



Praca dyplomowa inżynierska

na kierunku Informatyka

Aplikacja do nauki języków z napisów do
filmów udostępnianych na platformie YouTube

Paweł Niziołek

numer albumu 102488

Promotor

dr hab. inż. Szczepan Paszkiel prof. Uczelni

Opole, 2025

Aplikacja do nauki języków z napisów do filmów udostępnianych na platformie YouTube

Streszczenie

Aplikacja umożliwia użytkownikom naukę języków obcych poprzez interaktywne fiszki, które są generowane na podstawie napisów do filmów i seriali dostępnych na platformie YouTube oraz z własnych plików użytkownika. Użytkownicy mogą wybierać napisy w różnych językach i na różnych poziomach trudności, co pozwala na dostosowanie nauki do indywidualnych potrzeb. Aplikacja jest dostępna na różnych urządzeniach, co umożliwia naukę w dowolnym miejscu i czasie.

Słowa kluczowe: youtube, napisy, języki obce, fiszki, nauka, aplikacja

Aplikacja do nauki języków z napisów do filmów udostępnianych na platformie YouTube

Abstract

The application allows users to learn foreign languages through interactive flashcards, which are generated based on subtitles to movies and TV series available on the YouTube platform and from the user's own files. Users can choose subtitles in different languages and at different levels of difficulty, which allows them to tailor their learning to their individual needs. The application is available on various devices, which allows learning anywhere and anytime.

Keywords: youtube, subtitles, foreign languages, flashcards, learning, application

Spis treści

1	Wprowadzenie	6
1.1	Wstęp	6
1.2	Przegląd aktualnych rozwiązań	7
1.2.1	Anki	7
1.2.2	Language Reactor	8
1.2.3	Trancy	10
2	Cel i zakres pracy	12
2.1	Cel pracy	12
2.2	Zakres pracy	12
3	Projekt	14
3.1	Wprowadzenie	14
3.2	Przypadki użycia	14
3.2.1	Przypadek użycia: Rejestracja	15
3.2.2	Przypadek użycia: Logowanie	15
3.2.3	Przypadek użycia: oglądanie filmów	15
3.2.4	Przypadek użycia: Nauka słówek	16
3.3	Baza danych	16
3.3.1	Wprowadzenie	16
3.3.2	Struktura	17
3.3.3	Metody zapobiegania redundancji danych	18
3.4	Interfejs użytkownika	19
3.4.1	Widok startowy	20
3.4.2	Strona Nauki	21
3.4.3	Strona Napisów	22
3.4.4	Strona Słownika	24
3.4.5	Strona Statystyk	25
3.4.6	Strona logowania oraz rejestracji	26
3.4.7	Strona oglądania filmów	27
3.5	Podsumowanie	27

4	Implementacja	28
4.1	Wprowadzenie	28
4.2	Środowisko i narzędzia programistyczne	28
4.2.1	Środowisko programistyczne	28
4.2.2	Wybór technologii	28
4.2.3	Opis technologii	29
4.3	Baza danych	31
4.4	Backend	31
4.4.1	Struktura backendu w Next.js	31
4.4.2	Struktura backendu w Flask	32
4.4.3	LibretTranslate	33
4.5	Frontend	34
4.5.1	Struktura projektu	34
4.5.2	Panel nauki Fiszek	36
4.6	Problemy implementacyjne	37
4.6.1	Wirtualizacja List	37
4.6.2	Przetwarzanie języka naturalnego (NLP)	40
5	Podsumowanie	41
5.1	Wnioski	41
5.2	Kierunki dalszego rozwoju	41
	Bibliografia	41
	Spis rysunków	43
	Spis tabel	44

Rozdział 1

Wprowadzenie

1.1 Wstęp

W dzisiejszym globalnym świecie znajomość języków obcych jest nieocenioną umiejętnością. Tradycyjne metody nauki, takie jak lekcje w klasach, samouczki i aplikacje do nauki słownictwa, często są czasochłonne i nie zawsze dostosowane do indywidualnych potrzeb ucznia. Dodatkowo, oglądanie filmów i seriali w obcym języku jest powszechnie uznawane za skuteczny sposób na poprawę umiejętności językowych, ale brakuje narzędzi, które integrują te aktywności z formalnym procesem nauki. Niniejsza praca inżynierska koncentruje się na stworzeniu innowacyjnej aplikacji webowej, która połączy te dwa aspekty, oferując użytkownikom skuteczniejsze i przyjemniejsze doświadczenie edukacyjne.

Współczesny świat, w którym żyjemy, jest coraz bardziej zglobalizowany i wymaga od nas umiejętności komunikacji w różnych językach, a przede wszystkim w języku angielskim, który stał się międzynarodowym językiem na świecie. Wraz z rozwojem technologii, nauka języków obcych stała się bardziej dostępna i atrakcyjna. Znajomość języków obcych nie tylko otwiera drzwi do nowych możliwości zawodowych, ale także umożliwia pełniejsze zrozumienie innych kultur i poszerza horyzonty. Wraz z dynamicznym rozwojem technologii, nauka języków obcych stała się bardziej dostępna, a tradycyjne metody nauczania ewoluowały, oferując nowe, bardziej interaktywne formy edukacji. Jednym z najpopularniejszych sposobów nauki języków jest korzystanie z platform internetowych, takich jak duoLingo, gdzie użytkownicy mogą uczyć się od podstaw słów i zdań które zostały wcześniej przygotowane. Jednakże, nauka języka obcego w ten sposób ogranicza nas w kwestii wyboru czego chcielibyśmy się dokładnie uczyć.

Coraz więcej osób szuka alternatywnych metod nauki, które są bardziej angażujące i interaktywne. Filmy i seriale oferują naturalny kontekst, w którym używane są różne zwroty i słownictwo, co czyni je doskonałym narzędziem do nauki języka. Oglądanie treści w języku obcym nie tylko pomaga w nauce nowych słów i zwrotów, ale także w poprawie umiejętności słuchania i rozumienia języka w różnych akcentach i dialektach.

Aplikacja ta ma umożliwić użytkownikom aktywne uczestnictwo w procesie nauki, poprzez interaktywne narzędzia i funkcje, które wspomagają naukę słownictwa i gramatyki. Wśród nich znajdują się m.in. możliwość zapisu słów z listy napisów, które są wyświetlane pod lub obok odtwarzacza video, a także możliwość dodania ich do bazy danych, aby uniknąć powtórzeń baza nie przyjmie drugiego takiego samego słowa użytkownikowi. Użytkownik będzie miał dostęp do panelu nauki, słownika wszystkich słów, możliwości logowania z różnych urządzeń obsługujących przeglądarkę, a także do różnych sposobów prezentacji danych, takich jak słownik, flashcards czy moduł do edycji napisów.

Aplikacja ta ma również uwzględnić elementy gamifikacji, aby zachęcić użytkowników do nauki i śledzenia postępów. Na profilu użytkownika będą widoczne wszystkie nauczone słowa, a także osobna podstrona z wykresami i informacjami o postępach. Dzięki tej aplikacji, użytkownicy będą mogli efektywnie i atrakcyjnie uczyć się języka obcego, korzystając z platformy YouTube i własnych filmów z napisami z dysku własnego komputera. Napisy których użytkownik może użyć będą w różnych formatach, więc w aplikacji będzie można wybrać rodzaj pliku i przekopiować całą zawartość lub wrzucić plik w odpowiednie miejsce, napisy muszą zostać zapisane w systemie ponieważ nie ma możliwości zapisaniu ścieżki do żadnego pliku ze względów bezpieczeństwa w internecie.

Wybór technologii do tworzenia aplikacji webowej jest kluczowy dla jej stabilności, skalowalności i wydajności. W projekcie tej aplikacji językowej zdecydowano się na framework Next.js, który oparty jest na React i oferuje wiele korzyści. Jedną z głównych zalet Next.js jest możliwość elastycznego renderowania treści, zarówno po stronie serwera (SSR), jak i klienta (CSR). Dzięki SSR, aplikacja może szybko ładować wstępnie załadowane strony, co znacząco poprawia widoczność w wyszukiwarkach (SEO - Search Engine Optimization) i przyspiesza czas ładowania, co jest szczególnie istotne dla użytkowników korzystających z platformy edukacyjnej. CSR z kolei umożliwia dynamiczne i płynne aktualizacje interfejsu bez konieczności przeładowywania całej strony, co poprawia doświadczenie użytkownika.

1.2 Przegląd aktualnych rozwiązań

1.2.1 Anki

Anki to popularna aplikacja edukacyjna oparta na systemie powtórek rozłożonych w czasie (Spaced Repetition System – SRS). Dzięki temu mechanizmowi nauka jest bardziej efektywna, ponieważ aplikacja prezentuje użytkownikowi informacje w odpowiednich odstępach czasowych, co pomaga w utrwaleniu materiału. Anki wyróżnia się uniwersalnością i możliwością dostosowania do różnych potrzeb, takich jak nauka języków obcych, przygotowanie do egzaminów czy zapamiętywanie faktów w innych dziedzinach.

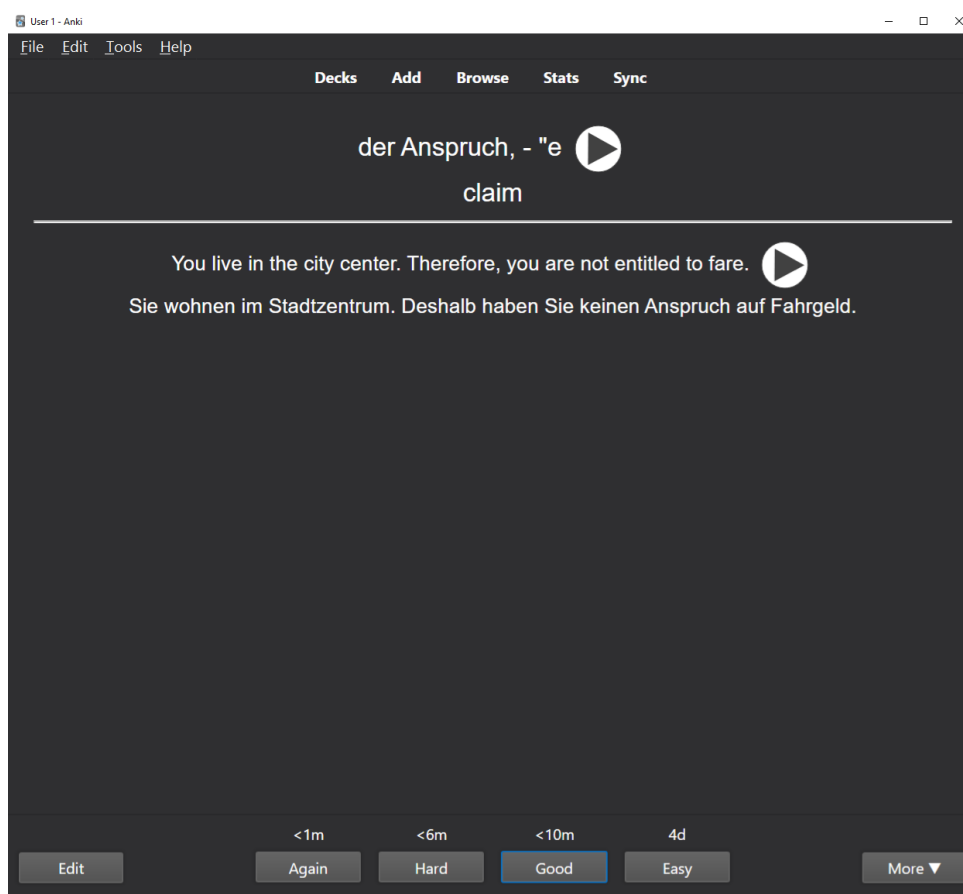
Funkcjonalności:

- Możliwość ręcznego dodawania kart z różnymi typami treści, w tym tekstów, obrazów, dźwięków i nagrań wideo.

