Engineer

Frontendutveckling

42 resultat

Koppla samman systemets delar

Arbetet under ”Frontendutveckling” visar en del av ett fungerande system. Vi kan hjälpa er att koppla den delen till resten av verksamheten. Gränssnitt som laddar snabbt, fungerar i mobilen och beter sig likadant i alla webbläsare dina kunder råkar använda.

Se våra tjänster Hör av er →

Hitta en jämförbar utmaning

Öppna ett exempel som liknar er situation. Genomgången förklarar besluten bakom resultatet, utöver teknikens namn. Styrde full stack-utvecklingen av GIS-kartor — kartutveckling i full stack, från databas till webbläsare

Styrde full stack-utvecklingen av GIS-kartor Hör av er →

Styrde full stack-utvecklingen av GIS-kartor

Styrde full stack-utvecklingen av GIS-kartor med ansvar för PostgreSQL, Mapbox, ReactJS och NodeJS för att leverera en integrerad lösning.

Presenterade 200 UI/UX-förbättringar

Presenterade 200 UI/UX-förbättringar för GIS-kartapplikationen och ökade intäkterna 10 gånger genom förbättrade programfunktioner.

Bidrog till designen av användargränssnitt

Bidrog till designprocesserna för användargränssnitt och säkerställde intuitiva, visuellt tilltalande och användarvänliga gränssnitt i projekten.

Utvecklade 100 webbapplikationer med HTML/HTML5

Utvecklade 100 webbapplikationer med HTML/HTML5, CSS/SCSS, Django, WordPress och Joomla och säkerställde en mångsidig närvaro på nätet.

Utformade 30 webbplatser från Photoshop till HTML

Utformade 30 webbplatser och levererade unika och flexibla lösningar genom att omvandla Photoshop-designer till HTML.

En maritim plattform med tydliga systemgränser

Utformade arkitekturen för en skiktad sjöfartsplattform som håller isär en PWA-frontend i Next.js, ett API i Go (Huma/Fiber) och ett funktionsskikt i PostgreSQL, med all affärslogik i databasen.

Byggde växling av organisationskontext

Utformade ett system för att växla organisationskontext med localStorage hos klienten och en speglad cookie på serversidan, så att användare kan agera som de organisationer de förvaltar medan auktorisering enligt minsta behörighet upprätthålls.

Byggde Next.js 16-frontenden: SSR, SSG och CSR

Byggde frontenden i Next.js 16 med medvetna SSR-, SSG- och CSR-strategier och en återanvändbar fabrik för förhämtade serversidor som tar bort N+1-kaskaden av hämtningar från varje inloggad sida som flyttas över till den.

Levererade fullt stöd för Progressive Web App

Levererade fullt stöd för Progressive Web App — installerbar och med offlinestöd — med körtidscachning i Workbox via next-pwa.

Byggde ett ”Liquid Glass”-designsystem för iOS 26

Utformade ett designsystem i stil med ”Liquid Glass” från iOS 26 och en kanonisk komponentförteckning som lintregler upprätthåller för att förhindra att gränssnittet spretar.

Utformade hela produktens UI och UX

Utformade produktens UI och UX från början till slut — katalogrutnät, dubbla kort- och tabellvyer, kravvalidering i realtid och brödsmulenavigering.

Härdade applikationen med nonce-baserad CSP

Härdade applikationen med nonce-baserad CSP, HSTS, SameSite-cookies, databasroller med minsta behörighet och förnyade kontroller av åtkomsträttigheter på serversidan.

Byggde ett i18n-system för engelska och danska

Etablerade ett internationaliseringssystem för engelska och danska med 12 027 meddelanden per språkversion i 458 namnrymdsfiler, med ett ordförråd som lintern upprätthåller och en budget på 400 rader per fil.

Fattade plattformens tekniska grundbeslut

Fattade plattformens grundläggande beslut under veckorna efter att repot öppnades i november 2025 — skiktningen, dataåtkomst med databasen först och ribban med noll varningar — och de håller fortfarande nio månader senare.

Byggde och underhöll Django-appar för IPTV

Utvecklade och underhöll Django-webbapplikationer för IPTV-plattformen, levererade nya funktioner och förbättrade prestanda och stabilitet.

