Engineer

Portefølje

Ingeniørtjenester med eksempler på utført arbeid.

Engineer ApS ble registrert i 2025. Mesteparten av arbeidet som vises her, ble levert før det av gründeren vår inne i kundenes egne team; noe er levert av selskapet selv, og noe siden i ansettelsesforhold. De samme hendene og de samme standardene — vi sier hva som er hva i stedet for å viske det ut.

Les om arbeidet **i sin sammenheng** — beslutningene, leveransen og de beskrevne resultatene.

Se hele katalogen

Finn erfaring det er verdt å bruke

Bruk oversikten til å finne erfaring som er relevant for prosjektet deres.

Se tjenestene våre Ta kontakt →

Les beslutningene bak arbeidet

Eksemplene beskriver levert arbeid på tvers av tjenester. Les en gjennomgang for å se situasjonen og beslutningene.

Prosjekter Ta kontakt →

Disse navnene **går igjen i arbeidet nedenfor** — ingenting eksotisk som teamet deres må overta.

Python PostgreSQL TypeScript React GitHub Docker

Se alle teknologiene

Bla etter kategori

PROGRAMVAREUTVIKLING

Backend-utvikling Frontend-utvikling Fullstack-utvikling API-er og integrasjon Webutvikling Designsystemer og brukergrensesnitt UX/UI-design Testing og kvalitetssikring Ytelsesoptimalisering Python

Internasjonalisering

DATA & DATABASER

Databaser · PostgreSQL · SQL · Data engineering · Datapipelines (ETL/ELT) · Dataanalyse · Datastyring · Datavarehus · GIS / geodata

SKY, INFRASTRUKTUR & DEVOPS

Skytjenester · Containere (Docker/Kubernetes) · DevOps · Automatisering og CI/CD · Infrastruktur · Linux og servere · Systemadministrasjon · Nettverk og VPN · Overvåking og observerbarhet · Pålitelighet og sikkerhetskopier · Sikkerhet · Plattformarkitektur · Migrering og modernisering

RÅDGIVNING & LEDELSE

Teknisk ledelse · Teamledelse · Mentorarbeid og coaching · Smidig og Scrum · Prosjektledelse · Produkt og krav · Interessenter og rapportering · Dokumentasjon · Løsningsarkitektur · Merkevare og markedsføring

Alt nedenfor **er levert** — ikke foreslått, ikke planlagt.

Hver oppføring forteller hva arbeidet endret, og lenker til tjenestene den viser.

Det vi tilbyr Hva det koster →

{ Programvareutvikling }

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.

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.

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.

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.

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 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.

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 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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

Et kartprodukt, håndtert fra ende til annen

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

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.

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.

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.

{ Programvareutvikling }

Backend- og API-utvikling

Motoren bak appen eller nettstedet ditt

Delen brukerne aldri ser: hvor dataene ligger, hvilke regler som gjelder, og hvordan andre systemer får lov til å snakke med den.

Nådde den ubrukte halvdelen av kjernen

Tok i bruk den halvdelen av meldingskjernen applikasjonen aldri hadde brukt — overføring av sikkerhetskopier, forsvinnende meldinger, redigering og ny sending av meldinger, verifiserte invitasjoner, proxyer og krypteringspolicy — og kjørte hver test mot det ekte biblioteket uten mocks.

Skrev en parser som verifiserer hvert FFI-kall

Skrev en parser som leser den faktiske C-headeren på 7 308 linjer og verifiserer hvert kallsted, hver enum-konstant og at hver klasse som eier en peker, er final, etter at en håndskrevet plassholder-header lot kall til tre fjernede funksjoner kompilere, lenke og krasje.

Flyttet hver tekststreng inn i et innholdstre

Flyttet alle brukerrettede strenger ut av Python og inn i et innholdstre på 874 filer på tre språk, etter å ha funnet døde oversettelser ingen kunne se var døde, og en kontroll som i det stille vurderte en tredjedel av resultatene.

Aktiverte alle linterregler som feil

Slo på alle reglene i Python-linteren på feilnivå og arbeidet seg gjennom 1 815 funn ned til null, med en skriftlig begrunnelse for hvert av de få unntakene og to av dem sikret av en kontroll i stedet for en kommentar.

Holdt generatoren til 981 tester og 92 % dekning

Holdt generatoren til 981 testtilfeller med et minstekrav på 92 % grendekning og advarsler behandlet som feil, og verifiserte idempotens ved å kjøre hele byggepipelinen to ganger fra en tom fil og kreve at den andre kjøringen ikke endret noe.

Bygde arbeidsflater for rekruttering og sjøfolk

Bygde arbeidsflater for rekruttering og sjøfolks karriere; det registrerte øyeblikksbildet av implementasjonen omfatter 151 lagrede funksjoner i 48 tabeller for stillinger, søknader, sertifikater, rangutvikling og fartstid.

Bygde det sosiale laget: innlegg, feed og grupper

Bygde plattformens sosiale lag — innlegg, feed, grupper, nevninger, en følgergraf og et sammendrag av merkedager — på det samme funksjonsbaserte datalaget som resten av produktet.

Bygde en lojalitetslommebok, ligaer og merker

Bygde lojalitets- og omdømmesystemet — 67 funksjoner over en hovedbok på 31 tabeller, med ligaer, merker og en innløsningsbutikk — med radlås på saldoen for å lukke vinduet for dobbeltforbruk.

Bygde fail-closed hastighetsgrenser mot misbruk

Bygde misbruksvern som stenger ved feil — 22 Redis-baserte hastighetsbegrensere, Cloudflare Turnstile, idempotente forespørsler og en origin-lås — slik at plattformen avviser boter og flom i stedet for å stole på dem som kaller den.

Bygde egen feilovervåking og sporing

Bygde egen feilovervåking og OpenTelemetry-sporing i stedet for å kjøpe dem — rensing av nyttelast, oppdagelse av topper og regresjoner, symbolisering og et syntetisk hjerteslag — bak 11 operatørvisninger.

Flyttet tregt arbeid over i en River-jobbkø

Flyttet tregt arbeid ut av forespørselsbanen og over i en River-jobbkø — 15 worker-moduler, 8 planlagte oppgaver og 20 pg_cron-jobber — slik at en forespørsel returnerer mens arbeidet bak den fortsetter.

Bygde tilgangsrettigheter for Stripe og appkjøp

Bygde laget for betaling og tilgangsrettigheter — Stripe sammen med kjøp i appen hos Apple og Google — som styrer tilgangen til katalogen, søket og eksporten gjennom en tilgangsmodell på 11 tabeller som kontrolleres på serveren.

Bygde en testsuite for Go, SQL og nettleser

Bygde en lagdelt automatisert testsuite — 981 Go-tester, 543 spesifikasjoner for frontend og nettleser, 494 atferdstester i SQL — med mutasjonstesting, egenskapsbaserte tester og en kvalitetsport for universell utforming.

Bygde og vedlikeholdt Django-apper for IPTV

Utviklet og vedlikeholdt Django-webapplikasjoner for IPTV-plattformen, leverte nye funksjoner og forbedret ytelse og stabilitet.

Herdet applikasjonen med nonce-basert CSP

Herdet applikasjonen med nonce-basert CSP, HSTS, SameSite-informasjonskapsler, databaseroller etter prinsippet om minste privilegium og ny kontroll av tilgangsrettigheter på serversiden.

Implementerte en katalogstyrt deep merge

Implementerte en katalogstyrt deep merge av lagrede JSON-innstillinger, som forhindrer krasj ved manglende nøkler når skjemaet utvikler seg.

Bygde e-post med failover og leveringslogging

Bygde e-post som en plattformfunksjon — tre leverandører med failover, leveringswebhooks, logging av sending og levering, maler og kampanjer — bak en oppstartskontroll som ikke lar plattformen starte uten minst én leverandør.

Genererte API-kontrakten med databasen som kilde

Genererte API-kontrakten fra databasen og utover — OpenAPI, en typet TypeScript-klient på 44 076 linjer, 61 mock-handlere og grensene brukergrensesnittet håndhever — med en kontroll i hvert ledd som feiler ved avvik.

Utviklet et rapporteringssystem for dataanalyse

Utviklet et rapporteringssystem for dataanalyse og økte kvartalsinntektene fra programvaren med 400 % gjennom Python-baserte PDF-rapporter.

Ledet 30+ GIS-prosjekter fra utvikling til støtte

Ledet utvikling, produksjonssetting og støtte for over 30 GIS-prosjekter, med ekspertise i teknologiene PostgreSQL, Bash, Python, JavaScript, GDAL, ArcGIS, PostGIS og Mapbox.

Styrte fullstack-utviklingen av GIS-kart

Styrte fullstack-utviklingen av GIS-kart med ansvar for PostgreSQL, Mapbox, ReactJS og NodeJS, og leverte en integrert løsning.

Leverte 8 Power BI-prosjekter med veiledninger

Leverte 8 Power BI-prosjekter med grundige brukerveiledninger, og integrerte Microsoft Power BI-verktøy med et NodeJS-API og Python FastAPI for effektiv dataanalyse og visualisering.

Designet og driftet 6 ETL/ELT-pipelines

Designet, implementerte og administrerte 6 ETL/ELT-pipelines med Google BigQuery, MSSQL, PostgreSQL, Shell-skripting, PL/pgSQL og Transact-SQL, og integrerte data for effektiv behandling i et Python-API.

