `study-buddy` · `journal-prompter` · `decision-journaler` · `crisis-mode` · `onboarding-aios`

Your own agents live at `agents/custom/` and survive framework updates; company-distributed agents land at `agents/<company>/` when you mount a venture. The `aios-builder` agent scaffolds new fleet members for you — compliant, registered, and actually loaded.

```
› spawn lawyer "review the NDA at ~/code/contracts/mutual-nda.docx"
› spawn-kill lawyer  # clean teardown when the work is done
```

> "Not a chat box you visit; a team that knows your traces, carries your standards, and works while you sleep."
>
> — The Ground · chuycepeda.com/manifesto

08 CONNECTORS

Where the system meets your tools.

Ten MCP servers ship vendored in the vault — policy: **local over remote**. Each authenticates independently of your Anthropic account, survives account switches, and syncs via git. `/aios:mcps-setup` walks each one: want it? → token → register → verify. Skip any; they sit on disk until you change your mind.

THE TEN BUNDLED MCPS

Google Workspace — Calendar, Tasks, Drive, Gmail, Contacts; feeds the daily rituals

GitHub — repos, issues, PRs, search; powers the engineering fleet

NotebookLM — notebooks, sources, audio summaries — beyond what the UI exposes

Playwright — browser automation: navigate, fill, screenshot, scrape

PDF Generator — markdown → branded PDF; powers every report export

Slack — read, send, react, threads; daily-recap triage for `/today`

Atlassian — Jira sprints & transitions, Confluence pages & search

Stitch — AI-native UI screens from text + DESIGN.md tokens

Nano Banana — Gemini image generation for decks, covers, brand visuals

Spotify DJ — playback + queue; music while you work, via Claude

THE HOOKS — PLUMBING THAT FIRES ON ITS OWN

`inject-datetime`

Stamps the real clock into every prompt you submit. Claude never drifts on what day it is — load-bearing for any time-sensitive reasoning.

`pipeline-executor`

Batch-loads Calendar + Tasks + Slack in one call for `/today` and `/close-day` — the rituals start pre-fed.

`markitdown`

Any file → clean markdown: PDF, Word, Excel, audio (with transcription), YouTube, EPUB. Wired into `/ingest` automatically.

`claude-identity`

Multi-account quota autopilot (macOS) — rotates accounts before rate limits hit and respawns sessions so transcripts continue uninterrupted.

> Connectors live in the vault, not in the cloud — your integrations survive every account switch.

09 GLASS

The graphical front door.

The AIOS is powerful but lives behind a terminal. **AIOS Glass** is the extension that changes that — a glass layer over the framework, running inside VS Code and Google Antigravity, that turns terminal rituals into clickable surfaces. Slash commands become buttons. Arguments become forms. `spawn` becomes a click. Git sync becomes "Sync now." Non-developers operate their full AIOS without ever typing a command.

> Glass, not engine. The extension surfaces and triggers what the framework already does — it never reimplements AIOS logic. It reads the framework's own source-of-truth files at runtime, so every `/aios:update` addition appears automatically.

THE SURFACES

HOME
Dashboard
App-like panel with live cards — your AIOS at a glance, rituals one click away.

CALENDAR
Live month grid
Day-dots from your daily notes; click to open or create. Today card launches the rituals.

AGENTS
Fleet browser
Runtime discovery of every agent, grouped by bundle. Pick agent → describe task → spawned.

CAPABILITIES
Skills · MCPs · plugins
Everything installed, discovered live, one click to its docs.

SPACES & STATUS
Sync as forms
Company and collaborate flows as pickers and inputs — no raw flags. One-click framework update.

ONBOARDING
Guided walkthrough
Eight steps from welcome to first spawned agent — auto-opens on first run.

Under the hood: a TypeScript VS Code extension that depends on Foam (knowledge-graph engine, not forked — it auto-updates upstream) and runs every ritual via **native Claude Code** in an integrated terminal. Full parity with the terminal experience — the same framework, now with a face.

10 SHARED WORKSPACES

One prompt onboards your whole team.

The AIOS is multiplayer by architecture, not by afterthought. Two commands cover the two shapes of working with others — a **company** you build with, and a **collaborator** you share projects with.

/AIOS:COMPANY

Mount a venture's brain

A company lives in a venture-context repo — positioning, personas, pricing, culture, design, brand — plus optional infra: agents, plugins, skills, templates your team has built. A teammate runs `--mount {repo-url}` and their Claude inherits all of it. `--sync` keeps every operator at byte-parity with HEAD.

`--create` · `--mount` · `--sync` · `--sync-all` · `--status` · `--invite`

/AIOS:COLLABORATE

Share a space with anyone

Storage is a plugin: Drive for non-coder collaborators, GitHub for code-adjacent work, local folders for testing. Same command, same artifacts — a shared workspace plus a `space-*` router note in your vault, so shared projects flow into your daily plan like everything else.

