Dokumentacja wymagań dla projektu Sudoku-sweeper

Projekt na Pracownię Programowania Zespołowego

Agnieszka Głowacka, Martyna Trębacz, Oliwia Skucha,

Jakub Rogoża, Krzysztof Emerling, Szymon Duda

Spis treści

Dokumentacja wymagań dla projektu Sudoku-sweeper 1

1. Wstęp 2
2. Opis działania aplikacji 2
3. Wymagania funkcjonalne 3
   - 3.1 Gra w Sudoku 3
   - 3.3 Baza danych i zarządzanie wynikami 3
   - 3.4 Panel admina 3
4. Wymagania niefunkcjonalne 4
5. Wymagania bezpieczeństwa 4
6. Technologie 4
7. Struktura aplikacji 5
8. Bezpieczeństwo 6
9. Harmonogram 6

1. Wstęp

Projekt zakłada współpracę w 6-osobowej grupie w celu stworzenia aplikacji webowej umożliwiającej użytkownikom grę w Sudoku. Strona będzie oferować możliwość gry jako gość albo zarejestrowany użytkownik. Logowanie umożliwia zapisywanie wyników w bazie danych. Zalogowani użytkownicy będą posiadać dostęp do historii rozegranych gier. Aplikacja zostanie zbudowana w technologii Flask oraz SQL Alchemy.

2. Opis działania aplikacji

1. **Strona główna** – zawiera link do gry oraz linki do rejestracji i logowania.
2. **Rejestracja / logowanie** – użytkownicy mogą utworzyć konto i zalogować się, aby śledzić swoje wyniki.
3. **Gra w Sudoku** – użytkownik wybiera poziom trudności, ładuje planszę i rozpoczyna grę.
4. **Zapisywanie wyników** – dla zalogowanego urzytkownika, po zakończeniu gry wynik zostaje automatycznie zapisany w bazie danych.
5. **Historia gier** – użytkownicy mogą zobaczyć swoje poprzednie gry i wyniki.
6. **Panel administratora** – administrator może przeglądać i usuwać konta użytkowników.

3. Wymagania funkcjonalne

3.1 Gra w Sudoku

- Wczytywanie planszy Sudoku o różnych poziomach trudności z pobranej bazy danych (**https://github.com/grantm/sudoku-exchange-puzzle-bank**).
- Możliwość wpisywania liczb przez użytkownika.
- Opcjonalna możliwość nawigacji samą klawiaturą.
- Sprawdzanie poprawności rozwiązania.
- Funkcja cofania oraz resetowania planszy.
- Pomiar czasu gry.
- Formularz z historią poprzednich gier.

3.2 Rejestracja i logowanie użytkowników

- Formularz rejestracyjny z podstawową walidacją.
- Logowanie użytkowników (z wykorzystaniem hashowania i solenia haseł).
- Logowanie administratora.
- Możliwość wylogowania.

3.3 Baza danych i zarządzanie wynikami

- Przechowywanie kont użytkowników.
- Zapisywanie ukończonych gier (czas rozwiązania, poziom trudności).
- Wyświetlanie historii gier użytkownika.
- Wyświetlanie bazy użytkowników dla administratora.

3.4 Panel admina

- Lista użytkowników.
- Możliwość usuwania kont.

4. Wymagania niefunkcjonalne

- Intuicyjny interfejs użytkownika.
- Szybkie ładowanie strony mniej niż 1.5 sec.

5. Wymagania bezpieczeństwa

- Odporność na ataki SQL Injection.
- Solenie hasła przechowywane jako hashe.
- Walidacja formularzy wejściowych.
- Wymóg stosowania silnych haseł przez użytkowników.

6. Technologie

- Backend: Python, Flask
- Frontend: HTML, CSS, JavaScript, Bootstrap
- Baza danych: SQLAlchemy

7. Struktura aplikacji

/app

/static

- style.css

- script.js

/templates

- index.html

- login.html

- register.html

- game.html

- history.html

- admin.html

/vendor

- /sudoku-exchange-puzzle-bank

models.py

routes.py

app.py

database.db

requirements.txt

8. Bezpieczeństwo

1. Odporność na ataki SQL injection.
2. Hasła przechowywane w bazie danych jako hashe.
3. Solenie haseł.
4. Walidacja formularzy wejściowych, wymaganie długich, bezpiecznych haseł.

9. Harmonogram

Harmonogram dostępny na stronie:
https://github.com/users/KrzysztofEmerling/projects/2