TOKEN ECONOMICS · 2026

Your AI coding bill is a metered-token money fire.

A 4-developer team burns **$42,240/yr** on AI-coding tokens. The same team on GraQle pays **$5,174**. This deck is the math — and the proof, at the real per-model price.

THE SHIFT NOBODY BUDGETED FOR

Flat seats are gone. You pay per token now.

- **GitHub Copilot** moved every plan to **usage-based per-token billing on June 1, 2026**.
- **Pro+ ($39/mo)** credits **don't roll over** — and can drain in ~1 hour of agentic coding.
- Cursor and Claude Code are usage-metered too. Microsoft reportedly burned a team's **entire annual AI budget in months**.
- It's not an Anthropic / Cursor / Copilot problem. It's a **flat-file context** problem — every tool re-feeds whole files on every question.

Sources: github.blog (usage-based billing, 2026-06-01); docs.github.com Copilot billing.

EVERY ASSUMPTION ON THE TABLE

The scenario

VARIABLE	VALUE
Codebase	**50,000 nodes** · ~200k edges · ~180–250k LOC · ~600k tokens
Team / period	**4 developers** · 12 months (264 active dev-days)
Burn today	**~12M tokens/day ≈ $40/dev/day**
Calibration	vs Anthropic ~$13/dev/day average · 3× for heavy agentic active coding

GraQle's **own** codebase, for scale: **64,449 nodes / 217,222 edges**. This is dogfooded — not theoretical.

Heavy agentic users sit at $40–100/dev/day; team bills of $5k–$15k/mo validate the midpoint.

THE LINE THAT GOES ON THE SLIDE

$42,240 → $5,174

per 4-developer team, per year

−53% in Year 1 −88% in Year 2

THREE WORKFLOWS · ONE TEAM · 12 MONTHS

The math

SCENARIO	$/DEV/DAY	ANNUAL (TEAM OF 4)	SAVING VS A
A Flat-file baseline	$40.00	**$42,240**	—
B **GraQle + Sonnet 4.6**	**$18.82**	**$19,874**	**−$22,366 (−53%)**
C_1 **GraQle + local SLM · Yr 1**	**$16.60**	**$17,530**	**−$24,710 (−58%)**
C_2 **GraQle + local SLM · Yr 2**	**$4.90**	**$5,174**	**−$37,066 (−88%)**

53% saved in Year 1 with GraQle alone — before you change anything else.

WHERE THE SAVING COMES FROM

GraQle activates the relevant subgraph, not the whole repo

Activity	Flat-file	GraQle	Save
12× cross-file reason	1.00M	0.14M	86%
3× impact analysis	0.25M	0.02M	92%
4× debug (3 rounds)	1.00M	0.06M	94% ★
2× code-review	0.20M	0.05M	75%
0.3× sentinel review	0.50M	0.07M	86%
TOTAL / dev / day	3.00M	0.35M	88%

8–25k focused tokens
vs **84k+ flat-file.**

★ **Debugging wins biggest:** `graq_learn` writes a failure pattern **once**; future bugs activate the cached node instead of re-feeding the full context. **The graph gets cheaper as it learns.**

NOT A MARKETING EXTRAPOLATION

Independent research backs the mechanism

- **Biomedical KG study (SPOKE, arXiv 2311.17330):** minimal-schema KG context + embedding pruning → **>50% token reduction, no accuracy loss.** GraQle applies the same to *code*.
- **Code-reasoning study (2025):** context-aware token reduction in repair tasks **cuts cost without degrading quality.**
- **Multi-agent debate (the inverse):** 5 rounds × 4 agents costs **90–101× more tokens** than single-agent. Every flat-file parallel-agent team pays this tax today.
- **GraQle dogfooded: 88% token reduction** on real `graq_reason` queries.

SCENARIO C — THE YEAR-2 CLIFF

Local model. Commodity GPU. Pays for itself in ~2 weeks.

HARDWARE

$1,600

RTX 4090, amortized 3 yrs

BREAK-EVEN

~500M

tokens processed

TEAM ANNUAL VOLUME

12.7B

= 25× break-even

Same KG-anchored workflow, model runs **locally** (Qwen3-Coder-Next / DeepSeek-class). The hardware isn't a science project — **it's paid off in ~2 weeks of this team's token volume.**

40 DEVELOPERS · 10 TEAMS

The number that gets a CFO's attention

YEAR 1 · GRAQLE + FRONTIER

~$224k

saved / year

YEAR 2 · GRAQLE + LOCAL SLM

~$371k

saved / year

TOKEN REDUCTION

88%

dogfooded on graq_reason

The savings scale linearly with team size. And the same substrate that cuts the bill also produces the **compliance trail** → next slide.

COST STORY = COMPLIANCE STORY

EU AI Act-aligned — by 2 August 2026

- **Article 26** (high-risk deployer duties) binds **2 Aug 2026**.
- Requires **human oversight**, **logs kept ≥6 months**, monitoring + incident reporting.
- Non-compliance: **€15M or 3% of global turnover.**

GraQle's **tamper-evidence + audit trail** is exactly the traceability Article 26 demands — generated automatically as you save tokens.

AN ARTICLE 26 FINE WIPES OUT

354 years

of this team's Scenario-A token spend

Sources: artificialintelligenceact.eu/article/26; EC digital-strategy (Aug-2026 high-risk applicability).

NEW IN V0.72.1 · OPEN-SOURCE ON PYPI

The savings number is now authentic

Real tokens, valued at the **real per-model price**, from one dated source of truth (`graqle/pricing.py`) — and it moves with the model you actually run.

MODEL	$ / 1M IN	$ / 1M OUT	COST SAVED @ 88.2M TOKENS
Claude Haiku 4.5	$1	$5	$88.20
Claude Sonnet 4.6	$3	$15	$264.60
Claude Opus 4.8	**$5**	**$25**	**$441.00**

Measured · remembered (`.graqle/metrics.json`) · emitted (`graq metrics`) — and it ships in the public Community wheel.

Prices as of 2026-05-26 (official Anthropic pricing). Opus is 3× cheaper than the deprecated 4/4.1 ($15/$75).

THE CLOSE

Cost down. Speed up. Compliance in.

Flat-file AI coding is a metered-token money fire. GraQle is the substrate that makes the bill **53–88% smaller** — and the only one that **proves** the saving at the real per-model price.

$42,240 → $5,174 / 4-dev team / yr

88% token reduction (dogfooded)

Authentic · dated · **open-source**