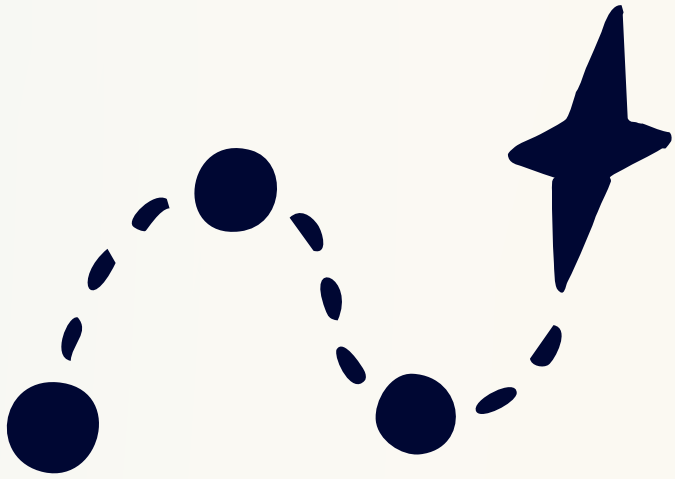
Oasium AI

GUIDE № 01



The build-vs-buy decision matrix for AI tools.

Founders, ops leads, and CTOs sitting in front of an AI build-or-buy decision they cannot afford to get wrong. Useful as a personal worksheet or as a structured way to run the conversation with a board.

STRATEGY · 28 PAGES · PDF

MAY 2026 EDITION

CONTENTS

What's inside

01

The decision you're actually making

Every AI procurement conversation starts with the wrong question.

The wrong question is *"should we build this in-house or buy it from a vendor?"* It sounds tactical. It feels like a 2x2 matrix waiting to be filled. The board likes it because it has two clean options. The team likes it because it's faster than thinking.

It's the wrong question because it presupposes that you should do *something* — that you have a workflow ready, a budget approved, a team aligned, and that the only remaining decision is sourcing. In our experience, fewer than half of build-vs-buy conversations meet those preconditions. Most of the others are using build-vs-buy as a proxy for an unspoken question: *do we even know what we're trying to do?*

The right question is harder, longer, and worth asking before you pick up a vendor demo. It has four parts.

1. What is the workflow we're trying to change? Not the *capability* (e.g., "AI agents," "summarization," "intelligent search"). The *workflow* — a specific sequence of steps a real person does today, with a real input shape, a real output people check, and a real frequency. If you can't write the workflow on one page, you're not ready to buy or build.

2. What's the cost of being wrong, and how often is it wrong today? Existing workflows have an existing error rate that humans tolerate because it's invisible. AI workflows make the error rate visible because they're new. That's not a strike against AI; it's a feature of replacing a familiar process. But the threshold matters. A summarization workflow that misclassifies 5% of inputs is fine if a person reviews; intolerable if it triggers a wire transfer.

3. What's the cost of *not* deciding? This is the cost of waiting. Sometimes it's enormous (your competitor automates and undercuts your price). Sometimes it's negligible (the technology will be cheaper and better in six months, and you don't lose anything by being second). Most of the time, it's between those — and most teams haven't put a number on it.

4. What's the configuration of build, buy, hybrid, or wait that minimizes total cost? Now and only now does the build-vs-buy matrix become useful. By this point you know the workflow, the tolerable error rate, the cost of waiting, and the urgency. You can score two paths against the same criteria. Most teams pick option three — *hybrid* — once they get this far.

The rest of this guide assumes you've answered the first three. If you haven't, stop reading and answer them first. Most of what looks like a build-vs-buy debate is a "we haven't agreed on the workflow" debate in disguise.

02

Why this decision is harder in 2026 than it was in 2024

Three forces have made the build-vs-buy question harder, not easier, in the last twelve months.

The vendor landscape is overcrowded. In 2024 there were roughly a dozen serious AI tooling vendors per category. In 2026 there are hundreds. They all demo similarly. A survey conducted by Forrester in late 2025 found that buyers can correctly distinguish among five competing AI agent platforms only 31% of the time during a structured demo. That's lower than chance for a five-option choice. The implication: vendor evaluation in 2026 isn't just hard — it's structurally unreliable. You cannot pick well by watching demos. You have to evaluate against your own workflow, with your own data, on your own infrastructure.