- Obsługa dodatków (pluginów), które rozszerzają możliwości aplikacji, np. importowanie napisów filmowych.
- Analiza postępów użytkownika z wykorzystaniem statystyk i wykresów.

Ograniczenia:

- Podczas oglądania filmu użytkownik musi ręcznie przerywać oglądanie, aby dodawać niezbędne informacje do kart. Utrudnia to płynność procesu i może obniżać komfort nauki.
- Mimo tych niedogodności Anki pozostaje dobrym rozwiązaniem do nauki języków, szczególnie dzięki możliwości personalizacji kart i śledzenia postępów.



Rysunek 1. Nauka słówek w aplikacji Anki

1.2.2 Language Reactor

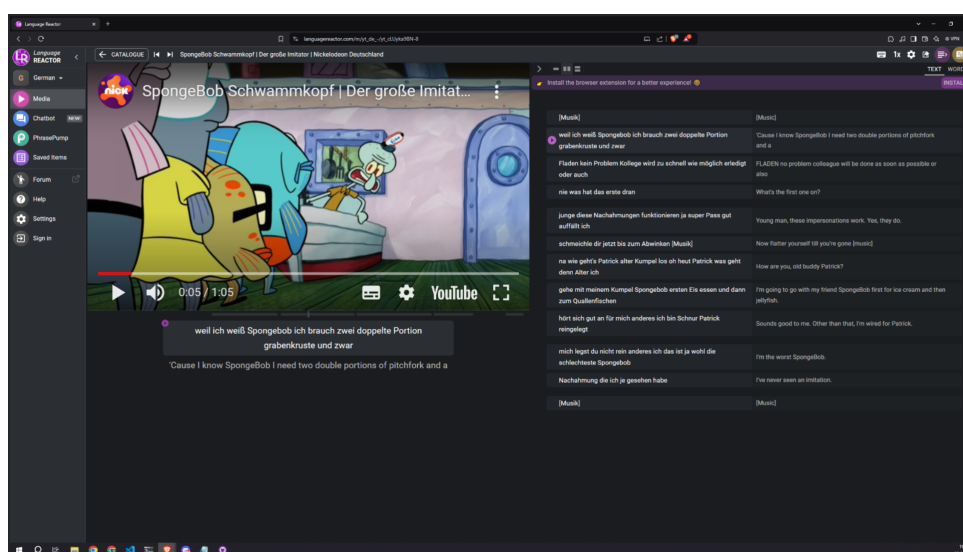
Language Reactor to narzędzie edukacyjne, które umożliwia naukę języków obcych w sposób przyjemny i efektywny poprzez oglądanie filmów i seriali. Aplikacja oferuje interaktywne napisy, które umożliwiają tłumaczenie słów i zwrotów bezpośrednio na ekranie. Dzięki tej funkcji użytkownicy mogą szybko sprawdzić znaczenie nowego słowa, klikając na nie podczas oglądania, lub oznaczyć jako słowo do nauki, ale jest to możliwe dopiero po zapłaceniu za usługę "pro" na stronie.

Funkcjonalności:

- **Interaktywne napisy:** Jedną z głównych zalet Language Reactor jest możliwość wyświetlania tłumaczeń słów i zwrotów w trakcie oglądania, co sprawia, że nauka odbywa się w naturalnym kontekście. Dzięki temu użytkownik może natychmiast zobaczyć, jak dane słowo funkcjonuje w zdaniu.
- **Integracja z platformami streamingowymi:** Aplikacja działa z popularnymi serwisami, takimi jak YouTube, Netflix czy Disney+, co oznacza, że użytkownicy mogą korzystać z niej podczas oglądania ulubionych filmów i seriali w obcym języku.
- **Słownik i lematyzacja:** Language Reactor automatycznie przetwarza słowa na ich formy podstawowe (lematy), co pomaga w nauce gramatyki oraz zapamiętywaniu nowych słówek bez względu na ich odmianę.
- **Baza słówek:** Użytkownicy mogą zapisywać słówka, które napotkali podczas oglądania, tworząc spersonalizowaną listę do późniejszego przyswajania. To rozwiązanie pozwala na systematyczną naukę i powtórki.
- **System powtórek SRS:** Aplikacja wprowadza system powtórek rozłożonych w czasie, co wspomaga długotrwałe zapamiętywanie materiału, podobnie jak w przypadku Anki.
- **ifDostosowanie poziomu trudności:** Użytkownicy mogą dostosować poziom trudności materiałów, co pozwala na naukę dostosowaną do ich umiejętności i tempa.

Ograniczenia:

- **Ograniczona dostępność:** Language Reactor jest dostępny tylko na wybranych platformach streamingowych, co ogranicza możliwość korzystania z aplikacji do nauki języków z innych źródeł.
- **Płatne funkcje:** Pełna funkcjonalność aplikacji, w tym możliwość zapisywania słówek i korzystania z systemu powtórek, jest dostępna tylko w wersji płatnej, co może być barierą dla niektórych użytkowników.



Rysunek 2. Interaktywne napisy na stronie Language Reactor

Możliwość nauki tylko z wybranych kanałów youtube nie można wybrać filmu który nie jest na liście strony internetowej.

1.2.3 Trancy

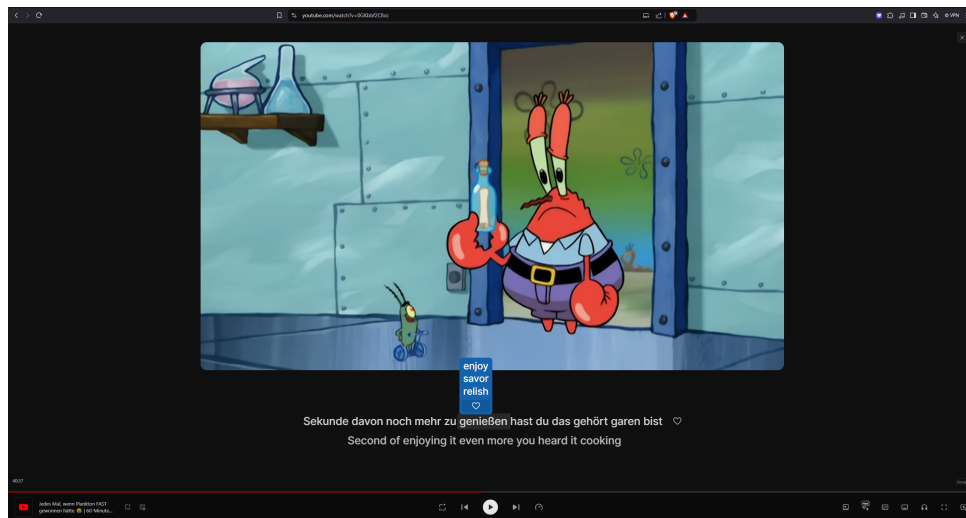
Trancy to aplikacja stworzona z myślą o nauce języków obcych, która integruje naukę słówek z oglądaniem filmów, seriali i innych materiałów wideo. Aplikacja oferuje funkcje, które pozwalają użytkownikom uczyć się języka poprzez interaktywne napisy, tłumaczenia oraz dodatkowe ćwiczenia, co sprawia, że proces nauki staje się bardziej angażujący i efektywny.

Funkcjonalności:

- **Interaktywne napisy:** Trancy umożliwia wyświetlanie napisów w różnych językach, z tłumaczeniami słów i zwrotów. Dzięki temu użytkownicy mogą szybko zrozumieć, co oznacza dane słowo, a także zobaczyć je w kontekście. Napisy są dostosowane do poziomu zaawansowania użytkownika, co pozwala na naukę w sposób dostosowany do indywidualnych potrzeb.
- **Zbieranie słówek:** Użytkownicy mogą zapisywać napotkane słówka, które następnie mogą zostać przekształcone w fiszki do nauki. Taki system pozwala na systematyczne utrwalanie materiału i umożliwia szybkie powtórki w dogodnym czasie.
- **Tłumaczenia i definicje:** Aplikacja oferuje tłumaczenie słów na różne języki oraz wyświetlanie ich definicji, co wspomaga naukę gramatyki i budowanie słownictwa. Użytkownicy mogą korzystać z wbudowanego słownika, aby na bieżąco zgłębiać znaczenie nowych wyrazów.
- **Fiszki SRS:** Trancy wykorzystuje system powtórek rozłożonych w czasie (SRS), który pomaga w długotrwałym zapamiętywaniu słówek. Użytkownik dostaje powiadomienia o konieczności powtórki, co pozwala na regularne utrwalanie materiału i skuteczniejszą naukę.

Ograniczenia:

- **Wymaga płatnej subskrypcji:** Aby w pełni skorzystać z funkcji aplikacji, użytkownik musi zapłacić, w wersji darmowej są tylko niezbędne narzędzia jak tłumaczenie i limit do 100 słów do zapisania.
- **Ograniczona integracja z platformami:** Trancy oferuje integrację z wybranymi platformami streamingowymi, i nie daje możliwości nauki z własnych zapisanych filmów.
- **Wymagana wtyczka w przeglądarce:** Żeby dodawać słowa lub oglądać filmy musimy robić to poza stroną Trancy z użyciem wtyczki Trancy.



Rysunek 3. Interaktywne napisy Trancy

Rozdział 2

Cel i zakres pracy

2.1 Cel pracy

Głównym celem niniejszej pracy jest zaprojektowanie i implementacja aplikacji webowej wspomagającej naukę języków obcych poprzez oglądanie filmów i seriali. Aplikacja ma umożliwiać użytkownikom naukę wybranego języka obcego podczas oglądania treści multimedialnych własnego wyboru. Główne założenia to łatwość w nauce, umożliwienie użytkownikom nauki języków poprzez napisy w filmach i serialach bez potrzeby tworzenia kart do nauki w innych aplikacjach, takich jak Anki, oszczędzając przy tym dużo czasu użytkownika. Połączenie nauki z rozrywką, użytkownicy będą mogli uczyć się języka podczas oglądania swoich ulubionych filmów i seriali, co czyni naukę bardziej przyjemną i efektowną. Spersonalizowane środowisko nauki, dostęp do panelu użytkownika z podsumowaniem postępów w nauce, nauka słownictwa i gramatyki bez ograniczeń poprzez fiszki, możliwość śledzenia statystyk i wykresów przedstawiających postępy w nauce, a także gamifikacja procesu nauki poprzez nagradzanie użytkowników za osiągnięcia. Aplikacja będzie motywować użytkowników do nauki poprzez wizualizację ich postępów za pomocą statystyk, wykresów i informacji widocznych na profilu użytkownika. Zapewnienie kompatybilności, aplikacja będzie dostępna z różnych przeglądarek internetowych i urządzeń. Innowacyjne rozwiązania technologiczne, wykorzystanie sztucznej inteligencji do tłumaczenia napisów i ich przetwarzanie przy użyciu NLP, oraz zastosowanie nowoczesnych technologii webowych do zapewnienia płynnego działania aplikacji.

