BUYER'S CHECKLIST



AI Vendor Evaluation Checklist

40 questions to ask before you sign

Dr. Hitesh Sarda · AI advisor, Hyderabad

hitesh.in

A scoring worksheet for leadership teams comparing up to three AI vendors, written from the buyer's side.

How to use this checklist

Most vendor questionnaires ask about certifications and uptime, which matter, and your security team already has that form. This checklist covers what that form usually misses: whether the demo measured the tool or the person driving it, whether you can check and cap the bill, what happens to your keys on the day the vendor is breached, what the vendor writes into your agents, and whether you can leave with what the system learned from your people.

Each question here comes from an argument I have made in public on my blog, and the posts are listed at the end so you can check the reasoning.

The routine

1. **Send the questions before the demo.** Paste them into an email as written. In a demo the vendor sets the pace, and written answers are easier to compare across vendors.
2. **Ask for evidence with each answer:** a document, a log, a contract clause, or a screen you can look at yourself.
3. **Score each answer 0 to 3** in the columns marked A, B and C, one column per vendor.
4. **Compare section by section** on the scoring sheet, because a vendor that wins the demo and loses on exit is a different purchase from one that does the reverse.
5. **Take every 0 into the contract negotiation.** The clauses to ask for are listed in the last part of this document.

How to score

Score	What you got
0	No answer, a refusal, or "we will get back to you" that never came back
1	A vague or marketing answer. Treat a vague answer as a no.
2	A clear, specific answer with no evidence yet
3	A clear answer with evidence you checked yourself

What a refusal means. A vendor may have a good reason not to share a system prompt or an incident report with a prospect, so ask whether they will share it under a non-disclosure agreement, or after signing with a right to walk away. If the answer is still no, score it 0 and assume the thing you asked about works against you. How a vendor handles a hard question before you are a customer is the best preview you will get of how they behave the day something breaks in production.

Fill this in first

The work we want this tool to do _____

The person who owns the outcome _____

Vendor A _____

Vendor B _____

Vendor C _____

Your own homework before you send anything

> Four of the risks in this checklist sit on your side of the table, and no vendor answer fixes them. The AI shortlist is the easiest to underrate: when Trellner Research looked at 7,534 citations behind Perplexity's software recommendations, nearly 60 percent pointed to domains ranked below the top 100,000 sites.

- [ ] We have a test set built from our own closed work (tickets, cases, merged changes, resolved claims) where we already know the right answer.
- [ ] We have run the business case at two to three times today's price and know which uses still make sense.
- [ ] If an AI assistant or a "best software" page gave us this shortlist, we asked for its sources and opened at least two of them.
- [ ] We know who in our company would rotate every key stored with this vendor, and roughly how long it would take.

The 40 questions

> In the questions, "you" is the vendor and "we" is your company, and the middle column is what a good answer sounds like. Write a score from 0 to 3 for each vendor. The scale is a starting point, not a standard, so if your company already has a scoring method, use that instead.

1. Fit and proof

A vendor demo is a controlled setting where the vendor picks the task, the operator and the framing, and each of those choices flatters the tool. When OpenAI shipped Astra for Law, it reported that the tool answered 54% of a 200-question legal research benchmark correctly, up from 38.7% with plain web search, and OpenAI published the figure itself, so you know whose test it was. Ask every vendor for that much clarity. These questions take the operator out of the result.

No.	Question	A good answer sounds like	A	B	C
1	Will you run the pilot on our own closed work, where we already know the right answer, and not on your demo data?	"Send us the inputs and keep the answers. We will run it and you grade it."			
2	Who operated the tool in the demo? Can one of our competent non-experts repeat it without your engineer in the room?	"Put an ordinary user on it. We will watch and keep our hands off the keyboard."			
3	For each finished task, how much of our people's time still goes into prompting, correcting and checking?	A measured figure from a comparable customer, and a way to log who solved each task during the pilot. "There is a human in the loop" is not an answer until they say how much human.			
4	Will you agree in writing, before the pilot, the baseline we compare against and the result that would justify going to production?	A one-page pilot plan with a baseline, a pass mark and an end date, signed by both sides.			
5	On which tasks in our workflow does your product do badly, and how often is it wrong on work like ours?	Named failure types and an error rate, with a plain statement of whose test produced the number.			
6	For every benchmark, review or ranking you cite: who ran it, who paid for it, and can we open the source?	Links you can open, to sources you have heard of, with sponsorship disclosed. Customer references you choose, not ones they choose.			
7	What was removed or filtered from the model's training data, as distinct from what the model refuses to do?	A pointer to the model card and a straight answer. If they cannot say, price that uncertainty in and test the capability on your own tasks.			

