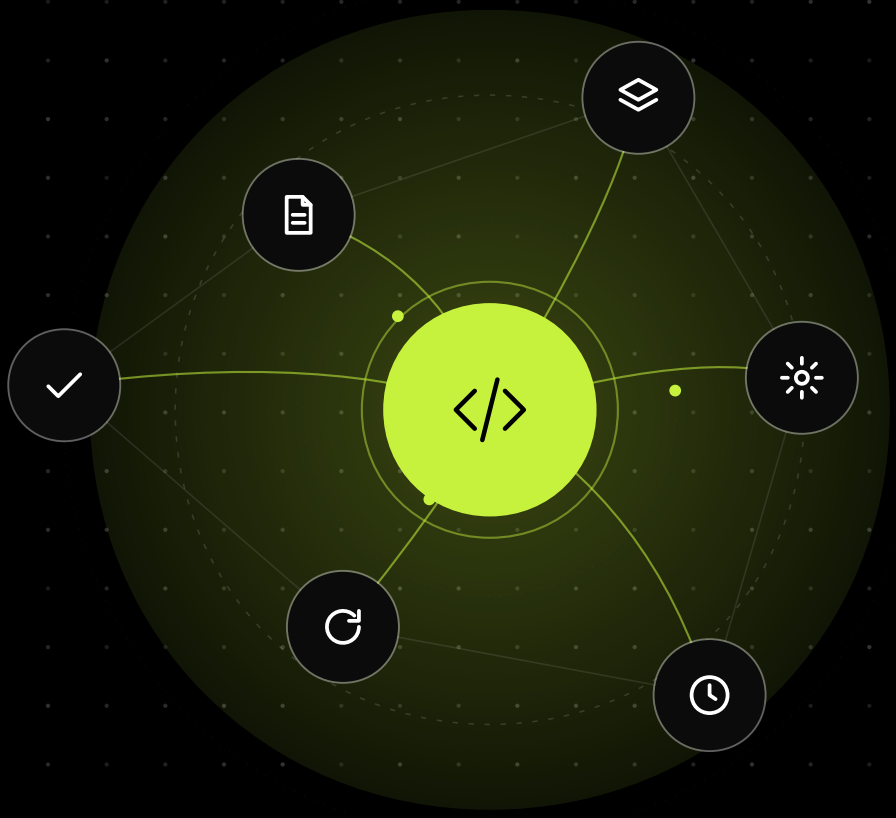


Airbnb

# A 1.5-year code migration finished in 6 weeks

Airbnb used an AI pipeline to move nearly 3,500 test files to a new testing framework. Work estimated at 1.5 years of engineering time by hand was finished in 6 weeks.

Travel and hospitality technology | Software engineering | AI pipeline | Automatic retries

**6 weeks**
to finish the whole migration

**75%**
of files migrated in the first 4 hours

**97%**
of files done after 4 days of tuning

AT A GLANCE

| COMPANY | THE JOB | AI USED | REPORTED |
| --- | --- | --- | --- |
| Airbnb | Nearly 3,500 React test files, from Enzyme to React Testing Library | Large language models (no vendor named) | The Airbnb Tech Blog, Mar. 13, 2025 |

01

# The challenge

Airbnb had used Enzyme for its React component tests since 2015. In 2020 it adopted React Testing Library for all new tests, which left nearly 3,500 older test files to convert.

Done by hand, the team estimated the work would take 1.5 years of engineering time.

Each file had to keep its original test intent and the codebase had to keep its overall code coverage, so this was careful rewriting, not a find and replace.

~3.5K

test files to convert

1.5 yrs

of engineering time, estimated by hand

2015

the year Enzyme tests started at Airbnb

BEFORE

- × Nearly 3,500 files on the old framework
- × Estimated 1.5 years of engineering time by hand
- × Every file a careful, manual rewrite
- × Engineers tied up on repetitive work

AFTER

- ✓ Files converted in hundreds at a time
- ✓ Each file checked automatically after every step
- ✓ Failures retried with the error messages
- ✓ Whole migration finished in 6 weeks

02

# The solution

The team broke the migration into separate steps for each file, so hundreds of files could run in parallel through a pipeline built in 2024.

