Projektowanie i programowanie systemów internetowych II

Ice Cream Places

1 Opis funkcjonalny systemu

Free Parking to system mający na celu ułatwienie użytkownikom znalezienia darmowych miejsc parkingowych w Polsce. Aplikacja zapewnia różne funkcje, które pomagają w tym celu. Dzięki temu użytkownicy mogą zaoszczędzić czas i pieniądze, a także uniknąć stresu związanego z szukaniem parkingu.

1.1 Aplikacja oferuje szereg funkcjonalności, takich jak:

- **Autentykacja za pomocą e-maila i hasła** Użytkownicy mogą zarejestrować się w systemie, podając swój adres e-mail i hasło. Po rejestracji, mogą się logować do systemu za pomocą tych danych.
- **Autentykacja Google** Użytkownicy mają również możliwość rejestracji i logowania za pomocą swojego konta Google.
- **Mapa z lokalizacjami** System wyświetla mapę Polski, na której zaznaczone są lokalizacje darmowych parkingów. Użytkownicy mogą przeglądać mapę, aby znaleźć najbliższy dostępny parking.
- **Ulubione parkingi** Użytkownicy mogą dodawać parkingi do listy ulubionych, co ułatwia szybki dostęp do informacji o tych parkingach. Lista ulubionych parkingów jest dostępna w panelu bocznym.
- **Wyszukiwanie parkingów** Użytkownicy mogą wyszukiwać parkingi według miast.
- **Edycja profilu** Użytkownicy mogą zmieniać swoją nazwę użytkownika oraz zmieniać hasło.
- **Admin panel** Administrator może edytować parkingi i użytkowników.

2 Streszczenie opisu technologicznego

Lista najważniejszych technologii i bibliotek, na bazie których powstała aplikacja:

- **Next.js** - framework dla Reacta, który umożliwia tworzenie aplikacji full-stack, rozszerzając najnowsze funkcje Reacta i integrując potężne narzędzia oparte na Rust do najszybszych kompilacji.

- **TypeScript** - silnie typowany język programowania, który buduje na JavaScript, dając lepsze narzędzia na dowolną skalę.
- **Prisma** - nowoczesny ORM dla Node.js i TypeScript, obsługujący PostgreSQL, MySQL itp. Zapewnia bezpieczeństwo typów, automatyczne migracje i intuicyjny model danych.
- **React** - biblioteka do tworzenia interfejsów użytkownika z indywidualnych elementów zwanych komponentami.
- **React Hook Form** - prosta i łatwa biblioteka do walidacji formularzy, która redukuje ilość kodu, której potrzebujesz do napisania, jednocześnie eliminując niepotrzebne ponowne renderowania.
- **TanStack Query** - jest często opisywany jako brakująca biblioteka do pobierania danych dla aplikacji internetowych, ale z bardziej technicznego punktu widzenia sprawia, że pobieranie, buforowanie, synchronizowanie i aktualizowanie stanu serwera w aplikacjach internetowych jest dziecinnie proste.
- **React Icons** - pakiet, który pozwala na importowanie i używanie popularnych ikon z różnych bibliotek ikon w projektach React.
- **Zod** - biblioteka do deklarowania i walidacji schematów w TypeScript, z inferencją statycznego typu.
- **ChakraUI** - biblioteka komponentów interfejsu użytkownika dla Reacta, umożliwiająca szybkie tworzenie atrakcyjnych i spójnych interfejsów.
- **GitHub** - miejsce do kontroli wersji kodu. Pozwala na łatwą synchronizację między członkami grupy oraz na prostą kontrolę zmian w kodzie.
- **ESLint** - narzędzie do analizy statycznej kodu JavaScript w celu wykrycia problemów i niezgodności ze standardami kodowania.
- **Prettier** - narzędzie do formatowania kodu, zapewniające spójny styl kodowania w całym projekcie.

