Politechnika
Wrocławska

Zaawansowane metody programowania
Opracowanie projektu

semestr zimowy 2025/2026

Autorzy: Kacper Krzesiński
Szymon Kluska

Spis treści

1 Streszczenie

Celem projektu jest stworzenie grywalizacyjnej aplikacji typu „zbierz je wszystkie", która zrzesza zdobywców Korony Gór Polski. System umożliwia ewidencjonowanie wizyt na szczytach poprzez skanowanie kodów QR, śledzenie postępów, odblokowywanie osiągnięć oraz udział w dyskusjach pod konkretnymi punktami (możliwy tylko dla zdobywców).
Podstawowe informacje:

- Klasa systemu: System typu klient-serwer z centralną bazą danych.
- Charakter systemu: Samodzielna aplikacja internetowa.
- Zastosowane wzorce i techniki: Architektura oparta na wzorcu MVC (Model-View-Controller).
- Języki implementacji: TypeScript, Python, HTML, CSS.

Środowiska i narzędzia:

- Frontend: Framework Vue wraz z biblioteką NuxtUI oraz TailwindCSS.
- Backend: Framework FastAPI.
- Baza danych: PostgreSQL.
- Format wymiany danych: JSON.
- Narzędzia: Visual Studio Code (IDE), GitHub (system kontroli wersji).

2 Wstępny opis słowny

Aplikacja ma na celu zrzeszanie i aktywizację osób zdobywających Koronę Gór Polski (KGP), w której skład wchodzą najwyższe szczyty z każdego pasma górskiego w Polsce. Użytkownik, po założeniu konta i zalogowaniu się, otrzymuje dostęp do bazy szczytów stanowiących punkty zainteresowania (POI). Głównym mechanizmem działania jest potwierdzanie fizycznej obecności na danym szczycie. Zakłada się, że na każdym z nich znajduje się tabliczka z unikalnym kodem QR. Zeskanowanie kodu przez użytkownika powoduje automatyczne zaliczenie szczytu do jego osobistej kolekcji. Zdarzenie to odblokowuje również dostęp do sekcji dyskusyjnej przypisanej do danej góry, która jest zastrzeżona wyłącznie dla osób, które faktycznie ją zdobyły.
W miarę postępów w wędrówkach system monitoruje stan posiadanej kolekcji i przyznaje użytkownikowi wirtualne osiągnięcia za realizację określonych celów. Wyniki te przekładają się na pozycję w publicznym rankingu, który pozwala na zdrową rywalizację z innymi pasjonatami górskich wędrówek.
Dostępne osiągnięcia do zdobycia:

- Zdobądź całą Koronę Gór Polski.
- Zdobądź wszystkie szczyty w Sudetach.
- Zdobądź wszystkie szczyty w Karpatach.
- Zdobądź wszystkie szczyty powyżej 1000m n. p. m.
- Zdobądź wszystkie szczyty poniżej 1000m n. p. m.

- Zdobądź najwyższy szczyt (Rysy).
- Zdobądź najniższy szczyt (Łysica).

3 Słownik pojęć z dziedziny problemu

- Użytkownik - osoba zarejestrowana w systemie, posiadająca unikalny profil, która gromadzi kolekcję zdobytych szczytów i osiągnięć.
- Korona Gór Polski (KGP) - lista 28 najwyższych szczytów poszczególnych pasm górskich Polski.
- Szczyt (POI) – konkretny punkt geograficzny wchodzący w skład Korony Gór Polski, posiadający swoją nazwę, opis oraz przypisaną grafikę.
- Kolekcja – zestawienie wszystkich szczytów zdobytych przez danego użytkownika, prezentujące jego postępy w formie wizualnej.
- Kod QR – fizyczny nośnik danych umieszczony na szczycie, służący jako dowód obecności i klucz do odblokowania zawartości w aplikacji.
- Osiągnięcie – wirtualna nagroda przyznawana za spełnienie określonych kryteriów.
- Dyskusja (Sekcja komentarzy) – forum wymiany opinii i informacji pod opisem szczytu, do którego dostęp mają wyłącznie użytkownicy, którzy wcześniej zeskanowali kod QR na danym szczycie.
- Komentarz - treść tekstowa dodawana przez użytkownika pod konkretnym szczytem, dostępna po jego zdobyciu
- Ranking – zestawienie użytkowników uszeregowane według liczby zdobytych osiągnięć, pozwalające na porównanie postępów w skali całej społeczności.