The cost of "buy" has fallen by an order of magnitude. API prices have dropped roughly 80% between early 2025 and early 2026, and roughly 10x over the past two years. GPT-4-class capability that cost $30 per million input tokens in early 2024 now costs under $1. Claude Opus pricing has fallen from $15/$75 (input/output per million tokens) in mid-2025 to $5/$25 today. OpenAI's smallest frontier models — GPT-5 nano — are at $0.05/$0.40. With cached-input pricing (Anthropic charges 10% of base for cache hits; OpenAI offers up to 90% off), the effective cost of running production AI at any reasonable scale has collapsed.

What this changes: the 2024 case for *building* in-house was largely cost-driven. "We can serve this internally for less than the API price." That argument is much weaker in 2026. The only people for whom on-prem is cheaper are those running massive scale or who have a sovereignty requirement. For everyone else, the cost case for building has flipped — buying is cheaper, often by a wide margin.

The cost of "build" has gone up at the high end. Senior ML engineers and AI architects in 2026 command significantly higher comp than 2024. A mid-market enterprise that needs two-to-five engineers to build a custom agent system is looking at $1M+ in fully-loaded annual cost before they ship anything. That's before infrastructure, evaluation, monitoring, and the inevitable rebuild when the underlying model changes.

The market keeps moving. A vendor you evaluate in February could be acquired, deprecated, or pivoted by August. A model you build against in March will be supplanted by something better in May. Both sides of the build-vs-buy decision now have a half-life shorter than the decision-making cycle most enterprises run on. This is the single biggest reason why "wait" is increasingly the right answer for non-urgent workflows. We come back to this in section 08.

The first practical implication: any framework you use to evaluate this decision in 2026 needs to be **dated**, **falsifiable**, and **revisitable**. A static decision made in Q2 with a 24-month horizon is almost certainly going to be wrong by Q4. Build the decision into a quarterly review.

03

Costs that show up in year two

The biggest mistake in build-vs-buy analyses is treating the comparison as a one-year problem. Year-one costs are visible. They show up in budget reviews, RFPs, and vendor quotes. Year-two costs are mostly invisible until they hit. Both sides — build *and* buy — have these. Both sides hide them.

If you build

COST	WHEN IT SHOWS UP	WHY IT'S INVISIBLE
Eval suite maintenance	Month 4-12	"We'll get to it" eval debt accumulates silently. The day a model release changes behavior is the day you realize you don't have one.
Model migration	Months 6-18	The model you started with will be deprecated, rate-limited, or surpassed. Every prompt + system message + tool schema needs revisiting.
Knowledge concentration	Months 6+	The two engineers who built it become irreplaceable. When they leave, six months of velocity goes with them.
Compliance creep	Months 8-18	What was a research prototype becomes a production system. Now it needs SOC 2, audit logs, data retention controls.
Operational tail	Indefinite	Monitoring, on-call, recovery from outages. None of this was scoped in the original build cost.
Rebuild	Months 12-24	The rate at which "the right way" changes is high enough that what you built in Q2 may need a structural rebuild by Q4 of the next year.

A useful rule of thumb: if you've never built a production AI system before, multiply your initial build estimate by three for true 24-month cost. Industry benchmarks bear this out — Forrester's 2025 study estimated 15-25% annual maintenance cost on initial build investment, but that's only the visible portion. Real teams report 40-60% when migration and rebuild costs are included.

If you buy

COST	WHEN IT SHOWS UP	WHY IT'S INVISIBLE
Integration tax	Months 1-6	Vendor demos run on toy data. Your real data has edge cases that need custom handling. Most integration projects take 2-3x the vendor's estimate.
Lock-in	Month 12+	Switching cost = your accumulated configuration + custom prompts + workflow training + data export friction. Vendors design this in.
Pricing trajectory	Year 2	Usage-based pricing creates surprise spikes. Most vendor contracts allow material price increases on renewal.
Vendor health	Indefinite	The startup you bet on may not exist in two years. Or it gets acquired by a competitor, or pivots, or changes pricing dramatically.
Customization ceiling	Months 6-12	The vendor handles 80% of your case beautifully and the last 20% becomes a procurement of escalation requests.
Compliance + data residency	Months 3-9	Especially in regulated industries or sovereignty-driven regions, vendor SOC 2 + GDPR + your local regulation + their data flows = a long review.

