PRAKTYKI	Dokumentacja projektu
Autor	Dawid Olko
Kierunek, rok	Informatyka, III rok, st. stacjonarne (3,5-l)
Temat projektu	*Responsywna strona internetowa VUE.js*

Simple Layout

12.02.2025r. - 18.02.2025r.

Spis treści

1. Narzędzia i technologie

1.1 Technologie użyte w projekcie

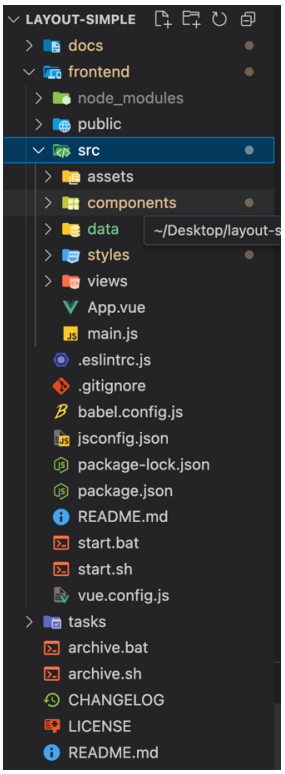
Projekt został zbudowany przy użyciu frameworka **Vue 3** w wersji ^3.2.13. Główne technologie i narzędzia to:

- **Vue CLI** – do tworzenia i konfiguracji aplikacji.
- **Babel** – transpilacja kodu dla kompatybilności z ES5.
- **ESLint** z pluginem **eslint-plugin-vue** – kontrola jakości kodu.
- **core-js** – zapewnienie wsparcia dla nowoczesnych funkcji JavaScript w starszych przeglądarkach.

1.2 Struktura projektu i konfiguracja

Projekt posiada uporządkowaną strukturę:

- **package.json** – definiuje zależności, skrypty (np. npm start, npm run build, npm run lint) oraz konfigurację projektu.
- **Konfiguracja narzędzi:**
  - **vue.config.js** – ustawienia Webpacka z wykorzystaniem DefinePlugin.
  - **babel.config.js** oraz **.eslintrc.js** – konfiguracje dla Babel i ESLint.
  - **tsconfig.json** – (jeśli używany) dla wsparcia TypeScript.
- **Struktura katalogów:**
  Główny folder src/ zawiera wszystkie komponenty i widoki.



1.3 Narzędzia do zarządzania projektem

Do zarządzania projektem wykorzystano:

- **Git** – system kontroli wersji.
- **Repozytorium** (tj. GitLab) – do przechowywania kodu.
- **npm** – zarządzanie zależnościami.
- **Skrypty wsadowe (.bat, .sh)** – automatyzacja instalacji zależności oraz uruchamiania aplikacji.

1.4 Proces budowania i uruchamiania aplikacji

- **Skrypty w package.json:**
    - `npm start lub npm run serve` – uruchomienie serwera developerskiego.
    - `npm run build` – budowanie wersji produkcyjnej.
    - `npm run lint` – sprawdzanie jakości kodu.
- **Skrypty wsadowe (.bat, .sh):**
  Automatyzują instalację (npm install) i uruchomienie projektu, co ułatwia rozpoczęcie pracy.

1.5 Konfiguracja narzędzi budujących

- **Webpack (vue.config.js):** Umożliwia definiowanie globalnych zmiennych i konfigurację builda.
- **Babel:** Zapewnia kompatybilność kodu.
- **ESLint:** Utrzymuje jednolity standard kodowania.

2. Sekcje strony

2.1 Aktualnosci.vue

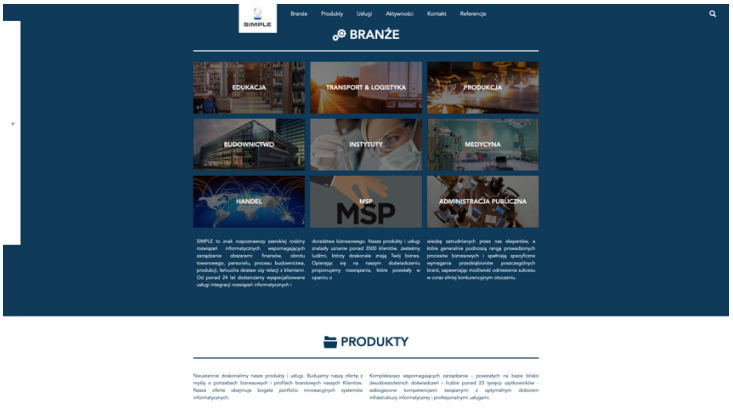
Komponent **Aktualnosci.vue** odpowiada za prezentację najnowszych informacji i wiadomości na stronie. Jego główne zadania to:

- Wyświetlanie dynamicznie aktualizowanych treści, pobieranych z plików danych (np. AktualnosciData.js).
- Implementacja slidera lub listy z aktualnościami, dzięki czemu użytkownik może przeglądać najważniejsze wpisy.
- Zapewnienie responsywności, tak aby treści były czytelne zarówno na urządzeniach mobilnych, jak i na komputerach stacjonarnych.

2.2 Branze.vue

Komponent **Branze.vue** służy do prezentacji różnych sektorów działalności (branż) firmy. Kluczowe cechy tego komponentu to:

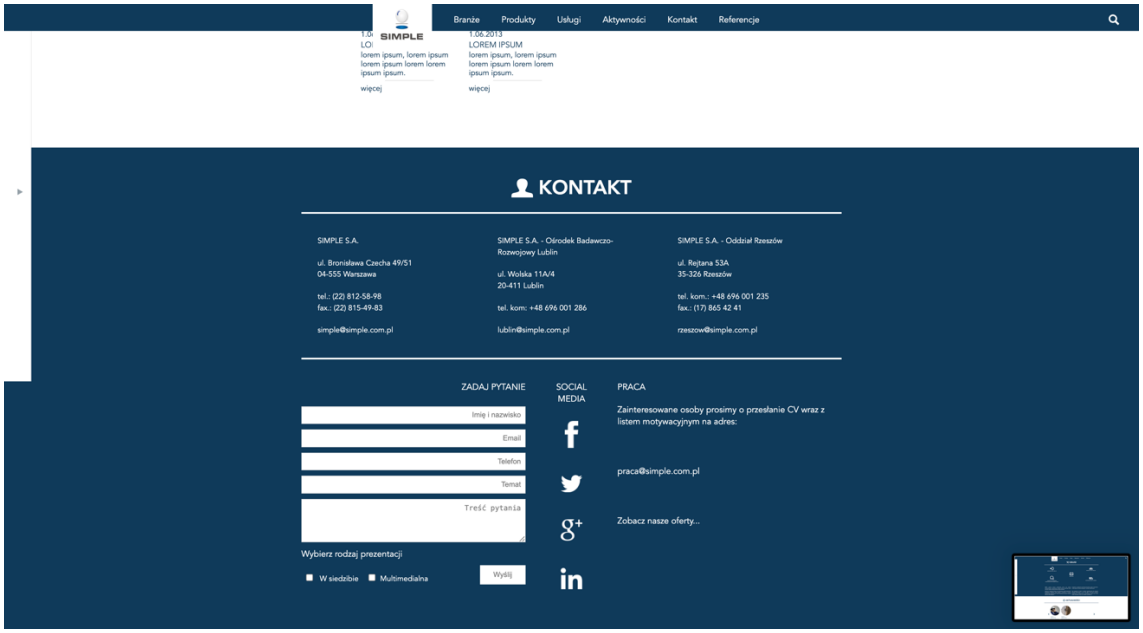
- Prezentacja graficzna 9 sektorów, często z efektami hover, np. zmiana tła lub podświetlenie danej sekcji.
- Wykorzystanie oddzielnych plików danych (np. BranzeData.js), które umożliwiają łatwą modyfikację prezentowanych informacji.
- Modularność, umożliwiająca szybkie dodawanie lub modyfikację wyświetlanych branż.