4 Analiza wymagań użytkownika

4.1 Aktorzy systemu

- Użytkownik (Zdobywca): Główny aktor, który rejestruje się w systemie, zdobywa szczyty, zbiera osiągnięcia, bierze udział w dyskusji.
- System (Backend): Odpowiada za weryfikację kodów QR, automatyczne przyznawanie osiągnięć oraz zarządzanie uprawnieniami do dyskusji.

4.2 Główny scenariusz: Zdobycie szczytu i odblokowanie zawartości

To najważniejsza sekwencja zdarzeń, która realizuje model gry "zbierz je wszystkie".

1. Dotarcie do celu: Użytkownik fizycznie zdobywa szczyt górski należący do Korony Gór Polski.
2. Skanowanie: Użytkownik odnajduje tabliczkę z kodem QR i skanuje go przy użyciu aplikacji.
3. Weryfikacja: System przesyła identyfikator szczytu do punktu końcowego.

4. Aktualizacja kolekcji: System dopisuje szczyt do listy zdobytych punktów danego użytkownika.

5. Nagroda natychmiastowa: Użytkownik otrzymuje dostęp do sekcji dyskusyjnej pod danym szczytem, gdzie może wymienić się uwagami z innymi zdobywcami.

6. Sprawdzenie postępu: System automatycznie weryfikuje, czy nowo zdobyty punkt nie powoduje odblokowania przez użytkownika nowego osiągnięcia (np. zdobycie wszystkich szczytów w Sudetach).

4.3 Scenariusze poboczne

Scenariusz: Monitorowanie postępów innych użytkowników.

- Akcja: Użytkownik otwiera widok "Ranking".
- Reakcja systemu: Serwer wykonuje zapytanie do bazy danych, celem pobrania aktualnej liczby osiągnięć każdego z użytkowników.
- Wynik: Użytkownik widzi swoją pozycję na tle innych.

Scenariusz: Wyświetlenie kolekcji zdobytych szczytów

- Akcja: Użytkownik wybiera z menu głównego opcję "Kolekcja".
- Reakcja systemu: Serwer wykonuje zapytanie do bazy danych, celem pobrania aktualnej liczby zdobytych szczytów przez tego użytkownika.
- Wynik: Użytkownik widzi zdobyte przez siebie szczyty należące do KGP.

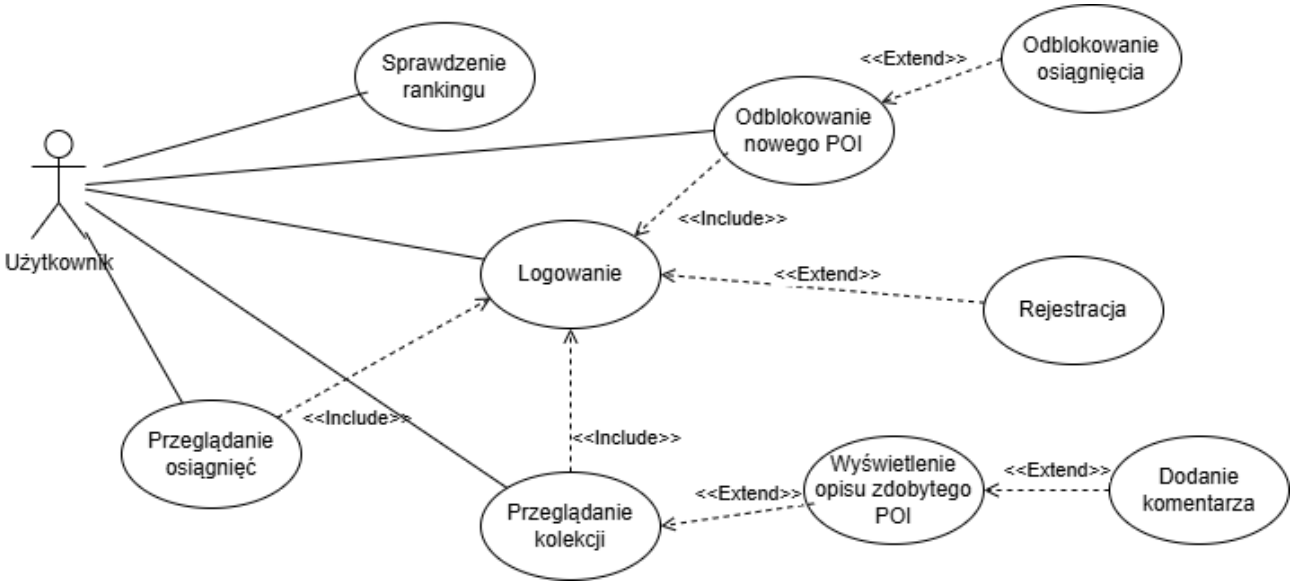
Scenariusz: Wyświetlenie zdobytych osiągnięć

