

One workspace at the centre. Every model, tool and GPU around it — governed.

aimo is where your people work: they talk to any model they are permitted, search company documents, call tools, pick up skills, search the web and run code — in one window, on one sign-in. Behind it, Bosphorus pools your own GPUs and Macs with 8 external providers, relays every tool call through one governed proxy, and carries each person's identity — and, where a tool needs it, their own key from their vault — from sign-in all the way to the tool. Every request is efficient, attributable and on the record.



● your capacity ● models ● tools, skills and knowledge ● ways of working ● visibility and governance

Every spoke crosses the ring. Whatever aimo reaches for — and the same is true of an agent, an IDE or a script — the request passes identity, policy, quota, rate limit, fair-share admission and audit on the way out, and a person's own key is read only for a request already known to be theirs. There is no second route.

EFFICIENT

Your hardware carries the base load

Requests go to the least-busy machine that serves the model. A machine that keeps failing is rested, and a request that fails before its first byte is sent again elsewhere. External providers become capacity you add, not a place you are locked into.

EFFECTIVE

Everything a person needs, in the window they already use

Models, documents, tools, skills, web search and code execution arrive in aimo together, on one sign-in. There is nothing to install and no shared key to pass around — joiners get access with their account, leavers lose it when the account is disabled, and a tool that needs a person's own account gets that person's own key from their vault.

VISIBLE AND GOVERNED

Every request has an owner and a record

Each call carries the person it acts for — to the model, to the tool server, into the audit trail and onto the bill. Policy runs before a request is sent, not in a report written afterwards.

— AT THE CENTRE

aimo — the one window people need

People sign in with the corporate account they already have. Everything they are permitted to use arrives together, and every request they make crosses the same governed passage as an agent's or a script's.

Every permitted model

Models on your own GPUs, Claude, GPT and open-weight models in one picker. Model profiles decide who may use which, and the gateway enforces that on every request.

Company documents

Collections mapped to the people allowed to see them and indexed on your own hardware. A dedicated rule stops retrieved passages from reaching any external model.

Tools and skills

Registered tool servers appear as tools in the chat. The credential is added server-side — the server's own, or the person's own key from their vault — and the shared skills library is one request away.

Web search

A private metasearch service in your own cluster: the search engines see the platform, never the person. Pages come back as clean text a model can read.

Code execution

Python runs in a sandboxed kernel with its own private workspace. Charts and generated files come back as downloads, not as text pasted into the answer.

Help and recovery built in

aimo-help answers from the person's live model list and their own notifications. A blocked message names the rule and the conversation carries on; long chats are compacted rather than cut off.

