Engineer Software

Portfolio

Engineering services, with examples of delivered work.

Engineer ApS was registered in 2025. Most of the work shown here was delivered before that by our founder inside client teams; some has been delivered by the company itself, and some since in employment. The same hands and the same standards — we say which is which rather than blurring it.

Find experience worth applying

Use this record to find experience that is relevant to your project.

Explore our services Get in touch →

Read the decisions behind the record

The entries describe delivered work across services. Read a review to see the situation and decisions behind it.

Cases Get in touch →

Browse by category

SOFTWARE ENGINEERING

Backend Engineering
Frontend Engineering
Full-Stack Development
APIs & Integration
Web Development
Design Systems & UI
UX / UI Design
Testing & QA
Performance Tuning
Python
Internationalization

DATA & DATABASES

Databases
PostgreSQL
SQL
Data Engineering
Data Pipelines (ETL/ELT)
Data Analytics
Data Governance
Data Warehousing

GIS / Geospatial

CLOUD, INFRASTRUCTURE & DEVOPS

Cloud · DevOps · Automation & CI/CD · Infrastructure · Linux & Servers · System Administration · Networking & VPN · Monitoring & Observability · Reliability & Backups · Security · Platform Architecture · Migrations & Modernization

CONSULTING & LEADERSHIP

Technical Leadership · Team Leadership · Mentoring & Coaching · Agile & Scrum · Project Management · Product & Requirements · Stakeholder & Reporting · Documentation · Solution Architecture · Brand & Marketing

Everything below **was delivered** — not proposed, not planned.

Each entry names what the work changed and links the services it demonstrates.

What we offer What it costs →

{ Software Engineering }

Platform & Solution Architecture

Deciding how the whole thing should be built

The shape of a system worked out on paper first, so the parts fit and the cost is known before anyone writes code.

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.

A bounded event pipeline for a macOS app

Bounded event buffers and logging after a recorded macOS incident with 41 MB per second growth and 111 GB of compressed pages on a 36 GB machine; a recorded full test run reduced core log lines from 610,996 to 1,411.

Built a native macOS client in Swift 6

Built a native macOS messaging client in Swift 6 and SwiftUI — 11,141 lines across 53 files — over the C interface of a Rust core linked as a static archive from a pinned revision.

Vendored a QR encoder in 522 lines

Vendored a QR encoder — Reed-Solomon over GF(256), fixed module layout, eight mask patterns — in 522 lines rather than take a third runtime dependency, and verified it by reading the finished matrix back with an independently written decoder.

Built a database-driven CV generator

Built a database-driven CV, references, portfolio and cover-letter generator in Python — 41 modules, 10,580 lines — rendering six output formats from one 23-table SQLite source assembled by a 19-step idempotent pipeline.

Public services on a small server

Hosted the company's public services on a single-core 512 MB host — a git forge, a web server for seven domains, Tor, two alternative-protocol servers and intrusion banning — treating 464 MB of usable memory as the design constraint; backup automation is a separate implementation.

Built the entire core infrastructure from scratch

As one of the first hires, designed and built the entire core infrastructure and supporting processes from scratch for a green-energy SaaS startup, laying the foundation for rapid growth.

Built infrastructure that still runs years later

Planned and implemented new infrastructure functionality for internal and external systems, building solutions durable enough to still run years later with minimal change.

Set the platform's founding technical decisions

Set the platform's founding decisions in the weeks after the repository opened in November 2025 — the layering, database-first data access and the zero-warnings bar — and they still hold nine months on.

Modeled 348 tables across 34 Postgres schemas

Modeled the maritime domain into 348 normalized tables across 34 PostgreSQL schemas — professionals, companies, ships, jobs, reviews and the rest — with SMALLINT lookups and UUID v7 keys.

Designed a function-first PostgreSQL data layer

Designed a PostgreSQL function-first data layer — 1,275 stored functions across 34 schemas — so every read and write goes through a function the database can grant, rather than through a table.

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.

Built organization context switching for users

Designed an organization context-switching system with client localStorage and server-side cookie mirroring, letting users act as managed organizations while enforcing least-privilege authorization.

Designed a JSON passthrough from Postgres to API

Designed a JSON passthrough architecture where PostgreSQL functions return complete JSON forwarded verbatim by the Go API, eliminating intermediate unmarshalling and decoupling the frontend from schema changes.

A maritime platform with clear system boundaries

Architected a layered maritime platform separating a Next.js PWA frontend, a Go (Huma/Fiber) API, and a PostgreSQL function layer, keeping all business logic in the database.

