Five Questions Before You Ship an AI Feature

How to decide which AI features are worth building, and how to keep them working after launch.

Your customers are asking for AI, and the tools to build it have never been easier to use. That's exactly when this matters. Most AI features don't fail dramatically. They fail quietly: a chatbot that's confidently wrong, an agent that drifts into decisions it was never approved to make, a bill that jumps tenfold, or a feature nobody uses that you still have to maintain.

The one thing to remember: generative AI is probabilistic. Ask it the same thing twice and you can get two different answers. It's only useful when it's wrapped in deterministic code: prompts you control, grounding data, output checks, guardrails, and monitoring. Every question below follows from that.

1 Value

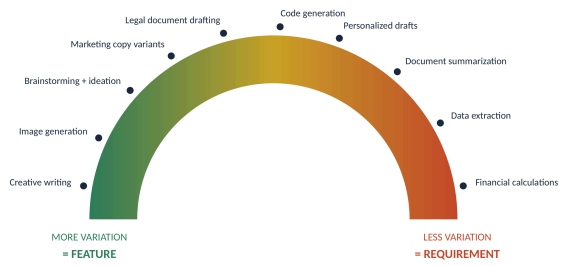
What measurable outcome does this produce, and what will it really cost?

- **Business value.** Pick one: revenue, retention, cost savings, or compliance. Quantify it against today's baseline. "It will save a lot of time" is not a metric. "Our support team spends 43 minutes per case" is.
- **Cost of doing it.** Not just the build. Ongoing maintenance, the human review layer, and token costs, with a plan to watch them as usage grows. AI features look a lot cheaper in a proof of concept than in production.
- **Cost of not doing it.** The number most teams skip. Compare against the status quo, not against zero. The status quo is almost never free.

2 Fit

Is this the right problem for a probabilistic system?

Variation is a feature for creative work, and a defect for business automation. Find where your use case sits on the dial, then build the guardrails that keep it there.



Easy fit: variation is a benefit and stakes are low. An easy pass.

Hard fit: doable with careful prompts, deterministic checks, drift monitoring, and human review.

No fit: "approximately right" isn't acceptable. Use deterministic code, or wrap AI tightly inside it.

Watch for two failure modes: **hallucination** (it states something untrue) and **interpretation failure** (it correctly answers a different question than the one you meant). The second is harder to catch.

3 Risk

What's the blast radius if it fails or is misused?

- **Data.** Classify what goes in (public, internal, or sensitive data like PII, PHI, or payment data). Decide what comes out, how it's protected, which regulations apply, and what your audit trail captures.
- **Ethics.** AI doesn't make sensitive determinations on its own. If a customer asks "who decided this?", the answer needs to be a person. And a human who approves every output in three seconds isn't oversight.
- **Vendors.** Your feature depends on someone else's pricing, terms, and model changes. Build for multiple providers, avoid vendor-specific lock-in, and read the contract terms.

4 Operability

Can you actually run it, monitor it, and pay for it at scale?

AI features decay differently than traditional software. Without monitoring, they can be wrong for a long time before anyone notices. Plan for eight things:

- **Quality drift:** an evaluation set run on a schedule, with alerts
- **Model version changes:** pin versions and test every update like a deployment
- **Cost drift:** cost per unit, watched
- **Prompt injection and abuse:** logged and reviewed
- **Human-in-the-loop health:** are reviewers actually reviewing?
- **Ownership:** a named owner, with a backup
- **User feedback:** signals that it's helping, or not
- **Team AI literacy:** people who understand how it works

If you don't have a plan for all eight, you don't have an operating model. You have hope.

5 Trust

How will you know it's still doing what you approved six months from now?

- **A review date on the calendar.** Quarterly, on every major model change, or whatever the risk justifies. When it hits, re-ask Value, Fit, Risk, and Operability against current reality.
- **Tuning between reviews.** Refine prompts as you learn what users actually ask, grow the evaluation set as new edge cases show up, and only roll forward to new models when they pass, on your schedule, not the vendor's.

Make the decision explicit

Green: ship. Internal, low risk, no sensitive data.

Yellow: ship with conditions. Customer-facing, sensitive data, or a new kind of use case. Spell out the conditions.

Red: not yet. Ethical concerns, sensitive data without the right controls, or no named owner.

Treat everything as yellow and nothing ships. Treat everything as green and you'll have incidents.

For any AI feature already in production

- [ ] A named owner, with a backup
- [ ] Scheduled evaluation runs with pass criteria
- [ ] A protocol for model version changes
- [ ] A cost-per-unit dashboard someone watches
- [ ] A prompt-injection and abuse log, reviewed
- [ ] A human-in-the-loop health metric
- [ ] An incident playbook for this feature

Can't check all seven? That's your next sprint.

Your Monday-morning action

1. Pick your top three candidate AI features.
2. Walk each one through Value, Fit, Risk, Operability, and Trust.
3. The ones that pass all five are your first wins. The rest need a redesign, or shouldn't ship.
4. Ask one more question: who owns AI governance at your company? If the answer is "no one," start there.

> Without governance, you're just building the next legacy system, faster.

Need help building AI into your platform?

PALADEM is the development partner for growing SaaS platforms that need an expert team and the executive leadership to run it. We put every AI feature through these five questions, then build and run it with the guardrails, monitoring, and oversight it needs.

Scott Bennett, PALADEM's founder and creator of the Software Stewardship Framework™, presented this approach at Adobe ColdFusion Summit 2026.

paladem.com/contact-us

© 2026 PALADEM LLC. The Software Stewardship Framework™ is a common-law trademark of PALADEM LLC.