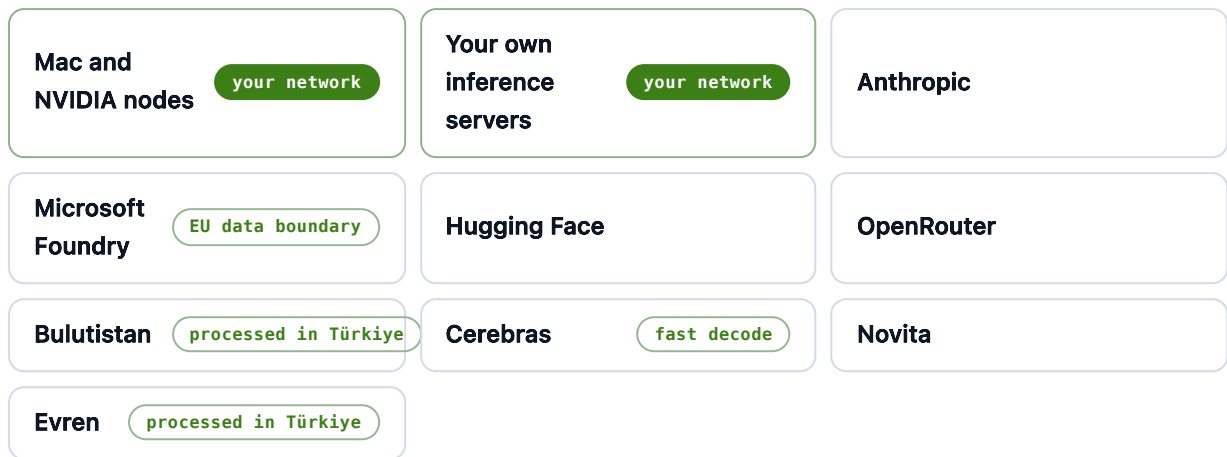
GPU AND MODEL EFFICIENCY

Use the GPUs you already own first

Macs, NVIDIA workstations and dedicated inference servers inside your network join one pool. The pool machines dial out to the platform, so nothing routes into your network, and the scheduler keeps your own hardware busy before anything is bought per token.

- **One API in three dialects** — OpenAI, Anthropic Messages and Ollama — so an existing tool joins by changing a URL, and every model sits behind the same address.
- **Least-busy routing with automatic retry.** A request that fails before its first byte is re-sent to another machine; one that keeps failing is rested and brought back on its own.
- **Fair-share admission.** Teams have priorities and concurrency limits, and waiting requests age, so a busy high-priority team cannot starve everyone else.
- **Budgets that bite before the bill.** Token budgets per team, requests and tokens per minute per key and per person — reserved when a request is admitted and settled to the real count when it finishes.
- **The right model for each team.** Model profiles steer a team, a key or a person to the models they should use, and a single alias picks the best available Claude model for them.
- **Prompt caching, billed at the cached rate,** so a long agent session that re-reads the same context is charged for what it really used.
- **Document indexing on your own GPUs,** balanced across machines, so turning company files into searchable knowledge never sends them outside your network.

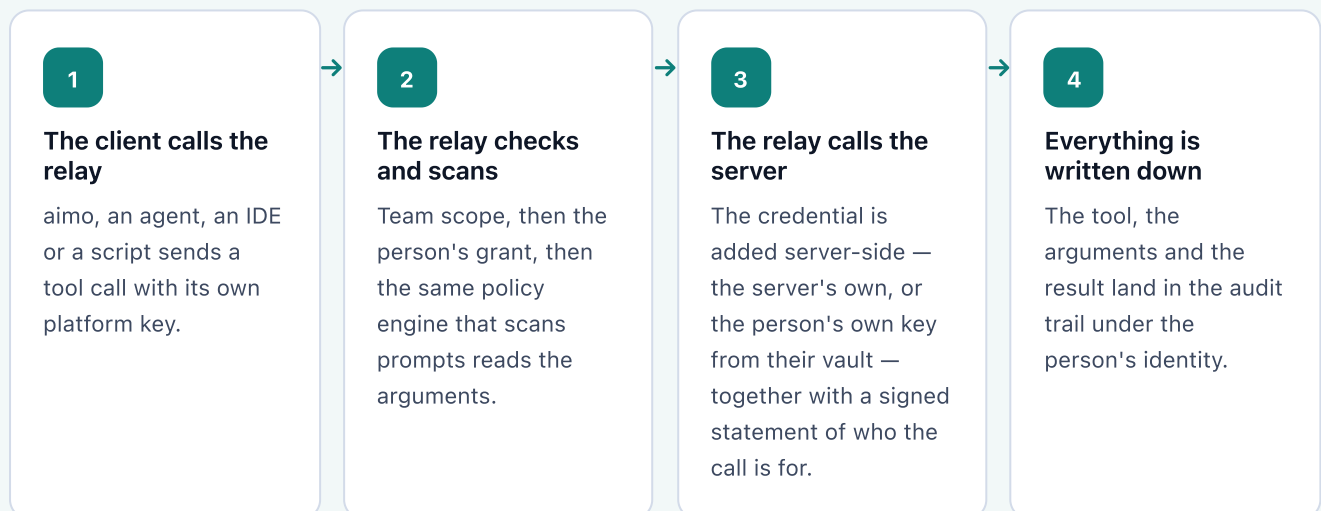
Behind the same API



MCP PROXY

39 tool servers, 1,522 tools, one governed relay

A tool server is registered once, by an administrator, and becomes available to aimo, to agents, to IDEs and to scripts alike. Clients call the platform; the platform calls the server — so no client ever holds a server's address or its secret.