Infrastructure simplified beyond AWS

Reworked an international client's infrastructure by removing unused services, adjusting capacity and moving workloads away from AWS. The founder reports a 10x reduction in cloud costs with no loss of productivity.

A mapping product, carried end to end

Architected, developed and supported infrastructure, data processing and a map application over 2 years, covering both software delivery and operational maintenance.

Overhauled internal processes, saving 8,000 hours

Overhauled internal processes, saving 8,000 hours by improving software architecture, systems, and scheduling efficiency.

Led a GIS map application to 10x revenue growth

Led the software development of a GIS map application, driving revenue growth by 10x and positioning the product as a primary data asset.

One infrastructure plan for 14 departments

Designed a comprehensive infrastructure framework for a research institute impacting 14 departments, featuring flexible modules, unified data pipelines, and structured support strategies for long-term adoption.

{ Software Engineering }

Backend & API Development

The engine behind your app or website

The part users never see: where the data lives, what the rules are, and how other systems are allowed to talk to it.

Reached the unused half of the core

Reached the half of the messaging core the application had never used — backup transfer, disappearing messages, message editing and resending, verified invitations, proxies and encryption policy — driving every test against the real library with no mocks.

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.

Moved every string into a content tree

Moved every user-facing string out of Python into a content tree of 874 files across three languages, after finding dead translations nobody could see were dead and a check silently grading a third of the achievements.

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.

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.

Built the hiring and seafarer career workspaces

Built hiring and seafarer career workspaces; the recorded implementation snapshot contains 151 stored functions across 48 tables for vacancies, applications, certificates, rank progression and sea time.

Built the social layer: posts, feed and groups

Built the platform's social layer — posts, feed, groups, mentions, a follower graph and an occasions digest — on the same function-first data layer as the rest of the product.

Built a loyalty wallet, leagues and badges

Built the loyalty and reputation system — 67 functions over a 31-table ledger, with leagues, badges and a redemption shop — taking a row lock on the balance to close the double-spend window.

Built fail-closed rate limiting and anti-abuse

Built fail-closed abuse controls — 22 Redis-backed rate limiters, Cloudflare Turnstile, request idempotency and an origin lock — so the platform sheds bots and floods instead of trusting its callers.

Built first-party error monitoring and tracing

Built first-party error monitoring and OpenTelemetry tracing rather than buying them — payload sanitising, spike and regression detection, symbolication and a synthetic heartbeat — behind 11 operator views.

Moved slow work onto a River job queue

Moved slow work off the request path onto a River job queue — 15 worker modules, 8 scheduled tasks and 20 pg_cron jobs — so a request returns while the work behind it carries on.

Built Stripe and in-app purchase entitlements

Built the payments and entitlements layer — Stripe alongside Apple and Google in-app purchase — gating the directory, search and export through an 11-table access model checked on the server.

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 and maintained Django apps for IPTV

Developed and maintained Django web applications for the IPTV platform, shipping new features and improving performance and stability.

Hardened the application with nonce-based CSP

Hardened the application with nonce-based CSP, HSTS, SameSite cookies, least-privilege database roles and server-side entitlement re-checks.

Implemented a catalogue-driven deep-merge

Implemented a catalogue-driven deep-merge for stored JSON preferences, preventing missing-key crashes as the schema evolves.

Built email with failover and delivery logging

Built email as a platform capability — three providers with failover, delivery webhooks, send and delivery logging, templating and campaigns — behind a startup check that will not boot without one.

Generated the API contract from the database out

Generated the API contract outward from the database — OpenAPI, a 44,076-line typed TypeScript client, 61 mock handlers and the limits the UI enforces — with a guard at every hop that fails on drift.

Developed a Data Analytics reporting system

Developed a Data Analytics reporting system, increasing quarterly software revenue by 400% through Python-based PDF reports.

Led 30+ GIS projects from build to support

Led the development, deployment, and support of over 30 GIS projects, demonstrating expertise in PostgreSQL, Bash, Python, JavaScript, GDAL, ArcGIS, PostGIS, and Mapbox technologies.

Directed full-stack GIS map development

Directed full-stack GIS map development, overseeing PostgreSQL, Mapbox, ReactJS, and NodeJS to deliver an integrated solution.

Delivered 8 Power BI projects with manuals

Delivered 8 Power BI projects with comprehensive manuals, integrating Microsoft Power BI tools with NodeJS API and Python FastAPI for effective data analytics and visualization.

