Programowanie aplikacyjne

-

specyfikacja wymagań projektu

Mikołaj Rożek, Marcin Lisowski, Sofiya Nasiakaila, Maksymilian Bilski

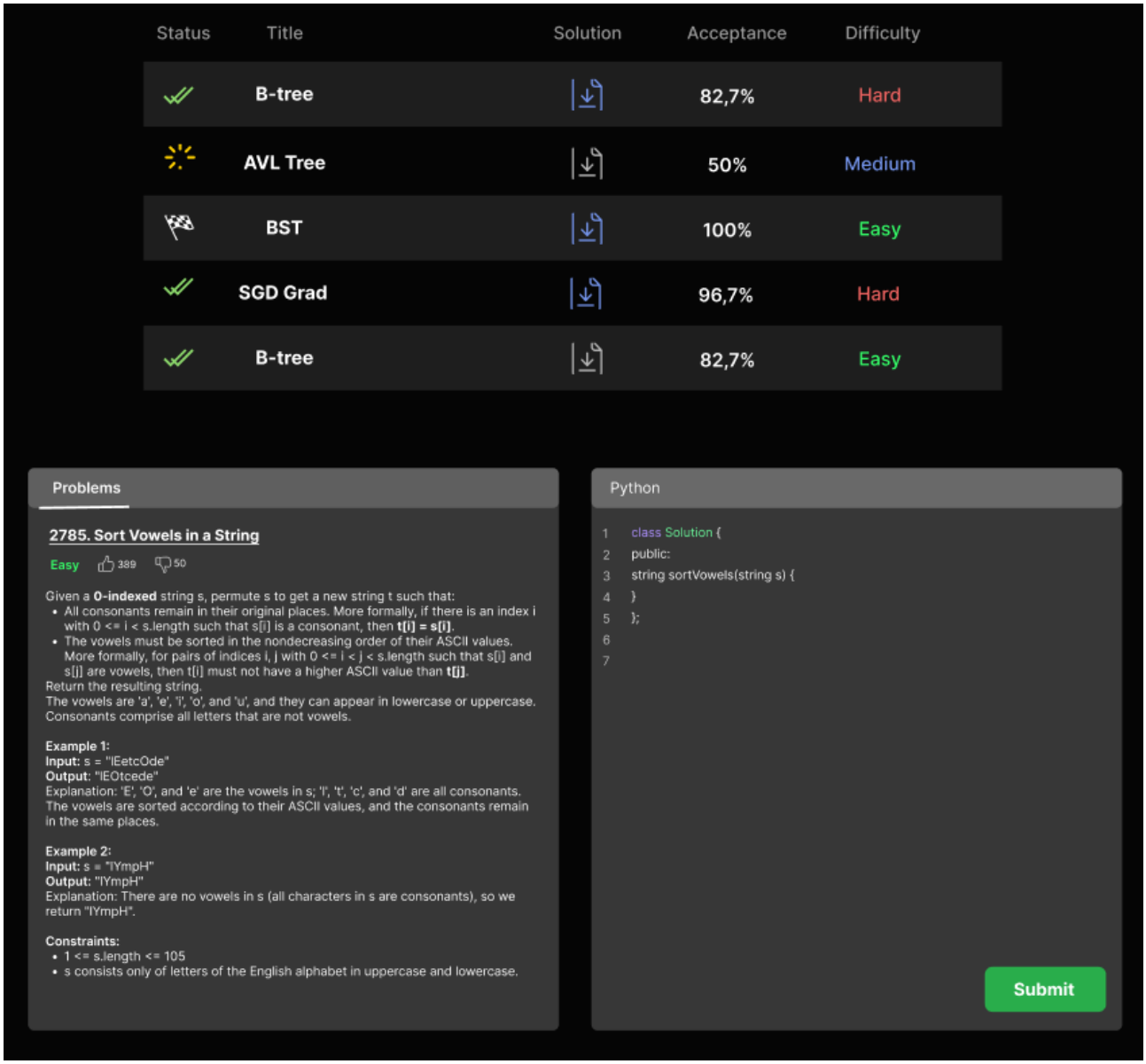
Spis treści

1. Wprowadzenie . . . 2
1.1. Cel projektu . . . 2
2. Tabela z podziałem wymagań funkcjonalnych . . . 3
3. Wymagania funkcjonalne . . . 3
3.1. Rozwiązywanie zadań programistycznych . . . 3
3.2. Rejestracja i logowanie . . . 3
3.3. Przeglądanie zadań . . . 3
3.4. Edytor kodu . . . 3
3.5. Sprawdzanie poprawności zadań . . . 3
3.6. Porównywanie jakości rozwiązania z innymi użytkownikami . . . 3
3.7. Profil użytkownika i społeczność . . . 3
3.8. Statystyki . . . 4
3.9. Komentarze pod zadaniami . . . 4
3.10. System powiadomień . . . 4
3.11. System ocen zadań . . . 4
3.12. Support . . . 4
4. Wymagania niefunkcjonalne . . . 4
4.1. Wydajność . . . 4
4.2. Bezpieczeństwo . . . 4
4.3. Kompatybilność . . . 4
4.4. Skalowalność . . . 4
4.5. Wsparcie Języków Programowania . . . 4
4.6. Archiwizacja Danych . . . 4
5. Wymagania techniczne . . . 5

1. Wprowadzenie

1.1. Cel projektu

Celem projektu jest stworzenie aplikacji webowej, która umożliwia użytkownikom rozwiązywanie zadań programistycznych, głównie algorytmicznych. Użytkownik ma mieć możliwość sprawdzania jakości przesłanych rozwiązań na tle innych użytkowników, śledzenia swojej aktywności, zyskiwania punktów za rozwiązanie zadania oraz brania udziału w turniejach programistycznych.



