Engineer

実績紹介

エンジニアリングサービスと、その実績をご紹介します。

Engineer ApSは2025年に登記されました。掲載している仕事の大半は、それ以前に創業者がお客様のチームの一員として実施したものです。会社として実施した仕事も、その後の雇用先で実施した仕事も含まれます。同じ担当者が同じ基準で取り組んでいますが、それぞれの立場を曖昧にせず明示しています。

実績を読む **その背景とともに** — 判断、実施内容、報告された成果をご覧ください。

実績一覧を見る

生かす価値のある経験を探す

この記録から、プロジェクトに関連する経験をお探しください。

サービスを見る お問い合わせ →

記録の背景にある判断を読む

各項目は、さまざまなサービスにわたる実施済みの仕事を説明しています。振り返りを読むと、その背景にある状況と判断が分かります。

事例 お問い合わせ →

これらの技術は **以下の実績全体で使われています** — 引き継ぐチームが特殊な技術に戸惑うことはありません。

Python PostgreSQL TypeScript React GitHub Docker

技術構成をすべて見る

カテゴリーから探す

ソフトウェアエンジニアリング

バックエンド開発 フロントエンド開発 フルスタック開発 API・システム連携 Web開発 デザインシステム・UI UX・UI設計 テスト・品質保証 性能改善 Python 国際化

データ・データベース

データベース / PostgreSQL / SQL / データエンジニアリング / データパイプライン（ETL/ELT） / データ分析 / データガバナンス / データウェアハウス / GIS・地理空間情報

クラウド・インフラ・DEVOPS

クラウド / コンテナ（Docker/Kubernetes） / DevOps / 自動化・CI/CD / インフラ / Linux・サーバー / システム管理 / ネットワーク・VPN / 監視・オブザーバビリティ / 信頼性・バックアップ / セキュリティ / プラットフォーム設計 / 移行・モダナイゼーション

コンサルティング・リーダーシップ

技術面の統率 / チームの統率 / 指導・育成 / アジャイル・Scrum / プロジェクト管理 / プロダクト・要件 / 関係者との調整・報告 / 文書作成 / ソリューション設計 / ブランド・マーケティング

以下の仕事はすべて **実施済みです** — 提案や計画ではありません。

各項目には、その仕事で何が変わったかを記載し、関連するサービスへのリンクを設けています。

提供するサービス 料金 →

{ ソフトウェアエンジニアリング }

プラットフォーム・ソリューション設計

システム全体のつくり方を決める

まず全体像を設計し、各部分のつながりと費用を、コードを書き始める前に明らかにします。

C上にSwift 6の厳密並行処理を採用

インターフェースに必要な順序がイベントごとのタスク生成で失われると確認し、独自アクターを用いずSwift 6の完全な厳密並行処理を採用。ブロッキングCイベントループとメインアクターを、生産者1つ、消費者1つ、順序1つで接続した。

macOSアプリのイベントパイプラインに上限を設定

毎秒41 MBの増加と、36 GB端末で111 GBの圧縮ページを記録したmacOSの障害を受け、イベントバッファとログ出力に上限を設定。記録された全テスト実行では、コアのログ行が610,996から1,411へ減った。

Swift 6でネイティブmacOSクライアントを構築

固定リビジョンから静的アーカイブとしてリンクしたRustコアのCインターフェース上に、Swift 6とSwiftUIで53ファイル、11,141行のネイティブmacOSメッセージクライアントを構築した。

522行のQR符号器を同梱

実行時の3つ目の依存関係を追加せず、GF(256)上のReed-Solomon、固定モジュール配置、8つのマスクパターンを持つQR符号器を522行で同梱し、独立に実装した復号器で完成行列を読み戻して検証した。

データベース駆動の履歴書生成器を構築

19段階の冪等な処理で組み立てる23テーブルのSQLiteデータを単一の情報源として、6形式を出力する、データベース駆動の履歴書・推薦状・実績集・応募書類生成器をPythonの41モジュール、10,580行で構築した。

小型サーバー上の公開サービス

利用可能メモリー464 MBを設計制約として、512 MB・1コアのホストで自社の公開サービスを運用。gitホスティング、7ドメインのWebサーバー、Tor、2つの代替プロトコルサーバー、不正接続元の遮断を備え、バックアップ自動化は別の実装として扱った。

基幹インフラ全体を一から構築

グリーンエネルギー分野のSaaSスタートアップで初期に採用された社員の1人として、基幹インフラ全体と運用工程を一から設計・構築し、急成長の基盤を整えた。

数年後も稼働するインフラを構築

社内外のシステム向けに新たなインフラ機能を計画・実装し、数年後も最小限の変更で稼働し続ける仕組みを構築した。

プラットフォームの基礎技術方針を策定

2025年11月のリポジトリ開設後の数週間に、多層構成、データベース中心のデータアクセス、警告ゼロという基礎方針を定め、9か月後も維持した。

34のPostgresスキーマに348テーブルをモデル化

専門家、企業、船舶、求人、レビューなどの海事領域を、34個のPostgreSQLスキーマ内の348個の正規化テーブルにモデル化し、SMALLINTの参照値とUUID v7キーを用いた。

PostgreSQL関数中心のデータ層を設計

34スキーマにまたがる1,275個のストアド関数を持つ、PostgreSQL関数中心のデータ層を設計。すべての読み書きをテーブル直接操作ではなく、データベースで実行権限を付与できる関数経由にした。

APIを100%の一致でHuma v2へ移行

HTTP APIの649パス・760操作をFiberからHuma v2へ移行し、サーバーが登録するルートと公開するOpenAPI記述の100%一致を達成・維持した。

利用者の組織コンテキスト切り替えを構築

クライアントのlocalStorageとサーバー側Cookieへの複製を用いて組織の操作コンテキストを切り替える仕組みを設計し、最小権限の認可を維持しながら管理対象の組織として操作できるようにした。

PostgresからAPIへのJSONパススルーを設計

PostgreSQL関数が完成したJSONを返し、Go APIがそのまま転送するJSONパススルー構成を設計。中間のアンマーシャリングをなくし、フロントエンドをスキーマ変更から切り離した。

システムの境界を明確にした海事プラットフォーム

Next.jsのPWAフロントエンド、Go（Huma/Fiber）API、PostgreSQL関数層を分離し、業務ロジックをすべてデータベースに置く、多層の海事プラットフォームを設計した。

AWSの外へ移し、インフラを整理

海外顧客のインフラを見直し、未使用サービスを廃止して容量を調整し、ワークロードをAWSから移した。創業者の報告では、生産性を損なうことなくクラウド費用を10分の1に削減した。

地図製品を最初から運用まで支える

2年間にわたり、インフラ、データ処理、地図アプリケーションを設計・開発・支援し、ソフトウェアの提供と運用保守の両方を担当した。

業務工程の刷新で8,000時間を削減

社内工程を見直し、ソフトウェア構成、システム、日程管理の効率化によって8,000時間を削減した。

GIS地図アプリの開発を主導し売上を10倍に

GIS地図アプリケーションのソフトウェア開発を主導し、売上を10倍に伸ばして、製品を主要なデータ資産として位置づけた。

14部門を支える共通インフラ計画

研究機関の14部門に関わるインフラ全体の枠組みを設計。柔軟なモジュール、統一されたデータパイプライン、長期的な定着を支える体系的なサポート方針を備えた。

{ ソフトウェアエンジニアリング }

バックエンド・API開発

アプリやWebサイトを裏側で動かす仕組み

データの保存先、処理のルール、他のシステムとの接続方法など、利用者の目に見えない部分をつくります。

未使用だったコアの半分の機能を実装

バックアップ転送、消えるメッセージ、編集・再送、確認済み招待、プロキシ、暗号化方針など、アプリケーションが未使用だったメッセージングコアの半分の機能に対応し、モックなしで実ライブラリに対して全テストを実行した。

全FFI呼び出しを検証する構文解析器を作成

手書きの仮Cヘッダーで、削除済みの3関数を呼んでもコンパイル・リンクが通り実行時に落ちた問題を受け、実際の7,308行のCヘッダーを読み、全呼び出し箇所、列挙定数、ポインター所有クラスのfinal指定を検証する構文解析器を作成した。

すべての文字列をコンテンツツリーへ移行

実際には使われていないと気づけない翻訳と、実績の3分の1だけを黙って評価する検査を発見し、利用者向け文字列をすべてPythonから3言語874ファイルのコンテンツツリーへ移した。

すべての静的検査ルールをエラーに設定

Python静的検査の全ルールをエラーとして有効にし、1,815件の指摘を0件まで解消。少数の例外には理由を書き、うち2件はコメントだけでなく検査によって根拠を守った。