2.3 Footer.vue

Komponent **Footer.vue** odpowiada za stopkę strony. W jego skład wchodzą:

- Sekcja kontaktowa z formularzem oraz danymi kontaktowymi.
- Ikony mediów społecznościowych umożliwiające szybki dostęp do profili firmy.
- Slider z logotypami partnerów lub referencjami, który dodatkowo wzbogaca wizualnie stopkę.
- Wykorzystanie dedykowanych plików danych (np. FooterData.js) oraz stylów, co zapewnia spójność i łatwość aktualizacji.



2.4 LogaFirm.vue

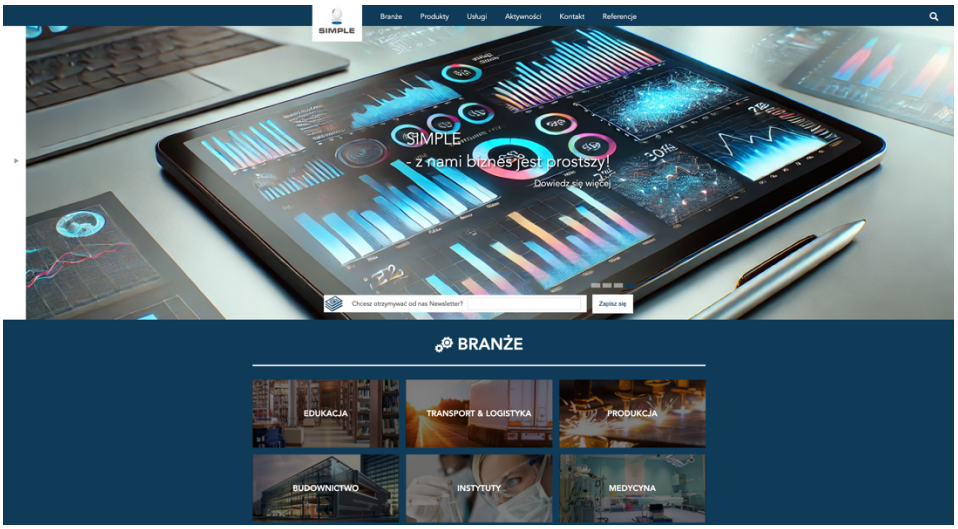
Komponent **LogaFirm.vue** specjalizuje się w wyświetlaniu logotypów firm partnerskich lub klientów. Jego główne zadania to:

- Prezentacja serii logotypów w formie slidera lub statycznej listy.
- Zapewnienie jednolitego stylu oraz responsywności, aby loga były czytelne na każdym urządzeniu.
- Integracja z danymi, które można łatwo aktualizować bez modyfikacji samego komponentu.

2.5 Navbar.vue

Komponent **Navbar.vue** tworzy główną nawigację strony. Do jego podstawowych funkcji należą:

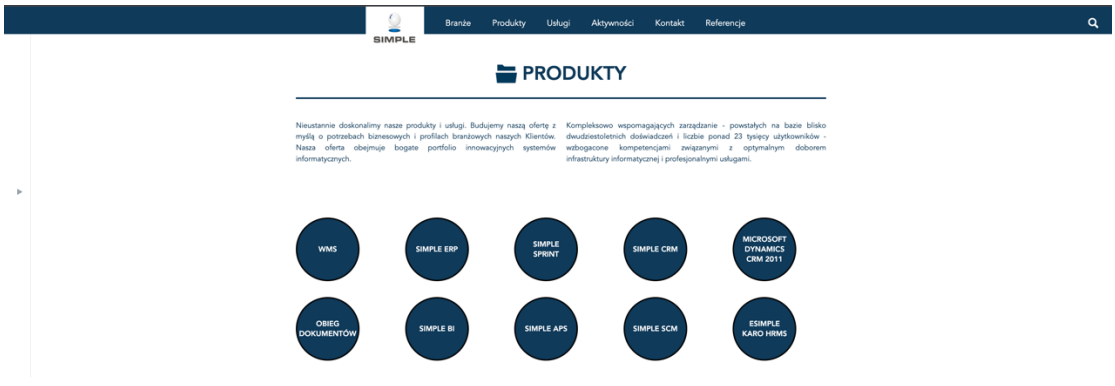
- Wyświetlanie menu z linkami do najważniejszych sekcji strony (takich jak: Produkty, Usługi, Aktualności, itd.).
- Obsługa zdarzeń kliknięć, które pozwalają na płynne przewijanie lub przełączanie widoków.
- Wykorzystanie responsywnego designu, umożliwiającego dostosowanie menu do różnych rozdzielczości ekranu.