- Akcja: Użytkownik wybiera z menu głównego opcję "Osiągnięcia".
- Reakcja systemu: Serwer wykonuje zapytanie do bazy danych, celem pobrania aktualnej liczby odblokowanych osiągnięć przez tego użytkownika.
- Wynik: Użytkownik widzi odblokowane przez siebie osiągnięcia oraz te, których jeszcze nie odblokował.

Scenariusz: Napisanie komentarza w dyskusji

- Warunek wstępny: Użytkownik musiał wcześniej zeskanować kod QR na danym szczycie.
- Akcja: Użytkownik wchodzi w opis konkretnego szczytu, wpisuje wiadomość w polu tekstowym i klika przycisk "Dodaj komentarz".
- Reakcja systemu: Serwer dodaje nowy komentarz do bazy danych.
- Wynik: Komentarz staje się widoczny dla innych zdobywców tego szczytu, umożliwiając tworzenie wątków dyskusyjnych.

4.4 Diagram przypadków użycia

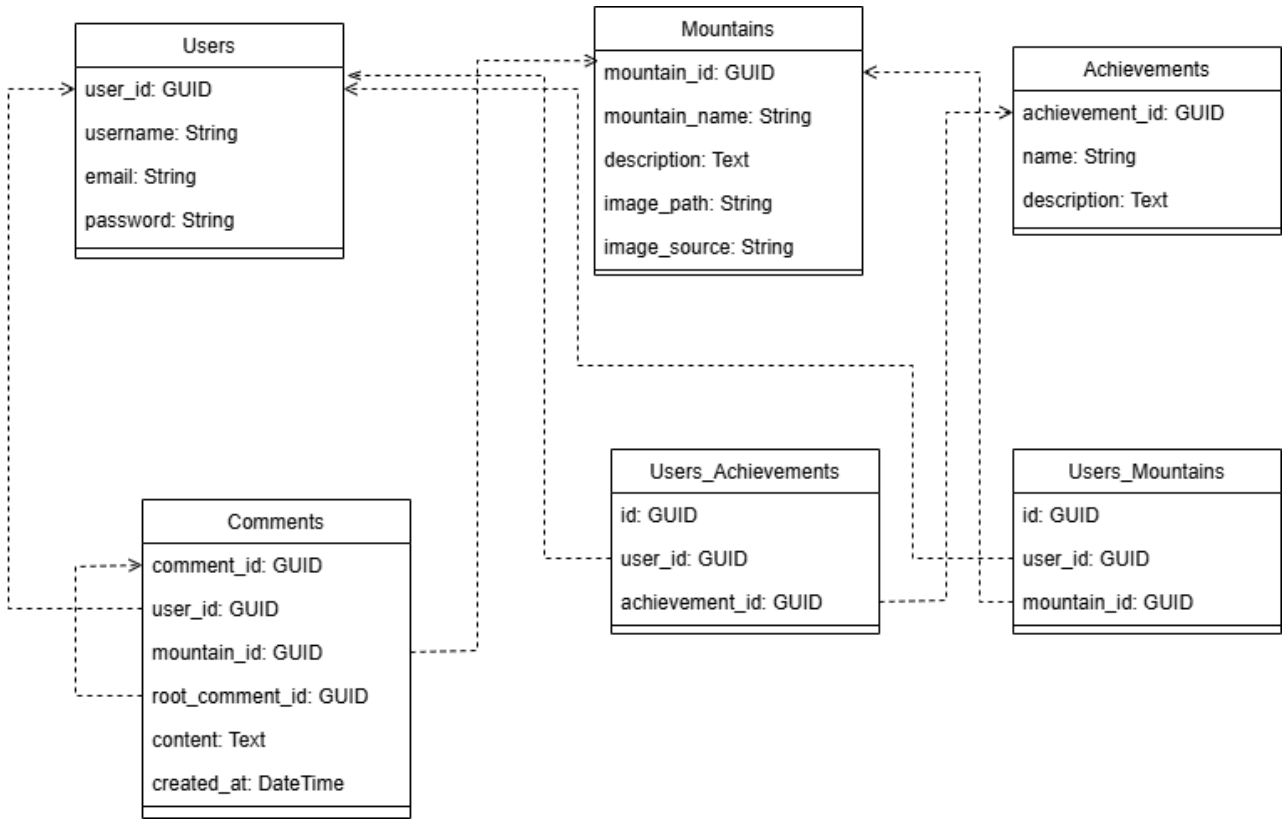


Rysunek 1: Diagram przypadków użycia

Na diagramie 1 przedstawiono przypadki użycia dla zaprojektowanego systemu. Zdefiniowano jednego aktora - użytkownika. Użytkownik niezalogowany ma możliwość przeglądania rankingu, pozostałe akcje wymagają zalogowania się do systemu (poprzedzonego ewentualnie rejestracją). Po zalogowaniu się użytkownik może przeglądać swoją kolekcję zdobytych szczytów. Stamtąd może przenieść się do opisu wybranego szczytu oraz dodać pod nim komentarz. Inną akcją możliwą po zalogowaniu jest odblokowanie nowego POI poprzez zeskanowanie kodu QR, a to z kolei może spowodować zdobycie przez użytkownika nowego osiągnięcia.

5 Modele systemu z różnych perspektyw

5.1 Diagram klas



Rysunek 2: Diagram klas

Na rysunku 2 przedstawiono diagram klas zaprojektowanego systemu.