生成器に981テストと92%カバレッジを適用

警告を失敗扱いにし、分岐カバレッジ92%を下限として生成器に981件のテストを適用。空のファイルから全構築処理を2回実行し、2回目に何も変わらないことを要求して冪等性を検証した。

採用と船員キャリアの管理画面を構築

採用と船員の経歴管理の作業領域を構築。記録された実装時点では、求人、応募、資格証明、職位の昇進、乗船期間を扱う48テーブルと151のストアド関数を含む。

投稿・フィード・グループの交流機能を構築

投稿、フィード、グループ、メンション、フォロー関係、記念日のまとめを含むソーシャル機能を、製品の他の部分と同じ関数中心のデータ層上に構築した。

ロイヤルティ残高、リーグ、バッジを構築

31テーブルの台帳と67個の関数に、リーグ、バッジ、特典交換ショップを備えるロイヤルティ・評判システムを構築し、残高の行ロックで二重使用の発生余地を塞いだ。

異常時拒否のレート制限・不正利用対策を構築

Redisによる22個のレートリミッター、Cloudflare Turnstile、リクエストの冪等性、オリジン制限を備え、異常時は拒否する不正利用対策を構築し、呼び出し元を無条件に信頼せずボットや大量アクセスを排除した。

独自のエラー監視・トレースを構築

ペイロードの無害化、急増・再発の検知、シンボル解決、擬似ハートビートを備える独自のエラー監視とOpenTelemetryトレースを外部購入せず構築し、11画面の運用ビューから扱えるようにした。

遅い処理をRiverジョブキューへ移行

時間のかかる処理をリクエスト経路からRiverジョブキューへ移し、15個のワーカーモジュール、8件の定期タスク、20件のpg_cronジョブにより、背後の作業が続く間も応答を返せるようにした。

Stripe・アプリ内購入の利用権限を構築

StripeとApple・Googleのアプリ内購入に対応する決済・利用権限層を構築し、サーバーで検証する11テーブルのアクセスモデルによって、一覧、検索、エクスポートの利用を制御した。

Go・SQL・ブラウザーのテスト群を構築

981件のGoテスト、543件のフロントエンド・ブラウザーテスト、494件のSQL動作テストを持つ多層の自動テスト群を構築し、ミューテーションテスト、プロパティベーステスト、アクセシビリティの品質ゲートを加えた。

IPTV用Djangoアプリを構築・保守

IPTVプラットフォームのDjango Webアプリケーションを開発・保守し、新機能の提供と性能・安定性の改善を行った。

nonce付きCSPでアプリケーションを堅牢化

nonce付きCSP、HSTS、SameSite Cookie、最小権限のデータベースロール、サーバー側の利用権限再検証により、アプリケーションを堅牢化した。

カタログに基づくディープマージを実装

保存済みJSON設定にカタログに基づくディープマージを実装し、スキーマ変更時のキー欠落によるクラッシュを防いだ。

切り替えと配信ログを備えるメール機能を構築

メール送信をプラットフォームの共通機能として構築。3つのプロバイダーによるフェイルオーバー、配信Webhook、送信・配信ログ、テンプレート、キャンペーンを備え、送信手段がなければ起動を拒否する検査を設けた。

データベースを起点にAPI契約を生成

データベースを起点に、OpenAPI、44,076行の型付きTypeScriptクライアント、61個のモックハンドラー、UIが適用する制限を生成し、各段階に不一致で失敗する検査を設けた。

データ分析レポートシステムを開発

PythonでPDFレポートを作成するデータ分析レポートシステムを開発し、ソフトウェアの四半期売上を400%増加させた。

30件以上のGIS開発を構築からサポートまで主導

PostgreSQL、Bash、Python、JavaScript、GDAL、ArcGIS、PostGIS、Mapboxの専門知識を活用し、30件を超えるGISプロジェクトの開発・デプロイ・サポートを主導した。

フルスタックのGIS地図開発を指揮

PostgreSQL、Mapbox、ReactJS、NodeJSを統括してフルスタックのGIS地図開発を指揮し、統合されたソリューションを提供した。

マニュアル付きPower BIプロジェクト8件を提供

Microsoft Power BIをNodeJS APIとPython FastAPIに連携させ、データ分析と可視化のための8件のPower BIプロジェクトを、詳しいマニュアルとともに納品した。

ETL/ELTパイプライン6本を設計・運用

Google BigQuery、MSSQL、PostgreSQL、Shellスクリプト、PL/pgSQL、Transact-SQLを用いて6本のETL/ELTパイプラインを設計・実装・管理し、Python APIで効率的に処理できるようデータを統合した。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

C上にSwift 6の厳密並行処理を採用 · 522行のQR符号器を同梱 · データベース駆動の履歴書生成器を構築 · プラットフォームの基礎技術方針を策定 · PostgreSQL関数中心のデータ層を設計 · APIを100%の一致でHuma v2へ移行 · 利用者の組織コンテキスト切り替えを構築 · PostgresからAPIへのJSONパススルーを設計 · システムの境界を明確にした海事プラットフォーム

{ ソフトウェアエンジニアリング }

フロントエンド開発

実際に人が使う画面をつくる

読み込みが速く、スマートフォンでも使え、お客様がどのブラウザーを使っていても同じように動く画面を提供します。

48トークンのmacOSデザインシステムを構築

プラットフォーム標準のガラス素材を使った48トークンのデザインシステムを構築し、マジックナンバー、固定文字サイズ、「動きを減らす」設定を無視するアニメーションを拒否する静的検査を作成した。

全17アイコンに読み上げ用の名前を付与

送信ボタンが「arrow up circle, button」と読み上げられる問題を受け、アイコンだけの全操作要素にスクリーンリーダー用の名前を付け、アイコンから8行以内のラベルを必須とする静的検査を作成した。

10,242行の品質ゲート自体に静的検査を適用

リポジトリ最大のコード群でありながら検査対象外だった品質ゲートのJavaScript 10,242行に整形・静的検査を適用し、13件の指摘を抑制せずに修正した。

CSSバンドルを87 KBから31 KBに削減

base64フォントを取り出してスタイルシートバンドルを87 KBから31 KBに削減し、毎回デプロイされながら参照元がない22ファイル756 KBも除去した。

637行の印刷スタイルシートを構築

ブラウザーがダーク表示を優先する場合に白地に白で印刷される行内見出しなど、文書化した4つの描画エンジンの挙動に対処する637行の印刷用スタイルシートを構築した。

「動きを減らす」設定を正しく反映

クラス付きルールには、全要素のトランジションを無効にするルールが詳細度で負けるため、「動きを減らす」設定でもアニメーションを続ける11個のセレクターを修正し、設定が正しく効くようにした。

ライトテーマに欠けた層を発見

両テーマで各多層面に同じ数の層が描画されることを検証し、ライトテーマに作成当初から全面オーバーレイ層が欠けていたことを発見した。

配色を28色に限定

配色を文書化した28色に限定し、描画されるのに未記録、記録されるのに未使用、測定誤り、リテラルによる別表記、既存色との知覚的距離が0.02以内、という場合に失敗する静的検査を設けた。

自動検査が見逃すアクセシビリティの問題

pointer-eventsで操作を隠してもTab移動に残るフッターリンク8個、3ページの奥付を切り落とすスクロールタイムライン、105ルール中70しか実行しない検査エンジンなど、ブラウザーが正常と報告した11件のアクセシビリティ不備を発見・修正した。

32ページを両テーマでアクセシビリティ検査

両配色テーマの代表32ページにWCAG 2.2 Level AA検査を実装し、適合記録とAAA基準1項目の検査を備え、基準1.4.13の部分的な達成を明示した。

27種類の文字サイズを6段階の体系に統一

隣り合うサイズの差が0.6%しかないものもあった27種類の文字サイズを、長3度比による6段階の体系へ置き換え、28種類目を拒否する静的検査を作成した。

30分も待たずに済む品質検査

最長の検査から順に、最大4並列のワーカープールへ割り当て、ブラウザーを使うサイト品質ゲートを1,636秒から615秒に短縮。名前順では224秒に対して320秒を要するという測定を踏まえた。

3言語の多言語Hugoサイトを構築

53個のpartialを含む108個のHugoテンプレートで多言語静的サイトを構築。1つのコンテンツツリーから3言語・4形式で各ページを公開し、同じツリーから7つの専門分野別サブドメインも構築した。

プラットフォームの運用管理画面を構築

申請、モデレーション、機能フラグ、キャッシュ、診断を扱う約40本の管理ルートと128個のコンポーネントからなる運用管理画面を構築し、データベースへ直接アクセスせずにプラットフォームを運用できるようにした。