`--add-project` · `--status` · `--dry-run`

HOW THE LAYERS STACK — NOTHING COLLIDES

```
agents/   skills/   plugins/   hooks/   mcps/   templates/
  aios                                  ← framework, always shipped
  custom/                              ← your personal layer, never touched by updates
  <company>/                           ← every mounted company's layer
```

The canonical scaffold is `The-AIOS/company-template` — 10 context files plus 6 infra folders, ready to receive your team's contributions. Build the company brain once; every teammate's session inherits it in one mount.

> Don't move information to authority. Move authority to information.

11 TEMPLATES

Start from a shape, not a blank page.

Twelve bundled templates at `templates/` are the scaffolds everything in the vault is born from. Commands copy them automatically — the cold-start interview fills the context kit with you, a new project note arrives already shaped as a dashboard, a new agent arrives already knowing its sections. You never face an empty file deciding what the structure should be.

THE TWELVE, IN FOUR GROUPS

DECLARED-CONTEXT KIT — 6

Who you are, structured

The starting shape of your declared layer — identity, voice, how you think, your ventures, your role, your psychometrics.

`about_me` · `personal_voice` · `working_style` · `about_business` · `role-expectations` · `psychometric-profile`

PROJECT — 1

Born a dashboard

Current State router, to-dos, decisions log, session notes — every project note starts wired for the routing on page 04.

`project-template`

AGENT — 1

A fleet member in one file

Purpose, when-to-invoke, tools, instructions, constraints, schedule. The `aios-builder` agent scaffolds from it — compliant and registered.

`agent-template`

VENTURE & RHYTHM — 4

The recurring shapes

A new venture's context seed, meeting briefings, role logs, structured reading projects.

`about_venture` · `meeting-prep` · `role-log` · `reading-project`

SAME EXTENSION PATTERN AS EVERYTHING ELSE

`templates/custom/`	Your own templates — survive every framework update.
`templates/<company>/`	Arrive on mount — including branded collateral packs (proposal, invoice, agreement, one-pager HTML) that a brochures agent renders straight to on-brand PDF.

> Commands copy them, operators evolve them — every note is born already knowing its shape.

12 PERSONALIZATIONS

The framework is shared. USER.md is yours.

Every command ships byte-identical to every operator — you never edit a command file. What makes your AIOS behave like *yours* is one file: `USER.md`. Claude reads it every session, before anything else runs. It is never overwritten by `/aios:update` — your configuration survives every framework release.

WHAT LIVES INSIDE

IDENTITY

Sessions & greetings

Which session names are primary, how each greets you, the session cascade — extra files Claude loads per identity.

SOURCES

Where your life lives

Google accounts, communication channels, tools, growth routines, dev projects — the map Claude uses to find your real data.

COMMAND PERSONALIZATIONS

Per-command overrides

A `### /command` section per command, read before executing. Want `/today` to skip Slack? Write one line here — not in the command file.

INFRASTRUCTURE

Machines & companies

Remote machines (SSH spawn patterns for your agent host), mounted companies and their sync state.

THE PERSONAL TRIO

`USER.md`	Your configuration — identity, sources, command overrides.
`INTENT.md`	Your trust contract — autonomy levels, tradeoff rules, escalation triggers.
`vault/`	Your life content — notes, context, calendar, projects, exports.

> Everything shared lives in the repo, identical for all. Everything personal lives in three places you own.

13 OPERATING CLAUDE

The operator is the multiplier.

The AIOS rides on Claude Code, and a handful of native features change how the system feels day to day. None are required — all compound. The thread through every one of them: **autonomy is graduated, never granted**.

MODEL

Pick depth deliberately

Run vault sessions on the most capable model — the rituals are orchestration-heavy and reward it. `/fast` keeps Opus quality at higher output speed when you're iterating.

PLAN FIRST

Plan mode for big moves

For multi-file or risky changes, have Claude propose the full plan before touching anything — you approve, it executes. Cheap insurance.

PERMISSIONS

Shrink the prompts safely

`/fewer-permission-prompts` scans your real usage and proposes an allowlist of what you already approve — prompts shrink as trust grows.

AUTO-MODE

Scoped, never default

Auto-accept belongs in repos and tasks you've already learned to trust — the INTENT.md autonomous tier. Never your first week. Graduate, don't grant.

