



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. Uczelnii

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

You Tube

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
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	Backend	31
4.3.1	Struktura backendu w Next.js	31
4.3.2	Struktura backendu w Flask	31
4.3.3	LibretTranslate	33
4.3.4	Przetwarzanie języka naturalnego (NLP)	34
4.4	Frontend	34
4.4.1	Struktura projektu	34
4.4.2	Panel nauki Fiszek	36
4.4.3	Wirtualizacja List	37
4.5	Problemy implementacyjne	40
4.5.1	Zacinający się interfejs tablicy napisów	40
4.5.2	Błędy w tłumaczeniach	40
5	Podsumowanie	41
5.1	Wnioski	41
5.2	Kierunki dalszego rozwoju	41
Bibliografia		42
Spis rysunków		45

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 [13].

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 kwesti wyboru czego chcielibyśmy się dokładnie uczyć [7].

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 [12].

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 nauki, takich jak słownik, flashcards, quiz czy lista 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 [6]. 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 zapisania scieżki 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 [3].

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.

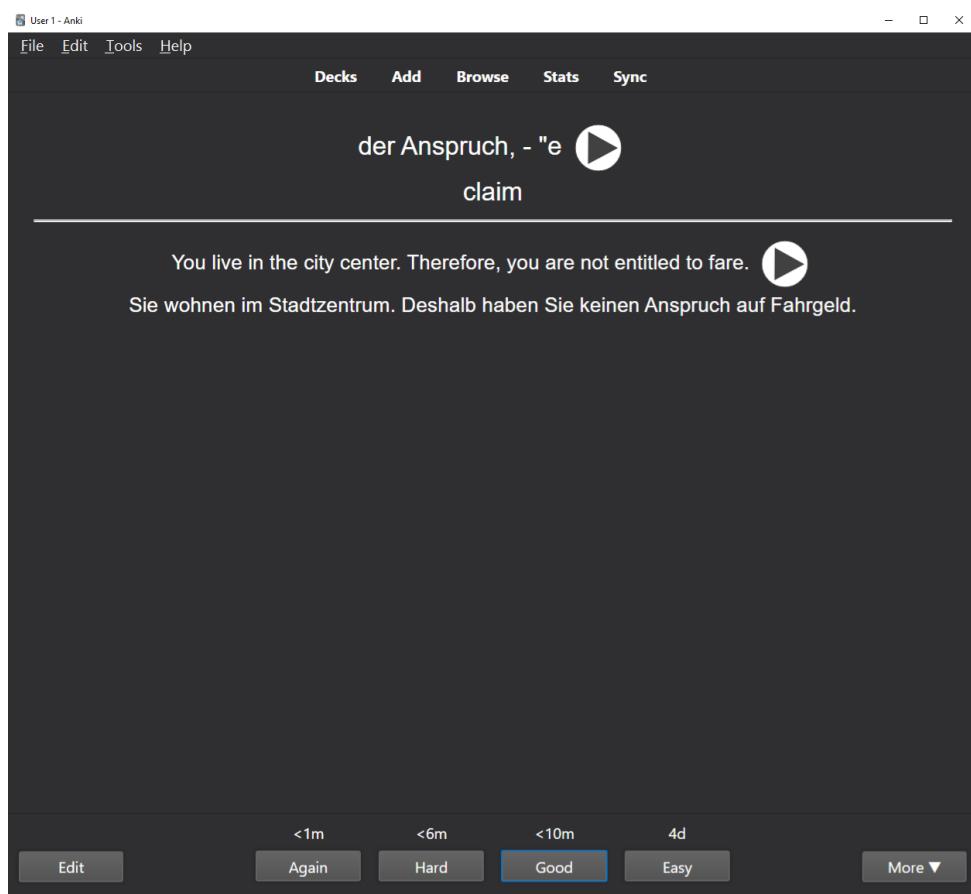
Funkcjonalności:

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

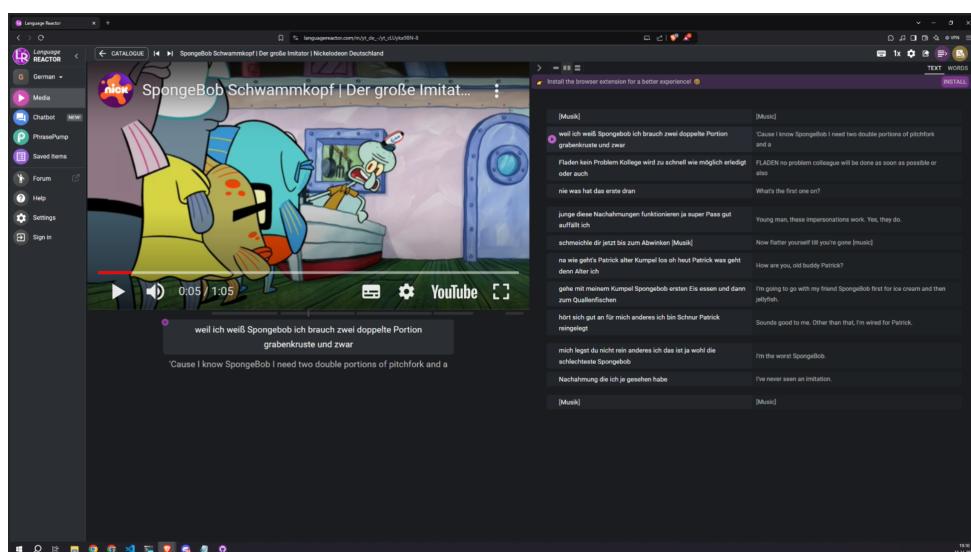
Language Reactor to narzędzie edukacyjne, które umożliwia naukę języków obcych w sposób przyjemny i efektywny poprzez oglądanie filmów i seriali. Aplikacja oferuje interaktywne napisy, które umożliwiają tłumaczenie słów i zwrotów bezpośrednio na ekranie. Dzięki tej funkcji użytkownicy mogą szybko sprawdzić 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.

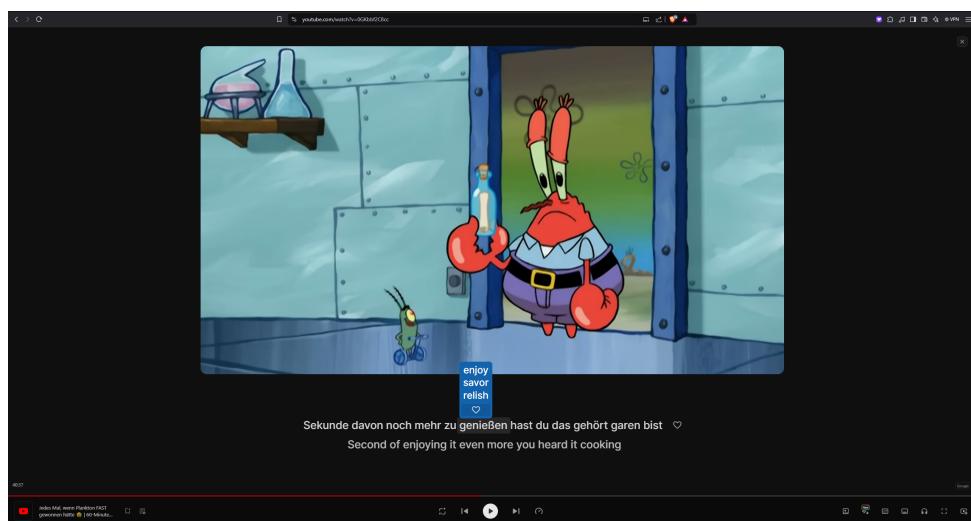
Funkcjonalności:

- **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.

1.2. Przegląd aktualnych rozwiązań



Rysunek 3. Interaktywne napisy na stronie Trancy

Rozdział 2

Cel i zakres pracy

2.1 Cel pracy

Główym 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 [13] [8]. Połączenie nauki z rozrywką, użytkownicy będą mogli uczyć się języka podczas oglądania swoich ulubionych filmów i seriali, co czyni naukę bardziej przyjemną i efektywną. 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 [12]. 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 tekstu 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 kliką 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 kliką przycisk "Zaloguj się".
 - (c) System weryfikuje dane logowania.
 - **Logowanie za pomocą konta Google:**
 - (a) Użytkownik kliką 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 odtwarza wybrany 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 lub quzu

• **Scenariusz:**

– Użytkownik otwiera wybraną stronę nauki.

– **Fiszki:**

1. Użytkownik wybiera zestaw fiszek do nauki.
2. System wyświetla fiszki z wybranego zestawu.
3. Użytkownik przegląda fiszki i próbuje odgadnąć tłumaczenie.
4. Użytkownik sprawdza poprawność tłumaczenia i ocenia swoją odpowiedź.

– **Quiz:**

1. Użytkownik wybiera ilość słówek do quizu.
2. System wyświetla pytania quizowe z różnymi odpowiedziami.
3. Użytkownik odpowiada na pytania quizowe.
4. System sprawdza poprawność odpowiedzi i wyświetla wynik.

– 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 tranzycie.

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 pre-ferowanym środowisku chmurowym. Dodatkowo, Atlas oferuje narzędzia do analizy danych, takie jak MongoDB Charts, które umożliwiają tworzenie interaktywnych wizualizacji danych bezpośrednio z bazy danych.

