ANTIDEFAULT

An over-engineered recruiting platform its own team couldn’t launch, cut down until it shipped

Case study — CPI

Florian Ionescu

August 17, 2026

Client	CPI – a recruiting firm productizing its internal hiring tool
Engaged via	Toptal, through the consultant who owned the client relationship
Industry	Recruiting software (finance placements)
Stack	React, TypeScript, Node.js, GraphQL, PostgreSQL, Docker
My role	Sole developer, from the rebuild to owning it in production
Engagement	Since September 2025, ongoing (now production maintenance)

A recruiting firm had built an applicant-tracking system to run its own hiring, liked it enough to sell, and set about turning it into CPI, a multi-tenant product. A consultant who'd inherited the codebase brought me in through Toptal to finish it. The people who built it knew what they were doing. They'd just built past the point where it could ship.

1 Well built, and unable to keep itself running

The code itself was fine. The trouble was everything around it: a stack of separate pieces, each a reasonable choice on its own, that kept knocking each other over.

The frontend and backend lived in separate repositories, shipped separately on CircleCI, so releasing to production meant the team lead deploying both halves by hand, in the right order, every time. Search ran on an OpenSearch cluster kept in step with Postgres by a sync daemon, and the daemon kept choking on write-ahead-log issues and taking the database down with it. A third piece had run off onto its own lambda, with a deploy all its own. Then there was the safety net that wasn't: a test suite that was broken and covered almost nothing.

2 Deleting the parts that kept breaking

I merged the two repositories into one, switched everything to GitHub Actions across three environments, and got the whole stack down to a single `docker compose`

up. Then I started removing things.

OpenSearch and its sync daemon were the first to go. I'd spent weeks nursing that setup along, and there's a commit of mine from back then that just reads: "don't have opensearch break on me just because my plate is 95% full." So I stopped propping it up and took it out.

But search is what recruiters do all day, so it had to happen without anyone noticing. Search moved onto plain Postgres, trigram indexes and all. I did it a piece at a time, behind a soft switch, diffing the new results against the old before I trusted them. It took weeks. There was never a cutover to sweat.

The lambda service and the Redis cache followed them out. I deleted about 8,500 lines of infrastructure in all, and the system had far fewer places left to break. That was about 40% of the job. The other 60% was making it safe to run: locking the API down to proper auth, rebuilding the broken test suite, and wiring in real telemetry.

3 Two days down to six hours

The migration I inherited took almost two days to run. The code underneath was sound, so the wins were unglamorous: I added a document cache and batched the queries that had been firing one at a time. Six hours. That's the difference between a migration you schedule a weekend around and one you run inside a working day.

4 Still the only name in the commit log

The platform's been live for months now, serving thousands of registered users without a dramatic incident. I came on as a new contractor and ended up the sole active developer; since mid-June 2026 every commit on the project has been mine. Most of the work is the quiet kind: backfilling the gaps the migration left behind, fixing what real use shakes loose, and keeping the telemetry sharp enough to catch the rest before users do.

Some of it isn't upkeep at all. Recruiters can now query the platform straight from Claude, through a read-only MCP server: candidate search, company lookups, statuses. I built it as a separate service, then folded it into the backend so it needed no infrastructure of its own.

5 If this sounds like your codebase

If you've got a good codebase that still can't reach production, too many moving parts, a deploy only one person dares run, a migration measured in days, that's my

favourite kind of problem. Message me and tell me what you're dealing with.