



PORTFOLIO CASE STUDY

Sausage Battlefield

Decision models vs foundation models: a playable benchmark of latency and cost for a well-defined decision.



Made by Benny R

CONTENTS

01	Why this was built	3
02	The concepts, plainly	4
03	The evidence	9
04	Limits of this evidence	11
05	When to use which	12
06	Decision models in marketing automation	13
07	How the game demonstrates this	15
08	How it is built	21
09	Try it	22
10	Get in touch	23

Summary

Executive summary

BOTTOM LINE

For a decision with a known set of options, a purpose-built decision model answered about as often in line with the game engine as nine foundation models did, at a fraction of the latency and cost. Jev took a median 286 ms and \$0.0000525 per decision. The foundation models took 869 ms to 6.5 s and \$0.000184 to \$0.0118.

Jev median latency

286 ms

foundation models 869 ms – 6.5 s

Jev cost per decision

\$0.0000525foundation models \$0.000184 –
\$0.0118

Agreement with engine

85%

Jev 17 of 20; others 80–90%

Sausage Battlefield is a fully playable browser game built to make one trade-off visible. Teams often send every AI decision to a large foundation model, including small choices with a fixed list of answers. This project measures what that costs, per decision, and lets anyone watch the difference in a live match.

- Jev was **3.0× to 23× faster** at the median and **3.5× to 225× cheaper** per decision than the nine foundation models reported (a tenth, Grok 4.7, was slower still at 9.2 s and left out after timeouts).
- The gap compounds: a million decisions cost about \$52.55 on Jev, \$184 to \$11,810 on the others.
- The question comes from marketing automation, where most everyday decisions have this shape (section 06).
- Every number comes from one recorded run (2026-10-05) from Australia: one location, 20 situations, one task type. Section 04 sets out the limits.

01

Why this was built

For over 25 years, I’ve been working to make marketing and growth automation smarter. In recent years, that has meant enhancing marketing stacks with AI, and in almost every engagement, one question keeps coming up: which model should we use, and why that one? This project started with that question.

Marketing and growth teams increasingly route every customer journey and automation decision through a large foundation model, simply because it’s the tool already on hand. That includes small, well-defined choices: which offer to show, which channel to use, which branch of a journey a contact takes.

The problem

A foundation model is built to generate text for almost any task. For a choice between a handful of known options, that breadth is paid for on every call. The model reads a long prompt, may spend tokens thinking, writes its answer as text and the application parses it. Each step adds latency and cost. One decision is cheap enough to ignore. A marketing platform that makes a decision for every contact, at every step, on every send, is not.

The idea

When the options are known in advance, a purpose-built decision model can answer the same question directly. It returns a probability for each option and a confidence, rather than generated text. The hypothesis this project tests is simple: for that kind of decision, a decision model is much faster and cheaper, with comparable quality.

Why a game

Latency and cost per decision are abstract, and a real marketing platform is hard to show in public without client data. A game makes the same trade-off concrete. Each turn, the computer player must choose one move from a shortlist, which is exactly a well-defined decision. Players can watch the decision model choose, see its probabilities and how sure it is, play against it, and compare its speed and cost with a foundation model's, one decision at a time.

The point is not that one kind of model is better. It is that the right tool for a well-defined decision is often not the largest one.

Decision model tested	Jev, TypeSafe's "System One" model (<code>jev-latest</code> , served as <code>jev-1.13.0</code>)
Foundation models tested	9 models from Anthropic, OpenAI, xAI, DeepSeek and Moonshot: Claude Opus 5.5, Claude Haiku 4.5, GPT-6 Astra, GPT-6 Luna, Grok 4.20 (non-reasoning), DeepSeek V4 Pro, DeepSeek Flash, Kimi K3, Kimi K2.6
The decision	Pick one move from up to six, plus three side questions (biggest threat, danger level, use a special weapon now?)
Where it applies	Marketing automation: next-best action, journey routing, channel, send time, variant choice and lead routing (section 06)
What was measured	Latency per decision, tokens, cost per decision, and agreement with the engine's top-ranked move

02

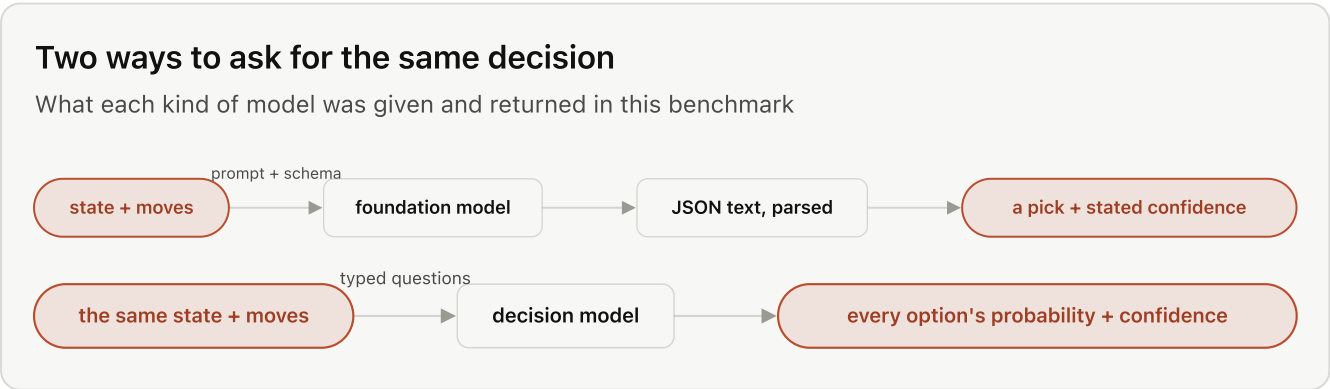
The concepts, plainly

Foundation models and decision models

A **foundation model** (Claude, GPT, Grok, DeepSeek, Kimi) is general-purpose. It generates text, so it can write, summarise, reason and answer almost any question. A **decision model** such as Jev does one thing: given a situation and a set of options, it picks one. It returns typed answers (a choice, a score, or a yes/no probability), and TypeSafe describes the probabilities as calibrated.

	FOUNDATION MODEL	DECISION MODEL (JEV)
Built for	Almost any language task	Choosing from options you define
Returns	Generated text (here, JSON that is parsed and checked)	A probability for every option and a confidence
Thinking	Some models cannot switch it off; those tokens are billed	Not applicable
Billing in this test	Input and output tokens	Input tokens only, \$42 per billion; output not billed
Good at	Open-ended work, explanation, broad knowledge	Fast, repeatable choices with a confidence to act on
Weak at	Speed and cost for small, repeated choices	Numbers and counting ("not a calculator"); open-ended output

Table 1. The two kinds of model compared on the properties that matter for a well-defined decision. Sources: docs/jev.md, TypeSafe documentation.



The game applies this split deliberately. Code does what code is good at; the decision model does what it is good at:



02 · continued

Latency and cost per decision

Latency is how long one decision takes, from sending the question to having a usable answer. This benchmark reports the median (p50), the time half the calls beat, and the slowest of each model’s 20 calls, the slow tail. (With 20 calls, a 95th percentile can’t be told apart from the slowest call, so we report the slowest.) In an interactive product, latency is time a person spends waiting.

Cost per decision is the tokens a call uses multiplied by the provider’s price. It is a fraction of a cent for every model here, which is why it is easy to ignore. It matters because decisions repeat. A game makes about 12 computer decisions per match. A support queue, a fraud check or a routing service can make millions.