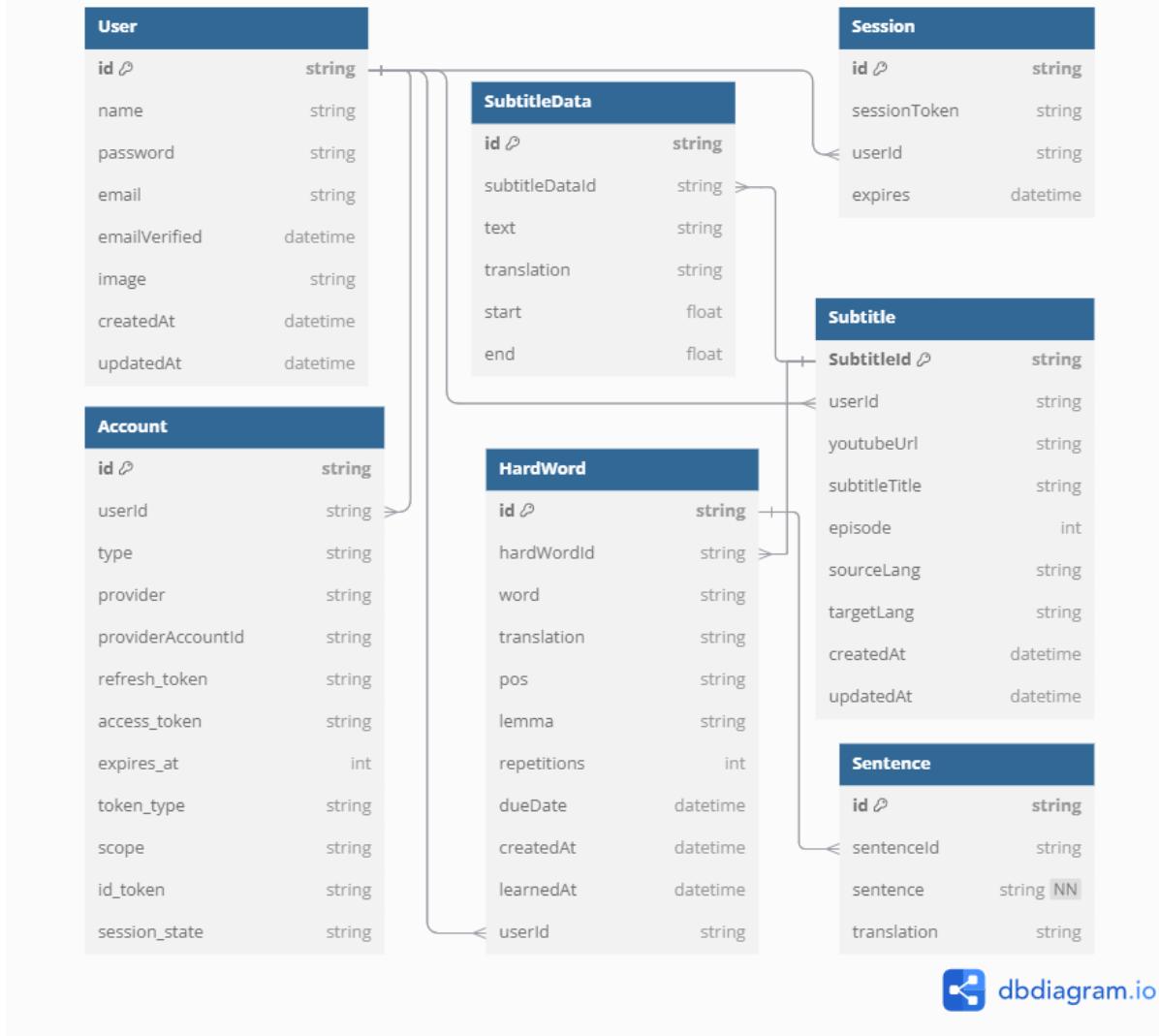
3.3.2 Struktura

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.

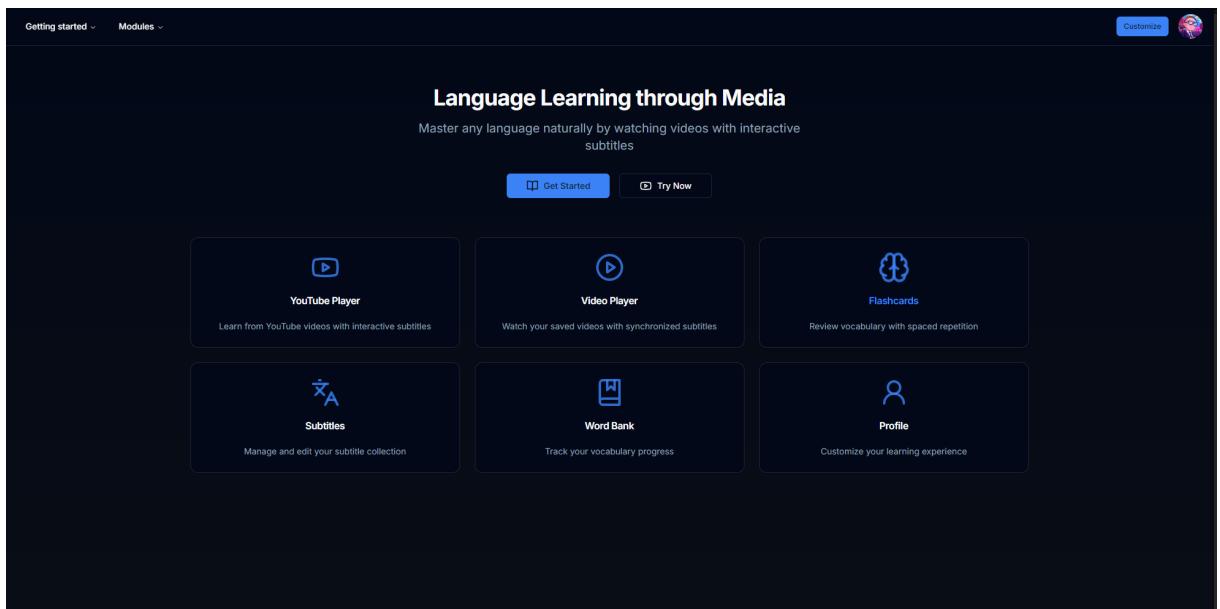
Kolejną metodą zapobiegania redundancji jest stosowanie referencji między kolekcjami. Zamiast przechowywać duplikaty danych, dokumenty w jednej kolekcji mogą zawierać referencje do dokumentów w innych kolekcjach. Na przykład, dokumenty w kolekcji sesji użytkownika mogą zawierać referencje do dokumentów 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 seriale dostępne w aplikacji oraz korzystać z wyszukiwarki, aby znaleźć interesujące ich treści.

3.4.2 Strona Nauki

Flashcard Sets

Getting started Modules Customize

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

26 words due

Set Name	Due	Mastery
Geister der Ark... (6 due)	0%	0 mastered
Was passiert in... (5 due)	0%	0 mastered
Fuji Kaze - Shi... (5 due)	0%	0 mastered
Tauchen in der ... (3 due)	0%	0 mastered
Star.Wars.Epis... (3 due)	0%	0 mastered
Harry.Potter.un... (3 due)	0%	0 mastered
六哲 - 黑竜江語... (1 due)	0%	0 mastered

Words in Subtitles
This chart shows the distribution of hard words across different subtitles.

26 Words

Geister der Arkis Das Geheimnis der Wes passiert in unserem Körper, wenn Fuji Kaze in Shinonoga E-Wa (Not-Kai)

Donut Chart Data:

- Geister der Arkis: 2 words
- Das Geheimnis der: 1 word
- Wes passiert in: 1 word
- unserem Körper, wenn: 1 word
- Shinonoga E-Wa (Not-Kai): 1 word

Rysunek 6. Widok strony głównej fiszek

Getting started Modules Customize

Learned: 2/10

Quiz Streak: 0/3

bleibt
Choose the correct translation

again remain

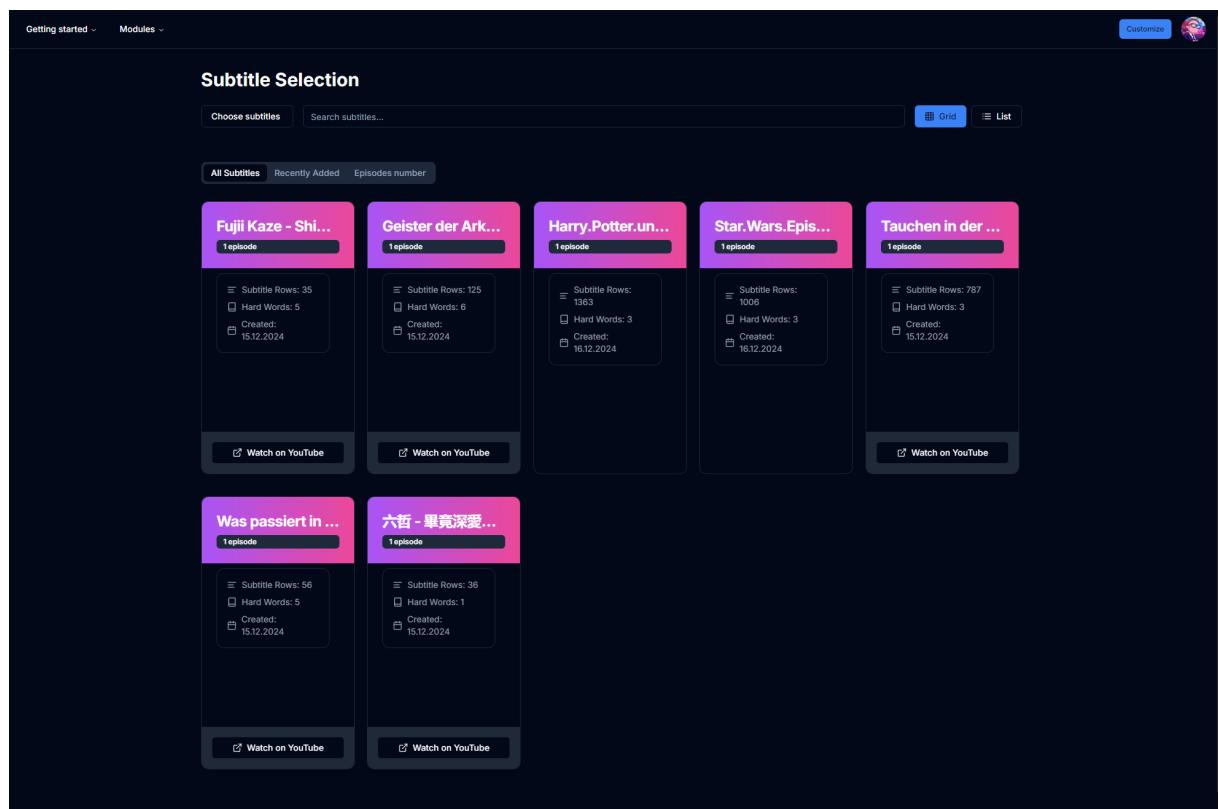
I understand your quest inaccessible

Word progress 0/3

Rysunek 7. Widok quizu

Strona Nauki to główne miejsce, w którym użytkownicy mogą korzystać z panelu nauki fiszek lub quizu. Panel fiszek zawiera informacje na temat stanu słów które są nowe, w trakcie nauki, lub skończone. Strona quizu umożliwia użytkownikom naukę poprzez quizy, w których użytkownicy muszą wybrać poprawne tłumaczenie słowa. Użytkownicy muszą poprawnie odpowiedzieć 3 razy z rzędu by słowo zostało uznane za nauczone i przeszło do kolejnego etapu którym jest powtórka po upływie odpowiedniego czasu [9].