2. Pricing and the bill

A metered bill is computed by the vendor, on the vendor's systems, with the vendor's counters, and the party doing the counting profits from the count. In July 2026 some AWS customers opened their billing console to estimates in the trillions of dollars, including nearly $2.5 billion for an account that had spent nineteen cents in June, which AWS traced to a unit-pricing defect that never touched actual charges. Nobody pays a bill like that, so it gets caught. The bill I would worry about is the one that is wrong by five percent in the vendor's favor and looks right, because that one gets paid.

No.	Question	A good answer sounds like	A	B	C
8	What exactly is the unit we are billed on, and what would one finished task of ours cost?	The unit in plain words (seat, token, call, agent action) and a worked example on your volumes, stated as cost per finished task.			
9	Can we export raw usage records and reconcile them against our own request logs?	A per-request export or API, available every day, and a stated process for when their count and yours disagree by more than noise.			

No.	Question	A good answer sounds like	A	B	C
10	Is there a hard spending cap, per key and per project, that stops usage? Or only alerts?	A cap you set yourself that stops the meter, plus a clear description of what your users see when it is hit. Alerts notify. They do not stop anything, and they arrive late.			
11	Which discounts in the headline price depend on how we build, such as caching or committed volume?	They look at your actual prompts and tell you which discounts you would reach and which you would miss.			
12	Is the price held for the contract term, and how much written notice do we get of a price or packaging change?	A price hold, a notice period in the contract, and a right to leave without penalty if the price goes up.			
13	Whose models and compute do you run on, and what happens to our price if their price to you rises two to three times?	They name the suppliers, say whether costs pass through to you, and accept a ceiling on the pass-through.			

3. Data and security

If the vendor can read your secret, an attacker who gets into the vendor can too, and an AI provider key is the easiest credential to cash in, because whoever holds it can spend your money at once. Once a vendor has an incident, your exposure depends on how fast you can rotate and how much a key can do in the meantime. On October 30, Northflank's status page told customers to "Assume all environment variables have been decrypted," and Northflank only switched on audit logs for all customers in the middle of that incident.

No.	Question	A good answer sounds like	A	B	C
14	Where are our secrets and API keys decrypted, and which of your systems and staff can reach that place?	A named system and a short list of roles. "Encrypted at rest" tells you about a stolen disk and nothing about an attacker inside the platform.			
15	Can we avoid storing static keys with you, by using short-lived credentials, workload identity or our own cloud account?	Yes, with documentation. For the few secrets that matter most, a way for you to hold the encryption key yourself.			
16	If a key of ours that you hold is stolen, what is the most an attacker can spend or reach before someone notices?	Separate keys per service and environment, hard limits at the provider, and usage anomalies routed to a person on call.			
17	How do we rotate every secret we store with you, and how long does a full rotation take?	A script or an API, a list of every secret you hold with them, and a time they have measured. More than a day is a risk you now own.			
18	When you have a security incident, how fast do you tell us, through which channel, and what do we get to see?	A time commitment in the contract, a named channel, and audit logs you can read today. Logs switched on during the incident are too late.			

No.	Question	A good answer sounds like	A	B	C
19	Can we see your incident record?	A list of what happened, what they disclosed and what they changed. A vendor that has never disclosed anything has either been lucky or is not looking.			
20	Which third-party components sit between us and the model (gateways, proxies, open-source packages), and how do you control their updates?	They name each one, pin versions, never pull a floating "latest" tag, and let the components that hold credentials lag new releases by days, with a human-approved fast lane for security fixes.			