2.6 Produkty.vue

Komponent **Produkty.vue** odpowiada za prezentację oferty produktów firmy. Jego główne cechy to:

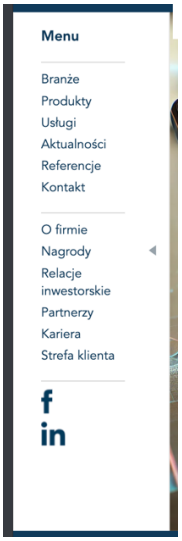
- Wizualna prezentacja 10 elementów (np. w formie okrągłych przycisków lub kafelków) z nazwami produktów.
- Łatwość aktualizacji danych dzięki oddzielnym plikom (np. ProduktyData.js).
- Zastosowanie stylów, które podkreślają estetykę i przejrzystość prezentowanych produktów.



2.7 Sidebar.vue

Komponent **Sidebar.vue** tworzy pasek boczny, który uzupełnia główną nawigację. Do jego zadań należy:

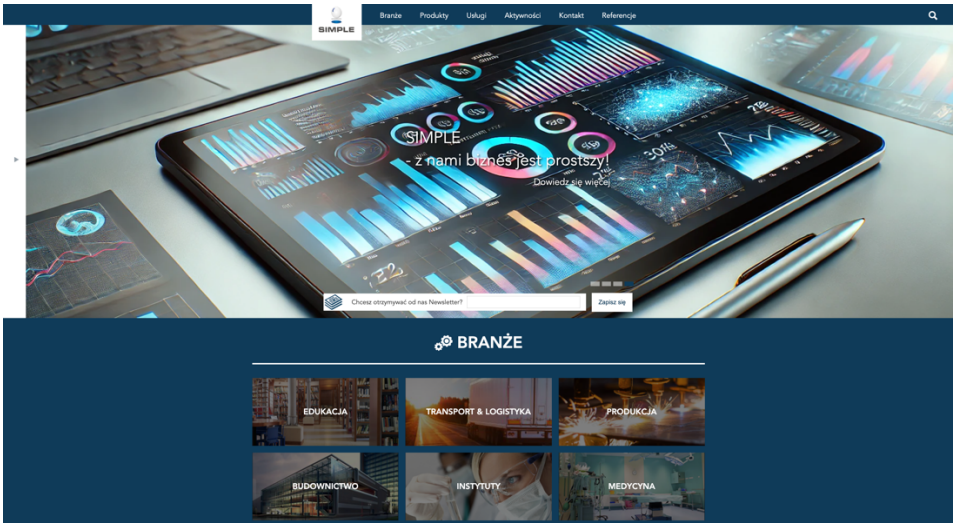
- Wyświetlanie dodatkowego menu z opcjami takimi jak: o nas, kariera, partnerzy, inwestorzy, strefa klienta.
- Prezentacja ikon mediów społecznościowych, co umożliwia szybki dostęp do profili na platformach takich jak Facebook czy LinkedIn.
- Integracja z resztą layoutu w sposób umożliwiający spójne działanie aplikacji na różnych urządzeniach.



2.8 Slider.vue

Komponent **Slider.vue** implementuje karuzelę obrazów, która jest często wykorzystywana do:

- Prezentacji wizualnych elementów strony, takich jak banery promocyjne czy galerie zdjęć.
- Integracji formularza subskrypcji newslettera, co pozwala użytkownikom na szybkie zapisanie się do biuletynu.
- Zastosowania efektów przejścia i animacji, które wzbogacają interfejs użytkownika.



2.9 Uslugi.vue

Komponent **Uslugi.vue** odpowiada za wyświetlanie oferty usług firmy. Jego główne funkcjonalności to:

- Prezentacja ikon oraz opisów poszczególnych usług, co umożliwia szybkie zorientowanie się w ofercie.
- Wykorzystanie oddzielnych plików danych (np. UslugiData.js) umożliwiających łatwą aktualizację treści.
- Zastosowanie dedykowanych stylów, które zapewniają spójność wizualną z resztą strony.