- A server can be limited to certain teams, require a real end user, or be granted person by person — and one you may not use answers exactly like one that does not exist.
- Images a tool returns travel beside the answer, not inside it, so a screenshot never floods the model's context or its bill.
- Request size is capped at the relay, so an oversized tool call is refused before it reaches a third party.

Built and operated on the platform

platform Platform self-service 5 TOOLS READ ONLY PER-USER IDENTITY	skills Skills library 11 TOOLS WRITES	calendar Calendar 10 TOOLS ACTS AS YOU PER-USER IDENTITY
outlook Mail 11 TOOLS ACTS AS YOU PER-USER IDENTITY	teams Chat delivery 5 TOOLS ACTS AS YOU PER-USER IDENTITY	zendesk Support tickets 7 TOOLS WRITES PER-USER IDENTITY
expense Expenses 13 TOOLS WRITES PER-USER IDENTITY	docsearch Document search 4 TOOLS READ ONLY PER-USER IDENTITY	elearn Learning platform 11 TOOLS WRITES PER-USER IDENTITY
markdown Document conversion 1 TOOLS READ ONLY	junos Network devices 12 TOOLS WRITES PER-USER IDENTITY	weather Weather and marine 3 TOOLS READ ONLY PER-USER IDENTITY
maritime Maritime safety 7 TOOLS READ ONLY PER-USER IDENTITY	places Places and routes 3 TOOLS READ ONLY PER-USER IDENTITY	tgarchive Chat archive 4 TOOLS READ ONLY PER-USER IDENTITY

...and 24 more that teams registered for their own systems

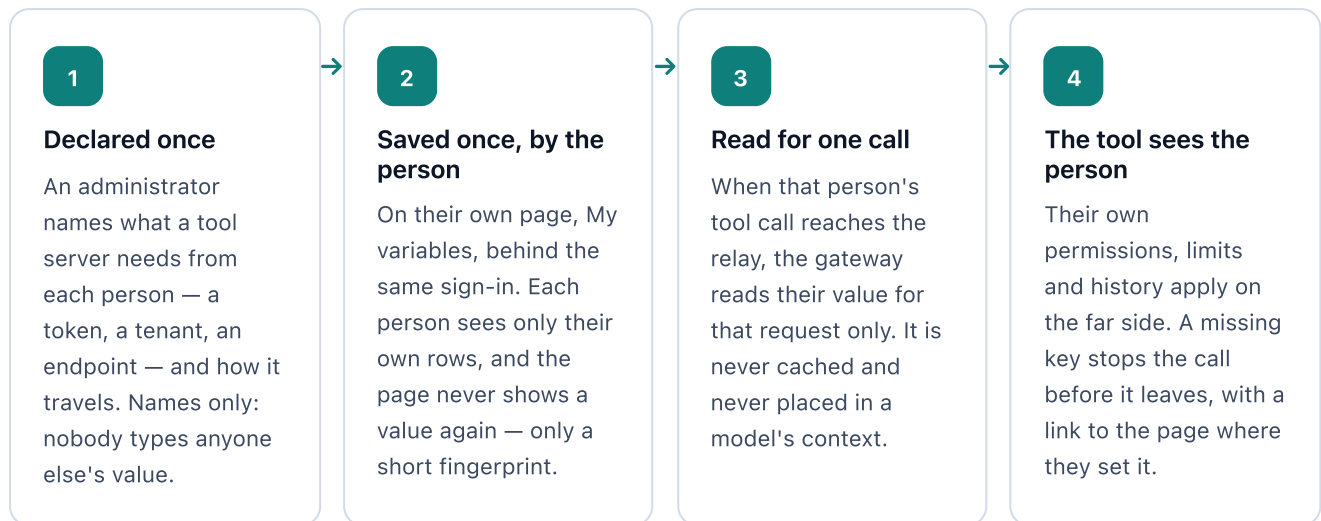
Network fabric and SD-WAN controllers, firewalls and network access control, DNS security, application delivery and performance, log search and SIEM, certificate management, voice, and library-documentation and code-intelligence servers. They are described here by kind rather than by name, because several carry a customer's or a colleague's. Registering one is a form and a credential; none of it needs a deployment.

