Getting started

Sign up, first login, and the guided setup.

Overview

Getting started

Sentilai is the control plane for the AI coding assistants your team already uses. Every request from Claude Code, Cursor, GitHub Copilot, Gemini CLI or Codex CLI routes through the Sentilai Gateway, where your policy is enforced in real time and everything is audited — your developers keep their tools and their workflow.

What you need

- A Sentilai workspace (your admin invite email contains a **passkey setup link** — no passwords).
- The **Sentilai Endpoint** app on each developer machine (macOS today, Windows coming), which configures the tools automatically.

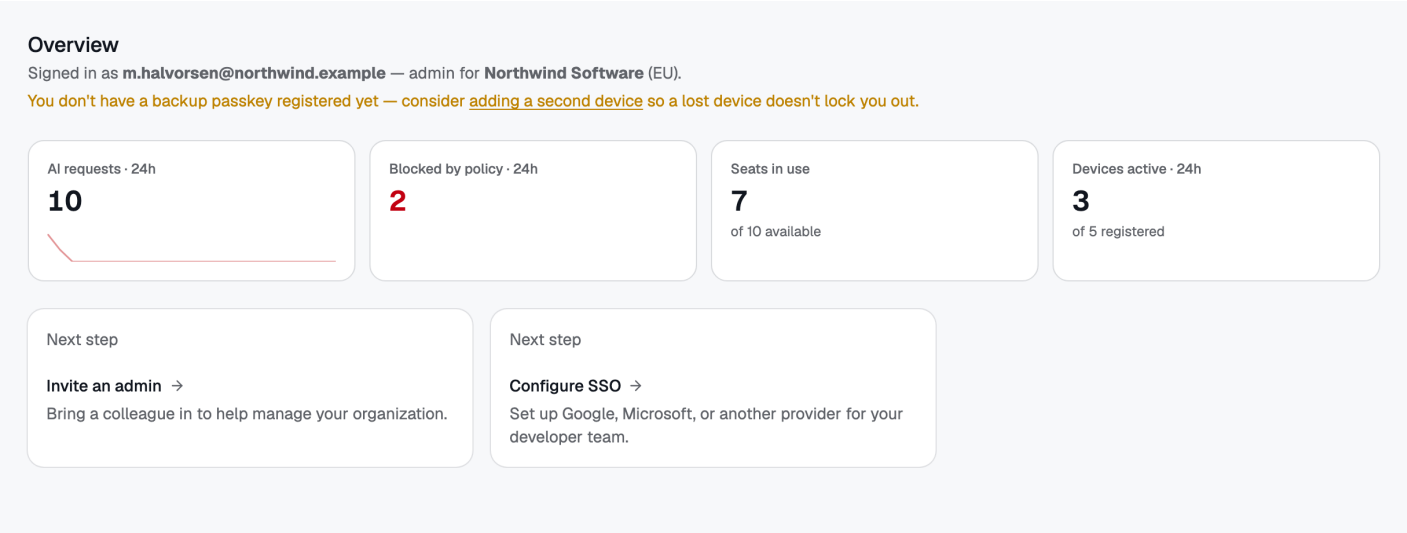
First steps

1. **Create your passkey.** Open the setup link from your invite email and register a passkey (Touch ID / Windows Hello / security key). This is how you sign in everywhere.
2. **Open the Admin Console.** You land on the Overview with the guided setup checklist.
3. **Install the Sentilai Endpoint** on a developer machine and sign in. Click **Govern all detected tools** — the app finds the installed AI tools and routes them through the Gateway. No per-tool manual setup.
4. **Watch Activity.** Within a minute of the first AI request you'll see audited events: model, tool, tokens, risk findings.

Where things live

- **Activity** — the audit trail of every AI request.
- **Policy** — MCP rules, risk detectors, real-time alerts, egress blocklist.
- **MCP Inventory** — every MCP server your developers use, with per-server and per-tool controls.
- **Users & Teams, Devices, Providers, Billing** — administration.

Next: Connect your tools and the per-feature guides in this help center.



The Overview answers "is everything OK?" first: requests, blocks, seats and active devices.