Tjenesten inngår i arbeid som står andre steder på siden:

Tok i bruk streng samtidighet i Swift 6 over C · Bygde inn en QR-enkoder på 522 linjer · Bygde en databasedrevet CV-generator · Tok plattformens grunnleggende tekniske valg · Designet et funksjonsbasert datalag i PostgreSQL · Migrerte API-et til Huma v2 med 100 % paritet · Bygde bytte av organisasjonskontekst for brukerne · Designet JSON rett fra Postgres til API-et · En maritim plattform med klare systemgrenser

{ Programvareutvikling }

Frontend-utvikling

Skjermbildene folk faktisk bruker

Grensesnitt som laster raskt, fungerer på mobilen og oppfører seg likt i alle nettlesere kundene dine måtte bruke.

Bygde et macOS-designsystem med 48 tokens

Bygde et designsystem med 48 tokens på plattformens eget glassmateriale og skrev linteren som avviser et magisk tall, en hardkodet skriftstørrelse eller en animasjon som ignorerer innstillingen for redusert bevegelse.

Ga alle 17 ikoner navn for skjermlesere

Ga hver kontroll med bare ikon i grensesnittet et navn for skjermlesere etter å ha funnet at send-knappen ble lest opp som «arrow up circle, button», og skrev linteren som krever etiketten innen åtte linjer fra ikonet.

Lintet selve kvalitetsporten på 10 242 linjer

Satte 10 242 linjer JavaScript i kvalitetsporten under en formaterer og en linter etter å ha slått fast at det var den største kodemengden i repoet og den eneste ingenting leste, og rettet 13 funn uten å undertrykke noen.

Kuttet stilarkpakken fra 87 KB til 31 KB

Kuttet stilarkpakken fra 87 KB til 31 KB ved å ta ut en base64-kodet skrift, og fjernet 756 KB fordelt på 22 filer som ble publisert ved hver produksjonssetting og som ingenting refererte til.

Bygde et utskriftsstilark på 637 linjer

Bygde et utskriftsstilark på 637 linjer mot fire dokumenterte atferder i gjengivelsesmotorer, blant dem innfelte overskrifter som ble skrevet ut hvitt på hvitt for enhver leser med nettleseren innstilt på mørkt.

Gjorde innstillingen for redusert bevegelse ærlig

Gjorde innstillingen for redusert bevegelse ærlig etter å ha funnet elleve selektorer som fortsatt animerte under den, fordi en universell transition-none-regel taper på spesifisitet mot enhver regel med en klasse.

Fant at det lyse temaet manglet et helt lag

Fant ut at det lyse temaet hadde manglet et helsides overleggslag helt siden det ble skrevet, ved å kreve at begge temaene maler hver lagdelte flate med like mange lag.

Lukket fargesystemet ved 28 farger

Lukket fargesystemet ved 28 dokumenterte farger med en linter som avviser en verdi som er malt, men udokumentert, dokumentert, men ikke malt, feilmålt, skrevet om som en literal, eller innenfor en perseptuell avstand på 0,02 fra en som allerede finnes.

Feil i universell utforming som kontrollverktøyet deres overser

Fant og rettet 11 feil i universell utforming som nettleseren meldte var i orden — åtte bunntekstlenker som fortsatt lå i tabulatorrekkefølgen bak pointer-events, en scroll-tidslinje som klippet kolofonen av tre sider, og en regelmotor som kjørte 70 av sine 105 regler.

Tilgjengelighetskontroller på 32 sider i begge temaer

Implementerte WCAG 2.2 Level AA-kontroller på 32 representative sider i begge fargetemaer, med en samsvarsoversikt, kontroll av ett AAA-kriterium og det delvise resultatet for kriterium 1.4.13 uttrykkelig angitt.

Byttet 27 skriftstørrelser mot en skala i 6 trinn

Erstattet 27 gjengitte skriftstørrelser, der de nærmeste naboene lå 0,6 % fra hverandre, med en Major Third-skala i seks trinn, og skrev linteren som avviser den tjueåttende.

Et sikkerhetsnett som ikke lenger koster en halvtime

Kuttet nettstedets nettleserdrevne kvalitetsport fra 1 636 sekunder til 615 ved å kjøre kontrollene med de lengste først gjennom en worker-pool begrenset til fire baner, etter å ha målt at alfabetisk rekkefølge kostet 320 sekunder mot 224.

Bygde et flerspråklig Hugo-nettsted på 3 språk

Bygde et flerspråklig statisk nettsted i Hugo med 108 malfiler, blant dem 53 partials, som publiserer hver side i fire representasjoner fra ett innholdstre på tre språk, og på nytt under sju fokuserte underdomener bygd fra det samme treet.

Bygde operatørenes backoffice for plattformen

Bygde back office for operatørene — rundt 40 administrasjonsruter og 128 komponenter for eierskapskrav, moderering, funksjonsflagg, cache og diagnostikk — slik at plattformen kan driftes uten databasetilgang.

Vedlikeholdt Hugo-nettsted, bedret produktets UI

Vedlikeholdt og videreutviklet det eldre statiske Hugo-nettstedet, og bidro samtidig med UI/UX-forbedringer til hovedproduktet for forvaltning av eiendeler.

Bygde et engelsk/dansk i18n-system

Etablerte et engelsk/dansk internasjonaliseringssystem med 12 027 meldinger per språkversjon fordelt på 458 navneromsfiler, med vokabular håndhevet av linter og et budsjett på 400 linjer per fil.

Designet produktets UI og UX fra ende til annen

Designet produktets UI og UX fra ende til annen — katalogrutenett, doble kort- og tabellvisninger, fortløpende validering av krav og brødsmulenavigasjon.

Designet «Liquid Glass»-designsystem for iOS 26

Designet et «Liquid Glass»-designsystem for iOS 26 og et kanonisk komponentregister håndhevet av lint-regler for å hindre at brukergrensesnittet spriker.

Leverte full støtte for Progressive Web App

Leverte full støtte for Progressive Web App — kan installeres og fungerer uten nettforbindelse — med caching under kjøring i Workbox via next-pwa.

Bygde Next.js 16-frontenden: SSR, SSG og CSR

Bygde Next.js 16-frontenden med bevisste strategier for SSR, SSG og CSR, og en gjenbrukbar fabrikk for forhåndshentede serversider som fjerner N+1-kaskaden av hentinger fra hver autentisert side som flyttes over på den.

Designet 30 nettsteder fra Photoshop til HTML

Designet 30 nettsteder og leverte unike og fleksible løsninger ved å konvertere Photoshop-design til HTML.

Utviklet 100 webapplikasjoner med HTML/HTML5

Utviklet 100 webapplikasjoner med HTML/HTML5, CSS/SCSS, Django, WordPress og Joomla, og sikret en variert tilstedeværelse på nett.

Bidro i designprosessene for brukergrensesnitt

Bidro i designprosessene for brukergrensesnitt og sikret intuitive, visuelt tiltalende og brukervennlige grensesnitt i prosjektene.

Presenterte 200 UI/UX-forbedringer

Presenterte 200 UI/UX-forbedringer for GIS-kartapplikasjonen og økte inntektene 10 ganger gjennom bedre programvarefunksjoner.

Tjenesten inngår i arbeid som står andre steder på siden:

En avgrenset hendelsesflyt i en macOS-app · Bygde en native macOS-klient i Swift 6 ·
Tok plattformens grunnleggende tekniske valg · Bygde bytte av organisasjonskontekst for brukerne ·
En maritim plattform med klare systemgrenser · Styrte fullstack-utviklingen av GIS-kart

{ Programvareutvikling }

Fullstack-produktutvikling

Et fungerende produkt, fra ende til annen

Ett team som tar en idé fra skisse til noe kundene dine kan logge inn på, i stedet for at den sendes videre mellom tre leverandører.

Bygde eierskapskrav og avgjørelse for selskaper

Bygde løsningen for eierskapskrav til selskaper fra ende til annen — en bruker gjør krav på et selskap, en administrator avgjør saken, og en godkjenning skriver om autorisasjonsgrafen som bestemmer hvem som får redigere hva.

Tjenesten inngår i arbeid som står andre steder på siden:

Bygde en native macOS-klient i Swift 6 · Bygde en databasedrevet CV-generator ·
Bygde et flerspråklig Hugo-nettsted på 3 språk · Bygde arbeidsflater for rekruttering og sjøfolk ·
Bygde det sosiale laget: innlegg, feed og grupper · Bygde operatørenes backoffice for plattformen ·
Bygde og vedlikeholdt Django-apper for IPTV · Tok plattformens grunnleggende tekniske valg ·
Bygde Next.js 16-frontenden: SSR, SSG og CSR · En maritim plattform med klare systemgrenser ·
Et kartprodukt, håndtert fra ende til annen · Utviklet 100 webapplikasjoner med HTML/HTML5 ·
Ledet 30+ GIS-prosjekter fra utvikling til støtte · Styrte fullstack-utviklingen av GIS-kart ·
Ledet en GIS-kartapp til 10 ganger mer omsetning

{ Programvareutvikling }

Utvikling av nettsteder og CMS

Et nettsted du kan oppdatere selv

Et nettsted som er raskt, lett å finne og kan redigeres av dine egne ansatte, uten en regning hver gang en formulering endres.

