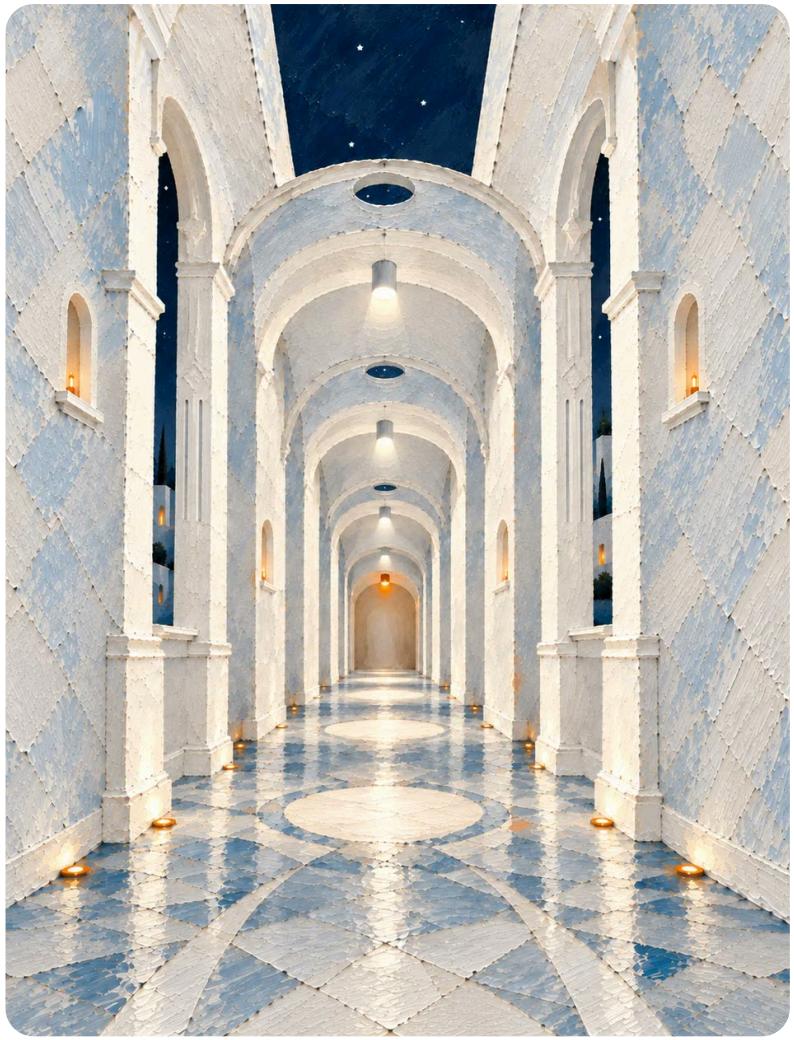
AI Gateway

AIRouter

Intelligent AI Gateway Solution

See AI costs before the invoice arrives,
and **filter sensitive data before it reaches a provider**.



This brochure doesn't start with a feature list. It starts with the seven concerns that companies running AI models actually raise.

Read it in order: one page on each concern, followed by one page on how it's solved.



Introduction

Introduction	Seven concerns of companies running AI models	p.03
Solution	What AIRouter is, and where it solves each concern	p.04

Seven concerns & solutions

Concern 01	AI costs show up only on the invoice	p.05
Concern 02	Separate keys, consoles and contracts for every provider	p.07
Concern 03	No idea where AI spend is leaking	p.08
Concern 04	Worry that personal data goes straight out to AI	p.10
Concern 05	No way to explain what an agent did after an incident	p.12
Concern 06	Switching one model means redeploying	p.14
Concern 07	When one provider goes down, your service goes down too	p.16

Use cases & adoption

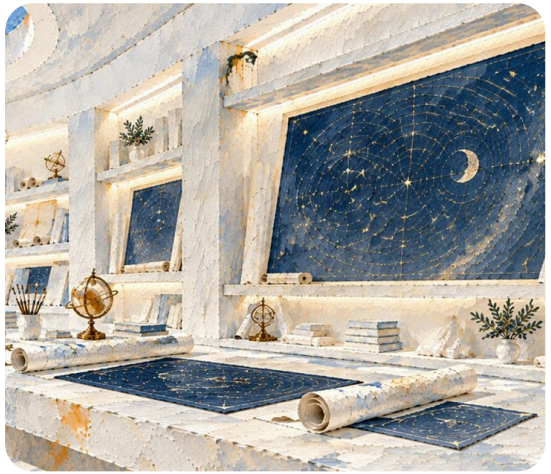
Use cases	How AI operations change with AIRouter in place	p.17
Adoption	Start with one service, then expand	p.19
Contact	Demo video · Contact sales	p.20

What we often hear

Seven concerns companies running AI models raise first

The moment you connect multiple AI models to your services, management asks for control and engineering asks for speed.