3. GUI

3.1 Wprowadzenie

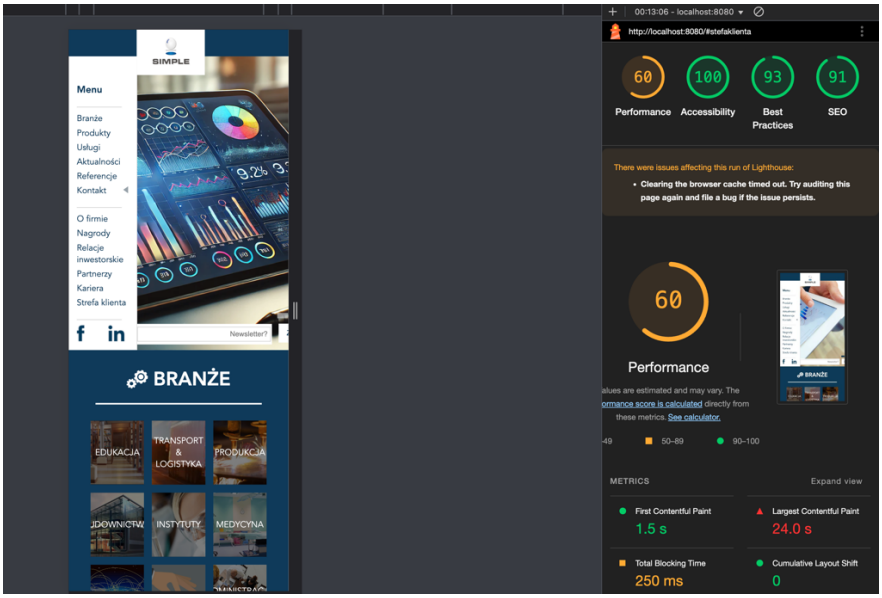
W ramach oceny interfejsu użytkownika (GUI) projektu wykonano testy przy użyciu narzędzia Lighthouse. Testy te pozwalają na ocenę wydajności, dostępności, najlepszych praktyk oraz SEO zarówno w wersji mobilnej, jak i desktopowej. Poniżej przedstawiono wyniki obu testów wraz z kluczowymi uwagami dotyczącymi optymalizacji i doświadczenia użytkownika.

3.2 Lighthouse – Wersja Mobilna

Lighthouse Mobile wykonuje analizę interfejsu pod kątem urządzeń mobilnych, uwzględniając specyfikę mniejszych ekranów oraz warunki sieciowe. Główne aspekty testu to:

- **Wydajność:**
  Ocena czasu ładowania strony, optymalizacji zasobów (obrazów, skryptów) oraz responsywności. Test wskazuje, czy aplikacja spełnia wymogi szybkości działania na urządzeniach mobilnych.
- **Dostępność:**
  Analiza, czy interfejs jest przyjazny dla użytkowników mobilnych, w tym osoby korzystające z czytników ekranu. Sprawdzone są m.in. kontrast kolorów, czytelność tekstu oraz intuicyjność nawigacji.
- **Najlepsze praktyki:**
  Test sprawdza zgodność implementacji z nowoczesnymi standardami webowymi oraz stosowanie rekomendowanych rozwiązań.
- **SEO:**
  Ocena elementów wpływających na optymalizację strony pod kątem wyszukiwarek internetowych.

Wyniki testu na zdjęciu:



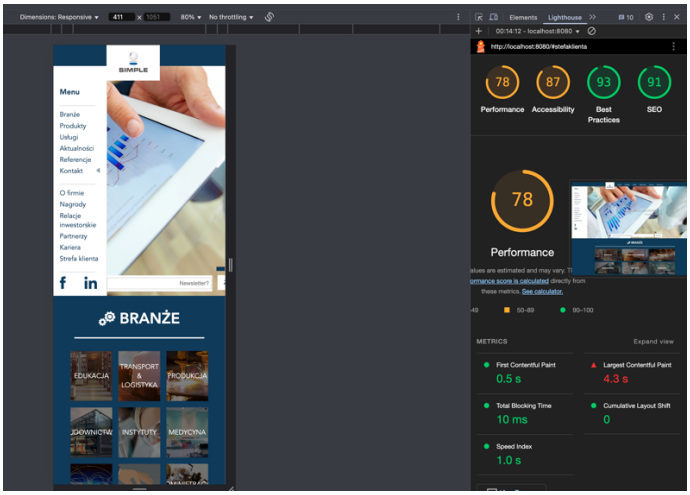
miejsce na zrzut ekranu z wynikami testu Lighthouse dla wersji mobilnej.

3.3 Lighthouse – Wersja Desktop

Lighthouse Desktop przeprowadza ocenę interfejsu na komputerach stacjonarnych, gdzie analizowane są nie tylko kwestie wydajności, ale także estetyki oraz interaktywności na większych ekranach. Kluczowe punkty testu to:

- **Wydajność:**
  Test mierzy czasy ładowania, wykorzystanie zasobów oraz sprawdza, czy strona działa płynnie przy pełnej rozdzielczości ekranu.
- **Dostępność:**
  Podobnie jak w wersji mobilnej, oceniane są aspekty dostępności, takie jak czytelność, kontrast oraz użyteczność interfejsu dla użytkowników z różnymi potrzebami.
- **Najlepsze praktyki:**
  Analiza zgodności kodu i implementacji z nowoczesnymi standardami, co wpływa na jakość oraz bezpieczeństwo strony.
- **SEO:**
  Ocena elementów optymalizacyjnych, które wpływają na widoczność strony w wyszukiwarkach.

Wyniki testu na zdjęciu:



miejsce na zrzut ekranu z wynikami testu Lighthouse dla wersji desktopowej.

3.4 Podsumowanie

Wyniki testów Lighthouse dla wersji mobilnej i desktopowej wskazują na wysoką jakość implementacji interfejsu użytkownika. Oceniono:

- **Wydajność:** Aplikacja charakteryzuje się szybkim ładowaniem i płynną obsługą na obu platformach.
- **Dostępność:** Zarówno wersja mobilna, jak i desktopowa spełniają standardy dostępności, co wpływa na komfort korzystania przez szeroką grupę użytkowników.
- **Najlepsze praktyki i SEO:** Strona implementuje nowoczesne rozwiązania zgodne z rekomendacjami Lighthouse, co przekłada się na jej wysoką ocenę w obu wersjach.

4. Uruchomienie aplikacji

4.1 Wymagania systemowe

Aby uruchomić aplikację frontendową stworzoną przy użyciu Vue, należy spełnić następujące wymagania:

- **Node.js:** Wersja 16+
- **npm:** Wersja 8+ (lub yarn)
- **Przeglądarka:** Wspierająca nowoczesne standardy JavaScript (np. Chrome, Firefox, Edge)

4.2 Konfiguracja i uruchomienie projektu

1. **Klonowanie repozytorium:**
   Pobierz kod źródłowy projektu:

   ```
   git clone https://gitlab.ideo.pl/m.koszyk/layout-simple
   cd layout-simple/frontend
   ```

2. **Instalacja zależności:**
   Zainstaluj wymagane pakiety, wpisując:

   ```
   npm install
   ```

3. **Uruchomienie aplikacji:**
   Aby rozpocząć pracę w trybie developerskim, użyj komendy:

   ```
   npm run serve
   ```

   lub (w zależności od konfiguracji):

   ```
   npm start
   ```

Aplikacja zostanie uruchomiona i będzie dostępna pod adresem: `http://localhost:8080` – lub innym, zgodnie z komunikatem w konsoli.

4.3 Budowanie wersji produkcyjnej

Aby przygotować zoptymalizowaną wersję aplikacji do wdrożenia na serwerze produkcyjnym, wykonaj następującą komendę:

```
npm run build
```

Po zakończeniu procesu, wygenerowane pliki znajdziesz w katalogu (np. dist/), które można przesłać na serwer WWW.