3 Instrukcja lokalnego i zdalnego uruchomienia

3.1 Lokalnie

- Najpierw należy pobrać i zainstalować Node.js ze strony

```
https://nodejs.org/en/
```

- Następnie należy sklonować repozytorium za pomocą

```
git clone https://github.com/parkingbezplatny/app.git
```

- Kolejno otworzyć terminal oraz przejść do folderu aplikacji oraz wpisać

```
npm install
```

- Po instalacji wystarczy wpisać w terminalu

```
npm run dev
```

- Po chwili kompilacji powinna otworzyć się nasza aplikacja pod adresem

```
http://localhost:3000/
```

- Aplikacja potrzebuje do działania pliku **.env**, w którym będą następujące informacje:
  - **DATABASE_URL** - URL do bazy danych
  - **DIRECT_URL** - direct URL do bazy danych
  - **GOOGLE_CLIENT_ID** - clientId otrzymane od Google
  - **GOOGLE_CLIENT_SECRET** - clientSecret otrzymany od Google
  - **NEXTAUTH_URL** - URL strony dla biblioteki NextAuth (http://localhost:3000)
  - **NEXTAUTH_SECRET** - klucz secret dla biblioteki NextAuth
  - **NEXT_PUBLIC_MAP_API_KEY** - klucz do API dostawcy mapy
  - **NEXT_PUBLIC_HERE_API_KEY** - klucz do API dostawcy geocodingu

3.2 Zdalnie

- Strona internetowa aplikacji jest dostępna pod adresem

```
https://freeparkingapp.vercel.app/
```

3.3 Testy

- Po uruchomieniu lokalnie uruchamiając terminal w folderze z projektem wpisać

```
npm run test
```

- Spis scenariuszy testowych wygenerowany po uruchomieniu testów:

```
PASS  __tests__/user/ChangePasswordModal.test.tsx
  ChangePasswordModal
    renders ChangePasswordModal correctly (762 ms)

PASS  __tests__/user/ChangeUsernameModal.test.tsx
  ChangeUsernameModal
    renders ChangeUsernameModal correctly (494 ms)

 PASS  __tests__/Sidepanel.test.tsx
```

```
  SidePanel
    renders SidePanel correctly (109 ms)
    handles click event correctly (34 ms)

 PASS  __tests__/AdminSidepanel.test.tsx
  AdminSidePanel
    renders AdminSidePanel correctly (104 ms)

 PASS  __tests__/WelcomePage.test.tsx
  WelcomePage
    redirects to dashboard if user is authenticated (104 ms)
    renders the WelcomePage component correctly (45 ms)

 PASS  __tests__/Navbar.test.tsx
  Navbar
    renders Navbar correctly (157 ms)
    handles click event correctly (104 ms)

 PASS  __tests__/MapTooltip.test.tsx
  MapTooltip
    renders MapTooltip correctly (71 ms)

 PASS  __tests__/Search.test.tsx
  Search
    renders Search correctly (122 ms)
    handles error correctly (55 ms)

 PASS  __tests__/AdminCard.test.tsx
  AdminCard
    renders AdminCard component correctly (59 ms)
    renders AdminCard component without footer (14 ms)

Test Suites: 9 passed, 9 total
Tests:       14 passed, 14 total
Snapshots:   0 total
Time:        17.822 s, estimated 18 s
```

4 Linki do dokumentacji projektu

- **GitHub** - https://github.com/parkingbezplatny/app

5 Wnioski projektowe

Podczas realizacji projektu systemu Free Parking, napotkaliśmy na kilka wyzwań. Najbardziej wymagającym elementem było zaimplementowanie mapy oraz zarządzanie dużą ilością danych o parkingach. Te zadania wymagały skomplikowanych rozwiązań technicznych i dużej ilości czasu na analizę i optymalizację. Mimo tych wyzwań, naszym głównym celem było stworzenie intuicyjnego i prostego interfejsu dla użytkowników. Dzięki ciężkiej pracy i zaangażowaniu zespołu, udało nam się osiągnąć ten cel. Interfejs jest łatwy w obsłudze, a funkcje, takie jak wyszukiwanie parkingów czy dodawanie do ulubionych, są proste i intuicyjne. Podsumowując, projekt zakończył się sukcesem. Współpraca w zespole była owocna, a wynikiem naszej pracy jest funkcjonalny i przyjazny dla użytkownika system.