Engineer Advisory

Testing & QA

21 achievements

Connect the parts of your system

The work collected under “Testing & QA” shows one part of a working system. We can help connect that part to the rest of your operation. Tested, released and reversible, so changes reach customers on a Tuesday afternoon rather than late on a Friday night.

Explore our services Get in touch →

Find a comparable challenge

Follow an example that resembles your situation. Its review describes the decisions behind the result, beyond the technology label. Overhauled internal processes, saving 8,000 hours — 8,000 hours saved by overhauling how the work ran

Overhauled internal processes, saving 8,000 hours Get in touch →

Streamlined CI/CD processes, saving 4,000 hours

Streamlined CI/CD processes, saving 4,000 hours by introducing automation in software development pipelines.

Migrated the API to Huma v2 at 100% parity

Migrated the HTTP API from Fiber to Huma v2 — 649 paths and 760 operations — reaching and holding 100% parity between the routes the server registers and the OpenAPI description it publishes.

Authored 578 go-task automation targets

Authored 578 go-task automation targets spanning native, Docker and HTTPS dev modes, linting, testing, database and deployment.

Set a zero-warnings bar across six languages

Set a zero-warnings quality bar across six languages — Go, TypeScript, SQL, Python, Shell and Markdown — enforced by pre-commit hooks.

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.

Built a 268-check guard engine for the repo

Built the repository's guard engine — 268 registered commit checks, 277 lint rules and 15 custom ESLint rules — plus 146 tests of the guards themselves, so the build holds the standard, not review.

Built a 22-linter commit gate

Built a commit gate of 22 one-line linters plus five that earn a paragraph, with no warning tier and no inline suppressions permitted, covering HTML, CSS, JavaScript, Python, YAML, Markdown, shell, links, spelling, secrets and typography.

A safety net that no longer costs half an hour

Cut the site's browser-driven quality gate from 1,636 seconds to 615 by scheduling its checks longest-first through a worker pool bounded to four lanes, after measuring that alphabetical order cost 320 seconds against 224.

Put the documentation under its own linter

Put the documentation under its own linter — a per-file line budget that only ratchets down, an index requirement and a real-path check — which found six of nine plans marked done while carrying unchecked boxes, and fourteen commit-gating tasks named in no document.

Linted the 10,242-line quality gate itself

Brought 10,242 lines of quality-gate JavaScript under a formatter and a linter after establishing it was the largest body of code in the repository and the only one nothing read, fixing 13 findings and suppressing none.

Held the generator to 981 tests and 92% coverage

Held the generator to 981 test cases at a 92% branch-coverage floor with warnings treated as failures, and asserted idempotence by running the whole build pipeline twice from an empty file and requiring the second pass to change nothing.

Passed 106 of 106 PDF/UA-1 profile checks

Recorded PDF/UA-1 validation across nine documents with 106 of 106 configured accessibility-profile checks passing, while the archival variant failed one of 146 rules.

Enabled every linter rule as an error

Selected every rule the Python linter has as an error, working through 1,815 findings to reach zero, with each of the few exemptions carrying a written reason and two of them backed by a checker instead of a comment.

Wrote tests for the checkers themselves

Wrote tests for the checkers themselves after establishing that a checker fed only clean input will one day report clean because it read nothing — planting a misspelling to confirm the spell-check finds it, and taking an id range from the database rather than from a number in the test.

Found the hooks and the gate disagreeing

Found the commit hooks and the quality gate running different checks while a document promised they were the same, by comparing the two lists in a test — the five that only ever ran by hand were the ones reading the CV prose.

Built a cross-language figure check

Built a cross-language content check that fails when a translation drops a figure the English states, and when a language uses notation it does not use — finding two Danish descriptions missing a metric and sixteen Ukrainian spans quoting in the English style.

Adopted Swift 6 strict concurrency over C

Adopted Swift 6 complete strict concurrency with no actors, bridging a blocking C event loop to the main actor through one producer, one consumer and one ordering — after establishing that a task per event loses the ordering the interface depends on.

Wrote a parser verifying every FFI call

Wrote a parser that reads the real 7,308-line C header and verifies every call site, every enum constant and that every pointer-owning class is final, after a hand-written placeholder header let calls to three removed functions compile, link and crash.

Took the test suite to 8.9 seconds

Took the test suite from six tests over a sixty-second limit to 135 passing in 8.9 seconds by profiling the main thread and removing the two calls it sat inside for 3,989 samples out of 4,017.

Rewrote error messages to stop blaming users

Rewrote the application's error messages against a written tone standard after a refused sign-in blamed the user for mistyping when the provider actually required an app-specific password, and covered it with a test that names the provider.

Made six collectors state what they skipped

Made six reporting collectors state the part of their input they never read — files that stayed shut, probes not installed, journal lines unrecognised — and fitted thresholds to 116 measured samples, after establishing that a blind collection and a healthy host render the same page.

Focus the engineering effort

Bring the deployment path, the operational risks and the people on call. We can scope a change with checks, a recovery route and clear ownership.

See what it costs Get in touch →

Put the subject in your context

Tell us how “Testing & QA” appears in your own system. We can help choose the relevant service and define the work around it.

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://advisory.engineer.company/categories/testing/