Here is what we hear again and again from both sides.



Click the concern that worries you most to jump straight to how AIRouter solves it.

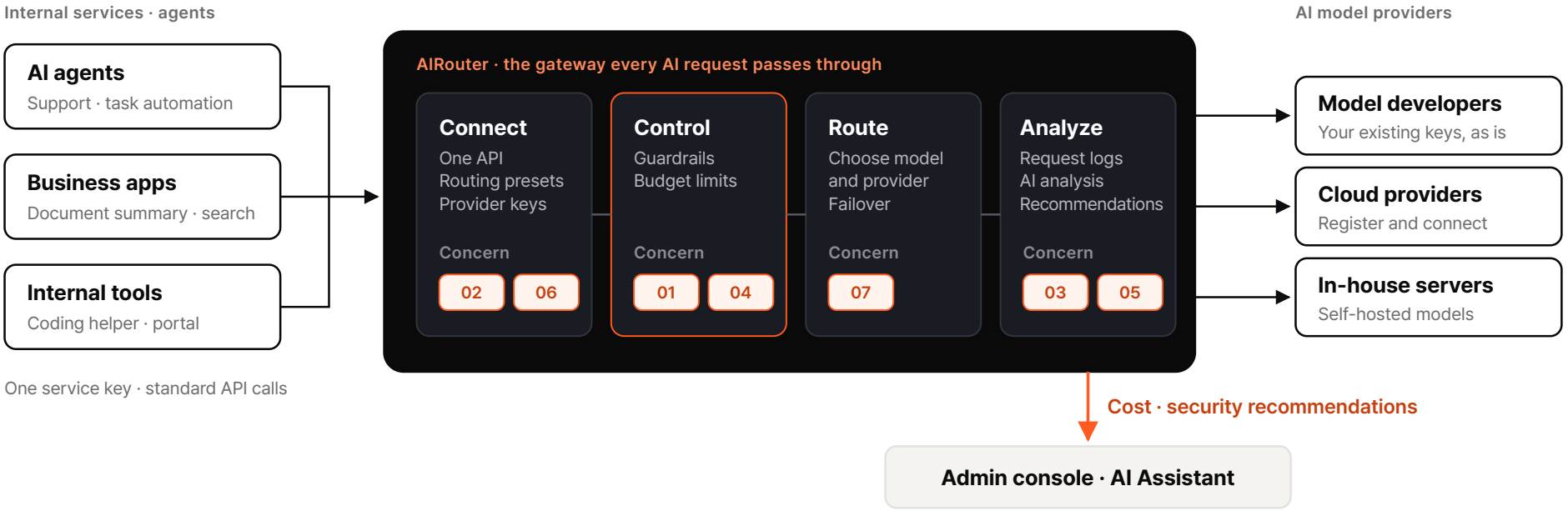
Concern 01	“We don’t know what AI costs us until the invoice arrives.”	Budget control →
Concern 02	“Every provider has its own keys, console and contract.”	Provider consolidation →
Concern 03	“We can’t tell where our AI spend is leaking.”	Cost reduction →
Concern 04	“We worry customer personal data goes straight to AI.”	Personal data →
Concern 05	“We can’t explain what an agent did after an incident.”	Agent control →
Concern 06	“Changing one model means a code change and a redeploy.”	Model switching →
Concern 07	“When one provider goes down, our service goes down too.”	Outage response →

One gateway

AIRouter solves these concerns at a single gateway.

Intelligent AI Gateway Solution

Budgets, security and outage response, now scattered across provider consoles and per-service custom builds, come together in one gateway. AI analyzes the records and recommends actions.



Click a number under “Concern” to jump to that concern.

Concern 01 · Budget control

We don’t know what AI costs us until the invoice arrives.

It’s the first thing finance and procurement teams bring up.



What happens on the ground

You see the cost after the fact

AI costs grow with every request, but the amount only shows up on next month’s invoice.

No way to split usage by team

Provider invoices are per account, so team and project usage must be split by hand.

No way to stop it

Even if one service suddenly makes a flood of calls, nothing stops it before it goes over budget.

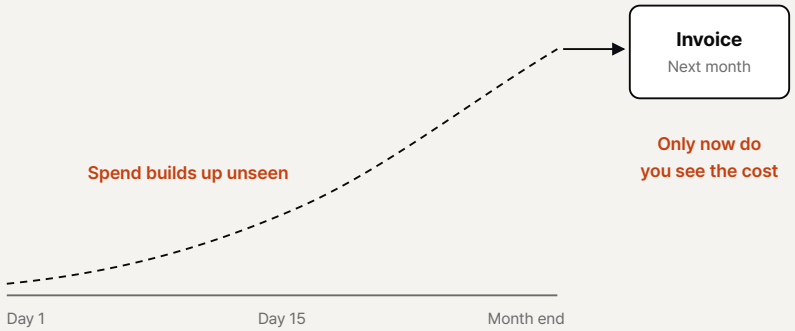
The hidden worry “Losing control of the AI budget”