Hugoサイトを保守し製品のUIを改善

従来のHugo静的Webサイトを保守・改善しつつ、主力の資産管理製品のUI/UX改善にも携わった。

英語・デンマーク語のi18nシステムを構築

458個の名前空間ファイルに言語ごと12,027件のメッセージを持つ英語・デンマーク語の国際化基盤を構築し、用語と1ファイル400行の上限を静的検査で守った。

製品のUIとUX全体を設計

一覧グリッド、カード・表の切り替え、リアルタイムの要件検証、パンくずナビゲーションを含む、製品のUI・UX全体を設計した。

iOS 26「Liquid Glass」デザインシステムを設計

iOS 26の「Liquid Glass」を基にデザインシステムと正本となるコンポーネント一覧を設計し、UIのばらつきを静的検査で防いだ。

Progressive Web Appに完全対応

next-pwa経由のWorkboxランタイムキャッシュで、インストール可能かつオフラインでも使える、完全なProgressive Web App対応を実装した。

Next.js 16でSSR・SSG・CSRの画面を構築

SSR、SSG、CSRを意図的に使い分けたNext.js 16フロントエンドを構築。再利用可能な事前取得付きサーバーページ生成処理に移行した認証済みページから、N+1の連鎖的なデータ取得を除去した。

PhotoshopからHTMLへWebサイト30件を設計

PhotoshopのデザインをHTMLへ変換し、独自性と柔軟性を備えたWebサイト30件を設計した。

HTML/HTML5でWebアプリケーション100個を開発

HTML/HTML5、CSS/SCSS、Django、WordPress、Joomlaを用いて100個のWebアプリケーションを開発し、多様なオンライン展開を支えた。

ユーザーインターフェースの設計に貢献

ユーザーインターフェースの設計に携わり、直感的で見やすく、使いやすい画面を実現した。

UI/UX改善200件を提案

GIS地図アプリケーションに200件のUI/UX改善を提案し、機能強化を通じて売上を10倍に伸ばした。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

macOSアプリのイベントパイプラインに上限を設定 · Swift 6でネイティブmacOSクライアントを構築 · プラットフォームの基礎技術方針を策定 · 利用者の組織コンテキスト切り替えを構築 · システムの境界を明確にした海事プラットフォーム · フルスタックのGIS地図開発を指揮

{ ソフトウェアエンジニアリング }

フルスタックのプロダクト開発

構想から完成まで、使える製品に仕上げる

3社の間で受け渡す代わりに、1つのチームがアイデアのスケッチから、お客様がログインして使える製品まで担当します。

企業所有権の申請・審査を構築

利用者による企業所有権の申請から管理者の審査までを実装し、承認時に編集権限を決める認可グラフが更新される、一連の仕組みを構築した。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

Swift 6でネイティブmacOSクライアントを構築 · データベース駆動の履歴書生成器を構築 · 3言語の多言語Hugoサイトを構築 · 採用と船員キャリアの管理画面を構築 · 投稿・フィード・グループの交流機能を構築 · プラットフォームの運用管理画面を構築 · IPTV用Djangoアプリを構築・保守 · プラットフォームの基礎技術方針を策定 ·

Next.js 16でSSR・SSG・CSRの画面を構築 · システムの境界を明確にした海事プラットフォーム · 地図製品を最初から運用まで支える · HTML/HTML5でWebアプリケーション100個を開発 · 30件以上のGIS開発を構築からサポートまで主導 · フルスタックのGIS地図開発を指揮 · GIS地図アプリの開発を主導し売上を10倍に

{ ソフトウェアエンジニアリング }

Webサイト・CMS開発

自社で更新できるWebサイト

表示が速く、検索で見つけやすく、文章を変えるたびに料金が発生せず、自社の担当者が編集できるサイトをつくります。

読みやすいTor onionミラーを公開

engineerで始まる読みやすいTor onionアドレスでサイトのミラーを公開。鍵はホスト外で生成し、このリポジトリにもノートパソコンにも置かず、4つの入口のうち計測しない唯一の入口にした。

2点の図からビジュアルアイデンティティを構築

手描きの2点の図から会社のアイコンとファビコン一式を構築し、すべての派生素材をスクリプトで再生成できるようにして、元ファイルより先に派生ファイルがコミットされた場合は検査で失敗させた。

PDF/UA-1プロファイルの106項目中106項目に合格

9文書のPDF/UA-1検証を記録し、設定済みアクセシビリティプロファイルの検査106項目中106項目に合格した一方、アーカイブ版は146規則のうち1項目で不合格だった。

更新日時が共通のサイトマップを修正

エクスポートが毎回すべてのファイルを書き直すことを確認し、git履歴から更新日を取得するよう変更。176件のURL中172件が同じ更新日時だったサイトマップを修正した。

963メタデータブロックを1グラフに統合

会社の説明を1,671回繰り返していた匿名の構造化データ963ブロックを、安定したオリジン識別子から生成する16型の連結グラフ1つへ置き換えた。

GeminiとGopherへサイトを複製

同じウェブサイトのソースからGemini版とGopher版を生成する仕組みを構築。この実績では、その時点で各形式777文書を記録し、形式の追加によって変更されたHTMLはゼロバイトだった。

SQLiteエクスポートから495ページを生成

読み取り専用のSQLiteエクスポートから3言語495件の実績ページを生成し、ページのアドレスをデータとして定義することで、文章を直してもURLが変わらずリンクが切れないようにした。

自社サーバーでコードを管理

Web管理画面を持たず、初期状態で非公開のSoft Serveによる自社gitホスティングを導入。SSHポートを踏み台ホストの背後のループバックに限定し、入口のページを手作業の複製ではなく主サイトのビルド成果物にした。

Web開発にSEOとコンテンツを統合

独自のWebサイト開発・デザインにSEO、コンテンツ戦略、コピーライティングを組み合わせ、顧客のWebでの成果につなげた。

Apache・NginxでWebサイト40件を管理

Apache・Nginxを使用するUbuntu Linuxホスティングサーバー上で40件のWebサイトを管理し、高い可用性と性能を維持した。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

CSSバンドルを87 KBから31 KBに削減 · 637行の印刷スタイルシートを構築 · 「動きを減らす」設定を正しく反映 · 自動検査が見逃すアクセシビリティの問題 · 32ページを両テーマでアクセシビリティ検査 · 3言語の多言語Hugoサイトを構築 · Hugoサイトを保守し製品のUIを改善 · IPTV用Djangoアプリを構築・保守 · Progressive Web Appに完全対応 · Next.js 16でSSR・SSG・CSRの画面を構築 · PhotoshopからHTMLへWebサイト30件を設計 · HTML/HTML5でWebアプリケーション100個を開発

{ ソフトウェアエンジニアリング }

AIエージェントへの対応

お客様がAIに尋ねたときにも見つかるようにする

お客様のAIアシスタントがサイトやアプリを見つけ、読み取り、操作できる状態を、最初の約束だけで済ませず、リリースごとに検証します。

51件の文書化した施策として開発を運営

54件の計画文書にわたる51件の施策としてサイト開発を進め、それぞれに未完了項目、測定結果、採用しなかった判断を記録。AIエージェント対応で見つかった8件中6件も、理由を記して見送った。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

更新日時が共通のサイトマップを修正 · 963メタデータブロックを1グラフに統合 · 32ページを両テーマでアクセシビリティ検査 · 3言語の多言語Hugoサイトを構築

{ ソフトウェアエンジニアリング }

AIエージェント用ツール・MCPサーバー

今あるソフトウェアをAIから使えるようにする

すでに動いている部分は書き直さず、既存のソフトウェアをAIエージェントから呼び出せるツールとして公開します。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

プラットフォームの運用管理画面を構築 · 遅い処理をRiverジョブキューへ移行 · データベースを起点にAPI契約を生成 · APIを100%の一致でHuma v2へ移行 · PostgresからAPIへのJSONパススルーを設計

{ ソフトウェアエンジニアリング }

GIS・地理空間ソリューション

問いに答える地図をつくる

船舶、車両、区画、センサーなど、場所に関わる情報を、検索・計測・共有できる地図にまとめます。

GISプロジェクト30件を提供まで管理

30件のGISプロジェクトの実行を管理し、各分野で新しいソリューションを提供するための統率を担った。

予測と投資戦略を最適化

データに基づくGISソリューションを通じて、11社の電力系統運用事業者の予測と投資戦略を最適化し、業務効率を向上させた。

GISデータパイプライン20本の提供を自動化

20本のGISデータパイプラインとアプリケーションデータのETL処理の提供を自動化し、インフラ自動化とレポート作成を効率化した。

GitHub ActionsでGIS SaaSのデプロイを自動化

GitHub Actions CI/CD、Python、Bash、SQLで、GIS SaaSアプリケーションのデプロイ、データ処理、レポート作成を自動化した。

PostgreSQL・MS SQL・BigQueryのデータウェアハウス100個を運用

主にGIS・時系列データを扱うPostgreSQL、MS SQL、Google BigQueryのデータウェアハウス100個を設計・構築・管理し、性能と拡張性を最適化した。

