**Aplikacja do nauki języka angielskiego - EnglishCards**

1. **Specyfikacja**
   1. Wymagania funkcjonalne:
      1. Rejestracja
      2. Logowanie
      3. Dodanie Słówka do bazy danych
      4. Weryfikacja słówka dodanego przez użytkownika przez administratora systemu
      5. Sprawdzenie znaczenia słówka w systemie(?)
      6. Nauka słówek z uwzględnieniem podziały na poziomy językowe
      7. Dodanie słówka do ulubionych w celu powtórzenia go w dowolnym momencie
      8. Funkcjonalny panel administratora do zarządzania bazą danych
      9. Wykonanie quizu przez użytkownika
      10. Wyświetlenie historycznych quizów użytkownika (?)
   2. Wymagania niefunkcjonalne
      1. Dostępność 24/h – po wdrożeniu
      2. Poprawność leksykalna – weryfikacja dodawanych słówek
      3. Weryfikacja progresu (?)
      4. Łatwość użycia – jednoznaczny interfejs użytkownika
   3. Krótki opis projektu – Aplikacja przeznaczona jest do pomocy w nauce języka obcego (początkowo język Angielski – możliwość dostosowania nowych języków). W aplikacji wykorzystywane są tak zwane „fiszki”, czyli słówka z opisem w obu językach. Aplikacja ma możliwość założenia prywatnego konta, które ułatwia korzystanie z niej w celu wyświetlania ulubionych słówek i dodania do nich niepewnych fiszek w celu dalszego ich powtórzenia oraz utrwalenia. Aplikacja zawiera podział na poziomy zaawansowania, dlatego każdy może sprawdzać się na wielu etapach.
   4. Potencjalni odbiorcy:
      1. Prywatni użytkownicy – każdy ma możliwość założenia konta i korzystania z aplikacji w wybranym przez siebie czasie
      2. Szkoły językowe – po wdrożeniu aplikacji oraz ewentualnym stworzenie źródeł dochodu (subskrypcja miesięczna, konta premium, reklamy) szkoły językowe miałyby możliwość nawiązania współpracy
   5. Potencjalne korzyści biznesowe:
      1. Korzyści finansowe – konta premium, subskrypcja miesięczna
      2. Współpraca ze szkołami językowymi w celu promocji aplikacji
2. **Dokumentacja techniczna**
   1. Stos technologiczny
      1. Relacyjna baza danych – SQL Server
      2. Backend – Django
      3. Frontend – HTML, CSS, Bootstrap, JavaScript
   2. Diagram Klas – dołączony do projektu
3. **Wykorzystanie wzorca MVC lub innego, alternatywnego**
   1. Separacja warstw w Django
      1. Modele/Baza danych – Django Models: W Django wykorzystywany jest ORM (Object–relational mapping), dzięki któremu tworząc modele w języku Python, mapujemy je przy pomocy migracji na relacyjną bazę danych, która odzwierciedla strukturę zmapowanego obiektu.

Przykładowa klasa modelowa w projekcieObraz zawierający tekst

Opis wygenerowany automatycznie

1. API/logika – Django Views: Widoki w Django odpowiadają za logikę działania aplikacji. Do widoku zostaję wysłane zapytanie (request) HTTP, po czym w wykonaniu zaimplementowanych czynności programistycznych zostaję zwrócona odpowiedź HTTP (respone), która wywołuję odpowiedni interfejs użytkownikowi pod postacią Django Template z przekazanymi parametrami z bazy danych.

Przykładowy widok dodający słówko do bazy danych z zachowaniem statusu odpowiedniego do osoby dodającej:  
Obraz zawierający tekst

Opis wygenerowany automatycznie

1. Wygląd – Django Templates: Za szatę graficzną w Django odpowiadają szablony, które są tworzone przy użyciu języka HTML oraz renderowane przez logikę w widokach.

Przykładowy szablon:  
Obraz zawierający tekst

Opis wygenerowany automatycznie

1. Wykorzystanie bazy danych

W Django dzięki wykorzystaniu ORM modele tworzone w języku Python są mapowane na bazę danych, zatem tabele tworzymy bezpośrednio w kodzie projektu.

Obraz zawierający tekst

Opis wygenerowany automatycznie

1. Testy jednostkowe

W projekcie stworzone zostały cztery testy jednostkowe:

1. Dodanie użytkownika do bazy – zakończony sukcesem
2. Dodanie dwóch użytkowników do bazy z tą samą wartością unikatowego pola – zakończony fiaskiem
3. Dodanie słówka do bazy z poprawnymi danymi – zakończony sukcesem
4. Dodanie słówka do bazy ze złym poziomem zaawansowania – zakończony

fiaskiem

Przykładowe testy jednostkowe:

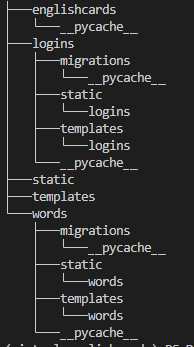
Obraz zawierający tekst, zewnętrzne, zrzut ekranu, srebrny

Opis wygenerowany automatycznie

1. Stosowanie konwencji nazewniczych w programowaniu i formatowania kodu typowego dla wybranej technologii

W Django do organizacji funkcjonalności w kodzie wykorzystywane są tak zwane „aplikacje”, które co do zasady powinny zawierać fragmenty kodu odpowiedzialne za osobną logikę programu. W naszym projekcie oddzieliliśmy zarządzanie kontami użytkowników (logins) od nauki słówek (words).

Struktura projektu:



Struktura pojedynczej aplikacji

Obraz zawierający tekst

Opis wygenerowany automatycznie