Designed and ran 6 ETL/ELT pipelines

Designed, implemented, and administered 6 ETL/ELT pipelines, utilizing Google BigQuery, MSSQL, PostgreSQL, Shell scripting, PL/pgSQL, and Transact-SQL, integrating data for efficient Python API processing.

This service is part of work listed elsewhere on this page:

Adopted Swift 6 strict concurrency over C · Vendored a QR encoder in 522 lines · Built a database-driven CV generator · Set the platform's founding technical decisions · Designed a function-first PostgreSQL data layer · Migrated the API to Huma v2 at 100% parity · Built organization context switching for users · Designed a JSON passthrough from Postgres to API · A maritime platform with clear system boundaries

{ Software Engineering }

Frontend Development

The screens people actually use

Interfaces that load quickly, work on a phone and behave the same in every browser your customers happen to use.

Built a 48-token macOS design system

Built a 48-token design system on the platform's own glass material and wrote the linter that rejects a magic number, a hardcoded font size, or an animation that ignores the reduce-motion preference.

Named all 17 icons for screen readers

Named every icon-only control in the interface for screen readers after finding the send button announced as "arrow up circle, button", and wrote the linter that requires the label within eight lines of the icon.

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.

Cut the stylesheet bundle from 87 KB to 31 KB

Cut the stylesheet bundle from 87 KB to 31 KB by taking a base64 font out of it, and dropped 756 KB across 22 files that were published on every deploy and referenced by nothing.

Built a 637-line print stylesheet

Built a 637-line print stylesheet against four documented rendering-engine behaviours, including run-in headings that printed white on white for any reader whose browser preferred dark.

Made the reduced-motion setting honest

Made reduced-motion honest after finding eleven selectors still animating under the preference, because a universal transition-none rule loses on specificity to any rule carrying a class.

Found the light theme missing a whole layer

Found that the light theme had been missing a full-page overlay layer since the day it was written, by asserting that both themes paint every layered surface with the same number of layers.

Closed the colour system at 28 colours

Closed the colour system at 28 documented colours with a linter that fails a value painted but undocumented, documented but unpainted, mis-measured, respelled as a literal, or within a perceptual distance of 0.02 of one already there.

The accessibility faults your checker misses

Found and fixed 11 accessibility defects the browser reported as healthy — eight footer links still in tab order behind pointer-events, a scroll timeline clipping the colophon off three pages, and a rule engine running 70 of its 105 rules.

Accessibility checks across 32 pages and both themes

Implemented WCAG 2.2 Level AA checks across 32 representative pages in both colour themes, with a conformance record, one AAA criterion checked and the partial result for criterion 1.4.13 stated explicitly.

Replaced 27 font sizes with a 6-step scale

Replaced 27 rendered font sizes whose nearest neighbours were 0.6% apart with a six-step Major Third scale, and wrote the linter that fails the twenty-eighth.

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.

Built a multilingual Hugo site in 3 languages

Built a multilingual static site in Hugo across 108 template files including 53 partials, publishing every page in four representations from one content tree in three languages, and again under seven focused subdomains built from that same tree.

Built the operator back-office for the platform

Built the operator back-office — around 40 admin routes and 128 components covering claims, moderation, feature flags, cache and diagnostics — so the platform can be run without database access.

Maintained a Hugo site and improved product UI

Maintained and enhanced the legacy Hugo static-site website while contributing UI/UX improvements to the primary asset-management product.

Built an English/Danish i18n system

Established an English/Danish internationalization system carrying 12,027 messages per locale across 458 namespace files, with linter-enforced vocabulary and a 400-line budget per file.

Designed the product's UI and UX end to end

Designed the product's UI and UX end to end — directory grids, dual card/table views, live requirement validators and breadcrumb navigation.

Designed an iOS 26 'Liquid Glass' design system

Designed an iOS 26 'Liquid Glass' design system and a canonical component inventory enforced by lint rules to prevent UI divergence.

Delivered full Progressive Web App support

Delivered full Progressive Web App support — installable and offline-capable — with Workbox runtime caching via next-pwa.

Built the Next.js 16 frontend: SSR, SSG and CSR

Built the Next.js 16 frontend with deliberate SSR, SSG and CSR strategies, and a reusable prefetched-server-page factory that removes the N+1 fetch cascade from each authenticated page moved onto it.

Designed 30 websites from Photoshop to HTML

Designed 30 websites, delivering unique and flexible solutions by converting Photoshop designs to HTML.

Developed 100 web applications using HTML/HTML5

Developed 100 web applications using HTML/HTML5, CSS/SCSS, Django, WordPress, and Joomla, ensuring diverse online presence.