THE VAULT

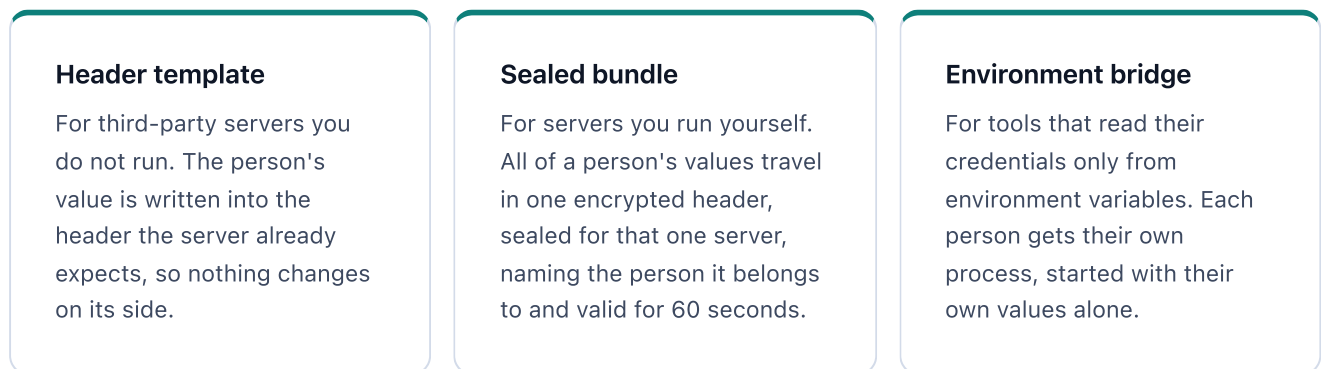
One vault, and every person on their own key

Many tools should see the person, not the platform: a code host, a ticketing system, a SaaS API, a cloud service — anywhere each colleague has their own account, their own permissions and their own history. The vault gives every person their own key for

every tool server that needs one. An administrator declares the name once, each person saves their value once, and the gateway adds it only to that person's own calls — whether they come from aimo, an agent, an IDE or a script.