See how AIRouter solves it →

Solution 01 · Control upfront, not review after the fact

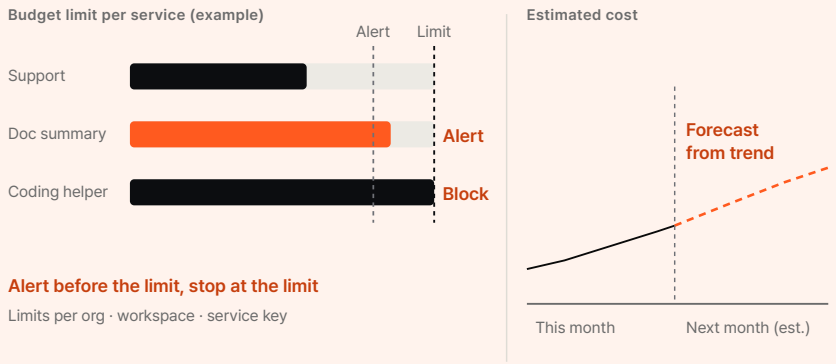
Get alerted before you reach the limit you set. When you hit it, calls stop.

Today



Between usage and the invoice, costs stay invisible for a whole month.

With AIRouter



See costs converted at list price as requests happen, and calls that would exceed the limit are rejected before they're sent.

Budget limits at every level

Set how much can be spent per organization, workspace and service key.

Alert first, block at the limit

Owners are notified as spending nears the limit, and calls stop once it's reached.

By team, project or model

See usage and list-price cost across providers in one place, broken down however you need.

Next month's cost forecast

The trend so far shows estimated cost for next week and next month.

What you gain

- ✓ Know estimated costs before the invoice arrives
- ✓ Operate within the budget you set
- ✓ Allocate budgets based on each team's actual usage

Concern 02 · Provider consolidation

Every provider has its own keys, its own console, its own contract.



A separate integration for every service

Each application connects to each provider on its own, so there's always more to manage.

As many consoles as providers

Usage, cost and security settings are checked separately in each provider's console.

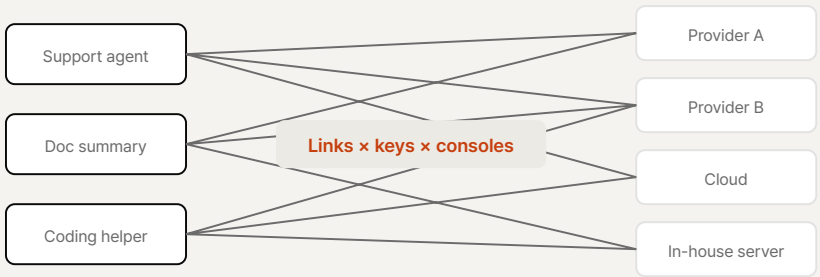
The hidden worry

"Losing control as providers multiply"

Solution 02 · One gateway

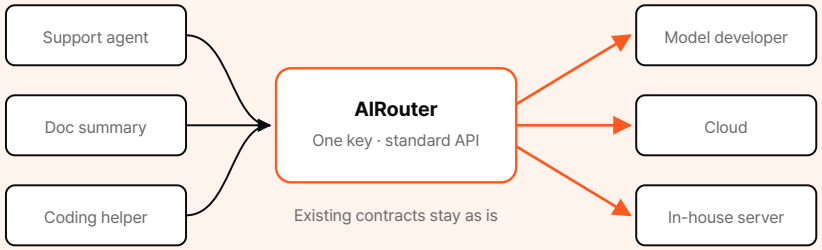
Use models from multiple providers with a single service key.

Today



Every new service or provider multiplies the connections.

With AIRouter

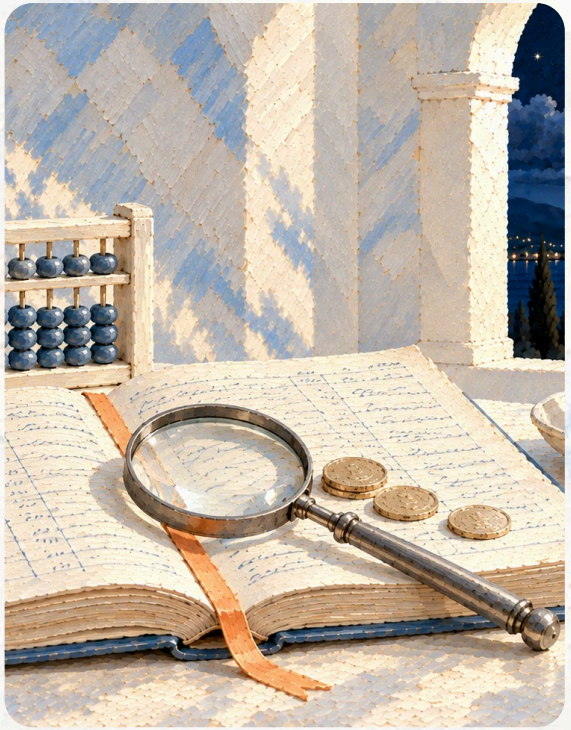


Services connect to one place; providers are managed behind the gateway.