Publiserte et lesbart onion-speil på Tor

Publiserte nettstedet som et Tor onion-speil på en lesbar adresse som begynner med engineer, med nøkkelen generert utenfor verten slik at den aldri havnet i dette repoet eller på en bærbar PC, og lot det være den ene av fire inngangsdører som aldri telles.

Bygde den visuelle identiteten fra 2 tegninger

Bygde selskapets ikon- og favicon-sett fra to håndlagde tegninger, med alle avledede filer regenerert av skript og en kontroll som feiler når en avledet fil ble committet før kilden.

Bestod 106 av 106 PDF/UA-1-profilkontroller

Dokumenterte PDF/UA-1-validering av ni dokumenter med 106 av 106 bestått kontroller i den konfigurerte profilen for universell utforming, mens arkivvarianten feilet på én av 146 regler.

Rettet et sitemap med ett felles tidsstempel

Rettet et sitemap der 172 av 176 URL-er delte ett endringstidspunkt, ved å hente datoen fra git-historikken etter å ha slått fast at eksporten skriver om hver fil ved hver kjøring.

Erstattet 963 metadatablokker med én graf

Byttet ut 963 anonyme blokker med strukturerte data, som gjentok selskapet 1 671 ganger, mot én lenket graf av 16 typer utledet fra stabile opprinnelsesidentifikatorer.

Speilet nettstedet til Gemini og Gopher

Bygde Gemini- og Gopher-utdata fra samme nettstedskilde; resultatbeskrivelsen oppgir 777 dokumenter i hvert format på det stadiet og null endrede HTML-byte ved tillegget av formatene.

Genererte 495 sider fra en SQLite-eksport

Genererte 495 resultatsider på tre språk fra en skrivebeskyttet SQLite-eksport, med sideadressen skrevet som data, slik at en rettet setning ikke lenger flyttet siden og brøt lenken.

Koden deres på deres egen server

Satte selskapets egen git-forge på Soft Serve i produksjon, privat som standard, uten webpanel og med SSH-porten bundet til loopback bak en hoppvert, og gjorde landingssiden foran den til en byggeartefakt fra hovednettstedet i stedet for en kopi som vedlikeholdes for hånd.

Kombinerte webutvikling med SEO og innhold

Drev frem kundenes suksess på nett ved å kombinere skreddersydd utvikling og design av nettsteder med SEO, innholdsstrategi og tekstforfatting.

Administrerte 40 nettsteder på Apache og Nginx

Administrerte 40 nettsteder på Ubuntu Linux-servere for webhotell med Apache og Nginx, og sikret høy tilgjengelighet og ytelse.

Tjenesten inngår i arbeid som står andre steder på siden:

Kuttet stilarkpakken fra 87 KB til 31 KB · Bygde et utskriftsstilark på 637 linjer · Gjorde innstillingen for redusert bevegelse ærlig · Feil i universell utforming som kontrollverktøyet deres overser · Tilgjengelighetskontroller på 32 sider i begge temaer · Bygde et flerspråklig Hugo-nettsted på 3 språk · Vedlikeholdt Hugo-nettsted, bedret produktets UI · Bygde og vedlikeholdt Django-apper for IPTV · Leverte full støtte for Progressive Web App · Bygde Next.js 16-frontenden: SSR, SSG og CSR · Designet 30 nettsteder fra Photoshop til HTML · Utviklet 100 webapplikasjoner med HTML/HTML5

{ Programvareutvikling }

Klar for AI-agenter

Synlig når kundene spør en AI

Et nettsted eller en app som AI-assistentene kundene dine bruker, kan finne, lese og handle ut fra, kontrollert ved hver utgivelse i stedet for lovet én gang.

Drev utviklingen som 51 skriftlige initiativer

Drev utviklingen av nettstedet som 51 skriftlige initiativer fordelt på 54 plandokumenter, hvert med sine åpne avkrysningsbokser, sine målinger og beslutningene det avslo — blant dem seks av åtte funn om agentberedskap som ble avvist med begrunnelsen registrert.

Tjenesten inngår i arbeid som står andre steder på siden:

Rettet et sitemap med ett felles tidsstempel · Erstattet 963 metadatablokker med én graf · Tilgjengelighetskontroller på 32 sider i begge temaer · Bygde et flerspråklig Hugo-nettsted på 3 språk

{ Programvareutvikling }

Verktøy for AI-agenter og MCP-servere

La AI bruke programvaren du allerede har

Programvaren du allerede har, åpnet som verktøy en AI-agent kan kalle, uten at noe som allerede virker, må skrives om.

Tjenesten inngår i arbeid som står andre steder på siden:

Bygde operatørenes backoffice for plattformen · Flyttet tregt arbeid over i en River-jobbkø · Genererte API-kontrakten med databasen som kilde · Migrerte API-et til Huma v2 med 100 % paritet · Designet JSON rett fra Postgres til API-et

{ Programvareutvikling }

GIS- og geodataløsninger

Kart som svarer på spørsmål

Alt som skjer på et sted — fartøy, kjøretøy, tomter, sensorer — lagt på et kart du kan søke i, måle på og dele.

Ledet 30 vellykkede GIS-prosjekter til levering

Styrte gjennomføringen av 30 vellykkede GIS-prosjekter og viste lederskap i leveransen av innovative løsninger på tvers av fagfeltet.

Optimaliserte prognose- og investeringsstrategier

Optimaliserte prognose- og investeringsstrategier for 11 nettselskaper og økte driftseffektiviteten gjennom datadrevne GIS-løsninger.

Automatiserte leveransen av 20 GIS-datapipelines

Automatiserte leveransen av 20 GIS-datapipelines og ETL-prosesser for applikasjonsdata, og effektiviserte infrastrukturautomatisering og rapportering.

Automatiserte GIS-SaaS-utrulling i GitHub Actions

Automatiserte produksjonssettingen, databehandlingen og rapporteringssystemet for en GIS SaaS-applikasjon med GitHub Actions CI/CD, Python, Bash og SQL.

Drev 100 datavarehus: Postgres, MS SQL, BigQuery

Utformet, opprettet og administrerte 100 databaser for datavarehus i PostgreSQL, MS SQL og Google BigQuery, hovedsakelig med GIS- og tidsseriedata, og optimaliserte ytelse og skalerbarhet.

Publiserte 500 datarapporter om strøm og GIS

Publiserte 500 analyserapporter om strøm- og GIS-data, basert på grundige undersøkelser og feilsøking, for å sikre presis innsikt i geografiske stordata og tidsseriedata.

Løste 1 000 problemer i geografiske data

Løste 1 000 problemer i geografiske data og tidsseriedata med GDAL, ArcGIS, PostGIS, Mapbox, QGIS, SQL (PL/pgSQL, Transact-SQL) og Bash, og sikret behandling av stordata med høy kvalitet.

Gjorde GIS-kart 10 ganger raskere på PostgreSQL

Forbedret ytelsen i en geografisk kartapplikasjon 10 ganger gjennom et strategisk databaseskifte fra MSSQL til PostgreSQL, inkludert omleggingen av dataene, og optimaliserte samtidig behandlingen og datasikkerheten.

Gjorde en geodatapipeline 50 ganger raskere

Registrerte en forbedring på 50 ganger i en geografisk datapipeline i grunnleggerens arbeid hos Utiligize gjennom SQL- og datamodellendringer i PostgreSQL, MS SQL og Google Cloud BigQuery.

Tjenesten inngår i arbeid som står andre steder på siden:

Et kartprodukt, håndtert fra ende til annen · Ledet 30+ GIS-prosjekter fra utvikling til støtte · Styrte fullstack-utviklingen av GIS-kart · Ledet en GIS-kartapp til 10 ganger mer omsetning

{ Programvareutvikling }

UI/UX-design og designsystemer

Gjøre bruken selvforklarende

Oppsett, ordvalg og flyt lagt opp slik at folk fullfører det de kom for å gjøre, i stedet for å ringe kundestøtten din.

Skrev om feilmeldingene som skyldte på brukerne

Skrev om applikasjonens feilmeldinger i tråd med en skriftlig tonestandard etter at en avvist innlogging ga brukeren skylden for en skrivefeil når leverandøren egentlig krevde et appspesifikt passord, og dekket det med en test som navngir leverandøren.

Tjenesten inngår i arbeid som står andre steder på siden:

Bygde den visuelle identiteten fra 2 tegninger · Bygde et macOS-designsystem med 48 tokens · Ga alle 17 ikoner navn for skjermlesere · Bygde en native macOS-klient i Swift 6 · Bestod 106 av 106 PDF/UA-1-profilkontroller · Bygde et utskriftsstilark på 637 linjer · Gjorde innstillingen for redusert bevegelse ærlig · Fant at det lyse temaet manglet et helt lag ·

Lukket fargesystemet ved 28 farger · Feil i universell utforming som kontrollverktøyet deres overser · Tilgjengelighetskontroller på 32 sider i begge temaer · Byttet 27 skriftstørrelser mot en skala i 6 trinn · Kombinerte webutvikling med SEO og innhold · Vedlikeholdt Hugo-nettsted, bedret produktets UI · Designet produktets UI og UX fra ende til annen · Designet «Liquid Glass»-designsystem for iOS 26 · Designet 30 nettsteder fra Photoshop til HTML · Bidro i designprosessene for brukergrensesnitt · Presenterte 200 UI/UX-forbedringer

{ Programvareutvikling }

Internasjonalisering og lokalisering

På mer enn ett språk

Programvare som leses naturlig i alle markeder du selger i, også datoene, valutaene og måten navn sorteres på.