3.4.3 Strona Napisów



Rysunek 8. 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.

3.4. Interfejs użytkownika

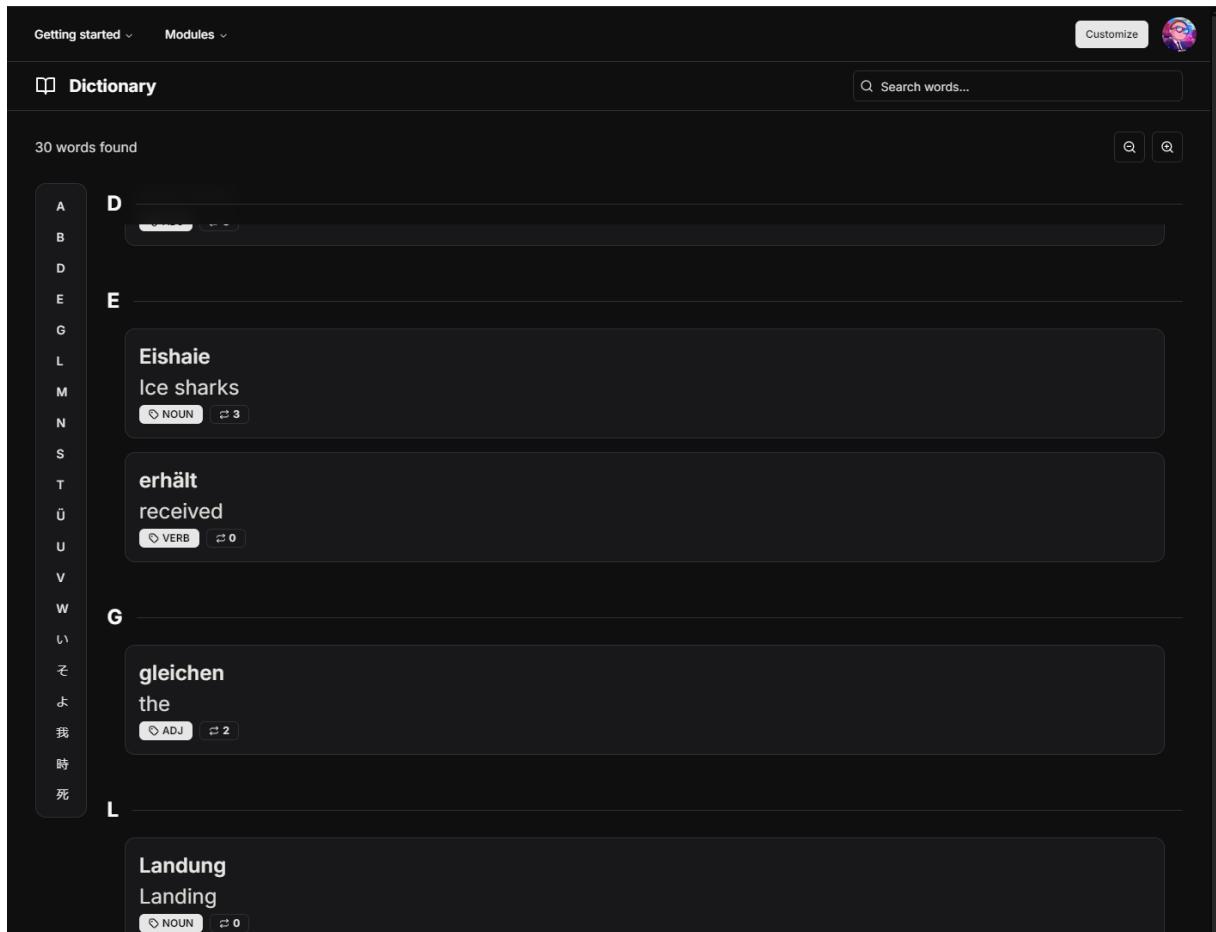
The screenshot shows a user interface for editing subtitles. At the top, there are navigation links for 'Getting started' and 'Modules', and a 'Customize' button with a profile icon. Below that, the title 'Was passiert in unserem Körper, wenn wir tief tauchen' is displayed, along with a note 'No episode information'. A 'Video URL:' field contains the link <https://youtu.be/l7SQAn28sGA>. Below the video URL are several buttons: 'Translate', 'Swap Translation', 'Most Used Words', 'Edit' (which is highlighted in blue), and 'Delete'. A 'Go Back' button is also present.

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-trial 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 dreunddreißig Meter tief und wir wollen schauen, wie fit du 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 dreunddreißig Meter.	Thirty-nine seconds. Thirty-nine seconds over water and now it goes to thirty-nine meters.	00:01:33
17	Dreunddreißig 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 9. 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.

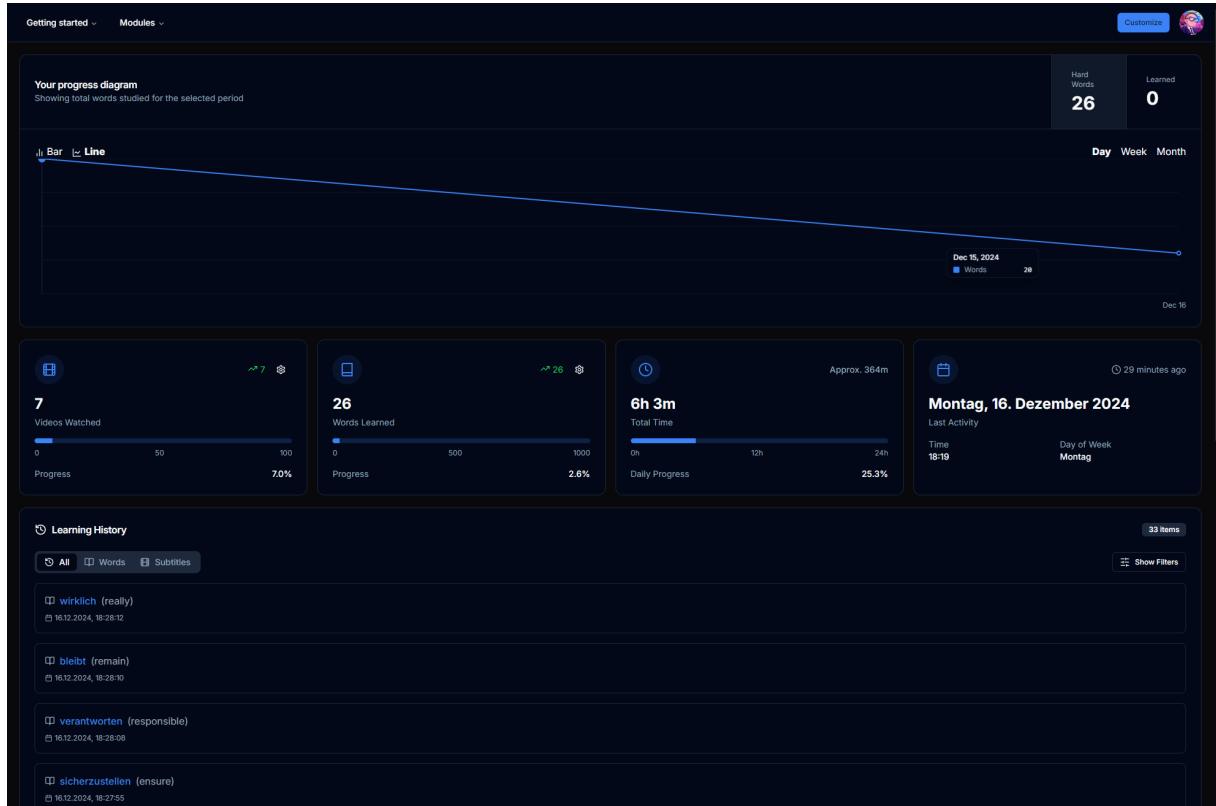
3.4.4 Strona Słownika



Rysunek 10. 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.