Opis klas:

1. Klasa **Users** - reprezentuje użytkownika systemu. Zawiera takie pola jak id, nazwę użytkownika, hasło oraz adres email.

2. Klasa **Mountains** - reprezentuje punkt POI, czyli szczyt Należący do Korony Gór Polski, który jest możliwy do odblokowania przez użytkownika. Klasa ta zawiera pola przechowujące id szczytu, jego nazwę, opis, ścieżkę do grafiki oraz źródło grafiki.

3. Klasa **Achievements** - reprezentuje osiągnięcie, jakie może odblokować użytkownik. Zawiera id osiągnięcia, jego nazwę oraz opis.

4. Klasa **Comments** - reprezentuje komentarz, jaki może napisać użytkownik pod sekcją dyskusyjną wybranego POI. Klasa jest połączona relacją zależności z klasą Users, gdyż to użytkownicy piszą komentarze, oraz z klasą Mountains, żrby można było określić, w sekcji dyskusyjnej pod jakim szczytem użytkownik ten komentarz napisał. Dodatkowo klasa jest połączona relacją zależności sama ze sobą, ponieważ możliwe jest dodawanie "komentarza do komentarza", celem tworzenia wątków dyskusyjnych. Klasa zawiera takie pola jak id komentarza, id użytkownika, id szczytu, id komentarza nadrzędnego, zawartość komentarza oraz datę utworzenia.

5. Klasa **Users_Achievements** - reprezentuje osiągnięcie zdobyte przez konkretnego użytkownika. Klasa jest połączona relacją zależności z klasami Users oraz Achievements i zawiera takie pola jak id, id użytkownika oraz id osiągnięcia.

6. Klasa **Users_Mountains** - reprezentuje szczyt zdobyty przez konkretnego użytkownika. Klasa jest połączona relacją zależności z klasami Users oraz Mountains i zawiera takie pola jak id, id użytkownika oraz id szczytu.

