

職務経歴書

基本情報

項目名	内容
名前	木村 拓光
生年月	1992 年 12 月
居住地	神奈川県
最終学歴	山形県立山形工業高等学校 機械システム科
職歴概要	フロントエンドエンジニア 3 年
資格	普通自動車第一種運転免許 IT パスポート

技術記事・GitHub

- ・ Zenn
- ・ 個人ブログ
- ・ GitHub

スキル

開発手法

- ・ アジャイル開発
- ・ スクラム開発
- ・ MVP 開発

言語

名称	経験年数

TypeScript	3 年
Node.js	1 年

フレームワーク・ライブラリ

名称	経験年数
React	3 年
Next.js (Pages Router)	2 年
Next.js (App Router)	1 年
Nuxt.js(3 系)	1 年
Nuxt.js(2 系)	1 年
Vue.js (2 系)	1 年
Astro	1 年
Express	1 年
React Hook Form (v7)	1 年
useSWR (v2)	1 年
openapi-typescript (v6)	1 年
Zod	1 年

UI ライブラリ・CSS フレームワーク

名称	経験年数
AntDesign (v4)	1 年
MUI (v5)	1 年
Headless UI	1 年
Vuetify(v3)	1 年
Tailwind CSS (v3)	1 年

バンドルツール

名称	経験年数
webpack	2 年
Vite	1 年

パッケージマネージャ

名称	経験年数
npm	3 年
Yarn	1 年

SaaS/PaaS

名称	経験年数
GitHub	3 年
Firebase	1 年
Firestore	1 年
Supabase	1 年
Vercel	1 年
GitHub Actions	1 年

テストツール等

名称	経験年数
Jest	1 年
React Testing Library	1 年
Storybook	1 年

デザインツール

名称	経験年数
Figma	1 年
AdobeXD	1 年
AdobePhotoShop	1 年

コミュニケーションツール

名称	経験年数
Slack	3 年
Chatwork	1 年
Microsoft Teams	1 年

タスク管理ツール

名称	経験年数
Notion	2 年
Asana	1 年
Backlog	1 年

エディタ

名称	経験年数
Visual Studio Code	3 年

その他

名称	経験年数
Git	3 年
OpenAPI	1 年
GAS	1 年

Docker	1 年
Google Tag Manager	1 年
Google Analytics	1 年
ESLint	3 年
Prettier	3 年

職務経歴詳細

旅行事業をメインに運営する企業

【雇用形態】

準委任契約

【在籍期間】

2023/02/01 ~ 現在

【チーム編成】

- ・ プロダクトマネージャ 1 人
- ・ デザイナー 1 人
- ・ マーケットチーム 3 人
- ・ フロントエンド 5 人
- ・ バックエンド 5 人
- ・ インフラ 2 人
- ・ QA 3 人

【チームでの役割】

2023/02/01 ~ 2023/08/31 フロントエンドチームメンバー 2023/09/01 ~ 2024/09/30 フロントエンドチームリーダー

【チーム課題と自分が工夫したこと】

マネジメント不足による作業指示の伝達不備と工夫したこと

マネジメントが行き届いておらず、作業指示が作業者にうまく伝わっていないと感じていた。そこで要件整理から始めて、時には実際にやりたいことは何かをヒアリングして要件の明確化をした。仕様や要件を整理して作業者に伝えることで認識の齟齬をなくし、作業者に必要以上の負担をかけないことを心がけた。

GoogleTagManager を使って画面書き換えを行なっており、その管理ができていなかった

外部から画面を書き換えていたため、開発側としては意図しない画面になつていると勘違いする、書き換えたものが正と認識されていて改修しようと思ったら想定外のコミュニケーションが発生するなどの問題が発生していた。外部から書き換えることで発生する可能性がある不具合の共有やドキュメント以上に管理・共有を徹底することの重要性をチーム内に広めた。