2.2 Zakres pracy

Zakres pracy obejmuje szczegółowe opisanie i realizację następujących etapów:

- **Opracowanie architektury systemu.**
- **Projektowanie i implementacja bazy danych.**
- **Tworzenie logiki aplikacji oraz jej frontendu i backendu.**
- **Zabezpieczenie aplikacji przed redundancją danych oraz niepoprawnymi wpisami.**
- **Umożliwienie użytkownikom logowania z różnych urządzeń obsługujących przeglądarki.**

- **Implementacja różnych metod nauki (słownik, flashcards, edycja napisów) oraz elementów gamifikacji.**

Na rynku dostępne są różne aplikacje wspomagające naukę języków obcych, takie jak Duolingo, Anki, Quizlet, Memrise, Babbel, Rosetta Stone, LingQ, FluentU, Clozemaster, itp.

Rozdział 3

Projekt

3.1 Wprowadzenie

Celem tego rozdziału jest przedstawienie pracy projektowej aplikacji webowej wspomagającej naukę języków obcych za pomocą filmów i seriali. W kolejnych sekcjach opisano architekturę systemu, projekt bazy danych, implementację logiki aplikacji oraz jej frontendu i backendu, zabezpieczenia przed redundancją danych oraz niepoprawnymi wpisami, możliwość logowania z różnych urządzeń obsługujących przeglądarki, implementację różnych metod nauki oraz elementów gamifikacji.

3.2 Przypadki użycia

Przypadki użycia (ang. use cases) są techniką modelowania funkcjonalności systemu z perspektywy użytkownika. Służą do opisywania interakcji między użytkownikami (aktorami) a systemem, które prowadzą do osiągnięcia określonego celu. Każdy przypadek użycia opisuje sekwencję kroków, które użytkownik wykonuje, aby osiągnąć zamierzony rezultat.

Przypadki użycia są często wykorzystywane w procesie analizy wymagań, ponieważ pomagają zrozumieć, jakie funkcje systemu są potrzebne i jak będą one wykorzystywane przez użytkowników. Mogą być również używane do tworzenia scenariuszy testowych, które sprawdzają, czy system spełnia określone wymagania.

Podstawowe elementy przypadku użycia to:

- **Aktorzy** - osoby, organizacje lub systemy, które wchodzą w interakcję z systemem.
- **Cel** - rezultat, który aktor chce osiągnąć.
- **Scenariusz** - sekwencja kroków, które prowadzą do osiągnięcia celu.

Przypadki użycia mogą być przedstawiane w formie tekstowej lub graficznej (diagramy przypadków użycia). Diagramy przypadków użycia są częścią języka UML (Unified Modeling Language) i składają się z aktorów, przypadków użycia oraz relacji między nimi.

Poniżej przedstawiono przykładowe przypadki użycia dla aplikacji webowej wspomagającej naukę języków obcych za pomocą filmów i seriali.

3.2.1 Przypadek użycia: Rejestracja

- **Aktorzy:** Użytkownik
- **Cel:** Utworzenie nowego konta użytkownika
- **Scenariusz:**
 1. Użytkownik otwiera stronę rejestracji.
 2. Użytkownik wprowadza adres e-mail, hasło i potwierdzenie hasła.
 3. Użytkownik klika przycisk "Zarejestruj się".
 4. System weryfikuje poprawność danych rejestracyjnych.
 5. System tworzy nowe konto użytkownika i przekierowuje go do strony głównej aplikacji.

3.2.2 Przypadek użycia: Logowanie

- **Aktorzy:** Użytkownik
- **Cel:** Zalogowanie się do aplikacji
- **Scenariusz:**
 1. Użytkownik otwiera stronę logowania.
 2. Użytkownik wybiera metodę logowania:
 - **Logowanie za pomocą adresu e-mail i hasła:**
 - (a) Użytkownik wprowadza adres e-mail i hasło.
 - (b) Użytkownik klika przycisk "Zaloguj się".
 - (c) System weryfikuje dane logowania.
 - **Logowanie za pomocą konta Google:**
 - (a) Użytkownik klika przycisk "Zaloguj się przez Google".
 - (b) System przekierowuje użytkownika do strony logowania Google.
 - (c) Użytkownik wprowadza dane logowania do konta Google.
 - (d) Google weryfikuje dane logowania.
 3. System autoryzuje użytkownika i przekierowuje go do strony głównej aplikacji.

3.2.3 Przypadek użycia: oglądanie filmów

- **Aktorzy:** Użytkownik
- **Cel:** Oglądanie zapisanych filmów i seriali
- **Scenariusz:**
 - Użytkownik otwiera stronę oglądania filmów.
 - **Użytkownik wybiera film z listy youtube:**
 1. Użytkownik wybiera film z listy.
 2. System odtwarza wybrany film w oknie odtwarzacza.

3. Użytkownik może zmieniać język napisów, przewijać film oraz dodawać słowa do bazy danych.

– **Użytkownik wybiera film z listy:**

1. Użytkownik wybiera napisy z listy.
2. Poniżej odtwarzacza wybiera odpowiedni film z swojego dysku.
3. System odtwarzaabrany film w oknie odtwarzacza.
4. Użytkownik może zmieniać język napisów, przewijać film oraz dodawać słowa do bazy danych.

3.2.4 Przypadek użycia: Nauka słówek

- **Aktorzy:** Użytkownik
- **Cel:** Nauka słówek za pomocą fiszek
- **Scenariusz:**
 1. Użytkownik otwiera stronę nauki słówek (FlashCards).
 2. Użytkownik wybiera zestaw fiszek do nauki.
 3. System wyświetla fiszki z wybranego zestawu.
 4. Użytkownik przegląda fiszki i próbuje odgadnąć tłumaczenie.
 5. Użytkownik sprawdza poprawność tłumaczenia i ocenia swoją odpowiedź.
 6. System zapisuje wynik i przechodzi do kolejnej fiszki.
 7. Po zakończeniu nauki system zapisuje postępy.

3.3 Baza danych

3.3.1 Wprowadzenie

Do przechowywania danych aplikacji wykorzystano bazę danych NoSQL MongoDB Atlas. Baza danych składa się z kilku kolekcji, które przechowują informacje o użytkownikach, napisach, słowach. Wszystkie kolekcje są połączone relacjami, które umożliwiają szybkie wyszukiwanie i filtrowanie danych.

MongoDB Atlas to usługa bazodanowa w chmurze oferowana przez firmę MongoDB. Jest to w pełni zarządzana platforma, która umożliwia łatwe tworzenie, zarządzanie i skalowanie baz danych MongoDB. Atlas oferuje wiele funkcji, takich jak automatyczne tworzenie kopii zapasowych, monitorowanie wydajności, automatyczne skalowanie, a także wysoki poziom bezpieczeństwa dzięki szyfrowaniu danych w spoczynku i w transzycie.

Jedną z głównych zalet MongoDB Atlas jest jego elastyczność i skalowalność. Użytkownicy mogą łatwo dostosować zasoby bazy danych do swoich potrzeb, zwiększając lub zmniejszając moc

obliczeniową oraz przestrzeń dyskową w zależności od wymagań aplikacji. Atlas obsługuje również replikację danych, co zapewnia wysoką dostępność i odporność na awarie.

MongoDB Atlas integruje się z wieloma popularnymi platformami chmurowymi, takimi jak AWS, Google Cloud Platform i Microsoft Azure, co pozwala na łatwe wdrożenie bazy danych w preferowanym środowisku chmurowym. Dodatkowo, Atlas oferuje narzędzia do analizy danych, takie jak MongoDB Charts, które umożliwiają tworzenie interaktywnych wizualizacji danych bezpośrednio z bazy danych.

3.3.2 Struktura

W kontekście aplikacji, MongoDB Atlas zapewnia niezawodne i skalowalne rozwiązanie do przechowywania danych użytkowników, sesji użytkownika, napisów, słów oraz zdań, co umożliwia szybkie i efektywne wyszukiwanie oraz filtrowanie informacji. Dane są przechowywane w kolekcjach, które zawierają dokumenty w formacie JSON (JavaScript Object Notation). Każda kolekcja może zawierać dokumenty o różnej strukturze, co zapewnia dużą elastyczność w przechowywaniu danych.

Aby przyspieszyć wyszukiwanie danych, MongoDB umożliwia tworzenie indeksów na polach dokumentów. Indeksy mogą znacznie poprawić wydajność zapytań, zwłaszcza w przypadku dużych zbiorów danych. W aplikacji wykorzystano indeksy domyślnie na relacjach i kluczach obcych, aby zapewnić szybkie wyszukiwanie i filtrowanie danych. MongoDB jest bazą danych schemaless, co oznacza, że dokumenty w tej samej kolekcji mogą mieć różne struktury. Dzięki temu możemy łatwo dostosowywać strukturę danych do zmieniających się wymagań aplikacji.

Dodatkowo, MongoDB zapewnia wysoką dostępność i skalowalność poprzez replikację i sharding. Replikacja polega na utrzymywaniu wielu kopii danych na różnych serwerach, co zwiększa niezawodność i dostępność bazy danych. Sharding pozwala na podział danych na mniejsze fragmenty, które są przechowywane na różnych serwerach, co umożliwia skalowanie poziome bazy danych.

Poniżej przedstawiono diagram struktury bazy danych, który ilustruje relacje między poszczególnymi kolekcjami.



Rysunek 4. Diagram struktury bazy danych

3.3.3 Metody zapobiegania redundancji danych

W celu zapobiegania redundancji danych w aplikacji zastosowano kilka metod. Przede wszystkim, projekt bazy danych został starannie zaplanowany, aby zminimalizować duplikację danych. Wykorzystano normalizację danych, która polega na podziale danych na mniejsze, logicznie powiązane tabele lub kolekcje, co pozwala na uniknięcie powtarzania tych samych informacji w różnych miejscach.

Dodatkowo, w aplikacji zaimplementowano mechanizmy walidacji danych, które sprawdzają poprawność i spójność danych przed ich zapisaniem do bazy. Dzięki temu możliwe jest wykrycie i eliminacja ewentualnych duplikatów na etapie wprowadzania danych. Poza jednym duplikatem, którym jest kolekcja zdania "sentence" gdy użytkownik usunie napisy z kolekcji "subtitles" to wtedy

w kolekcji "sentence" pozostanie zdanie, które w trakcie nauki z słowami pozostaną bezpiecznie w bazie.

Wykorzystano również indeksy unikalne, które zapewniają, że w danej kolekcji nie mogą istnieć dwa dokumenty z identycznymi wartościami w polach, które powinny być unikalne. Na przykład, w kolekcji użytkowników zastosowano indeks unikalny na polu adresu e-mail, co zapobiega rejestracji dwóch użytkowników z tym samym adresem e-mail. W kolekcji słów zastosowano indeks unikalny na polu słowa "word", co zapobiega dodaniu dwóch identycznych słów przez jednego użytkownika. To samo dotyczy kolekcji napisów, gdzie zastosowano indeks unikalny na polu tytułowym "subtitleTitle" w połączeniu z odcinkiem "episode" co nie pozwala przypadkowo dodać dwa takie same tytuły w połączeniu z odcinkiem.