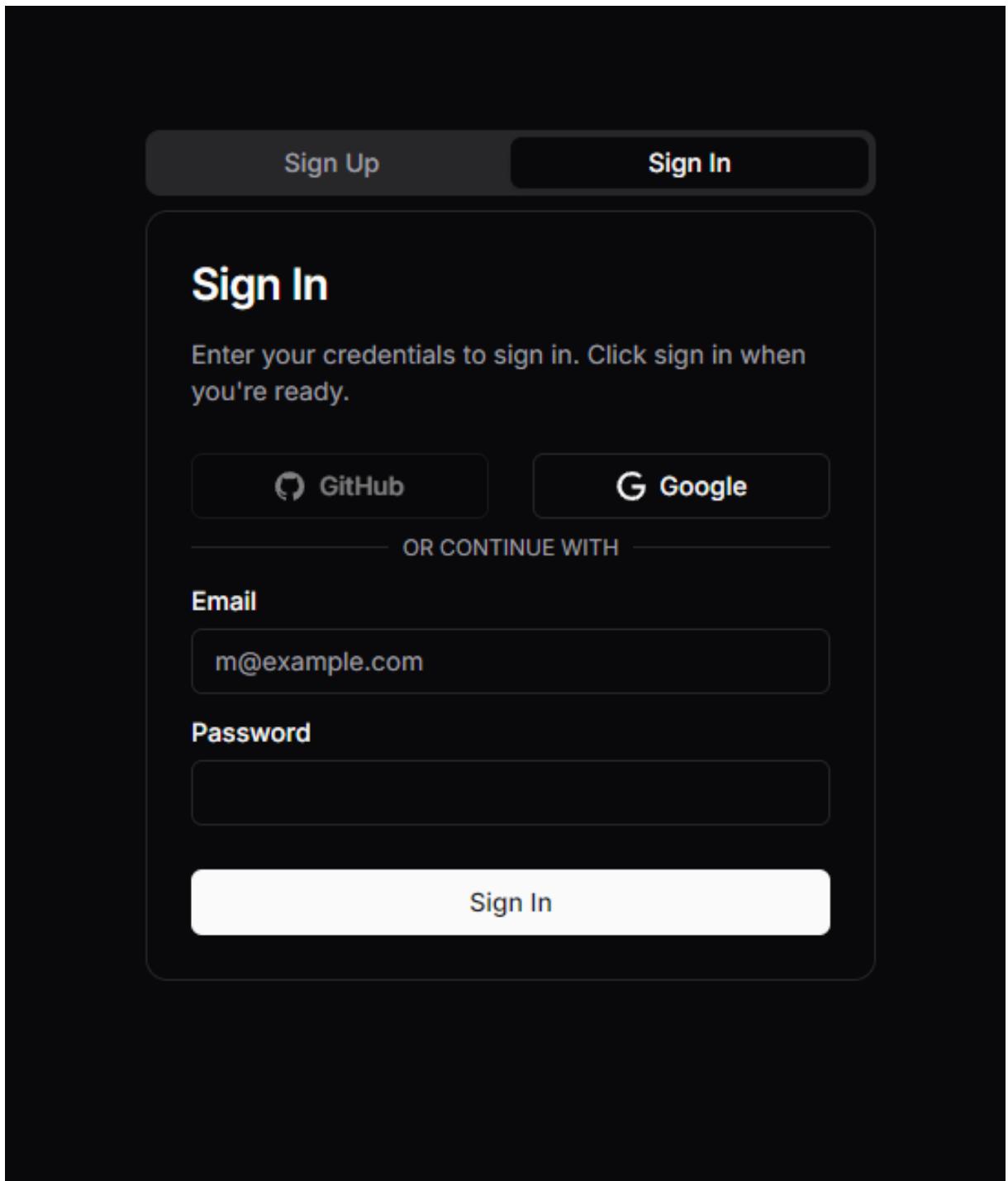
3.4.5 Strona Statystyk



Rysunek 11. 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

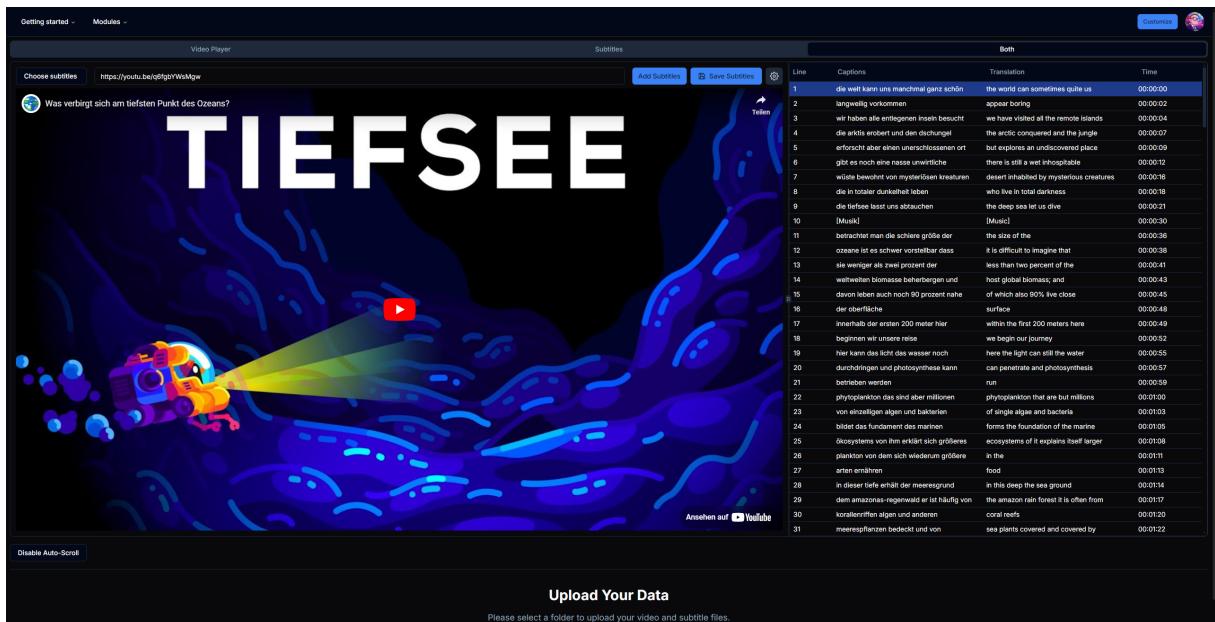


Rysunek 12. Widok logowania i rejestracji

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 13. 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.

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 kompliuje 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 [11].

- **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 [11]. 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 podkreślają jeśli będziemy próbować błędnie używać naszych funkcji lub zmiennych. Wszystkie te cechy czynią Next.js idealnym wyborem do stworzenia nowoczesnej i wydajnej aplikacji językowej [3].

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 sentymencu 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 [2].

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 objektową 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 [4].

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.

4.3 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) [3]. 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 [2].

4.3.1 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.

4.3.2 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) [5] [1]. 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 lemmatyzacja 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 lemmatyzacji i pos taggingu

```

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

```

Funkcja `analyze_text` przyjmuje żądanie zawierające słowo oraz język źródłowy, a następnie przetwarza tekst przy użyciu odpowiedniego modelu językowego. W przypadku braku określenia języka źródłowego lub podania nieobsługiwanej 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.

