生成AIと"協働"してみた！

Nuxtアップグレード実践レポート

2025年8月29日

株式会社オープントーン
ITエンジニアリング事業部 伊東直喜

OPEN TONE

自己紹介

趣味：テニス

伊東 直喜

株式会社オープントーン
ITエンジニアリング事業部

10名のチームで、Webシステム開発の
プロジェクトリーダーをつとめる

著書

『実装で学ぶフルスタックWeb開発』を共同執筆

https://www.shoeisha.co.jp/book/detail/9784798179834

はじめに

ゴール

フレームワークアップグレードを題材にした、生成AI活用による開発効率化手法の理解

ターゲット

生成AIによる開発効率化に関心のあるエンジニア

注意点

Nuxtの技術解説は最小限とし、生成AI活用手法に焦点を当てます

目次

Devin（今回使用する生成AI）の概要

エンジニアのように働いてくれる生成AI

何ができるの？

- 事前調査・見積もり
- 設計・実装

https://devin.ai/

Nuxt2 → Nuxt4 アップグレードの全体像

Nuxtとは

Webシステムを簡単に構築できるフレームワーク

https://nuxt.com/

アップグレードの必要性

- Nuxt2は公式サポートが終了済
- セキュリティリスクの増加

生成AIにアップグレードを全部任せてみよう

"Devin、Nuxt2のソースコードをNuxt4にアップグレードしてください"

生成AIにアップグレードを全部任せてみた

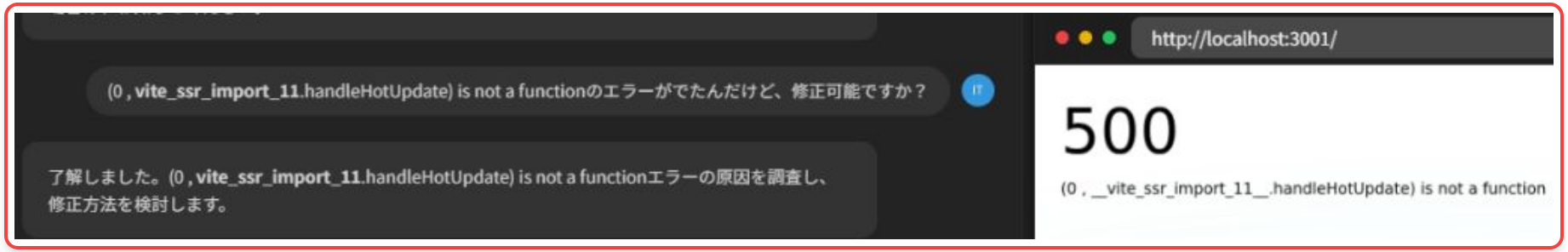
エンジニア: Nuxt2のソースコードをNuxt4にアップグレードしてください

Devin: 承知しました。Nuxt4の移行ガイドに沿って変換作業を進めます。Nuxt2からNuxt4への移行が完了しました。

エンジニア: ○○というエラーがでました。修正お願いします。

Devin: エラーを再現して修正し、ログインエラーを解決しました。

⚠ エラーを再現して修正する様子



全部任せた結果

結果

✓ Devinが画面を動かしながらエラーを解消して、2時間ほどでログイン画面が表示できた

✗ 修正コード量が多く（一度に100ソース以上）、エンジニアがレビューしきれなかった

✗ Devinが進めた対応方針が、エンジニアから見るといまいちだった

失敗から学んだ教訓と今後の方針

何を失敗したのか？

- 生成AIへの一度の依頼量が多すぎた
- エンジニアが判断するべき点（移行の方針決定等）を、生成AIに任せてしまった

今後の方針

- 工程を適切に分離して段階的にアップグレードを進める
- 生成AIには「提案・対応・レビュー支援」の役割を担わせる
- エンジニアは「判断・方針決定・最終確認」の責任を持つ

