Nous construisons les logiciels **que votre équipe reprendra** — et les écrivons pour qu'elle puisse le faire.

Architecture, backends, frontends, sites web, cartes, systèmes de design et plusieurs langues — depuis Copenhague.

Engineer company
Logiciels

Des logiciels conçus pour être maintenus, pas seulement livrés

+45 50 10 61 84 · Copenhague, Danemark

Créé avec soin et amour par D.

Des backends, des frontends et l'architecture qui les relie, écrits pour que la prochaine personne à ouvrir le dépôt puisse les modifier. Les comptes rendus décrivent la situation initiale, le travail effectué et le résultat.

Commencez par un processus métier, une intégration ou une partie d'application. Nous pouvons convenir d'une petite livraison avec des vérifications de réception, des interfaces documentées et une transmission permettant à votre équipe d'assurer la maintenance.

Travaux logiciels · Tarifs et modalités de collaboration · Parlons de votre projet

Partez du résultat recherché

Dites-nous ce que votre entreprise doit accomplir. Nous relions logiciels, données et infrastructure autour des personnes qui les utilisent.

Découvrir nos prestations Contactez-nous →

Découvrez notre façon de travailler

Trouvez la prestation qui correspond à votre objectif et découvrez, à travers nos réalisations, ce qui a été livré et comment.

Études de cas Contactez-nous →

Découvrez les réalisations **dans leur contexte** — les décisions, la réalisation et les résultats rapportés.

Voir l'ensemble du catalogue

Un seul partenaire pour **11 prestations** de l'architecture au dernier écran.

Parcourir le catalogue

{ Ingénierie logicielle }

Décider comment construire l'ensemble

Architecture de plateformes et de solutions

Définir d'abord le système sur le papier pour que ses éléments s'articulent et que son coût soit connu avant d'écrire du code.

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.

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.

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.

Voir les 20 réalisations

Notre façon de travailler

Comprendre d'abord

Nous étudions le problème avant de proposer une solution. Le cahier des charges que vous apportez est le début de cette conversation, pas sa conclusion.

Construire pour durer

La durabilité plutôt que la rapidité, la clarté plutôt que les astuces. Le système doit rester exploitable pour les personnes qui en hériteront.

Assumer la responsabilité

Toute exploitation ou assistance continue fait l'objet d'un accord dans le cadre de la mission. La transmission précise les responsabilités et la voie à suivre pour tout contact ultérieur.

{ Ingénierie logicielle }

Le moteur de votre application ou de votre site

Développement backend et API

La partie invisible pour les utilisateurs : où résident les données, quelles règles s'appliquent et comment les autres systèmes peuvent y accéder.

Moitié jusque-là inutilisée du cœur exploitée

Exploité la moitié du cœur de messagerie jusque-là inutilisée — transfert des sauvegardes, messages éphémères, modification et renvoi de messages, invitations vérifiées, proxys et politique de chiffrement — en exécutant chaque test contre la vraie bibliothèque, sans simulations.

Parseur vérifiant chaque appel FFI

Écrit un parseur lisant le véritable en-tête C de 7 308 lignes et vérifiant chaque site d'appel, chaque constante d'énumération et le caractère final de toute classe propriétaire de pointeurs, après qu'un en-tête de substitution manuscrit a laissé compiler et lier des appels à trois fonctions supprimées, puis provoqué un plantage.

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.

Voir les 32 réalisations

{ Ingénierie logicielle }

Les écrans que les gens utilisent

Développement frontend

Des interfaces rapides, utilisables sur téléphone et cohérentes dans tous les navigateurs de vos clients.

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.

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.

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.

Voir les 30 réalisations

{ Ingénierie logicielle }

Un produit qui fonctionne, de bout en bout

Développement de produits full-stack

Une seule équipe transforme une idée en un produit auquel vos clients peuvent se connecter, sans le faire passer entre trois prestataires.

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.

Générateur de CV piloté par base de données

Conçu en Python un générateur de CV, références, portfolio et lettres de motivation piloté par base de données — 41 modules, 10 580 lignes — produisant six formats à partir d'une source SQLite de 23 tables assemblée par un pipeline idempotent de 19 étapes.

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.

Voir les 16 réalisations

{ Ingénierie logicielle }

Un site que vous pouvez mettre à jour

Développement de sites web et CMS

Un site rapide, facile à trouver et modifiable par votre équipe, sans facture à chaque changement de texte.