What you gain

- ✓ Keep your existing provider contracts
- ✓ See each provider's cost at a glance

Concern 03 · Cost reduction



We can’t tell where our AI spend is leaking.

Something engineering and finance teams running AI services both tell us.

What happens on the ground

The same content, recomputed every time

Long instructions sent again and again aren’t cached, so you pay full price where a discount was available.

Failed requests still cost money

Spend on retries, failures and truncated responses is buried in the invoice, out of sight.

The priciest model for easy work

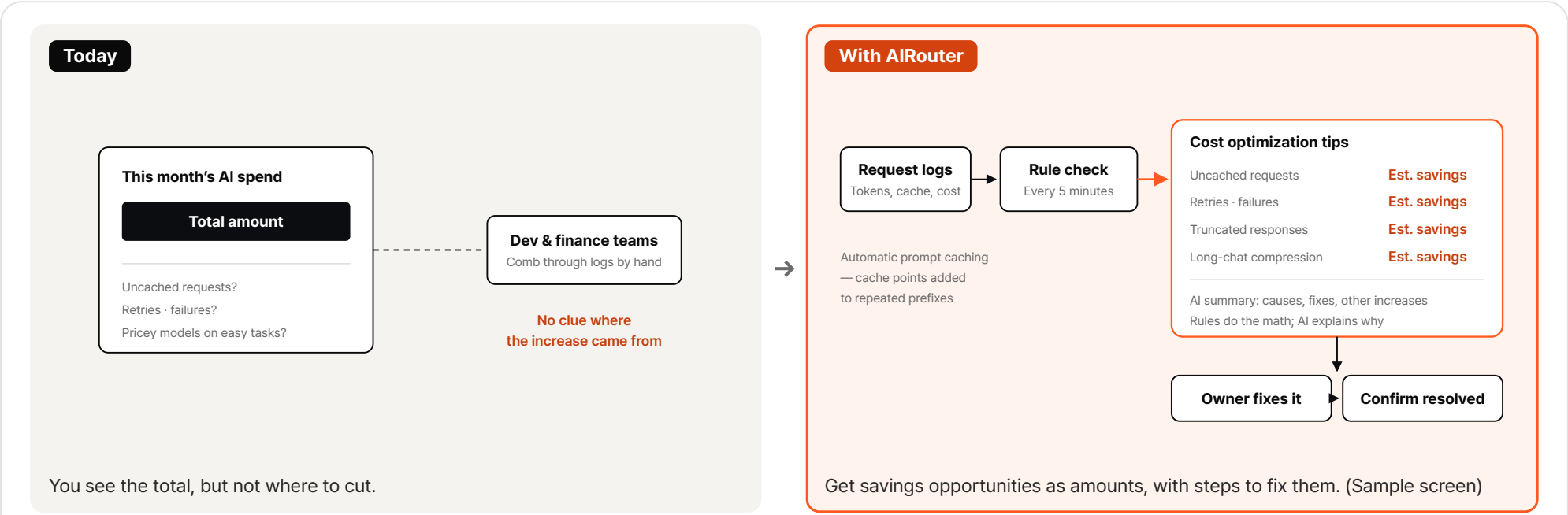
Even simple summaries and classification run on top-tier models.

The hidden worry **“Savings are there, but we can’t find them”**

See how AIRouter solves it →

Solution 03 · Start with the costs you can cut

Request logs are checked every 5 minutes to show which costs you can cut, and how.



Nine cost rules

Find uncached requests, retries, failures, truncated responses and long conversations, and estimate the savings.

Automatic prompt caching

Adds cache points to repeated prefixes and sends the same conversation to the same provider.

The right model for each condition

Split traffic across models by conditions you set, such as input length, tool use or image content.

Rules do the math, AI explains

Amounts and counts come from request logs; AI sums up causes and fixes.

What you gain

- ✓ See what you could save as estimated amounts
- ✓ Learn what to fix and how to fix it
- ✓ Verify each fix after you act

Concern 04 · Personal data · secrets



We worry our customers' personal data is going straight out to AI.

The first question security teams ask before signing off on an AI service.

What happens on the ground

No checkpoint in between

Each service calls providers directly, so there's nowhere to see what's leaving.

Agents follow hidden instructions

Hidden instructions in documents or search results can change what an agent does.