What Sentilai does

Your developers are already using AI coding assistants. Sentilai is the control plane that sits between those assistants and the AI providers behind them, so that every request is governed and audited without anyone changing how they work.

The shape of it

There are three parts.

The Gateway is a proxy. Claude Code, Cursor, GitHub Copilot, Gemini CLI and Codex CLI send their requests to it instead of directly to Anthropic or OpenAI. It applies your policy, records what happened, and forwards the request. Your developers see the same tools behaving the same way.

The Endpoint Suite is a small desktop app on each developer machine. It signs the developer in with a passkey, registers the machine, and rewrites each AI tool's configuration so it points at the Gateway. It also governs the MCP servers running locally on that machine — which the Gateway alone cannot see.

The Admin Console is where you set policy, see activity, manage people, and produce evidence. That is this documentation's main subject.

What you get that you didn't have

- **An audit trail.** Every AI request, who made it, which tool, which model, what it cost, and what the policy decided.
- **Enforcement, not just observation.** Blocking a dangerous MCP tool call, stopping a secret before it leaves the building, refusing a request that combines private data access with untrusted content and a way out.
- **Evidence.** A point-in-time record of your policy, your access inventory and your enforcement, for the auditor who asks.

What it is not

Sentilai does not read your source code repositories, does not sit in your CI, and does not replace endpoint security or a code scanner. It governs the traffic between AI assistants and AI providers, plus the MCP servers those assistants call.

It also does not stop a developer who is determined to bypass it — a personal laptop with a personal API key is outside any control plane. What it does is make the governed path the easy path, and make the ungoverned one visible.

Your first hour

This is the shortest path from a new organization to a governed request you can see in the console. The **Getting Started** page in the console tracks the same six steps and knows which ones you have finished.

1. Secure your own account (2 minutes)

You signed in with a passkey from your invite link. Add a **second** one now, from a different device, on **Account & Profile**. If your only passkey lives on a laptop and the laptop dies, nobody can let you back in — there is no password to fall back on.

2. Add a provider key (5 minutes)

Providers ? Add Provider. Sentilai needs a key for the AI provider your team's tools should use — Anthropic for Claude Code, OpenAI for Cursor and Copilot. Without one, tools in managed mode get an error instead of an answer.

If your developers pay for their own Claude or Copilot subscriptions and you only want governance, you can skip this — see Subscription or managed key.

3. Invite your team (5 minutes)

Users & Teams. Invite at least one other admin, so you are not the only person who can get in. Then invite the developers whose tools you want governed. Everyone gets a passkey setup link; nobody chooses a password.

4. Set your policy (10 minutes)

Policy. The defaults are deliberately mild — you will see everything and block almost nothing. Read Choosing your first policy before you start tightening; a policy that blocks too much on day one teaches your team to route around you.