Bygde en tallkontroll på tvers av språk

Bygde en innholdskontroll på tvers av språk som feiler når en oversettelse mister et tall den engelske teksten oppgir, og når et språk bruker en notasjon som ikke hører hjemme i det — og fant to danske beskrivelser som manglet en måleverdi, og seksten ukrainske tekstbiter med engelske anførselstegn.

Tjenesten inngår i arbeid som står andre steder på siden:

Flyttet hver tekststreng inn i et innholdstre · Bygde en databasedrevet CV-generator · Speilet nettstedet til Gemini og Gopher · Genererte 495 sider fra en SQLite-eksport · Tilgjengelighetskontroller på 32 sider i begge temaer · Bygde et flerspråklig Hugo-nettsted på 3 språk · Bygde et engelsk/dansk i18n-system

{ Programvareutvikling }

Utvikling av skreddersydde AI-agenter

En AI-assistent som kan faget ditt

En agent koblet til systemene du allerede bruker, testet på saker dine egne fagfolk skriver, under oppsyn av folkene dine og aldri låst til én modell.

Tjenesten inngår i arbeid som står andre steder på siden:

Fikk seks innsamlere til å si hva de hoppet over · Skrev tester for selve kontrollverktøyene · Bygde en commit-port med 22 lintere · Bygde egen feilovervåking og sporing · Bygde repoets kontrollmotor med 268 kontroller

{ Data & databaser }

Databasedesign og datamodellering

Et fornuftig sted for dataene dine

En struktur som samsvarer med hvordan virksomheten faktisk fungerer, slik at spørsmålene du trenger svar på senere, fortsatt er mulige å stille.

Bygde egen analyse inne i PostgreSQL

Bygde egen produktanalyse i PostgreSQL — 47 funksjoner over partisjonerte hendelsestabeller som rydder seg selv — pseudonymisert bak et roterende salt og betinget av den besøkendes samtykke.

Tidsordnede nøkler for en voksende datamodell

Tok i bruk tidsordnede UUID v7-identifikatorer (PostgreSQL 18) som entitetsnøkler for å redusere fragmenteringen i B-tre-indekser og gjøre spørringene raskere.

Tjenesten inngår i arbeid som står andre steder på siden:

Bygde en databasedrevet CV-generator · Bygde arbeidsflater for rekruttering og sjøfolk · Bygde en lojalitetslommebok, ligaer og merker · Flyttet tregt arbeid over i en River-jobbkø · Bygde tilgangsrettigheter for Stripe og appkjøp · Modellerte 348 tabeller i 34 Postgres-skjemaer · Implementerte en katalogstyrt deep merge · Designet et funksjonsbasert datalag i PostgreSQL · Designet JSON rett fra Postgres til API-et · En maritim plattform med klare systemgrenser · Drev 100 datavarehus: Postgres, MS SQL, BigQuery · Designet og driftet 6 ETL/ELT-pipelines

{ Data & databaser }

Databaseadministrasjon (DBA)

Ta vare på databasen

Noen som har ansvaret for det som hele selskapet hviler på: Den er oppe, den er sikkerhetskopiert, og den forfaller ikke i det stille.

1 022 migreringer bak en port for skjemaavvik

Holdt skjemaet ærlig gjennom 1 022 migreringer med en CI-port som bygger databasen begge veier — en ny installasjon, og en installasjon pluss alle migreringer — og feiler når de to ikke stemmer overens.

La sikkerhetskopi og katastrofeberedskap i kode

Konfigurerte lagringstiden for databasens sikkerhetskopier som infrastruktur som kode, gjennomgikk deretter gjenopprettingsberedskapen og dokumenterte fremgangsmåten for gjenoppretting — og navnga de gjenværende hullene i stedet for å la dem bli oppdaget under en hendelse.

Utviklet 600 ETL/ELT-pipelines i PL/pgSQL

Utviklet 600 PL/pgSQL-baserte ETL/ELT-pipelines for å effektivisere komplekse arbeidsflyter for databehandling på tvers av flere utviklings- og produksjonsmiljøer i PostgreSQL.

Automatiserte databehandling i Python og PL/pgSQL

Automatiserte databehandlingsoppgaver med Shell-skripting, PL/pgSQL, Python og Transact-SQL, og økte produktiviteten og effektiviteten.

Automatiserte 100 kritiske sikkerhetskopier

Automatiserte 100 kritiske sikkerhetskopier av data med Barman, Google Cloud, Bash og Python, og sikret dataintegriteten på tvers av databasene.

Økte PostgreSQL-ytelsen 10 ganger

Økte PostgreSQL-ytelsen 10 ganger gjennom strategisk indeksering, partisjonering og spørringsoptimalisering, og gjorde databasen mer effektiv for bruker- og tenantdata, geodata og elektriske tidsseriedata.

Skrev 1 000 RBAC-regler for å sikre Linux-servere

Styrket datasikkerheten ved å innføre 1 000 RBAC-regler for utviklere, applikasjonsinstanser, PostgreSQL, MS SQL og andre Linux-servere, og forhindret uautorisert tilgang; dokumentert med Ansible-automatisering.

Drev 10 PostgreSQL- og MS SQL-servere på Ubuntu

Designet, satte i produksjon og vedlikeholdt 10 PostgreSQL- og MS SQL-servere på Ubuntu Linux-VPS-er, og sikret optimal serverytelse og pålitelighet.

Tjenesten inngår i arbeid som står andre steder på siden:

Drev 100 datavarehus: Postgres, MS SQL, BigQuery

{ Data & databaser }

Optimalisering av databaseytelse

Gjøre de trege delene raske

Finne spørringene og de manglende indeksene bak et system som har blitt tyngre for hvert år, og rette dem som betyr noe.

30 dagers strømdata på 180 ms

Utviklet varianter for timesaggregering av strømforbruk i Python, SQL og Bash + Jq; Python-varianten behandlet ett JSONL-datasett for 30 dager på 180 ms i ytelsestesten.

Tjenesten inngår i arbeid som står andre steder på siden:

Tidsordnede nøkler for en voksende datamodell · Økte PostgreSQL-ytelsen 10 ganger · Drev 100 datavarehus: Postgres, MS SQL, BigQuery · Gjorde GIS-kart 10 ganger raskere på PostgreSQL · Gjorde en geodatapipeline 50 ganger raskere

{ Data & databaser }

Databasemigrering og modernisering

Bort fra det gamle systemet

Data hentet ut av noe som er i ferd med å gå ut på dato, og flyttet over til noe som fortsatt støttes, mens virksomheten går som vanlig.

Tjenesten inngår i arbeid som står andre steder på siden:

Flyttet hver tekststreng inn i et innholdstre · 1 022 migreringer bak en port for skjemaavvik · Gjorde GIS-kart 10 ganger raskere på PostgreSQL

{ Data & databaser }

Utvikling av datapipelines (ETL/ELT)

Dataene dine fra A til B av seg selv

Tall som kommer fram dit de trengs, til faste tider,
uten at noen må kopiere et regneark ved månedsslutt.

Sparte 4 000 timer med GitHub og GitLab CI/CD

Effektiviserte prosessene for dataanalyse og programvareutvikling og sparte 4 000 timer ved å innføre CI/CD-praksis med GitHub, GitLab, Bash og Python.

Tjenesten inngår i arbeid som står andre steder på siden:

Genererte 495 sider fra en SQLite-eksport · Bygde egen analyse inne i PostgreSQL · Et kartprodukt, håndtert fra ende til annen · Utviklet 600 ETL/ELT-pipelines i PL/pgSQL · Automatiserte databehandling i Python og PL/pgSQL · Ledet 30+ GIS-prosjekter fra utvikling til støtte · Automatiserte leveransen av 20 GIS-datapipelines · Automatiserte GIS-SaaS-utrulling i GitHub Actions · Designet og driftet 6 ETL/ELT-pipelines · Løste 1 000 problemer i geografiske data · Gjorde en geodatapipeline 50 ganger raskere · 30 dagers strømdata på 180 ms · Én infrastrukturplan for 14 avdelinger

{ Data & databaser }

Design av datavarehus

Ett sted der alle tallene stemmer overens

Tall fra salg, drift og økonomi ført sammen, slik at to avdelinger
slutter å komme med to ulike svar til det samme møtet.

Tjenesten inngår i arbeid som står andre steder på siden:

Drev 100 datavarehus: Postgres, MS SQL, BigQuery · Designet og driftet 6 ETL/ELT-pipelines · Gjorde en geodatapipeline 50 ganger raskere

{ Data & databaser }

Dataanalyse og BI-dashbord

Se hva som skjer

Dashbord og rapporter som viser de få tallene en beslutning faktisk hviler på, oppdatert uten at noen må be om det.

Designet et tidsstyrings- og rapportsystem

Designet et system for tidsstyring og rapportering som var i produksjon i årevis uten vesentlige endringer.

Leverte analyser og jevnlige fremdriftsrapporter

Leverte analyser og jevnlige fremdriftsrapporter til administrerende direktør, og omsatte tekniske KPI-er og leveransestatus til beslutninger.

Bygde selskapets første observerbarhetsdashbord

Utviklet og lanserte selskapets første dashbord for observerbarhet, med sanntidsinnsikt i systemytelsen og datavisualisering på den store TV-skjermen på kontoret.