A useful rule of thumb for buy: total Year-2 cost ≈ year-one license cost × 1.5. The increment is integration, expansion, and price increases.

What this means

You cannot evaluate build-vs-buy on Year-1 cost alone. The build path looks cheaper in Year 1 about half the time and more expensive by Year 2 in the vast majority of cases. The buy path looks more expensive in Year 1 and roughly comparable to build by Year 2 — except that you've avoided the technical debt and have someone else's problem to call when things break.

The exception is high-volume, narrow workloads where the per-call economics push hard toward build. We come back to this in section 04.

04

A worked example — a logistics SME with four vendor options

To make the framework concrete, here's a fictional but representative scenario. The numbers are illustrative; the reasoning is real.

The setup

A regional logistics company. 240 employees, $42M annual revenue, growing 22% year-over-year. They run freight matching across the GCC and adjacent regions. The workflow they want to automate: **inbound shipment classification and routing**. When a customer requests a shipment, an operator currently spends 6-8 minutes reviewing the request, classifying urgency, identifying suitable carriers, and producing an initial price quote. The operations team handles roughly 380 such requests per business day. Six operators do this work. The CEO has noticed that the team is bottlenecked at this step, especially during peak season, and that the company's growth is starting to outpace the ops team's capacity to absorb the workload.

Constraints:

- Two engineers on staff. Both are full-stack with no specialized ML experience.
- Annual technology budget for new initiatives: $250K.
- Data is a mix of structured (rates, customer profiles) and unstructured (emails, attachments).
- Some shipments involve regulated cargo with country-specific compliance flags.
- The company's CTO has been to two AI conferences in the last year and feels under-prepared for this decision.

The decision

The CEO wants to know: **should we build this, buy it, or hybrid?** The CTO has identified four vendor options:

- **Vendor A** — A specialized logistics-AI vendor with three customers in the GCC region. License: $48K/year for up to 200K classifications. Implementation: $35K. Support: standard.
- **Vendor B** — A general-purpose AI agent platform. License: $24K/year. Custom integration estimated at $40K. The vendor will provide a solution architect for 90 days.
- **Vendor C** — A frontier-model API (GPT-5.5 + Codex) directly, with the company's engineers writing the integration. Estimated API cost: $9K/year at expected volume. Engineering time to integrate: 4-6 weeks.

- **Vendor D** — An open-source self-hosted approach using Llama 4 Maverick on a Mac Studio M3 Ultra. Hardware: $11K. No software cost. Engineering time to integrate: 8-12 weeks.

Walking through the analysis

Step 1: Define the workflow tightly. The CEO and CTO sit down and write the actual workflow on one page. They discover that what they thought was a single workflow is actually three workflows that share an input but diverge after step two. Standard freight: 78% of volume, low complexity, currently 4 minutes per request. Regulated cargo: 15% of volume, high complexity, currently 14 minutes per request. Hazmat: 7% of volume, must be human-reviewed regardless of automation by current GCC regulation.

This changes the framing. Hazmat is out of scope for any AI workflow. Standard freight is the high-volume target. Regulated cargo is a middle case where AI can assist but not replace human review.

Step 2: Define error tolerance. What's the cost of a misclassified shipment? They estimate: standard freight misclassification costs an average of $180 in operator time + customer inconvenience. Regulated cargo misclassification could mean a $5K-$50K fine plus reputational risk. They set the targets: 95%+ accuracy on standard freight, 99%+ on regulated cargo, with explicit human review on regulated regardless. Hazmat stays human-only.

Step 3: Estimate the cost of waiting six months. They calculate that the ops team currently has roughly 4 hours of bottleneck time per day during peak season. Six months of unaddressed bottleneck = approximately $80K in opportunity cost (lost capacity, overtime, occasional missed-deadline penalties). Not enormous, not negligible.