Contributed to user interface design processes

Contributed to user interface design processes, ensuring intuitive, visually appealing, and user-friendly project interfaces.

Presented 200 UI/UX improvements

Presented 200 UI/UX improvements for the GIS map application, boosting revenue by 10x through enhanced software features.

This service is part of work listed elsewhere on this page:

A bounded event pipeline for a macOS app · Built a native macOS client in Swift 6 ·

Set the platform's founding technical decisions · Built organization context switching for users ·

A maritime platform with clear system boundaries · Directed full-stack GIS map development

{ Software Engineering }

Full-Stack Product Development

A working product, end to end

One team taking an idea from a sketch to something your customers can log into, instead of handing it between three suppliers.

Built company ownership claims and adjudication

Built company ownership claims end to end — a user claims a company, an administrator adjudicates, and an approval rewrites the authorization graph that decides who is allowed to edit what.

This service is part of work listed elsewhere on this page:

Built a native macOS client in Swift 6 · Built a database-driven CV generator ·

Built a multilingual Hugo site in 3 languages · Built the hiring and seafarer career workspaces ·

Built the social layer: posts, feed and groups · Built the operator back-office for the platform ·

Built and maintained Django apps for IPTV · Set the platform's founding technical decisions ·

Built the Next.js 16 frontend: SSR, SSG and CSR · A maritime platform with clear system boundaries · A mapping product, carried end to end · Developed 100 web applications using HTML/HTML5 · Led 30+ GIS projects from build to support · Directed full-stack GIS map development · Led a GIS map application to 10x revenue growth

{ Software Engineering }

Website Development & CMS

A website you can update yourself

A site that is quick, findable and editable by your own staff, without a bill for every change of wording.

Published a readable Tor onion mirror

Published the site as a Tor onion mirror on a readable address beginning engineer, with the key generated off the host so it never reached this repository or a laptop, and left it the one front door of four that is never counted.

Built the visual identity from 2 drawings

Built the company's icon and favicon sets from two hand-made drawings, with every derived asset regenerated by script and a check that fails when a derived file was committed before its source.

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.

Fixed a sitemap with one shared timestamp

Fixed a sitemap where 172 of 176 URLs shared one modification timestamp, by taking the date from git history after establishing that the export rewrites every file on every run.

Replaced 963 metadata blocks with one graph

Replaced 963 anonymous structured-data blocks that restated the company 1,671 times with one linked graph of 16 types minted from stable origin identifiers.

Mirrored the site to Gemini and Gopher

Built Gemini and Gopher outputs from the same website source; the achievement records 777 documents in each format at that stage, with zero HTML bytes changed by adding the formats.

Generated 495 pages from a SQLite export

Generated 495 achievement pages across three languages from a read-only SQLite export, with the page address authored as data so that correcting a sentence no longer moved the page and broke the link.

Your code on your own server

Deployed the company's own git forge on Soft Serve, private by default with no web panel and its SSH port bound to loopback behind a jump host, and made the landing page in front of it a build artefact of the main site rather than a hand-kept copy.

Combined web development with SEO and content

Drove client web success by combining custom website development and design with SEO, content strategy and copywriting.

Administered 40 websites on Apache and Nginx

Administered 40 websites on Ubuntu Linux hosting servers with Apache and Nginx, ensuring high availability and performance.

This service is part of work listed elsewhere on this page:

Cut the stylesheet bundle from 87 KB to 31 KB · Built a 637-line print stylesheet · Made the reduced-motion setting honest · The accessibility faults your checker misses · Accessibility checks across 32 pages and both themes · Built a multilingual Hugo site in 3 languages · Maintained a Hugo site and improved product UI · Built and maintained Django apps for IPTV · Delivered full Progressive Web App support · Built the Next.js 16 frontend: SSR, SSG and CSR · Designed 30 websites from Photoshop to HTML · Developed 100 web applications using HTML/HTML5

{ Software Engineering }

AI Agent Readiness

Showing up when customers ask an AI

A site or an app that the AI assistants your customers use can find, read and act on, checked on every release rather than promised once.

Ran development as 51 written initiatives

Ran the site's development as 51 written initiatives across 54 plan documents, each carrying its open boxes, its measurements and the decisions it declined — including six of eight agent-readiness findings refused with the reason recorded.

This service is part of work listed elsewhere on this page:

Fixed a sitemap with one shared timestamp · Replaced 963 metadata blocks with one graph · Accessibility checks across 32 pages and both themes · Built a multilingual Hugo site in 3 languages

{ Software Engineering }

AI Agent Tools & MCP Servers

Letting AI use the software you already have

Your existing software opened up as tools an AI agent can call, without rewriting anything that already works.