4.3.3 LibretTranslate

LibretTranslate to otwartoźródłowy system tłumaczenia maszynowego, który umożliwia tłumaczenie tekstuów pomiędzy różnymi językami. W projekcie został wykorzystany do tłumaczenia tekstuów 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[],
4     targetLang: string) {
5         try {
6             const texts = subtitleData.map(subtitle => subtitle?.text);
7             const response = await axios.post("http://127.0.0.1:5000/translate"
8         , {
9             q: texts,
10            source: "auto" ,
11            target: targetLang || "en",
12            format: "text"
13        });
14        let detectedLanguage = "auto";
15        if(response.data.detectedLanguage[0].language){
16            detectedLanguage = response.data.detectedLanguage[0].language;
17        }
18        return {
19            translatedSubtitleData: response.data.translatedText ,
20            detectedLanguage
21        };
22    } catch (error) {
23        console.error('Error translating subtitles:', error);
24        throw new Error('Failed to translate subtitles');
25    }
26}
27

```

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.

4.3.4 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 [10].

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 [10].

POS Tagging (oznaczanie części mowy)

Drugą techniką było oznaczanie części mowy, czyli przypisywanie każdemu słowa w tekście odpowiedniej etykiety gramatycznej (rzeczownik, czasownik, przymiotnik itd.) [10]. 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.

4.4 Frontend

4.4.1 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:

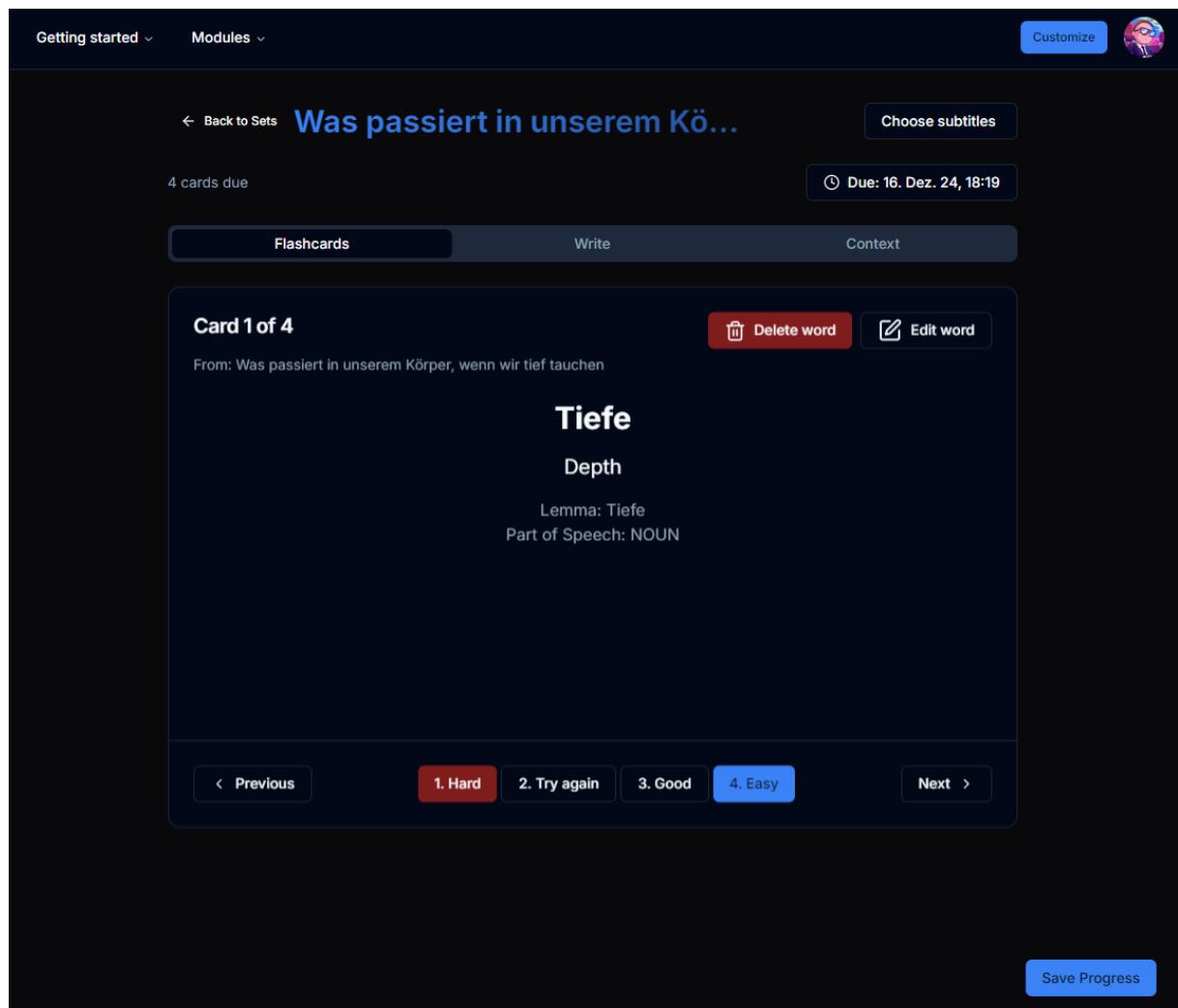
- **app:** Zawiera wszystkie strony aplikacji, które są renderowane przez Next.js. Każda strona jest reprezentowana przez plik JavaScript, który eksportuje komponent React.

- **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 autoryzacji.
- **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 kontrole 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 providerami, 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 działając 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.

4.4.2 Panel nauki Fiszek



Rysunek 14. 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 [9].

System powtórek SRS

System SRS jest algorymem, 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 [9].

4.4.3 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 płynnie. 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 pary sekund na reakcję interfejsu listy i pokazanie okna dodawania trudnych słów do systemu [14].

Mechanizm działania wirtualizacji listy

- **Obserwacja widocznych elementów:** Komponent śledzi pozycję widoku użytkownika w liście. Renderowane są tylko te elementy, które znajdują 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 }: {  
2    captions: Caption[], height: string }) {  
3      const [sorting, setSorting] = useState<SortingState>([]);  
4      const [currentIndex, setCurrentIndex] = useState<number>(0);  
5      const playedSeconds = useSelector((state: any) =>  
6        state.subtitle.playedSeconds);  
7      const autoScrollEnabled = useSelector((state: any) =>  
8        state.subtitle.autoScrollEnabled);  
9      const tableRef = useRef<TableVirtuosoHandle>(null);  
10     const table = useReactTable({  
11       data: captions,  
12       columns: columns,  
13       state: {  
14         sorting,  
15       },  
16       onSortingChange: setSorting,  
17       getCoreRowModel: getCoreRowModel(),  
18       getSortedRowModel: getSortedRowModel(),  
19     });  
20     const { rows } = table.getRowModel()  
21     useEffect(() => {  
22       if (rows.length > 0) {  
23         let newIndex = -1;  
24         for (let i = 0; i < rows.length; i++) {  
25           const row = rows[i];  
26           const startTime = row.original.start ?? 0;  
27           const nextStartTime = rows[i + 1]?.original.start ??  
28             Infinity;  
29           if (playedSeconds >= startTime && playedSeconds <  
30             nextStartTime) {  
31             newIndex = i;  
32             break;  
33           }  
34         }  
35         if (newIndex === -1) {  
36           newIndex = 0;  
37         }  
38         setCurrentIndex(newIndex);  
39         if (autoScrollEnabled && tableRef.current && newIndex !==  
40           -1) {  
41           tableRef.current.scrollToIndex(newIndex);  
42         }  
43       }  
44     }, [playedSeconds, currentIndex, tableRef]);  
45   }  
46   return {  
47     table,  
48     autoScrollEnabled,  
49     playedSeconds,  
50     currentIndex,  
51     setSorting,  
52     setCurrentIndex,  
53     tableRef,  
54     rows,  
55     captions,  
56     height,  
57     playedSeconds,  
58     autoScrollEnabled,  
59     sorting,  
60     onSortingChange,  
61     getCoreRowModel,  
62     getSortedRowModel,  
63     state,  
64     setSorting,  
65     setCurrentIndex,  
66     tableRef,  
67     rows,  
68     captions,  
69     height,  
70     playedSeconds,  
71     autoScrollEnabled,  
72     sorting,  
73     onSortingChange,  
74     getCoreRowModel,  
75     getSortedRowModel,  
76     state,  
77     setSorting,  
78     setCurrentIndex,  
79     tableRef,  
80     rows,  
81     captions,  
82     height,  
83     playedSeconds,  
84     autoScrollEnabled,  
85     sorting,  
86     onSortingChange,  
87     getCoreRowModel,  
88     getSortedRowModel,  
89     state,  
90     setSorting,  
91     setCurrentIndex,  
92     tableRef,  
93     rows,  
94     captions,  
95     height,  
96     playedSeconds,  
97     autoScrollEnabled,  
98     sorting,  
99     onSortingChange,  
100    getCoreRowModel,  
101    getSortedRowModel,  
102    state,  
103    setSorting,  
104    setCurrentIndex,  
105    tableRef,  
106    rows,  
107    captions,  
108    height,  
109    playedSeconds,  
110    autoScrollEnabled,  
111    sorting,  
112    onSortingChange,  
113    getCoreRowModel,  
114    getSortedRowModel,  
115    state,  
116    setSorting,  
117    setCurrentIndex,  
118    tableRef,  
119    rows,  
120    captions,  
121    height,  
122    playedSeconds,  
123    autoScrollEnabled,  
124    sorting,  
125    onSortingChange,  
126    getCoreRowModel,  
127    getSortedRowModel,  
128    state,  
129    setSorting,  
130    setCurrentIndex,  
131    tableRef,  
132    rows,  
133    captions,  
134    height,  
135    playedSeconds,  
136    autoScrollEnabled,  
137    sorting,  
138    onSortingChange,  
139    getCoreRowModel,  
140    getSortedRowModel,  
141    state,  
142    setSorting,  
143    setCurrentIndex,  
144    tableRef,  
145    rows,  
146    captions,  
147    height,  
148    playedSeconds,  
149    autoScrollEnabled,  
150    sorting,  
151    onSortingChange,  
152    getCoreRowModel,  
153    getSortedRowModel,  
154    state,  
155    setSorting,  
156    setCurrentIndex,  
157    tableRef,  
158    rows,  
159    captions,  
160    height,  
161    playedSeconds,  
162    autoScrollEnabled,  
163    sorting,  
164    onSortingChange,  
165    getCoreRowModel,  
166    getSortedRowModel,  
167    state,  
168    setSorting,  
169    setCurrentIndex,  
170    tableRef,  
171    rows,  
172    captions,  
173    height,  
174    playedSeconds,  
175    autoScrollEnabled,  
176    sorting,  
177    onSortingChange,  
178    getCoreRowModel,  
179    getSortedRowModel,  
180    state,  
181    setSorting,  
182    setCurrentIndex,  
183    tableRef,  
184    rows,  
185    captions,  
186    height,  
187    playedSeconds,  
188    autoScrollEnabled,  
189    sorting,  
190    onSortingChange,  
191    getCoreRowModel,  
192    getSortedRowModel,  
193    state,  
194    setSorting,  
195    setCurrentIndex,  
196    tableRef,  
197    rows,  
198    captions,  
199    height,  
200    playedSeconds,  
201    autoScrollEnabled,  
202    sorting,  
203    onSortingChange,  
204    getCoreRowModel,  
205    getSortedRowModel,  
206    state,  
207    setSorting,  
208    setCurrentIndex,  
209    tableRef,  
210    rows,  
211    captions,  
212    height,  
213    playedSeconds,  
214    autoScrollEnabled,  
215    sorting,  
216    onSortingChange,  
217    getCoreRowModel,  
218    getSortedRowModel,  
219    state,  
220    setSorting,  
221    setCurrentIndex,  
222    tableRef,  
223    rows,  
224    captions,  
225    height,  
226    playedSeconds,  
227    autoScrollEnabled,  
228    sorting,  
229    onSortingChange,  
230    getCoreRowModel,  
231    getSortedRowModel,  
232    state,  
233    setSorting,  
234    setCurrentIndex,  
235    tableRef,  
236    rows,  
237    captions,  
238    height,  
239    playedSeconds,  
240    autoScrollEnabled,  
241    sorting,  
242    onSortingChange,  
243    getCoreRowModel,  
244    getSortedRowModel,  
245    state,  
246    setSorting,  
247    setCurrentIndex,  
248    tableRef,  
249    rows,  
250    captions,  
251    height,  
252    playedSeconds,  
253    autoScrollEnabled,  
254    sorting,  
255    onSortingChange,  
256    getCoreRowModel,  
257    getSortedRowModel,  
258    state,  
259    setSorting,  
260    setCurrentIndex,  
261    tableRef,  
262    rows,  
263    captions,  
264    height,  
265    playedSeconds,  
266    autoScrollEnabled,  
267    sorting,  
268    onSortingChange,  
269    getCoreRowModel,  
270    getSortedRowModel,  
271    state,  
272    setSorting,  
273    setCurrentIndex,  
274    tableRef,  
275    rows,  
