Engineer Logiciels

Ingénierie frontend

42 réalisations

Reliez les parties de votre système

Les réalisations regroupées sous « Ingénierie frontend » montrent une partie d'un système opérationnel. Nous pouvons vous aider à relier cette partie au reste de votre activité. Des interfaces rapides, utilisables sur téléphone et cohérentes dans tous les navigateurs de vos clients.

Découvrir nos prestations Contactez-nous →

Trouvez un défi comparable

Suivez un exemple qui ressemble à votre situation. Son analyse décrit les décisions qui ont conduit au résultat, au-delà du nom de la technologie. Développement cartographique GIS full-stack — cartographie full-stack, de la base au navigateur

Développement cartographique GIS full-stack Contactez-nous →

Développement cartographique GIS full-stack

Dirigé le développement full-stack d'une application cartographique GIS, avec PostgreSQL, Mapbox, ReactJS et NodeJS, pour livrer une solution intégrée.

200 améliorations UI/UX présentées

Présenté 200 améliorations UI/UX pour l'application cartographique GIS, dont les nouvelles fonctionnalités ont contribué à multiplier le chiffre d'affaires par 10.

Contribution à la conception d'interfaces

Contribué à la conception des interfaces utilisateurs pour les rendre intuitives, agréables à regarder et faciles à utiliser.

100 applications web avec HTML/HTML5

Développé 100 applications web avec HTML/HTML5, CSS/SCSS, Django, WordPress et Joomla, pour répondre à différents besoins de présence en ligne.

30 sites web, de Photoshop au HTML

Conçu 30 sites web, en réalisant des solutions originales et flexibles à partir de maquettes Photoshop converties en HTML.

Une plateforme maritime aux frontières système claires

Conçu une plateforme maritime en couches séparant un frontend PWA Next.js, une API Go (Huma/Fiber) et une couche de fonctions PostgreSQL, avec toute la logique métier dans la base de données.

Changement de contexte d'organisation

Conçu un système de changement de contexte d'organisation avec localStorage côté client et réplication dans un cookie côté serveur, permettant d'agir au nom des organisations administrées tout en appliquant le moindre privilège.

Frontend Next.js 16 : SSR, SSG et CSR

Développé le frontend Next.js 16 en choisissant explicitement les stratégies SSR, SSG et CSR, avec une fabrique réutilisable de pages serveur préchargées supprimant la cascade de requêtes N+1 sur chaque page authentifiée migrée.

Prise en charge complète des Progressive Web Apps

Livré une prise en charge complète des Progressive Web Apps — installation et fonctionnement hors ligne — avec mise en cache à l'exécution par Workbox via next-pwa.

Système de design iOS 26 'Liquid Glass'

Conçu un système de design iOS 26 'Liquid Glass' et un inventaire de composants de référence, maintenu par des règles de linting pour prévenir les divergences d'interface.

Conception UI et UX du produit de bout en bout

Conçu l'UI et l'UX du produit de bout en bout — grilles d'annuaire, doubles vues cartes/tableau, validateurs d'exigences en temps réel et navigation par fil d'Ariane.

Application durcie avec une CSP à nonces

Durci l'application avec une CSP fondée sur des nonces, HSTS, des cookies SameSite, des rôles de base de données au moindre privilège et une nouvelle vérification des droits côté serveur.

Système i18n anglais/danois

Mis en place un système d'internationalisation anglais/danois comportant 12 027 messages par langue dans 458 fichiers d'espaces de noms, avec vocabulaire contrôlé par linting et limite de 400 lignes par fichier.

Choix techniques fondateurs de la plateforme

Défini les choix fondateurs de la plateforme dans les semaines suivant l'ouverture du dépôt en novembre 2025 — architecture en couches, accès aux données centré sur la base et zéro avertissement — toujours en vigueur neuf mois plus tard.

Applications Django pour IPTV

Développé et maintenu des applications web Django pour la plateforme IPTV, en livrant de nouvelles fonctionnalités et en améliorant les performances et la stabilité.

