

Projekt Zespołowy 2 Grupa 4M

Spis treści

- 1 Wprowadzenie.
- 1.1 Cel projektu.
- 1.2 Wstępna wizja projektu.
- 2 Metodologia wytwarzania.
- 3 Analiza wymagań.
- 3.1 Wymagania użytkownika i biznesowe.
- 3.2 Wymagania funkcjonalne i нефункционалне.
- 3.3 Przypadki użycia.
- 3.4 Potwierdzenie zgodności wymagań.
- 4 Architektura
- 4,1 Definicja architektury.
- 4.2 Specyfikacja analityczna i projektowa.
- 5 Dane trwałe.
- 5.1 Model ER
- 5.1 Model logiczny danych.
- 5.2 Przetwarzanie i przechowywanie danych.
- 6 Projekt standardu interfejsu użytkownika
- 7 Specyfikacja testów.
- 8 *Wirtualizacja/konteneryzacja.*
- 9 Bezpieczeństwo.
- 10 Podręcznik użytkownika.
- 11 Podręcznik administratora.
- 12 Podsumowanie.
- 13 Bibliografia.

1 Wprowadzenie

1.1 Cel projektu

Celem projektu jest opracowanie aplikacji umożliwiającej zarządzanie sesjami burzy mózgów. Aplikacja powinna odnotowywać start i koniec sesji, temat sesji, uczestników, listę pomysłów. Aplikacja będzie dokonywała transkrypcji sesji, w trybie strumieniowym, i zapamiętywała ją w rekordzie sesji. Również pożądane jest, aby aplikacja potrafiła dokonać podsumowania ustaleń z sesji (z wykorzystaniem modeli językowych).

Główną motywacją do projektu jest w przyszłości możliwość analizy dynamiki pracy zespołu w sesji kreatywnej i dzięki temu lepsze zarządzanie pracą kreatywną.

1.2 Wstępna wizja projektu

Nasza wizja projektu przewiduje aplikację desktopową z serwerem bazy danych. Użytkownik za jej pomocą po zalogowaniu się będzie mógł przeglądać wszystkie spotkania prywatne jak i grup do których należy uzyskując takie dane jak transkrypcja i podsumowanie spotkania

2 Metodologia wytwarzania

Projekt tworzymy w metodologii zwinnej. Spotkania zespołowe są przeprowadzane 2 razy w tygodniu zdalnie na platformie Discord.

Komunikacja z właścicielem projektu jest realizowana za pomocą platformy Teams i jest stała.

W trakcie tworzenia projektu występują spotkania z mentorami, którzy pomagają nam w pracy w zakresie ich kompetencji.

W połowie czasu przeznaczanego na realizację projektu organizowane jest spotkanie z komisją, która ocenia wykonaną prezentację śródsesjonalną, a w końcowej fazie projektu prezentację finalną.

3 Analiza wymagań

3.1 Wymagania użytkownika i biznesowe

Wymagania biznesowe:

- program ma zajmować się transkrypcją nagrań dźwiękowych i na tej podstawie generować podsumowania. Dane nagrań mają być zapisane w bazie danych dostępnej dla klientów.
- ma być możliwe tworzenie podsumowań w języku polskim
- baza danych powinna zawierać transkrypcję oraz podsumowanie
- aplikacja powinna nawiązywać połączenie z mikrofonem

Wymagania użytkowe:

- baza danych powinna umożliwiać dostęp do danych wyłącznie osobom, które uczestniczyły w spotkaniu
- aplikacja powinna umożliwiać logowanie się
- klienci powinni móc wykonywać zapytania na bazie danych
- klienci powinni móc dołączać do spotkania

3.2 Wymagania funkcjonalne i нефunkcjonalne

Wymagania funkcjonalne:

System:

- prowadzi konta użytkowników
- tworzy transkrypcji w czasie rzeczywistym
- umie rozpoznawać użytkowników po ich głosie czytaj kto teraz mówi
- może wygenerować podsumowanie spotkania
- pozwala użytkownikom tworzyć nowe spotkania z zaplanowaną datą i godziną oraz listą uczestników jako całe grupy lub pojedyncze osoby
- pozwala na przeglądanie informacji o zakończonych spotkaniach do których użytkownik miał dostęp
- przechowuje informacje o użytkownikach i spotkaniach w bazie danych

Wymagania нефunkcjonalne:

- w spotkaniu może uczestniczyć maksymalnie 30 osób.
- jednocześnie może się odbywać nieskończenie wiele spotkań, gdyż każde spotkanie odbywa się na urządzeniu klienta, a dane ze spotkania są umieszczane w bazie danych, która umożliwia jednoczesną aktualizację wyników.
- dane są przechowywane bezpiecznie
- aplikacja jest darmowa, ale po wyczerpaniu \$200 darmowych kredytów na platformie Deepgram należy utworzyć nowe konto, aby mieć ponowne \$200 darmowych kredytów.