276    captions,  
277    height,  
278    playedSeconds,  
279    autoScrollEnabled,  
280    sorting,  
281    onSortingChange,  
282    getCoreRowModel,  
283    getSortedRowModel,  
284    state,  
285    setSorting,  
286    setCurrentIndex,  
287    tableRef,  
288    rows,  
289    captions,  
290    height,  
291    playedSeconds,  
292    autoScrollEnabled,  
293    sorting,  
294    onSortingChange,  
295    getCoreRowModel,  
296    getSortedRowModel,  
297    state,  
298    setSorting,  
299    setCurrentIndex,  
300    tableRef,  
301    rows,  
302    captions,  
303    height,  
304    playedSeconds,  
305    autoScrollEnabled,  
306    sorting,  
307    onSortingChange,  
308    getCoreRowModel,  
309    getSortedRowModel,  
310    state,  
311    setSorting,  
312    setCurrentIndex,  
313    tableRef,  
314    rows,  
315    captions,  
316    height,  
317    playedSeconds,  
318    autoScrollEnabled,  
319    sorting,  
320    onSortingChange,  
321    getCoreRowModel,  
322    getSortedRowModel,  
323    state,  
324    setSorting,  
325    setCurrentIndex,  
326    tableRef,  
327    rows,  
328    captions,  
329    height,  
330    playedSeconds,  
331    autoScrollEnabled,  
332    sorting,  
333    onSortingChange,  
334    getCoreRowModel,  
335    getSortedRowModel,  
336    state,  
337    setSorting,  
338    setCurrentIndex,  
339    tableRef,  
340    rows,  
341    captions,  
342    height,  
343    playedSeconds,  
344    autoScrollEnabled,  
345    sorting,  
346    onSortingChange,  
347    getCoreRowModel,  
348    getSortedRowModel,  
349    state,  
350    setSorting,  
351    setCurrentIndex,  
352    tableRef,  
353    rows,  
354    captions,  
355    height,  
356    playedSeconds,  
357    autoScrollEnabled,  
358    sorting,  
359    onSortingChange,  
360    getCoreRowModel,  
361    getSortedRowModel,  
362    state,  
363    setSorting,  
364    setCurrentIndex,  
365    tableRef,  
366    rows,  
367    captions,  
368    height,  
369    playedSeconds,  
370    autoScrollEnabled,  
371    sorting,  
372    onSortingChange,  
373    getCoreRowModel,  
374    getSortedRowModel,  
375    state,  
376    setSorting,  
377    setCurrentIndex,  
378    tableRef,  
379    rows,  
380    captions,  
381    height,  
382    playedSeconds,  
383    autoScrollEnabled,  
384    sorting,  
385    onSortingChange,  
386    getCoreRowModel,  
387    getSortedRowModel,  
388    state,  
389    setSorting,  
390    setCurrentIndex,  
391    tableRef,  
392    rows,  
393    captions,  
394    height,  
395    playedSeconds,  
396    autoScrollEnabled,  
397    sorting,  
398    onSortingChange,  
399    getCoreRowModel,  
400    getSortedRowModel,  
401    state,  
402    setSorting,  
403    setCurrentIndex,  
404    tableRef,  
405    rows,  
406    captions,  
407    height,  
408    playedSeconds,  
409    autoScrollEnabled,  
410    sorting,  
411    onSortingChange,  
412    getCoreRowModel,  
413    getSortedRowModel,  
414    state,  
415    setSorting,  
416    setCurrentIndex,  
417    tableRef,  
418    rows,  
419    captions,  
420    height,  
421    playedSeconds,  
422    autoScrollEnabled,  
423    sorting,  
424    onSortingChange,  
425    getCoreRowModel,  
426    getSortedRowModel,  
427    state,  
428    setSorting,  
429    setCurrentIndex,  
430    tableRef,  
431    rows,  
432    captions,  
433    height,  
434    playedSeconds,  
435    autoScrollEnabled,  
436    sorting,  
437    onSortingChange,  
438    getCoreRowModel,  
439    getSortedRowModel,  
440    state,  
441    setSorting,  
442    setCurrentIndex,  
443    tableRef,  
444    rows,  
445    captions,  
446    height,  
447    playedSeconds,  
448    autoScrollEnabled,  
449    sorting,  
450    onSortingChange,  
451    getCoreRowModel,  
452    getSortedRowModel,  
453    state,  
454    setSorting,  
455    setCurrentIndex,  
456    tableRef,  
457    rows,  
458    captions,  
459    height,  
460    playedSeconds,  
461    autoScrollEnabled,  
462    sorting,  
463    onSortingChange,  
464    getCoreRowModel,  
465    getSortedRowModel,  
466    state,  
467    setSorting,  
468    setCurrentIndex,  
469    tableRef,  
470    rows,  
471    captions,  
472    height,  
473    playedSeconds,  
474    autoScrollEnabled,  
475    sorting,  
476    onSortingChange,  
477    getCoreRowModel,  
478    getSortedRowModel,  
479    state,  
480    setSorting,  
481    setCurrentIndex,  
482    tableRef,  
483    rows,  
484    captions,  
485    height,  
486    playedSeconds,  
487    autoScrollEnabled,  
488    sorting,  
489    onSortingChange,  
490    getCoreRowModel,  
491    getSortedRowModel,  
492    state,  
493    setSorting,  
494    setCurrentIndex,  
495    tableRef,  
496    rows,  
497    captions,  
498    height,  
499    playedSeconds,  
500    autoScrollEnabled,  
501    sorting,  
502    onSortingChange,  
503    getCoreRowModel,  
504    getSortedRowModel,  
505    state,  
506    setSorting,  
507    setCurrentIndex,  
508    tableRef,  
509    rows,  
510    captions,  
511    height,  
512    playedSeconds,  
513    autoScrollEnabled,  
514    sorting,  
515    onSortingChange,  
516    getCoreRowModel,  
517    getSortedRowModel,  
518    state,  
519    setSorting,  
520    setCurrentIndex,  
521    tableRef,  
522    rows,  
523    captions,  
524    height,  
525    playedSeconds,  
526    autoScrollEnabled,  
527    sorting,  
528    onSortingChange,  
529    getCoreRowModel,  
530    getSortedRowModel,  
531    state,  
532    setSorting,  
533    setCurrentIndex,  
534    tableRef,  
535    rows,  
536    captions,  
537    height,  
538    playedSeconds,  
539    autoScrollEnabled,  
540    sorting,  
541    onSortingChange,  
542    getCoreRowModel,  
543    getSortedRowModel,  
544    state,  
545    setSorting,  
546    setCurrentIndex,  
547    tableRef,  
548    rows,  
549    captions,  
550    height,  
551    playedSeconds,  
552    autoScrollEnabled,  
553    sorting,  
554    onSortingChange,  
555    getCoreRowModel,  
556    getSortedRowModel,  
557    state,  
558    setSorting,  
559    setCurrentIndex,  
560    tableRef,  
561    rows,  
562    captions,  
563    height,  
564    playedSeconds,  
565    autoScrollEnabled,  
566    sorting,  
567    onSortingChange,  
568    getCoreRowModel,  
569    getSortedRowModel,  
570    state,  
571    setSorting,  
572    setCurrentIndex,  
573    tableRef,  
574    rows,  
575    captions,  
576    height,  
577    playedSeconds,  
578    autoScrollEnabled,  
579    sorting,  
580    onSortingChange,  
581    getCoreRowModel,  
582    getSortedRowModel,  
583    state,  
584    setSorting,  
585    setCurrentIndex,  
586    tableRef,  
587    rows,  
588    captions,  
589    height,  
590    playedSeconds,  
591    autoScrollEnabled,  
592    sorting,  
593    onSortingChange,  
594    getCoreRowModel,  
595    getSortedRowModel,  
596    state,  
597    setSorting,  
598    setCurrentIndex,  
599    tableRef,  
600    rows,  
601    captions,  
602    height,  
603    playedSeconds,  
604    autoScrollEnabled,  
605    sorting,  
606    onSortingChange,  
607    getCoreRowModel,  
608    getSortedRowModel,  
609    state,  
610    setSorting,  
611    setCurrentIndex,  
612    tableRef,  
613    rows,  
614    captions,  
615    height,  
616    playedSeconds,  
617    autoScrollEnabled,  
618    sorting,  
619    onSortingChange,  
620    getCoreRowModel,  
621    getSortedRowModel,  
622    state,  
623    setSorting,  
624    setCurrentIndex,  
625    tableRef,  
626    rows,  
627    captions,  
628    height,  
629    playedSeconds,  
630    autoScrollEnabled,  
631    sorting,  
632    onSortingChange,  
633    getCoreRowModel,  
634    getSortedRowModel,  
635    state,  
636    setSorting,  
637    setCurrentIndex,  
638    tableRef,  
639    rows,  
640    captions,  
641    height,  
642    playedSeconds,  
643    autoScrollEnabled,  
644    sorting,  
645    onSortingChange,  
646    getCoreRowModel,  
647    getSortedRowModel,  
648    state,  
649    setSorting,  
650    setCurrentIndex,  
651    tableRef,  
652    rows,  
653    captions,  
654    height,  
655    playedSeconds,  
656    autoScrollEnabled,  
657    sorting,  
658    onSortingChange,  
659    getCoreRowModel,  
660    getSortedRowModel,  
661    state,  
662    setSorting,  
663    setCurrentIndex,  
664    tableRef,  
665    rows,  
666    captions,  
667    height,  
668    playedSeconds,  
669    autoScrollEnabled,  
670    sorting,  
671    onSortingChange,  
672    getCoreRowModel,  
673    getSortedRowModel,  
674    state,  
675    setSorting,  
676    setCurrentIndex,  
677    tableRef,  
678    rows,  
679    captions,  
680    height,  
681    playedSeconds,  
682    autoScrollEnabled,  
683    sorting,  
684    onSortingChange,  
685    getCoreRowModel,  
686    getSortedRowModel,  
687    state,  
688    setSorting,  
689    setCurrentIndex,  
690    tableRef,  
691    rows,  
692    captions,  
693    height,  
694    playedSeconds,  
695    autoScrollEnabled,  
696    sorting,  
697    onSortingChange,  
698    getCoreRowModel,  
699    getSortedRowModel,  
700    state,  
701    setSorting,  
702    setCurrentIndex,  
703    tableRef,  
704    rows,  
705    captions,  
706    height,  
707    playedSeconds,  
708    autoScrollEnabled,  
709    sorting,  
710    onSortingChange,  
711    getCoreRowModel,  
712    getSortedRowModel,  
