

THOA TRAN • PRODUCT OWNER • AI AND COMPLEX B2B

I turn messy product problems into specs, rubrics and shipped fixes.

Two and a half years owning product scope, AI quality and support triage on a regulated lending platform, after nine years designing B2B products for investment funds, supply chains, hospitals and telecom operators.

Case 1 • Lending platform

From one lender to a lender-agnostic platform

Case 2 • AI quality

Making AI trustworthy where a wrong answer is a compliance risk

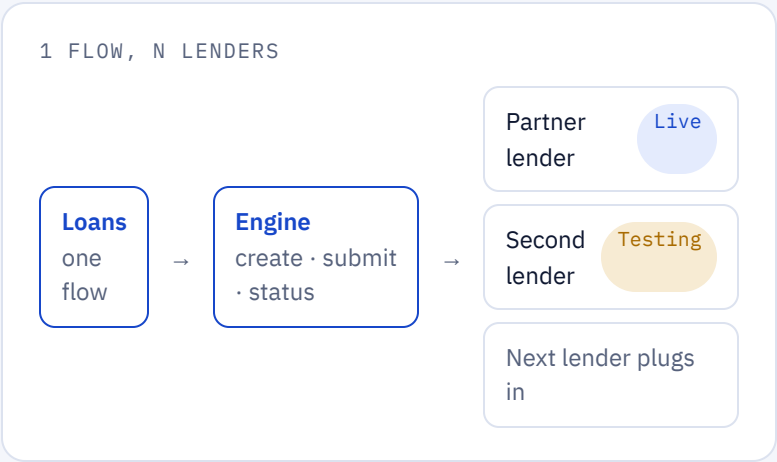
Case 3 • Open banking

Keeping brokers' trust when open banking breaks

First lender live · second in final testing

From one lender to a lender-agnostic platform

Brokers could submit personal loans to one partner lender. When a second lender arrived, with no screens of its own and an API-only integration, the easy path was a second one-off build. I led the work: designed the first flow from scratch, turned it into a pattern any lender can plug into, and scoped the second lender's first phase to about a week.



MY ROLE	TEAM	TIMELINE	USERS
Led the project: scope, phases, requirements, design and QA	Our engineers, the partner lender's team and its lead	June to September 2026, in phases	Mortgage brokers and advisers submitting loans for clients

Two statuses, not one

An application has its own state in our product: drafted, documents in, submitted. A lender has a separate state on its side. Merging them into one column looks simpler and breaks the moment one application goes to two lenders.

CLIENT	LENDER	APPLICATION STATUS	LENDER STATUS
J. Sample	Partner lender	Submitted	Approved
A. Example	Second lender	Submitted	Awaiting information
R. Demo	Second lender	Draft	—

Rebuilt Loans list, sample data

From a lender's email to a field dictionary

The lender's first request was a short email listing the fields it wanted. I checked each one against our data model, decided what we could send and what was out of scope, and wrote it up as a field dictionary. Engineering built the API from that document, adjusting the technical structure where they needed to; the scope and field meanings stayed the ones I set. I did the same for personal loans earlier: I wrote the base payload, and engineering built on it.

In interviews: *"I own what the data means and what's in scope; engineering owns how it's structured."*

Agree the contract before anyone designs a screen

The second lender's API was still being finalised. If engineers built against a guessed payload, every change on the lender's side would mean rework on ours. I broke the work into four tickets and put a gate in front of the UI.

Confirm lender API contract before building the submission UI

TASK · PRIORITY HIGH · BLOCKS 2 STORIES

- Create and submit payloads agreed in writing, field by field, against our field dictionary
- Every required lender field maps to data we already collect; gaps listed, not built, in Phase 1
- Sample requests pass against the lender's test endpoint for create, submit and status
- Lender status values mapped to our shared status list

Out of scope: new form fields; one application to several lenders.

The gate ticket as written, names removed

Live

Personal loan submission to the partner lender, with status syncing back

~1 week

Phase 1 scope for the second lender, reusing the existing flow and data

Testing

Second lender's agreement in place; end-to-end flow in final testing

1 flow

for every lender, with one shared lender status list

Decisions I'd defend, and what they cost

One Loans module for every lender

Brokers learn one flow; a new lender is one more option.

Cost I accepted: each lender's form must fit a shared structure.

Separate application and lender status

Ready for one application going to several lenders.

Cost I accepted: one more column to explain at first.

Contract gate before UI work

Engineers never build against a payload that might change.

Cost I accepted: UI work waits on the lender's timeline.

Phase 1 with existing data only

Proves the pattern in about a week.

Cost I accepted: a thinner first submission.

Dropping my own first idea

A submit option in the applications list would give one status column two meanings.

Cost I accepted: sketches and a review cycle already spent.

What I learned: the second integration is where a product decides whether it's building features or a platform. Most of the leverage came from deciding what to keep the same.