Skötte Hugo-webbplats, förbättrade produktens UI

Underhöll och vidareutvecklade den äldre statiska webbplatsen i Hugo och bidrog samtidigt med UI/UX-förbättringar till den huvudsakliga produkten för tillgångsförvaltning.

Kombinerade webbutveckling med SEO och innehåll

Drev kundernas framgång på webben genom att kombinera skräddarsydd webbutveckling och design med SEO, innehållsstrategi och copywriting.

Byggde en testsvit för Go, SQL och webbläsare

Byggde en skiktad automatiserad testsvit — 981 Go-tester, 543 specifikationer för frontend och webbläsare, 494 beteendetester i SQL — med mutationstestning, egenskapsbaserade tester och en tillgänglighetsgrind.

Byggde plattformens backoffice för operatörer

Byggde operatörernas backoffice — ca 40 administrationsrutter och 128 komponenter för anspråk, moderering, funktionsflaggor, cache och diagnostik — så att plattformen kan drivas utan databasåtkomst.

Byggde anspråk på företagsägarskap och prövning

Byggde anspråk på företagsägarskap från början till slut — en användare gör anspråk på ett företag, en administratör avgör, och ett godkännande skriver om auktoriseringsgrafen som bestämmer vem som får redigera vad.

Byggde socialt skikt: inlägg, flöde och grupper

Byggde plattformens sociala skikt — inlägg, flöde, grupper, omnämnanden, en följargraf och en sammanställning av bemärkelsedagar — på samma funktionsbaserade dataskikt som resten av produkten.

Byggde en flerspråkig Hugo-webbplats på 3 språk

Byggde en flerspråkig statisk webbplats i Hugo med 108 mallfiler, varav 53 partials, som publicerar varje sida i fyra representationer från ett innehållsträd på tre språk, och en gång till under sju fokuserade underdomäner byggda från samma träd.

Byggde en commit-grind med 22 linters

Byggde en commit-grind med 22 linters på en rad var plus fem som förtjänar ett eget stycke, utan varningsnivå och utan tillåtna inline-undantag, som täcker HTML, CSS, JavaScript, Python, YAML, Markdown, shell, länkar, stavning, hemligheter och typografi.

Ett skyddsnät som inte längre kostar en halvtimme

Kortade webbplatsens webbläsarstyrda kvalitetsgrind från 1 636 sekunder till 615 genom att schemalägga kontrollerna med den längsta först i en workerpool begränsad till fyra spår, efter att ha mätt att alfabetisk ordning kostade 320 sekunder mot 224.

Ersatte 27 teckenstorlekar med en skala i 6 steg

Ersatte 27 renderade teckenstorlekar, vars närmaste grannar låg 0,6 % från varandra, med en Major Third-skala i sex steg, och skrev lintern som fäller den tjugoåttonde.

Tillgänglighetskontroller på 32 sidor i båda teman

Implementerade WCAG 2.2 Level AA-kontroller på 32 representativa sidor i båda färgteman, med en överensstämmelseförteckning, kontroll av ett AAA-kriterium och det partiella resultatet för kriterium 1.4.13 uttryckligen angivet.

Tillgänglighetsfelen som ert verktyg missar

Hittade och åtgärdade 11 tillgänglighetsbrister som webbläsaren rapporterade som felfria — åtta sidfotslänkar som låg kvar i tabbordningen bakom pointer-events, en scroll-tidslinje som klippte bort kolofonen på tre sidor och en regelmotor som körde 70 av sina 105 regler.

Genererade 495 sidor från en SQLite-export

Genererade 495 resultatsidor på tre språk från en skrivskyddad SQLite-export, med sidadressen angiven som data så att en rättad mening inte längre flyttade sidan och bröt länken.

Stängde färgsystemet vid 28 färger

Stängde färgsystemet vid 28 dokumenterade färger med en linter som fäller ett värde som målas men inte är dokumenterat, dokumenteras men inte målas, är felmätt, skrivs om som en literal eller ligger inom ett perceptuellt avstånd på 0,02 från en befintlig färg.

Fann att det ljusa temat saknade ett helt skikt

Upptäckte att det ljusa temat hade saknat ett helsidesöverlägg sedan den dag det skrevs, genom att kräva att båda teman målar varje skiktad yta med samma antal skikt.

