Engineer Data

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.

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 →

Bla etter kategori

PROGRAMVAREUTVIKLING

Backend-utvikling · Frontend-utvikling · Fullstack-utvikling · API-er og integrasjon · Webutvikling · 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 · 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 →

{ 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 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 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 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 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.

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.

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.

Implementerte en katalogstyrt deep merge

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

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.

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.

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.

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.

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.

{ 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.

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.

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

{ 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.

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.

Tjenesten inngår i arbeid som står andre steder på siden:

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.

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.

Et kartprodukt, håndtert fra ende til annen

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

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.

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.

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.

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.

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.

Tjenesten inngår i arbeid som står andre steder på siden:

Bygde egen analyse inne i PostgreSQL · Utviklet 600 ETL/ELT-pipelines i PL/pgSQL · Automatiserte databehandling i Python og PL/pgSQL · Designet og driftet 6 ETL/ELT-pipelines · Gjorde en geodatapipeline 50 ganger raskere · 30 dagers strømdata på 180 ms

{ 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.

Utviklet et rapporteringssystem for dataanalyse

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

Optimaliserte prognose- og investeringsstrategier

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

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.

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.

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 · Sparte 4 000 timer med GitHub og GitLab CI/CD · 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.

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.

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.

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.

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:

Flyttet hver tekststreng inn i et innholdstre · 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

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://data.engineer.company/nb/portfolio/