4. Reliability and operations

A vendor measures how many of its customers use a feature, while you should be counting how many of your processes rely on it. Something rare across their base can be the one thing your month-end depends on, and the vendor can be entirely reasonable in removing it. Capacity has a similar weak point, because the GPU landlords behind many AI services are partly financed by their chip supplier: Nvidia agreed to buy up to $6.3 billion of unsold CoreWeave capacity through 2032. If that financing slows, capacity can get tight, and question 24 asks who gets served first when it does.

No.	Question	A good answer sounds like	A	B	C
21	How do you decide to retire a feature, an API or a model version, and how much notice do we get?	A written policy with a minimum notice period and a named contact on your side who gets told. Treat any extension they offer as a deadline, not a fix.			
22	When the underlying model changes, can we pin the current version and test the new one before the switch?	Version pinning, advance notice, and a window long enough to re-run your own test set.			
23	How long does a security fix take to reach our workloads after it is available upstream?	A figure in days, with a recent example. Anything beyond a few days needs a compensating control from them or from you.			
24	What happens to our service when capacity is short or we hit a rate limit? Who gets served first?	A plain description of how the service degrades, how priority is decided, and what you can reserve.			
25	When we send a technical question today, before signing, who answers and how fast?	A real engineer, within hours. Test it: send one hard question this week and time the reply.			

5. Agents and autonomy

An agent acts on its instructions without anyone watching it happen, which makes the agent's configuration valuable to every vendor who would like to be in it. The files that steer an agent often sit outside your code repository and change when a vendor ships an update, with no review and no diff. On October 25 a developer reported that an automatic Postman desktop update had installed plugins into their Codex and Claude Code setups, with a session-start hook telling the agent to load a Postman skill "before responding to any non-trivial API engineering task." Postman says the bundling is disclosed and is reviewing the hook behavior.

No.	Question	A good answer sounds like	A	B	C
26	Does your software write anything outside its own directory, including into our AI agents' settings, hooks, plugins or instruction files? Does it ask first?	A complete list of what it writes and where. Integrations are opt-in and nothing approves itself.			
27	Can we read every instruction your product gives the model on our behalf, and see changes before they ship?	The full text of system prompts and tool descriptions, a change log, and a named owner. Nothing in it that they would mind having quoted back to them.			
28	Where does agent-generated code run? Does each run get its own kernel, or does it share the host's?	They name the isolation layer. With a shared kernel, their patch process is your whole security model, so ask question 23 again.			
29	Are network rules and credentials enforced outside the sandbox, where the agent cannot change them?	Network filtering on the host, and short-lived tokens issued per task through a broker. No long-lived keys in environment variables.			
30	Which actions can the agent take without a person approving them, who sets that, and can we limit spend and reach per run?	You set the review rate and the escalation rules. Each run has its own spend limit and network scope, which you can test by trying to exceed both. There is a log of what the agent did.			

6. Lock-in and exit

The loop that runs an agent is getting cheaper and more alike across vendors. Shrivu Shankar argues that every SaaS business will become a harness around a model, and that the harness becomes something a company no more outsources than its product team. I think he has the diagnosis right and the conclusion only half right. What grows in value is what the system learns from your people: every time someone overrides a draft or rejects a change, your company has recorded a little of what "good" means here. If that record lives only inside the vendor, you have built a dependency.

No.	Question	A good answer sounds like	A	B	C
31	Can we change the underlying model without rewriting our workflows? Have you done that for a customer?	Yes, with an example. Better still, they route different tasks to different models already.			
32	Can we export every human correction, rejection and override of the AI's output, with the reason, in a plain format?	A scheduled export into your own storage during the contract, and not only on the way out.			
33	Can our data, prompts, rules and test sets move to another vendor within a quarter without a rewrite?	A documented export in open formats, and a customer who has done it. Ask to test the export during the pilot.			
34	Is the product open source or source-available? What rights do we have to fork it or host it ourselves?	A clear license. You may never fork it, but the right to fork limits their pricing power and gives you an exit a closed product cannot.			

No.	Question	A good answer sounds like	A	B	C
35	When we leave, how long do we keep access, in what form do we get our data, and what does that help cost?	An exit period, formats and fees written into the contract before you sign.			