5. Connect the first tool (10 minutes)

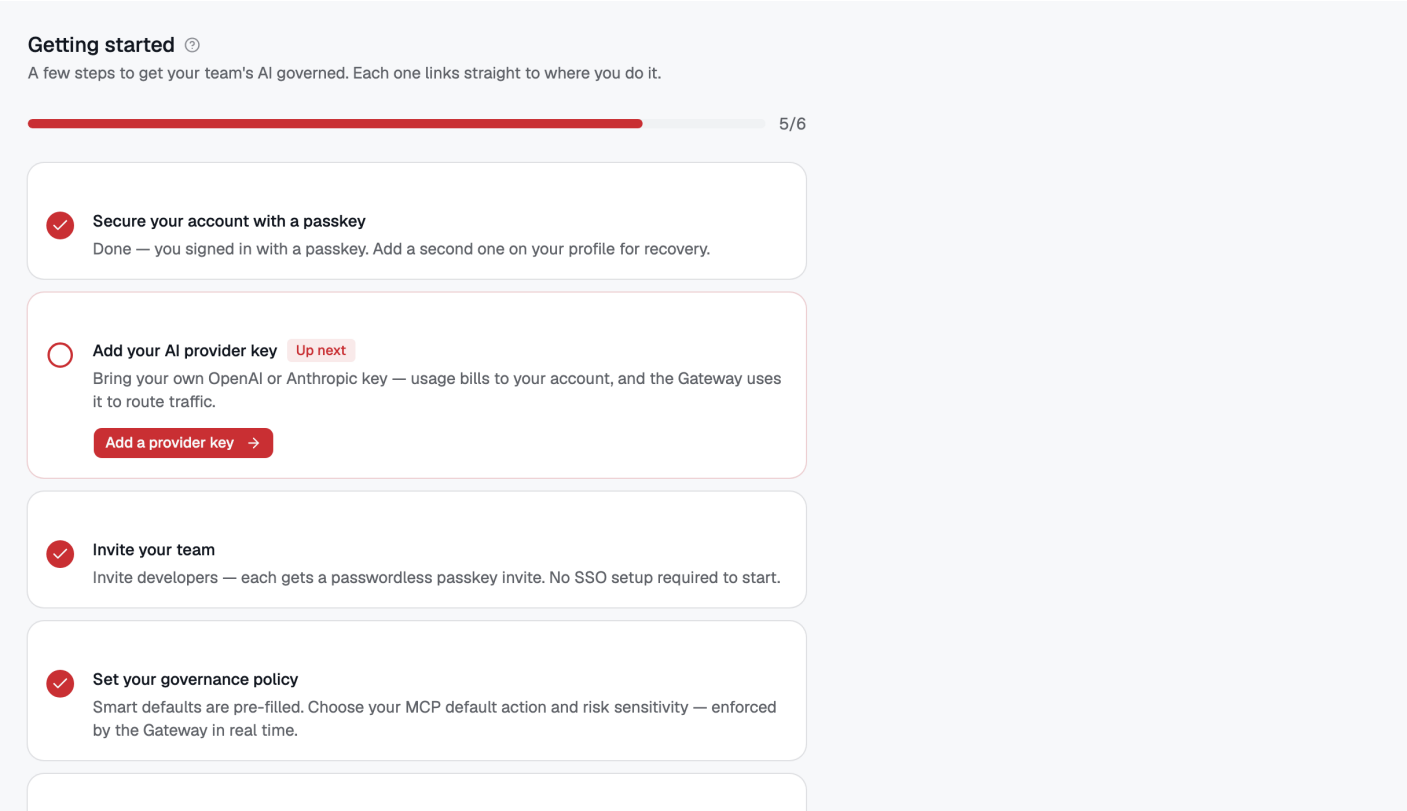
Install the Endpoint Suite on one machine — yours, or a willing developer's. Sign in, press **Govern all detected tools**, restart the tool. See Install the Endpoint Suite.

6. Watch it work

Activity. Within seconds of the first prompt, a row appears: who, which tool, which model, what the policy decided. That row is the whole product in one line.

Then what

Once one machine works end to end, the rest is repetition: roll the Endpoint Suite out, optionally through your MDM, and tighten policy as you learn what your team actually does. Rolling out to the team covers the scale-up.



The Getting Started checklist. Each step knows whether you have finished it, from real data rather than a box you tick.

The Overview screen

Overview answers one question — **is everything all right?** — before it offers you anything to do.

The five tiles

Each is clickable, and the first four cover the last 24 hours.

- **AI requests · 24h** with an hourly sparkline. The shape matters more than the number: a flat line during working hours usually means tools stopped routing through Sentilai, not that nobody worked.
- **Blocked by policy · 24h**. Turns a warning colour above zero. Click through to see which rules fired.
- **Seats in use**, `x of y available`. Admins and developers share one pool.
- **Devices active · 24h**, and how many are registered in total. A large gap means machines are enrolled but not being used — often people who left.
- **Ungoverned AI agents**, a standing count rather than a 24-hour one. Most days it reads "none detected", which is the answer you want. Above zero it names an autonomous agent found on a device that Sentilai can see but does not route. See Ungoverned AI agents under **Devices and logs**.