Step 4: Run each path through 24-month cost. They build a simple model:

PATH	YEAR 1 COST	YEAR 2 COST	ENGINEERING TIME	RISK OF FAILURE
Vendor A (specialized)	$83K (license + impl)	$58K (license + 20% increase)	4 weeks integration	Low — vendor knows the domain
Vendor B (general agent)	$64K	$36K	6 weeks	Medium — generic platform, custom logic needed
Vendor C (frontier API)	$28K (API + 5 weeks eng = $19K loaded)	$11K (API only, eng absorbed)	5 weeks	Medium-high — model migration risk, no operational tooling

PATH	YEAR 1 COST	YEAR 2 COST	ENGINEERING TIME	RISK OF FAILURE
Vendor D (open-source self-hosted)	$35K (hardware + 10 weeks eng = $24K loaded)	$4K (electricity, occasional retraining)	10 weeks	High — engineers have no ML experience, on-prem operations is real work
Wait	$80K (bottleneck cost)	Unknown — likely cheaper tooling	None	Low — maintains optionality

Step 5: Score against criteria. Using the scoring sheets in sections 05 and 06, the team scores each path. The build paths (C, D) score well on cost and control but poorly on risk and engineering capacity. The buy paths (A, B) score well on time-to-value and risk but worse on cost over 24 months. Wait scores well on cost but poorly on urgency.

The outcome

The team chose a hybrid: **Vendor C for the standard-freight path, with Vendor A's framework as the medium-term fallback.** They allocated 5 weeks of engineering to wire up GPT-5.5 against their data, with a clear eval suite (built first) covering 60 representative cases — 40 standard freight, 20 regulated. They committed to revisiting the decision at month 6 with Vendor A as the rip-and-replace option if the build-track proved fragile.

A year later: the build-track held up. They never invoked the Vendor A fallback. The eval suite caught one regression when GPT-5.5 was migrated to GPT-5.5-mini for cost reasons. Total Year-1 cost: $31K. Total Year-2 cost: $13K. The CEO reports the bottleneck is gone and the ops team has shifted to higher-value work.

What you should take from this

Three lessons:

01 **The first round of vendor evaluation reframed the problem.** They came in thinking "one workflow, four options." They left understanding "three workflows, different answers per workflow." Most teams skip this step.

02 **The hybrid was structurally cheaper than any pure path.** Buy was safest but expensive. Build was cheapest but risky. Hybrid (build with a buy fallback) gave them most of the cost benefit with most of the risk reduction.

03 **The eval suite was the unsung hero.** Without it, they wouldn't have caught the model migration regression. With it, they had measurable confidence to keep extending the build path.

The next two sections give you the scoring sheets they used.

05

The build-track scoring sheet

Use this when evaluating a build path. Score each criterion 1-5, where 5 is "strongly favors building" and 1 is "strongly disfavors building." Multiply by the weight to get a weighted score. Higher total = stronger case for build.

CRITERION	WEIGHT	WHAT 5 LOOKS LIKE	WHAT 1 LOOKS LIKE	YOUR SCORE
Workflow specificity	4	Can describe the workflow on one page; have real data	Can't define what the system needs to do	_
Engineering capacity	5	2+ engineers with 30%+ time available, including one with AI/ML experience	No engineers available, or all engineers fully booked	_
24-month cost vs buy	4	Build is meaningfully cheaper over 24 months (>30% savings)	Build is more expensive over 24 months	_
Strategic differentiation	5	This is core IP; we want full control of how it works	Commodity workflow; control doesn't matter	_
Vendor adequacy	3	No vendor adequately covers our case	Strong vendors exist that handle 80%+ of our case	_
Time pressure	3	Have 6+ months before the workflow becomes urgent	Need solution in <4 weeks	_
Data sovereignty	4	Data cannot leave our network; must be on-prem	No data sovereignty constraints	_
Eval/monitoring discipline	4	We have or can hire someone to own the eval suite	We have not done evaluation on any internal system before	_

CRITERION	WEIGHT	WHAT 5 LOOKS LIKE	WHAT 1 LOOKS LIKE	YOUR SCORE
Tolerance for technical debt	3	High — this is a strategic bet, we'll live with iterations	Low — the team is already over-stretched	_
Long-term roadmap	3	Plan to extend this with 5+ adjacent workflows over 24 months	This is a one-off; no plans to extend	_

Score interpretation:

- 130-180: Strong case for build. Most criteria favor it.
- 90-129: Mixed signal. Look at the lowest scores; if any are 1-2 on a high-weight criterion, hybrid is probably right.
- 50-89: Weak case for build. Buy or hybrid is probably better.
- Below 50: Don't build. The lowest-scoring criteria will sink the project.