7. The vendor's business

A product works for whoever pays for it, and model providers have a standing reason to move up into applications, because inference is a low-margin commodity and applications hold the margin. OpenAI's Astra for Law launch moved it into legal AI and, in the same announcement, named Harvey and Legora as partners who would build on top of it. That is a snapshot of a relationship, so read what a platform ships and give less weight to what it says.

No.	Question	A good answer sounds like	A	B	C
36	Which part of your product would still be yours if your model provider shipped the same feature tomorrow?	Something specific: workflow built into how customers operate, data nobody else has, or accountability they take for the result. A tidy interface over someone else's model has no answer here.			
37	Do you sell, or plan to sell, a product that competes with what we are building on your platform?	A direct answer about the roadmap, and contract terms on how they may use your data and usage patterns. A spoken promise not to compete is a snapshot, and snapshots move.			
38	Who pays you besides us? Does any advertiser, sponsor or partner pay to influence what your product recommends or which tools it reaches for?	Users pay. Any paid placement is labelled and you can switch it off.			
39	Does your price cover your costs today, or is part of it promotional or subsidized?	They can explain in plain words how one unit of your usage makes them money, without needing to open their books.			
40	When the output is wrong and it costs us, what do you take responsibility for in the contract?	A plain statement of what they stand behind. Most platforms do not want your liability, so the checking stays with you. Staff and budget it.			

Scoring sheet

> Add up each vendor's scores by section, write the totals below, and then count the zeros. Read the rows across before you look at the bottom line.

Section	Max	Vendor A	Vendor B	Vendor C
1. Fit and proof (questions 1 to 7)	21			

Section	Max	Vendor A	Vendor B	Vendor C
2. Pricing and the bill (8 to 13)	18			
3. Data and security (14 to 20)	21			
4. Reliability and operations (21 to 25)	15			
5. Agents and autonomy (26 to 30)	15			
6. Lock-in and exit (31 to 35)	15			
7. The vendor's business (36 to 40)	15			
Total	120			
Number of questions scored 0				
Red flags seen (next page)	10			

How to read it

- **A low score in one section is a finding, and a high total does not cancel it.** A strong demo does not make up for a bill you cannot cap or an exit that does not exist.
- **Every 0 needs a decision.** Either it becomes a contract clause, or you build the control yourself, or someone senior accepts the risk by name.
- **Sections 2, 3 and 6 are where regret comes from.** Problems with fit tend to show up in the pilot, while problems with the bill, a breach or the exit show up after you already depend on the product.
- **If no vendor scores well on agents, narrow the first deployment.** Start with every action reviewed by a person and lower the rate task by task as the evidence earns it.

The decision

Vendor we are going ahead with _____

Conditions to meet before we sign _____

Risks we are accepting, and who accepted them _____

Controls we will build ourselves _____

Date we re-run this comparison _____

Put a date on the re-run. Prices and models move fast enough that a comparison done six months ago may well have flipped. The vendor you standardized on may now be both weaker and more expensive than something newly available. Re-running is cheap if you kept your test set.

Ten red flags

> Tick any you have seen, for any vendor. One of these is a reason to slow down, and three from the same vendor is a pattern to raise before anyone signs.

- [ ] **The demo cannot leave the vendor's hands.** It works with their engineer, on their data, and nobody will let your ordinary user try it on your work.
- [ ] **The praise cannot be traced.** The rankings and reviews they cite sit on sites you have never heard of, or the shortlist came from an AI answer whose sources nobody opened.
- [ ] **You can neither check the meter nor stop it.** There is no usage export to reconcile and no hard cap, only a dashboard and alerts.
- [ ] **The price has nothing holding it up.** It is a promotional rate with no hold, no notice period, and no answer on what happens when their own costs rise.
- [ ] **"Encrypted at rest" is the whole security answer.** Nobody can tell you where your keys are decrypted or who can reach that system.
- [ ] **The incident record is spotless.** No disclosures, ever, and no description of how they would tell you about one.
- [ ] **The software writes where it was not invited.** It installs plugins, hooks or instructions into other tools and agents, and the answer about asking first is vague.
- [ ] **You cannot read what the model is told.** The system prompt and tool instructions are "proprietary", and they change without notice to you.
- [ ] **The exit exists only on paper.** Export is "on request", or only at termination, and the corrections your people made are not part of it.
- [ ] **They are slow before the signature.** A technical question takes weeks and the contract needs three rounds of legal to become reasonable. There is little reason to expect either to improve once you have paid.