Each file moved through a series of steps, and each step had to pass a check before the next. When a step failed, the pipeline gave the language model the validation errors and the latest version of the file, and tried again.

The prompts carried a lot of context, eventually 40,000 to 100,000 tokens each.

## HOW IT WORKS

STEP 01

### Refactor the test

The language model rewrites the Enzyme test for React Testing Library.

STEP 02

### Fix the test run

The file must pass its Jest tests before moving on.

STEP 03

### Fix lint and type checks

Lint and TypeScript checks must pass too.

STEP 04

### Retry with the errors

On failure, the errors and the latest file go back to the model.

STEP 05

### Mark complete

Files that pass every step are marked done.

STEP 06

### Finish the rest by hand

The last files were completed by engineers.

## BUILT WITH

Large language models · Step-by-step pipeline · Jest · Lint and TypeScript checks · Parallel runs · Automatic retries

03

# The results

The first bulk run migrated 75% of the files in 4 hours. Four days of tuning pushed that to 97%, and engineers finished the rest by hand in about another week.

**4 hours**
to migrate 75% of the target files

**97%**
done after 4 days of tuning, under 100 files left

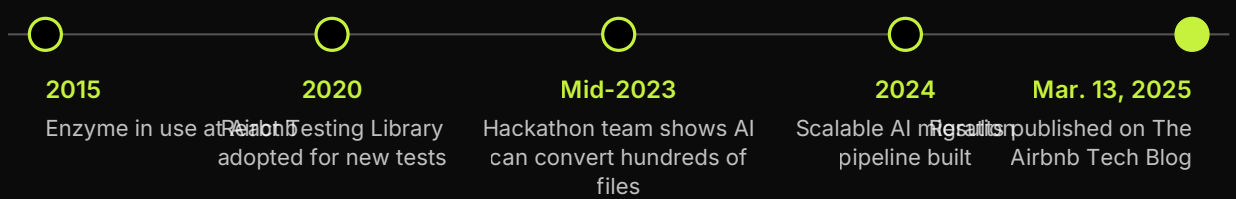
**6 weeks**
for the entire migration, versus 1.5 years by hand

**50 to 100**
retries on some of the hardest files

SHARE OF FILES MIGRATED OVER TIME




TIMELINE



04

# What made it work, and what to watch

> "We found the best approach was to push as much relevant context as possible into our prompts."
>
> Charles Covey-Brandt, The Airbnb Tech Blog

| | | |
|---|---|---|
| 1 | **Break the job into checkable steps** | Small, separate steps with a pass or fail check made the work safe to run in parallel. |
| 2 | **Let failures teach the next try** | Feeding the error messages back to the model turned failures into fixes. |
| 3 | **Context beats clever wording** | The team found that choosing the right context mattered more than how the prompt was worded. |
| 4 | **Keep people for the last mile** | About 3% of files were finished by hand. Test intent and code coverage were kept. |

## IS THIS A FIT FOR YOUR BUSINESS?

- ✓ Do you have hundreds or thousands of records or documents that need the same change?
- ✓ Can you describe what a correct result looks like for each item?
- ✓ Is the job big enough that it keeps getting pushed back?
- ✓ Would your team rather review exceptions than do every item by hand?