A common pattern: teams score 90-110 on this sheet but score below 60 on engineering capacity. They build anyway because they want to. Six months later they're in trouble. **A single low score on engineering capacity or strategic differentiation should override the total.** These two criteria are veto-class.

06

The buy-track scoring sheet

Use this when evaluating a buy path. Same scoring approach: 1-5 per criterion, multiplied by weight. Higher total = stronger case for buy.

CRITERION	WEIGHT	WHAT 5 LOOKS LIKE	WHAT 1 LOOKS LIKE	YOUR SCORE
Vendor fit on our actual workflow	5	Vendor has 3+ customers like us, with real testimony, doing exactly this	No customers like us; we're the proof of concept	_
Time-to-value	4	We can be in production within 6 weeks	Implementation takes 6+ months	_
24-month cost vs build	4	Buy is meaningfully cheaper over 24 months	Buy is dramatically more expensive	_
Vendor health	5	Profitable, well-funded, growing, 4+ years in business	Recent startup, single round of funding, growing on assumptions	_
Lock-in cost	4	Can export our data and switch in <2 weeks if needed	Switching means starting over; data lock is high	_
Customization ceiling	3	Vendor allows extensive custom prompts, tool integrations, branding	Cookie-cutter — minimal configuration possible	_
Integration cost realism	3	Vendor's quote matches similar customers' actual experience	Vendor underestimates integration by 2-3x	_
Compliance + data residency	4	Vendor passes our compliance review with current contracts	Compliance/residency is unresolved or requires custom contract	_

CRITERION	WEIGHT	WHAT 5 LOOKS LIKE	WHAT 1 LOOKS LIKE	YOUR SCORE
Pricing trajectory	3	Multi-year pricing locked; minimal usage-based surprises	Usage-based with material year-over-year increases	_
Operational support	3	Real human support, response in <4 hours	Self-serve only or chatbot support	_

Score interpretation:

- 130-180: Strong case for this vendor. Most criteria check out.
- 90-129: Mixed signal. Re-evaluate with the lowest-scoring criteria — if any high-weight criterion is below 3, look at alternatives.
- 50-89: Weak case for this vendor. Look at others or consider build/hybrid.
- Below 50: Don't sign with this vendor.

Veto-class criteria (a single low score should override the total):

- **Vendor health** at 1-2 — you're betting on the vendor being there in 24 months. If they're not, your switching cost is enormous.
- **Vendor fit on our actual workflow** at 1-2 — this is the entire reason you'd buy. If the vendor doesn't actually solve your case, the rest doesn't matter.

07

The hybrid path most enterprises actually pick

If you've worked through the scoring sheets and the answer is mixed — high scores on some criteria for build, high scores on others for buy — congratulations. You've arrived at where most enterprises arrive in 2026: the hybrid.

The hybrid path has three forms in practice. Knowing which form you're picking matters more than picking *between* hybrid and a pure path.

Form 1: Buy the boring layer, build the differentiated layer

This is the dominant pattern in 2026. You buy:

- **Systems of record** — CRM, ERP, ticketing, document storage. AI features inside these are usually "good enough" and not worth replicating.
- **Compliance-heavy platforms** — anything with SOC 2 / GDPR / HIPAA / regional sovereignty already built in.
- **Frontier model access** — APIs from Anthropic, OpenAI, Google. Building your own foundation model is not a serious option for almost anyone.

You build:

- **Your workflow logic** — the agent that knows your domain, your data, your business rules.
- **Your evaluation harness** — built first, before the system.
- **Your integration layer** — the piece that connects the bought components to the built ones.
- **Your differentiation** — anything that, if a competitor copied it, would erode your edge.

Logically: you should buy what's commodity, build what's strategic. Most teams know this in principle and ignore it in practice — they build commodity infrastructure and buy off-the-shelf for their differentiated work. Reverse this.