電力・GISデータレポート500件を公開

詳細な調査と問題解決に基づき、地理情報・時系列の大規模データを正確に読み解く、電力・GISデータ分析レポート500件を公開した。

地理情報データの問題1,000件を解決

GDAL、ArcGIS、PostGIS、Mapbox、QGIS、SQL（PL/pgSQL、Transact-SQL）、Bashを用いて地理情報・時系列データの問題1,000件を解決し、大規模データ処理の品質を確保した。

PostgreSQLでGIS地図を10倍に高速化

データの切り替えを含め、データベースをMSSQLからPostgreSQLへ計画的に移行。処理とデータセキュリティを最適化し、地図アプリケーションの性能を10倍に向上させた。

地理情報パイプラインを50倍に高速化

創業者のUtiligizeでの業務で、PostgreSQL、MS SQL、Google Cloud BigQueryのSQLとデータモデルを変更し、地理データパイプラインの50倍の改善を記録した。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

地図製品を最初から運用まで支える · 30件以上のGIS開発を構築からサポートまで主導 · フルスタックのGIS地図開発を指揮 · GIS地図アプリの開発を主導し売上を10倍に

{ ソフトウェアエンジニアリング }

UI/UX設計・デザインシステム

使い方が自然にわかるようにする

配置、文言、操作の流れを整え、サポートに
電話しなくても目的を果たせるようにします。

利用者を責めないエラーメッセージに改訂

実際にはプロバイダーがアプリ専用パスワードを要求していたのに、サインイン拒否を利用者の入力誤りとするエラー表示を受け、文書化した文体基準に従ってメッセージを書き直し、プロバイダー名を明示するテストを加えた。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

2点の図からビジュアルアイデンティティを構築 · 48トークンのmacOSデザインシステムを構築 · 全17アイコンに読み上げ用の名前を付与 · Swift 6でネイティブmacOSクライアントを構築 · PDF/UA-1プロファイルの106項目中106項目に合格 · 637行の印刷スタイルシートを構築 · 「動きを減らす」設定を正しく反映 · ライトテーマに欠けた層を発見 · 配色を28色に限定 · 自動検査が見逃すアクセシビリティの問題 · 32ページを両テーマでアクセシビリティ検査 ·

27種類の文字サイズを6段階の体系に統一 · Web開発にSEOとコンテンツを統合 · Hugoサイトを保守し製品のUIを改善 · 製品のUIとUX全体を設計 · iOS 26「Liquid Glass」デザインシステムを設計 · PhotoshopからHTMLへWebサイト30件を設計 · ユーザーインターフェースの設計に貢献 · UI/UX改善200件を提案

{ ソフトウェアエンジニアリング }

国際化・ローカライズ

複数の言語で使えるようにする

日付、通貨、名前の並び順まで含め、販売先のそれぞれの市場で自然に読めるソフトウェアにします。

言語間の数値照合を構築

英語原文にある数値の欠落と、対象言語に合わない表記で失敗する言語間コンテンツ検査を構築し、指標が抜けたデンマーク語説明2件と英語式引用符を使うウクライナ語16箇所を検出した。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

すべての文字列をコンテンツツリーへ移行 · データベース駆動の履歴書生成器を構築 · GeminiとGopherへサイトを複製 · SQLiteエクスポートから495ページを生成 · 32ページを両テーマでアクセシビリティ検査 · 3言語の多言語Hugoサイトを構築 · 英語・デンマーク語のi18nシステムを構築

{ ソフトウェアエンジニアリング }

専用AIエージェント開発

自社の業務を理解するAIアシスタント

使用中のシステムに接続し、自社の専門家が用意した事例でテストし、自社の担当者が監督できる、特定のモデルに縛られないエージェントをつくります。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

6収集処理が省略した入力を明示 · 検査自体のテストを作成 · 22静的検査のコミットゲートを構築 · 独自のエラー監視・トレースを構築 · 268検査を持つリポジトリの検査基盤を構築

{ データとデータベース }

データベース設計・モデリング

事業に合うデータの置き場をつくる

実際の業務に合った構造を設計し、将来必要になる問いにも答えられるようにします。

PostgreSQL内に独自の分析機能を構築

自動的に古いパーティションを削除するイベントテーブルと47個のPostgreSQL関数による独自の製品分析を構築し、定期変更するソルトで仮名化し、利用者の同意に基づいて動作させた。

成長するデータモデルの時系列順キー

B-treeインデックスの断片化を抑え、クエリを高速化するため、PostgreSQL 18の時系列順UUID v7をエンティティキーとして採用した。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

データベース駆動の履歴書生成器を構築 · 採用と船員キャリアの管理画面を構築 · ロイヤルティ残高、リーグ、バッジを構築 · 遅い処理をRiverジョブキューへ移行 · Stripe・アプリ内購入の利用権限を構築 · 34のPostgresスキーマに348テーブルをモデル化 · カタログに基づくディープマージを実装 · PostgreSQL関数中心のデータ層を設計 · PostgresからAPIへのJSONパススルーを設計 · システムの境界を明確にした海事プラットフォーム · PostgreSQL・MS SQL・BigQueryのデータウェアハウス100個を運用 · ETL/ELTパイプライン6本を設計・運用

{ データとデータベース }

データベース管理（DBA）

データベースをきちんと保つ

会社全体を支えるデータベースについて、稼働、バックアップ、気づかないうちに進む劣化まで責任を持って管理します。

1,022件のマイグレーションを差分検査で保護

新規インストールと、インストール後に全マイグレーションを適用する2通りでデータベースを構築し、不一致で失敗するCIゲートを設け、1,022件のマイグレーションにわたるスキーマの整合性を保った。

バックアップと災害復旧をコードで実装

データベースのバックアップ保持をコードで設定し、復旧可能な状態を監査して復元手順を文書化。残る不足を障害時に初めて知ることがないよう明記した。

PL/pgSQL ETL/ELTパイプライン600本を構築

複数のPostgreSQL開発・本番環境にまたがる複雑なデータ処理を効率化するため、PL/pgSQLによるETL/ELTパイプライン600本を構築した。

PythonとPL/pgSQLでデータ処理を自動化

Shellスクリプト、PL/pgSQL、Python、Transact-SQLでデータ処理を自動化し、生産性と効率を向上させた。

Barmanで重要データのバックアップ100件を自動化

Barman、Google Cloud、Bash、Pythonで重要データのバックアップ100件を自動化し、データベース間のデータ整合性を確保した。

PostgreSQLを10倍に高速化

インデックス、パーティション、クエリを計画的に最適化してPostgreSQLを10倍に高速化し、ユーザー、テナント、地理空間、電力時系列データの処理効率を改善した。

Linuxサーバーを守る1,000件のRBACルールを作成

開発者、アプリケーションインスタンス、PostgreSQL、MS SQL、その他のLinuxサーバーに1,000件のRBACルールを実装して不正アクセスを防ぎ、データセキュリティを強化。Ansibleによる自動化として記録した。

UbuntuでPostgreSQL・MS SQLサーバー10台を運用

Ubuntu Linux VPS上にPostgreSQL・MS SQLサーバー10台を設計・デプロイ・保守し、性能と信頼性を維持した。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

PostgreSQL・MS SQL・BigQueryのデータウェアハウス100個を運用

{ データとデータベース }

データベース性能改善

遅いところを速くする

年々重くなってきたシステムのクエリと不足するインデックスを調べ、効果のある箇所を改善します。

30日分の電力データを180 msで処理

Python、SQL、Bash + Jqで時間別の電力消費量を集計する複数の実装を構築。Pythonの測定では、30日分のJSONLデータセット1つを180 msで処理した。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

成長するデータモデルの時系列順キー · PostgreSQLを10倍に高速化 · PostgreSQL・MS SQL・BigQueryのデータウェアハウス100個を運用 · PostgreSQLでGIS地図を10倍に高速化 · 地理情報パイプラインを50倍に高速化

{ データとデータベース }

データベース移行・モダナイゼーション

古いシステムから乗り換える

業務を止めずに、老朽化したシステムからサポートのある環境へデータを移します。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

