Engineer Software

{ Platform- & løsningsarkitektur }

At beslutte, hvordan det hele skal bygges

Systemets form lagt fast på papir først, så delene passer sammen, og prisen er kendt, før nogen skriver kode.

Er I stadig ved at finde ud af, hvad I har brug for?

Fortæl os, hvad der går galt, eller hvad I prøver at bygge. Så siger vi, hvad det vil kræve, cirka hvad det vil koste, og om vi er de rigtige til det.

Fra 1.400 DKK i timen, ekskl. moms — og mindre pr. time ved en længere booking.

Kontakt os Se, hvad det koster

Udvalgte resultater, der demonstrerer denne ydelse.

Gør ydelsen til et resultat

Lad »Platform- & løsningsarkitektur« understøtte et forretningsmål. Vi forbinder det tekniske arbejde med de brugere, den drift og de beslutninger, det skal støtte. Systemets form lagt fast på papir først, så delene passer sammen, og prisen er kendt, før nogen skriver kode.

Kontakt os

Undersøg det relevante arbejde

Brug arbejdet på siden til at undersøge vores tilgang. Spørg til afvejningerne og forudsætningerne bag hvert resultat. Én infrastrukturplan til 14 afdelinger — 14 forskningsafdelinger på én infrastrukturplan

Én infrastrukturplan til 14 afdelinger Kontakt os →

Én infrastrukturplan til 14 afdelinger

Har designet en omfattende infrastrukturramme for et forskningsinstitut, der omfatter 14 afdelinger, med fleksible moduler, samlede datapipelines og strukturerede supportstrategier, med henblik på langsigtet forankring.

Har ført en GIS-kortapp til 10x omsætningsvækst

Har ledet softwareudviklingen af en GIS-kortapplikation, øget omsætningen 10 gange og positioneret produktet som et primært dataaktiv.

Har fornyet processer og sparet 8.000 timer

Har gennemgribende fornyet interne processer og sparet 8.000 timer ved at forbedre softwarearkitektur, systemer og planlægningseffektivitet.

Et kortprodukt, båret fra ende til anden

Har designet, udviklet og understøttet infrastruktur, databehandling og en kortapplikation over 2 år med både softwareleverancer og driftsvedligeholdelse.

Forenklet infrastruktur uden for AWS

Har omlagt en international kundes infrastruktur ved at fjerne ubrugte tjenester, tilpasse kapaciteten og flytte arbejdsbelastninger væk fra AWS. Grundlæggeren beskriver en reduktion af skyomkostningerne med en faktor 10 uden tab af produktivitet.

En maritim platform med klare systemgrænser

Har designet arkitekturen for en lagdelt maritim platform, der adskiller en Next.js PWA-frontend, et Go (Huma/Fiber) API og et PostgreSQL-funktionslag og holder al forretningslogik i databasen.

Har designet en JSON-passthrough-arkitektur

Har designet en JSON passthrough-arkitektur, hvor PostgreSQL-funktioner returnerer komplet JSON, som Go-API'et videresender uændret, så der ikke er nogen mellemliggende unmarshalling, og frontenden er afkoblet fra skemaændringer.

Har bygget skift af organisationskontekst

Har designet et system til skift af organisationskontekst med localStorage i klienten og en spejlet cookie på serveren, så brugere kan optræde som de organisationer, de administrerer, mens autorisationen følger mindste privilegium.

Har migreret API'et til Huma v2 med 100 % paritet

Har migreret HTTP-API'et fra Fiber til Huma v2 — 649 paths og 760 operationer — og opnået og fastholdt 100 % paritet mellem de ruter, serveren registrerer, og den OpenAPI-beskrivelse, den udgiver.

Har designet et funktionsbaseret Postgres-datalag

Har designet et funktionsbaseret datalag i PostgreSQL — 1.275 lagrede funktioner fordelt på 34 skemaer — så al læsning og skrivning går gennem en funktion, databasen kan tildele rettigheder til, og ikke direkte gennem en tabel.

