Sprawozdanie z Projektu

System Rejestracji i Analizy Aktywności Fizycznej
"Activis"

Autorzy (Zespół Projektowy):

Jakub Kozubek
Dawid Bazylewicz
Dawid Skowroński

Collegium Witelona w Legnicy
Wydział Nauk Technicznych i Ekonomicznych
Kierunek Informatyka

12 grudnia 2025

Spis treści

1 Opis przedmiotu zamówienia **2**
- 1.1 Składniki systemu . . . 2
- 1.2 Główne funkcjonalności . . . 2

2 Opis technologiczny rozwiązania **3**
- 2.1 Architektura Systemu . . . 3
- 2.2 Stos Technologiczny . . . 3
  - 2.2.1 Backend (PUM_API) . . . 3
  - 2.2.2 Aplikacja Mobilna (pum_project) . . . 3
  - 2.2.3 Panel Administracyjny (PUM_ADMIN) . . . 4

3 Instrukcja uruchomienia i wdrożenia **5**
- 3.1 Uruchomienie Lokalne (Development) . . . 5
  - 3.1.1 Krok 1: Baza Danych i Backend (API) . . . 5
  - 3.1.2 Krok 2: Panel Administracyjny . . . 5
  - 3.1.3 Krok 3: Aplikacja Mobilna . . . 5
- 3.2 Wdrożenie Produkcyjne (Deployment) . . . 5
  - 3.2.1 Backend (API) . . . 5
  - 3.2.2 Panel Administracyjny . . . 6
  - 3.2.3 Aplikacja Mobilna . . . 6

4 Wnioski z pracy projektowej **7**
- 4.1 Status realizacji wymagań . . . 7
- 4.2 Co udało się zrealizować . . . 7
- 4.3 Napotkane trudności i obszary do poprawy . . . 8

Opis przedmiotu zamówienia

Celem projektu było wytworzenie kompletnego systemu informatycznego służącego do rejestrowania, monitorowania oraz analizy aktywności fizycznych użytkowników, wzorowanego na popularnych rozwiązaniach typu Strava.

1.1 Składniki systemu

Zgodnie z wymaganiami, system składa się z trzech integralnych części:

1. **Aplikacja mobilna (Flutter):** Wieloplatformowe narzędzie dla użytkownika końcowego do rejestracji treningów z wykorzystaniem GPS.
2. **Panel administracyjny (Web):** Interfejs przeglądarkowy służący do zarządzania użytkownikami i treściami w systemie.
3. **REST API (Backend):** Serwer pośredniczący w wymianie danych i komunikacji z bazą danych.

1.2 Główne funkcjonalności

System "Activis"realizuje kluczowe założenia projektowe:

- **Rejestracja i Profil:** Możliwość zakładania konta, logowania oraz edycji parametrów fizycznych (waga, wzrost) i avatara.
- **Śledzenie Aktywności:** Rejestracja trasy w czasie rzeczywistym (GPS), obliczanie dystansu, tempa i czasu trwania dla różnych typów aktywności (bieg, rower, marsz).
- **Historia i Statystyki:** Przeglądanie wykonanych treningów, filtrowanie oraz dostęp do globalnych statystyk i rankingów.
- **Zarządzanie (Admin):** Możliwość moderacji treści, usuwania aktywności oraz podglądu statystyk systemowych przez administratora.

Opis technologiczny rozwiązania

Zaproponowane rozwiązanie opiera się na nowoczesnej architekturze klient-serwer. Dzięki zastosowaniu frameworka Flutter, aplikacja mobilna posiada jeden kod źródłowy, co ułatwia jej rozwój i utrzymanie.

2.1 Architektura Systemu

Projekt zrealizowano w modelu rozproszonym, gdzie centralnym punktem jest API, z którym komunikują się niezależne aplikacje klienckie (mobilna i webowa).

2.2 Stos Technologiczny

2.2.1 Backend (PUM_API)

Serwer aplikacji odpowiedzialny za logikę biznesową.

- **Język/Framework:** .NET 9.0 (C#) / ASP.NET Core Web API.
- **Baza Danych:** Microsoft SQL Server.
- **ORM:** Entity Framework Core (podejście Code First).
- **Dane Przestrzenne:** NetTopologySuite (obsługa typów geograficznych w bazie danych).
- **Dokumentacja:** Swagger / OpenAPI.
- **Autoryzacja:** Tokeny JWT (JSON Web Token) + Identity.

2.2.2 Aplikacja Mobilna (pum_project)

Aplikacja mobilna zrealizowana w technologii cross-platformowej.

- **Framework:** Flutter.
- **Język:** Dart (SDK: 3.9.2+).
- **Kluczowe biblioteki:**
  - **flutter_map:** Wyświetlanie map i tras (OpenStreetMap).
  - **geolocator:** Pobieranie współrzędnych GPS.
  - **flutter_foreground_task:** Obsługa procesów w tle (rejestracja trasy przy wygaszonym ekranie).

  - **http:** Komunikacja z API.
  - **provider:** Zarządzanie stanem aplikacji.
  - **flutter_secure_storage:** Bezpieczne przechowywanie danych logowania.
  - **sqflite:** Lokalna baza danych (SQLite).

2.2.3 Panel Administracyjny (PUM_ADMIN)

Aplikacja webowa typu SPA (Single Page Application).