713    state,  
714    setSorting,  
715    setCurrentIndex,  
716    tableRef,  
717    rows,  
718    captions,  
719    height,  
720    playedSeconds,  
721    autoScrollEnabled,  
722    sorting,  
723    onSortingChange,  
724    getCoreRowModel,  
725    getSortedRowModel,  
726    state,  
727    setSorting,  
728    setCurrentIndex,  
729    tableRef,  
730    rows,  
731    captions,  
732    height,  
733    playedSeconds,  
734    autoScrollEnabled,  
735    sorting,  
736    onSortingChange,  
737    getCoreRowModel,  
738    getSortedRowModel,  
739    state,  
740    setSorting,  
741    setCurrentIndex,  
742    tableRef,  
743    rows,  
744    captions,  
745    height,  
746    playedSeconds,  
747    autoScrollEnabled,  
748    sorting,  
749    onSortingChange,  
750    getCoreRowModel,  
751    getSortedRowModel,  
752    state,  
753    setSorting,  
754    setCurrentIndex,  
755    tableRef,  
756    rows,  
757    captions,  
758    height,  
759    playedSeconds,  
760    autoScrollEnabled,  
761    sorting,  
762    onSortingChange,  
763    getCoreRowModel,  
764    getSortedRowModel,  
765    state,  
766    setSorting,  
767    setCurrentIndex,  
768    tableRef,  
769    rows,  
770    captions,  
771    height,  
772    playedSeconds,  
773    autoScrollEnabled,  
774    sorting,  
775    onSortingChange,  
776    getCoreRowModel,  
777    getSortedRowModel,  
778    state,  
779    setSorting,  
780    setCurrentIndex,  
781    tableRef,  
782    rows,  
783    captions,  
784    height,  
785    playedSeconds,  
786    autoScrollEnabled,  
787    sorting,  
788    onSortingChange,  
789    getCoreRowModel,  
790    getSortedRowModel,  
791    state,  
792    setSorting,  
793    setCurrentIndex,  
794    tableRef,  
795    rows,  
796    captions,  
797    height,  
798    playedSeconds,  
799    autoScrollEnabled,  
800    sorting,  
801    onSortingChange,  
802    getCoreRowModel,  
803    getSortedRowModel,  
804    state,  
805    setSorting,  
806    setCurrentIndex,  
807    tableRef,  
808    rows,  
809    captions,  
810    height,  
811    playedSeconds,  
812    autoScrollEnabled,  
813    sorting,  
814    onSortingChange,  
815    getCoreRowModel,  
816    getSortedRowModel,  
817    state,  
818    setSorting,  
819    setCurrentIndex,  
820    tableRef,  
821    rows,  
822    captions,  
823    height,  
824    playedSeconds,  
825    autoScrollEnabled,  
826    sorting,  
827    onSortingChange,  
828    getCoreRowModel,  
829    getSortedRowModel,  
830    state,  
831    setSorting,  
832    setCurrentIndex,  
833    tableRef,  
834    rows,  
835    captions,  
836    height,  
837    playedSeconds,  
838    autoScrollEnabled,  
839    sorting,  
840    onSortingChange,  
841    getCoreRowModel,  
842    getSortedRowModel,  
843    state,  
844    setSorting,  
845    setCurrentIndex,  
846    tableRef,  
847    rows,  
848    captions,  
849    height,  
850    playedSeconds,  
851    autoScrollEnabled,  
852    sorting,  
853    onSortingChange,  
854    getCoreRowModel,  
855    getSortedRowModel,  
856    state,  
857    setSorting,  
858    setCurrentIndex,  
859    tableRef,  
860    rows,  
861    captions,  
862    height,  
863    playedSeconds,  
864    autoScrollEnabled,  
865    sorting,  
866    onSortingChange,  
867    getCoreRowModel,  
868    getSortedRowModel,  
869    state,  
870    setSorting,  
871    setCurrentIndex,  
872    tableRef,  
873    rows,  
874    captions,  
875    height,  
876    playedSeconds,  
877    autoScrollEnabled,  
878    sorting,  
879    onSortingChange,  
880    getCoreRowModel,  
881    getSortedRowModel,  
882    state,  
883    setSorting,  
884    setCurrentIndex,  
885    tableRef,  
886    rows,  
887    captions,  
888    height,  
889    playedSeconds,  
890    autoScrollEnabled,  
891    sorting,  
892    onSortingChange,  
893    getCoreRowModel,  
894    getSortedRowModel,  
895    state,  
896    setSorting,  
897    setCurrentIndex,  
898    tableRef,  
899    rows,  
900    captions,  
901    height,  
902    playedSeconds,  
903    autoScrollEnabled,  
904    sorting,  
905    onSortingChange,  
906    getCoreRowModel,  
907    getSortedRowModel,  
908    state,  
909    setSorting,  
910    setCurrentIndex,  
911    tableRef,  
912    rows,  
913    captions,  
914    height,  
915    playedSeconds,  
916    autoScrollEnabled,  
917    sorting,  
918    onSortingChange,  
919    getCoreRowModel,  
920    getSortedRowModel,  
921    state,  
922    setSorting,  
923    setCurrentIndex,  
924    tableRef,  
925    rows,  
926    captions,  
927    height,  
928    playedSeconds,  
929    autoScrollEnabled,  
930    sorting,  
931    onSortingChange,  
932    getCoreRowModel,  
933    getSortedRowModel,  
934    state,  
935    setSorting,  
936    setCurrentIndex,  
937    tableRef,  
938    rows,  
939    captions,  
940    height,  
941    playedSeconds,  
942    autoScrollEnabled,  
943    sorting,  
944    onSortingChange,  
945    getCoreRowModel,  
946    getSortedRowModel,  
947    state,  
948    setSorting,  
949    setCurrentIndex,  
950    tableRef,  
951    rows,  
952    captions,  
953    height,  
954    playedSeconds,  
955    autoScrollEnabled,  
956    sorting,  
957    onSortingChange,  
958    getCoreRowModel,  
959    getSortedRowModel,  
960    state,  
961    setSorting,  
962    setCurrentIndex,  
963    tableRef,  
964    rows,  
965    captions,  
966    height,  
967    playedSeconds,  
968    autoScrollEnabled,  
969    sorting,  
970    onSortingChange,  
971    getCoreRowModel,  
972    getSortedRowModel,  
973    state,  
974    setSorting,  
975    setCurrentIndex,  
976    tableRef,  
977    rows,  
978    captions,  
979    height,  
980    playedSeconds,  
981    autoScrollEnabled,  
982    sorting,  
983    onSortingChange,  
984    getCoreRowModel,  
985    getSortedRowModel,  
986    state,  
987    setSorting,  
988    setCurrentIndex,  
989    tableRef,  
990    rows,  
991    captions,  
992    height,  
993    playedSeconds,  
994    autoScrollEnabled,  
995    sorting,  
996    onSortingChange,  
997    getCoreRowModel,  
998    getSortedRowModel,  
999    state,  
1000   setSorting,  
1001   setCurrentIndex,  
1002   tableRef,  
1003   rows,  
1004   captions,  
1005   height,  
1006   playedSeconds,  
1007   autoScrollEnabled,  
1008   sorting,  
1009   onSortingChange,  
1010   getCoreRowModel,  
1011   getSortedRowModel,  
1012   state,  
1013   setSorting,  
1014   setCurrentIndex,  
1015   tableRef,  
1016   rows,  
1017   captions,  
1018   height,  
1019   playedSeconds,  
1020   autoScrollEnabled,  
1021   sorting,  
1022   onSortingChange,  
1023   getCoreRowModel,  
1024   getSortedRowModel,  
1025   state,  
1026   setSorting,  
1027   setCurrentIndex,  
1028   tableRef,  
1029   rows,  
1030   captions,  
1031   height,  
1032   playedSeconds,  
1033   autoScrollEnabled,  
1034   sorting,  
1035   onSortingChange,  
1036   getCoreRowModel,  
1037   getSortedRowModel,  
1038   state,  
1039   setSorting,  
1040   setCurrentIndex,  
1041   tableRef,  
1042   rows,  
1043   captions,  
1044   height,  
1045   playedSeconds,  
1046   autoScrollEnabled,  
1047   sorting,  
1048   onSortingChange,  
1049   getCoreRowModel,  
1050   getSortedRowModel,  
1051   state,  
1052   setSorting,  
1053   setCurrentIndex,  
1054   tableRef,  
1055   rows,  
1056   captions,  
1057   height,  
1058   playedSeconds,  
1059   autoScrollEnabled,  
1060   sorting,  
1061   onSortingChange,  
1062   getCoreRowModel,  
1063   getSortedRowModel,  
1064   state,  
1065   setSorting,  
1066   setCurrentIndex,  
1067   tableRef,  
1068   rows,  
1069   captions,  
1070   height,  
1071   playedSeconds,  
1072   autoScrollEnabled,  
1073   sorting,  
1074   onSortingChange,  
1075   getCoreRowModel,  
1076   getSortedRowModel,  
1077   state,  
1078   setSorting,  
1079   setCurrentIndex,  
1080   tableRef,  
1081   rows,  
1082   captions,  
1083   height,  
1084   playedSeconds,  
1085   autoScrollEnabled,  
1086   sorting,  
1087   onSortingChange,  
1088   getCoreRowModel,  
1089   getSortedRowModel,  
1090   state,  
1091   setSorting,  
1092   setCurrentIndex,  
1093   tableRef,  
1094   rows,  
1095   captions,  
1096   height,  
1097   playedSeconds,  
1098   autoScrollEnabled,  
1099   sorting,  
1100   onSortingChange,  
1101   getCoreRowModel,  
1102   getSortedRowModel,  
1103   state,  
1104   setSorting,  
1105   setCurrentIndex,  
1106   tableRef,  
1107   rows,  
1108   captions,  
1109   height,  
1110   playedSeconds,  
1111   autoScrollEnabled,  
1112   sorting,  
1113   onSortingChange,  
1114   getCoreRowModel,  
1115   getSortedRowModel,  
1116   state,  
1117   setSorting,  
1118   setCurrentIndex,  
1119   tableRef,  
1120   rows,  
1121   captions,  
1122   height,  
1123   playedSeconds,  
1124   autoScrollEnabled,  
1125   sorting,  
1126   onSortingChange,  
1127   getCoreRowModel,  
1128   getSortedRowModel,  
1129   state,  
1130   setSorting,  
1131   setCurrentIndex,  
1132   tableRef,  
1133   rows,  
1134   captions,  
1135   height,  
1136   playedSeconds,  
1137   autoScrollEnabled,  
1138   sorting,  
1139   onSortingChange,  
1140   getCoreRowModel,  
1141   getSortedRowModel,  
1142   state,  
1143   setSorting,  
1144   setCurrentIndex,  
1145   tableRef,  
1146   rows,  
1147   captions,  
1148   height,  
1149   playedSeconds,  
1150   autoScrollEnabled,  
1151   sorting,  