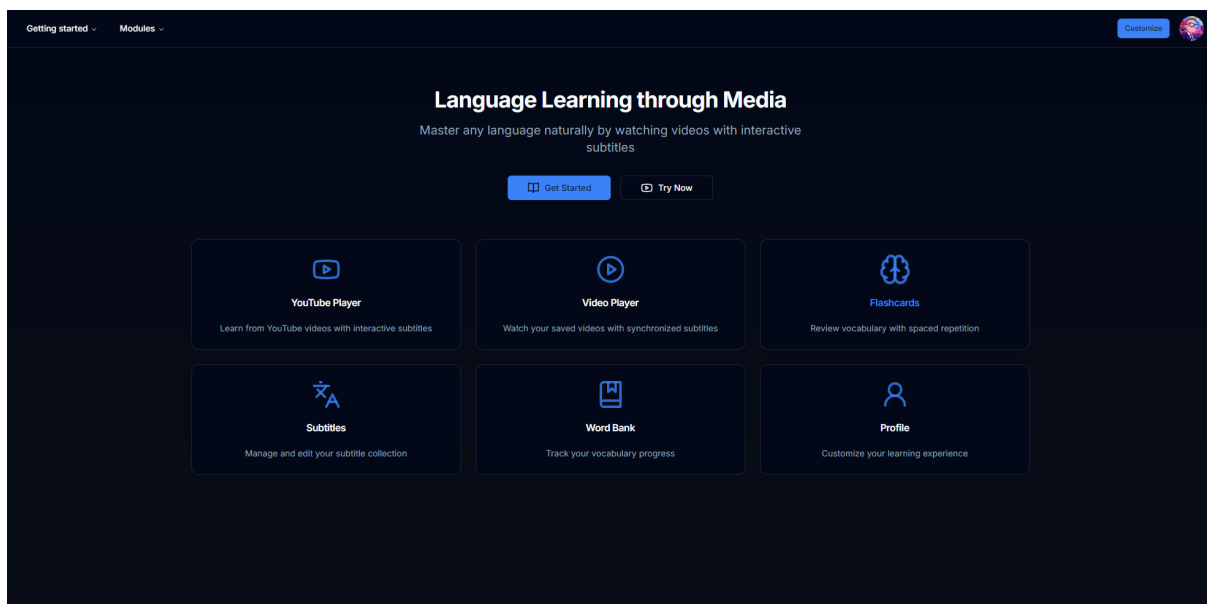
Kolejną metodą zapobiegania redundancji jest stosowanie referencji między kolekcjami. Zamiast przechowywać duplikaty danych, dokumenty w jednej kolekcji mogą zawierać referencje do dokumentów w innych kolekcjach. Na przykład, dokumenty w kolekcji sesji użytkownika mogą zawierać referencje do dokumentów w kolekcji użytkowników, co pozwala na przechowywanie informacji o użytkownikach w jednym miejscu i uniknięcie redundancji.

Wszystkie te metody razem zapewniają, że dane w bazie są przechowywane w sposób efektywny i bez zbędnej redundancji, co przekłada się na lepszą wydajność i spójność danych.

3.4 Interfejs użytkownika

Interfejs użytkownika aplikacji składa się z kilku głównych widoków, które umożliwiają użytkownikom korzystanie z różnych funkcji. Wszystkie widoki zostały zaprojektowane z myślą o prostocie i intuicyjności, aby zapewnić użytkownikom łatwą nawigację i szybki dostęp do potrzebnych informacji. Poniżej przedstawiono najważniejsze elementy interfejsu użytkownika.

3.4.1 Widok startowy



Rysunek 5. Strona główna aplikacji

Strona główna aplikacji zawiera przyciski kierujące do innych sekcji aplikacji, takich jak nauka, słownik, statystyki, profil użytkownika, ustawienia, itp. Użytkownicy mogą również przeglądać najnowsze filmy i serie dostępne w aplikacji oraz korzystać z wyszukiwarki, aby znaleźć interesujące ich treści.

3.4.2 Strona Nauki

The screenshot displays the 'Flashcard Sets' interface. At the top, there's a navigation bar with 'Getting started' and 'Modules'. A search bar and filters (Due: High to Low, All, Due, New, Learning, Mastered) are present. A badge indicates '26 words due'. The main area contains eight flashcard sets, each with a title, word count, mastery percentage, and a list of words with translations. Each set has a 'Delete' button and a 'Learn' button.

Flashcard Sets

Search by title or word... Due: High to Low All Due New Learning Mastered

Words in Subtitles 26 words 0% mastered

Geister der Ark... 6 words 0% mastered

Was passiert in... 5 words 0% mastered

Fuji Kaze - Shi... 5 words 0% mastered

Tauchen in der... 3 words 0% mastered

Star.Wars.Epis... 3 words 0% mastered

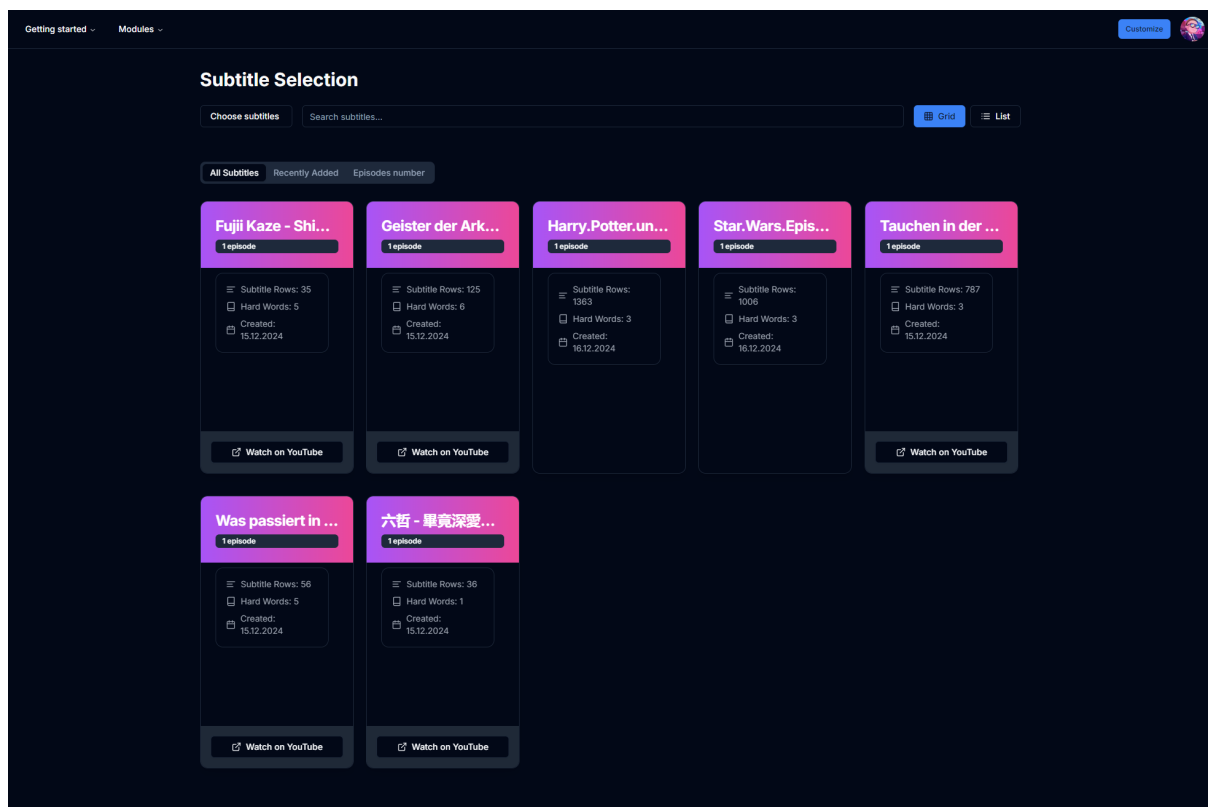
Harry.Potter.un... 3 words 0% mastered

六哲 - 畢竟深愛... 1 words 0% mastered

Rysunek 6. Widok strony głównej fiszek


Strona Nauki to główne miejsce, w którym użytkownicy mogą korzystać z panelu nauki fiszek. Panel zawiera informacje na temat stanu fiszek które są nowe, w trakcie nauki, lub skończone.

3.4.3 Strona Napisów



Rysunek 7. Widok strony napisów

Strona główna umożliwia wyszukiwanie zapisanych napisów, dostępne jest wyszukiwanie po tytule. Użytkownicy mogą również sortować napisy według daty dodania oraz liczby odcinków. Panel daje również możliwość zmiany wyświetlania między siatką a listą domyślnie jest to siatka maksymalnie 5 elementów na wiersz w zależności od miejsca na ekranie.