- **Framework:** React.js.
- **Język:** JavaScript / TypeScript.
- **Zarządzanie stanem:** React Hooks.
- **Stylowanie:** CSS Modules / Bootstrap.
- **Komunikacja:** Axios.

Instrukcja uruchomienia i wdrożenia

3.1 Uruchomienie Lokalne (Development)

3.1.1 Krok 1: Baza Danych i Backend (API)

1. Wymagane zainstalowane: .NET SDK 9.0 oraz SQL Server. 2. Skonfiguruj `ConnectionStrings` w pliku `appsettings.json` do lokalnej bazy. 3. Wykonaj migrację i uruchom API:

```
dotnet ef database update
dotnet run
```

3.1.2 Krok 2: Panel Administracyjny

1. Zainstaluj zależności i uruchom środowisko:

```
npm install
npm run dev
```

3.1.3 Krok 3: Aplikacja Mobilna

1. Skonfiguruj adres API na lokalny (np. `10.0.2.2` dla emulatora). 2. Uruchom aplikację:

```
flutter pub get
flutter run
```

3.2 Wdrożenie Produkcyjne (Deployment)

System został pomyślnie wdrożony na serwery zewnętrzne i jest dostępny w sieci Internet. Infrastruktura została podzielona między dwóch dostawców usług hostingowych.

3.2.1 Backend (API)

Część serwerowa aplikacji (.NET 9 + SQL Server) została wdrożona na hostingu **Webio.pl**.

- **Adres API:** https://activitis.hostingasp.pl.hostingasp.pl
- **Dokumentacja techniczna (Swagger):** https://activitis.hostingasp.pl.hostingasp.pl/api/documentation/index.html

Wdrożenie przeprowadzono przy użyciu profilu publikacji (Publish Profile) bezpośrednio ze środowiska Visual Studio.

3.2.2 Panel Administracyjny

Warstwa wizualna dla administratora (Frontend React) została opublikowana na hostingu **SEOHost.pl**.

- **Adres panelu:** https://www.srv101211.seohost.com.pl/login

Aplikacja została zbudowana poleceniem `npm run build`, a wygenerowane pliki statyczne zostały przesłane na serwer za pomocą protokołu FTP.

3.2.3 Aplikacja Mobilna

Aplikacja mobilna została skonfigurowana do komunikacji ze zdalnym adresem API. Dzięki temu użytkownicy mogą korzystać z aplikacji na swoich telefonach, będąc w dowolnym miejscu z dostępem do Internetu, a dane są synchronizowane z centralną bazą na serwerze Webio.

Wnioski z pracy projektowej

4.1 Status realizacji wymagań

Poniższa tabela przedstawia stopień realizacji wymagań funkcjonalnych.

Nr	Wymaganie	Status
1	Pobieranie aplikacji na Android	Zrealizowano
2	Rejestracja i logowanie użytkownika	Zrealizowano
3	Resetowanie hasła	Zrealizowano
4	Edycja profilu (waga, wzrost, avatar)	Zrealizowano
5	Rozpoczynanie i kończenie aktywności	Zrealizowano
6	Rejestracja GPS (GeoJSON, NetTopologySuite)	Zrealizowano
7	Dodawanie szczegółów po treningu	Zrealizowano
8	Lista historii aktywności	Zrealizowano
9	Szczegóły aktywności (mapa, statystyki)	Zrealizowano
10	Filtrowanie i sortowanie aktywności	Zrealizowano
11	Statystyki użytkownika	Zrealizowano
12	Ranking użytkowników	Zrealizowano
13	Synchronizacja z serwerem	Zrealizowano
14	Logowanie Administratora (Web)	Zrealizowano
15	Przeglądanie listy użytkowników (Admin)	Zrealizowano
16	Filtrowanie i usuwanie aktywności (Admin)	Zrealizowano
17	REST API (OpenAPI 3.0)	Zrealizowano
18	Praca w tle (Foreground Service)	Zrealizowano
19	Eksport do CSV (Opcjonalne)	Do rozwoju
20	Funkcje AI (Opcjonalne)	Do rozwoju

4.2 Co udało się zrealizować

W ramach projektu udało się stworzyć w pełni funkcjonalny system, który został nie tylko zaprojektowany, ale również wdrożony w środowisku produkcyjnym. Kluczowym osiągnięciem jest:

- Działająca aplikacja mobilna we Flutterze, poprawnie rejestrująca trasy GPS na mapach OpenStreetMap.
- Stabilny backend w .NET 9 hostowany na Webio, obsługujący żądania od klientów.
- Dostępny publicznie panel administracyjny na SEOHost, umożliwiający zarządzanie systemem z dowolnej przeglądarki.

4.3 Napotkane trudności i obszary do poprawy

Największym wyzwaniem technicznym była obsługa usług działających w tle (`flutter_foreground_tas`
oraz konfiguracja hostingu IIS dla aplikacji ASP.NET Core (konfiguracja puli aplikacji i połączenia z bazą danych).

Możliwe kierunki rozwoju:

- Rozbudowa modułu społecznościowego (komentarze, polubienia aktywności).
- Implementacja powiadomień Push.
- Zastosowanie algorytmów AI do sugerowania planów treningowych na podstawie historii aktywności.