デザインスタイルガイドラインが存在しない

次のような事象が発生しており、細かいところで統一感がないサイトになっていた。

- ・ボタンのサイズやフォントサイズがバラバラ
- ・カラーコードが微妙に違う
- ・意図しないようなデザインのズレが発生している
- ・ブレイクポイントの設定が曖昧
- ・要素幅に無理やりデザインを合わせていて文字数などの条件が変わるとすぐに崩れる

そこでスタイルガイドライン作成を提案した。 レビューを通して実装目標・ユーザ目標から実装しやすいデザインやわかりやすいデザインをデザイナーに共有し、アップデートできた。

旅行予約サイトリプレイス開発

【言語・フレームワーク】

- ・TypeScript
- ・Next.js (App Router)
- ・NestJS

【インフラ】

- ・AWS
 - ・S3
 - ・CloudWatch Logs
 - ・Cognito

【その他】

- ・GitHub Actions
- ・Salesforce

【プロダクト概要】

運用中の旅行予約サイトリプレイスプロジェクト。 SEO を考慮して Next.js の App Router を採用した。 次に挙げられる既存サイトの課題を解決することを目標とした。

- ・コードの可読性や保守性の高さ

- ・ドキュメント・仕様書の整備
- ・テスト作成
- ・パフォーマンス向上
- ・複雑性の排除

【苦労したこと】

採用技術のキャッチアップ

Next.js (App Router)の採用が決まっていた。App Router が Stable になってまだ間もないくらいの時期だったため、キャッチアップが必須だった。特に server components と client components の違い、キャッシング制御あたりは App Router のコアな部分かつ難しい部分だった。ドキュメントや海外の情報を参考にしつつ動作確認と検証、情報共有を重ねて知見を増やしていった。この知見が残るようドキュメント化した。

またテスト関係の知見もなかったためフロントエンドを取り巻くテストの情報をまとめてチーム内で発表を行いテストの重要性や手法を広めることができた。

【うまくいったこと】

リプレイスプロジェクトの目的の明確化

リプレイスプロジェクトの立案には直接関与ていなかったものの、その目的と現状の課題を明確化することで、チーム全体に次の重要な認識を共有できた。

- ・単体テストの重要性
- ・コードレビューの重要性・観点
- ・バックエンドとの情報共有の必要性
- ・OpenAPI 導入
- ・Linter・Formatter による静的テスト自動化

openapi-typescript を導入

プロジェクト当初から OpenAPI を作成することは決まっていたものの、OpenAPI を利用して型定義を作成するなどのことは考えられていなかった。OpenAPI から型を生成することによるメリットを調査して必要性をチームに展開した。openapi-typescript というライブラリを導入することで次のようなメリットを得られた。

- ・フロントとバック間の型を共有できる
- ・API リクエストを型安全に行えるようになった
 - ・パスやクエリの指定時にコード補完が効くようになり、SwaggerUI を見ずとも API リクエストがかけるようになった
- ・API レスポンスの型が自動で補完されるようになり、型安全にレスポンスを使用できるようになった

【言語・フレームワーク】

- Java
- Spring Boot
- Thymeleaf
- TypeScript
- React
- jQuery
- Express

【インフラ】

- AWS
 - S3
 - CloudWatch Logs

【その他】

- GitHub Actions
- AWS CodePipeline
- Salesforce
- Datadog

【プロダクト概要】

会員登録者数 800 万人以上の旅行予約サイトの保守・運用・開発 運用期間が長いことと、チームメンバーの入れ替わりによりコード規約が守れていなかったり技術負債が溜まっている状態だった。

【苦労したこと】

既存サイトの問題点とリプレイスプロジェクト

長年の運用により技術的負債が蓄積した既存サイトのリプレイスを計画していた。 具体的には、TypeScript の型定義の不備、テストの欠如、コードの構造化不足、CSS の管理問題、そしてシステム全体の理解不足といった課題を抱えていた。 これらの問題は、保守性や拡張性を著しく低下させ、新たな機能追加や改修を困難にしていた。