Form 2: Build with a buy fallback (or vice versa)

You commit to one path, but with an explicit decision point and an alternative ready. Example: you start with the frontier API + custom integration (Vendor C in our worked example). At month 6, if the build path has run into specific blockers, you migrate to a vendor (Vendor A) that you've already evaluated.

This works only if you actually pre-evaluate the fallback. Most teams "have a fallback in mind" but haven't run the procurement evaluation. When the moment comes, they're starting from scratch.

Form 3: Buy now, plan to build later

This is the right answer when (a) the workflow is urgent, (b) the vendor lock-in is low, and (c) you want to use the vendor as a forcing function to learn what your actual requirements are.

You sign with a vendor for 12 months. You operate the workflow with the vendor's tooling. You monitor what does and doesn't work for your specific case. At month 9, you decide: extend the vendor contract or build the v2 internally with a year of operating learnings to inform it.

This is structurally a "use the vendor as your first prototype" play. It's underused because most teams don't think of vendor contracts as time-boxed prototypes. They sign and renew.

Picking the right hybrid form

IF YOU HAVE...	THE RIGHT HYBRID IS
Strong engineering team, no urgency	Buy commodity, build differentiated (Form 1)
Mixed engineering capacity, moderate urgency, real risk	Build with buy fallback (Form 2)
Limited engineering, high urgency, clarity on what to learn	Buy now, build later (Form 3)

The second mistake of build-vs-buy decisions, after "we never defined the workflow," is "we picked a hybrid without picking *which* hybrid." All three above are valid; mixing them is not.

08

When 'wait' is the right answer

Half of strategy is permission to not chase the loudest thing. The "wait" path is consistently the option most decision-makers don't seriously consider. Here's when it's correct.

Wait when:

The technology is moving faster than your decision cycle. If a meaningful vendor release happens every quarter and your procurement cycle is 9 months, anything you decide in Q1 is partially obsolete by Q3. For categories where this is true — AI agent platforms, retrieval systems, evaluation tools — waiting six months and re-deciding is often cheaper than committing now and locking in to something that will be embarrassing soon. The cost of waiting is calculable. The cost of locking in to a stale tool is harder to calculate but usually higher.

The cost of the workflow not being automated is bounded. If the workflow currently costs $80K of opportunity cost per year and you can wait six months for the tool quality to double, the math often favors waiting. Some workflows are not bounded — competitive pressure, regulatory pressure, security pressure. Most are.

The team that would own this is already at capacity. Adding an AI initiative to a team that's behind on its existing work is the most reliable way to ensure neither gets done well. Wait until either you've made room or you've hired in.

The vendor space is in active consolidation. AI agent platforms, vector databases, eval tools, and observability tools have all undergone visible consolidation in 2025-2026. Five vendors per category becomes two. Picking before consolidation increases the chance of betting on the loser.

You don't yet know what you'd do with it. This is the most honest reason to wait, and the one most rarely admitted in board meetings. *We don't know what we want from AI yet, and we're going to spend $300K finding out.* That's a research budget, not a procurement decision. If that's what's happening, name it that — and the framework changes entirely.

Don't wait when:

- A competitor is automating the same workflow and you can quantify share-of-market loss
- Regulation is creating an unavoidable deadline (e.g., EU AI Act compliance milestones)
- Your team is willing to learn from running into walls; the learning is worth more than the avoided wall
- The workflow is a small, bounded experiment whose downside is capped at <$25K

How to wait well

Waiting is not the same as not deciding. Waiting well looks like this:

01 **Document the decision to wait, with the trigger conditions for revisiting.** "We'll re-decide at the end of Q3 unless [vendor] launches enterprise tier or [model] crosses [benchmark threshold]."

02 **Use the wait time to refine the workflow definition.** Most "wait" periods are wasted because the team treats the decision as deferred rather than as research-in-progress.

03 **Conduct one or two free or paid pilots** with leading vendors during the wait. Pilots inform the eventual decision more than additional research does.

04 **Set a budget for the wait period.** Time is not free; allocate maybe 5-10% of what the build/buy would cost to vendor pilots, internal proofs of concept, or an external advisor.

Waiting is a serious option. Treat it seriously.

APPENDIX A

A 12-question vendor-interview script

Print this. Bring it to the call. Ask the questions in order. The vendor will hate it. That's good.