Getting started ▾ Modules ▾ Customize 

Was passiert in unserem Körper, wenn wir tief tauchen
No episode information

Video URL:
<https://youtu.be/I7SQAn28sGA>

Translate Swap Translation Most Used Words Edit Delete

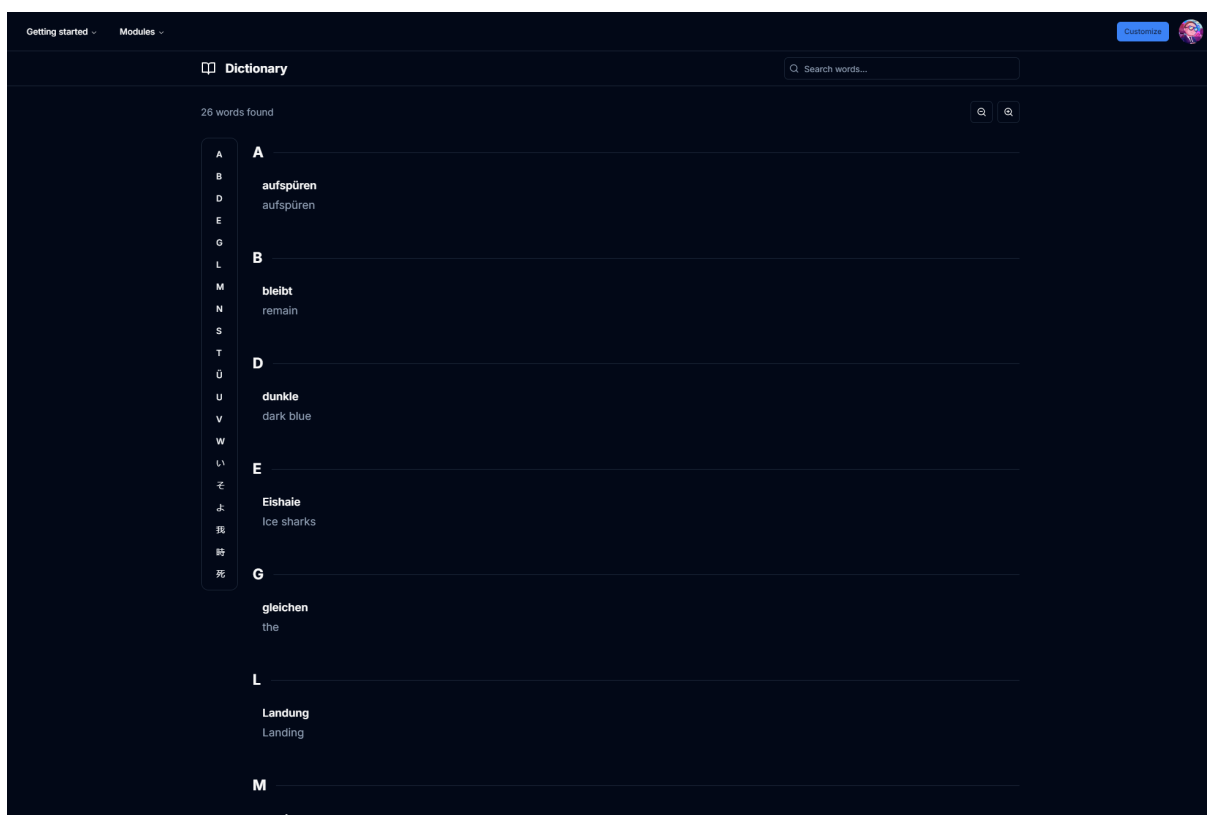
Go Back

Line	Captions	Translation	Time
1	Sporttauchen macht Spaß, ist aber nicht ganz ungefährlich.	Sports diving is fun, but is not quite harmless.	00:00:11
2	Ab einer gewissen Tiefe, da passiert etwas mit uns. Und Schuld daran ist der Wasserdruck.	From a certain depth, something happens to us. And it's the water pressure.	00:00:15
3	Wie wirkt sich extrem hoher Wasserdruck auf meinen Körper aus und wann setzt bei mir der Tiefenrausch ein?	How does extremely high water pressure affect my body and when does the depth rush work?	00:00:20
4	Ich starte meinen Selbstversuch und tauche ab. Wir sind in der belgischen Hauptstadt Brüssel. Meine Tauchlehrerin Bettina ist auch wieder mit dabei.	I'll start my self-try and dive. We are in the Belgian capital Brussels. My diving instructor Bettina is back in.	00:00:27
5	Und ich werde hier gleich einen sehr, sehr hohen Wasserdruck erleben. Denn es geht hier sehr, sehr tief nach unten.	And I will experience a very, very high water pressure here. Because it's very, very deep down here.	00:00:35
6	Genau, es geht dreiunddreißig Meter tief und wir wollen schauen, wie fit du da unten bist.	Exactly, it goes thirty-three feet deep and we want to see how fit you are down there.	00:00:41
7	Deshalb werden wir einen Test machen, einen kleinen Konzentrationstest.	That's why we're gonna do a test, a little concentration test.	00:00:47
8	Den machen wir über Wasser und den gleichen Test machen wir dann auch in der Tiefe nochmal.	We make it over water and the same test then we do it again in depth.	00:00:50
9	Okay. Und dann können wir vergleichen, wie ist es hier oben und wie ist es dann eben da unten.	Okay. And then we can compare how it is up here and how it is down there.	00:00:54
10	Genau. Gut. Der Test beginnt. Deine Aufgabe ist es die Förmchen in diesen Ball reinzukriegen.	Right. The test begins. Your job is to get the balls in this ball.	00:00:58
11	Das ist ja eigentlich was für kleine Kinder, oder? So ein Babyspiel ist das.	That's something for little kids, isn't it? That's a baby game.	00:01:04
12	Es geht nicht darum, dass du das nicht kannst, sondern es geht um Konzentration, wie lange du dafür brauchst und ich stoppe die Zeit.	It's not about you not being able to do this, it's about concentration, how long you need it and I'm gonna stop the time.	00:01:09
13	Und wir machen das Ganze dann später in der Tiefe nochmal. Okay. Also, ich stecke die Förmchen rein und du stoppst die Zeit.	And we'll do it again later in the depths. Okay. So, I'm gonna put the balls in and you stop the time.	00:01:15
14	Ja, einen Moment. Und los.	Yeah, a moment. Let's go.	00:01:20
15	Formen zuordnen auf Zeit, ein Kinderspiel. Fertig.	To assign shapes to time, a children's play. Ready.	00:01:27
16	Neununddreißig Sekunden. Neununddreißig Sekunden über Wasser und jetzt geht's auf dreiunddreißig Meter.	Thirty-nine seconds. Thirty-nine seconds over water and now it goes to thirty-three feet.	00:01:33
17	Dreiunddreißig Meter. So tief bin ich noch nie getaucht. Was erwartet mich wohl da unten? Wie reagiert mein Körper auf den hohen Druck? Werde ich das aushalten?	Thirty-three feet. I've never been so deep. What do you expect me down there? How does my body react to the high pressure? Am I gonna hold that?	00:01:44
18	Schon auf drei Meter spüre ich den Wasserdruck auf den Ohren.	I feel the water pressure on my ears at three meters.	00:02:01
19	Die Trommelfelle werden nämlich nach innen gedrückt; könnten sogar reißen. Wichtig deshalb: Der Druckausgleich.	The drums are pressed inward; could even tear. This is why it is important: pressure compensation.	00:02:06
20	Nase zuhalten und mit Luft von innen die Trommelfelle wieder nach außen drücken.	Keep nose and press the drums out from the inside with air.	00:02:12

Rysunek 8. Widok edycji napisów

Strona Napisów umożliwia użytkownikom przeglądanie i edycję napisów w bazie danych. Użytkownicy mogą edytować istniejące napisy, usuwać napisy, a także przeglądać listę wszystkich napisów w bazie z pomocą sortowania i wyszukiwania.

3.4.4 Strona Słownika



Rysunek 9. Widok słownika

Strona Słownika umożliwia użytkownikom przeglądanie i edycję słów w bazie danych. Użytkownicy mogą edytować istniejące słowa, usuwać słowa, a także przeglądać listę wszystkich słów w bazie z pomocą sortowania i wyszukiwania. Panel obsługuje również chińskie znaki.

3.4.5 Strona Statystyk



Rysunek 10. Widok statystyk i postępów

Strona Statystyk zawiera wykresy i statystyki dotyczące postępów w nauce użytkowników. Użytkownicy mogą śledzić swoje postępy w nauce słówek. Panel zawiera również informacje na temat liczby nowych słówek, oraz słówek nauczonych już wraz z datami utworzenia słowa i ukończenia nauki.

3.4.6 Strona logowania oraz rejestracji

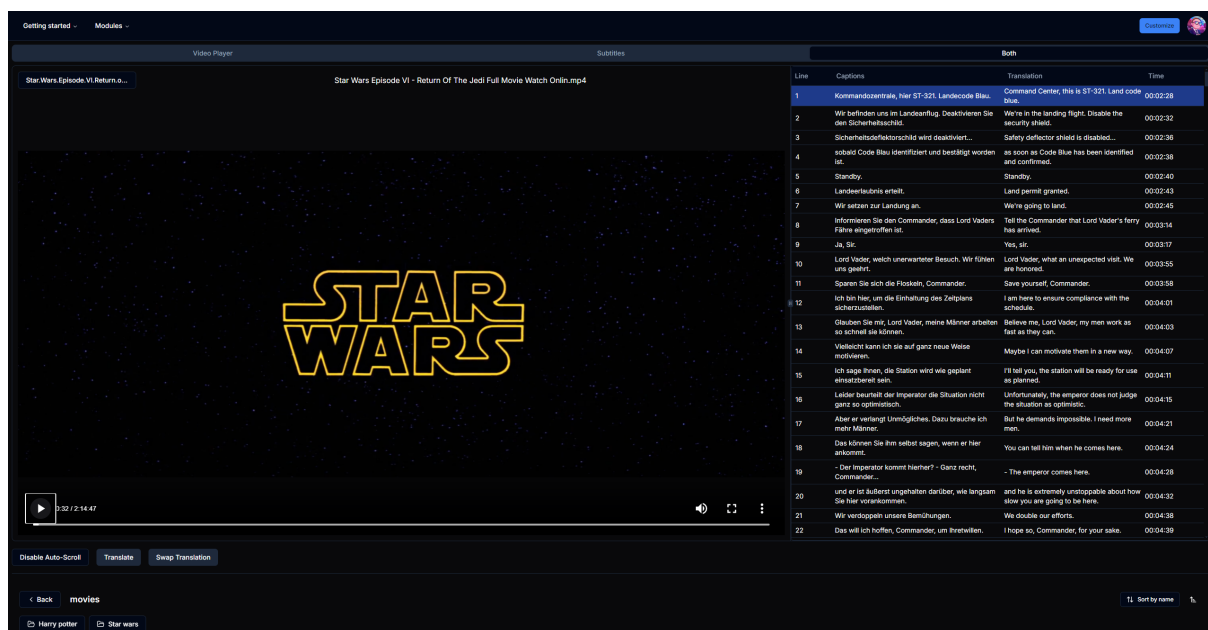
The image shows a user interface for login and registration. At the top, there are two tabs: "Sign Up" and "Sign In". The "Sign In" tab is selected. Below the tabs, the heading "Sign In" is displayed. A message reads: "Enter your credentials to sign in. Click sign in when you're ready." Below this, there are two buttons for social login: "GitHub" and "Google". A separator line with the text "OR CONTINUE WITH" is positioned below these buttons. Underneath, there are two input fields: "Email" with the placeholder text "m@example.com" and "Password". At the bottom, there is a large "Sign In" button.

Rysunek 11. Widok logowania i rejestracji

Strona logowania i rejestracji umożliwia użytkownikom zalogowanie się do aplikacji lub utworzenie nowego konta. Użytkownicy mogą wprowadzić swoje dane logowania, takie jak adres e-mail i hasło,

aby uzyskać dostęp do aplikacji. Panel obsługuje również logowanie za pomocą konta Google, Logowanie za pomocą GitHub będzie dodane również później.

3.4.7 Strona oglądania filmów



Rysunek 12. Widok oglądania filmów

Strona oglądania filmów umożliwia użytkownikom przeglądanie i odtwarzanie filmów i seriali z listy zapisanych napisów i wybrania odpowiedniego filmu z dysku. Użytkownicy mogą zamienić szybko język oryginalny z przetłumaczonym lub włączyć śledzenie aktualnego wiersza napisów względem filmu "auto-scroll", a następnie rozpocząć oglądanie. Panel obsługuje również przewijanie filmu, zmianę języka napisów, oraz dodawanie słów do bazy w celu późniejszej nauki. Użytkownik ma do wyboru albo filmy z rozszerzeniem mp4 albo mkv a napisy w formacie srt lub ass.

3.5 Podsumowanie

Rozdział 4

Implementacja

4.1 Wprowadzenie

W tej sekcji przedstawiono szczegółowy opis implementacji projektu, w tym wybór technologii, narzędzi programistycznych oraz środowiska, w którym został zrealizowany. Omówione zostaną również kluczowe aspekty techniczne, takie jak struktura bazy danych, architektura backendu i frontendu, a także proces testowania i napotkane problemy implementacyjne. Celem tej sekcji jest dostarczenie pełnego obrazu technicznego projektu oraz uzasadnienie wyboru poszczególnych rozwiązań technologicznych.

4.2 Środowisko i narzędzia programistyczne

4.2.1 Środowisko programistyczne

Do implementacji projektu wykorzystano następujące narzędzia i środowiska programistyczne:

- **Visual Studio Code:** Środowisko programistyczne do tworzenia aplikacji internetowych w JavaScript i Python.
- **Git:** System kontroli wersji do zarządzania kodem źródłowym projektu.
- **MongoDB Atlas:** Usługa do hostowania bazy danych MongoDB w chmurze.
- **Flask:** Środowisko do tworzenia aplikacji webowych w Pythonie.
- **TypeScript:** Język programowania, który kompiluje się do JavaScriptu i dodaje typy danych w kodzie.

