和歌山大学協働教育センター クリエプロジェクト
＜2021 年度ミッション成果報告書＞

プロジェクト名：programming project dlopp

ミッション名：Web 領域のモダンな技術を活用して、アプリケーションを企画・開発・運用する。

ミッションメンバー：
システム工学部 2 年前川大樹，システム工学部 2 年坂根美優，経済学部 2 年植木結土，システム工学部 1 年小川剛

キーワード：Web 開発，テスト，CI/CD, React, Next.js, Heroku, Vue.js, Nuxt.js, Firebase, Git, GitHub, AWS, Azure

1. 背景と目的

Web 技術はトレンドの変化が激しい。１年経つと技術トレンドが変わったり、バージョンアップの必要が出てきたりする。本ミッションでは、ミッション名を「Web 領域のモダンな技術を活用して、アプリケーションを企画・開発・運用する。」と定めている。技術トレンドを追い、モダンな環境で開発することで、特定の環境に依存しない汎用的な能力を身につけることを目的としている。具体的には、本プロジェクトで１年前に作成したアプリケーションの古くなったシステムを置き換える作業を行なった。本ミッションで作成したアプリは、「本プロジェクトに対して自由に質問を投稿できるサービス(以下、本アプリケーションという)」である。管理者権限を持ったユーザーは、投稿された質問に回答することができ、質問内容とそれに対する回答をアプリ上や Twitter 等の SNS で共有することができる。

図 １本アプリケーションのトップ画面

2. 活動内容

本ミッションでは、本アプリケーションを継続開発するにあたって発生した技術的な負債を解消する活動を行なった。具体的には以下のような負債を解消した。

2-1 バージョンアップすべきフレームワークの放置

2-2 開発メンバーの習熟度に起因する負債

2-3 PaaS のプランが開発当初と変わっていたこと

2-1 バージョンアップすべきフレームワークの放置

本アプリケーションは、フロントエンドフレームワークに Nuxt.js を用いている。Nuxt.js は開発当初 v2.x 系であったが、v3.x 系が利用可能になっていた。(以下、それぞれ Nuxt2、Nuxt3 という。)開発当初、ビルド時間が遅いという問題が発生しており、webpack プラグインの hard-source-webpack-plugin でモジュールをキャッシュ管理することによりビルドを高速化する方法を取っていた。この方法でビルド時間を大幅に短縮することができていたが、よくビルドが失敗するなど不安定であったことから、デプロイする段階で利用しないことにしていた。Nuxt3 では Nitro Engine という新しいサーバーエンジンを使用しており、高速でビルドが走らせる事ができる。その他、後述する TypeScript、Composition API がサポートされていることも移行するメリットであると考え、Nuxt2 から Nuxt3 にアップデートする作業を行なった。それぞれの間に互換性はほとんどないため、基本的に大半のコードを書き換える必要がある。本ミッションでは、公式で提供されている Nuxt Bridge という Nuxt 2 のプロジェクトで Nuxt 3 の機能を体験できる上位互換性レイヤーを利用し、Nuxt 2 と Nuxt 3 のコードを共存させながら少しずつ移行することで、大幅な書き直しやリスクのあ

る変更をせずに Nuxt3 の機能を利用することができた。

2-2 開発メンバーの習熟度に起因する負債

本アプリケーションは、フロントエンドの開発言語に JavaScript を使用している。JavaScript に静的型付けとクラスベースオブジェクト指向を加えたスーパーセットに TypeScript があるが、実装当初の開発メンバーの習熟度と、アプリケーションの規模を拡大させていく予定がなかったこと、型をつけることがコストになると考えていたことから JavaScript を採用していた。しかし、今後アプリケーションの規模を拡大させていく上で、型がないことに起因するバグの調査や修正が増えていくことが想定されるため、規模が小さい今のうちに TypeScript を活用しようと決めた。また本ミッションを実施する段階で、開発メンバーに TypeScript が浸透していたことも要因の一つである。本ミッションでは、TypeScript の設定ファイルに JavaScript を許可する一文を追加し、JavaScript と TypeScript を共存させることで、少しずつ移行作業を進めた。Nuxt 3 では、TypeScript がサポートされているので、移行作業も行いやすくなっていた。