Secrets slip out in responses

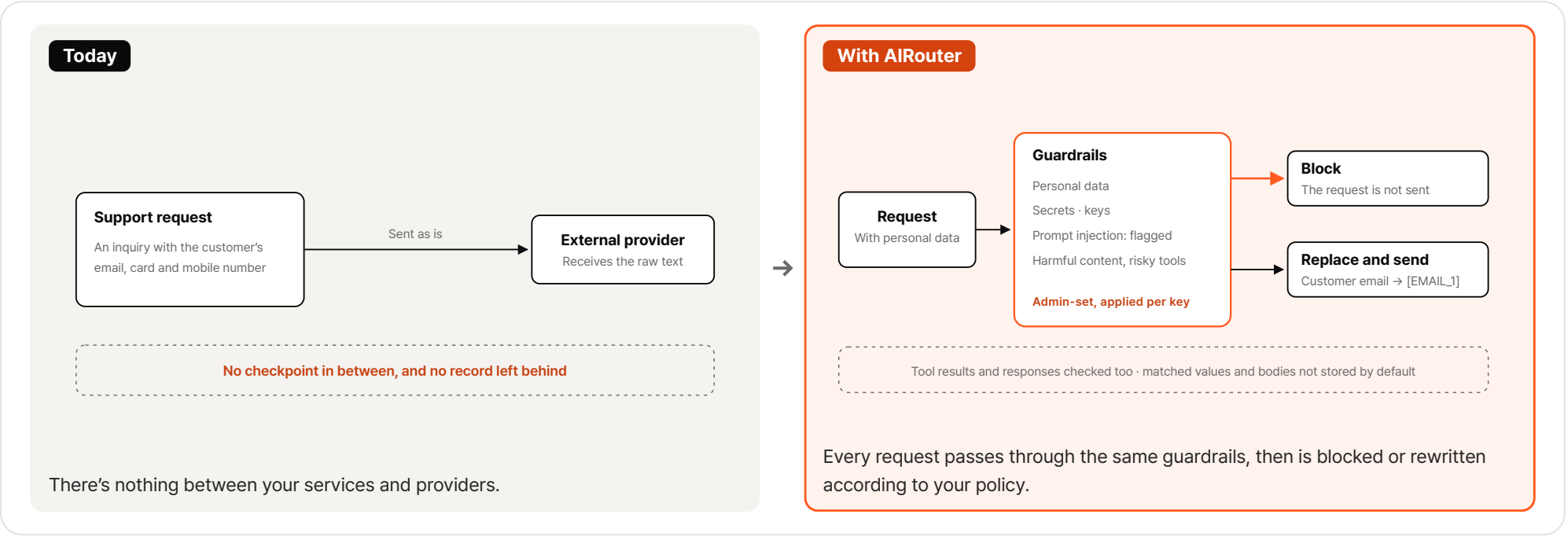
When a model's response contains secrets such as keys or passwords, nothing filters them out.

The hidden worry "The blame for a leak landing on whoever approved the AI rollout"

See how AIRouter solves it →

Solution 04 · Before it reaches the provider

National ID numbers, card numbers and secret keys are filtered out before they reach the provider.



Requests, tool results, responses

Checks not just outgoing requests, but also the tool results agents receive and the model's responses.

Block, or replace and send

By default, type labels replace national ID, foreign resident ID, passport, driver's license and bank account numbers.

Policies developers can't turn off

Admins set guardrails per service key, and callers can't switch them off.

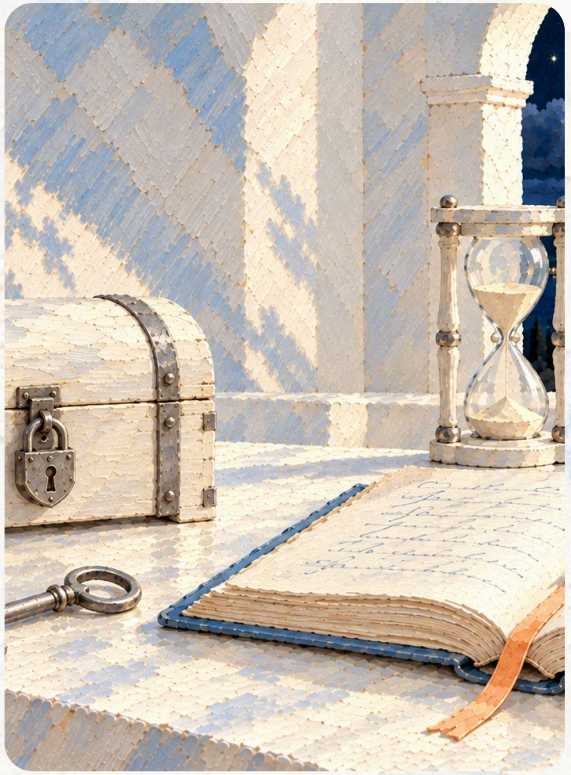
'Log only' first, then block

Start a rule in log-only mode to see how many requests it would block.