MODEL	COST			WAITING TIME (MEDIAN × DECISIONS)	
	PER DECISION	PER 1,000	PER MILLION	ONE MATCH (12)	1,000 IN A ROW
Jev	\$0.0000525	\$0.0525	\$52.55	3.4 s	4.8 min
DeepSeek Flash	\$0.000293	\$0.293	\$293	10.4 s	14.5 min
GPT-6 Luna	\$0.000184	\$0.184	\$184	41.9 s	58.2 min
Claude Haiku 4.5	\$0.00153	\$1.53	\$1,533	15.9 s	22.1 min
Claude Opus 5.5	\$0.00845	\$8.45	\$8,446	32.9 s	45.7 min
GPT-6 Astra	\$0.0118	\$11.81	\$11,810	41.3 s	57.3 min

Table 2. The same per-decision measurements scaled up. Six of the ten models shown: Jev, the fastest and cheapest foundation models, both Anthropic models and the most expensive. Cost is mean measured tokens × dated price; waiting time is the median latency multiplied out, as if decisions ran one after another.

 At one decision, the difference is under a cent and a few seconds. At a million decisions, it is **\$52.55 against \$11,810**, and the waiting time is the difference between an answer that feels instant and one a user notices.

Running calls in parallel shortens total wall-clock time, but not the wait for any single decision. Prices are per decision as measured; volume discounts, batch pricing and caching beyond what providers reported were not applied.

02 · continued

Calibrated confidence and routing

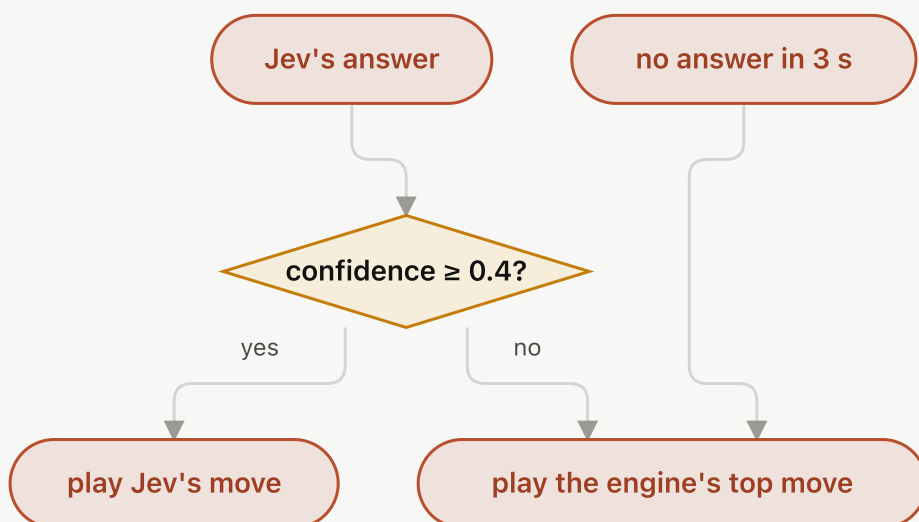
A decision model gives a probability for each option. **Confidence** turns that into one number: how far the top option stands above an even split. Jev's documented formula for a choice is $(p_{\max} - 1/n) / (1 - 1/n)$, which is 0 when every option is equally likely and 1 when one option is certain. **Calibrated** means the probabilities can be read at face value: of answers given at 70%, about 70% should be right. That is TypeSafe's claim for Jev; this project uses the confidence but did not test calibration itself.

WORKED EXAMPLE, FROM A REAL CALL

Six moves; Jev gave the top one 0.73. Confidence = $(0.73 - 1/6) / (1 - 1/6) = 0.67$. That is above the 0.4 bar, so Jev's move was played. The call took 278 ms upstream and 314 ms for the whole round trip.

Confidence routing

The answer says what to do; the confidence says whether to act on it



Why routing matters. A confidence score lets the application decide when to trust the model. Uncertain answers go to a fallback that costs nothing extra: here, the game engine's own top-ranked move. In the live tournaments, 12% of Jev's decisions fell under the bar and went to the engine. The same recorded answers show how the bar works as a dial:

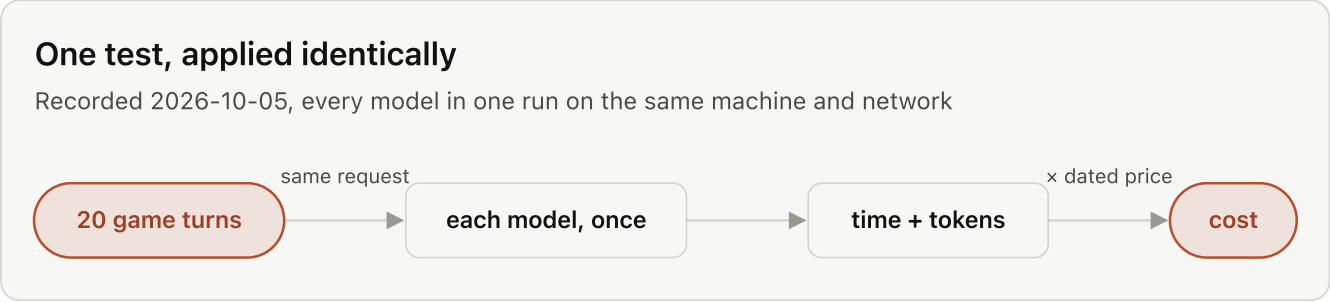
ROUTING BAR	0.4 (SHIPPED)	0.5	0.6	0.7
Times Jev overrode the engine, of 240 decisions	19	13	6	5

Table 3. One live run (240 decisions) replayed at different bars. 17 of the 19 overrides at 0.4 chose a move the engine itself rated "strong". Source: docs/jev.md.

02 · continued

What makes this a fair test

A comparison is only useful if every model does the same job under the same conditions. The benchmark was designed so that the only thing that changes between rows is the model.



- **The same question.** Twenty real turns from the game’s headless engine (fixed seeds), each sent as the exact words-only request Jev gets in play. Foundation models got the same JSON with one fixed system prompt and one fixed answer schema, enforced by structured output where the API supports it.
- **The same machine and network.** Every model, Jev included, was called in one run from the same machine and network in Australia, one call per situation, with the order rotating each round and an unmeasured warm-up call first.
- **Each model at its fastest sensible setting.** Thinking off where it can be switched off; the lowest reasoning effort where it cannot (Claude Opus 5.5, GPT models, Kimi K3).
- **Measured, not estimated.** Latency is the wall-clock round trip. Tokens are as each provider reported them. Cost is tokens × the provider’s standard price on a recorded date.

What is real and what is simulated in the game

IN “JEV VS THE BIG MODELS”	STATUS	HOW
Jev’s decisions	Live	A real call to TypeSafe for every decision; latency timed by the game, tokens and model id from TypeSafe’s response.
The opponent’s moves	Simulated	Chosen by the game’s own engine, standing in for the foundation model.
The opponent’s time and cost	Replayed	Drawn from that model’s recorded benchmark calls.

Table 4. The site never calls Claude, GPT, Grok, DeepSeek or Kimi. Labels on screen say Live or Simulated wherever a number appears.

02 · continued

Why the foundation models are simulated

In “Jev vs the big models”, Jev plays live but the foundation model’s side does not. That was a deliberate design choice, not a shortcut. The comparison the game makes is about **latency and cost per decision**, and those were measured live, under controlled conditions, in the benchmark. The game replays those measurements instead of calling nine paid APIs on every visitor’s turn.