Merkevarekommunikasjon ut fra publikums reaksjoner

Brukte markedsundersøkelser og tilbakemeldinger fra publikum til merkevarekommunikasjonen og det oppsøkende arbeidet til Engineer ApS.

Tjenesten inngår i arbeid som står andre steder på siden:

Bygde egen analyse inne i PostgreSQL · Utviklet et rapporteringssystem for dataanalyse · Sparte 4 000 timer med GitHub og GitLab CI/CD · Optimaliserte prognose- og investeringsstrategier · Publiserte 500 datarapporter om strøm og GIS · Leverte 8 Power BI-prosjekter med veiledninger · 30 dagers strømdata på 180 ms

{ Data & databaser }

Datastyring og datakvalitet

Kunne stole på tallene

Regler og kontroller som fanger opp dårlige data på vei inn, og et klart svar på hvem som eier hvilket tall og hvor det kommer fra.

Målte 13 adresser uten sporing

Målte tretten adresser ut fra webserverens egen tilgangslogg alene — ingen informasjonskapsel, ikke noe sporingsverktøy og ingen tredjepartstjeneste — maskerte klientnettverket idet hver linje skrives, og samlet resultatet i et permanent arkiv med 21 daglige tellinger og 20 månedlige fasetter.

Kjenn lisensen til alt dere leverer

Reviderte 644 Rust-crates for lisenskompatibilitet ved hvert bygg, og beviste at kontrollen slår ut ved å skrive om lisensen til én crate og ved å flytte den låste kjernerevisjonen uten å regenerere.

Skrev tester for selve kontrollverktøyene

Skrev tester for selve kontrollene etter å ha slått fast at en kontroll som bare får ren inndata, en dag vil melde rent fordi den ikke leste noe — plantet en stavefeil for å bekrefte at stavekontrollen finner den, og hentet et id-område fra databasen i stedet for fra et tall i testen.

Skrev en avgrensningsregel etter tilgangslekkasje

Skrev en avgrensningsregel inn i repoet etter at en omstrukturering hadde ført et annet selskaps inventar, brannmuråpninger og tekst inn i det — og holdt de karantenesatte restene under skanneren for hemmeligheter i stedet for å unnta dem.

Samlet inn og analyserte forretningskrav

Samlet inn og analyserte forretningskrav og omsatte dem til gjennomførbare funksjoner og brukerhistorier i tråd med standardene for datastyring.

Tjenesten inngår i arbeid som står andre steder på siden:

Bygde en tallkontroll på tvers av språk · Flyttet hver tekststreng inn i et innholdstre · Erstattet 963 metadatablokker med én graf · Bygde egen analyse inne i PostgreSQL · Modellerte 348 tabeller i 34 Postgres-skjemaer · Implementerte en katalogstyrt deep merge · Designet et funksjonsbasert datalag i PostgreSQL · Utviklet 600 ETL/ELT-pipelines i PL/pgSQL · Publiserte 500 datarapporter om strøm og GIS · Løste 1 000 problemer i geografiske data · Én infrastrukturplan for 14 avdelinger

{ Data & databaser }

Klargjøring av data for AI

Data en AI faktisk kan bruke

Dataene dine samlet der AI-agenter kan nå dem, med én versjon av hver kunde, uten at noen får se mer enn de skal, og med hvert svar sporet tilbake til kilden.

Tjenesten inngår i arbeid som står andre steder på siden:

Erstattet 963 metadatablokker med én graf · Genererte 495 sider fra en SQLite-eksport · 1 022 migreringer bak en port for skjemaavvik · Modellerte 348 tabeller i 34 Postgres-skjemaer · Designet og driftet 6 ETL/ELT-pipelines · Gjorde en geodatapipeline 50 ganger raskere

{ Sky, infrastruktur & DevOps }

Skyinfrastruktur og migrering

Der programvaren kjører

Servere dimensjonert etter det du faktisk bruker, flyttet inn i skyen eller ut av den igjen, med månedsregningen regnet ut på forhånd.

Klargjorde en server nummer to fra kode

Automatiserte klargjøringen av en server nummer to hos en annen skyleverandør, der brannmuren opprettes før maskinen slik at den blir født bak en, og begge leverandørenes brannmurer er skrevet direkte mot deres REST-API-er for å slippe en collection fra tredjepart.

Kjørte containere på Podman og Quadlet

Satte et containerlag på Podman og Quadlet under systemd i produksjon i stedet for Docker, fordi Dockers standardpublisering av porter på bridge-nettverk kan omgå UFW-regler — og ga produksjonssettinger en uprivilegert bruker med én fast kommando i stedet for root.

Infrastruktur beskrevet i et kodelager

Bygde selskapets egen infrastruktur som 19 Ansible-playbooks og 34 roller på 12 065 linjer YAML, som konvergerer en vert i drift mot en deklarert tilstand der hver play er idempotent.

Hadde ansvar for Azure-utrulling av plattformen

Hadde ansvaret for produksjonssettingen av plattformen til Azure fra ende til annen, og styrte utgivelser gjennom utviklings-, staging- og produksjonsmiljøene.

Klargjorde Azure-infrastruktur som kode med Bicep

Klargjorde Azure-infrastruktur som kode med Bicep — Container Apps, PostgreSQL Flexible Server, Front Door/WAF og nettverk — i utviklings-, staging- og produksjonsmiljøene.

Driftet 20 Docker-apper på Google Cloud og AWS

Satte i produksjon og vedlikeholdt 20 containeriserte Docker-applikasjoner, feilsøkte med Podman og Kubernetes og administrerte R-baserte applikasjoner på Google Cloud og AWS.

Tjenesten inngår i arbeid som står andre steder på siden:

Offentlige tjenester på en liten server · Bygde hele kjerneinfrastrukturen fra bunnen av · Forenklet infrastruktur utenfor AWS

{ Sky, infrastruktur & DevOps }

Infrastruktur som kode

Servere du kan bygge opp igjen fra versjonsstyrt konfigurasjon

Hele oppsettet skrevet ned som filer, slik at det kan gjenskapes nøyaktig og aldri bare finnes i hodet på én person.

Prøvekjørte CI-hooken i forgen og fant 2 feil

Prøvekjørte CI-hooken på forge-siden og fant to feil som ikke kunne oppdages ved å lese filen: en reserveløsning som la et uoppløselig argument på hookens inndata, og gits egen miljøvariabel som fulgte kvalitetsporten inn i utsjekkingen og gjorde 19 tester røde.

Delte opp hver driftshemmelighet per vert

Delte opp de fire driftshemmelighetene per vert etter å ha slått fast at to verter som deler én dødmannsknapp, varsler mindre enn to knapper, ikke mer, og at en felles passordfrase for sikkerhetskopier kan gi tilgang til begge vertenes kopier, selv i separate repoer, hvis noen får tak i den og kan lese de krypterte dataene og nøkkelfilene i begge repoene.

Strammet inn systemd-sandkassen for hver enhet

Reduserte eksponeringen i systemd-sandkassen for hver enhet plattformen installerer — tjenesten for dødmannsknappen fra 9,6 UNSAFE til 1,5, en internetteksponert git-forge fra 8,3 EXPOSED til 1,5 — og la til en parserkontroll ved konvergering etter å ha funnet et feilstavet direktiv som ble ignorert i det stille i tre enhetsmaler.

Avstemte en DNS-sone med 20 poster som kode

Avstemte en DNS-sone med 20 poster deklarativt mot Cloudflare-API-et med egne inngangspunkter for revisjon og BIND-eksport, og slo CDN-proxyen av igjen av personvernhensyn etter å ha bygd den.

La til ett sekunds preflight for hemmeligheter

La til en preflight-play som kjører den samme koden som konvergeringen mot operatørens lokale hemmeligheter på omtrent ett sekund, etter at en halvveis gjennomført produksjonskjøring stoppet på den niende oppgaven med swap-innstillingene allerede skrevet til verten i drift.

Fikk Ansibles check-modus til å si sannheten

Fikk check-modus til å fortelle sannheten på tvers av hele plattformen etter å ha funnet seks sonder som avgjorde ut fra en verdi verten aldri hadde gitt, fordi Ansibles command-modul melder suksess under --check og samtidig hopper helt over kommandoen.

Bygde vertsovervåking med dødmannsknapp

Bygde overvåking med dødmannsknapp som bare sender ping så lenge minne og disk er i orden, slik at en svekket vert utløser et varsel ved å bli stille — og fanget opp seks variabelnavn som sa «free» der kontrollen korrekt målte «available», en størrelsesorden fra hverandre på en maskin med 464 MB.

Backupautomatisering med en tydelig gjenopprettingsmetode

Implementerte restic-backupautomatisering med lagringsregler, integritetskontroll og en arbeidsflyt for månedlig gjenopprettingstest; dokumenterte aktivering av lagring og kontroll av en virkelig gjenoppretting som separate driftsoppgaver.

Herdet SSH til 24 validerte direktiver

Herdet SSH til 24 kontrollerte direktiver med validering i tre trinn — kandidatfilen, den sammensatte konfigurasjonen og så daemonens egen tilbakelesing — etter at tilbakelesingen avslørte at den kjørende serveren i det stille overstyrte to av de tjuefire.