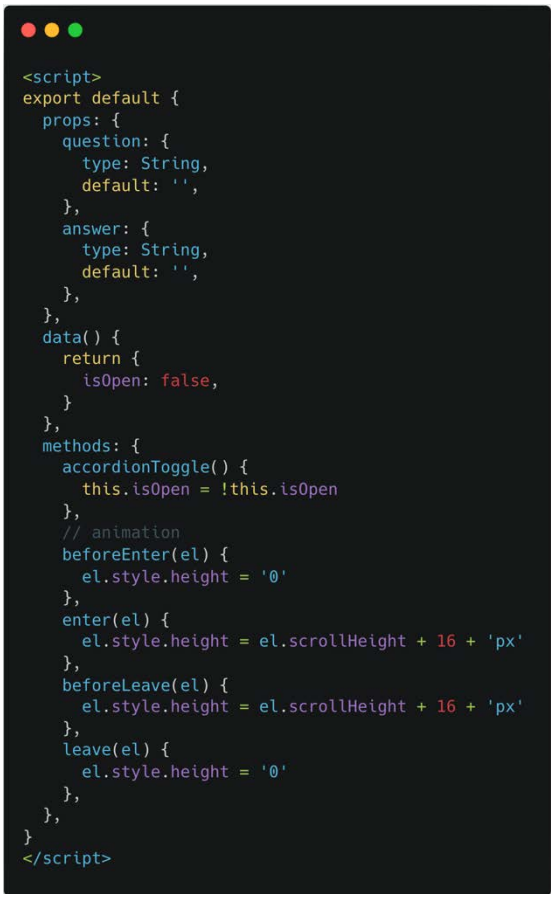
2-3 PaaS のプランが開発当初と変わっていたこと

本アプリケーションでは、PaaS に Heroku を用いている。Microsoft Azure、Amazon Web Services、Vercel などの他サービスを比較検討した上で、無料で GitHub の組織リポジトリからデプロイできることや、クレジットの登録が不要なことなどを理由に Heroku を選定していた。Heroku では、月 550 時間、無料で Web Dyno を利用することができる。ただし、Web Dyno は、30 分以上アクセスが無いとスリープ状態になってしまい、アプリケーションの起動時間が長くなってしまう。開発

当初は、30 分に一度 URL にアクセスするスクリプトを実行することでスリープすることを防いでいたが、550 時間の制限があるため月末 20 日あたりでサーバが停止してしまっていた。本ミッションとして予算申請することによって、Microsoft Azure や、Amazon Web Services を用いたデプロイを想定していたが、Vercel の無料プランの体験版で GitHub の組織アカウントからデプロイすることが可能になっていたため、予算を執行せずに Vercel に移行することにした。Vercel に移行するにあたって、Nuxt.js と Firebase を用いたプロジェクトではエラーが発生するトラブルがあったが、GitHub にある同一エラーの Issue を参考に設定ファイルを書き、動作を確認することができた。

