

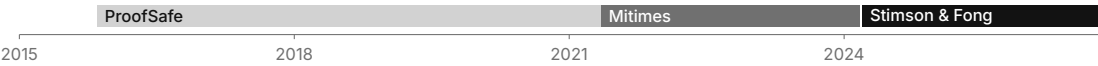
Calvin Fong

Senior Software Engineer · functional programming · mathematics

SUMMARY

I make fragile software correct and slow systems fast, usually by setting up a fast, reliable feedback loop that lets me identify simple abstractions for the core problem. Ten years shipping production systems in Elixir, F#, C#, Haskell and Python.

EXPERIENCE



Stimson & Fong Software Consulting Co-founder 2024 – now
I co-founded and run a software consultancy. We have delivered 14+ client projects, many end to end from scoping to production support, in healthcare, hospitality, mining, government, agriculture and research computing. These include Auth0 identity and access management for GIS-PAX (mining) and Wine Australia (government).

- I scope iteratively. Words are easy to misinterpret, so clients see working software early and often.
- I start each project by setting up the fastest reliable feedback loop for the task, whether that's a type checker, automated tests, pair programming or a simple generated PDF report.

Mitimes Software Engineer 2021 – 2024
I worked on a 3-person team across the Elixir backend and Elm frontend, maintained an F# on-premises phone connector, and owned the PostgreSQL database holding millions of captured calls and emails.

ProofSafe Software Engineer 2015 – 2021
I built and maintained a cross-platform field data collection app in C# and .NET, used by international clients including the Green Riyadh tree-planting project. It had offline-first sync with conflict resolution. I returned on contract in 2025.

SELECTED WORK

Subcontract
via Comunet
2026
NDA client

Fixing farm work allocation bugs

I was brought in because users were reporting bugs in the work allocation software. I discovered that work was often assigned to regions rather than to the underlying farms, which caused bugs when regions overlapped. I recognised the problem was similar to spreadsheet pivot tables, which meant the existing design was structurally flawed. I restructured the code and database to fix it.

Fixing the problem at its source resolved many reported and unreported bugs at once. I saw a general mathematical solution (an adjunction), but chose a simpler one.

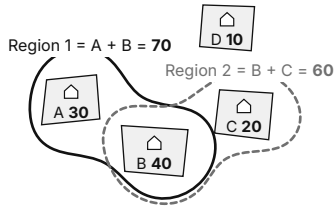


Fig. 1 Work assigned to a region is distributed to its farms. Viewing a region aggregates its farms. (Formally, an adjunction.)

ProofSafe
2015–2021
C#
.NET

Speeding up bulk data loading

Loading tree records slowed sharply as projects grew. 300 trees took an hour and 1,000 took 10 hours. Each tree was added one at a time, and every addition re-ran the form's calculations, refreshed the screen and wrote to the database. I moved the calculations into memory, so all trees are calculated first and then saved in one bulk write.

30,000 trees now load in under 5 minutes (a requirement for the Saudi Government Green Riyadh project).

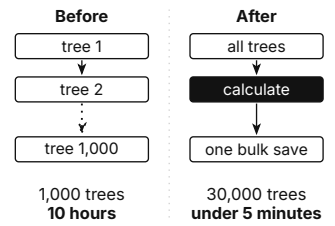


Fig. 2 Before, every tree was calculated, redrawn and saved on its own. After, the trees are calculated together and saved once.

ProHealthCare
2024-2025
C#
.NET
Azure OpenAI

AI-assisted medical document filing

Staff at a medical practice matched 100+ incoming documents a day to the right patient and doctor by hand. I built an on-site tool where OCR and an LLM suggest the match from 500,000+ patient records, and a staff member confirms each one.

Filing took about half the time. The practice stopped using it in 2025 because of staff adoption, not the software.

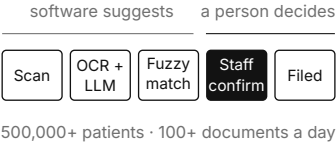


Fig. 3 The software suggests. A person decides.

Sirius Beta
Queensland
2024-2025
Haskell
Cloud Haskell

Parallel scientific computation in Haskell

A research client asked for a parallel version of one of their functions. I implemented a published algorithm that breaks a problem into a tree of smaller sub-problems, and used Cloud Haskell to run independent branches on separate machines. I used ThreadScope profiling to find bottlenecks.

Large speedups for a few specific types of computation. The rest ran in about the same time as before.

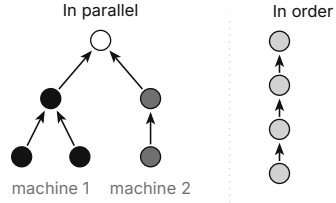


Fig. 4 Branches run at the same time on separate machines. A chain must run in order.

Kadabra AI
Singapore
2024-2025
Python
Twilio

AI ordering assistant for a Singapore startup

Kadabra AI hired me and my business partner as senior engineers to build its product. It started as an AI phone concierge for restaurants, using speech-to-text, an LLM and text-to-speech. It then pivoted to ordering over WhatsApp, where an LLM understands orders in the customer's own words instead of a fixed menu. It integrated with Stripe and Deliverect.

Ran in production with real customers, handling thousands of conversations, until the startup shut the product down in 2025.

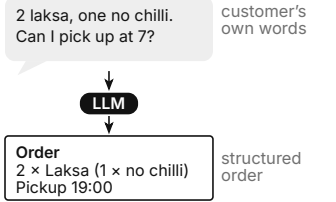


Fig. 5 Customers order in their own words. The LLM turns that into a structured order.

Personal
ongoing
Agda

Category theory textbook, formalised in Agda

I am translating Fong and Spivak's *Seven Sketches in Compositionality* into Agda, a language where the computer checks every proof. Chapters 1 and 2 are complete. It reflects how I approach all software. A precise type signature states what code is for, so you can think at the level of abstraction that matters without wading through irrelevant detail.

Published as a website designed so each topic can be understood from its type signatures in 30 seconds, with the proofs one click away.

github.com/ThereIsNoPie/an-invitation-to-category-theory

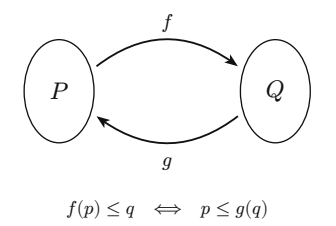


Fig. 6 A Galois connection, from chapter 1.

Personal
2026
React
Firebase

Wedding scavenger hunt app

I designed and built a scavenger hunt for our wedding guests. It had 27 puzzles, including QR codes hidden around the venue, riddles, online games and a 3D-printed cube decoded by shining a light through it.

Each solved puzzle unlocked a video about us, and a live leaderboard tracked everyone's progress.

thereisnopie.github.io/scavenger-hunt



Fig. 7 Scan to play.

EDUCATION

Bachelor of Engineering (Civil & Structural), First Class Honours
Bachelor of Mathematical and Computer Sciences
University of Adelaide · 2015

SKILLS

Languages Haskell, Elixir, F#, Elm, C#, Python, TypeScript, Agda, SQL
Methods Functional programming, property-based testing, pair programming, algorithm design, distributed systems, formal verification
Platforms PostgreSQL, AWS, Azure, Terraform, Auth0, React, Firebase, Twilio