Tjenesten inngår i arbeid som står andre steder på siden:

Publiserte et lesbart onion-speil på Tor · Klargjorde en server nummer to fra kode · Koden deres på deres egen server · Offentlige tjenester på en liten server · Infrastruktur beskrevet i et kodelager · La sikkerhetskopi og katastrofeberedskap i kode · Klargjorde Azure-infrastruktur som kode med Bicep · Skrev 1 000 RBAC-regler for å sikre Linux-servere

{ Sky, infrastruktur & DevOps }

DevOps og CI/CD-automatisering

Endringer ut i produksjon uten drama

Testet, utgitt og mulig å reversere, slik at endringer når kundene en tirsdag ettermiddag i stedet for sent en fredagskveld.

Holdt 4 repoer til én historikkstandard

Holdt fire repoer til én historikkstandard — konvensjonell, uten emojier og uten attribusjonslinjer, håndhevet av en hook for commit-meldinger — sammen med 48 instruksjonsdokumenter som styrer hvordan arbeidet gjøres.

Fikk testsuiten ned i 8,9 sekunder

Tok testsuiten fra seks tester over en grense på seksti sekunder til 135 bestått på 8,9 sekunder ved å profilere hovedtråden og fjerne de to kallene den satt fast i under 3 989 av 4 017 prøver.

Fant at hookene og porten var uenige

Fant ut at commit-hookene og kvalitetsporten kjørte ulike kontroller mens et dokument lovet at de var de samme, ved å sammenligne de to listene i en test — de fem som bare noen gang ble kjørt manuelt, var de som leste CV-teksten.

Bygde en commit-port med 22 lintere

Bygde en commit-port med 22 enlinjes lintere pluss fem som fortjener et avsnitt, uten advarselsnivå og uten tillatte undertrykkinger i koden, som dekker HTML, CSS, JavaScript, Python, YAML, Markdown, shell, lenker, staving, hemmeligheter og typografi.

Drev et løpende program for sikkerhetsskanning

Etablerte et kontinuerlig sikkerhetsprogram — kodeskanning, DAST, kontroll av avhengigheter og sårbarheter, SBOM-generering, skanning etter hemmeligheter og SHA-låste actions — sammen med 21 skriftlige sikkerhetsrevisjoner.

Bygde repoets kontrollmotor med 268 kontroller

Bygde repoets kontrollmotor — 268 registrerte commit-kontroller, 277 lint-regler og 15 egne ESLint-regler — pluss 146 tester av selve kontrollene, slik at det er bygget, ikke gjennomgangen, som holder standarden.

Automatiserte samarbeid i teamet

Automatiserte samarbeid i teamet, passordhåndtering, oppgave- og tidsstyring, og bygde et halvautomatisk system for å vise frem prosjekter, noe som økte teamets produktivitet.

Satte et krav om null advarsler på seks språk

Satte en kvalitetsstandard med null advarsler for seks språk — Go, TypeScript, SQL, Python, Shell og Markdown — håndhevet av pre-commit-hooks.

Skrev 578 automatiseringsmål i go-task

Skrev 578 go-task-automatiseringsmål som dekker native-, Docker- og HTTPS-utviklingsmoduser, linting, testing, database og produksjonssetting.

Bygde CI/CD-pipelines i GitHub Actions

Bygde CI/CD-pipelines i GitHub Actions med et distroless-produksjonsimage for frontend og promotering gjennom flere miljøer.

Effektiviserte CI/CD og sparte 4 000 timer

Effektiviserte CI/CD-prosessene og sparte 4 000 timer ved å innføre automatisering i pipelines for programvareutvikling.

Automatiserte SSL/TLS for 100 Docker-apper

Automatiserte opprettelsen av SSL/TLS-sertifikater for 100 Docker-applikasjoner og sikret trygge forbindelser på tvers av Ubuntu Linux-verter.

Reduserte driftsrisiko med Grafana og Prometheus

Reduserte driftsrisikoen ved å innføre et overvåkingsdashbord med Grafana og Prometheus, og forbedret systemets pålitelighet.

Tjenesten inngår i arbeid som står andre steder på siden:

Kjenn lisensen til alt dere leverer · Nådde den ubrukte halvdelen av kjernen ·

Skrev en parser som verifiserer hvert FFI-kall · Prøvekjørte CI-hooken i forgen og fant 2 feil ·

Skrev tester for selve kontrollverktøyene · Aktiverte alle linterregler som feil ·

Holdt generatoren til 981 tester og 92 % dekning · Lintet selve kvalitetsporten på 10 242 linjer · Rettet et sitemap med ett felles tidsstempel · Speilet nettstedet til Gemini og Gopher · Et sikkerhetsnett som ikke lenger koster en halvtime · Kjørte containere på Podman og Quadlet · Koden deres på deres egen server · La til ett sekunds preflight for hemmeligheter · Fikk Ansibles check-modus til å si sannheten · Infrastruktur beskrevet i et kodelager · 1 022 migreringer bak en port for skjemaavvik · Bygde egen feilovervåking og sporing · Bygde en testsuite for Go, SQL og nettleser · Bygde hele kjerneinfrastrukturen fra bunnen av · Hadde ansvar for Azure-utrulling av plattformen · Genererte API-kontrakten med databasen som kilde · Forenklet infrastruktur utenfor AWS · Automatiserte databehandling i Python og PL/pgSQL · Sparte 4 000 timer med GitHub og GitLab CI/CD · Fornyet interne prosesser og sparte 8 000 timer · Driftet 20 Docker-apper på Google Cloud og AWS · Automatiserte 100 kritiske sikkerhetskopier · Automatiserte leveransen av 20 GIS-datapipelines · Automatiserte GIS-SaaS-utrulling i GitHub Actions

{ Sky, infrastruktur & DevOps }

Containerisering og orkestrering

Programvare som kjører likt overalt

Pakket slik at den oppfører seg helt likt på en bærbar PC, på en testserver og i produksjon, og skalerer opp når en travel uke kommer.

Tjenesten inngår i arbeid som står andre steder på siden:

Kjørte containere på Podman og Quadlet · Strammet inn systemd-sandkassen for hver enhet · Bygde CI/CD-pipelines i GitHub Actions · Automatiserte SSL/TLS for 100 Docker-apper · Driftet 20 Docker-apper på Google Cloud og AWS

{ Sky, infrastruktur & DevOps }

Pålitelighet og overvåking

Vite det før kundene dine gjør det

Overvåking og varsler som sier fra mens en feil fortsatt er liten, og tallene som viser hvor ofte den holder seg liten.

Fikk seks innsamlere til å si hva de hoppet over

Fikk seks rapporteringsinnsamlere til å oppgi den delen av inndataene de aldri leste — filer som forble lukket, sonder som ikke var installert, journallinjer som ikke ble gjenkjent — og tilpasset terskelverdier til 116 målinger, etter å ha slått fast at en blind innsamling og en vert der alt er i orden, gir den samme siden.

Bygde 7 skrivebeskyttede rapportroller for verter

Bygde sju skrivebeskyttede rapporteringsroller som gjengir en vert i drift som Markdown — fakta, tilgang, git, metrikker, trafikk, sikkerhet og leverandørens inventar — etter en regel om at ingen tall skrives ut som kjøringen ikke har målt.

Beviste at utestengingen virker ved hver kjøring

Beviste at utestengingen virker fra ende til annen ved hver herdingskjøring ved å utestenge en reservert testadresse, lese den resulterende kjerneregelen og oppheve utestengingen i en garantert oppryddingsblokk, slik at en jail som slutter å virke, får kjøringen til å feile i stedet for å melde at alt er i orden.

Tre låser på døren, ingen av dem låst

Fant og rettet tre SSH-vern mot brute force som aldri hadde virket: en jail for utestenging som overvåket port 22 mens daemonen lyttet på 1986, en hastighetsgrense skygget av en bredere regel over den, og en utestengingshandling der binærfilen aldri ble funnet, slik at ingen utestenging noen gang hadde trådt i kraft.

Holdt IPTV-strømmesignaler i gang uten stopp

Sikret uavbrutt levering av IPTV-strømmesignaler mellom leverandører og kunder, og overvåket og vedlikeholdt strømmenettverket og IP-telefonien døgnet rundt.

Ga 24/7-støtte for ca. 1 000 IPTV/OTT-servere

Ga infrastrukturstøtte døgnet rundt, 24/7, for en IPTV/OTT-strømmeplattform og administrerte ca. 1 000 servere i tillegg til kundeeide systemer for globale kunder i Kina, USA og Tyskland.

Tjenesten inngår i arbeid som står andre steder på siden:

Målte 13 adresser uten sporing · En avgrenset hendelsesflyt i en macOS-app ·

Delte opp hver driftshemmelighet per vert · Fikk Ansibles check-modus til å si sannheten ·

Bygde vertsovervåking med dødmannsknapp ·

Backupautomatisering med en tydelig gjenopprettingsmetode ·

Bygde fail-closed hastighetsgrenser mot misbruk · Bygde egen feilovervåking og sporing ·

Flyttet tregt arbeid over i en River-jobbkø · Bygde infrastruktur som fortsatt kjører år senere ·

Hadde ansvar for Azure-utrulling av plattformen · Bygde e-post med failover og leveringslogging · Et kartprodukt, håndtert fra ende til annen · Administrerte 40 nettsteder på Apache og Nginx · Reduserte driftsrisiko med Grafana og Prometheus · Drev 10 PostgreSQL- og MS SQL-servere på Ubuntu · Bygde selskapets første observerbarhetsdashbord

{ Sky, infrastruktur & DevOps }

Sikkerhetskopiering og gjenoppretting

Alt tilbake etter en dårlig dag

Sikkerhetskopier som er testet i stedet for tatt for gitt, og en skriftlig plan for morgenen da en server ikke kommer opp igjen.