Miroir Tor onion à une adresse lisible

Publié un miroir Tor onion du site à une adresse lisible commençant par engineer, avec une clé générée hors de l'hôte et jamais présente dans ce dépôt ni sur un portable, en conservant ce quatrième point d'entrée comme le seul des quatre à ne jamais être compté.

Identité visuelle créée à partir de 2 dessins

Créé les icônes et favicons de l'entreprise à partir de deux dessins faits à la main, chaque ressource dérivée étant régénérée par script, avec un contrôle échouant si un fichier dérivé a été commité avant sa source.

106 contrôles sur 106 du profil PDF/UA-1 réussis

Consigné la validation PDF/UA-1 de neuf documents avec 106 vérifications sur 106 réussies dans le profil d'accessibilité configuré, tandis que la variante d'archivage échouait sur une règle parmi 146.

Voir les 22 réalisations

{ Ingénierie logicielle }

Être visible quand vos clients interrogent une IA

Préparation aux agents d'IA

Un site ou une application que les assistants d'IA de vos clients peuvent trouver, lire et utiliser : une capacité vérifiée à chaque livraison, plutôt qu'une promesse ponctuelle.

Développement piloté par 51 initiatives écrites

Piloté le développement du site à travers 51 initiatives écrites dans 54 documents de planification, chacun conservant ses points ouverts, ses mesures et les décisions écartées, dont six des huit constats de préparation aux agents refusés avec justification consignée.

Dates identiques du sitemap corrigées

Corrigé un sitemap dont 172 des 176 URL partageaient la même date de modification, en utilisant l'historique git après avoir établi que l'export réécrivait chaque fichier à chaque exécution.

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.

Voir les 5 réalisations

{ Ingénierie logicielle }

Permettre à l'IA d'utiliser vos logiciels existants

Outils pour agents d'IA et serveurs MCP

Vos logiciels existants exposés sous forme d'outils qu'un agent d'IA peut appeler, sans réécrire ce qui fonctionne déjà.

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.

Traitements lents confiés à une file River

Sorti les traitements lents du chemin des requêtes pour les confier à une file de tâches River — 15 modules de traitement, 8 tâches planifiées et 20 tâches pg_cron — afin de répondre aux requêtes pendant la poursuite du travail en arrière-plan.

Contrat d'API généré à partir de la base

Généré le contrat d'API à partir de la base de données — OpenAPI, client TypeScript typé de 44 076 lignes, 61 gestionnaires de simulation et limites appliquées par l'interface — avec un contrôle à chaque étape qui échoue en cas de divergence.

Voir les 5 réalisations

{ Ingénierie logicielle }

Des cartes qui répondent aux questions

Solutions GIS et géospatiales

Tout ce qui se situe quelque part — navires, véhicules, parcelles, capteurs — sur une carte où chercher, mesurer et partager.

Un produit cartographique pris en charge de bout en bout

Conçu l'architecture, développé et maintenu l'infrastructure, le traitement des données et une application cartographique pendant 2 ans, en couvrant la livraison logicielle et la maintenance d'exploitation.

Plus de 30 projets GIS, du code à l'assistance

Dirigé le développement, le déploiement et l'assistance de plus de 30 projets GIS, avec PostgreSQL, Bash, Python, JavaScript, GDAL, ArcGIS, PostGIS et Mapbox.

30 projets GIS menés jusqu'à la livraison

Piloté l'exécution de 30 projets GIS réussis, avec une responsabilité de direction dans la livraison de solutions nouvelles dans ce domaine.

Voir les 13 réalisations

{ Ingénierie logicielle }

Rendre l'utilisation évidente

Conception UI/UX et systèmes de design

Une organisation des écrans, des textes et des parcours qui permet aux utilisateurs d'accomplir leur tâche sans appeler l'assistance.

Identité visuelle créée à partir de 2 dessins

Créé les icônes et favicons de l'entreprise à partir de deux dessins faits à la main, chaque ressource dérivée étant régénérée par script, avec un contrôle échouant si un fichier dérivé a été commité avant sa source.

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.

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.

Voir les 20 réalisations

{ Ingénierie logicielle }

Fonctionner dans plusieurs langues

Internationalisation et localisation

Un logiciel qui se lit naturellement sur chaque marché où vous vendez, y compris les dates, les devises et le classement des noms.

Contrôle des chiffres entre langues

Conçu un contrôle multilingue échouant quand une traduction omet un chiffre présent en anglais ou utilise une notation étrangère à sa langue, révélant deux descriptions danoises privées d'une mesure et seize passages ukrainiens utilisant les guillemets anglais.