- **Open-ended cost on a free, public site.** Every match would bill the site owner. 12 decisions a match means about \$142 per 1,000 matches on GPT-6 Astra and \$101 on Claude Opus 5.5, against \$0.631 on Jev. An anonymous, kid-friendly game cannot carry that, and the number grows with every visitor.
- **Keys and abuse.** Live calls would put nine providers’ paid API keys behind a public endpoint. Jev’s single key is already protected by a human check, a short-lived signed pass and a rate limit; multiplying that exposure by nine adds risk for no extra insight.
- **A fair, repeatable comparison.** Live latency depends on where the visitor is, the time of day and each provider’s load. Replaying one controlled run means every visitor sees the same measured numbers, all taken on the same machine and network, so the rows stay comparable.
- **Reliability during play.** Slow tails would stall a turn: Kimi K3 took up to 23.4 s on its slowest call, and Grok 4.7 timed out often enough to be excluded. A game needs a turn to finish.

WHAT THIS TRADE-OFF COSTS

In the game, the opponent’s **moves** are chosen by the game engine, not by the foundation model, so a simulated match says nothing about how well that model would play. Decision quality is measured separately, in the benchmark, as agreement with the engine’s top-ranked move. The screens label this honestly: **Live** for Jev, **Simulated** for the opponent, and the opponent’s panel never shows a confidence it did not produce.



Jev is kept live on purpose: it shows a decision model answering in real time, inside a real game loop, at a cost low enough (about \$0.000631 a match) to run for every visitor.

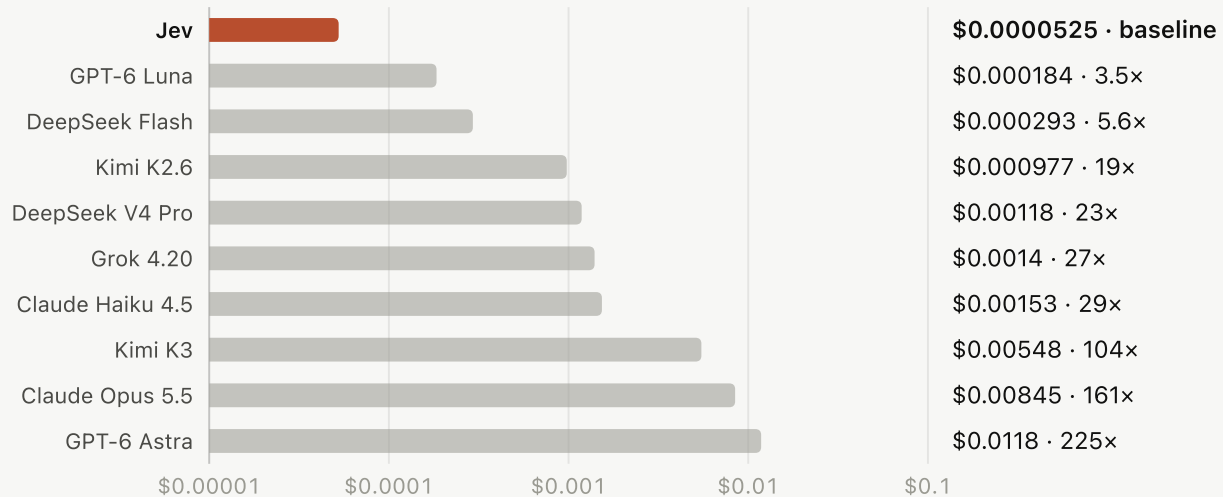
03

The evidence

Jev was the fastest and cheapest model in the run: by a wide margin on cost, a clear one on speed.

Cost per decision

US dollars, log scale (each gridline 10×). Recorded 2026-10-05.

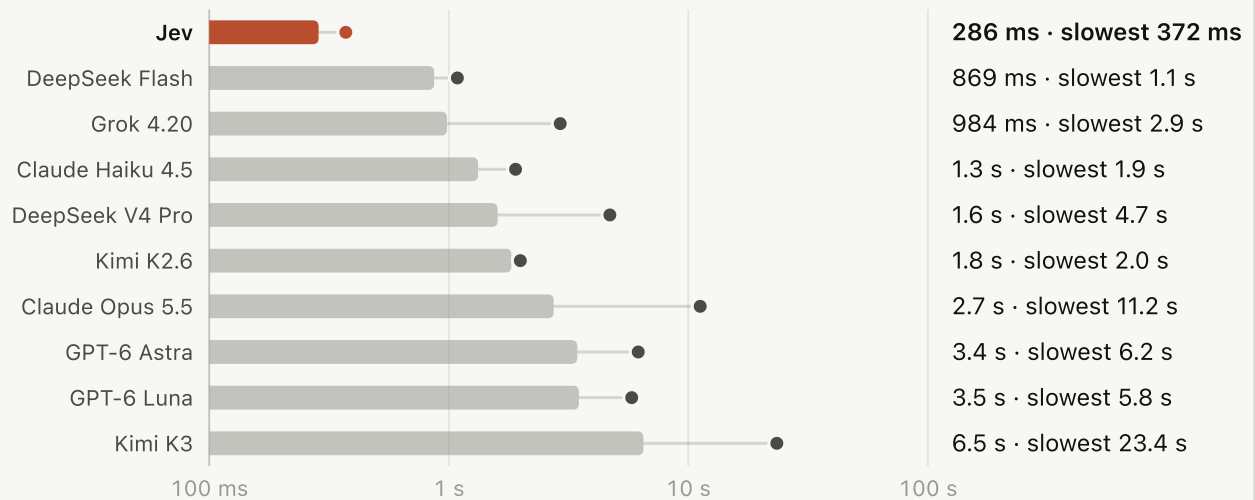


Source: src/jev/benchmark.json. Mean measured tokens × the provider's dated standard price. Grok 4.20 is the non-reasoning variant.

Figure 1. Cost per decision, sorted. Ratios are relative to Jev.

Latency per decision

Bar: median. Dot: slowest of 20 calls. Log scale, 100 ms to 100 s.



Source: src/jev/benchmark.json. Wall-clock round trip, 20 calls per model, from Australia.

Figure 2. Median and slowest of 20 calls, sorted by median.

03 · continued

Every model, every number

MODEL	PROVIDER	COST / DECISION	VS JEV	P50	SLOWEST	VS JEV	AGREES
Jev	TypeSafe	\$0.0000525	–	286 ms	372 ms	–	17/20
GPT-6 Luna	OpenAI	\$0.000184	3.5×	3.5 s	5.8 s	12×	18/20
DeepSeek Flash	DeepSeek	\$0.000293	5.6×	869 ms	1.1 s	3.0×	17/20
Kimi K2.6	Moonshot	\$0.000977	19×	1.8 s	2.0 s	6.4×	17/20
DeepSeek V4 Pro	DeepSeek	\$0.00118	23×	1.6 s	4.7 s	5.6×	17/20
Grok 4.20 (non-reasoning)	xAI	\$0.0014	27×	984 ms	2.9 s	3.4×	18/20
Claude Haiku 4.5	Anthropic	\$0.00153	29×	1.3 s	1.9 s	4.6×	18/20
Kimi K3	Moonshot	\$0.00548	104×	6.5 s	23.4 s	23×	16/20
Claude Opus 5.5	Anthropic	\$0.00845	161×	2.7 s	11.2 s	9.6×	16/20
GPT-6 Astra	OpenAI	\$0.0118	225×	3.4 s	6.2 s	12×	17/20

Table 5. Recorded 2026-10-05, 20 situations, every model answered all 20. “vs Jev” is cost or median latency divided by Jev’s. “Agrees” counts how often the model picked the engine’s top-ranked move. Grok 4.7 is not shown: timed out repeatedly during the benchmark (2 of 20 calls).

