Dokumentacja Projektu grupowego

Dokumentacja techniczna projektu

Wydział Elektroniki, Telekomunikacji i Informatyki

Politechnika Gdańska

wersja 2/2023

Nazwa i akronim projektu: *Shopper - Aplikacja asystująca realizację zakupów w sklepie wielkopowierzchniowym.*	**Zleceniodawca:** *Dr hab. inż. Rafał Lech*	
Numer zlecenia: *9@KIMA'2023/2024*	**Kierownik projektu:** *Maciej Danielewicz*	**Opiekun projektu:** *Dr hab. inż. Rafał Lech*

Nazwa / kod dokumentu: **Dokumentacja techniczna produktu – DTP**	**Nr wersji:** *1.00*
Odpowiedzialny za dokument: *Janicki, Julian*	**Data pierwszego sporządzenia:** *25.01.2024*
	Data ostatniej aktualizacji: *25.01.2024*
	Semestr realizacji Projektu grupowego: *1*

Historia dokumentu

Wersja	Opis modyfikacji	Rozdział / strona	Autor modyfikacji	Data
1.00	*Wstępna wersja*	*Całość*	*Janicki, Julian*	*25.01.2024*

Spis treści

1. Wprowadzenie - o dokumencie
    1.1. Cel dokumentu
    1.2. Zakres dokumentu
    1.3. Odbiorcy
    1.4. Terminologia

2. Dokumentacja techniczna projektu
    2.1. Architektura systemu
        2.1.1. Ogólny przegląd architektury
        2.1.2. Składniki systemu i ich zależności
    2.2. Technologie i narzędzia
        2.2.1. Wykorzystane technologie
        2.2.2. Środowisko programistyczne i wspomagające
    2.3. Projekt interfejsu użytkownika
        2.3.1. Zasady projektowania UI/UX
        2.3.2. Prototypowanie i wzornictwo
        2.3.3. Paleta barw i logo
    2.4. Implementacja aplikacji
        2.4.1. Struktura frontendu
        2.4.2. Struktura backendu
        2.4.3. Integracja z bazą danych
    2.5. Nawigacja i lokalizacja
        2.5.1. Implementacja technologii Bluetooth BLE
        2.5.2. Algorytmy wyznaczania trasy
        2.5.3. Zarządzanie sygnałami
    2.6. Bezpieczeństwo i wydajność
        2.6.1. Zabezpieczenia aplikacji
        2.6.2. Optymalizacja wydajności
3. Załączniki
    3.1. Lista załączników

1. Wprowadzenie - o dokumencie

1.1. Cel dokumentu

Celem dokumentu jest udokumentowanie informacji dotyczących produktu, jego cech funkcjonalnych, parametrów technicznych, schematów blokowych, oprogramowania, wyników działania, zdjęć produktu, pomiarów, testów oraz innych elementów wymaganych przez opiekuna i klienta.

1.2. Zakres dokumentu

Zakres dokumentu obejmuje architekturę systemu, użyte technologie i narzędzia, projekt interfejsu użytkownika, plan implementacji i testowania, a także strategie dotyczące bezpieczeństwa, wydajności oraz dokumentacji i wsparcia. Dokument nie obejmuje natomiast szczegółowych planów biznesowych, analiz rynkowych czy strategii marketingowych.

1.3. Odbiorcy

- Opiekun projektu i zleceniodawca:
  Dr hab. inż. Rafał Lech
- Członkowie zespołu projektowego:
  Julian Janicki, Michał Jaskulski, Maciej Danielewicz, Kacper Kulaszewicz

1.4. Terminologia

- Nadajniki BLE
  Nadajniki Bluetooth Low Energy, kluczowe w technologii lokalizacji wewnątrz budynków.
- Kierownik Projektu
  Osoba odpowiedzialna za nadzorowanie i koordynację prac projektowych.
- Frontend
  Część aplikacji z którą bezpośrednio wchodzi w interakcję użytkownik.
- Backend
  Serwerowa część aplikacji, zarządzająca logiką, przetwarzaniem danych i komunikacją z bazą danych.
- NativeScript, Angular17, Node.js, PostgreSQL, Docker
  Główne technologie wykorzystywane w projekcie.
- Figma
  Narzędzie do projektowania UI/UX.
- GitHub
  Platforma do kontroli wersji i współpracy w zespole deweloperskim.