5. Podsumowanie

5.1 Kluczowe funkcjonalności

Projekt **Layout-Simple** to prosty, responsywny layout strony biznesowej, który dzięki zastosowaniu modularnego podejścia pozwala na łatwą edycję i rozwój. Kluczowe cechy projektu to:

- **Modułowość:**
  Aplikacja została podzielona na oddzielne komponenty (np. Navbar, Sidebar, Slider, Aktualnosci, Branze, Produkty, Uslugi, Footer, LogaFirm), co umożliwia szybkie modyfikacje poszczególnych sekcji bez ingerencji w całość.
- **Responsywność:**
  Dzięki zastosowaniu nowoczesnych technik CSS oraz podejścia mobile-first, układ strony dostosowuje się do różnych rozdzielczości ekranów – zarówno na urządzeniach mobilnych, jak i stacjonarnych.
- **Przejrzysta struktura:**
  Oddzielenie logiki komponentów od danych (przechowywanych w osobnych plikach) oraz dedykowane arkusze stylów dla każdego elementu wpływają na łatwość zarządzania kodem i utrzymanie projektu.
- **Intuicyjny interfejs:**
  Zastosowany design pozwala na szybkie odnalezienie potrzebnych informacji, co przekłada się na pozytywne doświadczenie użytkownika.

5.2 Technologie wykorzystane w projekcie

W projekcie Layout-Simple wykorzystano następujące narzędzia i technologie:

- **Vue.js 3:**
  Główny framework umożliwiający budowanie interfejsu użytkownika w oparciu o komponenty.
- **Vue CLI:**
  Narzędzie do szybkiej konfiguracji i zarządzania projektem, zapewniające spójność środowiska developerskiego.
- **Node.js oraz npm:**
  Środowisko wykonawcze i menedżer pakietów, które umożliwiają instalację zależności oraz uruchomienie aplikacji.
- **Babel i ESLint:**
  Narzędzia zapewniające kompatybilność kodu z różnymi przeglądarkami oraz utrzymanie wysokiej jakości kodu poprzez automatyczną analizę.
- **CSS:**
  Dedykowane arkusze stylów dla każdego komponentu, umożliwiające precyzyjne dopasowanie wyglądu aplikacji.

5.3 Wnioski i dalsze możliwości rozwoju

Projekt **Layout-Simple** stanowi solidną bazę do dalszych modyfikacji i rozbudowy. Warto podkreślić następujące aspekty:

- **Łatwość modyfikacji:**
  Modułowa struktura aplikacji pozwala na szybkie wprowadzanie zmian w pojedynczych sekcjach bez ryzyka uszkodzenia całości projektu.
- **Skalowalność:**
  Dzięki oddzieleniu logiki, danych i stylów, projekt można łatwo rozbudowywać o nowe funkcjonalności, takie jak dodatkowe sekcje, integracje z API czy zaawansowane mechanizmy personalizacji treści.
- **Potencjał integracji:**
  Możliwość dodania narzędzi analitycznych, systemów komentarzy czy dynamicznych filtrów treści pozwala na ciągłe ulepszanie doświadczenia użytkownika.
- **Rozwój interfejsu:**
  W przyszłości warto rozważyć wdrożenie dodatkowych usprawnień interfejsu, np. animacji czy mikrointerakcji, które mogą jeszcze bardziej wzbogacić prezentację strony.

5.4 Podsumowanie końcowe

Podsumowując, projekt **Layout-Simple** to nowoczesne, responsywne rozwiązanie oparte na Vue, które cechuje się przejrzystością, łatwością modyfikacji oraz intuicyjnym interfejsem użytkownika. Dzięki zastosowaniu nowoczesnych technologii frontendowych, aplikacja stanowi solidną podstawę do dalszego rozwoju i adaptacji do zmieniających się potrzeb rynkowych. Projekt jest doskonałym przykładem na to, jak modularne podejście i wydzielenie kluczowych funkcjonalności wpływa na skalowalność i efektywność zarządzania kodem.

Jeśli pojawią się nowe pomysły lub wymagania, projekt można łatwo rozbudować o dodatkowe funkcje, co czyni go elastycznym narzędziem dla różnych zastosowań w środowisku biznesowym.