Administrerte 30 Ubuntu Linux-VPS-instanser

Administrerte 30 Ubuntu Linux-VPS-instanser, innførte strategier for katastrofegjenoppretting og sikret optimale nettverkskonfigurasjoner.

Tjenesten inngår i arbeid som står andre steder på siden:

Delte opp hver driftshemmelighet per vert · Backupautomatisering med en tydelig gjenopprettingsmetode · La sikkerhetskopi og katastrofeberedskap i kode · Automatiserte 100 kritiske sikkerhetskopier

{ Sky, infrastruktur & DevOps }

Systemadministrasjon

Holde hjulene i gang

Daglig ettersyn av servere, e-post, kontoer og oppdateringer, slik at det alle er avhengige av, forblir kjedelig.

Verifiserte brannmurregler etter plassering

Verifiserte brannmurregler etter plassering i stedet for tilstedeværelse ved å lese den nummererte regellisten og den aktive pakkefilterkjeden, fordi en regel som finnes, ikke er det samme som en regel en pakke faktisk når.

Integrerte en passordbehandler i hele selskapet

Integrerte et passordhåndteringssystem for hele selskapet, styrket sikkerheten og effektiviserte tilgangskontrollen.

Administrerte 100 Bare Bone-servere

Administrerte 100 Bare Bone-servere, fysiske nettverk og IP-telefonisystemer, og sikret en robust infrastruktur for selskapets vekst.

Konfigurerte og installerte 1 000 Wi-Fi-rutere

Konfigurerte og installerte 1 000 Wi-Fi-rutere og forbedret nettverkstilgangen og ytelsen for kundene.

Administrerte nettverksinfrastrukturen

Administrerte nettverksinfrastrukturen for over 1 000 servere og sikret optimal produksjonssetting, sikkerhet og feilsøking av systemene.

Administrerte og feilsøkte 8 VPN-forbindelser

Administrerte og feilsøkte 8 VPN-forbindelser med WireGuard og IPsec, og sikret trygg kommunikasjon på tvers av Google Cloud og Linux-systemer.

Ledet tilgangsstyring på tvers av M365 og SSO

Forhindret sikkerhetsbrudd ved å lede tiltak for tilgangsstyring med M365, 1Password, Red Hat SSO og OKTA SSO.

Tjenesten inngår i arbeid som står andre steder på siden:

Publiserte et lesbart onion-speil på Tor · Fikk seks innsamlere til å si hva de hoppet over · Prøvekjørte CI-hooken i forgen og fant 2 feil · Koden deres på deres egen server · Bygde 7 skrivebeskyttede rapportroller for verter · Strammet inn systemd-sandkassen for hver enhet · Avstemte en DNS-sone med 20 poster som kode · Bygde vertsovervåking med dødmannsknapp · Backupautomatisering med en tydelig gjenopprettingsmetode · Beviste at utestengingen virker ved hver kjøring · Herdet SSH til 24 validerte direktiver · Offentlige tjenester på en liten server · Infrastruktur beskrevet i et kodelager · Bygde hele kjerneinfrastrukturen fra bunnen av · Bygde infrastruktur som fortsatt kjører år senere · Holdt IPTV-strømmesignaler i gang uten stopp · Ga 24/7-støtte for ca. 1 000 IPTV/OTT-servere · Administrerte 40 nettsteder på Apache og Nginx · Administrerte 30 Ubuntu Linux-VPS-instanser · Skrev 1 000 RBAC-regler for å sikre Linux-servere · Drev 10 PostgreSQL- og MS SQL-servere på Ubuntu

{ Sky, infrastruktur & DevOps }

Nettverk og VPN-oppsett

Koble sammen kontorene og menneskene dine

Wi-Fi, brannmurer og fjerntilgang satt opp slik at de ansatte kan jobbe hvor som helst, og bare de rette personene slipper inn.

Tjenesten inngår i arbeid som står andre steder på siden:

Klargjorde en server nummer to fra kode · Avstemte en DNS-sone med 20 poster som kode · Verifiserte brannmurregler etter plassering · Tre låser på døren, ingen av dem låst · Holdt IPTV-strømmesignaler i gang uten stopp · Klargjorde Azure-infrastruktur som kode med Bicep · Administrerte 100 Bare Bone-servere · Konfigurerte og installerte 1 000 Wi-Fi-rutere · Administrerte nettverksinfrastrukturen · Administrerte og feilsøkte 8 VPN-forbindelser · Administrerte 30 Ubuntu Linux-VPS-instanser

{ Sky, infrastruktur & DevOps }

Sikkerhet og tilgangsstyring

Bestemme hvem som får tilgang til hva

Kontoer, passord og tilganger ordnet slik at folk har nøyaktig den tilgangen jobben krever, og de som slutter, mister den samme dag.

Forbedret kommunikasjonen og samarbeidet i teamet

Forbedret kommunikasjonen og samarbeidet i teamet ved å innføre Slack, Mattermost, 1Password og Jira, og sparte 8 000 arbeidstimer.

Tjenesten inngår i arbeid som står andre steder på siden:

Publiserte et lesbart onion-speil på Tor · Målte 13 adresser uten sporing · Kjenn lisensen til alt dere leverer · Nådde den ubrukte halvdelen av kjernen · Delte opp hver driftshemmelighet per vert · Skrev en avgrensningsregel etter tilgangslekkasje · Klargjorde en server nummer to fra kode · Kjørte containere på Podman og Quadlet · Strammet inn systemd-sandkassen for hver enhet · Avstemte en DNS-sone med 20 poster som kode · La til ett sekunds preflight for hemmeligheter · Verifiserte brannmurregler etter plassering · Beviste at utestengingen virker ved hver kjøring · Herdet SSH til 24 validerte direktiver ·

Tre låser på døren, ingen av dem låst · Drev et løpende program for sikkerhetsskanning · Bygde eierskapskrav og avgjørelse for selskaper · Bygde fail-closed hastighetsgrenser mot misbruk · Bygde tilgangsrettigheter for Stripe og appkjøp · Integrerte en passordbehandler i hele selskapet · Automatiserte samarbeid i teamet · Herdet applikasjonen med nonce-basert CSP · Klargjorde Azure-infrastruktur som kode med Bicep · Bygde e-post med failover og leveringslogging · Bygde bytte av organisasjonskontekst for brukerne · Automatiserte SSL/TLS for 100 Docker-apper · Administrerte nettverksinfrastrukturen · Administrerte og feilsøkte 8 VPN-forbindelser · Ledet tilgangsstyring på tvers av M365 og SSO · Skrev 1 000 RBAC-regler for å sikre Linux-servere

{ Sky, infrastruktur & DevOps }

Sikkerhet for AI-agenter

Holde AI-en unna det den ikke skal røre

Hver agents tilgang kartlagt og begrenset til det nødvendige,
hver handling logget der agenten ikke kan endre loggen,
og hvert sikkerhetstiltak testet i stedet for tatt på tro.

Tjenesten inngår i arbeid som står andre steder på siden:

Delte opp hver driftshemmelighet per vert · Skrev en avgrensningsregel etter tilgangslekkasje · La til ett sekunds preflight for hemmeligheter · Beviste at utestengingen virker ved hver kjøring · Tre låser på døren, ingen av dem låst · Drev et løpende program for sikkerhetsskanning · Bygde fail-closed hastighetsgrenser mot misbruk · Ledet tilgangsstyring på tvers av M365 og SSO · Skrev 1 000 RBAC-regler for å sikre Linux-servere

{ Rådgivning & ledelse }

Teknisk ledelse og rådgivning

En senioringeniør i rommet

Noen på din side av bordet når en leverandør, et budsjett eller en arkitekturbeslutning trenger en ny, kyndig vurdering.

La dokumentasjonen under sin egen linter

Satte dokumentasjonen under sin egen linter — et linjebudsjett per fil som bare kan strammes inn, et krav om indeks og en kontroll av reelle stier — som fant seks av ni planer merket som ferdige med uavkryssede bokser, og fjorten commit-sperrende oppgaver som ikke var nevnt i noe dokument.

Veiledet selskapets prosjekter gjennom utvikling

Veiledet gjennomføringen av en rekke prosjekter i selskapet og ga ekspertstøtte i programvareutviklingens faser.

Ledet selskapets vekst fra 4 til 14 ansatte

Ledet selskapets vekst fra 4 til 14 ansatte med smidige metoder og effektiv prosjektstyring.

Skrev 50 000 ord programvaredokumentasjon

Skrev 50 000 ord grundig programvaredokumentasjon i Markdown i GitHub, Craft og Confluence, og sikret dermed at kunnskapen ble bevart og prosessen gjennomsiktig.

Anslot 4 000 timer spart, team fra 2 til 18

Anslot 4 000 timer spart gjennom veiledning av et team som vokste fra 2 til 18 medlemmer, bedre arbeidsflyter og koordinering mellom avdelinger.

