Cut the agent bill.
Keep the outcomes.

The hands-on cost playbook for engineering orgs running AI agents in production. Save it — you'll use it Monday.

INSIDE

BUDGET ENVELOPE · ROUTING LADDER ·
REQUEST TEMPLATE · FOUR AUDITS ·
WEEK-ONE CHECKLIST

Pini Shvartsman

in/pinishv · pinishv.com

One metric makes every lever legible.

unit-economics

```
cost per accepted task =
  total AI cost for the workflow
  ÷ outputs accepted without rework
```

→ Count merged PRs, resolved incidents, approved reviews — not tokens

→ Charge failed runs to the workflow, not a memory hole

05

Five levers move the bill.

01 Which model handles each task

02 How much context is re-sent per request

03 How many tools, files & logs get loaded

04 How long the agent is allowed to loop

05 Caching & batch discounts vs. retail

Four audits you can run today.

AUDIT 01

Dead-weight context

Open your 10 longest sessions — how much is left over from already-finished tasks?

AUDIT 02

Verification theater

Tool calls made after the final test already passed. Count them.

AUDIT 03

Tool-schema tax

Sum tool-definition tokens on one request. Typical multi-server setup: ~55k.

AUDIT 04

Faith-based escalation

Premium-model calls with no failed cheaper attempt behind them.

One gateway in front of everything.

A **Open-source org**

LiteLLM + Langfuse

B **Already on Cloudflare**

AI Gateway + Langfuse or Phoenix

C **AWS-centric**

Bedrock as the front door — IAM-scoped, tagged — + Langfuse or Phoenix

D **Want turnkey**

Portkey or Helicone

required on every request

```
tags: user · team · repo · agent ·
      workflow · task_type · tier
```

A budget is a contract, not a kill switch.

budget.yaml — per session

```
max usd_per_session
max input · output · reasoning tokens
max steps & tool_calls
max retries_per_failure_signature
max subagents · elapsed_time
max premium_model_calls
```

On exhaustion the agent returns: done, remaining, evidence, next action. Never a hard kill — that throws the work away.

Route by tier. The default is cheap.

TIER 01 · ECONOMY — THE DEFAULT

Repo search, edits, formatting, test generation, docs, patches from approved plans

TIER 02 · STANDARD

Multi-file bugs, code review, API integration, unfamiliar code

TIER 03 · PREMIUM

Architecture, security, migrations, incidents, tasks that failed a tier below

One step up at a time. Never economy → premium in one jump.

Escalate on evidence. Not vibes.

ESCALATE WHEN

Findings contradict each other

The work is security-classified

The task failed one tier down

NEVER FOR

”Looks difficult”

A missing import

Noisy test output

Drive-by cleanup

Paste this into every agent request:

request.md

```
GOAL   fix the duplicate-notification race
SCOPE  notification_service.ts + its tests
LIMITS no API changes · no drive-by refactors
VERIFY pnpm test notifications
STOP   when the suite passes.
        nothing outside scope.
```

The STOP line is the money line. An agent without one stops when the context window does.

10

Context is a budget, not a backpack.

- [ ] One task per session — new task, new session
- [ ] Root AGENTS.md under 200 lines
- [ ] Skills load on demand, ≤500 lines each
- [ ] Ignore files cover node_modules, builds, fixtures, datasets
- [ ] Logs filtered to errors + traces before they enter context

The numbers are already public.

−85%

tool-definition tokens with on-demand tool search (from ~55k per request)

0.1×

cache reads vs. base input price (writes: 1.25× / 2×)

−50%

batch APIs for async work — OpenAI, Anthropic & Gemini alike

7×

tokens for agent teams vs. a single session (plan mode)

Provider-documented figures — Anthropic, OpenAI, Google, GitHub. July 2026.

Claim the discounts you're already owed.

01 **Stable content first, task content last**

Timestamps and request IDs near the top silently kill the cache behind them

02 **Track cache hit ratio weekly**

A collapsing ratio is an incident, not a curiosity

03 **Nobody waiting? Batch it at ~50% off**

Evals, embeddings, doc refreshes, backlog labeling, migration inventories

13

Week one, in order:

01 Tag every request: user, team, repo, workflow

02 Produce the baseline spend report

03 Define three tiers; flip the default off premium

04 Add one-escalation + retry limits

05 Root AGENTS.md; 2–3 skills for specifics

06 Ignore files; disable unused MCP servers

07 List batch-eligible workflows

08 Add cost + quality fields to every eval

Make the cheap path the default. Make the expensive path prove it.

FULL WRITE-UP

pinishv.com/articles/ai-agent-cost-reduction-playbook

Tappable here — and in the comments.