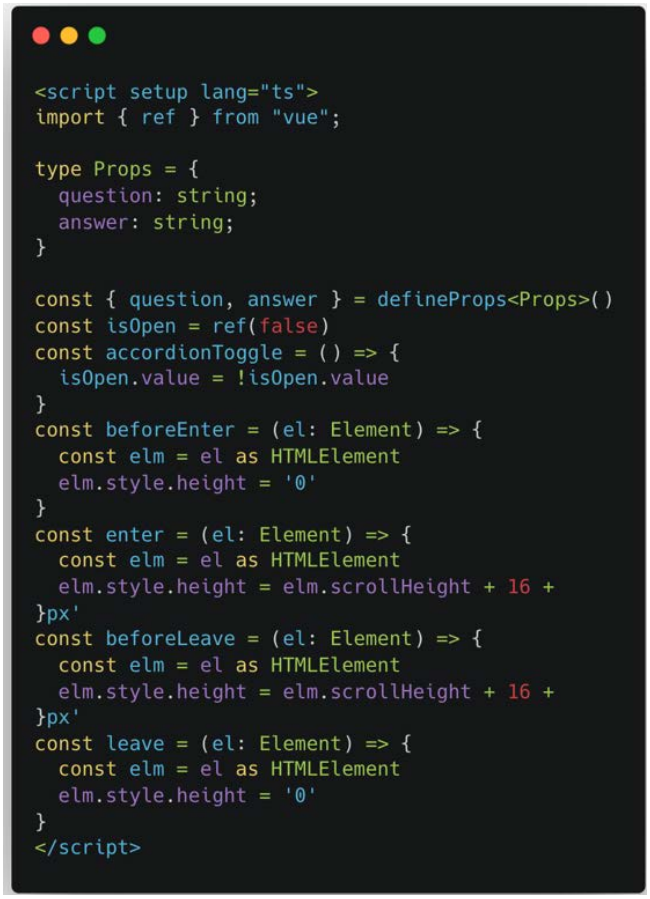
3. 活動の成果や学んだこと

本ミッションでは、古くなったシステムを最新技術に置き換える作業を行い、アプリケーションを継続的に開発し、規模を拡大していくための基盤を整えることができた。また、技術トレンドの変化についていく文化をプロジェクト内に根付かせることができた。バージョンアップに伴うコードの具体的な変化は以下の図の通りである。



```vue
<script>
export default {
  props: {
    question: {
      type: String,
      default: '',
    },
    answer: {
      type: String,
      default: '',
    },
  },
  data() {
    return {
      isOpen: false,
    }
  },
  methods: {
    accordionToggle() {
      this.isOpen = !this.isOpen
    },
    // animation
    beforeEnter(el) {
      el.style.height = '0'
    },
    enter(el) {
      el.style.height = el.scrollHeight + 16 + 'px'
    },
    beforeLeave(el) {
      el.style.height = el.scrollHeight + 16 + 'px'
    },
    leave(el) {
      el.style.height = '0'
    },
  },
}
</script>
```

図 2 Nuxt2 (Options API)を用いたコード



```vue
<script setup lang="ts">
import { ref } from "vue";

type Props = {
  question: string;
  answer: string;
}

const { question, answer } = defineProps<Props>()
const isOpen = ref(false)
const accordionToggle = () => {
  isOpen.value = !isOpen.value
}
const beforeEnter = (el: Element) => {
  const elm = el as HTMLElement
  elm.style.height = '0'
}
const enter = (el: Element) => {
  const elm = el as HTMLElement
  elm.style.height = elm.scrollHeight + 16 +
}px'
const beforeLeave = (el: Element) => {
  const elm = el as HTMLElement
  elm.style.height = elm.scrollHeight + 16 +
}px'
const leave = (el: Element) => {
  const elm = el as HTMLElement
  elm.style.height = '0'
}
</script>
```

図 3 Nuxt3(Composition API) + TypeScript を用いたコード

上記コードは、質問と回答を表示するための簡単なコンポーネントである。行数が多くなるため、テンプレート部分とスタイル部分は省略している。

図上部の props の型宣言について、図 3 では defineProps を用いることで一般的な TypeScript と同じ形式で書くことができるようになっている。また分割代入にも対応しており、値へのアクセスもしやすくなった。TypeScript で型定義できるようになったため、誤ったプロパティへのアクセスや代入、null や undefined チェックの不足、数値を入れるはずの変数に文字列やオブジェクトが入っているなど、型がわからないことに起因するバグの調査や修正が不要となった。さらに、エディタの入力補完の恩恵も受けることができた。図 2 では、data や methods、props などの Options に

分けて書かれているところが、図３では関数ベースになっていることがわかる。これによって、ロジックの分離が可能になり、より見通しの良いコードにすることができた。さらにコードの行数も削減された。

4. 今後の展開

完全に移行作業を完了させることができていないため、今後完全に移行できるようにしたい。また、本アプリケーションをさらに広く利用できるアプリにするために、ユーザー認証機能を導入するなど、機能開発を進めていきたい。さらに、本ミッションで得ることができた知見を情報共有サービスである、Zenn や Qiita 等で共有していきたい。

5. まとめ

本ミッションでは、古くなったシステムを最新技術で置き換えることで、アプリケーションを継続的に開発する基盤を整えることができた。また、技術的な負債をどのように返却していくかの知見を得ることができた。今後は、これらの知見を情報共有サービス等を利用して公開していきたい。特に、Nuxt 2 から Nuxt 3 に移行する作業はあまり資料が充実しておらず、需要があると考えている。このような外部発信をしていくことで、内部では積極的に発信できる環境と文化が形成され自主的な活動が活発化すると考えている。また外部では、IT 人材が育ちやすい環境として一層プロジェクトの認知度が向上することに加えて、純粋な社会貢献にもなると考えている。このようなプロジェクト内外での成長プロセスを確立させていきたい。