5.2 Diagram fizyczny bazy danych

Aplikacja wymaga wykorzystania bazy danych w celu przechowywania informacji o użytkownikach, ich osiągnięciach, zdobytych szczytach oraz napisanych komentarzach. Opisane w poprzednim podrozdziale klasy, znalazły swoje odzwierciedlenie w strukturze bazy danych. Opis tabel bazy danych:

1. **users** - przechowuje podstawowe dane o profilach użytkowników, w tym unikalny identyfikator user_id (typ uuid), nazwę użytkownika, adres e-mail oraz zaszyfrowane hasło.

2. **mountains** - zawiera szczegółowy wykaz szczytów Korony Gór Polski, przechowując ich identyfikator mountain_id, nazwę, opis, kategorię oraz dane o plikach graficznych wraz z ich źródłem.

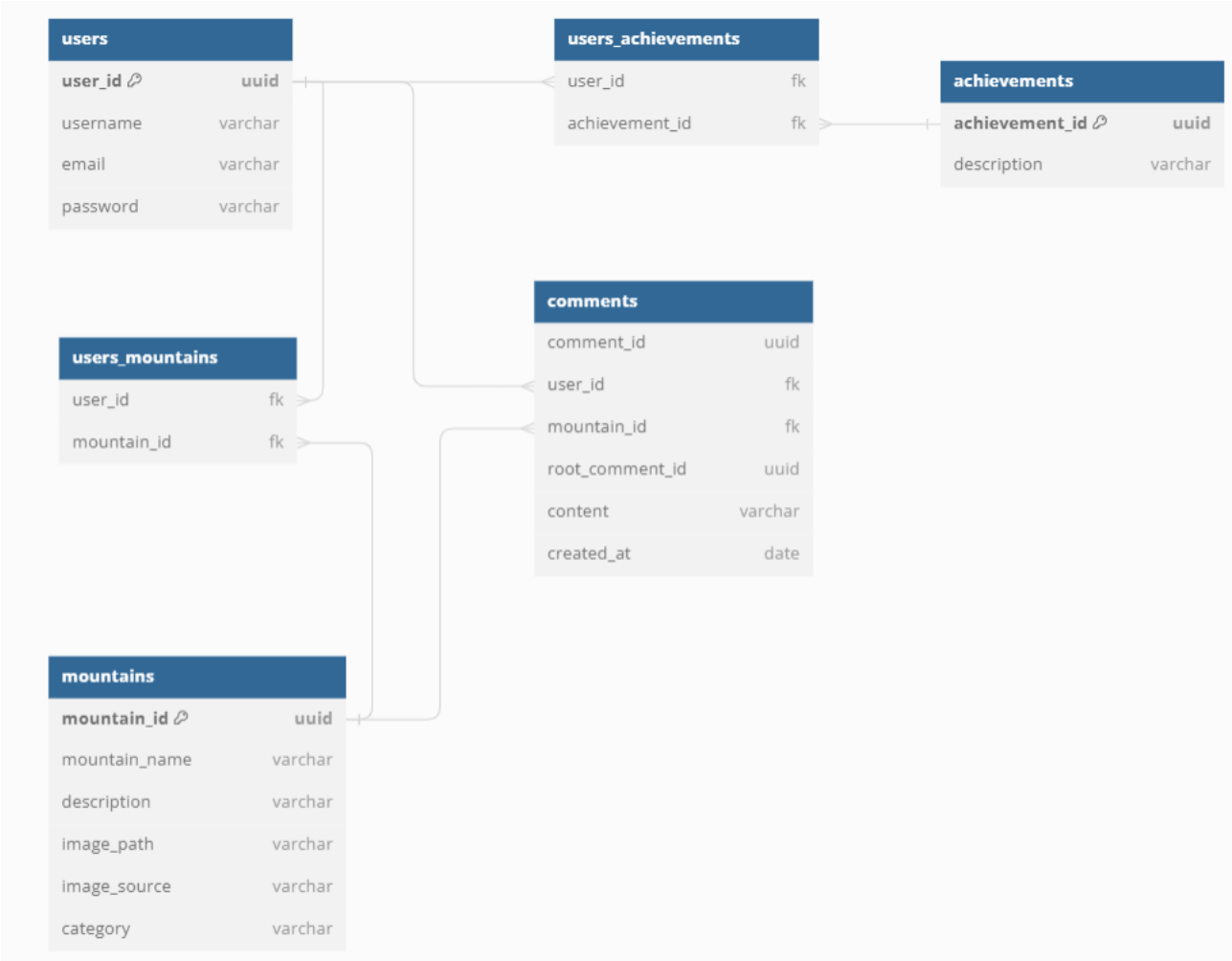
3. **users_mountains** - pełni rolę tabeli łączącej (asocjacyjnej) między użytkownikami a szczytami, rejestrując fakt zdobycia konkretnej góry poprzez powiązanie kluczy obcych user_id oraz mountain_id.