✔

Jev was the cheapest and fastest model tested

\$0.0000525 and 286 ms per decision. The nearest on cost was GPT-6 Luna at \$0.000184 (3.5×); the nearest on speed was DeepSeek Flash at 869 ms (3.0×).

Cost 3.5×–225× lower · median 3.0×–23× faster

✔

Agreement was comparable

Jev matched the engine’s top-ranked move in 17 of 20 situations; the foundation models matched in 16 to 18 of 20. A gap of one or two situations in 20 is not evidence of a quality difference either way.

Jev 85% · foundation models 80–90%

⚠

Tail latency varied most among models that think

Kimi K3 (always thinks) had a slowest call of 23.4 s and Claude Opus 5.5 one of 11.2 s. Grok 4.7 was left out after two of its 20 calls hit the 60-second timeout.

Jev slowest 372 ms

04

Limits of this evidence

The benchmark answers one narrow question well. It does not settle which model is “better”, and it should not be read that way.

! ONE LOCATION, ONE RUN

Every call came from one machine and network in Australia. Absolute latency includes the distance to each provider, the same for every model but not the same as from another city or a data centre. In the game, through the Cloudflare Worker, Jev’s live round trip was a median of 234–236 ms over 481 decisions.

! TWENTY SITUATIONS, ONE TASK TYPE

Twenty decisions per model is enough to show differences of 3x to 225x. It is not enough to rank models that are close, and it covers one kind of decision: choosing a move in this game.

! AGREEMENT IS NOT CORRECTNESS

Agreement is measured against the engine’s top-ranked move, which is itself a heuristic. In the live tournaments, overriding the engine usually cost the game: Jev went 7–13 and 8–12 against Professor Woof, where the engine alone went 12–8 on the same seeds. Jev makes the computer player explainable and steerable, not a better shot.

i PRICES ARE DATED

Prices are each provider’s standard rate on the recorded date (2026-09-25 to 2026-10-06). Off-peak, batch and volume discounts were not applied. Prices change; the formulas in the game’s “How we worked this out” screen show the inputs so the sums can be redone.

i THE COMPARISON SLIGHTLY FAVOURS THE FOUNDATION MODELS ON COST

Jev returns a probability for every option; the foundation models were asked only for a pick and a confidence, so they wrote fewer output tokens than a like-for-like answer would need.

i SETTINGS MATTER

Each model ran at its fastest sensible setting. Other settings (more reasoning, a different region, provisioned throughput) would change its numbers. One flagship, Grok 4.7, was excluded after repeated timeouts; its data is kept in the archive, marked excluded.

05

When to use which

These are takeaways from one benchmark and the experience of building the game, not universal rules. They are best read as questions to ask before choosing a model for a decision.

A decision model fits when...

- the answer is one of a known set of options;
- the decision repeats often, so cost per decision adds up;
- someone is waiting, so latency is felt;
- you want a confidence score to decide when to act;
- the inputs can be described in words, with any arithmetic done in code first.

A foundation model fits when...

- the output is open-ended: writing, summarising, explaining;
- the options are not known in advance;
- the task needs broad knowledge, tools or multi-step reasoning;
- volume is low, so per-call cost and latency matter less;
- a person needs a written explanation, not just a choice.

A pattern that combines them

1 Let code do the maths and generate the options

In the game, the engine simulates about 40 shots and shortlists up to six distinct moves. The model never counts or calculates.

2 Ask a decision model to choose, in words

One request with typed questions. Describing options in plain buckets ("big hit", "safe for our team") suits a model that is not a calculator.

3 Route on confidence

Act on confident answers; send uncertain or late ones to a fallback. Here the fallback is the engine; elsewhere it could be a rule, a person or a larger model.

4 Measure on your own task

Run the same requests through each candidate, from where your users are, and compare latency, cost and agreement before committing.

06

Decision models in marketing automation

Marketing automation is where this question matters most in my work. A marketing platform makes a very large number of small, well-defined decisions, for every contact, at every step of a journey, often while the person is still on the page or in the inbox.

Most of those decisions have the same shape as a turn in the game: a known set of options, some context about the contact, and one choice to make now. Teams increasingly hand them to a foundation model because it is already wired in. The game was built to show, with measured numbers, what that habit costs.

DECISION	THE KNOWN OPTIONS	HOW OFTEN IT IS MADE
Next-best action	A catalogue of actions: an offer, a piece of content, a nudge, or do nothing	Every visit, event or journey step
Journey routing	The branches already drawn in the journey	Every contact at every split
Channel choice	Email, SMS, push, in-app, or hold	Every send
Send time	A set of time windows	Every contact, every send
Offer or content variant	The variants already written and approved	Every impression, often in real time
Lead scoring and routing	Hot, warm or cold; which team or rep	Every new lead or form fill
Reply and intent triage	Question, buying signal, complaint, unsubscribe, out of office	Every inbound reply
Eligibility and suppression	Send, hold or exclude	Every send, before anything goes out

Table 6. Typical decisions in a marketing automation platform. Each picks one answer from a list that is known before the question is asked, which is the kind of decision the benchmark measured.



In the game, Jev chooses one move from up to six. Swap “move” for “next-best action”, “channel” or “variant” and the request is the same: describe the situation in words, list the options, ask for one choice and a confidence.

06 · continued

What it means at campaign scale

Take an illustrative programme: **100,000 contacts**, each touched by **5 automated decisions a week** (a routing split, a channel, a send time, a variant and an eligibility check). That is 500,000 decisions a week, or 26 million a year, which is modest for a mid-sized brand.

MODEL	PER DECISION	PER WEEK	PER YEAR	MEDIAN WAIT PER DECISION
Jev	\$0.0000525	\$26.27	\$1,366	286 ms
GPT-6 Luna	\$0.000184	\$91.98	\$4,783	3.5 s
DeepSeek Flash	\$0.000293	\$147	\$7,623	869 ms
Claude Haiku 4.5	\$0.00153	\$766	\$39,855	1.3 s
Claude Opus 5.5	\$0.00845	\$4,223	\$219,606	2.7 s
GPT-6 Astra	\$0.0118	\$5,905	\$307,064	3.4 s

Table 7. Illustrative volume, measured prices: the per-decision cost and median latency are the game benchmark's (2026-10-05); the volume is an example, not a client's figures. A marketing request may be longer or shorter than a game turn, so treat these as orders of magnitude.

Why it matters here in particular

- **Volume compounds.** Marketing decisions repeat for every contact and every send, so a fraction of a cent becomes a line in the budget. Cost that scales with the list makes teams ration where they use AI.
- **Real time is common.** Web and app personalisation is decided while the page loads. 286 ms fits inside that; 2.7 s at the median, and 11.2 s on its slowest call, does not.
- **Confidence gives a safe default.** When the decision model is unsure, the contact goes to the default branch, the control group or a person to review, exactly as the game falls back to its engine. That is easier to explain to a brand, legal or compliance team than free text.
- **The choice is auditable.** A probability for each option, logged with the decision, shows why a contact got what they got.

Foundation model: writes

Drafts subject lines, copy and variants, summarises replies, explains results. Low volume, once per campaign, reviewed by people.

Decision model: chooses

Picks the variant, channel, time and next step for each contact. High volume, every send, routed on confidence.

 **MEASURE ON YOUR OWN DECISIONS**

The benchmark measured a game decision, not a marketing one. The pattern carries over; the exact numbers will not. Before committing, run your own decisions through each candidate from where your platform runs, and describe contacts in words without sending personal data the decision does not need.

07

How the game demonstrates this

The game is a full product, not a mock-up: single player, pass-and-play on one device, private online matches by link, and two modes where the computer plays itself. Each screen below is captioned with the idea it shows.



