**Engineer**

# Built a test suite across Go, SQL and browser

Built a layered automated test suite — 981 Go tests, 543 frontend and browser specs, 494 SQL behavioural tests — with mutation testing, property-based tests and an accessibility gate.

2025

## 981

Go tests, beside 543 browser and 494 SQL specs

**Situation.** A platform that keeps its business logic in the database has a testing problem most projects do not. The logic is not in the language the test framework is good at — it is in SQL, behind function boundaries, and SQL is exactly the sort of code that goes untested because testing it is awkward. Add a Go API and a Next.js frontend on top of that, and "the important parts are covered" quietly turns into "the parts that were easy to cover are covered."

**Task.** Each layer needed its behaviour checked where that behaviour actually lives, rather than everything being checked from the outside through a browser.

**Action.** Four layers got four kinds of test. The Go API carries 981 test functions across 310 files. The frontend carries 543 Vitest and Playwright specs, including 48 end-to-end files and 30 browser specs. The database carries 494 behavioural test files — 116,876 lines of SQL asserting through 6,654 raised exceptions — so a stored function is tested in the database instead of through three layers of application above it. Above those sit the tests that test the tests: Stryker mutation testing deliberately breaks a line and fails when nothing notices, and fast-check generates inputs nobody thought to write down. An axe-core gate asserts zero WCAG 2.0 and 2.1 A and AA violations, which makes accessibility a build failure rather than an audit finding months later. Coverage thresholds only ever move upward. The whole suite runs as 10 jobs in a 612-line workflow.

**Result.** Changing something structural stopped being frightening, which is the only thing that keeps a codebase this size from calcifying. The honest cost is time — the suite is slow, it taxes every change, and at this size it needs maintenance of its own. What it buys is the ability to keep moving quickly, and that is worth more than the minutes it takes.

### Bring this experience to your project

Use “Built a test suite across Go, SQL and browser” as a starting point for a similar challenge in your organisation. The services behind this work are available for your project. The part users never see: where the data lives, what the rules are, and how other systems are allowed to talk to it.

**Explore our services** Get in touch →

### Look behind the result

Read the situation, decisions and result in this example. We can discuss which parts transfer to your constraints and which need another approach. Built a test suite across Go, SQL and browser — 981 Go tests, beside 543 browser and 494 SQL specs

**Explore our services** Get in touch →

## Want tests that actually catch things?

Talk through your project with the engineer responsible for delivery. We agree the scope, price basis and handover in writing before work begins.

Role context: Test & Quality Engineer. It draws on Backend Engineering, Frontend Engineering, PostgreSQL, DevOps, Automation & CI/CD, Reliability & Backups and Testing & QA. And while you are here, we are rather good at Backend & API Development, DevOps & CI/CD Automation and Technical Leadership & Consulting.

**Get in touch** Explore our services

## One piece of **a longer record** — and it is all on this site.

Every entry is written the same way: the situation, the task, what we did and what changed.

**See the whole record**

### Define the improvement you need

Describe the workflow, the existing interfaces and the users who need the change. We can define a testable delivery and a handover that lets your team maintain it.

**See what it costs** Get in touch →

# Want tests that actually catch things?

Ask about the decisions behind “Built a test suite across Go, SQL and browser”. You will speak with the people who can explain the work and assess your own project.

Explore our services

welcome@engineer.company +45 50 10 61 84 Copenhagen, Denmark

Take this page with you

Contact card

Carefully Crafted With Love by D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – present · Engineer ApS · All rights reserved.

Printed from https://engineer.company/portfolio/built-a-test-suite-across-go-sql-and-browser/