Next steps

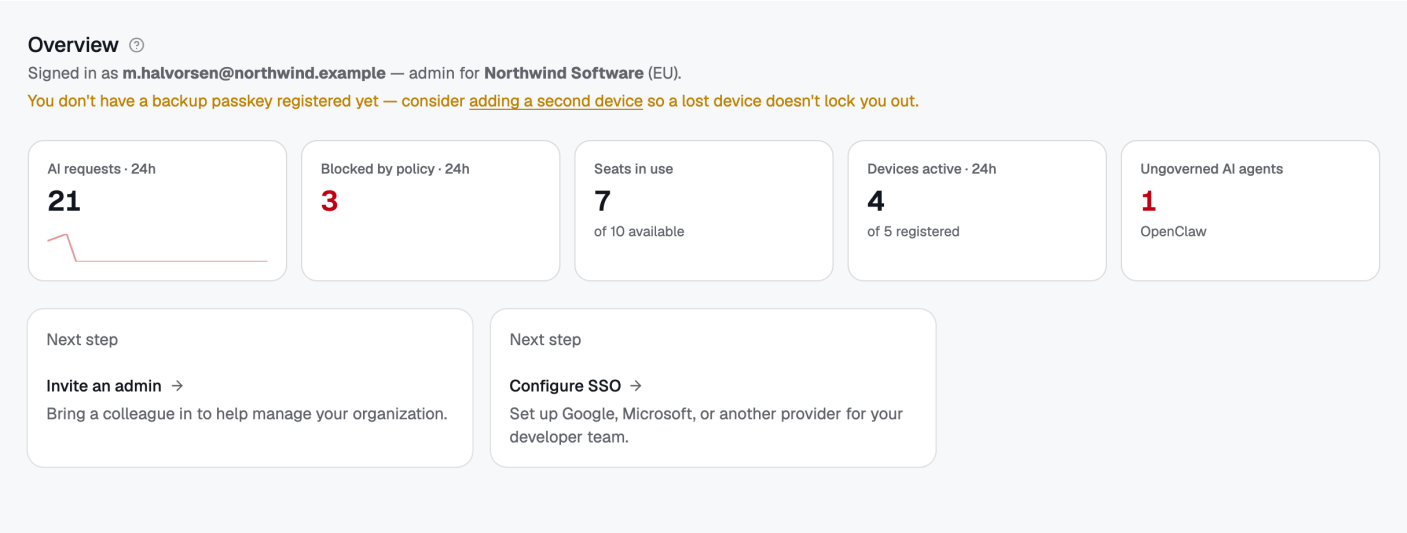
Below the tiles are the two things most new organizations have not done yet: invite another admin, and configure SSO. They disappear from your attention once done.

The backup-passkey warning

If your account has only one passkey, a banner sits at the top of this page until you add a second. It is the single most common way an admin locks themselves out permanently.

The header line

Your signed-in email, your organization name, and your **region** — EU or US. The region determines which Gateway your developers' tools talk to and where the audit data lives.



The Overview answers "is everything OK?" first: requests, blocks, seats, active devices, and any ungoverned AI agent found on a machine.

Admins, developers and seats

Sentilai has two kinds of people, and they use the product completely differently.

Admins

Admins use the console: policy, providers, people, activity, billing. An admin does **not** need the Endpoint Suite and does not need to be a developer.

Developers

Developers are the people whose AI tools you govern. They never open the admin console — if they try, they get a polite "this account can't open the admin console". What they use is the Endpoint Suite on their machine.

A developer can arrive two ways: invited by an admin, or created automatically the first time they sign in through your identity provider (see Set up single sign-on).

Seats

Both kinds consume a seat, and they share one pool. The count is on Users & Teams as `X of Y seats used`.

If you hit the cap, invitations are refused with a message naming your limit. Two ways out: offboard people who have left — which frees their seat immediately — or move to a larger plan on **Billing & Plan**.

A common surprise: developers who left the company keep holding seats until somebody offboards them. Offboarding is one click and it also revokes their devices and removes their passkeys, so it is worth doing promptly rather than at renewal time.

What neither role can do

Nobody in your organization can read another organization's data, and Sentilai support cannot read your prompt content. Support can see that a request happened and what the policy decided; the content, when captured at all, is yours.

Choosing your first policy

The defaults are chosen so that switching Sentilai on changes nothing about how your developers work. That is deliberate. The mistake to avoid is turning everything to **block** on day one: the tools break in ways your team cannot diagnose, and they learn that the way to get work done is to stop using the governed path.

Here is a sequence that works.