Figure 3. A familiar wrapper for an abstract trade-off. The start page. Under PLAY, "It's not all fun and games" opens the explainer for what the game is really for.



Figure 4. Same decision, different deciders. Choose your battle. Single player and Robot Rumble use the free, offline engine; Jev Rumble hands the final choice to Jev.

07 · continued

Watching a decision model decide

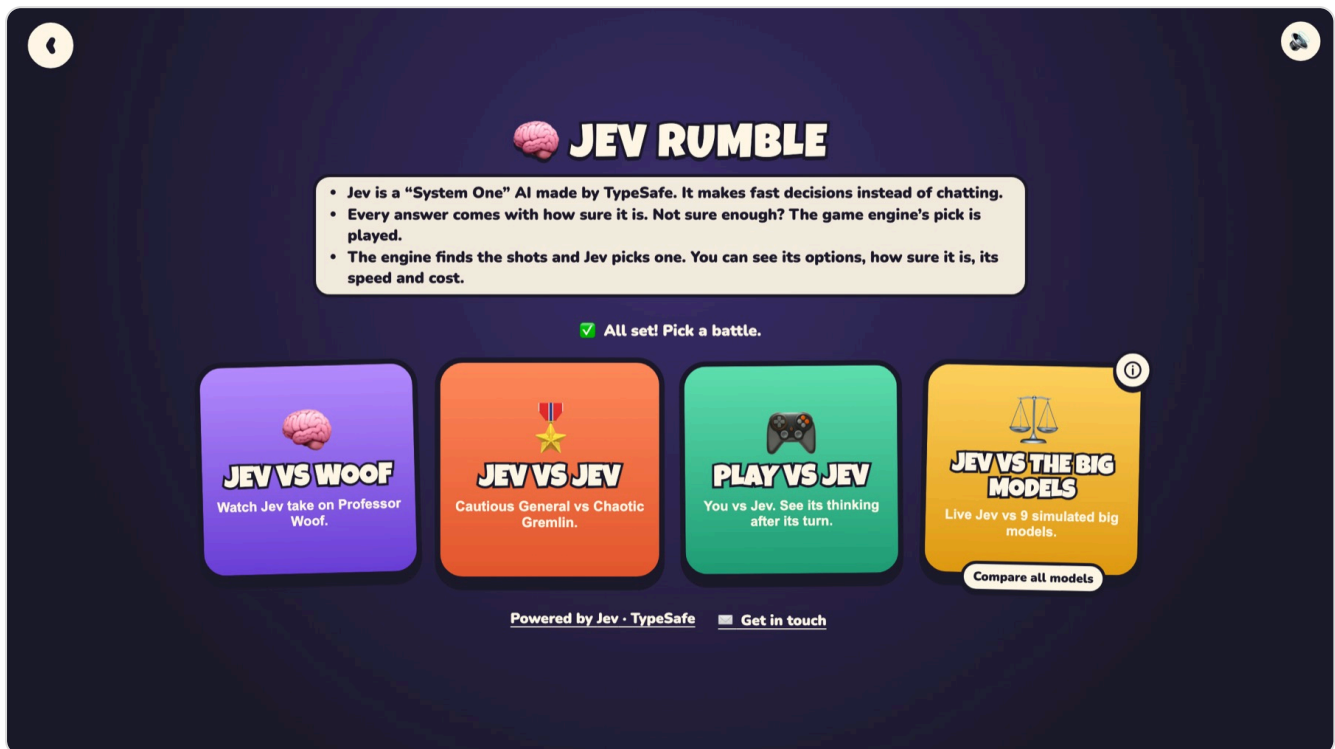


Figure 5. Protecting a paid API on a public page. The Jev Rumble hub. A mostly invisible Cloudflare Turnstile check swaps for a signed 30-minute pass before any real Jev call; the key never reaches the browser.



Figure 6. Probabilities and confidence on show. Jev vs Professor Woof, a live match. Jev's brain lists every shortlisted move with its probability, marks the engine's pick and the move played, and shows who decided.

07 · continued

Live against simulated

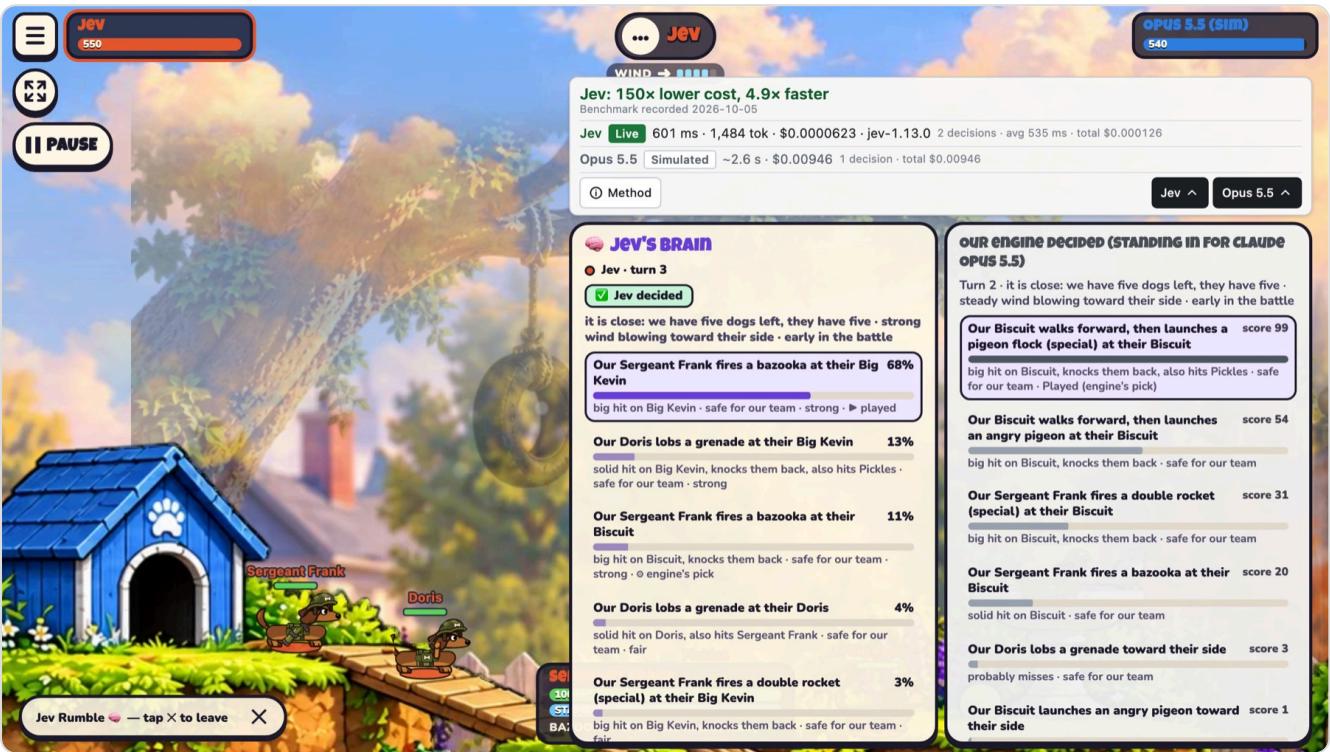


Figure 7. Real versus simulated, labelled on screen. Jev vs the big models. Jev's decisions are live calls (green Live tag); the opponent's moves come from the engine, with Claude Opus 5.5's recorded time and cost replayed (Simulated tag). Its panel shows engine scores, never invented probabilities.