What you gain

- ✓ Sensitive data is stopped before it leaves the company
- ✓ Test policies without halting work
- ✓ The same security baseline across all your services

Concern 05 · Agent control · audit



When something goes wrong, we can’t explain what the agent did.

The question IT and security teams ask before putting agents to work.

What happens on the ground

One task becomes dozens of calls

A single agent task fans out into dozens of model and tool calls, so no single call reveals the risk.

Risky tools run unchecked

Nothing sits in between to check tool calls like sending email, making payments, deleting data or running commands.

No idea where keys are used

Use from an unfamiliar IP, or a long-idle key suddenly coming back, goes unnoticed.

The hidden worry “No record to trace what happened after an incident”

See how AIRouter solves it →

Solution 05 · All the way to what agents do

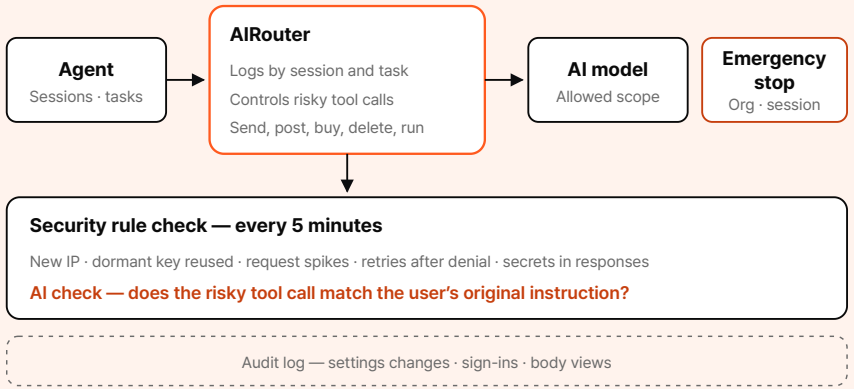
Risky tool calls stay under control, and warning signs in the logs are flagged early.

Today



No checks and no records between agents and tools.

With AIRouter



Every call is logged; rules and AI flag risks together. (Sample screen)

Risky tool call control

Tool calls that send, post, buy, delete or run commands are governed by policy.

Fifteen security rules

Every 5 minutes, look for signs like new IPs, dormant keys, request spikes and retries after denial.

Does it match the instruction?

For sessions that call risky tools, AI compares the action with the user's original instruction.

Emergency stop and audit log

Stop at once by organization, workspace or session, and log settings changes, sign-ins and body views.

What you gain

- ✓ Spot warning signs before an incident
- ✓ Explain incidents with records
- ✓ Stop at once when problems arise

Concern 06 · Model switching



“Changing one model means editing code and redeploying.

What AI development teams tell us most often.

What happens on the ground

Model names are hard-coded

Every new model release means another round of code changes, testing and deployment.

Rolling back takes a deploy too

If the new model causes problems, going back to the old one means redeploying.

Every provider works differently

Switching providers means reworking request formats and response handling from scratch.

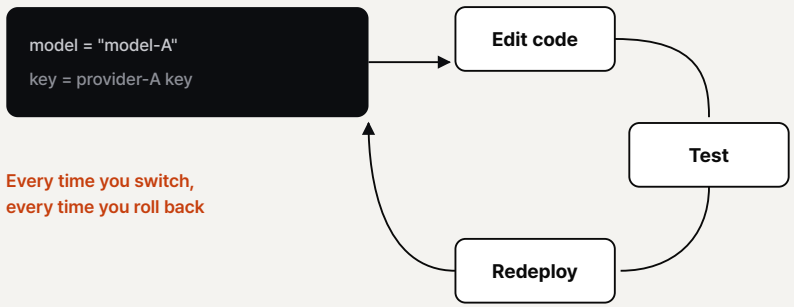
The hidden worry “Sticking with an older model because switching feels too risky”

See how AIRouter solves it →

Solution 06 · Code stays put; operations move to the console

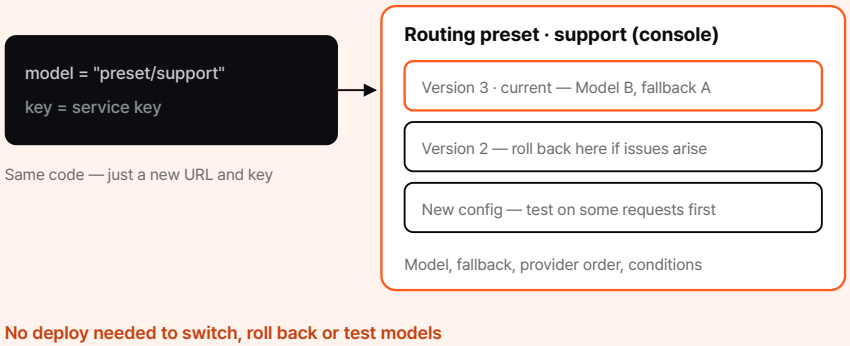
Your code holds only a preset name. Switch and roll back models in the console.