Site Hugo maintenu et interface produit améliorée

Maintenu et amélioré le site statique Hugo existant tout en contribuant à des améliorations UI/UX du produit principal de gestion d'actifs.

Développement web, SEO et contenu réunis

Amélioré les résultats web des clients en associant développement et conception de sites sur mesure, SEO, stratégie de contenu et rédaction.

Suite de tests Go, SQL et navigateur

Conçu une suite de tests automatisés en couches — 981 tests Go, 543 spécifications frontend et navigateur, 494 tests de comportement SQL — avec tests par mutation, tests fondés sur les propriétés et contrôle obligatoire de l'accessibilité.

Back-office d'exploitation de la plateforme

Conçu le back-office d'exploitation — environ 40 routes d'administration et 128 composants pour les revendications de propriété, la modération, les indicateurs de fonctionnalités, le cache et le diagnostic — afin d'exploiter la plateforme sans accès à la base de données.

Revendications de propriété et arbitrage

Conçu de bout en bout les revendications de propriété d'entreprise : un utilisateur revendique une entreprise, un administrateur statue, puis l'approbation réécrit le graphe d'autorisation déterminant qui peut modifier quoi.

Couche sociale : publications, fil et groupes

Conçu la couche sociale de la plateforme — publications, fil, groupes, mentions, graphe d'abonnements et récapitulatif d'occasions — sur la même couche de données fondée sur les fonctions que le reste du produit.

Site Hugo multilingue en 3 langues

Conçu un site statique multilingue Hugo avec 108 fichiers de modèles, dont 53 partiels, publiant chaque page sous quatre représentations à partir d'une seule arborescence de contenu en trois langues, puis sous sept sous-domaines spécialisés issus de la même arborescence.

Contrôle avant commit avec 22 linters

Conçu un contrôle obligatoire avant commit composé de 22 linters tenant sur une ligne et de cinq nécessitant un paragraphe, sans niveau d'avertissement ni suppression locale autorisée, couvrant HTML, CSS, JavaScript, Python, YAML, Markdown, shell, liens, orthographe, secrets et typographie.

Des vérifications qui ne prennent plus une demi-heure

Réduit les vérifications qualité du site pilotées par navigateur de 1 636 à 615 secondes, en ordonnançant les contrôles du plus long au plus court dans un pool de quatre processus, après avoir mesuré un coût de 320 secondes en ordre alphabétique contre 224.

27 tailles remplacées par une échelle de 6

Remplacé 27 tailles de police rendues, dont certaines voisines n'étaient séparées que de 0,6 %, par une échelle de six tailles en tierce majeure, puis écrit le linter qui refuse la vingt-huitième.

Contrôles d’accessibilité sur 32 pages dans les deux thèmes

Mis en place des contrôles WCAG 2.2 niveau AA sur 32 pages représentatives dans les deux thèmes, avec un dossier de conformité, un critère AAA vérifié et le résultat partiel du critère 1.4.13 explicitement indiqué.

Les défauts d'accessibilité ignorés par votre outil

Détecté et corrigé 11 défauts d'accessibilité que le navigateur jugeait sains : huit liens de pied de page encore accessibles par tabulation derrière pointer-events, une chronologie de défilement masquant le colophon sur trois pages et un moteur n'exécutant que 70 de ses 105 règles.

495 pages générées par un export SQLite

Généré 495 pages de réalisations en trois langues à partir d'un export SQLite en lecture seule, avec des adresses définies comme données pour qu'une correction de phrase ne déplace plus la page et ne casse plus son lien.

Système limité à 28 couleurs

Limité le système de couleurs à 28 couleurs documentées, avec un linter refusant toute valeur utilisée mais non documentée, documentée mais inutilisée, mal mesurée, réécrite sous forme littérale ou située à moins de 0,02 de distance perceptuelle d'une couleur existante.

Une couche entière manquait au thème clair

Découvert qu'une couche de superposition pleine page manquait au thème clair depuis sa création, en vérifiant que les deux thèmes dessinent chaque surface composite avec le même nombre de couches.