4.2.2 Wybór technologii

W trakcie realizacji projektu wykorzystano następujące technologie:

- **Next.js:** Framework do tworzenia aplikacji internetowych w React, który oferuje wiele wbudowanych funkcji, takich jak routing, server-side rendering czy generowanie statyczne [2].

- **Python:** Język programowania, który wykorzystałem do implementacji algorytmów przetwarzania języka naturalnego (NLP).
- **React Virtuoso:** Biblioteka do wirtualizacji list w aplikacjach internetowych.
- **Node.js:** Środowisko uruchomieniowe JavaScript, które pozwala na tworzenie aplikacji serwerowych.
- **MongoDB:** Baza danych NoSQL, która umożliwia przechowywanie danych w formacie JSON.
- **React.js:** Biblioteka do tworzenia interfejsów użytkownika w aplikacjach internetowych.
- **Shadcn:** Biblioteka gotowych komponentów do budowy interfejsu użytkownika w React.
- **Redux:** Biblioteka do zarządzania stanem aplikacji w React.
- **Prisma:** ORM do zarządzania bazą danych w Node.js.
- **Axios:** Biblioteka do wykonywania zapytań HTTP w JavaScript.
- **Tailwind CSS:** Narzędzie do tworzenia stylów CSS za pomocą gotowych klas, umożliwiające szybkie projektowanie responsywnych interfejsów.

4.2.3 Opis technologii

Next.js

Next.js to framework do tworzenia aplikacji internetowych w React, który oferuje wiele wbudowanych funkcji, takich jak routing, server-side rendering czy generowanie statyczne. Dzięki temu można tworzyć wydajne i skalowalne aplikacje internetowe, które są przyjazne dla SEO i łatwe w utrzymaniu. Next.js oferuje również wiele gotowych rozwiązań, takich jak automatyczne generowanie ścieżek, obsługa dynamicznych routów czy optymalizacja obrazów [2]. Dzięki temu można skupić się na tworzeniu funkcjonalności, zamiast martwić się o konfigurację i optymalizację aplikacji. Kolejną istotną zaletą Next.js jest intuicyjny i wydajny system routingu oparty na strukturze plików. Ułatwia to organizację aplikacji i nawigację po niej, co jest kluczowe dla zachowania przejrzystości i spójności struktury. Next.js oferuje również uproszczone pobieranie danych oraz wsparcie dla różnych metod stylizacji, takich jak moduły CSS i Tailwind CSS, co pozwala na tworzenie estetycznego i responsywnego interfejsu użytkownika szybciej i łatwiej. Dodatkowo, framework zapewnia wsparcie dla TypeScript, co umożliwia tworzenie bezpiecznego i stabilnego kodu przy użyciu typów które nam podkreślą jeśli będziemy próbowali błędnie używać naszych funkcji lub zmiennych. Wszystkie te cechy czynią Next.js idealnym wyborem do stworzenia nowoczesnej i wydajnej aplikacji językowej.

Python

Python to język programowania, który wykorzystałem do implementacji algorytmów przetwarzania języka naturalnego (NLP). Python jest popularny w dziedzinie analizy danych i uczenia maszynowego, dzięki czemu można znaleźć wiele gotowych bibliotek i narzędzi do przetwarzania tekstu. W moim projekcie wykorzystałem biblioteki takie jak NLTK, spaCy czy TextBlob do lematyzacji, oznaczania części mowy i analizy sentymentu tekstu.

React Virtuoso

React Virtuoso to biblioteka do wirtualizacji list w aplikacjach internetowych. Umożliwia renderowanie długich list danych w sposób efektywny i wydajny, co przyczynia się do poprawy wydajności i płynności interfejsu użytkownika. Dzięki React Virtuoso można renderować tylko widoczne elementy i kilka dodatkowych poza obszarem.

Node.js

Node.js to środowisko uruchomieniowe JavaScript, które pozwala na tworzenie aplikacji serwerowych. Dzięki Node.js można pisać zarówno frontend, jak i backend w jednym języku programowania, co ułatwia rozwój i utrzymanie aplikacji. Node.js oferuje również wiele gotowych modułów i bibliotek, które ułatwiają tworzenie aplikacji internetowych, takich jak Express.js, Socket.io czy Mongoose.

MongoDB

Wybór bazy danych do aplikacji webowej ma ogromne znaczenie dla jej wydajności i skalowalności. W tym projekcie zdecydowano się na MongoDB Atlas, która jest obiektową bazą danych typu NoSQL. MongoDB charakteryzuje się elastyczną strukturą danych, co pozwala na szybkie i efektywne przechowywanie oraz zarządzanie różnorodnymi danymi w formacie JSON. W projekcie baza danych została wykorzystana do przechowywania danych o użytkownikach, słowach do nauki, napisach oraz postępach w nauce.

React.js

React.js to biblioteka do tworzenia interfejsów użytkownika w aplikacjach internetowych. React.js oferuje wiele funkcji i narzędzi, które ułatwiają tworzenie interaktywnych i responsywnych interfejsów. Dzięki React.js można tworzyć komponenty UI, zarządzać stanem aplikacji i reagować na interakcje użytkownika w sposób efektywny i wydajny.

Shadcn

Shadcn to kolekcja komponentów, które można kopiować i wklejać do swoich aplikacji. Nie jest to biblioteka komponentów, którą można zainstalować jako zależność. Shadcn nie jest dostępny ani dystrybuowany przez npm (node package manager). Użytkownik wybiera potrzebne komponenty, kopiuje i wkleja kod do swojego projektu, a następnie dostosowuje go do swoich potrzeb. Kod jest własnością użytkownika. Shadcn może służyć jako odniesienie do budowy własnych bibliotek komponentów do rozbudowy interfejsu użytkownika w React. Shadcn oferuje wiele gotowych rozwiązań, takich jak przyciski, formularze, tabele czy karty, które można łatwo dostosować do własnych potrzeb. Dzięki Shadcn można tworzyć interfejsy użytkownika w sposób szybki i efektywny, co przyczynia się do skrócenia czasu potrzebnego na rozwój aplikacji.

Jedną z kluczowych zalet MongoDB jest jej skalowalność. Baza ta umożliwia łatwe skalowanie poziome, co oznacza, że możemy dodawać nowe serwery do naszego klastra bazodanowego w miarę wzrostu ilości danych i liczby użytkowników. Jest to szczególnie ważne dla aplikacji edukacyjnych, które mogą szybko rosnąć w popularność i wymagać zwiększonej mocy obliczeniowej. Dzięki temu, nasza aplikacja będzie mogła obsługiwać rosnącą liczbę użytkowników bez utraty wydajności. Dodatkowo, MongoDB Atlas oferuje wsparcie dla replikacji danych, co zwiększa niezawodność i dostępność systemu. Funkcja replikacji zapewnia, że dane są automatycznie kopiowane na wiele serwerów, co chroni przed utratą danych i zapewnia ciągłość działania aplikacji. Dzięki tym funkcjom MongoDB Atlas jest idealnym wyborem dla naszej aplikacji, zapewniając jej wydajność, skalowalność i elastyczność w zarządzaniu danymi.

4.3 Baza danych

4.4 Backend

Backend aplikacji został zrealizowany przy użyciu dwóch technologii: Next.js oraz Flask. Każda z tych technologii pełni inną rolę w architekturze aplikacji, co pozwala na wykorzystanie ich mocnych stron. W Flask został wykonany moduł pełniący funkcje lematyzacji i pos taggingu ponieważ modele NLP są tam bardzo mocno rozwinięte. W Next.js zostały wykonane endpointy do komunikacji z bazą danych oraz do obsługi logiki biznesowej aplikacji. Dzięki temu możliwe jest oddzielenie części serwerowej od frontendowej, co pozwala wykorzystać zalety szybkiego serwowania stron statycznych. Poniżej przedstawiono architekturę backendu z wykorzystaniem Next.js i Flask. Dodatkowo został wykorzystany moduł LibretTranslate do tłumaczenia maszynowego w aplikacji.

Next.js

Next.js jest wykorzystywany do obsługi części serwerowej aplikacji, która jest odpowiedzialna za renderowanie stron po stronie serwera (server-side rendering) oraz generowanie statycznych stron (static site generation). Dzięki temu aplikacja jest szybka i przyjazna dla SEO. Next.js umożliwia również łatwe zarządzanie ścieżkami strony które są wyświetlane w pasku url przeglądarki. Głównym celem w projekcie jest obsługa żądań HTTP, komunikacja z bazą danych MongoDB oraz wykonywanie operacji związanych z komunikacją z modułami NLP w Flask jak i modułem LibretTranslate do tłumaczenia maszynowego w aplikacji. Next.js oferuje również wsparcie dla różnych metod autoryzacji i uwierzytelniania, co zapewnia bezpieczeństwo aplikacji jak i szybkość działania i implementacji.

4.4.1 Struktura backendu w Next.js

Struktura backendu w Next.js została podzielona na kilka głównych katalogów, z których każdy odpowiada za określoną część aplikacji. Wszystkie pliki związane z budową backendu znajdują się w katalogu `src/app/api`, który zawiera następujące podkatalogi:

- **auth:** Obsługuje autoryzację i uwierzytelnianie użytkowników.
- **captions:** Zajmuje się pobieraniem napisów z youtube.
- **hardWords:** Zajmuje się zarządzaniem trudnymi słowami.
- **profile:** Obsługuje aktualizacje profilu użytkownika.
- **sentence:** Zajmuje się aktualizacjami zdań.
- **signup:** Obsługuje rejestrację nowych użytkowników.
- **subtitles:** Zajmuje się zarządzaniem napisami do filmów.

4.4.2 Struktura backendu w Flask

Flask jest wykorzystywany do tworzenia API oraz logiki biznesowej aplikacji. Flask to lekki framework webowy dla Pythona, który umożliwia szybkie tworzenie i rozwijanie aplikacji webowych. W projekcie Flask jest odpowiedzialny za przetwarzanie żądań HTTP, oraz wykonywanie operacji związanych z przetwarzaniem języka naturalnego (NLP). użyte do tego zostały modele:

Listing 1. kod do importowania modeli NLP

```

1 nlp_de = spacy.load('de_core_news_md')
2 nlp_ja = spacy.load('ja_core_news_md')
3 nlp_en = spacy.load('en_core_web_md')
4 nlp_pl = spacy.load('pl_core_news_md')
```

zostały one użyte do przetwarzania języka naturalnego w aplikacji. Dzięki temu możliwe jest lematyzacja i pos tagging słów w różnych językach. Do projektu zostały wykorzystane modele językowe dla języków: niemieckiego, japońskiego, angielskiego i polskiego. Reszta języków nie jest obsługiwana w aplikacji, ale można dodać nowe modele językowe w przyszłości.

Listing 2. kod do obsługi lematyzacji i pos tagingu

```

1 @app.post("/nlp")
2 async def analyze_text(request: AnalyzeTextRequest):
3     word = request.word
4     sourceLang = request.sourceLang
5     doc = None
6
7     if not sourceLang:
8         raise HTTPException(status_code=400, detail="Source language is
9         required")
10    if sourceLang == 'de':
11        doc = nlp_de(word)
12    elif sourceLang == 'ja':
13        doc = nlp_ja(word)
14    elif sourceLang == 'en':
15        doc = nlp_en(word)
16    elif sourceLang == 'pl':
17        doc = nlp_pl(word)
```



```

17     elif sourceLang == 'auto':
18         doc = nlp_de(word)
19     else:
20         raise HTTPException(status_code=400, detail=f"Source language
is not currently used: {sourceLang}")
21
22     tokens_with_pos = {'lemma': doc[0].lemma_, 'pos': doc[0].pos_}
23     return {'result': tokens_with_pos}

```

a następnie przetwarza tekst przy użyciu odpowiedniego modelu językowego. W przypadku braku określenia języka źródłowego lub podania nieobsługiwanego języka, zwracany jest odpowiedni komunikat błędu. Wynikiem przetwarzania jest lemat naszego słowa oraz oznaczenie części mowy w obu przypadkach wybrane są te najbardziej prawdopodobne w przetworzonym dokumencie który zwrócił model NLP.