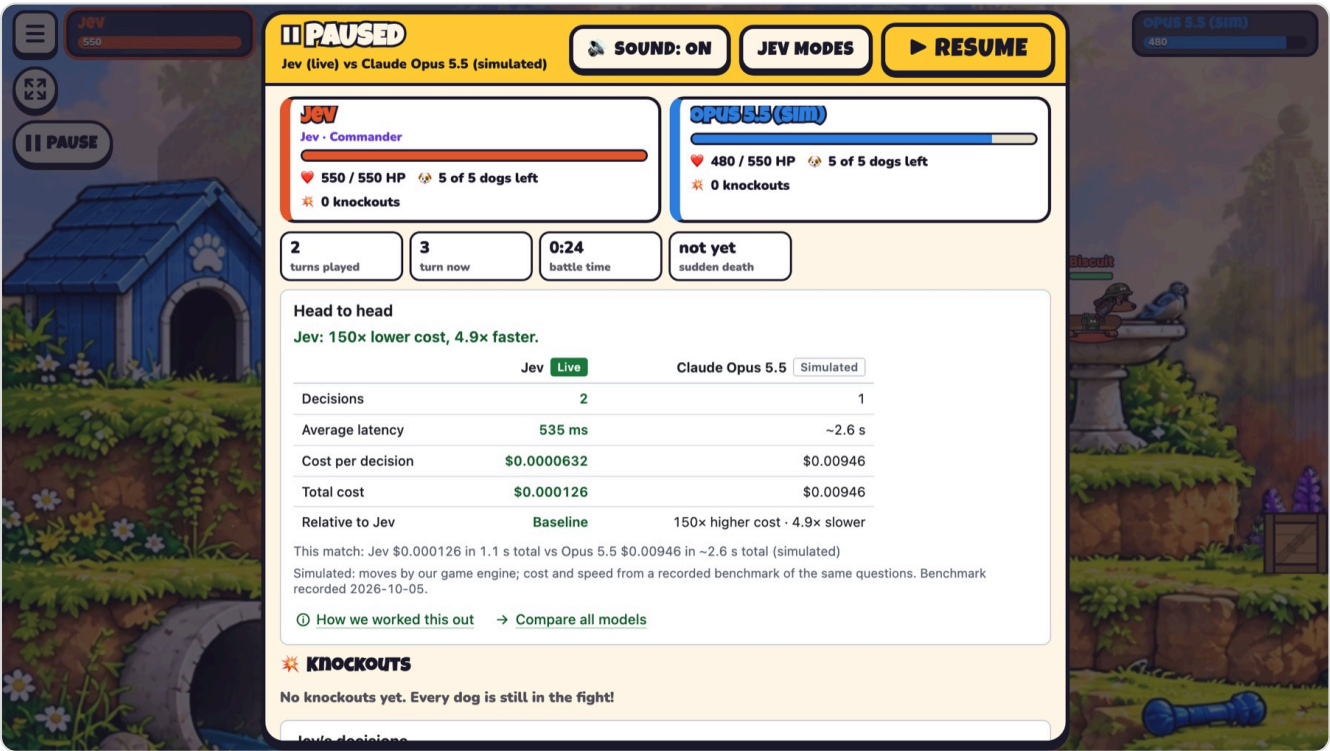


Figure 8. Per-decision costs compound over a match. The pause card's head-to-head, two decisions into a live match. Live figures include this machine's round trip through the local Worker, so they differ from the benchmark medians.

07 · continued

Showing the working

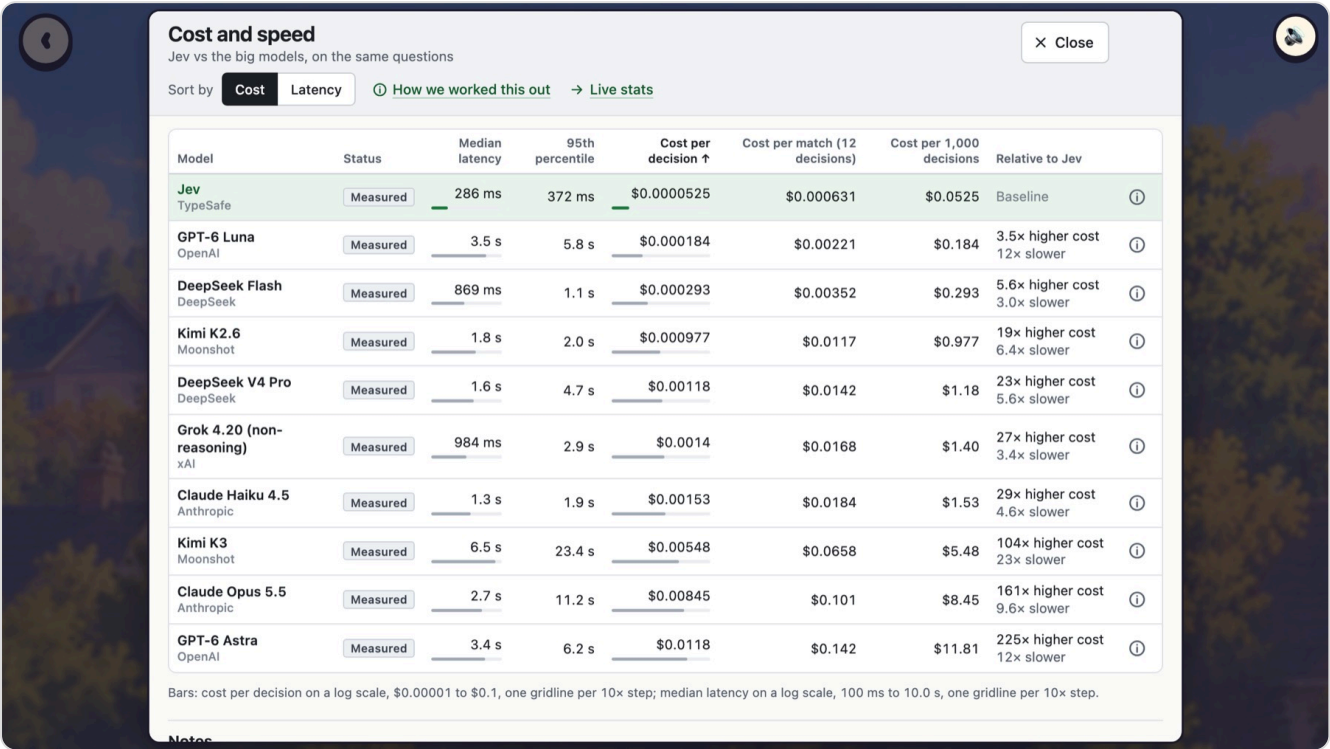


Figure 9. Log scales keep a 225× spread readable. The cost and speed comparison, sortable by cost or latency, with every model’s status shown as Measured. The same data as section 03.