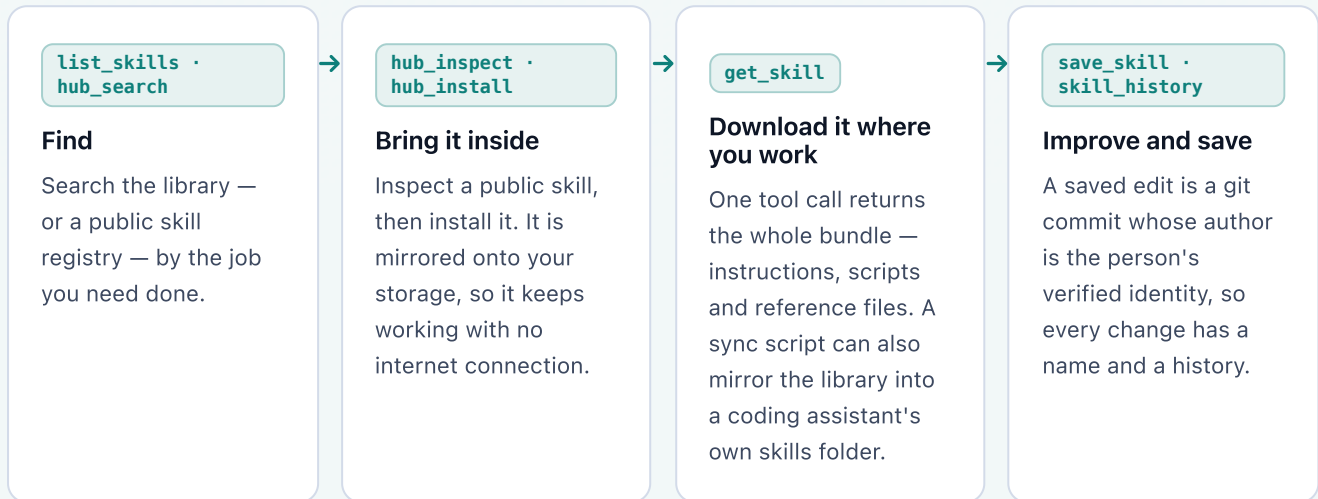
Three ways to reach a tool, one vault



- **Never in the chat, the model or the log.** The audit trail records which variable was sent, never its value, and if a tool echoes the value back, the stored copy of its reply is masked.
- **Console administrators see counts, never values.** They see how many people have set a key and can remove a leaver's values in one step, but cannot read them.
- **Unattended runs keep working.** A scheduled agent run at 03:00 still uses its owner's own key, because the platform reads it on their behalf — which is also where the remaining trust sits: guard the platform, and the backups that carry the vault, like the vault itself.
- **Your choice of store.** Values live in a secrets store — today OpenBao, where each person also gets a folder of their own to open — or sealed to one person and one server in the platform's own database. More secrets stores are planned.

Download a skill through MCP

A skill is a folder of instructions and scripts that teaches a model to do one job well. The library keeps 39 of them in git on your own storage and serves them over MCP, so the same skill reaches the chat, an agent, an IDE and a coding assistant — and every download is a governed, audited tool call.



A few of the skills in the library today — 39



Write it, run it, review it — on models you govern

Developers keep their tools and point them at the platform. Every token their assistant spends is policy-checked, attributed to them and billed to their team, exactly like everyone else's.

In the editor

The aimo extension for VS Code signs the developer in with their corporate account and works as an agent inside the editor: it reads and edits files, searches the codebase and runs commands, with every step visible.

In the terminal

Claude Code and other Anthropic-format tools connect through the Messages API unchanged — tool use, structured output and Claude's own web search included — and can be pointed at a model on your own hardware instead.

In the browser

An agentic coding workstation behind corporate sign-in, for people who should not need a local set-up to get real help with code.

Code review as a skill

The library's code-review skill gives every client the same bar — read the change, then judge correctness, security and maintainability — so a review in the IDE and one run by an agent at night hold the same standard.

Current documentation at hand

Library-documentation and repository-knowledge servers answer "how does this API work now" from current sources rather than from a model's memory.

Run it, safely

Code the model writes can be executed in a sandboxed kernel, one private workspace at a time, and the result comes back as a file you can download.