Réduction des mouvements réellement appliquée

Rendu effective la préférence de réduction des animations après avoir trouvé onze sélecteurs qui continuaient à animer : une règle universelle supprimant les transitions perd en spécificité face à toute règle portant une classe.

637 lignes de styles d'impression

Conçu une feuille de style d'impression de 637 lignes tenant compte de quatre comportements documentés de moteurs de rendu, notamment des titres intégrés qui s'imprimaient en blanc sur blanc quand le navigateur préférait le thème sombre.

Miroirs du site sur Gemini et Gopher

Construit les sorties Gemini et Gopher depuis la même source web ; la réalisation consigne 777 documents par format à ce stade et zéro octet HTML modifié lors de leur ajout.

963 blocs de métadonnées remplacés par un graphe

Remplacé 963 blocs anonymes de données structurées répétant l'entreprise 1 671 fois par un graphe lié unique de 16 types, fondé sur des identifiants d'origine stables.

Styles allégés : de 87 KB à 31 KB

Réduit le bundle de styles de 87 KB à 31 KB en retirant une police base64, et supprimé 756 KB répartis dans 22 fichiers publiés à chaque déploiement sans être référencés nulle part.

Contrôle qualité : 10 242 lignes vérifiées

Soumis 10 242 lignes de JavaScript des contrôles qualité à un formateur et à un linter, après avoir établi qu'il s'agissait du plus gros ensemble de code du dépôt et du seul que rien ne contrôlait, en corrigeant 13 constats sans en masquer aucun.

Client natif macOS en Swift 6

Développé un client de messagerie natif macOS en Swift 6 et SwiftUI — 11 141 lignes dans 53 fichiers — sur l'interface C d'un cœur Rust lié comme archive statique depuis une révision épinglée.

Un pipeline d'événements borné pour une application macOS

Borné les tampons d'événements et la journalisation après un incident macOS enregistré : croissance de 41 MB par seconde et 111 GB de pages compressées sur une machine de 36 GB ; une exécution complète des tests consignée a réduit les journaux du cœur de 610 996 à 1 411 lignes.

Concurrence stricte de Swift 6 sur C

Adopté le mode complet de concurrence stricte de Swift 6 sans acteurs, en reliant une boucle d'événements C bloquante à l'acteur principal avec un producteur, un consommateur et un ordre unique, après avoir établi qu'une tâche par événement perdait l'ordre nécessaire à l'interface.

17 icônes nommées pour les lecteurs d'écran

Nommé chaque commande composée uniquement d'une icône pour les lecteurs d'écran après avoir trouvé le bouton d'envoi annoncé comme « flèche vers le haut dans un cercle, bouton », puis écrit le linter exigeant un libellé à huit lignes au plus de l'icône.

Système de design macOS de 48 tokens

Conçu un système de design de 48 tokens sur le matériau de verre natif de la plateforme et écrit le linter refusant les nombres arbitraires, tailles de police codées en dur et animations ignorant la préférence de réduction des mouvements.

Erreurs réécrites sans blâmer les utilisateurs

Réécrit les messages d'erreur de l'application selon une charte de ton après qu'un refus de connexion reprochait une faute de frappe à l'utilisateur alors que le fournisseur exigeait un mot de passe d'application, avec un test nommant ce fournisseur.

Ciblez l'effort d'ingénierie

Présentez le processus de travail, les interfaces existantes et les utilisateurs qui ont besoin du changement. Nous pouvons définir une livraison testable et une transmission permettant à votre équipe d'assurer la maintenance des éléments livrés.

Consulter les tarifs Contactez-nous →

Replacez le sujet dans votre contexte

Dites-nous comment « Ingénierie frontend » se manifeste dans votre propre système. Nous pouvons vous aider à choisir la prestation pertinente et à définir les travaux correspondants.

Découvrir nos prestations

welcome@engineer.company +45 50 10 61 84 Copenhague, Danemark

Emportez cette page avec vous

Fiche de contact

Créé avec soin et amour par D.

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

Droit d'auteur © 2025 – aujourd'hui · Engineer ApS · Tous droits réservés.

Imprimé depuis https://software.engineer.company/fr/categories/frontend/