This service is part of work listed elsewhere on this page:

Built the operator back-office for the platform · Moved slow work onto a River job queue · Generated the API contract from the database out · Migrated the API to Huma v2 at 100% parity · Designed a JSON passthrough from Postgres to API

{ Software Engineering }

GIS & Geospatial Solutions

Maps that answer questions

Anything that happens in a place — vessels, vehicles, plots, sensors — put on a map you can search, measure and share.

Managed 30 successful GIS projects to delivery

Managed the execution of 30 successful GIS projects, demonstrating leadership in delivering innovative solutions across the field.

Optimized forecasting and investment strategies

Optimized forecasting and investment strategies for 11 electricity grid operators, driving operational efficiency through data-driven GIS solutions.

Automated delivery of 20 GIS data pipelines

Automated delivery of 20 GIS data pipelines and app data ETL processes, streamlining infrastructure automation and reporting.

Automated GIS SaaS deployment with GitHub Actions

Automated GIS SaaS application deployment, data processing, and reporting system using GitHub Actions CI/CD, Python, Bash, and SQL.

Ran 100 PostgreSQL, MS SQL, BigQuery warehouses

Architected, created, and managed 100 PostgreSQL, MS SQL, and Google BigQuery data warehouse databases with primarily GIS and time-series data, optimizing performance and scalability.

Released 500 electricity and GIS data reports

Released 500 electricity and GIS data analysis reports, utilizing deep research and troubleshooting to ensure accurate geographic and time series big data insights.

Resolved 1,000 issues in geographical data

Resolved 1,000 issues in geographical data and time-series data, using GDAL, ArcGIS, PostGIS, Mapbox, QGIS, SQL (PL/pgSQL, Transact-SQL), Bash, ensuring high-quality big data processing.

Improved GIS map performance 10x on PostgreSQL

Improved geographical map application performance by 10x through strategic database transition from MSSQL to PostgreSQL, including the data cutover, optimizing processing and data security.

Made a geographical data pipeline 50x faster

Recorded a 50x geographical data-pipeline improvement in the founder's Utiligize work through SQL and data-model changes across PostgreSQL, MS SQL and Google Cloud BigQuery.

This service is part of work listed elsewhere on this page:

A mapping product, carried end to end · Led 30+ GIS projects from build to support ·

Directed full-stack GIS map development · Led a GIS map application to 10x revenue growth

{ Software Engineering }

UI/UX Design & Design Systems

Making it obvious how to use

Layout, wording and flow arranged so people finish what they came to do instead of ringing your support line.

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.

This service is part of work listed elsewhere on this page:

Built the visual identity from 2 drawings · Built a 48-token macOS design system ·

Named all 17 icons for screen readers · Built a native macOS client in Swift 6 ·

Passed 106 of 106 PDF/UA-1 profile checks · Built a 637-line print stylesheet ·

Made the reduced-motion setting honest · Found the light theme missing a whole layer ·

Closed the colour system at 28 colours · The accessibility faults your checker misses ·

Accessibility checks across 32 pages and both themes · Replaced 27 font sizes with a 6-step scale · Combined web development with SEO and content · Maintained a Hugo site and improved product UI · Designed the product's UI and UX end to end · Designed an iOS 26 'Liquid Glass' design system · Designed 30 websites from Photoshop to HTML · Contributed to user interface design processes · Presented 200 UI/UX improvements

{ Software Engineering }

Internationalization & Localization

Working in more than one language

Software that reads naturally in every market you sell in, including the dates, the currencies and the way names sort.

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.

This service is part of work listed elsewhere on this page:

Moved every string into a content tree · Built a database-driven CV generator · Mirrored the site to Gemini and Gopher · Generated 495 pages from a SQLite export · Accessibility checks across 32 pages and both themes · Built a multilingual Hugo site in 3 languages · Built an English/Danish i18n system

{ Software Engineering }

Custom AI Agent Development

An AI assistant that knows your trade

An agent wired to the systems you use, tested on cases your own experts write, overseen by your people and never tied to one model.

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.

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.

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.

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.

This service is part of work listed elsewhere on this page:

Built first-party error monitoring and tracing

Compare your starting point

Choose a comparable piece of work and identify what differs in your own situation.

See what it costs Get in touch →

Discuss a relevant approach

Ask us to turn the relevant experience into an approach you can evaluate.

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://software.engineer.company/portfolio/