Shipped

Making AI trustworthy where a wrong answer is a compliance risk

Financial advisers use our AI to summarise client meetings, explain bank letters and fill bank forms. I owned the quality of what that AI wrote: templates, prompts, evaluation and bug tickets. When our vendor retired the model we ran on, I tested every candidate against an answer key and recommended a model for each feature.

MEETING SUMMARIES

Recorded disclosures as given when nobody gave them

Before

The adviser provided the disclosure statement...

After

Not discussed in this meeting. Confirm before next contact.

Change: report only what the transcript supports; one template generation instead of nine variants.

CLIENT LETTERS

Turned a bank's small print into the client's to-do list

Before

1. Sign the loan offer.
2. Arrange a registered valuation.

After

1. Sign the loan offer.
- Your approval has no extra conditions.

Change: separate a bank's generic conditions from the ones specific to this approval.

FORM FILLING

Filled fields it had no data for

Before

- ☒ I have disclosed all fees. no source
- ☒ I have sighted ID. no source

After

- ☐ Left for the adviser to confirm

Change: rules that hold for any bank form: no data, no fill; never sign or attest.

Snippets rebuilt with sample data

When the sources disagree

A client report pulls from the advice document, meeting transcripts, the client record and uploaded files. The hardest errors came from how the AI combined them: a gross salary read as net, rental income dropped, long statements never read. Deciding which source wins is a product decision, not a model setting.

DESIGN PROPOSAL · ORDER OF TRUST

1. Verified documents

→

2. Client record

→

3. Meeting transcript

When sources conflict, the AI shows the conflict to the adviser instead of choosing quietly, and every figure links back to its source.

How I know the output is right

"It reads well" is not a test. I wrote an answer key from one complete client file, ran each model three times per feature, and scored every check.

FORM FILLING CHECK	MODEL A	MODEL B	MODEL C
Leaves declarations unticked without evidence	3/3	1/3	0/3
Invents no loan details	3/3	3/3	0/3
Fills the branch field the file supports	3/3	0/3	—
Checks passed	9 of 9	4 of 9	0 of 6
DIARY NOTES CHECK	MODEL A	MODEL B	MODEL C
States the correct meeting date	2/3	2/3	3/3
Includes follow-up meeting and deadlines	6/6	0/6	—
Adds no advice that wasn't given	3/3	3/3	0/3
Checks passed	11 of 12	5 of 12	3 of 6

Model names withheld. A dash means the model had already failed a must-pass check.

20 / 21

answer key checks passed by the model I recommended

0 of 6

form filling checks passed by the default upgrade, so it was kept out

4

AI features evaluated before the vendor's retirement date

1

template generation to maintain, down from two

Decisions I'd defend

Answer key, not "reads well"

Fluent output hid a wrong date and a misread salary.

Cost I accepted: writing the key by hand before any run.

Repeated runs, not one demo

Caught a checkbox count jumping from 0 to 9.

Cost I accepted: three times the testing time.

One model per feature

Each feature gets the model that meets its definition of correct.

Cost I accepted: more config to retest at the next vendor change.

Flag risks beyond the ticket

Long financial documents were skipped silently, so business income could look like zero. Proposed a visible warning.

Cost I accepted: raised separately instead of widening scope.

Shipped and ongoing

Keeping brokers' trust when open banking breaks

Open banking went live in May. A large incident followed, then a steady stream of bank-side problems that brokers saw as our bugs. I took on support triage, found the causes, fixed what was ours to fix, and wrote the words brokers saw.

THE FIRST WEEK

Silence would read as "they don't know". A blocking alert would stop work that still mostly functioned. I designed a dismissible banner that named the problem, said what still worked, and promised the next update.

Some bank connections are delayed. Statements may take longer to appear. Existing applications aren't affected. We'll update you here by 5 pm.

Rebuilt banner, sample wording · hotfix live 7 days after go-live

Why statements went missing

A statement only arrives if its whole cycle falls inside the window the client consented to. Some home loans produce a statement every three to six months, so a three-month window can miss it entirely. I turned that into guidance brokers could act on: start a new application with a longer window, or ask the client to download the statement.

Defending a 93-day default

Brokers asked for every window to be long. Lenders need about three months for most applications; a year by default would expose older parts of a client's financial life nobody needs. **93 days stays the default**, with up to 365 days available for clients who need it. Privacy by default, flexibility on request.

Bank-side bugs that looked like ours

WHAT BROKERS SAW	ACTUAL CAUSE	WHAT WE DID
No running balance on transactions	One bank's API doesn't provide it	Explained the limit; client downloads the official statement meanwhile
Negative closing balances overnight	A faulty bank release, fixed next day but not retroactively	Told brokers exactly which connections needed a reconnect
"Duplicate" files per account	A statement and a transaction file, by design	Guidance article, so brokers stopped raising it

A playbook, not heroics

Most tickets fell into a few shapes, so I wrote down how to handle "it should have worked, but it didn't."