すべての文字列をコンテンツツリーへ移行 · 1,022件のマイグレーションを差分検査で保護 · PostgreSQLでGIS地図を10倍に高速化

{ データとデータベース }

データパイプライン開発（ETL/ELT）

データを必要な場所へ自動で届ける

月末に誰かが表計算ファイルをコピーしなくても、決まった時刻に必要な場所へ数値が届く仕組みをつくります。

GitHub・GitLab CI/CDで4,000時間を削減

GitHub、GitLab、Bash、PythonによるCI/CDを導入し、データ分析とソフトウェア開発の工程を効率化して4,000時間を削減した。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

SQLiteエクスポートから495ページを生成 · PostgreSQL内に独自の分析機能を構築 · 地図製品を最初から運用まで支える · PL/pgSQL ETL/ELTパイプライン600本を構築 · PythonとPL/pgSQLでデータ処理を自動化 · 30件以上のGIS開発を構築からサポートまで主導 · GISデータパイプライン20本の提供を自動化 · GitHub ActionsでGIS SaaSのデプロイを自動化 · ETL/ELTパイプライン6本を設計・運用 · 地理情報データの問題1,000件を解決 · 地理情報パイプラインを50倍に高速化 · 30日分の電力データを180 msで処理 · 14部門を支える共通インフラ計画

{ データとデータベース }

データウェアハウス設計

部門ごとに食い違う数字をそろえる

営業、運用、経理の数値をまとめ、同じ会議に部門ごとに異なる答えを持ち寄る状況を解消します。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

PostgreSQL・MS SQL・BigQueryのデータウェアハウス100個を運用 · ETL/ELTパイプライン6本を設計・運用 · 地理情報パイプラインを50倍に高速化

{ データとデータベース }

データ分析・BIダッシュボード

今何が起きているかを把握する

意思決定を左右する少数の指標を示すダッシュボードや
レポートを、依頼しなくても更新される形で提供します。

時間管理・レポートシステムを設計

大きな変更を加えずに何年も本番利用された、時間管理・レポートシステムを設計した。

分析と定期的な進捗報告を提供

分析と定期的な進捗報告をCEOへ届け、技術指標と開発状況を意思決定につなげた。

社内初のオブザーバビリティダッシュボードを構築

社内初のオブザーバビリティダッシュボードを開発・公開し、オフィスの大型テレビでシステム性能とデータをリアルタイムに把握できるようにした。

読み手の反応に基づくブランド発信

市場調査と読み手からの意見を基に、Engineer ApSのブランド発信と働きかけを整えた。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

PostgreSQL内に独自の分析機能を構築 · データ分析レポートシステムを開発 · GitHub・GitLab CI/CDで4,000時間を削減 · 予測と投資戦略を最適化 · 電力・GISデータレポート500件を公開 · マニュアル付きPower BIプロジェクト8件を提供 · 30日分の電力データを180 msで処理

{ データとデータベース }

データガバナンス・品質管理

数字を信頼できるようにする

取り込み時に不適切なデータを検出する規則と検査を設け、各数値の責任者と出所を明らかにします。

追跡せずに13アドレスを測定

Cookie、トラッカー、第三者サービスを使わず、Webサーバーのアクセスログのみから13アドレスを測定。書き込み時に接続元ネットワークをマスクし、21の日別集計と20の月別分析軸からなる永続アーカイブへまとめた。

公開するすべての部品のライセンスを把握

毎回のビルドで644個のRust crateのライセンス互換性を監査し、1つのcrateのライセンスを書き換えた場合と、再生成せずコアの固定リビジョンを変えた場合に検査が働くことを実証した。

検査自体のテストを作成

正常入力しか与えない検査は、何も読まずに正常と報告するようになり得ると確認し、検査自体のテストを作成。誤字を埋めて綴り検査の検出を確かめ、テスト内の固定値ではなくデータベースからID範囲を取得した。

認証情報漏出後に対象範囲の規則を記述

構成変更によって別会社の資産一覧、ファイアウォール許可、文章が混入したことを受け、リポジトリの対象範囲を定める規則を記述し、隔離した残存物も除外せず秘密情報検査の対象に残した。

業務要件を収集・分析

業務要件を収集・分析し、データガバナンス基準に沿った、実装可能な機能とユーザーストーリーへ落とし込んだ。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

言語間の数値照合を構築 · すべての文字列をコンテンツツリーへ移行 · 963メタデータブロックを1グラフに統合 · PostgreSQL内に独自の分析機能を構築 · 34のPostgresスキーマに348テーブルをモデル化 · カタログに基づくディープマージを実装 · PostgreSQL関数中心のデータ層を設計 · PL/pgSQL ETL/ELTパイプライン600本を構築 · 電力・GISデータレポート500件を公開 · 地理情報データの問題1,000件を解決 · 14部門を支える共通インフラ計画

{ データとデータベース }

AIが使えるデータの整備

AIが実際に利用できるデータにする

AIエージェントが記録にアクセスでき、顧客ごとの情報が一本化され、
権限を超えた情報が表示されず、回答の出所をたどれる状態にします。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

963メタデータブロックを1グラフに統合 · SQLiteエクスポートから495ページを生成 · 1,022件のマイグレーションを差分検査で保護 · 34のPostgresスキーマに348テーブルをモデル化 · ETL/ELTパイプライン6本を設計・運用 · 地理情報パイプラインを50倍に高速化

{ クラウド・インフラ・DevOps }

クラウドインフラ・移行

ソフトウェアを動かす場所を整える

実際の使用量に合わせてサーバーを構成し、月額費用を事前に把握
したうえで、クラウドへの移行もクラウドからの移転も進めます。

2台目のサーバーをコードで構築

別のクラウド事業者で2台目のサーバーの構築を自動化。先にファイアウォールを作成し、起動時から保護されるようにし、両事業者のファイアウォールをREST APIから直接操作して、第三者のcollectionへの依存を避けた。

PodmanとQuadletでコンテナを運用

Dockerの既定のブリッジネットワークでのポート公開がUFWルールを迂回する場合があるため、systemd配下のPodman・Quadletでコンテナ基盤を構築。デプロイにはrootではなく、固定コマンド1つを使う非特権ユーザーを用いた。

リポジトリで定義するインフラ

12,065行のYAMLにわたる19本のAnsible プレイブックと34個のロールで自社インフラを構築し、すべてのプレイを冪等にして、稼働中のホストを宣言した状態へ収束させた。

Azureへのプラットフォーム導入を一貫して担当

プラットフォームのAzureへのデプロイを最初から最後まで担当し、開発・ステージング・本番環境のリリースを管理した。

Azureインフラをコードとして構築

Bicepで、開発・ステージング・本番の各環境に、Container Apps、PostgreSQL Flexible Server、Front Door/WAF、ネットワークを含むAzureインフラをコードとして構築した。

Google Cloud・AWSでDockerアプリ20個を運用

Dockerでコンテナ化した20個のアプリケーションをデプロイ・保守。PodmanとKubernetesで問題を調査し、Google Cloud・AWS上のRアプリケーションも管理した。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

小型サーバー上の公開サービス · 基幹インフラ全体を一から構築 · AWSの外へ移し、インフラを整理

{ クラウド・インフラ・DevOps }

コードによるインフラ管理

バージョン管理された設定から再構築できるサーバー

構成全体をファイルに記述し、特定の人の記憶だけに
頼らず、まったく同じ環境を再現できるようにします。

gitホスティングCIを試し2不具合を発見

gitホスティング側のCIフックを予行実行し、ファイルを読むだけでは到達できない2件の不具合を発見。解決不能な引数を入力へ渡すフォールバックと、検査先に引き継がれ19テストを失敗させるgit自身の環境変数を修正した。

すべての運用秘密情報をホストごとに分離

2ホストで1つのデッドマンズスイッチを共有すると2つのスイッチより監視能力が下がり、バックアップのパスフレーズ共有では、リポジトリが別々でも、そのパスフレーズを入手し、両リポジトリの暗号化データと鍵ファイルを読める相手が2ホストのバックアップを復号できることを確認し、ホストごとの運用秘密情報4つを分離した。

全ユニットのsystemdサンドボックス露出を低減

この基盤が導入するすべてのユニットのsystemdサンドボックスの露出度を低減。デッドマンズスイッチのサービスを9.6 UNSAFEから1.5へ、インターネット公開のgitホスティングを8.3 EXPOSEDから1.5へ改善し、3つのユニットテンプレートで黙って無視されていた設定名の誤記を受け、収束時の構文検査を加えた。