WEB SEARCH

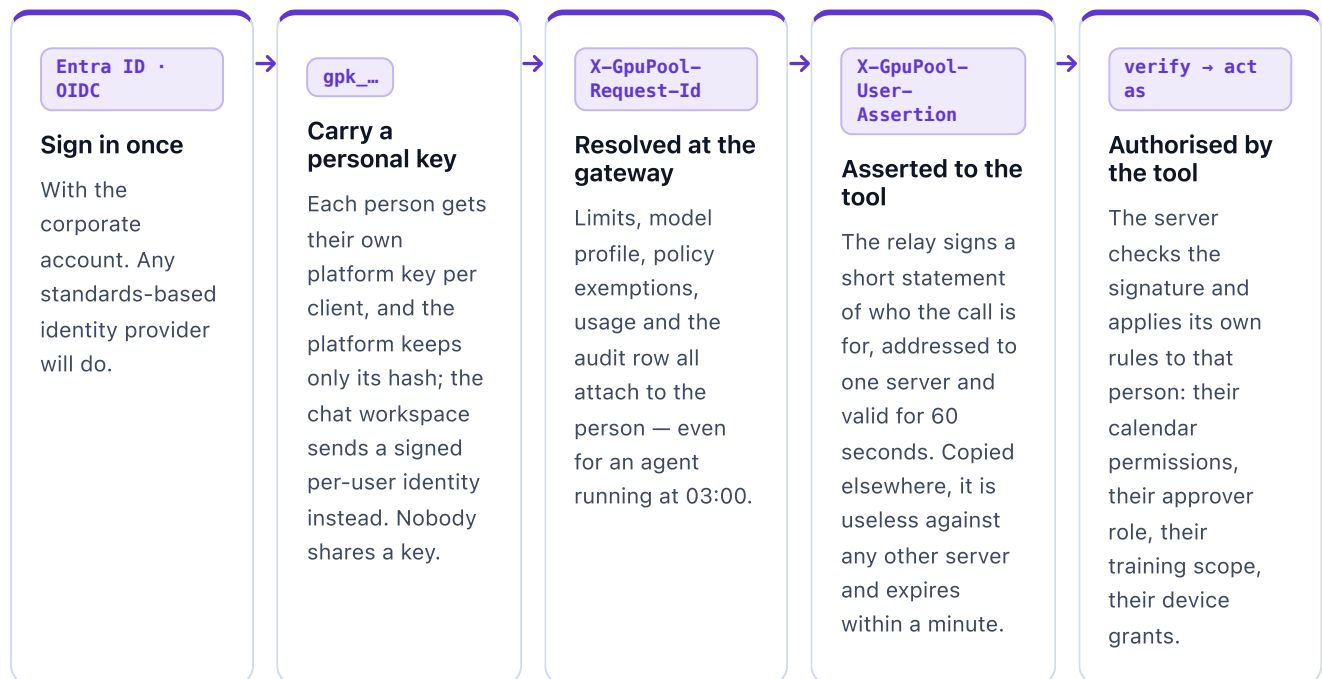
The open web, without exposing who asked

Searching and reading happen where policy can see them, and the answer becomes part of a conversation that is scanned and recorded like any other.

- **Private metasearch.** aimo searches through a metasearch service running in your own cluster. It needs no per-engine API key, and the search engines see the platform — never which employee searched for what.
- **Claude's own web search and web fetch** are carried end to end, billed per search, and refused with a clear error on a model that cannot run them — never silently dropped.
- **Any page into Markdown.** A public web page or document — PDF, Office, HTML, a video transcript — becomes text a model can read, through a converter that refuses private network addresses on every redirect.
- **Still governed.** What comes back joins the conversation, so the next turn is policy-scanned and audited like every other.

A personal key and a signed assertion carry one person from sign-in to the tool

Authentication happens once, at your identity provider. After that the person travels with the request: the gateway knows who is asking, and every tool server receives a signed statement of who the call is for — valid for 60 seconds and for that one server only.



Why not one shared service key? Because a shared key makes every call anonymous. One runaway agent uses up everybody's limit, the audit trail says "service", and a tool cannot tell a manager from an intern. Here, authentication and authorisation travel with the person — end to end — and a tool that needs the person's own credential gets it from their vault, not from a key everyone shares.

See who used what, what it cost, and what happened

Every request leaves the same trail, whichever client sent it — which is what makes the numbers comparable and the chargeback defensible.

Usage and chargeback

Requests, tokens, cached tokens and cost per person, key, team, model and provider, over whichever window you choose — the same numbers the bill is built from.

The audit trail

Every decision, denials included, with the full request and response, the rule that fired and the machine that answered. One request id links the client, the row and the log line.

Session replay

Step through an agent run or a long chat request by request, and see what was new on each turn.

Tags and sessions

Callers label traffic by application, environment or run, so spend can be split below the key and a failing session found with one filter.

Dashboards and alerts

Metrics and dashboards for the gateway, the machines and the cluster, with 48 alert rules in the box — and firing alerts surface inside the workspace, not only in an operator's inbox.

Into your own SIEM

Audit events can stream to your security information and event management system as they happen, so the evidence lives where your security team already looks.