このような状況で次のような工夫をした。

- まず、コードレビューと動作検証を通じて既存サイトの現状を詳細に把握し仕組みを理解し情報を共有した
- 大規模なリファクタリング(型定義の厳格化)などは現実的ではなかったため、定数の切り出しや新規実装時は可読性の高いコードにするなど小さくできることを当時のリーダーと決めた

介護サービス運営企業

【雇用形態】

請負契約

【在籍期間】

2022/09/01 ~ 2023/01/31

【チーム編成】

- ・ プロダクトマネージャ 1 人
- ・ フロントエンド 2 人
- ・ バックエンド 1 人

【チームでの役割】

フロントエンド

事務効率化ツール開発

【言語・フレームワーク】

- ・ TypeScript
- ・ Nuxt.js(3 系)

【インフラ】

- ・ Supabase

【その他】

- ・ Vuetify
- ・ Pinia

【プロダクト概要】

介護サービスを扱う会社が補助金の申請などを行うために PDF ファイルのやり取りに関する事務作業が発生していた。この作業を効率化するために、PDF ファイルをアップロードするとファイル名から自動的に分類してくれる仕組み作りをトライした。

【苦労したこと】

仕様が決まりきっていない部分があり、コミュニケーションを重ねて仕様を決めながら実装していく。た。

Vue (3 系) になったことでライブラリがうまく対応しておらず、実現したいことがなかなかうまくいかなかった。

フレームワークとして Nuxt が選定されたが、Nuxt を扱える人がおらず学びながらの実装になった。難しさは感じなかった。React と比べて周辺ライブラリは少ないので、習熟度が上がればその分ライブラリ選定に悩むことなくスピーディーに対応できるのかもしれない。

【うまくいったこと】

MVP 開発ということで、できること・できないことを可視化して最低限の機能実装に注力できた。

インターネットメディア事業運営企業

【雇用形態】

準委任契約

【在籍期間】

2022/08/01 ~ 2023/01/31

【チーム編成】

- ・ プレイヤーマネージャ 1 人
- ・ フロントエンド 2 人
- ・ デザイナー 5 人

【チームでの役割】

フロントエンド

【チーム課題と自信が工夫したこと】

デザイナーと連携を密にとり、お互いの認識齟齬が発生しないことを心がけた。また、エンジニアの視点も共有することでデザインがより実装しやすいものになることや UI・UX の向上を目指した。

医療事業予約サイト開発

【言語・フレームワーク】

- ・ React
- ・ Next.js (v12)
- ・ AntDesign
- ・ MUI
- ・ React Hook Form

【インフラ】

- Firestore
- Firebase Hosting
- Firebase Authentication
- Cloud Functions

【その他】

- GitHub
- Git

【プロダクト概要】

脱毛予約サイトの管理画面と予約 LP サイト開発。 主に管理画面側の作成を担当した。 主な機能は次のとおり。

- ログイン機能
- 新規予約機能
- 管理画面側からの予約確認や予約状況などのステータス管理機能
- 店舗切り替え機能
- ページネーション
- 予約 LP との同期

【苦労したこと】

UI ライブライアリは開発初期段階では良いが、カスタマイズしようとすると途端に使いづらくなると感じた。プロダクトで使用した MUI, AntDesign は特にそうだった。導入時にそういう観点でも検討することの必要性を感じた。

海外のチームと連携をとって作業することがあり、その連携を私がとる事もあった。より簡潔でわかりやすいテキストや、スクショを共有するなど言葉以外の部分でも伝えることの有効性を感じた。