Fick reduced-motion att hålla vad den lovar

Fick reduced-motion att hålla vad den lovar efter att ha hittat elva selektorer som fortfarande animerade under inställningen, eftersom en universell transition-none-regel förlorar på specificitet mot varje regel som har en klass.

Byggde en utskriftsstilmall på 637 rader

Byggde en utskriftsstilmall på 637 rader mot fyra dokumenterade beteenden i renderingsmotorer, bland annat run-in-rubriker som skrevs ut vitt på vitt för varje läsare vars webbläsare föredrog mörkt läge.

Speglade webbplatsen till Gemini och Gopher

Byggde Gemini- och Gopher-utdata från samma webbkälla; resultatbeskrivningen anger 777 dokument i varje format vid det skedet och noll ändrade HTML-byte genom tillägget av formaten.

Ersatte 963 metadatablock med en enda graf

Ersatte 963 anonyma block av strukturerad data som upprepade företaget 1 671 gånger med en enda länkad graf av 16 typer, präglade från stabila ursprungsidentifierare.

Krympte stilmallspaketet från 87 KB till 31 KB

Krympte stilmallspaketet från 87 KB till 31 KB genom att ta ut ett base64-kodat typsnitt ur det, och tog bort 756 KB i 22 filer som publicerades vid varje driftsättning utan att något refererade till dem.

Lintade själva kvalitetsgrinden på 10 242 rader

Förde in 10 242 rader JavaScript för kvalitetsgrinden under en formaterare och en linter efter att ha konstaterat att det var den största kodmassan i repot och den enda som inget läste, och åtgärdade 13 fynd utan att undertrycka något.

Byggde en native macOS-klient i Swift 6

Byggde en native meddelandeklient för macOS i Swift 6 och SwiftUI — 11 141 rader i 53 filer — ovanpå C-gränssnittet till en Rust-kärna som länkas som ett statiskt arkiv från en låst revision.

Ett begränsat händelseflöde i en macOS-app

Begränsade händelsebuffertar och loggning efter en dokumenterad macOS-incident med en tillväxt på 41 MB i sekunden och 111 GB komprimerade sidor på en maskin med 36 GB; en dokumenterad fullständig testkörning minskade kärnans loggrader från 610 996 till 1 411.

Införde strikt samtidighet i Swift 6 ovanpå C

Införde fullständig strikt samtidighet i Swift 6 utan actors och överbryggade en blockerande händelseloop i C till main actor genom en producent, en konsument och en ordning — efter att ha konstaterat att en task per händelse förlorar den ordning som gränssnittet är beroende av.

Gav alla 17 ikoner namn för skärmläsare

Gav varje kontroll i gränssnittet som bara består av en ikon ett namn för skärmläsare efter att ha hittat skicka-knappen uppläst som ”arrow up circle, button”, och skrev lintern som kräver etiketten inom åtta rader från ikonen.

Byggde ett designsystem för macOS med 48 tokens

Byggde ett designsystem med 48 tokens på plattformens eget glasmaterial och skrev lintern som avvisar ett magiskt tal, en hårdkodad teckenstorlek eller en animering som ignorerar inställningen för reducerad rörelse.

Skrev om felmeddelanden som skyllde på användare

Skrev om applikationens felmeddelanden enligt en skriftlig tonstandard efter att en nekad inloggning hade skyllt på användaren för felskrivning när leverantören i själva verket krävde ett appspecifikt lösenord, och täckte det med ett test som namnger leverantören.

Fokusera den tekniska insatsen

Beskriv arbetsflödet, befintliga gränssnitt och användarna som behöver förändringen. Vi kan definiera en testbar leverans och en överlämning som gör att teamet kan underhålla lösningen.

Se vad det kostar Hör av er →

Sätt ämnet i ert sammanhang

Berätta hur ”Frontendutveckling” ingår i ert system. Vi kan hjälpa er att välja rätt tjänst och avgränsa arbetet.

Se våra tjänster

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

Ta med dig den här sidan

Visitkort

Omsorgsfullt skapad med kärlek av D.

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

Upphovsrätt © 2025 – nu · Engineer ApS · Alla rättigheter förbehållna.

Utskrivet från https://engineer.company/sv/categories/frontend/