Tjenesten inngår i arbeid som står andre steder på siden:

Holdt 4 repoer til én historikkstandard · Fikk testsuiten ned i 8,9 sekunder ·

Tok i bruk streng samtidighet i Swift 6 over C · Fant at hookene og porten var uenige ·

Skrev tester for selve kontrollverktøyene · Aktiverte alle linterregler som feil ·

Holdt generatoren til 981 tester og 92 % dekning · Lintet selve kvalitetsporten på 10 242 linjer ·

Drev utviklingen som 51 skriftlige initiativer · Et sikkerhetsnett som ikke lenger koster en halvtime ·

Bygde repoets kontrollmotor med 268 kontroller · Bygde en testsuite for Go, SQL og nettleser ·

Leverte analyser og jevnlige fremdriftsrapporter · Tok plattformens grunnleggende tekniske valg ·

Satte et krav om null advarsler på seks språk · Effektiviserte CI/CD og sparte 4 000 timer ·

Fornyet interne prosesser og sparte 8 000 timer · Ledet 30+ GIS-prosjekter fra utvikling til støtte ·

Styrte fullstack-utviklingen av GIS-kart · Ledet en GIS-kartapp til 10 ganger mer omsetning ·

Én infrastrukturplan for 14 avdelinger

{ Rådgivning & ledelse }

Teambygging og mentorarbeid

Utvikle teamet du allerede har

Rekruttering, onboarding og coaching, slik at kunnskapen blir værende hos dine egne folk i stedet for å forsvinne med en konsulent.

Ledet et team som leverte IT-støtte

Ledet et team som leverte IT-støtte, datagjenoppretting og maskinvarereparasjoner til over 1 000 kunder, og sikret høy servicekvalitet.

Sparte 150 timer i måneden på 30 prosjekter

Økte effektiviteten i prosjektene og sparte 150 timer i måneden fordelt på 30 prosjekter ved å optimalisere arbeidsflyter og ressursstyring.

Etablerte en avdeling for teknisk støtte

Etablerte en avdeling for teknisk støtte som betjente over 10 000 kunder med IT-støtte og feilsøking, og fulgte opp kundehenvendelsene i et CRM-system.

Designet og styrte 5 000 timer med kartutvikling

Designet og styrte 5 000 timer med kartutvikling, med Scrum og Jira for effektiv prosjektstyring.

Tjenesten inngår i arbeid som står andre steder på siden:

Veiledet selskapets prosjekter gjennom utvikling ·

Forbedret kommunikasjonen og samarbeidet i teamet · Ledet selskapets vekst fra 4 til 14 ansatte ·

Ledet 30 vellykkede GIS-prosjekter til levering · Anslot 4 000 timer spart, team fra 2 til 18

{ Rådgivning & ledelse }

Prosjektledelse (smidig)

Få det levert

Noen som står ansvarlig for omfang, datoer og de ubehagelige samtalene, slik at et prosjekt har én eier i stedet for fem meninger.

Tjenesten inngår i arbeid som står andre steder på siden:

Holdt 4 repoer til én historikkstandard · La dokumentasjonen under sin egen linter · Drev utviklingen som 51 skriftlige initiativer · Automatiserte samarbeid i teamet · Designet et tidsstyrings- og rapportsystem · Leverte analyser og jevnlige fremdriftsrapporter · Sparte 150 timer i måneden på 30 prosjekter · Etablerte en avdeling for teknisk støtte · Fornyet interne prosesser og sparte 8 000 timer · Veiledet selskapets prosjekter gjennom utvikling · Forbedret kommunikasjonen og samarbeidet i teamet · Ledet selskapets vekst fra 4 til 14 ansatte · Ledet 30 vellykkede GIS-prosjekter til levering · Samlet inn og analyserte forretningskrav · Designet og styrte 5 000 timer med kartutvikling · Anslot 4 000 timer spart, team fra 2 til 18

{ Rådgivning & ledelse }

Produktstrategi og krav

Finne ut hva som skal bygges

Gjøre det virksomheten ønsker seg, om til noe en utvikler kan bygge, og si høyt hva som ikke er verdt å bygge.

En sammenhengende identitet på tvers av kanaler

Utviklet merkevareidentiteten til Engineer ApS i trykt og digitalt materiale, med forbindelse til selskapets ingeniørportefølje. Grunnleggeren beskriver en økning i merkevarekjennskap på 200 ganger som resultat av arbeidet.

Tjenesten inngår i arbeid som står andre steder på siden:

Skrev om feilmeldingene som skyldte på brukerne · Drev utviklingen som 51 skriftlige initiativer · Bygde en commit-port med 22 lintere · Bygde arbeidsflater for rekruttering og sjøfolk · Bygde det sosiale laget: innlegg, feed og grupper · Bygde en lojalitetslommebok, ligaer og merker · Bygde eierskapskrav og avgjørelse for selskaper · Bygde operatørenes backoffice for plattformen · Bygde tilgangsrettigheter for Stripe og appkjøp · Designet et tidsstyrings- og rapportsystem · Designet produktets UI og UX fra ende til annen · Utviklet et rapporteringssystem for dataanalyse · Samlet inn og analyserte forretningskrav · Designet og styrte 5 000 timer med kartutvikling · Presenterte 200 UI/UX-forbedringer · Optimaliserte prognose- og investeringsstrategier · Ledet en GIS-kartapp til 10 ganger mer omsetning · Merkevarekommunikasjon ut fra publikums reaksjoner

{ Rådgivning & ledelse }

Teknisk dokumentasjon

Skrive det ned, så den neste kan følge det

Overleveringsdokumenter, driftsinstrukser og håndbøker som lar dine egne ansatte drifte systemet uten å ringe den som bygde det.

Tjenesten inngår i arbeid som står andre steder på siden:

Fikk seks innsamlere til å si hva de hoppet over · Holdt 4 repoer til én historikkstandard · Skrev om feilmeldingene som skyldte på brukerne · Skrev en parser som verifiserer hvert FFI-kall · Bygde en tallkontroll på tvers av språk · Fant at hookene og porten var uenige · Bestod 106 av 106 PDF/UA-1-profilkontroller · La dokumentasjonen under sin egen linter · Drev utviklingen som 51 skriftlige initiativer · Bygde en commit-port med 22 lintere · Skrev en avgrensningsregel etter tilgangslekkasje · Bygde 7 skrivebeskyttede rapportroller for verter · Drev et løpende program for sikkerhetsskanning · Bygde repoets kontrollmotor med 268 kontroller · Skrev 578 automatiseringsmål i go-task · Migrerte API-et til Huma v2 med 100 % paritet · Skrev 50 000 ord programvaredokumentasjon · Leverte 8 Power BI-prosjekter med veiledninger · Én infrastrukturplan for 14 avdelinger

{ IT-støtte & maskinvare }

IT-støtte og helpdesk

Datahjelp i hverdagen

Den bærbare PC-en, skriveren, påloggingen og e-postkassen, ordnet raskt og på klart og enkelt språk, uten at noen føler seg dum.

Diagnostiserte og reparerte bærbare datamaskiner

Diagnostiserte og reparerte maskinvare i bærbare datamaskiner — skjermer, hengsler, tastaturer, styreflater, hovedkort og strømforsyning — og løste programvareproblemer på Linux, Windows og Mac.

Gjenopprettet data fra SD-kort, HDD, SSD og RAID

Utførte datagjenoppretting fra et bredt spekter av medier — SD-kort, HDD-er, SSD-er, RAID-oppsett, eksterne disker og Mac-systemer.

Styrte 4 000 datamaskinreparasjoner

Styrte 4 000 datamaskinreparasjoner og sikret rask og effektiv løsning av maskinvare- og programvareproblemer.

Tjenesten inngår i arbeid som står andre steder på siden:

Ga 24/7-støtte for ca. 1 000 IPTV/OTT-servere · Konfigurerte og installerte 1 000 Wi-Fi-rutere · Ledet et team som leverte IT-støtte · Etablerte en avdeling for teknisk støtte

{ IT-støtte & maskinvare }

Maskinvarereparasjon og dataredning

Redde filer fra en død maskin

Reparere det som er verdt å reparere, og hente fotografiene, fakturaene eller tegningene ut av en disk som har sluttet å virke.

Tjenesten inngår i arbeid som står andre steder på siden:

Diagnostiserte og reparerte bærbare datamaskiner · Gjenopprettet data fra SD-kort, HDD, SSD og RAID · Styrte 4 000 datamaskinreparasjoner · Ledet et team som leverte IT-støtte

{ Merkevare & markedsføring }

Merkevare, markedsføring og SEO

Bli funnet og gjøre et godt inntrykk

Et navn, et uttrykk og et nettsted som dukker opp når noen søker etter det du selger, og som fremstår likt overalt der du er synlig.

Tjenesten inngår i arbeid som står andre steder på siden:

Bygde den visuelle identiteten fra 2 tegninger · Bygde et macOS-designsystem med 48 tokens · Rettet et sitemap med ett felles tidsstempel · Kuttet stilarkpakken fra 87 KB til 31 KB · Erstattet 963 metadatablokker med én graf · Lukket fargesystemet ved 28 farger · Genererte 495 sider fra en SQLite-eksport · Byttet 27 skriftstørrelser mot en skala i 6 trinn ·

Kombinerte webutvikling med SEO og innhold ·
Merkevarekommunikasjon ut fra publikums reaksjoner ·
En sammenhengende identitet på tvers av kanaler

Sammenlign utgangspunktet deres

Velg et sammenlignbart arbeid og identifiser forskjellene i deres egen situasjon.

Se hva det koster Ta kontakt →

Drøft en relevant tilnærming

Be oss gjøre relevant erfaring til en tilnærming dere kan vurdere.

Se tjenestene våre

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://engineer.company/nb/portfolio/