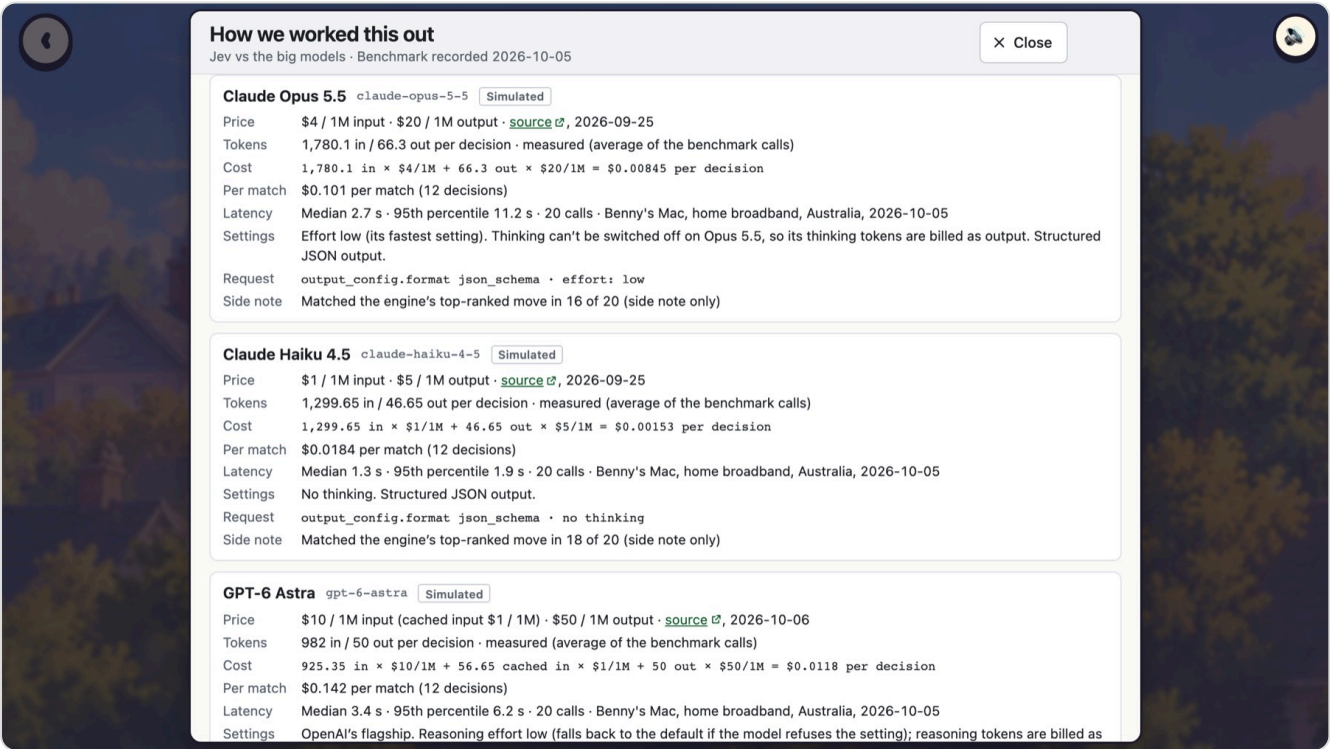


Figure 10. Every number traceable to tokens × a dated price. “How we worked this out” prints each model’s price, source and date, measured tokens and the exact formula behind its cost per decision.

07 · continued

On a phone, and in the open



Figure 11. The same decision panel at phone size. A phone in landscape (844 × 390). The brain collapses to a chip ("Jev 78% sure · 354 ms") and opens as a sheet.

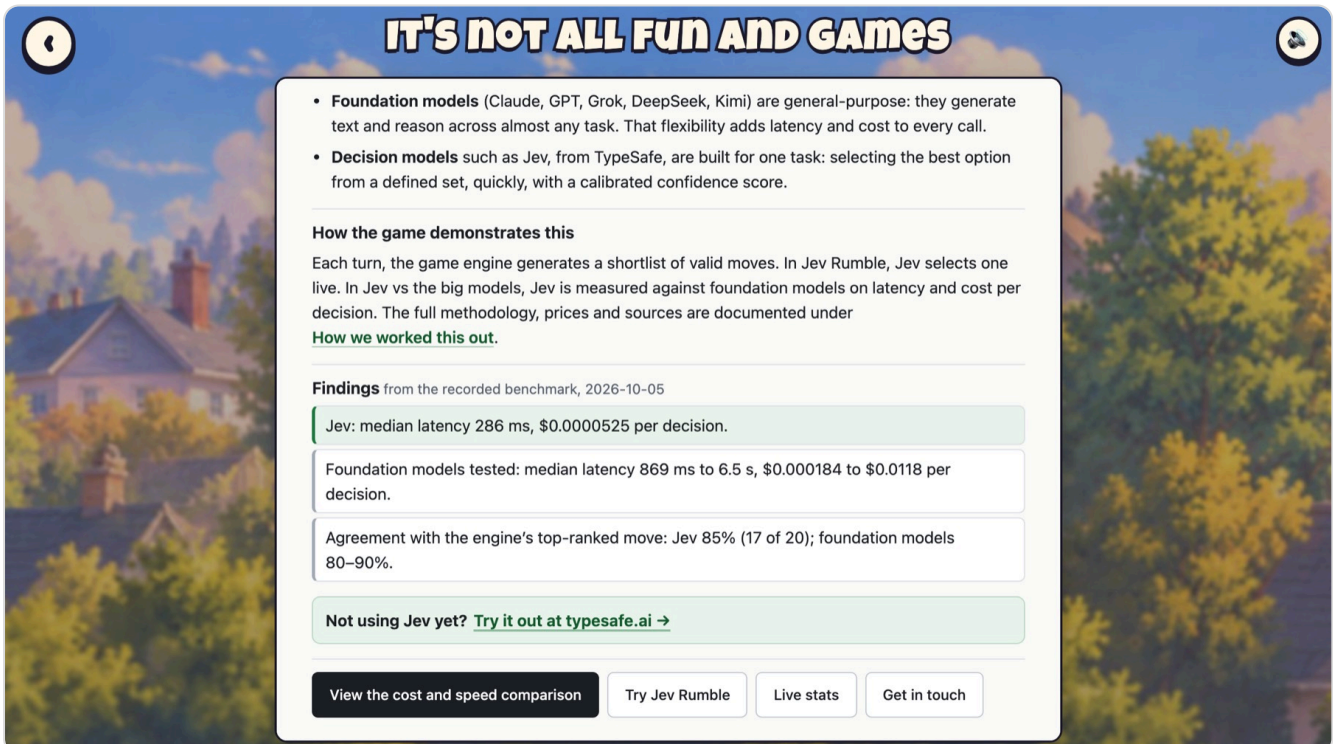


Figure 12. The argument in one page. "It's not all fun and games" explains foundation and decision models and states the findings, generated from the benchmark file rather than typed in.