4.4.3 LibretTranslate

LibretTranslate to otwartoźródłowy system tłumaczenia maszynowego, który umożliwia tłumaczenie tekstów pomiędzy różnymi językami. W projekcie został wykorzystany do tłumaczenia tekstów w aplikacji, co pozwala na obsługę użytkowników mówiących różnymi językami. LibretTranslate wspiera wiele języków, co czyni go wszechstronnym narzędziem do tłumaczeń.

Integracja LibretTranslate

Integracja LibretTranslate w aplikacji została zrealizowana poprzez stworzenie funkcji w Next.js, która komunikuje się z API LibretTranslate. Dzięki temu możliwe jest wysyłanie napisów do przetłumaczenia oraz odbieranie przetłumaczonych wyników bezpośrednio w aplikacji. Poniżej przedstawiono przykładowy kod obsługujący tłumaczenie napisów za pomocą LibretTranslate:

Listing 3. Przykładowy kod integracji LibretTranslate w Next.js

```

1 import axios from 'axios';
2 export default async function translateText(req, res) {
3     async function translateSubtitleData(subtitleData: SubtitleData[],
targetLang: string) {
4         try {
5             const texts = subtitleData.map(subtitle => subtitle?.text);
6             const response = await axios.post("http://127.0.0.1:5000/translate"
, {
7                 q: texts,
8                 source: "auto" ,
9                 target: targetLang || "en",
10                format: "text"
11            });
12            let detectedLanguage = "auto";

```

```
13     if(response.data.detectedLanguage[0].language){
14         detectedLanguage = response.data.detectedLanguage[0].language;
15     }
16     return {
17         translatedSubtitleData: response.data.translatedText,
18         detectedLanguage
19     };
20 } catch (error) {
21     console.error('Error translating subtitles:', error);
22     throw new Error('Failed to translate subtitles');
23 }
24 }
25 }
```

Funkcja `translateSubtitleData` przyjmuje dane napisów oraz język docelowy, a następnie wysyła zapytanie do API LibretTranslate w celu przetłumaczenia tekstu. Odpowiedź zawiera przetłumaczone napisy oraz wykryty język źródłowy, co pozwala na dalsze przetwarzanie danych w aplikacji.

4.5 Frontend

4.5.1 Struktura projektu

Struktura projektu została podzielona na kilka głównych katalogów, z których każdy odpowiada za określoną część aplikacji. Wszystkie pliki związane z budową aplikacji znajdują się w katalogu `src`, który zawiera następujące podkatalogi:

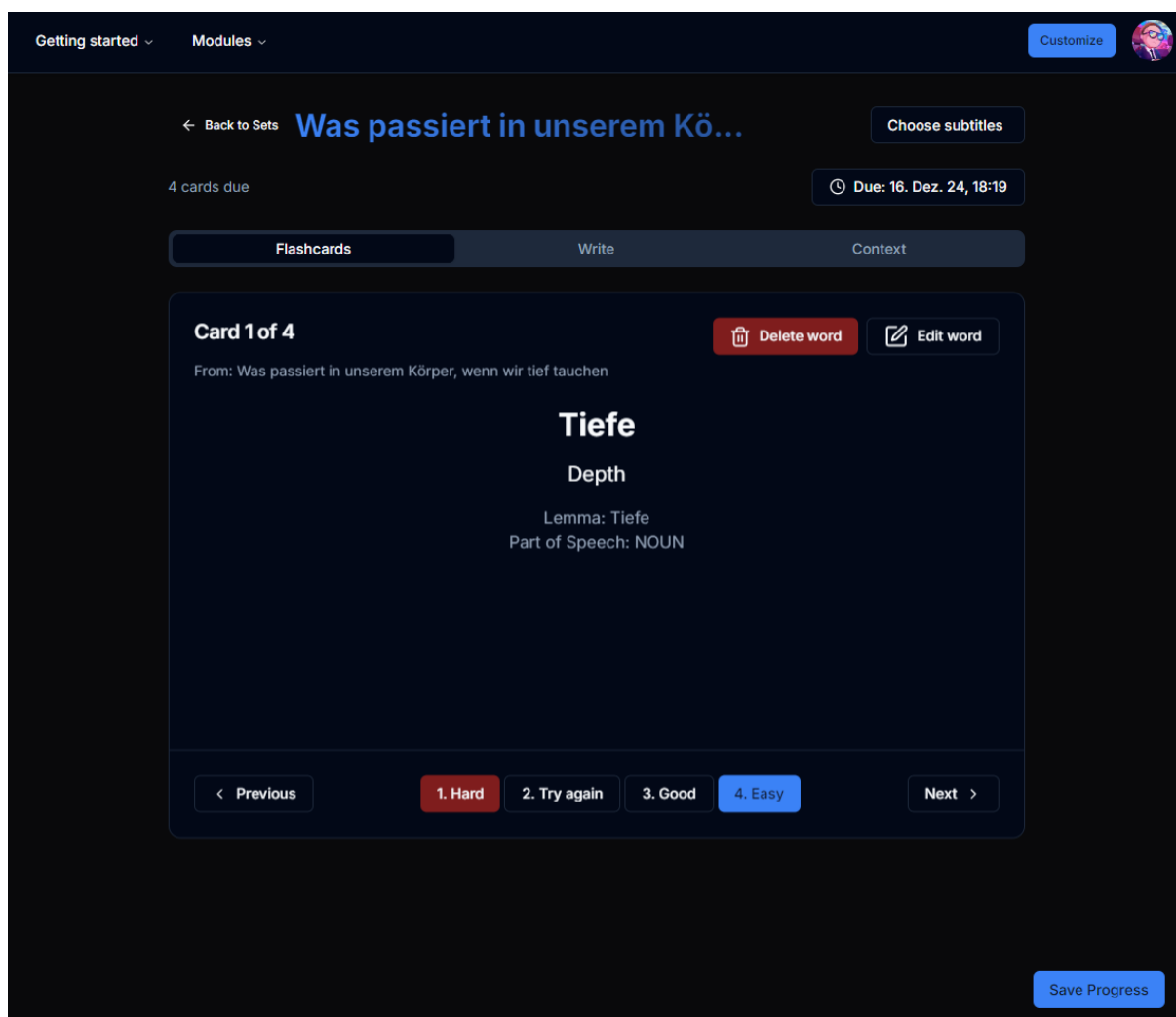
- **app**: Zawiera wszystkie strony aplikacji, które są renderowane przez Next.js. Każda strona jest reprezentowana przez plik JavaScript, który eksportuje komponent React.
- **app/home**: Jest to strona główna aplikacji, która wyświetla różne sekcje, takie jak podstrona napisów, podstrona do nauki czy podstrona statystyk użytkownika, razem z resztą podstron wszystkie są zawarte w tym katalogu `home` poza stroną authoryzacji.
- **app/auth**: Jest to strona autoryzacji, która wyświetla formularze logowania i rejestracji użytkownika.
- **app/api**: Zawiera pliki z funkcjami, które obsługują zapytania API do serwera. W projekcie wykorzystano bibliotekę `axios` do wykonywania zapytań HTTP.
- **styles**: Zawiera pliki CSS, które definiują styl aplikacji. W projekcie wykorzystano ją do tworzenia domyślnych stylów aplikacji i stylów motywów które dają kontrolę nad kolorami i zaokrągleniem obramowań.
- **components**: Zawiera wszystkie komponenty interfejsu użytkownika, takie jak przyciski, formularze, tabele czy karty.

- **hooks:** Zawiera pliki z hookami, które są wykorzystywane w różnych częściach aplikacji. W projekcie wykorzystano hooki do zarządzania stanem, efektami ubocznymi i logiką aplikacji.
- **providers:** Zawiera pliki z providermi, które są wykorzystywane do dostarczania kontekstów i hooków do komponentów aplikacji.
- **lib:** Zawiera pliki z funkcjami pomocniczymi, które są wykorzystywane w różnych częściach aplikacji. W projekcie znajduje się tam głównie kod związany z Redux, który jest używany do zarządzania stanem aplikacji.
- **styles:** Zawiera pliki CSS, które definiują styl aplikacji. W projekcie wykorzystano bibliotekę styled-components do tworzenia stylów w JavaScript.
- **types:** Zawiera pliki z typami, które definiują struktury danych w aplikacji. W projekcie wykorzystano TypeScript do dodawania typów danych w kodzie. są one domyślnie w plikach .d.ts które pozwalają na dodanie typów do plików JavaScript bez podawania ścieżki działają one automatycznie.

poza tym katalogiem znajdują się również pliki konfiguracyjne, takie jak `package.json`, `tsconfig.json` czy `.env`, które definiują zależności, ustawienia TypeScript i zmienne środowiskowe aplikacji. Struktura projektu została zaprojektowana w taki sposób, aby była czytelna i łatwa w utrzymaniu, co przyczynia się do szybszego rozwoju aplikacji i łatwiejszej nawigacji.

I pozostałe katalogi w projekcie. Katalog `public` zawiera pliki statyczne, takie jak obrazy, ikony czy pliki konfiguracyjne, które są dostępne publicznie i mogą być bezpośrednio serwowane przez serwer. Katalog `prisma` zawiera pliki konfiguracyjne i schematy bazy danych używane przez Prisma ORM do zarządzania bazą danych. Prisma umożliwia łatwe definiowanie modeli danych i wykonywanie zapytań do bazy danych. Katalog `node_modules` zawiera wszystkie zainstalowane zależności projektu, które są pobierane za pomocą npm (Node Package Manager). Katalog ten jest automatycznie generowany i zarządzany przez npm na podstawie pliku `package.json`.

4.5.2 Panel nauki Fiszek



Rysunek 13. Panel nauki fiszek

Panel nauki w aplikacji został zaprojektowany w oparciu o system fiszek oraz system powtórek oparty na algorytmie SRS (Space Repetition System). System fiszek umożliwia użytkownikom naukę nowych słów i zwrotów poprzez prezentowanie im kart z słowami i tłumaczeniami. Użytkownik może ocenić swoją znajomość danego słowa, co wpływa na to czy słowo przejdzie dalej czy nasze powtórki zostaną cofnięte.

System powtórek SRS

System SRS jest algorytmem, który optymalizuje proces nauki poprzez dostosowanie interwałów powtórek do indywidualnych potrzeb użytkownika. W praktyce oznacza to, że słowa, które użytkownik zna dobrze, będą pojawiać się rzadziej, natomiast te, które sprawiają trudności, będą powtarzane

częściej. Dzięki temu użytkownik może efektywnie zarządzać swoim czasem nauki i skupić się na materiałach, które wymagają większej uwagi.