None of these means the vendor is dishonest. Most come from a company that has not thought about your side of the arrangement yet, and you are entitled to wait until it has.

What to put in the contract

> Hand this page to whoever negotiates. A good answer in a sales call is a preference, and preferences move with the vendor's balance sheet, so each line below turns one of those answers into an obligation. Tick the ones you have in writing.

The bill

- [ ] **Audit rights on metering.** Raw usage records available to you on a regular schedule, the right to reconcile them against your own logs, and a process with a deadline for resolving differences.

- [ ] **A hard cap.** Spending limits you set, per key and per project, that stop usage. State what happens to the service when the cap is reached.
- [ ] **Price-change notice.** The price held for the term, a written notice period for any change to price or packaging, and the right to leave without penalty if it rises.
- [ ] **A ceiling on pass-through.** If the vendor's model or compute costs go up, a limit on how much reaches you.

Security

- [ ] **Breach notification.** A fixed time limit for telling you about any incident that may touch your data or secrets, through a named channel, to a named person on your side.
- [ ] **What you get to see.** Access to audit logs for your account from day one, and the vendor's findings after an incident.
- [ ] **Help with rotation.** The vendor supports a full rotation of your secrets on request, and you have rehearsed it once.
- [ ] **No uninvited writes.** The software does not modify other tools, agents or configuration outside its own directory without your explicit opt-in.

Change

- [ ] **Model and version changes.** Advance notice, the right to pin the current version for a stated period, and time to test before a switch.
- [ ] **Deprecation notice.** A minimum notice period before any feature or API you use is retired.
- [ ] **Instruction changes.** Notice of changes to the system prompts and tool instructions that act on your behalf, and the right to read them.

Exit

- [ ] **Export of data and learned corrections.** Your data, prompts, rules, test sets and every human correction with its reason, in a plain format, available throughout the term.
- [ ] **Exit help.** A period of continued access after termination, the formats you will receive, and the fees, all fixed now.
- [ ] **Use of your data.** What the vendor may and may not do with your data and usage patterns, including if it launches a product in your category.

A control you do not own is not a control. Where a vendor will not agree a clause, build the crude version yourself: your own request counts, a budget alarm wired to something that actually switches usage off (and expect that alarm to arrive late), and your own log of corrections.

Further reading

- Your best people are the worst judges of your AI pilot, on why a demo measures the operator and how to run a pilot that cannot flatter you.

- The scariest cloud bill is the one that looks right, on metering you cannot audit or cap, and the reconcile-and-kill-switch habit.
- Someone is subsidizing your AI compute, on why to run your numbers at two to three times today's price.
- Assume your platform vendor will lose your secrets, on what "encrypted at rest" leaves out and why rotation time is the control.
- Your coding agent now takes instructions from your software vendors, on software that writes into your agents' configuration.
- Rent the AI harness. Own what it learns., on capturing corrections in a format you control and testing the exit.
- When your AI vendor ships your product, on assuming your supplier will enter your category.
- Someone built 215,000 pages to game your AI, on manufactured review pages and why you should open the sources behind an AI shortlist.

About the author

Dr. Hitesh Sarda is an AI advisor based in Hyderabad and the founder of 256 Technologies, an applied AI lab. He has spent more than 25 years building production software and now helps leadership teams decide what AI to build, buy or skip. He is the author of *Implementing AI Agents in Your Company* and writes at hitesh.in.

> **Want a second pair of eyes on what you filled in?**
>
> Bring this document to an advisory hour. We go through your answers, and you leave with the simplest next step, or an honest view that AI is not the answer yet.
>
> hitesh.in/work-with-me · hitesh@256tec.com

You are welcome to use and adapt this document inside your own organisation. Please do not republish or resell it.