20レコードのDNSゾーンをコードで照合

20レコードのDNSゾーンをCloudflare APIと宣言的に照合し、監査とBIND形式の書き出しを別の実行入口として提供。CDNプロキシは構築後、プライバシーの観点から再び無効にした。

秘密情報の1秒の事前検査を追加

本番実行が9番目のタスクで停止し、すでにスワップ設定が稼働ホストへ書き込まれた事態を受け、運用担当者のローカルな秘密情報に対し、収束処理と同じコードを約1秒で実行する事前検査プレイを追加した。

Ansibleチェックモードを実態に即した検証に

Ansibleのcommandモジュールが--checkでコマンドを省略しながら成功を返すため、ホストから得ていない値で判断していた6個の検査を修正し、プラットフォーム全体のチェックモードが実態を伝えるようにした。

デッドマンズスイッチによるホスト監視を構築

メモリーとディスクが正常な間だけ通知を送るデッドマンズスイッチ監視を構築し、ホストの悪化を通知の停止で知らせるようにした。また、正しく測っていた「available」を「free」と記す変数名を6件発見。464 MBのホストでは両者に1桁の差があった。

復元方法を明示したバックアップ自動化

resticのバックアップを、保持期間、整合性検査、毎月の復元テストとともに自動化し、ストレージの有効化と実環境での復元検証を別の運用手順として記録した。

SSHの24設定項目を検証・堅牢化

SSHの24設定項目を、候補ファイル、統合済み設定、デーモン自身の読み戻しによる3段階で検証するよう堅牢化。読み戻しによって、稼働中のサーバーが24項目のうち2項目を黙って上書きしていたことを検出した。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

読みやすいTor onionミラーを公開 · 2台目のサーバーをコードで構築 · 自社サーバーでコードを管理 · 小型サーバー上の公開サービス · リポジトリで定義するインフラ · バックアップと災害復旧をコードで実装 · Azureインフラをコードとして構築 · Linuxサーバーを守る1,000件のRBACルールを作成

{ クラウド・インフラ・DevOps }

DevOps・CI/CD自動化

落ち着いて変更を届ける

変更を検証し、リリースし、元に戻せる仕組みを整え、金曜の深夜ではなく火曜の午後にお客様へ届けられるようにします。

4リポジトリで共通の履歴基準を維持

4つのリポジトリに対し、Conventional Commits、絵文字禁止、帰属トレーラー禁止という共通の履歴基準をコミットメッセージフックで適用し、作業方法を定める48件の指示文書とともに運用した。

テスト群を8.9秒に短縮

メインスレッドを測定し、4,017サンプル中3,989サンプルを費やす2つの呼び出しを除去。6テストで60秒の制限を超えていたテスト群を、135テストが8.9秒で通る状態に改善した。

フックと品質ゲートの不一致を発見

同じ検査を行うという文書の記述に反し、コミットフックと品質ゲートが異なる検査を行っていたことを、一覧を比較するテストで発見。手動でしか動かなかった5個は履歴書の文章を検査するものだった。

22静的検査のコミットゲートを構築

HTML、CSS、JavaScript、Python、YAML、Markdown、Shell、リンク、綴り、秘密情報、組版を対象に、1行で説明できる検査22個と詳しい説明が必要な検査5個からなるコミットゲートを構築し、警告扱いや行内抑制を認めない方針にした。

継続的なセキュリティ検査を運用

コード検査、DAST、依存関係・脆弱性検査、SBOM生成、秘密情報検査、SHA固定のActionsと、21件の文書によるセキュリティ監査を組み合わせた継続的なセキュリティ活動を確立した。

268検査を持つリポジトリの検査基盤を構築

登録済みコミット検査268件、静的検査ルール277件、独自ESLintルール15件からなるリポジトリの検査基盤と、検査自体を試す146件のテストを構築し、人のレビューだけに頼らずビルドで基準を守るようにした。

チームの協働を自動化

チームの協働、パスワード管理、タスク・時間管理を自動化し、プロジェクト実績を半自動で紹介する仕組みも構築して、チームの生産性を高めた。

6言語に警告ゼロの基準を設定

Go、TypeScript、SQL、Python、Shell、Markdownの6言語に警告ゼロの品質基準を設け、pre-commitフックで適用した。

go-taskの自動化ターゲット578個を作成

ネイティブ、Docker、HTTPSの開発モード、静的検査、テスト、データベース、デプロイをカバーする、go-taskの自動化ターゲット578個を作成した。

GitHub Actions CI/CDパイプラインを構築

本番フロントエンド用のdistrolessイメージと複数環境間の昇格を備えた、GitHub ActionsのCI/CDパイプラインを構築した。

CI/CDの効率化で4,000時間を削減

ソフトウェア開発パイプラインを自動化してCI/CD工程を効率化し、4,000時間を削減した。

Dockerアプリ100個のSSL/TLS証明書を自動化

Dockerアプリケーション100個のSSL/TLS証明書の作成を自動化し、Ubuntu Linuxホスト上での安全な接続を確保した。

GrafanaとPrometheusで運用リスクを低減

GrafanaとPrometheusの監視ダッシュボードを導入して運用リスクを低減し、システムの信頼性を高めた。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

公開するすべての部品のライセンスを把握 · 未使用だったコアの半分の機能を実装 ·

全FFI呼び出しを検証する構文解析器を作成 · gitホスティングCIを試し2不具合を発見 ·

検査自体のテストを作成 · すべての静的検査ルールをエラーに設定 ·

生成器に981テストと92%カバレッジを適用 · 10,242行の品質ゲート自体に静的検査を適用 ·

更新日時が共通のサイトマップを修正 · GeminiとGopherへサイトを複製 ·

30分も待たずに済む品質検査 · PodmanとQuadletでコンテナを運用 · 自社サーバーでコードを管理 · 秘密情報の1秒の事前検査を追加 · Ansibleチェックモードを実態に即した検証に · リポジトリで定義するインフラ · 1,022件のマイグレーションを差分検査で保護 · 独自のエラー監視・トレースを構築 · Go・SQL・ブラウザーのテスト群を構築 · 基幹インフラ全体を一から構築 · Azureへのプラットフォーム導入を一貫して担当 · データベースを起点にAPI契約を生成 · AWSの外へ移し、インフラを整理 · PythonとPL/pgSQLでデータ処理を自動化 · GitHub・GitLab CI/CDで4,000時間を削減 · 業務工程の刷新で8,000時間を削減 · Google Cloud・AWSでDockerアプリ20個を運用 · Barmanで重要データのバックアップ100件を自動化 · GISデータパイプライン20本の提供を自動化 · GitHub ActionsでGIS SaaSのデプロイを自動化

{ クラウド・インフラ・DevOps }

コンテナ化・オーケストレーション

どこでも同じように動くソフトウェア

ノートパソコン、テストサーバー、本番環境で同じ動作になるようにまとめ、忙しい週には処理能力を増やせるようにします。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

PodmanとQuadletでコンテナを運用 · 全ユニットのsystemdサンドボックス露出を低減 · GitHub Actions CI/CDパイプラインを構築 · Dockerアプリ100個のSSL/TLS証明書を自動化 · Google Cloud・AWSでDockerアプリ20個を運用

{ クラウド・インフラ・DevOps }

サイト信頼性・監視

お客様より先に異変に気づく

障害が小さいうちに知らせる監視とアラートを整え、安定した状態をどれだけ維持できているかを数値で示します。

6収集処理が省略した入力を明示

6つのレポート収集処理が、未読ファイル、未導入の検査、認識できなかったjournal行など、読み取れなかった入力範囲を明示するよう変更。収集不能と正常なホストが同じページになると確認し、116件の実測サンプルから閾値を定めた。

読み取り専用のホストレポートロールを7個構築

事実、アクセス、git、指標、トラフィック、セキュリティ、プロバイダー資産を、稼働ホストからMarkdownにまとめる7個の読み取り専用レポートロールを構築し、実行で測定していない数値は出力しない規則を設けた。

実行ごとに侵入元の遮断を実証

堅牢化の実行ごとに、予約済みテストアドレスを遮断し、生成されたカーネルルールを読み、必ず実行する後処理で遮断を解除する一連の検証を設け、jailの故障が正常報告ではなく実行失敗になるようにした。

3つあったが、どれも機能していなかった防御

一度も機能していなかったSSH総当たり攻撃対策を3件発見・修正。デーモンが1986番で待ち受けるのに22番を監視する遮断jail、上位の広いルールに隠されたレート制限、バイナリを解決できず遮断を一度も実行できなかったアクションを直した。