Rysunek 1. Mankieta poglądowa

Osoba	Funkcjonalności
Wspólne	Rozwiązywanie zadań programistycznych
Mikołaj Rożek	Przeglądanie zadań, profil użytkownika, statystyki
Sofiya Nasiakaila	Ranking użytkowników, edytor kodu, system ocen zadań
Marcin Lisowski	Rejestracja i logowanie, porównywanie rozwiązań, System powiadomień
Maksymilian Bilski	Sprawdzanie poprawności zadań, komentarze pod zadaniami, support

Tabela 1. Tabela wymagań funkcjonalnych

2. Tabela z podziałem wymagań funkcjonalnych

3. Wymagania funkcjonalne

3.1. Rozwiązywanie zadań programistycznych

Użytkownik ma możliwość rozwiązywania zadań o wybranym poziomie trudności i o specyficznej tematyce.

3.2. Rejestracja i logowanie

Użytkownik może utworzyć konto, podając unikalny adres e-mail oraz hasło. Logowanie przebiega w analogiczny sposób. Rejestracja nastepuje poprzez formularz, gdzie nowi użytkownicy wprowadzają unikalny adres e-mail i hasło, a następnie dane są weryfikowane i zapisywane w bazie danych. Proces logowania obejmuje wprowadzenie danych uwierzytelniających przez użytkownika, a system sprawdza ich poprawność w bazie danych, udzielając dostępu do funkcji aplikacji w przypadku pozytywnej weryfikacji.

3.3. Przeglądanie zadań

Zadania są podzielone na kategorie, takie jak algorytmy, struktury danych, bazy danych. Użytkownik może przeglądać zadania w ramach wybranej kategorii oraz sortować je według poziomu trudności, popularności czy daty dodania. Filtry pozwalają zawęzić widok zadań na podstawie struktur danych, tagów.

3.4. Edytor kodu

Użytkownik ma dostęp do edytora kodu online, który obsługuje różne języki programowania. Uwzględniona będzie funkcjonalność podpowiedzi kodu i kolorowania składni.

3.5. Sprawdzanie poprawności zadań

Po wysłaniu rozwiązania zadania, jego poprawność będzie sprawdzana na podstawie gotowych testów jednostkowych. Użytkownik otrzyma wiadomość zwrotną dotyczącą poprawności swojego rozwiązania, razem z czasem jego wykonania.

3.6. Porównywanie jakości rozwiązania z innymi użytkownikami

Razem z wiadomością zwrotną dotyczącą poprawności rozwiązania, użytkownik zobaczy szybkość swojego kodu, oraz jak plasuje się jego kod względem innych użytkowników.

3.7. Profil użytkownika i społeczność

Użytkownik ma możliwość wprowadzenia swoich danych, ustawienie ich widoczności dla innych użytkowników. Możliwe będzie przeglądanie również profili pozostałych użytkowników.

3.8. Statystyki

Użytkownik w profilu ma dostęp do statystyk swojego postępu w rozwiązywaniu zadań. W ramach tego może zobaczyć ile zadań na danym poziomie trudności rozwiązał, czasu spędzonego na platformie, procent poprawnie rozwiązanych zadań.

3.9. Komentarze pod zadaniami

Możliwe będzie pisanie widocznych dla innych użytkowników komentarzy pod każdym zadaniem, np. w celu podania swojego rozwiązania, bądź skomentowania cudzego, jak również like'owanie innych komentarzy.

3.10. System powiadomień

Informowanie użytkowników o nowych zadaniach za pośrednictwem adresu mailowego o nowych zadaniach, czy odpowiedziach na komentarze.

3.11. System ocen zadań

Zadania dostępne do rozwiązania podlegają ocenie w skali 1-5 - czy przydzielony poziomu trudności jest adekwatny, czy zadany problem jest poprawnie wytłumaczony, oraz czy zadania są interesujące.

3.12. Support

Stworzenie funkcji wsparcia dla platformy obejmuje integrację czatu online dla użytkowników do zgłaszania problemów oraz udostępnienie kompleksowego interfejsu dla agentów wsparcia do reagowania na pytania. System uwzględniałby możliwość skutecznej komunikacji i rozwiązania problemów użytkowników.

4. Wymagania niefunkcjonalne

4.1. Wydajność

Czas ładowania strony i działania funkcji powinien być akceptowalny, nawet przy dużej liczbie użytkowników.

4.2. Bezpieczeństwo

Bezpieczne przechowywanie haseł użytkowników. Zabezpieczenie przed atakami typu SQL injection i XSS.

4.3. Kompatybilność

Aplikacja powinna być dostępna na różnych przeglądarkach (Chrome, Firefox, Safari).

4.4. Skalowalność

System powinien być łatwo skalowalny, aby obsłużyć wzrost liczby użytkowników.

4.5. Wsparcie Języków Programowania

Aplikacja powinna obsługiwać co najmniej kilku popularnych języków programowania, takich jak Java, Python, JavaScript.

4.6. Archiwizacja Danych

Tworzenie kopii zapasowych danych użytkowników w celu zapewnienia bezpieczeństwa.

5. Wymagania techniczne

1. Baza danych - PostgreSQL - będziemy korzystać z pgadmin4
2. Serwer WEB - Node.js
3. Obsługa żądań serwera oraz routingu - Express.js
4. UI - Angular.js