Engineer Programvare

{ Plattform- og løsningsarkitektur }

Bestemme hvordan helheten skal bygges

Formen på et system tegnet opp på papiret først, slik at delene passer sammen og kostnaden er kjent før noen skriver kode.

Avklarer dere fortsatt hva dere trenger?

Fortell oss hva som går galt, eller hva dere prøver å bygge. Vi sier hva det vil kreve, omtrent hva det vil koste, og om vi er de rette til oppgaven.

Fra 1 400 DKK timen, eksklusive mva. — og mindre per time ved et lengre oppdrag.

Ta kontakt Se hva det koster

Utvalgt arbeid som viser denne tjenesten.

Gjør tjenesten til et resultat

La «Plattform- og løsningsarkitektur» støtte et forretningsmål. Vi kan knytte det tekniske arbeidet til brukerne, driften og beslutningene det skal støtte. Formen på et system tegnet opp på papiret først, slik at delene passer sammen og kostnaden er kjent før noen skriver kode.

Ta kontakt

Undersøk det relevante arbeidet

Bruk arbeidet på denne siden til å undersøke tilnærmingen vår. Spør om avveiningene og forutsetningene bak hvert resultat. Én infrastrukturplan for 14 avdelinger — 14 forskningsavdelinger på én felles infrastrukturplan

Én infrastrukturplan for 14 avdelinger Ta kontakt →

Én infrastrukturplan for 14 avdelinger

Designet et helhetlig infrastrukturrammeverk for et forskningsinstitutt som berørte 14 avdelinger, med fleksible moduler, samlede datapipelines og strukturerte støttestrategier for varig innføring.

Ledet en GIS-kartapp til 10 ganger mer omsetning

Ledet programvareutviklingen av en GIS-kartapplikasjon, økte inntektene 10 ganger og posisjonerte produktet som en primær dataressurs.

Fornyet interne prosesser og sparte 8 000 timer

Fornyet interne prosesser og sparte 8 000 timer ved å forbedre programvarearkitekturen, systemene og effektiviteten i planleggingen.

Et kartprodukt, håndtert fra ende til annen

Utformet, utviklet og støttet infrastruktur, databehandling og en kartapplikasjon over 2 år, med både programvareleveranser og driftsvedlikehold.

Forenklet infrastruktur utenfor AWS

La om infrastrukturen til en internasjonal kunde ved å fjerne ubrukte tjenester, tilpasse kapasiteten og flytte arbeidslaster bort fra AWS. Grunnleggeren beskriver en reduksjon i skykostnadene med en faktor på 10 uten tap av produktivitet.

En maritim plattform med klare systemgrenser

Utformet arkitekturen for en lagdelt maritim plattform som skiller en Next.js PWA-frontend, et Go-API (Huma/Fiber) og et funksjonslag i PostgreSQL, med all forretningslogikk i databasen.

Designet JSON rett fra Postgres til API-et

Designet en JSON-passthrough-arkitektur der PostgreSQL-funksjoner returnerer komplett JSON som Go-API-et videresender uendret, noe som fjernet mellomliggende unmarshalling og frikoblet frontend fra skjemaendringer.

Bygde bytte av organisasjonskontekst for brukerne

Designet et system for å bytte organisasjonskontekst med localStorage på klienten og speiling i informasjonskapsler på serversiden, slik at brukere kan opptre som organisasjoner de administrerer, med autorisasjon etter prinsippet om minste privilegium.

Migrerte API-et til Huma v2 med 100 % paritet

Migrerte HTTP-API-et fra Fiber til Huma v2 — 649 stier og 760 operasjoner — og nådde og holdt 100 % samsvar mellom rutene serveren registrerer og OpenAPI-beskrivelsen den publiserer.

Designet et funksjonsbasert datalag i PostgreSQL

Designet et funksjonsbasert datalag i PostgreSQL — 1 275 lagrede funksjoner fordelt på 34 skjemaer — slik at hver lesing og skriving går gjennom en funksjon databasen kan gi tilgang til, i stedet for gjennom en tabell.