CADENCE

Schedule the rhythm

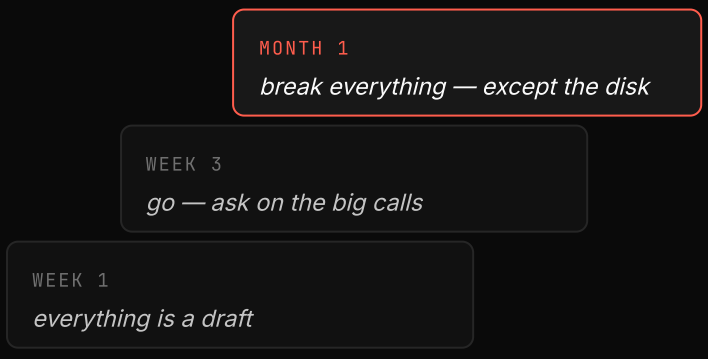
`/schedule` runs routines on cron — your daily plan can be waiting at 7am. `/loop` re-runs a task on an interval; `/goal` pins a long objective the session keeps working toward.

ANYWHERE

Remote control & browser

Remote-controlled sessions (the `spawn` wrapper sets this up) can be steered from claude.ai or your phone. Claude Code also lives in the browser, desktop, and IDE — the vault is the same everywhere.

THE TRUST LADDER IN PRACTICE — WRITTEN DOWN IN INTENT.MD



`INTENT.md` is where judgment lives — what runs autonomously, what needs you. It starts cautious and earns its way up, exactly like trust with a new hire. The Fortress (next page) is the far end of the same ladder.

> Match the mode to the trust level — the same task can be a draft in week one and autonomous by month two.

14 CONTAINMENT

Autonomy needs walls.

Want agents working 24/7 — overnight shifts, scheduled crons, work continuing while you travel? `FORTRESS.md` is the manual for running them safely: a two-machine architecture where a primary MacBook drives day-to-day and an always-on Mac mini hosts the autonomous fleet — firewalled so agents can act without exposing credentials, networks, or sensitive systems.

A prompt is not a boundary. In Anthropic's own red-team test, 24 of 25 phishing exfiltrations passed the model layer — only the environment boundary held. Containment lives in the environment, not the instructions.

SIX DEFENSIVE LAYERS

1 · Network isolation	OS-level firewall on the mini — no inbound except SSH from the main; no lateral movement to other devices on your Wi-Fi.
2 · Ecosystem lockdown	Handoff, AirDrop, Bonjour, Bluetooth, file/screen sharing — all off. The agent host is invisible to the Apple ecosystem.
3 · SSH hardening	Key-based auth only, no forwarding or tunneling, connection limits — one narrow door, from one authorized machine.
4 · Permission gates	Curated tool allowlist in weeks 1–2, graduated permissions from week 3 — autonomy expands the way trust does: earned.
5 · One-way data flow	Results push out via git; management flows in via SSH. The mini never reaches into the main — by construction.
6 · Recovery	Tailscale mesh for access from anywhere, a bridge monitor for observability, session-recovery helpers to restart interrupted work.

Total cost of a 24/7 agent fleet: a ~$600 Mac mini one-time, ~$25/month running. Claude reads FORTRESS.md end-to-end and walks both machines through the configuration.

> Trust = Speed / Cost. The walls are what make the autonomy affordable.

15 GETTING STARTED

Running in 30 seconds.

Have Claude Code installed? Open any terminal and say:

```
# the entire setup, in one line
› Set up my AI-OS from https://github.com/The-AIOS/aios
```

THE CANONICAL SEQUENCE

STEP 1 · ~10 MIN

Claude-driven setup

Claude reads `SETUP.md`, clones the framework, installs MCPs, configures your private vault, and walks every choice.

STEP 2 · ~20 MIN

Cold-start interview

`/aios:cold-start-interview` walks identity, declared context, the INTENT.md trust contract, bundles, and your first daily plan.

DAY 1 ONWARD

The rhythm

`/aios:today` morning · `/aios:close-day` evening. That's it. The system compounds from there.

MAKE IT YOURS — AND IT KEEPS ITSELF CURRENT

`custom/` — yours

Every infra layer has a `custom/` subfolder reserved for your additions. Framework updates never touch it.

`<company>/` — your team's

Mount a venture-context repo and every teammate's session inherits the full layer your team has built — in one prompt.

`/aios:update` — automatic

Pulls the latest framework and auto-applies it: infra reconciled to HEAD, your personal layer untouched, and Claude walks you through anything the CHANGELOG flags — detect, ask, act.

Reading this as Claude? Your operator pointed you here. Read `SETUP.md` and follow its "Reading this as Claude?" block — you're the executor; the human reads alongside. If the operator says "I'm lost," invoke `/agent onboarding-aios` — it knows the whole map and routes them without overwhelming.

16 TROUBLESHOOTING

Diagnose the system, not the symptom.

Things break — a connector goes silent, a command behaves like an old version, a config change seems to do nothing. The AIOS handles every break in two passes: **fix it now** (this page), then **write the rule** so it can't happen silently again (the antifragile loop, below).