アップグレードの工程

1. 事前調査・工数見積もり 生成AI活用
2. 設計・実装 生成AI活用
3. テスト
4. リリース

事前調査・工数見積もり：従来手法

具体的な手順

事前調査

公式移行ガイドとソースコードを見て移行ガイド作成

工数見積もり

変更箇所の目視確認と経験に基づく工数設定

従来手法による課題

- 時間と手間がかかる
- 精度のバラつき（担当者のスキル・経験に依存）

従来手法の課題を解決したい。そうだ、生成AIを活用してみよう

事前調査・工数見積もり：生成AIを活用

エンジニア

Nuxt2で作成したソースコードをNuxt4に移行したいです。

：（略）

事前調査と、工数見積もりをしてください。

Devin

Nuxt2→Nuxt4移行ガイド（MIGRATION_GUIDE.md）を作成しました。

移行ガイドにソースコードの修正箇所と実装例を記載しました。

総工数見積もり: 50-70日 ※以下、内訳

・Vuetify1→3への移行：15人日

・JavaScript→TypeScriptへの移行：5人日

：（略）

事前調査・工数見積もり：成果と課題

事前調査

- ソースコードの修正箇所を一覧化
- Nuxt2とNuxt4の実装例を表示

工数見積もり

- 総工数を瞬時に算出
- 内訳を提示

課題

- 対応工数の不確実性
- 対応方針の比較検討が不足

エンジニアレビューで調査・見積もりの精度を向上させよう

事前調査・工数見積もり：エンジニアのレビューで精度を向上

実施したこと

- サンプル実装（10%）で実績工数を測定し見積もり精度を向上
- 生成AIと相談して対応方針に調整（例：TypeScriptへの移行を中止）

アップグレード作業項目	Devin自動見積(人日)	エンジニア見積(生成AIと相談後)(人日)	差異(エンジニア見積 - Devin見積)
Nuxt2→4の基本設定	3	4	+1
Vuetify1→3への移行	15	25	+10
JavaScript→TypeScriptへの移行	8	0	-8

事前調査・工数見積もり：学び

学び

早く大まかな見積もり

生成AIを使うと、事前調査や工数の大まかな見積もりを短時間で作れた

より正確に

生成AIが作った結果をエンジニアがチェックすることで、より正確な結果が得られた

設計・実装：従来手法によるアップグレード

従来手法によるアップグレード

- パッケージの差し替え
- エラー箇所の修正
- 非推奨機能への対応

従来手法だと時間と労力がかかる。そうだ、生成AIを活用してみよう →

設計・実装：【進め方1】生成AIと壁打ち

どんな時に生成AIと壁打ちするか

- 課題を解決したいとき
- 複数のアプローチを比較検討したい時
- 大規模な変更や移行の戦略を立てる時

エンジニア

axios（バックエンドとの通信ライブラリ）はNuxt4で使えないので、fetchに移行する必要があります。axiosを使っている箇所が膨大なので大変です。解決策を提案して。

Devin

axiosと同様のAPIを提供しながら、内部ではfetchを使用する抽象化レイヤーを作成します。この実装により、既存のコードはほぼそのまま動作します。コードを示します。

設計・実装：【進め方2】生成AIに対応依頼

どんな時に生成AIに依頼するか

- 明確な作業指示がある場合
- 定型的なコード変換が必要な場合

エンジニア

工数を抑える為に、Options APIのままでNuxt4に移行したい。https://nuxt.com/docs/api/utils/define-nuxt-component を参考にしてdefineNuxtComponentを使うよう対応して。

Devin

承知しました。defineNuxtComponentを使うよう対応します。

設計・実装：まとめ

壁打ちにより、実装に迷うことが少なくなった

実装の大部分を生成AIで実施し、実装工数が少なくなった

生成AIの結果をレビューすることで、バグ発生率が下がった

まとめ

生成AIと"協働"してみた！

生成AIの活用効果

生産性→**約25%向上**

不具合→**約25%減少**

応用の可能性

様々な言語・全工程で活用可能

生成AIとの協働

~~「まかせる」~~ → **「一緒に進める」**

生成AIは**開発パートナー**として協働することで最大の効果を発揮

質疑応答 / Thank You

ご質問をぜひお聞かせください！

生成AIとの協働についての疑問点や、ご自身のプロジェクトへの適用方法など、どんな内容でもお気軽にご質問ください。皆様の経験や課題についてもぜひシェアしていただけると嬉しいです。

ご清聴ありがとうございました

伊東 直喜
株式会社オープントーン
ITエンジニアリング事業部