Richard J. Doherty

Solo Founder & Technical Lead · AI-Assisted Product Delivery

richardjdoherty@gmail.com linkedin.com/in/richardjdoherty Melbourne, AU · Remote Work rights: AU · UK · USA

CASE STUDY · SHIPPING WITH AI TOOLING

Idea to App Store in ~7 Weeks — Solo, with Disciplined AI Development

Founder & sole developer · Kyvara Pty Ltd · multiple shipped products · 2026

TenantEvidence

IOS APP · SUBMITTED TO APP STORE

PropTech app helping Victorian renters document issues and generate compliant correspondence. React Native + Firebase, with five LLM-powered document generators.

Kirevra Press

WEB + YOUTUBE · LIVE

Editorial venture: a 10-part learning series. Privacy-pure static site at kirevra.com (HTTPS/HSTS) plus a published YouTube channel — built, deployed, and launched end-to-end.

WHAT THIS DEMONSTRATES

Over the past six months I designed, built, and shipped multiple production-quality products as a solo operator — using Claude (web, Claude Code, and Cowork) as a development partner under a deliberate engineering discipline. The differentiator isn't "I used AI." It's **how**: a structured, multi-session workflow with human-in-the-loop verification gates that caught real defects before launch.

THE WORKING METHOD

1. **Separation of duties across AI sessions.**
   A planning/spec session, a coding agent (Claude Code), and a design/deploy session — each in its own scope, with me as the deciding human courier between them. No AI surface acted without my ratification.
2. **Specify, generate, review, ship.**
   I owned product strategy, architecture, and the prompt/specification layer; directed code generation; then reviewed and tested every change before it merged. Authorship stays with the person directing and verifying — not the tool.
3. **Verify against ground truth, not the diff.**
   Every "fixed" claim was checked on-device or in production, not just in the code. This caught defects neither the code nor the review caught alone.
4. **Document as you go.**
   Contemporaneous records per work session and a running ledger of defects found and closed — an auditable trail, not just a finished artefact.

WHY IT'S RELEVANT TO A DATA / APP ENGAGEMENT

- **I ship.** Real products, in production, on tight timelines — not prototypes that stall.
- **Data-model & authorization rigour.** Designed the full data schema, access rules, and query-correctness gates; a pre-launch verification pass caught a cross-user data-exposure defect in a server-side function before any user was affected.
- **Applied LLM work, honestly scoped.** Built and version-controlled production prompts with an automated evaluation harness scoring outputs against a five-dimension accuracy rubric — prompt engineering and evaluation discipline, not model training.

Note: internal identifiers, specific defect details, and pre-launch metrics are withheld. Figures shown are real and conservative; products and company are matters of public record.

STACK & TOOLING

React Native · Expo / EAS · TypeScript · Firebase · Firestore · Cloud Functions · Node.js · Anthropic API · Python · Netlify · GitHub · Sentry · Claude Code · Claude Cowork

BY THE NUMBERS

~7 wk	company incorporation → iOS App Store submission, solo
2+	products shipped to web, YouTube & App Store
5	LLM document generators built, each version-controlled & evaluation-gated
~118	process/product defects caught & closed via multi-session verification

THE HONEST READ

This is applied AI-assisted development with engineering discipline — not ML model engineering, and not the SQL/relational work that sits in my earlier data-science and database background. Both are real; I keep them clearly distinct.