05

# How Agent Victoria could do this for you

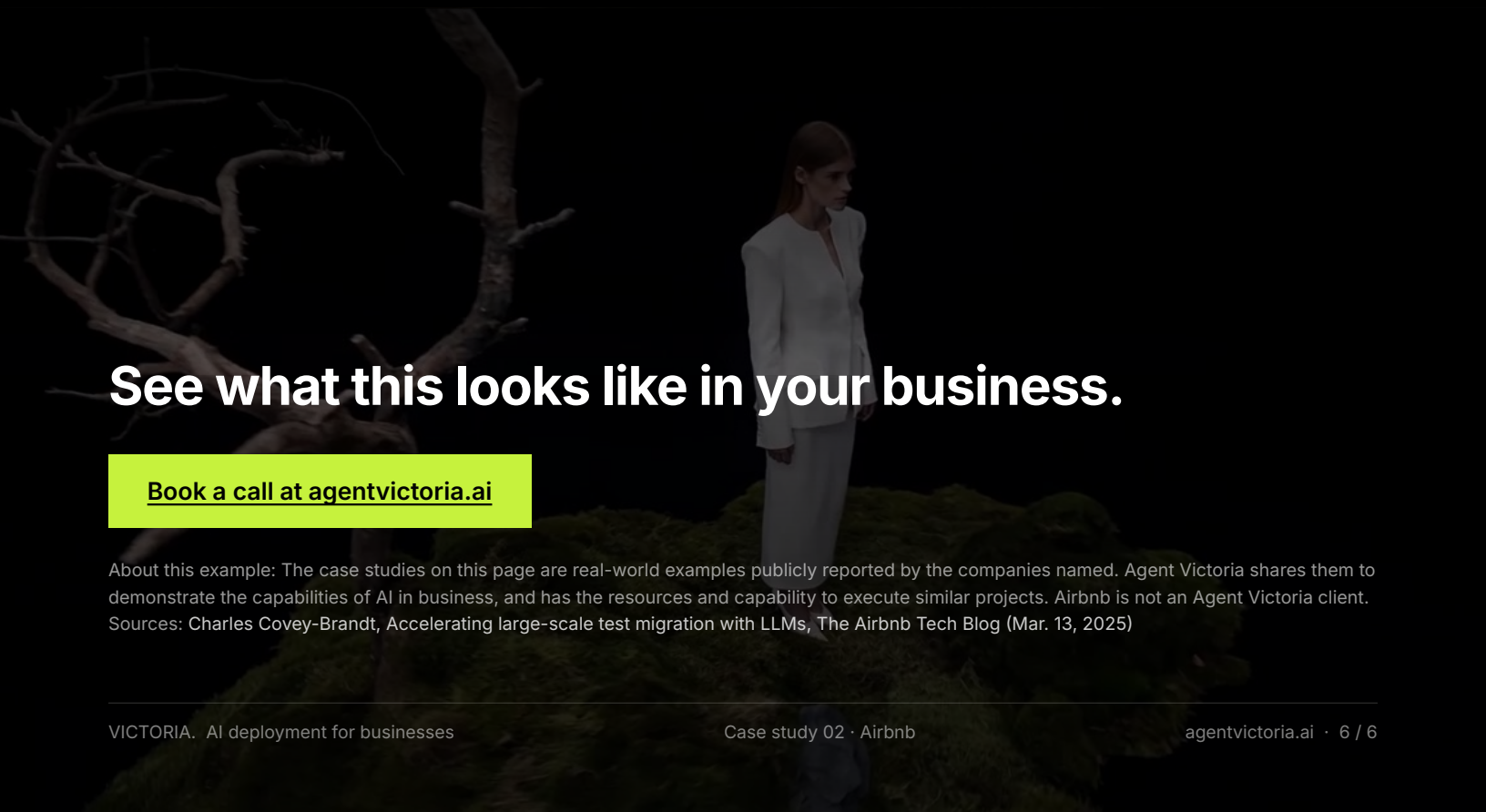
If you have a big, repetitive cleanup job, like moving records between systems or updating hundreds of documents to a new format, we could build an agent that does the work, checks it and retries until it passes. Your team reviews the exceptions instead of doing every item by hand.

01
**Define the job**
What each item looks like before and after.

02
**Set the checks**
Clear rules that say when an item is done right.

03
**Run, check, retry**
The agent works through every item until it passes.

04
**You handle exceptions**
Only the hard leftovers come to your team.



## See what this looks like in your business.

Book a call at agentvictoria.ai

About this example: The case studies on this page are real-world examples publicly reported by the companies named. Agent Victoria shares them to demonstrate the capabilities of AI in business, and has the resources and capability to execute similar projects. Airbnb is not an Agent Victoria client.
Sources: Charles Covey-Brandt, Accelerating large-scale test migration with LLMs, The Airbnb Tech Blog (Mar. 13, 2025)