Modellerte 348 tabeller i 34 Postgres-skjemaer

Modellerte det maritime domenet i 348 normaliserte tabeller fordelt på 34 PostgreSQL-skjemaer — fagfolk, selskaper, skip, stillinger, omtaler og resten — med SMALLINT-oppslag og UUID v7-nøkler.

Tok plattformens grunnleggende tekniske valg

Tok plattformens grunnleggende beslutninger i ukene etter at repoet ble opprettet i november 2025 — lagdelingen, datatilgang med databasen først og kravet om null advarsler — og de står seg fortsatt ni måneder senere.

Bygde infrastruktur som fortsatt kjører år senere

Planla og implementerte ny infrastrukturfunksjonalitet for interne og eksterne systemer, og bygde løsninger så solide at de fortsatt kjører år senere med minimale endringer.

Bygde hele kjerneinfrastrukturen fra bunnen av

Designet og bygde, som en av de første ansatte, hele kjerneinfrastrukturen og de tilhørende prosessene fra bunnen av for en SaaS-oppstartsbedrift innen grønn energi, og la grunnlaget for rask vekst.

Offentlige tjenester på en liten server

Driftet selskapets offentlige tjenester på en vert med én kjerne og 512 MB — en git-forge, en webserver for sju domener, Tor, to servere for alternative protokoller og utestenging av inntrengere — med 464 MB brukbart minne som designbegrensning; backupautomatisering er en separat implementasjon.

Bygde en databasedrevet CV-generator

Bygde en databasedrevet generator for CV, referanser, portefølje og søknadsbrev i Python — 41 moduler, 10 580 linjer — som gjengir seks utdataformater fra én SQLite-kilde med 23 tabeller, satt sammen av en idempotent pipeline i 19 trinn.

Bygde inn en QR-enkoder på 522 linjer

Bygde inn en QR-enkoder — Reed-Solomon over GF(256), fast moduloppsett, åtte maskemønstre — på 522 linjer i stedet for å ta inn en tredje kjøretidsavhengighet, og verifiserte den ved å lese den ferdige matrisen tilbake med en uavhengig skrevet dekoder.

Bygde en native macOS-klient i Swift 6

Bygde en native meldingsklient for macOS i Swift 6 og SwiftUI — 11 141 linjer fordelt på 53 filer — over C-grensesnittet til en Rust-kjerne lenket inn som et statisk arkiv fra en låst revisjon.

En avgrenset hendelsesflyt i en macOS-app

Avgrenset hendelsesbuffere og logging etter en registrert macOS-hendelse med vekst på 41 MB i sekundet og 111 GB komprimerte sider på en maskin med 36 GB; en registrert full testkjøring reduserte kjernens logglinjer fra 610 996 til 1 411.

Tok i bruk streng samtidighet i Swift 6 over C

Tok i bruk fullstendig streng samtidighet i Swift 6 uten actors, og koblet en blokkerende C-hendelsesløkke til main actor gjennom én produsent, én konsument og én rekkefølge — etter å ha slått fast at én task per hendelse mister rekkefølgen grensesnittet er avhengig av.

Leter dere etter Plattform- og løsningsarkitektur?

Bruk de tilknyttede arbeidene til å vurdere relevant erfaring. Fortell oss hva dere skal bygge; vi kan diskutere omfang, fremgangsmåte og overlevering.

Ta kontakt Hele porteføljen

Avgrens oppgaven før bestilling

Beskriv arbeidsflyten, eksisterende grensesnitt og brukerne som trenger endringen. Vi kan definere en testbar leveranse og en overlevering som gjør teamet i stand til å vedlikeholde løsningen.

Se hva det koster Ta kontakt →

Gå gjennom kravene med oss

Fortell hva dere trenger innen «Plattform- og løsningsarkitektur», inkludert fristen og teamet som skal vedlikeholde løsningen. Vi kan drøfte en praktisk vei videre.

Ta kontakt

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

Ta med deg denne siden

Visittkort

Laget med kjærlighet av D.

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

Opphavsrett © 2025 – nå · Engineer ApS · Alle rettigheter forbeholdt.

Skrevet ut fra https://software.engineer.company/nb/services/platform-architecture/