SYMPTOM → FIX

"Claude seems lost in the vault"	Invoke `/agent onboarding-aios` — it knows the whole doc map and routes you (or it) to the right place without overwhelming.
An MCP stopped responding	Run `/doctor` for a health check, then re-run `/aios:mcps-setup` for that connector. Each MCP's README at `mcps/{name}-mcp/` carries the auth specifics.
"My config change does nothing"	Restart semantics: permission changes apply live, but MCP source and hook-command changes need a session restart. "Nothing happened" usually means "restart pending."
A command acts like an old version	Runtime cache lag — the plugin cache trails canonical. Run `/aios:update` (or `claude plugin update aios@the-aios`) to re-sync, then retry.
Git refuses to commit (stale lock)	Two writers are racing the repo. Don't just delete the lock — find and disable the second auto-committer (an editor plugin's auto-backup is the usual culprit).
Setup feels broken on a new machine	Everything is re-runnable: SETUP.md end-to-end, `mcps/setup.sh` (idempotent), `/aios:cold-start-interview` per section. Re-run; don't hand-patch.

THEN CLOSE THE LOOP — ANTIFRAGILE

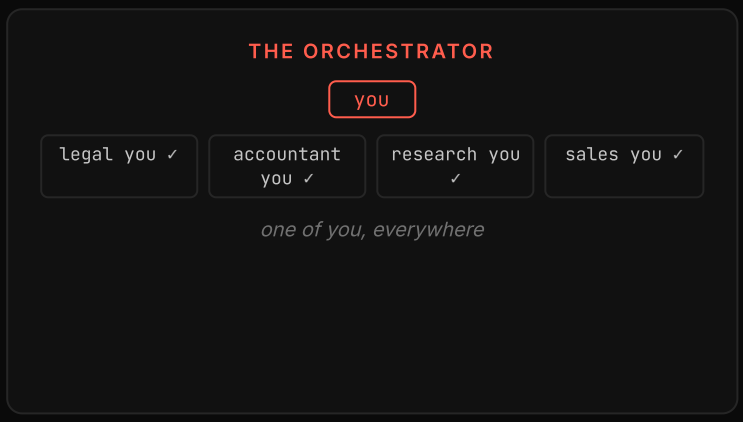
When a fix reveals a gap no existing rule covers, the rule is written into `antifragile.md` at the moment of failure — not in a retro, not in a backlog. Claude scans it before executing commands, so the system never breaks the same way twice. Ask *"was the decision process flawed, or did good process produce a bad outcome?"* — and fix the process, not the instance.

> Every failure is a system upgrade waiting to happen.

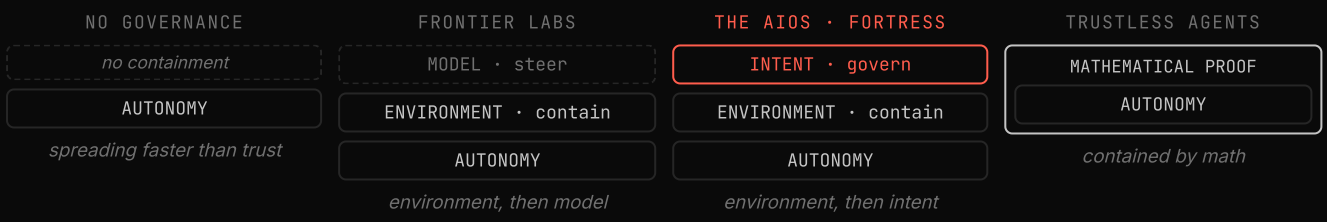
17 THE HORIZON

Verticality is dying. Horizontality is born.

You won't out-type the machine — and you were never meant to. Your role isn't the **Inspector**, checking draft after draft, a bottleneck descending. It's the **Orchestrator** — a legal you, an accountant you, a research you, all acting on your behalf. The AIOS is the vehicle for that flip: context makes the copies *you*, intent makes them governed, rituals make them compound.



HOW DO YOU CONTAIN AUTONOMY? THE SAME LESSON, AT FOUR SCALES



Ungoverned autonomy outruns trust. Frontier labs contain the environment, then steer the *model*. The Fortress contains the environment, then governs with *intent*. Trustless agents contain autonomy with the one thing that can't be talked out of: math. **The arc only bends one way — toward proof.**

> Stop checking your AI drafts. Start proving your AI process.

FROM AI CLARITY TO TRUE AUTONOMY

Not zero people. Compounded people.

This isn't a productivity hack. It's compounding self-awareness — an operating system that actually remembers, and a team that gets better every session.

> "The colony reads your traces; the seeds read your soil; the agents read your context — and what grows there, grows because you were there."
>
> — The Ground · chuycepeda.com/manifesto

the-aios.com · github.com/The-AIOS/aios