Technical due diligence for software: what we check before we take a system over

By Andy Slack | November 15, 2026 | AI Development

What a technical due diligence review of software covers, how we run one before taking a system over, and what you receive.

Search for technical due diligence in the UK and most of what comes back is about buildings. The Royal Institution of Chartered Surveyors publishes a professional standard for the technical due diligence of commercial property, where the term means a survey of a building before it changes hands (RICS, Technical due diligence of commercial property, 1st edition (https://www.rics.org/profession-standards/rics-standards-and-guidance/sector-standards/real-estate-standards/technical-due-diligence-of-commercial-property)). This page is the other meaning: a review of a software product. It is written for a nontechnical founder who already depends on a system and wants to know what such a review covers before anyone commits to it.

We should be plain about what we sell. We don't sell a standalone due diligence report to buyers or investors. The technical assessment of the product, its code, its infrastructure and how the team works is part of onboarding on our fractional CTO service. We run it on an existing system before we commit to take it over (Andy Slack, confirmed 15 November 2026). If you are buying a company and want a report written for the deal, this is not that service.

Access comes before the assessment

The order is fixed, and it starts with access. Before anything else we check that the repository and the hosting are held in the company's name, not one person's. Access comes first so the company can see what it has (Andy Slack, confirmed 15 November 2026). If an agency or a freelancer still holds the keys, the questions to ask them are in who owns my app code (/articles/who-owns-my-app-code). If the person who built the product has already gone, the first weeks are covered in when your technical cofounder or only developer has just left (/articles/technical-cofounder-left). This page starts once the company can see the system.

Then we assess four things: the product, its code, its infrastructure and how the team works. From that we write a prioritised roadmap and agree a first milestone, which can be taking over the existing system (Andy Slack, confirmed 15 November 2026). The assessment is what the commitment waits on. Nothing is agreed before it is done.

How we judge the way the team works

We judge another team's delivery practice against how we work ourselves (Andy Slack, confirmed 15 November 2026). Four questions:

- Does each change start as a written issue with acceptance criteria?
- Do tests run on every change?
- Is every change reviewed before it is released?

- Are the decisions and their reasons kept in the repository, where the founder can read them?

How a spec is written, and how agents build and review from it, are already explained in spec-driven development with AI coding agents (/articles/spec-driven-development-ai-coding-agents) and the agentic development loop (/articles/agentic-loop-development). We point there rather than repeat them.

Whether the code was written with AI does not change what we look for. We would be more concerned if a development team were not using AI tooling to improve its performance (Andy Slack, confirmed 15 November 2026).

What to look at before you depend on the system

Three you can check without reading code, before you commit to depend on a system (Andy Slack, confirmed 15 November 2026):

- Whose company account holds the repository and the hosting?
- Are the decisions and their reasons written down where you can read them, or are they in one person's head?
- Do changes reach users in small, frequent releases, or in large, rare batches?

If a technical lead is still in place, the conversation is a different one. A fractional CTO's view when the playbook predates AI-native engineering (/articles/my-ctos-playbook-predates-ai-native-engineering) covers how to spot a way of working that is holding delivery back and how to raise it. The three above are the ones that matter before you commit.

A product that only one person understands

In an assessment, a product that only one person understands is a risk in itself, so we look for whether the decisions are written down where the company can read them (Andy Slack, confirmed 15 November 2026). We are often called when that person has already gone, or when someone else holds the codebase. When your technical cofounder or only developer has just left (/articles/technical-cofounder-left) covers those first weeks, and this page does not repeat them. Telling a people problem from a process problem is a separate judgement, and is your development team underperforming (/articles/underperforming-development-team) covers how we make it.

What you receive, and when we decline

It isn't a separate report. Onboarding gives you the assessment, your environment and AI engineering team set up, a prioritised roadmap and an agreed first milestone. Onboarding targets two to four weeks from when we receive the access and inputs we need from you. It is covered by the published fractional CTO price on our homepage (/), not priced on its own (Andy Slack, confirmed 15 November 2026).

If the assessment shows the fractional CTO service isn't a fit, we say so and decline, or we offer a tailored scope of the same service. We don't sell a different service (Andy Slack, confirmed 15 November 2026).

No vendor lock-in: every solution comes with full training to self-administer. Your code, your accounts and the written record stay in your name, and the homepage states those ownership terms.

If you want to talk through a system you already depend on, Book a founder fit call (/#lead). The price and the onboarding terms are on the homepage (/), and the other pieces are on the articles index (/articles).