W panelu nauki użytkownik ma dostęp do różnych zestawów fiszek, które mogą być tworzone ręcznie lub generowane automatycznie na podstawie jego postępów i preferencji. Każda fiszka zawiera słowo lub zwrot w języku obcym oraz jego tłumaczenie lub definicję. Użytkownik może również edytować fiszki, dodać tłumaczenie lub usuwać niepotrzebne lub niepoprawne informacje.

Dzięki zastosowaniu systemu SRS, panel nauki w aplikacji zapewnia skuteczną metodę nauki języka, która pozwala na osiąganie lepszych wyników w krótszym czasie. System automatycznie dostosowuje interwały powtórek na podstawie wyników użytkownika, co optymalizuje proces nauki bez konieczności ręcznej ingerencji.

4.6 Problemy implementacyjne

4.6.1 Wirtualizacja List

Wirtualizacja listy w aplikacjach internetowych to technika, która optymalizuje renderowanie długich list danych. Bez niej aplikacja renderuje wszystkie elementy listy na raz, co może prowadzić do problemów z wydajnością, zwłaszcza gdy lista jest duża. Wirtualizacja polega na renderowaniu jedynie tych elementów, które aktualnie są widoczne w przeglądarce użytkownika, dzięki czemu zużycie zasobów jest minimalne, a aplikacja działa płynniej. Bez wirtualizacji listy, użytkownik mógłby doświadczać opóźnień w interakcji z interfejsem, tym większych im dłuższa lista danych przy 2 tysiącach wierszy opóźnienie stawało się uciążliwe ponieważ czekało się parę sekund na reakcję interfejsu listy i pokazanie okna dodawania trudnych słów do systemu.

Jak działa wirtualizacja listy

- **Obserwacja widocznych elementów:** Komponent śledzi pozycję widoku użytkownika w liście. Renderowane są tylko te elementy, które mieszczą się w aktualnie widocznym obszarze (viewport) oraz kilka dodatkowych elementów „na zapas” wokół tego obszaru.
- **Renderowanie na żądanie:** Gdy użytkownik przewija listę, niewidoczne elementy są dynamicznie usuwane z DOM-u, a nowe – wczytywane na ich miejsce.
- **Stała wysokość elementów (lub szacowana):** Dla prawidłowego działania, komponent wirtualizujący często wymaga, aby elementy listy miały stałą lub przynajmniej przewidywalną wysokość. Dzięki temu może łatwo obliczać, które elementy powinny być aktualnie wyświetlane.
- **Oszczędność zasobów:** Dzięki renderowaniu tylko niewielkiej liczby elementów, zmniejsza się zużycie pamięci i obciążenie procesora, co prowadzi do szybszego działania aplikacji.

Przykładowy kod JavaScript

Poniżej przedstawiono przykładowy kod tabeli Tanstack, który ilustruje operowanie na wierszach tabeli, takie jak śledzenie aktualnego do filmu wiersza napisów. Wiersz jest automatycznie przewijany do środkowego obszaru widocznego dla użytkownika, co ułatwia śledzenie napisów w czasie rzeczywistym. A index wiersza jest wybierany na podstawie czasu odtwarzania filmu, wybierany jest pierwszy wiersz na początek a potem przechodzi się na kolejny jeśli czas przekroczy wartość czasu w sekundach kolejnego wiersza.

Listing 4. Tablica Tanstack do wirtualizacji listy

```

1  export function DataTable<TData, TValue>({ captions, height }: {
    captions: Caption[], height: string }) {
2      const [sorting, setSorting] = useState<SortingState>([]);
3      const [currentIndex, setCurrentIndex] = useState<number>(0);
4      const playedSeconds = useSelector((state: any) =>
    state.subtitle.playedSeconds);
5      const autoScrollEnabled = useSelector((state: any) =>
    state.subtitle.autoScrollEnabled);
6      const tableRef = useRef<TableVirtuosoHandle>(null);
7      const table = useReactTable({
8          data: captions,
9          columns: columns,
10         state: {
11             sorting,
12         },
13         onSortingChange: setSorting,
14         getCoreRowModel: getCoreRowModel(),
15         getSortedRowModel: getSortedRowModel(),
16     });
17     const { rows } = table.getRowModel()
18     useEffect(() => {
19         if (rows.length > 0) {
20             let newIndex = -1;
21             for (let i = 0; i < rows.length; i++) {
22                 const row = rows[i];
23                 const startTime = row.original.start ?? 0;
24                 const nextStartTime = rows[i + 1]?.original.start ??
Infinity;
25                 if (playedSeconds >= startTime && playedSeconds <
nextStartTime) {
26                     newIndex = i;
27                     break;
28                 }
29             }
30         }

```

```

31         if (newIndex === -1) {
32             newIndex = 0;
33         }
34         setCurrentIndex(newIndex);
35         if (autoScrollEnabled && tableRef.current && newIndex !==
-1) {
36             tableRef.current.scrollToIndex({
37                 index: newIndex,
38                 align: "center",
39                 behavior: "smooth",
40             });
41         }
42     }
43     }, [playedSeconds, rows, autoScrollEnabled]);

```

Dlaczego React Virtuoso

React Virtuoso jest biblioteką do wirtualizacji list, która znacznie upraszcza implementację tego mechanizmu w React. Automatycznie obsługuje:

- **Przewijanie:** Zajmuje się wykrywaniem widocznych elementów, reagując na przewijanie użytkownika.
- **Niestandardowe wysokości elementów:** Obsługuje zarówno stałe, jak i zmienne wysokości elementów, co czyni go bardziej elastycznym.
- **Lazy loading:** Umożliwia ładowanie danych w locie, co jest kluczowe dla dużych list z elementami, które mogą być dynamicznie ładowane z serwera.

Zastosowania

Virtualizacja listy jest szczególnie przydatna w przypadku:

- **Długich list:** Kiedy lista zawiera setki lub tysiące elementów.
- **Aplikacji mobilnych:** Gdzie zasoby są ograniczone i każda optymalizacja wydajności jest istotna.
- **Interfejsów użytkownika z dużą ilością dynamicznych danych:** Takich jak portale społecznościowe, aplikacje e-commerce czy dashboardy.

Wady

Mimo licznych zalet, wirtualizacja listy ma również pewne wady:

- **Złożoność implementacji:** Wprowadzenie wirtualizacji może wymagać dodatkowego kodu i konfiguracji, co może zwiększyć złożoność projektu.
- **Problemy z dostępnością:** Renderowanie dynamiczne może wpływać na narzędzia do czytania ekranu i inne technologie wspomagające, co może utrudniać dostępność aplikacji.

Wykorzystanie React Virtuoso przyczyniło się do poprawy wydajności i płynności interfejsu użytkownika aplikacji, co miało kluczowe znaczenie dla zadowolenia użytkowników i jakości doświadczenia użytkownika.

4.6.2 Przetwarzanie języka naturalnego (NLP)

W pracy inżynierskiej zastosowano techniki lematyzacji oraz oznaczania części mowy (POS tagging) w ramach przetwarzania języka naturalnego (NLP). Obie te techniki odegrały kluczową rolę w analizie tekstu i umożliwiły bardziej precyzyjne przetwarzanie danych z napisów, przy zapisywaniu wybranych przez użytkownika słów do nauki, lub przy wyświetlaniu częstości występowania słów w napisach [1].

Lematyzacja

Proces lematyzacji polega na sprowadzaniu różnych form gramatycznych wyrazów do ich podstawowej formy, zwanej lematem. Dzięki temu możliwe jest ujednolicenie wyrazów, które w zależności od kontekstu występują w różnych odmianach gramatycznych. Przykładowo, formy takie jak "chodzę", "chodził" czy "chodziliśmy" są sprowadzane do podstawowej formy "chodzić". Umożliwia to bardziej spójne analizowanie tekstów i wyciąganie wniosków na temat ich zawartości, np. poprzez obliczanie częstotliwości występowania poszczególnych słów [1].

POS Tagging (oznaczanie części mowy)

Drugą techniką było oznaczanie części mowy, czyli przypisywanie każdemu słowu w tekście odpowiedniej etykiety gramatycznej (rzeczownik, czasownik, przymiotnik itd.) [1]. Dzięki temu możliwe było lepsze zrozumienie struktury zdań oraz funkcji słów w kontekście. Oznaczanie części mowy okazało się kluczowe w procesie analizy tekstu, umożliwiając podział słów na kategorie zależne od ich funkcji gramatycznej. W przyszłości może to być przydatne do tworzenia funkcji umożliwiających użytkownikom wybór nauki określonych kategorii słów, takich jak czasowniki czy przymiotniki itd.

Rozdział 5

Podsumowanie

5.1 Wnioski

Projekt aplikacji webowej wspomagającej naukę języków obcych za pomocą filmów i seriali stanowi nowoczesne i efektywne narzędzie edukacyjne. Integracja funkcji nauki i oglądania w jednym miejscu ułatwia użytkownikom proces przyswajania wiedzy, eliminując konieczność korzystania z dodatkowych aplikacji. Aplikacja umożliwia przechowywanie trudnych słów, śledzenie postępów oraz korzystanie z panelu nauki, co czyni ją wszechstronnym narzędziem wspomagającym naukę języków. Ponadto, interaktywne podejście do nauki poprzez kontekstualne użycie języka w filmach i serialach zwiększa zaangażowanie użytkowników i poprawia efektywność nauki. Dzięki temu użytkownicy mogą uczyć się w sposób bardziej naturalny i przyjemny, co sprzyja długotrwałemu zapamiętywaniu nowych informacji.

5.2 Kierunki dalszego rozwoju

Dalszy rozwój aplikacji może obejmować dodanie nowych funkcji, takich jak integracja z innymi platformami streamingowymi, rozwój panelu nauki o dodatkowe metody jak quizy, gry edukacyjne czy interaktywne ćwiczenia. Dodatkowe modele NLP do innych języków również mogą okazać się przydatne, umożliwiając naukę mniej popularnych języków. Ponadto, wprowadzenie funkcji społecznościowych, takich jak fora dyskusyjne czy możliwość współpracy z innymi użytkownikami, mogłoby zwiększyć zaangażowanie i motywację do nauki.

Bibliografia

- [1] **Rada Mihalcea, Hugo Liu i Henry Lieberman**, „NLP (Natural Language Processing) for NLP (Natural Language Programming)”, w *Computational Linguistics and Intelligent Text Processing*, **Alexander Gelbukh**, red., Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, s. 319–330, ISBN: 978-3-540-32206-1.
- [2] **Guillermo Rauch**. „Building Modern Web Applications with Next.js”. Dostęp z dnia 12 grudnia 2024 roku. (), adr.: <https://nextjs.org/docs>.

Spis rysunków

1	Nauka słówek w aplikacji Anki	8
2	Interaktywne napisy na stronie Language Reactor	9
3	Interaktywne napisy Trancy	11
4	Diagram struktury bazy danych	18
5	Strona główna aplikacji	20
6	Widok strony głównej fiszek	21
7	Widok strony napisów	22
8	Widok edycji napisów	23
9	Widok słownika	24
10	Widok statystyk i postępów	25
11	Widok logowania i rejestracji	26
12	Widok oglądania filmów	27
13	Panel nauki fiszek	36

Spis tabel

Listings

1	kod do importowania modeli NLP	32
2	kod do obsługi lematyzacji i pos taggingu	32
3	Przykładowy kod integracji LibretTranslate w Next.js	33
4	Tablica Tanstack do wirtualizacji listy	38