1152   onSortingChange,  
1153   getCoreRowModel,  
1154   getSortedRowModel,  
1155   state,  
1156   setSorting,  
1157   setCurrentIndex,  
1158   tableRef,  
1159   rows,  
1160   captions,  
1161   height,  
1162   playedSeconds,  
1163   autoScrollEnabled,  
1164   sorting,  
1165   onSortingChange,  
1166   getCoreRowModel,  
1167   getSortedRowModel,  
1168   state,  
1169   setSorting,  
1170   setCurrentIndex,  
1171   tableRef,  
1172   rows,  
1173   captions,  
1174   height,  
1175   playedSeconds,  
1176   autoScrollEnabled,  
1177   sorting,  
1178   onSortingChange,  
1179   getCoreRowModel,  
1180   getSortedRowModel,  
1181   state,  
1182   setSorting,  
1183   setCurrentIndex,  
1184   tableRef,  
1185   rows,  
1186   captions,  
1187   height,  
1188   playedSeconds,  
1189   autoScrollEnabled,  
1190   sorting,  
1191   onSortingChange,  
1192   getCoreRowModel,  
1193   getSortedRowModel,  
1194   state,  
1195   setSorting,  
1196   setCurrentIndex,  
1197   tableRef,  
1198   rows,  
1199   captions,  
1200   height,  
1201   playedSeconds,  
1202   autoScrollEnabled,  
1203   sorting,  
1204   onSortingChange,  
1205   getCoreRowModel,  
1206   getSortedRowModel,  
1207   state,  
1208   setSorting,  
1209   setCurrentIndex,  
1210   tableRef,  
1211   rows,  
1212   captions,  
1213   height,  
1214   playedSeconds,  
1215   autoScrollEnabled,  
1216   sorting,  
1217   onSortingChange,  
1218   getCoreRowModel,  
1219   getSortedRowModel,  
1220   state,  
1221   setSorting,  
1222   setCurrentIndex,  
1223   tableRef,  
1224   rows,  
1225   captions,  
1226   height,  
1227   playedSeconds,  
1228   autoScrollEnabled,  
1229   sorting,  
1230   onSortingChange,  
1231   getCoreRowModel,  
1232   getSortedRowModel,  
1233   state,  
1234   setSorting,  
1235   setCurrentIndex,  
1236   tableRef,  
1237   rows,  
1238   captions,  
1239   height,  
1240   playedSeconds,  
1241   autoScrollEnabled,  
1242   sorting,  
1243   onSortingChange,  
1244   getCoreRowModel,  
1245   getSortedRowModel,  
1246   state,  
1247   setSorting,  
1248   setCurrentIndex,  
1249   tableRef,  
1250   rows,  
1251   captions,  
1252   height,  
1253   playedSeconds,  
1254   autoScrollEnabled,  
1255   sorting,  
1256   onSortingChange,  
1257   getCoreRowModel,  
1258   getSortedRowModel,  
1259   state,  
1260   setSorting,  
1261   setCurrentIndex,  
1262   tableRef,  
1263   rows,  
1264   captions,  
1265   height,  
1266   playedSeconds,  
1267   autoScrollEnabled,  
1268   sorting,  
1269   onSortingChange,  
1270   getCoreRowModel,  
1271   getSortedRowModel,  
1272   state,  
1273   setSorting,  
1274   setCurrentIndex,  
1275   tableRef,  
1276   rows,  
1277   captions,  
1278   height,  
1279   playedSeconds,  
1280   autoScrollEnabled,  
1281   sorting,  
1282   onSortingChange,  
1283   getCoreRowModel,  
1284   getSortedRowModel,  
1285   state,  
1286   setSorting,  
1287   setCurrentIndex,  
1288   tableRef,  
1289   rows,  
1290   captions,  
1291   height,  
1292   playedSeconds,  
1293   autoScrollEnabled,  
1294   sorting,  
1295   onSortingChange,  
1296   getCoreRowModel,  
1297   getSortedRowModel,  
1298   state,  
1299   setSorting,  
1300   setCurrentIndex,  
1301   tableRef,  
1302   rows,  
1303   captions,  
1304   height,  
1305   playedSeconds,  
1306   autoScrollEnabled,  
1307   sorting,  
1308   onSortingChange,  
1309   getCoreRowModel,  
1310   getSortedRowModel,  
1311   state,  
1312   setSorting,  
1313   setCurrentIndex,<
```

```

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.

4.5 Problemy implementacyjne

4.5.1 Zacinający się interfejs tablicy napisów

Podczas implementacji napotkano problem zacinającego się interfejsu tablicy napisów. Problem ten wynika z braku wirtualizacji listy, która polega na renderowaniu jedynie tych elementów, które są aktualnie widoczne na ekranie, zamiast renderowania całej listy na raz. Brak tej optymalizacji powoduje, że interfejs staje się mniej responsywny, zwłaszcza gdy lista zawiera dużą liczbę elementów.

W celu rozwiązania tego problemu, konieczne jest zaimplementowanie mechanizmu wirtualizacji, który dynamicznie renderuje i usuwa elementy listy w miarę przewijania. Dzięki temu interfejs będzie działał płynnie, a użytkownik nie będzie doświadczał opóźnień ani zacięć.

4.5.2 Błędy w tłumaczeniach

Podczas testowania aplikacji napotkano problemy z jakością tłumaczeń maszynowych. W niektórych przypadkach tłumaczenia były niepoprawne lub niezrozumiałe, co utrudniało korzystanie z aplikacji. Problem ten wynika z ograniczeń algorytmów tłumaczenia maszynowego, które nie zawsze są w stanie zapewnić dokładne i precyzyjne tłumaczenia. Jedyne rozwiązanie tego problemu to danie użytkownikowi możliwości ręcznego edytowania tłumaczeń, co pozwoli na poprawienie błędów i dostosowanie tłumaczeń do własnych potrzeb.

Rozdział 5

Podsumowanie

5.1 Wnioski

Projekt aplikacji webowej wspomagającej naukę języków obcych za pomocą filmów i seriali stanowi nowoczesne 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.

Celem pracy inżynierskiej było stworzenie aplikacji, która pozwoli użytkownikom na naukę języka w sposób bardziej naturalny, efektywny i dostosowany do indywidualnych potrzeb. Cel ten został zrealizowany poprzez zaprojektowanie systemu, który integruje naukę języka z oglądaniem treści video, co czyni proces edukacyjny bardziej dynamicznym i przyjaznym dla użytkownika.

5.2 Kierunki dalszego rozwoju

Dalszy rozwój aplikacji powinien obejmować rozbudowę jej funkcjonalności oraz poszerzenie zakresu dostępnych możliwości. W pierwszej kolejności wskazane jest wprowadzenie integracji z popularnymi platformami streamingowymi, co pozwoli na korzystanie z różnorodnych materiałów video bez konieczności dodatkowej konfiguracji.

Panel nauki może zostać wzbogacony o nowe metody wspierające proces edukacyjny, takie jak gry edukacyjne, interaktywne ćwiczenia czy testy wiedzy. Rozszerzenie aplikacji o obsługę większej ilości języków mogłoby zostać zrealizowane poprzez wdrożenie zaawansowanych modeli przetwarzania języka naturalnego (NLP), które umożliwią naukę mniej popularnych języków.

Wprowadzenie funkcji społecznościowych, takich jak fora dyskusyjne, grupy nauki czy możliwość współpracy między użytkownikami, stanowiłoby kolejny krok w rozwoju aplikacji. Takie rozwiązania mogłyby sprzyjać budowaniu motywacji oraz zwiększać zaangażowanie użytkowników w proces nauki.

Rozszerzenie funkcjonalności aplikacji o mechanizmy personalizacji, takie jak algorytmy rekommendujące treści dostosowane do poziomu i zainteresowań użytkownika, mogłyby znaczco wpływać na efektywność nauczania. Dodanie audio do treści w procesie nauki mogłyby stanowić dodatkowy element wspierający naukę, umożliwiając użytkownikom lepsze zrozumienie wymowy i intonacji w naturalnym kontekście językowym.

Bibliografia

- [1] **ASAE AbuSahyon, Abdelhadi Alzyoud, Omar Alshorman i Basim Al-Absi**, „AI-driven Technology and Chatbots as Tools for Enhancing English Language Learning in the Context of Second Language Acquisition: A Review Study”, *International Journal of Membrane Science and Technology*, t. 10, nr. 1, s. 1209–1223, 2023.
- [2] **Daniel Bugl**, *Modern Full-Stack React Projects: Build, maintain, and deploy modern web apps using MongoDB, Express, React, and Node.js*. Packt Publishing Ltd, 2024.
- [3] **Duy Dinh Bui i Dinh Bui**, „Next. Js for Front-End and Compatible Backend Solutions”, 2023.
- [4] **Zerihun Dinku**, „React. js vs. Next. js”, 2022.
- [5] **Michail St Fountoulakis**, „Evaluating the impact of AI tools on language proficiency and intercultural communication in second language education”, *International Journal of Second and Foreign Language Education*, t. 3, nr. 1, s. 12–26, 2024.
- [6] **María-Blanca Ibáñez, Ángela Di-Serio i Carlos Delgado-Kloos**, „Gamification for Engaging Computer Science Students in Learning Activities: A Case Study”, *IEEE Transactions on Learning Technologies*, t. 7, nr. 3, s. 291–301, 2014. DOI: 10.1109/TLT.2014.2329293.
- [7] **Myong Hee Ko**, „Learner perceptions and preferences of device type in vocabulary learning”, *Multimedia-Assisted Language Learning*, t. 17, nr. 3, s. 37–68, 2014.
- [8] **Geza Kovacs i Robert C Miller**, „Smart subtitles for vocabulary learning”, w *Proceedings of the SIGCHI conference on human factors in computing systems*, 2014, s. 853–862.
- [9] **Richard Mallett, Jessica Hagen-Zanker, Rachel Slater i Maren Duvendack**, „The benefits and challenges of using systematic reviews in international development research”, *Journal of development effectiveness*, t. 4, nr. 3, s. 445–455, 2012.
- [10] **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.
- [11] **Guillermo Rauch**. „Building Modern Web Applications with Next.js”. Dostęp z dnia 12 grudnia 2024 roku. (), adr.: <https://nextjs.org/docs>.
- [12] **Noa Talaván Zanón i in.**, „Using subtitles to enhance foreign language learning”, 2006.

- [13] **Selen Tekalp, Serde Yerlikaya i Sibel Polat**, „The Effects of Subtitles on Language Learning”, *Note*, t. 2, nr. 1, s. 14, 2023.
- [14] **Li Weng, Gagan Agrawal, Umit Catalyurek, T Kur, Sivaramakrishnan Narayanan i Joel Saltz**, „An approach for automatic data virtualization”, w *Proceedings. 13th IEEE International Symposium on High performance Distributed Computing, 2004.*, IEEE, 2004, s. 24–33.

Spis rysunków

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

Listings

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