IPTV配信信号の連続稼働を維持

配信ネットワークとIP電話を昼夜問わず監視・保守し、供給元と顧客の間でIPTV信号の途切れない配信を維持した。

約1,000台のIPTV/OTTサーバーを24/7で支援

中国、米国、ドイツの世界各地の顧客に向け、IPTV/OTT配信プラットフォームのインフラを24/7体制で支援し、約1,000台のサーバーと顧客所有のシステムを管理した。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

追跡せずに13アドレスを測定 · macOSアプリのイベントパイプラインに上限を設定 · すべての運用秘密情報をホストごとに分離 · Ansibleチェックモードを実態に即した検証に · デッドマンズスイッチによるホスト監視を構築 · 復元方法を明示したバックアップ自動化 · 異常時拒否のレート制限・不正利用対策を構築 · 独自のエラー監視・トレースを構築 · 遅い処理をRiverジョブキューへ移行 · 数年後も稼働するインフラを構築 · Azureへのプラットフォーム導入を一貫して担当 · 切り替えと配信ログを備えるメール機能を構築 ·

地図製品を最初から運用まで支える · Apache・NginxでWebサイト40件を管理 · GrafanaとPrometheusで運用リスクを低減 · UbuntuでPostgreSQL・MS SQLサーバー10台を運用 · 社内初のオブザーバビリティダッシュボードを構築

{ クラウド・インフラ・DevOps }

バックアップ・災害復旧

大きな障害のあとにすべてを取り戻す

あると信じるだけでなく、復元を試したバックアップと、
サーバーが起動しない朝に使える手順書を用意します。

Ubuntu Linux VPSインスタンス30台を管理

Ubuntu Linux VPSインスタンス30台を管理し、災害復旧方針の実装とネットワーク構成の最適化を行った。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

すべての運用秘密情報をホストごとに分離 · 復元方法を明示したバックアップ自動化 · バックアップと災害復旧をコードで実装 · Barmanで重要データのバックアップ100件を自動化

{ クラウド・インフラ・DevOps }

システム管理

日々の稼働を支える

サーバー、メール、アカウント、更新を日々管理
し、誰もが頼る仕組みを平穏に動かし続けます。

ファイアウォールルールを位置で検証

存在するルールが実際のパケットに到達されるとは限らないため、番号付きルール一覧と稼働中のパケットフィルターチェーンを読み、ファイアウォールルールを存在だけでなく位置で検証した。

全社共通のパスワード管理を導入

全社共通のパスワード管理システムを導入し、セキュリティを強化してアクセス管理を効率化した。

ベアボーンサーバー100台を管理

ベアボーンサーバー100台、物理ネットワーク、IP電話システムを管理し、会社の成長を支えるインフラを維持した。

Wi-Fiルーター1,000台を設定・導入

1,000台のWi-Fiルーターを設定・導入し、顧客のネットワーク接続性と性能を改善した。

ネットワークインフラを管理

1,000台を超えるサーバーのネットワークインフラを管理し、適切なシステム構築、セキュリティ、障害対応を行った。

WireGuard・IPsec VPN接続8本を管理

WireGuard VPN・IPsec VPNの8接続を管理して問題に対処し、Google CloudとLinuxシステム間の安全な通信を確保した。

M365・SSO全体のアクセス管理を主導

M365、1Password、Red Hat SSO、OKTA SSOを用いたアクセス管理を主導し、セキュリティ侵害を防いだ。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

読みやすいTor onionミラーを公開 · 6収集処理が省略した入力を明示 · gitホスティングCIを試し2不具合を発見 · 自社サーバーでコードを管理 · 読み取り専用のホストレポートロールを7個構築 · 全ユニットのsystemdサンドボックス露出を低減 · 20レコードのDNSゾーンをコードで照合 · デッドマンズスイッチによるホスト監視を構築 · 復元方法を明示したバックアップ自動化 · 実行ごとに侵入元の遮断を実証 · SSHの24設定項目を検証・堅牢化 · 小型サーバー上の公開サービス · リポジトリで定義するインフラ · 基幹インフラ全体を一から構築 · 数年後も稼働するインフラを構築 · IPTV配信信号の連続稼働を維持 · 約1,000台のIPTV/OTTサーバーを24/7で支援 · Apache・NginxでWebサイト40件を管理 · Ubuntu Linux VPSインスタンス30台を管理 · Linuxサーバーを守る1,000件のRBACルールを作成 · UbuntuでPostgreSQL・MS SQLサーバー10台を運用

{ クラウド・インフラ・DevOps }

ネットワーク・VPN構築

拠点と働く人をつなぐ

Wi-Fi、ファイアウォール、リモートアクセスを整え、どこからでも働け、権限のある人だけが接続できるようにします。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

2台目のサーバーをコードで構築 · 20レコードのDNSゾーンをコードで照合 ·
ファイアウォールルールを位置で検証 · 3つあったが、どれも機能していなかった防御 ·
IPTV配信信号の連続稼働を維持 · Azureインフラをコードとして構築 ·
ベアボーンサーバー100台を管理 · Wi-Fiルーター1,000台を設定・導入 · ネットワークインフラを管理
· WireGuard・IPsec VPN接続8本を管理 · Ubuntu Linux VPSインスタンス30台を管理

{ クラウド・インフラ・DevOps }

セキュリティ・アクセス管理

誰が何にアクセスできるかを決める

アカウント、パスワード、権限を整理し、業務に必要なアクセスだけを付与し、退職者の権限は当日に解除します。

チームの意思疎通と協働を改善

Slack、Mattermost、1Password、Jiraを導入してチームの意思疎通と協働を改善し、8,000時間の作業を削減した。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

読みやすいTor onionミラーを公開 · 追跡せずに13アドレスを測定 ·
公開するすべての部品のライセンスを把握 · 未使用だったコアの半分の機能を実装 ·
すべての運用秘密情報をホストごとに分離 · 認証情報漏出後に対象範囲の規則を記述 ·
2台目のサーバーをコードで構築 · PodmanとQuadletでコンテナを運用 ·
全ユニットのsystemdサンドボックス露出を低減 · 20レコードのDNSゾーンをコードで照合 ·
秘密情報の1秒の事前検査を追加 · ファイアウォールルールを位置で検証 ·
実行ごとに侵入元の遮断を実証 · SSHの24設定項目を検証・堅牢化 ·
3つあったが、どれも機能していなかった防御 · 継続的なセキュリティ検査を運用 ·
企業所有権の申請・審査を構築 · 異常時拒否のレート制限・不正利用対策を構築 ·
Stripe・アプリ内購入の利用権限を構築 · 全社共通のパスワード管理を導入 · チームの協働を自動化 ·
nonce付きCSPでアプリケーションを堅牢化 · Azureインフラをコードとして構築 ·
切り替えと配信ログを備えるメール機能を構築 · 利用者の組織コンテキスト切り替えを構築 ·
Dockerアプリ100個のSSL/TLS証明書を自動化 · ネットワークインフラを管理 ·
WireGuard・IPsec VPN接続8本を管理 · M365・SSO全体のアクセス管理を主導 ·
Linuxサーバーを守る1,000件のRBACルールを作成

{ クラウド・インフラ・DevOps }

AIエージェントのセキュリティ

AIが扱ってよいものだけを扱わせる

各エージェントのアクセスを必要な範囲に限定し、エージェント自身が改変できない場所に操作を記録し、安全策を信じるだけでなく検証します。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

すべての運用秘密情報をホストごとに分離 · 認証情報漏出後に対象範囲の規則を記述 · 秘密情報の1秒の事前検査を追加 · 実行ごとに侵入元の遮断を実証 · 3つあったが、どれも機能していなかった防御 · 継続的なセキュリティ検査を運用 · 異常時拒否のレート制限・不正利用対策を構築 · M365・SSO全体のアクセス管理を主導 · Linuxサーバーを守る1,000件のRBACルールを作成

{ コンサルティングとリーダーシップ }

技術面の統率・コンサルティング

経験豊富なエンジニアが判断を支える

取引先、予算、設計について、知識に基づく別の見解が必要なときに、同じ側に立って判断を支えます。

文書を独自の静的検査で管理

文書に、引き下げのみ可能なファイル別行数上限、目次要件、実在パス検査を設けた。完了扱いの9計画中6計画に未完了項目があり、コミットを制御する14タスクがどの文書にも記載されていないことを検出した。