4. **achievements** - definiuje możliwe do zdobycia nagrody wirtualne, przechowując ich unikalne identyfikatory achievement_id oraz opisy określające kryteria ich przyznania.

5. **users_achievements** - odpowiada za przypisywanie odblokowanych osiągnięć do konkretnych kont użytkowników, wykorzystując powiązania kluczy obcych user_id i achievement_id.

6. **comments** - zarządza treściami generowanymi przez użytkowników w sekcjach dyskusyjnych, przechowując treść wpisu, datę utworzenia, powiązanie z autorem i szczytem oraz identyfikator root_comment_id do obsługi dodawania odpowiedzi do już istniejących komentarzy.

Tabele wraz z relacjami między nimi przedstawiono na rysunku 3.



Rysunek 3: Diagram fizyczny bazy danych

6 Kwestie implementacyjne

6.1 Wybrane języki i technologie

- Frontend: Wykorzystano język TypeScript wraz z frameworkiem Nuxt (Vue). Warstwa wizualna budowana jest za pomocą TailwindCSS oraz biblioteki komponentów NuxtUI, co pozwala na szybkie i spójne projektowanie interfejsu.
- Backend: Serwer został zaimplementowany w języku Python przy użyciu frameworka FastAPI, który charakteryzuje się wysoką wydajnością i asynchronicznością.
- Baza danych: Wybrano system PostgreSQL, który zapewnia trwałość i relacyjność danych niezbędną do obsługi użytkowników, szczytów i osiągnięć.

6.2 Organizacja projektu i architektura

- Wzorzec MVC: Aplikacja realizuje wzorzec Model-View-Controller, dzieląc system na warstwę logiki (Model), prezentacji (Widok) oraz zarządzania przepływem danych (Kontroler).
- Podział repozytoriów: Projekt jest podzielony na dwa oddzielne repozytoria na platformie GitHub: jedno dla strony klienckiej (frontend) oraz drugie dla serwera (backend). Pozwala

to na niezależny rozwój i wdrażanie obu części systemu.

- Środowisko pracy: Głównym narzędziem programistycznym jest Visual Studio Code.

6.3 Narzędzia ułatwiające pracę i automatyzacja

- Automatyczna dokumentacja API: Dzięki wykorzystaniu FastAPI, system generuje interaktywną dokumentację punktów końcowych, co znacząco ułatwia testowanie zapytań bez konieczności używania zewnętrznych narzędzi.
- Serializacja danych: Do wymiany informacji między klientem a serwerem wybrano format JSON. Jest on obsługiwany natywnie przez większość technologii, co automatyzuje proces mapowania obiektów na tekst i odwrotnie.
- Biblioteki komponentów: Użycie NuxtUI eliminuje potrzebę ręcznego pisania każdego elementu interfejsu (przyciski, pola tekstowe), co przyspiesza prace nad warstwą wizualną.

7 Podsumowanie i dyskusja krytyczna

7.1 Cele i charakterystyka realizowanego systemu

Głównym założeniem projektu było stworzenie platformy integrującej społeczność zdobywców Korony Gór Polski poprzez mechanizmy grywalizacji typu „zbierz je wszystkie". System skutecznie łączy funkcje aplikacji mobilnej z elementami społecznościowymi, pozwalając na dokumentowanie aktywności turystycznej w czasie rzeczywistym. Unikalność rozwiązania opiera się na wykorzystaniu fizycznych znaczników w postaci kodów QR umieszczonych na szczytach, co stanowi nie tylko potwierdzenie obecności użytkownika, ale również klucz do odblokowania interaktywnych funkcji systemu, takich jak fora dyskusyjne przypisane do konkretnych gór.