Har modelleret 348 tabeller i 34 Postgres-skemaer

Har modelleret det maritime domæne i 348 normaliserede tabeller fordelt på 34 PostgreSQL-skemaer — fagfolk, virksomheder, skibe, jobs, anmeldelser og resten — med SMALLINT-opslagstabeller og UUID v7-nøgler.

Har fastlagt platformens grundbeslutninger

Har fastlagt platformens grundlæggende beslutninger i ugerne efter, at repositoryet blev oprettet i november 2025 — lagdelingen, database-first-dataadgangen og standarden med nul advarsler — og de holder stadig ni måneder senere.

Har bygget infrastruktur, der holder i årevis

Har planlagt og implementeret ny infrastrukturfunktionalitet til interne og eksterne systemer og bygget løsninger, der er så robuste, at de stadig kører år senere med minimale ændringer.

Har bygget hele kerneinfrastrukturen fra bunden

Har som en af de første medarbejdere designet og bygget hele kerneinfrastrukturen og de understøttende processer fra bunden for en SaaS-startup inden for grøn energi og lagt fundamentet for hurtig vækst.

Offentlige tjenester på en lille server

Har drevet virksomhedens offentlige tjenester på en vært med 512 MB og én kerne — en git-forge, en webserver til syv domæner, Tor, to servere til alternative protokoller og blokering af indtrængningsforsøg — med 464 MB brugbar hukommelse som designbegrænsning; backupautomatik er en særskilt implementering.

Har bygget en databasedrevet CV-generator

Har bygget en databasedrevet generator til CV, referencer, portefølje og ansøgninger i Python — 41 moduler, 10.580 linjer — der genererer seks outputformater fra én SQLite-kilde med 23 tabeller, som en idempotent pipeline i 19 trin bygger op.

Har indlejret en QR-encoder på 522 linjer

Har indlejret en QR-encoder — Reed-Solomon over GF(256), fast modullayout, otte maskemønstre — på 522 linjer i stedet for at tage en tredje runtime-afhængighed ind og verificeret den ved at læse den færdige matrix tilbage med en uafhængigt skrevet dekoder.

Har bygget en native macOS-beskedklient i Swift 6

Har bygget en native macOS-beskedklient i Swift 6 og SwiftUI — 11.141 linjer fordelt på 53 filer — oven på C-grænsefladen til en Rust-kerne linket som et statisk arkiv fra en fastlåst revision.

Et afgrænset hændelsesflow i en macOS-app

Har begrænset hændelsesbuffere og logning efter en registreret macOS-hændelse med vækst på 41 MB i sekundet og 111 GB komprimerede sider på en maskine med 36 GB; en registreret fuld testkørsel reducerede kernens loglinjer fra 610.996 til 1.411.

Har indført Swift 6 strict concurrency over C

Har indført Swift 6's complete strict concurrency uden actors og bygget bro fra en blokerende C-hændelsesløkke til main actor gennem én producent, én forbruger og én rækkefølge — efter at have fastslået, at en opgave pr. hændelse mister den rækkefølge, grænsefladen afhænger af.

Leder I efter Platform- & løsningsarkitektur?

Brug de tilknyttede opgaver til at vurdere relevant erfaring. Fortæl os, hvad I skal bygge; vi kan tale om omfang, fremgangsmåde og overdragelse.

Kontakt os Hele porteføljen

Afgræns opgaven før bestilling

Beskriv arbejdsgangen, de eksisterende grænseflader og de brugere, der har brug for ændringen. Vi kan definere en testbar leverance og en overdragelse, der gør teamet i stand til at vedligeholde løsningen.

Se, hvad det koster Kontakt os →

Gennemgå jeres krav med os

Fortæl os, hvad I har brug for inden for »Platform- & løsningsarkitektur«, herunder fristen og det team, der skal vedligeholde løsningen. Vi kan drøfte en praktisk vej frem.

Kontakt os

welcome@engineer.company +45 50 10 61 84 København, Danmark

Tag denne side med dig

Visitkort

Lavet med kærlighed af D.

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

Copyright © 2025 – nu · Engineer ApS · Alle rettigheder forbeholdes.

Udskrevet fra https://software.engineer.company/da/services/platform-architecture/