Week 1 — see everything, block almost nothing

- **MCP default action: Report.** You will get a full inventory of the MCP servers your developers actually use, without interrupting anyone.
- **Risk classifier sensitivity: Medium.**
- **Risk detectors:** leave at their defaults. Secret scanning warns, credentials and personal data report, lethal-trifecta warns.
- **Prompt capture: off.** Turn it on later, deliberately, and tell your team first.

At the end of the week, **MCP Inventory** shows you what your team is really connected to. Most organizations find at least one thing they did not know about.

Week 2 — decide about MCP

Go through the inventory and set an explicit rule per server: allow the ones you recognize, block the ones you do not. Then turn on **Require approval for new MCP servers** with a pending action of **Report** or **Warn**, so anything new shows up before it becomes normal.

Week 3 — start enforcing

- Move **Secret scanning** to **Block**. This is the one nearly everybody agrees on: an API key pasted into a prompt should not leave your network. Note that detected secrets are blocked regardless of the sensitivity dial.
- Move **Lethal-trifecta enforcement** to **Block** if your team uses MCP heavily.
- Consider **MCP default action: Block** now that the servers you approve have explicit allow rules.

What to tell your team

Tell them before you start, not after the first block. Two facts do most of the work: their subscription still pays for their usage in subscription mode, and nobody is reading their prompts unless prompt capture is on — in which case say so explicitly.

Subscription or managed key

Every governed request runs in one of two **modes**, and the difference is who pays.

Subscription

The developer's own AI subscription pays for the request. Claude Code with a personal or company Claude plan is the common case. Sentilai governs and audits the request but does not touch billing.

Use this when your developers already have subscriptions you are happy with, and you want governance rather than centralized spend.

Managed API key

The request goes out on **your** provider key, from **Providers**, and is billed to your organization. You get one bill, per-model control through the key's enabled-models list, and the ability to cut off access by removing the key.

Use this when you want the spend centralized or when the tool cannot carry a subscription login.

Setting it

Policy ? Connection modes, per tool. Claude Code, Cursor, GitHub Copilot, Gemini CLI and Codex CLI each get their own setting.

Cursor is locked to managed API key — its custom-endpoint mechanism cannot carry a subscription login. If you want Cursor governed, you need an OpenAI provider key.

What goes wrong

- Managed mode with no matching provider key: requests fail. The tool shows a provider error and Activity shows the request with an upstream error. Add the key.
- Managed mode with two keys of the same type whose model lists overlap: Sentilai cannot tell which key you meant and refuses the request. Make each key's **enabled models** list distinct.

Both modes are audited identically. In **Activity**, the **Mode** column tells you which one a given request used.

Welcome

Sentilai is the control plane for the AI coding assistants your team already uses. Every request from Claude Code, Cursor, GitHub Copilot, Gemini CLI or Codex CLI routes through the Sentilai Gateway, where your policy is enforced in real time and everything is audited. Your developers keep their tools and their workflow.

This is the documentation for the people who run it.

Start here

- **New to Sentilai?** Read What Sentilai does for the shape of the product in three minutes, then Your first hour for the shortest path from an empty organization to a governed request you can watch arrive.
- **Rolling it out?** Install the Endpoint Suite, then Govern all detected tools, then Rolling out to the team when the pilot works.
- **Setting policy?** Choosing your first policy is the sequence that avoids the mistake everyone makes — blocking too much on day one and teaching your team to route around you.
- **Something is wrong?** Start at Nothing appears in Activity; four times out of five the answer is that the tool was not restarted.

How this is organized

Eighteen sections, one per area of the product. **Browse them from the Books page** — each carries the settings, the workflows and the failure modes for one part of the system.

If you would rather ask than browse, the search box at the top searches every article.

What we promise about this documentation

It describes what is **built**, not what is planned. Where a feature has limits — Cursor's autocomplete is not routed, Copilot's inline completions are not routed, per-tool MCP rules apply where the Gateway can see a tool name — those limits are written down in the article about that feature, not omitted. What Sentilai does not do collects them in one place, and is five minutes better spent than any feature list.

Where something is not yet certified, we say so rather than implying otherwise.

Getting help

Admins can open a ticket from **Support** inside the console — we already know who you are and which organization you belong to, so there is nothing to prove. See *Open a support ticket* for what to attach.