【うまくいったこと】

開発に携われる人数が少なかったかつ短納期、PM と密に連携し課題や進捗を共有。管理画面の 8 割方は私が担当した領域だったと言える。開発期間 3 ヶ月ほどで無事リリースでき、その後の運用にも問題は発生しなかった。人伝に聞いた話では、プロダクトがスケールして外部販売もできるようになったそう。

toC サイト運用・保守・開発

【言語・フレームワーク】

- HTML
- CSS
- JavaScript

- Nuxt.js(2 系)
- Vue.js(2 系)
- WordPress
- pug

【その他】

- FTP

【プロダクト概要】

主に女性向け下着や健康食品など、様々な toC のサイト運用・保守・開発。独自の EC サイトの他に楽天市場、yahoo ショップなどの EC サイトに出店していた。

【苦労したこと】

楽天市場、yahoo ショップなどの EC サイトには管理画面があるが気軽に更新作業が行える分、他の作業者とやり方が異なるたり、担当者がいなくなったりした場合にどうやっていたかがわからない状態になりやすかった。また管理画面の UI が変わる事もあり、変わった途端にわかりづらくなったりドキュメントを残してもその通りにできなくなってしまう事もあった。

FTP でファイルをアップロードすることが初めてで、繋がってしまえばワンクリックで壊れてしまう可能性があり驚きと恐怖を感じた。

【うまくいったこと】

ドキュメントやアカウント情報の整理をして、探す手間を削減できた。

ロジックがソースコード上に散らばってしまっていたので、リファクタリングを遂行し、コードの品質を向上できた。

SNS アカウントの閲覧数等のデータ収集自動化

【言語・フレームワーク】

- GAS

【その他】

- Instagram Graph API

【プロダクト概要】

Instagram で特定のアカウントの投稿や閲覧数・リール数などのデータを取得。Google Spread Sheet へ定期的に集計して統計データを作成。

【苦労したこと】

Instagram Graph API でできることの調査。セキュリティ的に取れるデータと取れないデータがあり、本当に取れないデータなのか、取れないのであれば取れない理由を説明することが大変だった。

【うまくいったこと】

投稿を時間毎に監視するための仕組み作り。データが毎回正常に取得できるわけではなく、アクセス数などの影響で取れない事もある。なので対象時間に幅を持たせてデータが取れていない場合に、データ取得トライ回数を増やしてなるべくデータを取得できるようにした。

ライブイベント運営企業

【雇用形態】

準委任契約

【在籍期間】

2022/06/01 ~ 2022/07/31

【チーム編成】

- ・ プロダクトマネージャ 1 人
- ・ フロントエンド 3 人
- ・ インフラ 1 人

【チームでの役割】

フロントエンド

【チーム課題と自信が工夫したこと】

納品までのスピード感が求められたため、品質が多少落ちがちだった。共通化できるような箇所をリストアップしてテンプレート化。それらを作業共有することでその部分の品質担保と余計な工数を削減できた。

デプロイ作業を違う作業者が担当することもあったので、作業のドキュメント作成し作業がスムーズにいくようになるまで確認とコミュニケーションをとり、作業者が必要以上に作業負担を感じないようにした。

数百万人規模のイベント告知 LP 作成・運用

【言語・フレームワーク】

- ・ HTML
- ・ CSS(Sass)

【インフラ】

- Firebase Hosting

【その他】

- GitHub
- Git
- AdobePhotoShop

【プロダクト概要】

有名アーティストのドームツアーなど、数百万人規模のライブイベントの告知 LP 作成と運用。主に作成を担当し、10 件程の作業に携わった。

【苦労したこと】

Sass のコンパイル方法が決まっていなかったり、フォーマッターのルールが決まっていないなど、差分が発生してほしくないところで差分が発生してしまっていた。次の対応をとった。

- Sass は使わずに CSS を直接編集するようにした
- フォーマッターは使用せずに余計な差分が発生しないようにした

【うまくいったこと】

チームメンバーと進捗を共有し納期遅れが発生することなく作業できた。

チームメンバーへの引き継ぎがスムーズにできた。

次のようなマネージャが把握しきれていないことを密に連携できた。

- リリーススケジュール
- デザインの不具合・疑問点