How to adopt AI in software development: check the work, not the licences

By Andy Slack | November 11, 2026 | AI Development

A founder's guide to AI adoption: what to measure, what slowed our own delivery, and where quality, access and budget need a human decision.

How do you adopt AI in software development without just buying licences? Change how work is accepted, checked and released, then look at the evidence. For a nontechnical founder, the useful question is not which model the team uses. It is whether the work reaches customers, what holds it up, and who takes responsibility when something goes wrong.

Our spec-driven development article (/articles/spec-driven-development-ai-coding-agents) explains the foundations: specs, tests and review. This page starts where that process meets real delivery. It uses our own site's results and problems, confirmed by Andy Slack for this article on 11 November 2026. These are not client results or a promise of what your team will achieve.

Ask for evidence of adoption

A licence tells you that a tool is available. It does not tell you what changed. Ask your technical lead to walk you through a recent change: when it went live, what counted as done, and where it was reviewed before release. If the answer is still just a developer saying it is finished, the new tools have not given you a better way to judge the work.

Use that conversation to agree what you will look at next time. A tool demonstration is not the same as evidence that a product change reached its acceptance criteria. You should be able to ask about the result without reading the code yourself.

What we measure, including the gap

Our delivery record for the week to 9 November 2026 gives a mixed answer, not a success headline. We track cycle time, review rounds and test coverage of changed lines. Here is what that record actually contained, confirmed by Andy Slack:

- Cycle time: the median from an issue being picked up to being closed was 8.3 hours over 20 issues, against a target of at most 24 hours.
- Review rounds: the mean was 4.7 across the 19 issues with a merged pull request, against our target of at most 3. That was outside target.
- Coverage of changed lines: no usable reading. The covered and total line counts were not recorded in the verification notes for those merged changes.

A passing test run and a coverage reading answer different questions. We could report the delivery time and the review work, but we could not report how much of the changed code the tests covered. We would rather leave that gap visible than turn missing evidence into a reassuring number.

What slowed us down

Checking the work became the slow part. On our own site, end-to-end tests began running out of time on a busy machine, so checks failed even when the code was right. We gave the tests longer waits and made the report distinguish a time-out from a bug. The first version of that change left the main branch failing its checks; our main-branch check caught it and opened a fix. This is our own delivery experience, confirmed by Andy Slack, not a claim that the repair eliminated all failures.

That is why review rounds belong beside delivery time. A short delivery time does not tell you how much correction happened along the way. Ask what caused the repeats, and what changed because of them, before treating faster coding as a better delivery process.

Who handles a failed check?

There is an important boundary in our process beyond the order of tests and code. If a test itself is wrong, the code agent sends it back to the test stage rather than editing it to make its work pass. If a specification needs a decision that belongs to a person, work stops and asks. Otherwise it can proceed to tests. Both boundaries are part of the practice Andy Slack confirmed for this page.

On our own site, a person also approves the merge before anything reaches the live branch. The point is not to ask a founder to inspect every line. It is to make clear where the agents can correct their work and where their authority ends.

Keep quality, access and cost in the same conversation

Do not assess adoption only by output. Ask who can change the checks, what the agents can access, and who approves more spending. Our quality process is described in the spec article above. Our published offer (/) sets the other boundaries: agreed workflows and access controls, accounts in your name, and a customised budget with controls. Any budget increase needs your approval.

AI usage is paid from your own accounts and is outside our service fees. Output follows the AI budget you agree, not a fixed number of features. Those are spending and accountability boundaries, not a guarantee that AI-generated work is secure or error-free.

For an existing product, start with an assessment, not a promise to replace everything. We assess an existing system before committing to take it over; the first milestone is agreed during onboarding. The named fractional CTO remains accountable for technical decisions and delivery oversight, while you keep final product priorities and commercial decisions. The full responsibilities and onboarding limits are on the homepage.

Book a founder fit call (/) if you want to discuss that assessment for your product.