GOVERNANCE

Policy runs before the model sees anything

Content rules, model allowlists and data-residency routing apply to every request from every client, in the same order, and a denial states the rule's reason and the instrument behind it.

28

policy modules in the box

20

governance instruments — 13 enforcing by default, 7 opt-in

790

policy assertions run before any rule change ships

- **One engine, every door.** The rules run in Open Policy Agent before any model or tool sees a request, and they read the whole conversation: prompts, retrieved documents, earlier tool results and the arguments of every tool call, from every client.
- **Rules in the regulator's own vocabulary.** Card numbers, national identity and tax numbers, IBANs, special-category personal data and cross-border transfer — alongside financial-sector and EU instruments. A denial cites the rule and the instrument it is mapped to.
- **Monitor before you enforce.** A new rule can run in shadow mode first, recording what it would have blocked on real traffic before anyone switches it on.

- **Context that must not leave, cannot.** Retrieved company documents and training-monitoring results are stamped, and egress rules keep that stamp off external providers.
- **Honest about the edges.** Responses can be scanned after delivery; that scan flags, it does not block. And if the policy engine is unreachable, the default is to let traffic through and record that it did — the posture is a setting you choose, not a surprise.

— EVERY COMPONENT

What is in the box

One product, deployed on your Kubernetes and your hardware. Each piece below is part of the same identity, the same policy and the same bill.

aimo

The chat workspace for the whole company: models, documents, tools, skills, web search and code execution on one sign-in.

The gateway

One API in three dialects, and the six gates every request crosses: identity, policy, quota, rate limit, fair-share admission, audit.

The MCP relay

The governed proxy in front of every tool server: credentials server-side, arguments scanned, calls audited, access per team or per person.

The vault

Each person's own keys for tool servers: declared by name by an administrator, saved once by the person, read only for that person's calls. Kept in a secrets store such as OpenBao, or sealed in the platform database.

Skills library

Reusable skills in git on your storage, mirrored from public registries, downloaded over MCP by every client.

agentdesk

Where people build their own agents — a fixed pipeline or an autonomous loop with budgets — and run them on a schedule, from Teams or Telegram, or on demand.

aimo for VS Code

An agent in the editor that reads, edits, searches and runs commands under the developer's own identity.

Document administration

Curators create document collections and map people from the identity directory onto them; indexing runs in the background on your hardware.

Learning platform

Courses written by models served through the gateway, searchable from chat and agents, with role-scoped progress for managers.

Expenses

Photograph a receipt; a vision model reads it, a validated record is created and the right approver is asked.

The console

Usage, activity, replay, keys and teams, providers, tool servers, policies and access — one place to see what happened and decide who may do what.

Policy engine

28 modules in the box, enforce or shadow per module, exemptions per person or team, and new rules tested before they ship.

Observability

Metrics, dashboards and alert rules for the gateway, the machines, storage and the network — including which flows a network policy refused.

Your capacity

Mac and NVIDIA machines that dial out to join the pool, and inference servers on your own GPUs, balanced and retried as one fleet.

Model providers

8 external providers behind the same API, including processing in Türkiye and inside the EU data boundary.

Identity

Sign-in through your corporate identity provider or any standard OIDC provider, carried end to end by personal keys and signed assertions.

Backups

Scheduled backups of every datastore and the cluster configuration, with retention you set and a restore that rebuilds the platform on a fresh cluster.

Bring us your hardest governance requirement.

We will stand a proof of concept up on your cluster, with your identity provider, your rules and hardware you already own — and hand you the evidence at the end of it.

A PoC needs a small Kubernetes cluster, one or two GPU or Mac nodes you already own, an identity tenant, and five to twenty pilot users. Everything else ships with the product.

Bosphorus — 2026. All rights reserved.

Built and supported by Morten Bilgi ve İletişim Hizmetleri A.Ş. · 0850 755 02 83 · iletisim@morten.com.tr

Product names are trademarks of their owners; their appearance here indicates interoperability, not endorsement.

[Cookie notice](#)