Hypothesis	Name the most likely cause from past tickets
One diagnostic question	The single answer that confirms or rules it out
Evidence request	A screenshot, an application number
No blame	Never imply the broker or client did it wrong
Workaround	Keep the application moving today
Contingency	What we do next if the hypothesis is wrong

Every recurring ticket also goes back to the product backlog as a signal about how broker firms actually work.

7 days

from go-live incident to hotfix in production

93 days

default kept, with up to 365 days for long statement cycles

Where my manager and I disagree

When a client reconnects their bank, summaries still show data from the first connection. We agree it needs fixing; we disagree on how.

MY PROPOSAL

Anchor every summary to one fixed window. One consistent period that matches what a lender assesses. Cost: a new application when data goes stale.

MANAGER'S PREFERENCE

Refresh to the latest data on every reconnect. Always current, no extra step. Cost: the period behind a summary can change between views.

I wrote up both with their trade-offs and left the call with my manager. The team will build from a written comparison, not a hallway opinion.

5+ months

owning open banking support since launch

1 playbook

plus rules for bank-side issues

What I learned: when your product depends on other companies' systems, trust comes less from never failing and more from explaining clearly what happened and what to do now.

New domains, same method

Each time, I learned the domain the same way: map who does what, which data moves between them, and where it breaks.

LEONARDO • INVESTMENT ANALYTICS • UK • 2019-23

Analytics and data visualisation for a platform funds use to review a company's track record before approving capital.

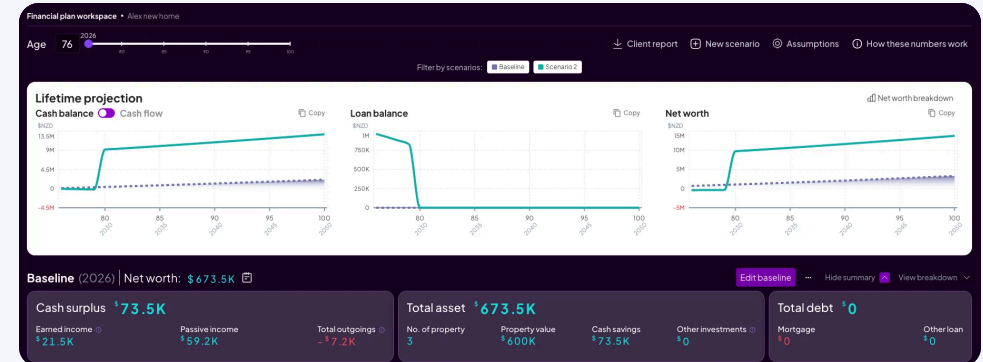
TRUSTONIC • DEVICE FINANCING • UK • 2019-23

Onboarding and support channel for a platform protecting 350M+ devices in 50+ countries: video and audio guides plus technical docs.

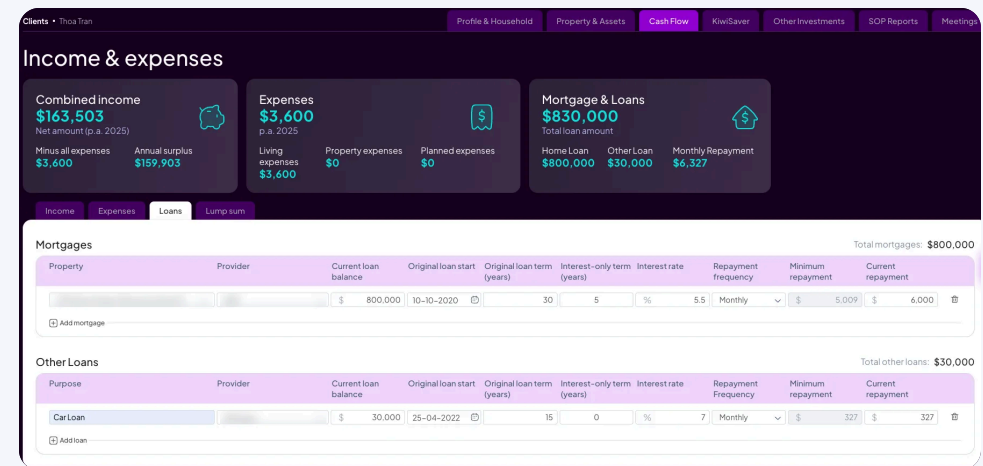
DICENTRAL • SUPPLY CHAIN AND HEALTHCARE • US • 2017-19

A hospital group's supply tool, an automotive suite redesign, warehouse management, and role and contact management across an EDI ecosystem.

PRODUCT SCREENS I DESIGNED • AFFORDX, CROPPED, DETAILS BLURRED



Financial plan workspace. Lifetime projections and side-by-side scenarios.



Client cash flow. Income, expenses and loans in one view.