- API REST
  Interfejs programowania aplikacji oparty na architekturze REST, używany do komunikacji między front-endem a back-endem.
- Model-View-ViewModel (MVVM)
  Wzorzec architektoniczny używany w projektowaniu interfejsu użytkownika, oddzielający logikę biznesową od UI.
- BEM (Block, Element, Modifier)
  Metodologia nazewnictwa i organizacji CSS, która pomaga w tworzeniu skalowalnych i łatwych do utrzymania arkuszy stylów.
- ORM (Object-Relational Mapping)
  Technika programistyczna służąca do konwersji danych między niekompatybilnymi systemami typu baza danych i obiektowy język programowania.
- Trilateracja
  Metoda wyznaczania pozycji obiektu poprzez pomiar odległości od grupy znanych punktów.
- A* (A-star)
  Algorytm służący do wyznaczania ścieżki w grafie między węzłem początkowym a docelowym.
- Dijkstra
  Algorytm służący do znajdowania najkrótszej ścieżki między węzłami w grafie.
- HTTPS
  Protokół bezpiecznej komunikacji w Internecie, wykorzystujący szyfrowanie.
- JWT (JSON Web Tokens)
  Metoda bezpiecznej transmisji informacji między dwoma stronami jako obiekt JSON..
- Code Splitting, Tree Shaking
  Techniki optymalizacji kodu w JavaScript, pomagające w redukcji rozmiaru plików i poprawie wydajności ładowania.
- Lazy Loading
  Technika polegająca na ładowaniu elementów (np. obrazów, modułów) tylko wtedy, gdy są one potrzebne, co może znacząco poprawić wydajność aplikacji.

2. Dokumentacja techniczna projektu

2.1. Architektura systemu

2.1.1. Ogólny przegląd architektury

Projekt opiera się na architekturze klient-serwer, gdzie klient mobilny (wykorzystujący NativeScript i Angular17) komunikuje się z serwerem backendowym (Node.js) zarządzającym logiką aplikacji i bazą danych PostgreSQL. System będzie integrował technologię Bluetooth BLE do nawigacji wewnątrz sklepu.

2.1.2. Składniki systemu i ich zależności

- Klient mobilny

Interfejs użytkownika opracowany w NativeScript i Angular17

- Backend

Serwer aplikacji w Node.js, zarządzający przetwarzaniem danych i komunikacją z bazą danych.

- Bazy danych

PostgreSQL do przechowywania informacji o produktach i układzie sklepu.

- Bluetooth BLE

Do lokalizacji użytkownika wewnątrz sklepu.

2.2. Technologie i narzędzia

2.2.1. Wykorzystane technologie

Wybrane technologie (NativeScript, Angular17, Node.js, PostgreSQL) zapewniają elastyczność, wydajność i skalowalność potrzebną do realizacji projektu. Bluetooth BLE jest kluczowy dla funkcji nawigacji.

2.2.2. Środowisko programistyczne i wspomagające

Wykorzystane narzędzia obejmują Visual Studio Code, Docker dla konteneryzacji, GitHub do kontroli wersji i Figma do projektowania UI/UX.

2.3. Projekt interfejsu użytkownika

2.3.1. Zasady projektowania UI/UX

Interfejs użytkownika będzie skoncentrowany na łatwości nawigacji i intuicyjności, z wyraźnymi wizualnymi wskazówkami dla użytkownika.

2.3.2. Prototypowanie i wzornictwo

Wykorzystanie Figma do prototypowania interfejsu użytkownika pozwoli na iteracyjne udoskonalanie projektu.

2.3.3. Paleta barw i logo

Paleta barw i logo aplikacji zostały opracowane, aby były atrakcyjne wizualnie, a zarazem funkcjonalne. Patrz: *Tabela 3.1. Lista załączników.*

2.4. Implementacja aplikacji

2.4.1. Struktura frontendu

Frontend aplikacji zostanie zbudowany z wykorzystaniem Angular17, stosując wzorzec projektowy Model-View-ViewModel (MVVM) dla zapewnienia separacji logiki biznesowej od interfejsu użytkownika. Do stylizacji zastosujemy Sass CSS z architekturą BEM (Block, Element, Modifier), co przyczyni się do zwiększenia czytelności i łatwości utrzymania kodu CSS. Routing w aplikacji będzie zarządzany przez Angular Router, umożliwiający dynamiczne ładowanie komponentów i leniwe ładowanie (lazy loading) modułów, co poprawi wydajność aplikacji.