The vendor's job is to demo. Your job is to disrupt the demo with questions a good vendor can answer cleanly and a bad vendor can't. These twelve questions — designed for a 45-minute call — separate the two.

1. Walk me through three customers most like us. Names if you can; otherwise specifics on size, sector, and use case. What you're listening for: real customers, real workloads, comparable scale. If the vendor can't name comparable customers, you're the proof of concept.

2. What's the single hardest case your platform handles well, and what's the case it struggles with? A confident vendor knows where their product breaks. A defensive vendor will deflect. The latter is a red flag.

3. Show me a real customer's eval suite — anonymized. How do they measure that the system is working? If the vendor doesn't have customer eval suites to point to, the customers aren't measuring quality, which means you'll inherit the responsibility for measurement.

4. What's the integration cost in engineering hours, based on customers most like us? Vendors quote ranges. Push for specifics. "Customer X took 3 weeks; customer Y took 8 weeks; the difference was [specific factor]."

5. What does pricing look like in year two and year three? What price increases have you taken in the last two years? The vendor's history of price increases tells you their future. If they can't answer, they're freelancing.

6. Walk me through your data residency, sovereignty, and retention policies. Where exactly does my data live, and who can see it? Especially for GCC, EU, and regulated US customers. The right answer is specific. Vague answers ("it's secure") are non-answers.

7. What's your approach to model migration? When the underlying frontier model changes, what does my deployment look like? This is a real cost most vendors hide. Their answer reveals whether they've done this work for customers before.

8. Show me your status page and incident history. What did you do in the last major outage? A vendor with no public incident history hasn't been operational long enough to have one — or is hiding them.

9. How do I export my data, prompts, configurations, and customizations if I want to leave? The "exit clause" question. The right answer is concrete: "API endpoint, format X, takes Y minutes." Wrong answer is anything that sounds like it's never been done.

10. What's your security audit posture? Walk me through your most recent SOC 2 / ISO 27001 / equivalent. For enterprise customers, this is non-negotiable. The vendor should have current attestations and be willing to share them under NDA.

11. What does your customer success org look like? Who is the human I would actually email at month four when things are wrong? A named person, with a response-time SLA, beats a chatbot or community Slack.

12. If I gave you a $250K budget over two years and our exact workflow, what would your honest recommendation be — your platform, a competitor, build, or wait? The most diagnostic question of the twelve. A confident, customer-aligned vendor will sometimes recommend against themselves. That's the kind of vendor you want.

APPENDIX B

About Oasium AI

Oasium AI is an applied AI consultancy. Two PhDs (UC Berkeley and UCLA), based across San Francisco and Dubai, who use AI in their own daily work — and help organizations do the same. Three service lines: **Educate** (workshops and training), **Strategize** (AI feasibility and roadmaps, including the kind of build-vs-buy work this guide describes), and **Implement** (production AI workflows, agent systems, on-prem deployments).

We wrote this guide because we run versions of this conversation every week, and most of what's published online about build-vs-buy in AI is shaped by vendors trying to sell something. We don't sell a platform. We don't take partner commissions. We just do the work.

If you're sitting in front of an AI procurement decision and want a structured second opinion — or if any part of this framework is unclear in your specific situation — we offer free 30-minute discovery calls. No slides, no script, just a conversation.

Book a discovery call → (https://oasium.ai/contact#book-a-call)

Or write us at hello@oasium.ai.

If this guide was useful and you'd like more like it, the rest of our writing lives at oasium.ai/writing (https://oasium.ai/writing). We post when something's worth saying — usually a few times a quarter.

— Ziad and Jonah Co-founders, Oasium AI

Oasium AI

COLOPHON

About this guide

A practical framework for the AI procurement decision every operator faces in 2026: build, buy, hybrid, or wait. Two scoring sheets, a worked example, an honest case for waiting, and a printable vendor-interview script.

AUTHORS

Ziad Yassine and **Jonah Lipsitt**

Co-founders, Oasium AI

San Francisco + Dubai

CONTACT

oasium.ai

hello@oasium.ai

For consulting work, the workshop, or to point out something we got wrong.

EDITION

Guide № 01 · May 2026

The latest version is always at oasium.ai/guides.