7.2 Architektura i wybory technologiczne

Fundamentem strukturalnym aplikacji stał się wzorzec MVC (Model-View-Controller), który zapewnił klarowny podział na logikę biznesową, interfejs użytkownika oraz mechanizmy kontrolne. Dzięki takiemu podejściu poszczególne części systemu są ze sobą połączone w celu sprawnej wymiany danych, pozostając jednocześnie niezależnymi modułami. Zastosowanie nowoczesnych frameworków, takich jak FastAPI po stronie serwera oraz Nuxt (Vue) po stronie klienta, pozwoliło na uzyskanie wysokiej wydajności i responsywności aplikacji, a trwałe przechowywanie danych w relacyjnej bazie PostgreSQL umożliwiło sprawne generowanie rankingu użytkowników opartego na liczbie zdobytych osiągnięć oraz obsługę systemu komentarzy.

7.3 Innowacyjne mechanizmy interakcji i grywalizacji

Szczególną wartość projektową wnosi system warunkowego dostępu do sekcji komentarzy pod konkretnymi szczytami. Mechanizm ten wymusza realną aktywność w terenie – dopiero po zeskanowaniu kodu QR na danej górze system dopisuje szczyt do kolekcji użytkownika i nadaje mu uprawnienia do publikowania wpisów w dedykowanej dyskusji. Takie podejście promuje autentyczność wymiany informacji i buduje elitarną społeczność zdobywców. Dodatkowym elementem motywacyjnym jest rozbudowany system osiągnięć, który nagradza użytkowników nie tylko za pojedyncze wejścia, ale za realizację celów zbiorczych, takich jak zdobycie wszystkich szczytów w Sudetach czy Karpatach.

7.4 Wnioski

Podsumowując, zastosowany stos technologiczny oraz wybrane narzędzia programistyczne pozwoliły na stworzenie stabilnego i skalowalnego systemu, który w sposób atrakcyjny wizualnie i funkcjonalnie realizuje potrzeby nowoczesnego turysty górskiego. Mimo pomyślnej implementacji, analiza systemu pozwala wskazać obszary, które można było rozwiązać w sposób bardziej zaawansowany. Obecny mechanizm weryfikacji zdobycia szczytu opiera się wyłącznie na zeskanowaniu kodu QR. W warunkach rzeczywistych jest to rozwiązanie podatne na nadużycia – użytkownik mógłby przesłać zdjęcie kodu innej osobie lub znaleźć je w internecie. Bardziej wiarygodnym podejściem byłoby wprowadzenie walidacji geograficznej, która sprawdzałaby współrzędne GPS urządzenia w momencie skanowania kodu. Dodatkowo, struktura bazy danych mogłaby zostać rozszerzona o historię wizyt. Wprowadzenie daty i godziny każdego skanowania pozwoliłoby na generowanie ciekawszych statystyk, takich jak najszybsze zdobycie korony czy powtarzalność wejść.
Projekt posiada duży potencjał rozbudowy, który mógłby znacząco podnieść zaangażowanie społeczności. Jednym z naturalnych kierunków jest integracja z zewnętrznymi mapami (np. Google Maps), co pozwoliłoby na wizualizację postępów użytkownika na interaktywnej mapie Polski bezpośrednio w widoku kolekcji. Rozszerzenie bazy szczytów o inne pasma górskie lub lokalne odznaki turystyczne mogłoby przekształcić aplikację w kompleksowy dziennik wypraw górskich dla szerokiego grona turystów.

8 Wykaz materiałów źródłowych

1. Dokumentacja języka TypeScript: `https://www.typescriptlang.org/docs/`
2. Dokumentacja Nuxt: `https://nuxt.com/docs/4.x/getting-started/introduction`
3. Dokumentacja vue.js: `https://vuejs.org/guide/introduction.html`
4. Biblioteka komponentów NuxtUI: `https://ui.nuxt.com/`
5. Dokumentacja narzędzia TailwindCSS: `https://tailwindcss.com/docs/installation/using-vite`
6. Dokumentacja frameworka FastAPI: `https://fastapi.tiangolo.com/`
7. Dokumentacja PostgreSQL: `https://www.postgresql.org/docs/`