2.4.2. Struktura backendu

Backend aplikacji zostanie zaimplementowany w Node.js z wykorzystaniem Express.js. Do obsługi asynchronicznych operacji zastosujemy wzorzec Promise i async/await, co poprawi czytelność kodu i ułatwi obsługę błędów. Architektura backendu zostanie oparta na wzorcu mikroserwisów, co umożliwi skalowanie poszczególnych usług niezależnie. Komunikacja z bazą danych PostgreSQL będzie realizowana przez ORM, który umożliwi efektywne zarządzanie schematem bazy danych i złożonymi zapytaniami.

2.4.3. Integracja z bazą danych

Schemat bazy danych PostgreSQL zostanie zaprojektowany zgodnie z zasadami normalizacji, aby zwiększyć integralność danych. Wdrożymy indeksowanie dla często wyszukiwanych kolumn, takich jak identyfikatory sklepów i produktów, aby zwiększyć wydajność zapytań. Struktura bazy danych zostanie dostosowana do potrzeb aplikacji, z tabelami dla użytkowników, produktów, sklepów oraz tras zakupowych.

2.5. Nawigacja i lokalizacja

2.5.1. Implementacja technologii Bluetooth BLE

Bluetooth BLE zostanie wykorzystany do stworzenia systemu indoor positioning, który pozwoli aplikacji na śledzenie pozycji użytkownika w sklepie. Nadajniki BLE będą rozmieszczone w strategicznych punktach sklepu, a algorytmy trilateracji i fingerprintingu pozwolą na dokładne określenie lokalizacji użytkownika.

2.5.2. Algorytmy wyznaczania trasy

Do wyznaczania tras zakupów zastosujemy algorytmy oparte na grafach, takie jak A* lub Dijkstra, aby znaleźć najkrótszą ścieżkę przez sklep. Te algorytmy będą korzystać z danych o układzie sklepu i dostępności produktów, a także z analizy historycznych danych o ruchu klientów, wykorzystując techniki uczenia maszynowego, aby zoptymalizować proponowane trasy. Możliwą alternatywą dla grafu jest praca z macierzą danych.

2.5.3. Zarządzanie sygnałami

Do efektywnego zarządzania sygnałami Bluetooth Low Energy (BLE) i lokalizacji użytkowników w sklepie wykorzystamy bibliotekę Noble w Node.js. Noble jest wszechstronną biblioteką do obsługi BLE, która umożliwia skanowanie urządzeń BLE, odczytywanie ich charakterystyk i interpretowanie sygnałów.

- Skanowanie i Detekcja Urządzeń BLE

  Zaimplementujemy funkcjonalność skanowania BLE w celu wykrywania nadajników rozmieszczonych w sklepie oraz urządzeń użytkowników.

- Obliczanie Pozycji Użytkownika

  Wykorzystamy dane z sygnałów BLE, aby obliczyć pozycję użytkownika w sklepie, stosując metody takie jak trilateracja.

- Integracja z Aplikacją Mobilną i Backendem

  Dane o lokalizacji będą przesyłane z aplikacji mobilnej do backendu w celu dalszej analizy i wyznaczania optymalnej trasy zakupów.

2.6. Bezpieczeństwo i wydajność

2.6.1. Zabezpieczenia aplikacji

Zabezpieczenia aplikacji obejmą stosowanie protokołu HTTPS do szyfrowania komunikacji, JWT (JSON Web Tokens) do uwierzytelniania i autoryzacji użytkowników.

Mechanizmy obrony przed typowymi zagrożeniami, takimi jak SQL Injection i Cross-Site Scripting (XSS), będą wdrożone na wszystkich poziomach aplikacji.

2.6.2. Optymalizacja wydajności

Wykorzystamy caching (Redis lub podobne rozwiązania) do przechowywania często używanych danych, co zredukuje obciążenie bazy danych. Optymalizacja wydajności frontendu obejmie techniki takie jak code splitting, tree shaking oraz lazy loading obrazów i innych zasobów.

3. Załączniki

Tabela 3.1. Lista załączników

L.p.	Nazwa dokumentu	Nazwa pliku
1.	**Logo aplikacji**	*shopper-logo.png*
2.	**Paleta barw**	*shopper-colours.png*