07 · continued

Measuring in production, privately

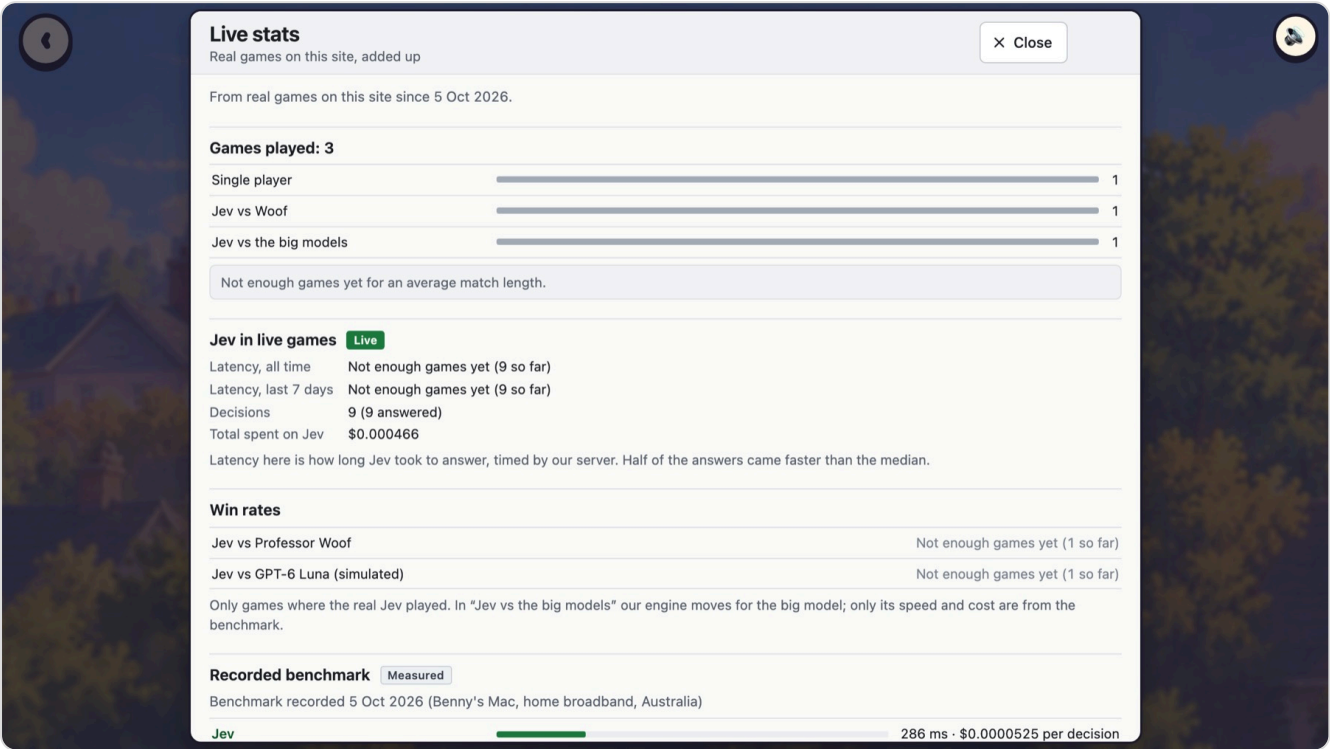


Figure 13. Benchmark and live measurements kept apart. Live stats, shown here from a local test database (3 games, 9 Jev decisions), so most panels honestly read “Not enough games yet”. Live latency is timed inside the Worker; the benchmark is a round trip from one machine.



Figure 14. Anonymous by design. A first-visit notice explains the anonymous usage counts and offers an opt-out at any time from Privacy.

08

How it is built

The whole site runs on Cloudflare: static files from Workers Assets and one Worker for every API route. No other servers.

How the site is built

Five layers; the Worker is the trust boundary that holds every key

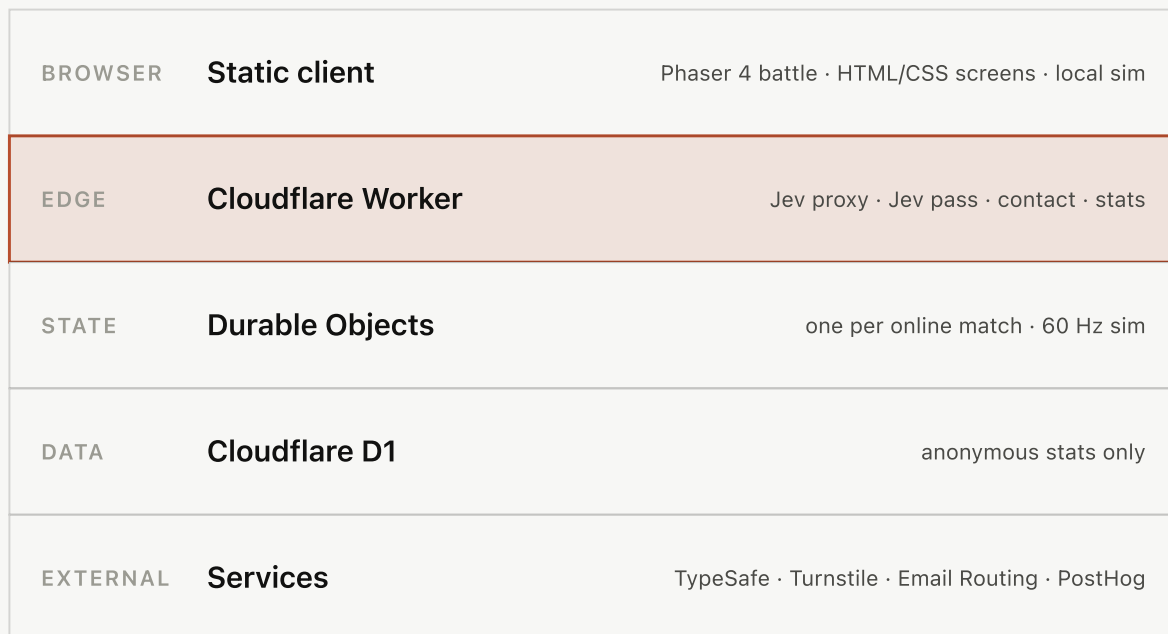


Figure 15. The architecture as a layer stack. The Worker (highlighted) holds the Jev key and the Turnstile secret.

- **Deterministic simulation.** One headless TypeScript simulation (no randomness except a seeded generator, no wall-clock time) runs in the browser for local play and on the server for online play. The benchmark's 20 situations came from it.
- **Online matches.** One Durable Object per match runs the simulation at 60 Hz, sends state about 20 times a second and checkpoints so a refresh or dropped connection rejoins the same seat.
- **Jev proxy.** `POST /api/jev` accepts only the narrow shape the game sends, rate-limits per IP, adds the key, fixes the model and times out after 4 s. Upstream errors are mapped, never passed through.
- **Jev pass.** Turnstile proves a browser is human once; the Worker issues an HMAC-signed pass valid for 30 minutes.
- **Contact form.** Sent through Cloudflare Email Routing with Turnstile, a honeypot and a rate limit. Nothing is stored.
- **Privacy.** PostHog counts anonymous usage only (no autocapture, no recordings), with an opt-out. The D1 stats hold no names, emails, IPs, user agents, tokens, match ids or free text; rows get random ids and minute-level timestamps.

09

Try it

The fastest way to understand the trade-off is to watch it. Open the game, choose Jev Rumble, and open Jev's brain.

Not using Jev yet?

Jev is TypeSafe's decision model. Try it on your own well-defined decisions.

typesafe.ai

Documentation: docs.typesafe.ai

Play Sausage Battlefield

Free, in the browser, on a phone or a desktop. No account.

sausagebattlefield.made-by-benny.com

Start page → PLAY → Jev Rumble

Sources

- Benchmark data: `src/jev/benchmark.json` (compact) and `docs/jev/benchmark.full.json` (every call), recorded 2026-10-05 21:13 UTC.
- Live tournaments, routing analysis and method: `docs/jev.md`. Product intent: `notes/prd.md`, `notes/plan.md`.
- Prices: Anthropic (2026-09-25); TypeSafe (2026-10-04); OpenAI, xAI, DeepSeek, Moonshot (fetched 2026-10-06). Each source is linked in the game's "How we worked this out" screen.
- Screenshots: a local build of the site (Cloudflare Worker via `wrangler dev`) with live Jev calls, 7 October 2026.

10

Get in touch

If you are working out which model belongs where in your marketing or growth stack, I would be glad to talk it through.

Email

Questions about this project, the benchmark, or your own decisions.

enquiries@benny.com

AI Marketing Engineer

My personal offerings.

aimarketingengineer.co

Agent Labs

Our more commercial offerings.

www.agntlabs.com

Questions about the game itself can also go through the **Get in touch** form on the site: bottom-left of the start page, in the Jev Rumble hub, and at the foot of the "It's not all fun and games" page.

Made by Benny · 7 October 2026