社内プロジェクトの開発を支援

多数の社内プロジェクトの実行を導き、ソフトウェア開発の各段階で専門的な支援を行った。

従業員4人から14人への成長を主導

アジャイル手法と効果的なプロジェクト管理を用い、従業員4人から14人への会社の成長を主導した。

50,000語のソフトウェア文書を執筆

GitHub、Craft、ConfluenceでMarkdownを使い、50,000語の詳しいソフトウェア文書を執筆。知識の蓄積と工程の透明性を確保した。

推定4,000時間を削減、チームは2人から18人へ

2人から18人へ成長したチームを指導し、作業手順と部門間の連携を改善する中で、4,000時間の削減を見積もった。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

4リポジトリで共通の履歴基準を維持 · テスト群を8.9秒に短縮 · C上にSwift 6の厳密並行処理を採用 · フックと品質ゲートの不一致を発見 · 検査自体のテストを作成 · すべての静的検査ルールをエラーに設定 · 生成器に981テストと92%カバレッジを適用 · 10,242行の品質ゲート自体に静的検査を適用 · 51件の文書化した施策として開発を運営 · 30分も待たずに済む品質検査 · 268検査を持つリポジトリの検査基盤を構築 · Go・SQL・ブラウザーのテスト群を構築 · 分析と定期的な進捗報告を提供 · プラットフォームの基礎技術方針を策定 · 6言語に警告ゼロの基準を設定 · CI/CDの効率化で4,000時間を削減 · 業務工程の刷新で8,000時間を削減 · 30件以上のGIS開発を構築からサポートまで主導 · フルスタックのGIS地図開発を指揮 · GIS地図アプリの開発を主導し売上を10倍に · 14部門を支える共通インフラ計画

{ コンサルティングとリーダーシップ }

チームづくり・人材育成

今いるチームを育てる

採用、受け入れ、指導を通して、知識をコンサルタントとともに持ち去らせず、自社の担当者に根づかせます。

ITサポートチームを管理

1,000を超える顧客にITサポート、データ復旧、機器修理を提供するチームを管理し、サービス品質を確保した。

30プロジェクト全体で月150時間を削減

作業手順と資源配分を最適化し、30件のプロジェクト全体で月150時間を削減した。

技術サポート部門を設立

技術サポート部門を立ち上げ、10,000を超える顧客にITサポートと問題解決を提供し、CRMで問い合わせを管理した。

5,000時間の地図開発を管理

ScrumやJiraを用い、5,000時間の地図開発を計画・管理した。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

社内プロジェクトの開発を支援 · チームの意思疎通と協働を改善 ·
従業員4人から14人への成長を主導 · GISプロジェクト30件を提供まで管理 ·
推定4,000時間を削減、チームは2人から18人へ

{ コンサルティングとリーダーシップ }

プロジェクト管理（アジャイル）

完成まで責任を持って進める

範囲、日程、難しい調整に責任を持つ担当者を置き、5つの意見が並ぶだけでなく、1人の責任者がいる状態にします。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

4リポジトリで共通の履歴基準を維持 · 文書を独自の静的検査で管理 ·
51件の文書化した施策として開発を運営 · チームの協働を自動化 ·
時間管理・レポートシステムを設計 · 分析と定期的な進捗報告を提供 ·
30プロジェクト全体で月150時間を削減 · 技術サポート部門を設立 ·
業務工程の刷新で8,000時間を削減 · 社内プロジェクトの開発を支援 · チームの意思疎通と協働を改善 · 従業員4人から14人への成長を主導 · GISプロジェクト30件を提供まで管理 · 業務要件を収集・分析 ·
5,000時間の地図開発を管理 · 推定4,000時間を削減、チームは2人から18人へ

{ コンサルティングとリーダーシップ }

プロダクト戦略・要件定義

何をつくるべきかを見極める

事業側の要望を開発可能な内容に変え、つくる価値がないものもはっきり伝えます。

複数チャネルで一貫したアイデンティティ

Engineer ApSのオンライン・オフライン素材に共通するブランドアイデンティティを構築し、見せ方を技術実績集へ結びつけた。創業者の報告では、この取り組みによりブランド認知度が200倍になった。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

利用者を責めないエラーメッセージに改訂 · 51件の文書化した施策として開発を運営 · 22静的検査のコミットゲートを構築 · 採用と船員キャリアの管理画面を構築 · 投稿・フィード・グループの交流機能を構築 · ロイヤルティ残高、リーグ、バッジを構築 · 企業所有権の申請・審査を構築 · プラットフォームの運用管理画面を構築 · Stripe・アプリ内購入の利用権限を構築 · 時間管理・レポートシステムを設計 · 製品のUIとUX全体を設計 · データ分析レポートシステムを開発 · 業務要件を収集・分析 · 5,000時間の地図開発を管理 · UI/UX改善200件を提案 · 予測と投資戦略を最適化 · GIS地図アプリの開発を主導し売上を10倍に · 読み手の反応に基づくブランド発信

{ コンサルティングとリーダーシップ }

技術文書の作成

次の担当者が使える形で書き残す

構築した人に電話しなくても自社の担当者が運用できるよう、引き継ぎ資料、運用手順書、マニュアルを整えます。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

6収集処理が省略した入力を明示 · 4リポジトリで共通の履歴基準を維持 · 利用者を責めないエラーメッセージに改訂 · 全FFI呼び出しを検証する構文解析器を作成 · 言語間の数値照合を構築 · フックと品質ゲートの不一致を発見 · PDF/UA-1プロファイルの106項目中106項目に合格 · 文書を独自の静的検査で管理 · 51件の文書化した施策として開発を運営 · 22静的検査のコミットゲートを構築 · 認証情報漏出後に対象範囲の規則を記述 · 読み取り専用のホストレポートロールを7個構築 · 継続的なセキュリティ検査を運用 · 268検査を持つリポジトリの検査基盤を構築 · go-taskの自動化ターゲット578個を作成 · APIを100%の一致でHuma v2へ移行 · 50,000語のソフトウェア文書を執筆 · マニュアル付きPower BIプロジェクト8件を提供 · 14部門を支える共通インフラ計画

{ ITサポートとハードウェア }

ITサポート・ヘルプデスク

日々のパソコンの困りごとを解決する

ノートパソコン、プリンター、ログイン、メールの問題を、
利用者が気後れしない平易な説明で、速やかに解決します。

ノートパソコンのハードウェアを診断・修理

画面、ヒンジ、キーボード、トラックパッド、マザーボード、電源など、ノートパソコンのハードウェアを診断・修理し、Linux、Windows、Macのソフトウェア問題も解決した。

SDカード・HDD・SSD・RAIDからデータを復旧

SDカード、HDD、SSD、RAIDアレイ、外付けドライブ、Macシステムなど、幅広い媒体からデータを復旧した。

コンピューター修理4,000件を管理

4,000件のコンピューター修理を管理し、ハードウェア・ソフトウェアの問題を迅速かつ適切に解決した。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

約1,000台のIPTV/OTTサーバーを24/7で支援 · Wi-Fiルーター1,000台を設定・導入 · ITサポートチームを管理 · 技術サポート部門を設立

{ ITサポートとハードウェア }

機器修理・データ復旧

動かなくなった機器からファイルを救い出す

修理する価値のある機器を直し、動作しなくなった
ディスクから写真、請求書、図面を取り出します。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

ノートパソコンのハードウェアを診断・修理 · SDカード・HDD・SSD・RAIDからデータを復旧 · コンピューター修理4,000件を管理 · ITサポートチームを管理

{ ブランドとマーケティング }

ブランド・マーケティング・SEO

見つけてもらい、ふさわしい印象を伝える

商品やサービスを探す人に見つかり、どこで接しても同じ
印象が伝わる名前、見た目、Webサイトをつくります。

このサービスは、このページの別の箇所に掲載した仕事にも含まれています：

2点の図からビジュアルアイデンティティを構築 · 48トークンのmacOSデザインシステムを構築 · 更新日時が共通のサイトマップを修正 · CSSバンドルを87 KBから31 KBに削減 · 963メタデータブロックを1グラフに統合 · 配色を28色に限定 · SQLiteエクスポートから495ページを生成 · 27種類の文字サイズを6段階の体系に統一 · Web開発にSEOとコンテンツを統合 · 読み手の反応に基づくブランド発信 · 複数チャネルで一貫したアイデンティティ

ご自身の出発点と比べる

比較できる実績を選び、ご自身の状況との違いを明らかにしてください。

料金を見る お問い合わせ →

状況に合う進め方を相談する

関連する経験を、ご評価いただける進め方に具体化するようご依頼ください。

サービスを見る

welcome@engineer.company +45 50 10 61 84 デンマーク、コペンハーゲン

このページを持ち歩く

連絡先カード

心を込めて丁寧に制作： D.

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

著作権 © 2025 – 現在 · Engineer ApS · すべての権利を留保します。

印刷元： https://engineer.company/ja/portfolio/