Tous les textes déplacés dans le contenu

Déplacé chaque texte destiné aux utilisateurs hors de Python vers une arborescence de 874 fichiers en trois langues, après avoir trouvé des traductions inutilisées dont l'inactivité était invisible et un contrôle n'évaluant silencieusement qu'un tiers des réalisations.

Générateur de CV piloté par base de données

Conçu en Python un générateur de CV, références, portfolio et lettres de motivation piloté par base de données — 41 modules, 10 580 lignes — produisant six formats à partir d'une source SQLite de 23 tables assemblée par un pipeline idempotent de 19 étapes.

Voir les 8 réalisations

{ Ingénierie logicielle }

Un assistant d'IA qui connaît votre métier

Développement d'agents d'IA sur mesure

Un agent connecté à vos systèmes, testé sur des cas rédigés par vos experts, supervisé par votre équipe et indépendant d'un modèle unique.

Six collecteurs indiquent les entrées ignorées

Fait expliciter à six collecteurs de reporting la partie de leurs entrées jamais lue — fichiers non ouverts, sondes non installées, lignes du journal non reconnues — et ajusté les seuils sur 116 échantillons mesurés, après avoir établi qu'une collecte aveugle et un hôte sain produisaient la même page.

Tests des contrôleurs eux-mêmes

Écrit des tests pour les contrôleurs après avoir établi qu'un contrôleur alimenté uniquement en entrées correctes peut finir par déclarer une réussite sans rien lire : introduction d'une faute pour vérifier sa détection et lecture d'une plage d'identifiants dans la base plutôt que dans une constante du test.

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.

Voir les 5 réalisations

Nous travaillons avec **les plateformes** sur lesquelles votre entreprise s'appuie déjà.

Apple Microsoft DigitalOcean AWS Google

Voir l'ensemble des technologies

Prêt pour les agents IA que vos clients vous envoient

Repérable

Un fichier robots qui indique ce que les agents peuvent faire, un plan du site avec des dates exactes, et chaque changement annoncé par IndexNow le jour de sa mise en ligne.

Lisible

Une copie Markdown épurée de chaque page, un fichier llms.txt servant de guide aux modèles de langage et des données structurées qui décrivent votre activité.

Utilisable

Des formulaires et des commandes accessibles, ainsi qu'une API documentée pour les opérations de l'application sélectionnées pour l'accès des agents.

Voir le fonctionnement Parlons de votre site →

Des agents IA conçus pour les tâches de votre métier

Connectés

Reliés à votre ERP, votre CRM, votre système de réservation et votre messagerie par des outils documentés, chacun chargé d'une tâche précise et limitée.

Évalués

Évalués sur des cas réels rédigés par vos experts, afin que chaque modification d'une instruction, d'un outil ou d'un modèle soit testée avant sa mise en service.

Supervisés

Ce que l'agent fait seul et ce qu'une personne approuve : des règles rédigées avec vous, appliquées dans les outils, avec une traçabilité étape par étape.

Voir comment les agents sont construits Parlons de votre métier →

Vos logiciels transformés en outils que les agents peuvent appeler

Décomposés

Les règles cachées dans les écrans, formulaires et procédures stockées sont identifiées, documentées et regroupées dans une seule couche.

Exposés

Chaque opération est proposée sous forme d'outil typé et documenté via une API ou un serveur MCP, agissant uniquement dans les limites des droits de son utilisateur.

Maintenus en service

Une intégration au périmètre défini peut exposer une opération à la fois, parallèlement aux écrans existants ; les modifications de l'application sont convenues après examen de ses interfaces.

Voir comment ouvrir les logiciels Parlons de vos logiciels →

Dites-nous ce que vous construisez.

Pas de formulaire, pas de tunnel de conversion, pas de séquence commerciale — simplement un courriel à une personne qui le lira.

Engager la conversation

Cette adresse ne présente que les travaux logiciels. L'entreprise travaille aussi sur les données, exploite des infrastructures et conseille des équipes : ces activités sont sur engineer.company.
Voir tout ce que fait Engineer ApS →

Commencez par une amélioration ciblée

Un processus de travail, un flux de données ou un système à améliorer peut constituer une première mission utile. Nous pouvons en définir le périmètre ensemble.

Consulter les tarifs Contactez-nous →

Préparez le premier échange

Indiquez-nous le résultat attendu, ce qui est déjà en place et le calendrier souhaité. Nous pourrons alors discuter de la prochaine étape.

Découvrir nos prestations

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

Côté serveur, nous utilisons **Linux en logiciel libre** — sans licence à payer ni rien de caché.



Ubuntu



Debian

Comment nous exploitons les serveurs



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/