Today



Even one model change shifts the development schedule.

With AIRouter



Model operations move to the console, so your developers can stay focused on the service itself. (Sample screen)

Just change the URL and key

Compatible with two widely used AI API formats, so existing code works once you change the URL and key.

Routing presets

Models, fallback models, provider order and conditions live in the console; code calls only the name.

Versions and rollback

Presets are versioned, so if something goes wrong you can return to the previous version right away.

Test on some requests first

Apply new settings to some requests, check cost and latency, then expand.

What you gain

- ✓ Switch models without a deploy
- ✓ Roll back at once if issues arise
- ✓ Try better models without the worry

Concern 07 · Outage response

When one provider goes down, our service goes down with it.



Rate-limit errors reach users

When a provider's rate limit is exceeded, the error goes all the way to the user's screen.

Fallback logic, built per service

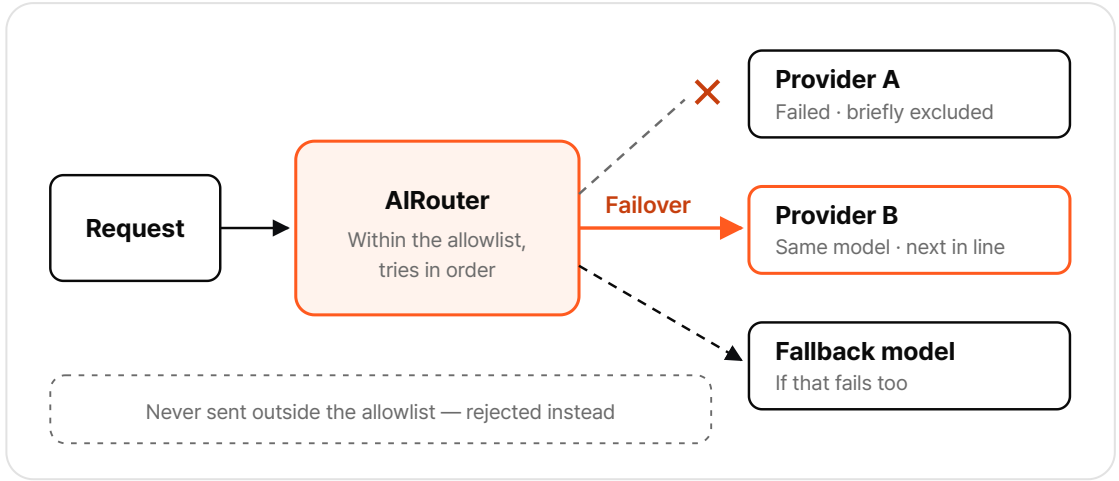
Engineering builds retries and switching to another provider separately for every service.

The hidden worry

"An AI provider's outage becoming our outage"

Solution 07 · Failover within allowed bounds

When a provider has trouble, traffic moves to another allowed provider.



Automatic failover

If a call fails before responding, it moves to the next provider, then to the fallback model.

Pause what keeps failing

Providers with repeated failures in a short window are skipped for a while.

Never outside policy

Anything outside the allowlist, no-retention or no-training terms is rejected, not sent.

What you gain

- ✓ No need to build outage handling per service
- ✓ Failover happens only within your security policy

Back to contents

Use case examples

With AIRouter in place, AI operations change like this.

Agent development · pre-launch checks

Catch cost and security issues before launch

What you gain — fix cost and security issues **before launch**.

Today	What AIRouter handles	What your team does
What a support agent costs, and which tools it calls, only becomes clear once real customers use it.	Traces sessions run in the dev environment to find cache waste, risky tool calls and guardrail hits, then sums up causes and fixes.	Fix what was flagged, rerun, confirm the issues are gone, then launch.

AIRouter capabilities used: Solution 03 Cost analysis · Solution 05 Tool control · Solution 04 Guardrails

Connects to: Dev workspace · Service key · Session · task logs

Finance · team budgets

No more splitting AI costs by team by hand

What you gain — allocate budgets **based on each team's actual usage**.

Today	What AIRouter handles	What your team does
Download usage from each provider console, split it by department, and build the monthly report in a spreadsheet.	Enforces per-workspace limits and, when asked in chat, produces a monthly report with tables and charts.	Review the report, then check and apply the budget changes the AI Assistant suggests.

AIRouter capabilities used: Solution 01 Budget control · Solution 02 Unified totals · Solution 03 Cost analysis

Connects to: Workspaces · Budget limits · AI Assistant

These are illustrative use cases. We'll scope the coverage and impact for your services together when you contact us (page 20).

Use case examples (continued)

The same approach, across other work.

Security · regulated industries

Test policies first instead of blocking right away

What you gain — tighten security policy **without halting work**.

Today	What AIRouter handles	What your team does
Worried that switching on blocking rules will disrupt work, teams end up switching on none at all.	Runs rules in log-only mode first and shows, per service key, how many requests would have been blocked.	After reviewing the impact, switch to blocking and use the audit log as evidence in security reviews.

AIRouter capabilities used: Solution 04 Guardrails · Solution 05 Security rules · Solution 01 Budget control

Connects to: Guardrails · Audit log · Alert webhooks

Service operations · model switching

Don’t roll a new model out to everyone at once

What you gain — move to a better model **without a deploy**.

Today	What AIRouter handles	What your team does
Even when a better model comes out, a full switch feels like too much, so the old model stays in use.	Applies a new routing preset version to a share of requests, and fails over to another allowed provider during an outage.	Compare cost and response time in the request logs, then expand to all traffic or roll back.

AIRouter capabilities used: Solution 06 Routing presets · Solution 07 Failover · Solution 03 Cost analysis

Connects to: Routing presets · Request logs · Fallback models

These are illustrative use cases. We’ll scope the coverage and impact for your services together when you contact us (page 20).

Adoption steps

No need to move every service at once. Connect one service, confirm it works, then expand.

1

Review the current state together

See which services use which providers, and where keys and costs are managed.

What you gain

A clear view of scattered AI calls and keys

2

Connect one service

Connect one service by changing only the URL and key in existing code. Guardrails start in ‘log only’ mode.

What you gain

A low-risk trial: restore the URL and key and you’re back where you started

3

Turn on policies

Set budget limits and routing presets, then switch guardrail rules to blocking once you’ve checked their impact.

What you gain

Operations within the budget and security standards you set

4

Expand to other services

Split workspaces and service keys by department or project, and expand under the same policies.

What you gain

Every AI call managed from one place

For a first meeting, a few of these are enough

AI providers in use

A list of the AI providers and clouds you contract with

Services that call AI

A list of agents, applications and internal tools

Monthly AI spend

Rough spend by provider

Security standards

Data that must not go outside, and your internal security policies

Start light with SaaS

Get a service key and connect right away, keeping your existing provider contracts.

Dedicated environment

With strict security requirements? Discuss a customer-dedicated cloud configuration.

Request bodies not stored by default

Request bodies are stored only if your organization turns this on, for the period you set.

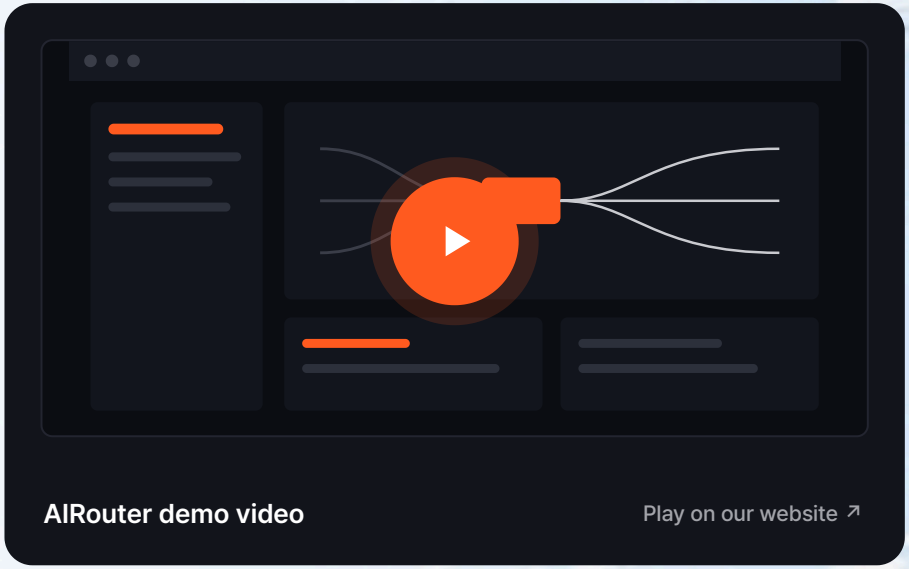
Talk to us

Start by telling us your challenges.

Tell us about one service that calls AI, and we'll use it in a demo to walk through where costs leak, what sensitive data goes out, and how failover would work. A few items from the list on page 19 are enough, and we'll also explain how adoption costs are estimated. Want a look first? Watch the demo video.

Contact sales →

▶ Watch the demo



When you get in touch, tell us your most urgent concern

- Budget control
- Provider consolidation
- Cost reduction
- Personal data
- Agent control
- Model switching
- Outage response

Email
sales@vicsnc.com

Website
vicsnc.com