Historyjki użytkowników:

- **Uczestnik:**
 - Jako uczestnik spotkania chcę móc się zalogować do aplikacji w łatwy sposób, podając nazwę użytkownika i hasło
 - Jako uczestnik chce móc przeglądać zaplanowane spotkania w przejrzysty sposób w żeby wiedzieć kiedy ono będzie i kto inny na nim będzie
 - Spotkania mają być pokazywane w postaci kalendarza dni, gdzie same spotkania mają być reprezentowane w postaci kafelków z datą i tematem spotkania
 - Jako Uczestnik chce móc przeglądać ukończone spotkania aby wiedzieć kiedy był obecny na nim oraz co było mówione i poznać jego krótkie podsumowanie
 - Chcę, aby przegląd ukończonego spotkania był dostępny maksymalnie 10 minut po zakończeniu spotkania
- **Moderator**
 - Jako Moderator chce móc tworzyć spotkania i edytować ich szczegóły aby zarządzać nimi zgodnie z moimi wymaganiami
 - Nie chcę, aby do spotkania dołączył się użytkownik, który nie jest do niego zaproszony
 - Jako Moderator muszę móc rozpoczynać i kończyć spotkania
 - Nie chcę aby ktoś z użytkowników mógł rozpocząć spotkanie przedwcześnie ani go przedwcześnie skończyć
- **Administrator**
 - Jako Administrator chce móc tworzyć i edytować grupy aby zapewnić organizację uczestników
 - Chcę, aby łatwiej było moderatorowi dodawać grupy do spotkań, zamiast dodawać osobę po osobie wystarczy, że doda grupę do spotkania
 - Jako administrator chce móc tworzyć konta i nadawać im odpowiednie rolę aby zapewnić odpowiedni kształt organizacji
 - Chcę, aby odpowiednie osoby mogły mieć stworzone konta o wyższych uprawnieniach, nie mogą o tym sami decydować

biznesowe przypadki użycia:

PB1 scenariusz główny utworzenie spotkania:

1.użytkownik loguje się

2.użytkownik tworzy spotkanie z datą i grupami lub użytkownikami mającymi brać udział w spotkaniu

Scenariusz udany: Spotkanie zostało utworzone i użytkownicy widzą na swoim koncie informacje o spotkaniu.

Scenariusz nieudany: Nie udało się utworzyć spotkania lub użytkownicy nie otrzymali informacji o spotkaniu lub otrzymali niepełne informacje np. brak daty spotkania

PB2 przeglądanie konta: scenariusz główny:

1 użytkownik loguje się

2 użytkownik wybiera grupę w obrębie której chce obejrzeć spotkania lub wchodzi w specjalną grupę inne czyli tam gdzie ma dostęp jako luźny użytkownik

3 wybiera konkretne spotkanie i przegląda jego szczegóły

Scenariusz nieudany: Nawet jak użytkownik brał udział w spotkaniu to nie ma dostępu do szczegółów tego spotkania.

PB3 korzystanie ze spotkania: scenariusz główny:

1 użytkownik loguje się

2 użytkownik rozpoczyna spotkanie

3 przebywa na spotkaniu

4. kończy spotkanie

5. sprawdza szczegóły spotkania

Scenariusz nieudany: Spotkanie się nie rozpoczyna lub nie kończy. Nie tworzą się szczegóły ze spotkania po spotkaniu.

Funkcjonalne przypadki użycia :

FU1: rejestracja użytkownika :scenariusz główny

1 wypełnia dane nazwę i hasło

2.dodania do bazy danych

3.akceptuje dane

rejestracja użytkownika : scenariusz alternatywny - zajęta nazwa użytkownika lub hasło niespełniające wymagania

podpunkty 1-2 jak w głównym

3 wyświetlenie komunikatu o zajętości nazwy lub niespełnianiu wymagań w zależności co zostało wykryte przez baze danych i powrót do 1

FU2: logowanie : scenariusz główny

1 wpisz dane nazwę i hasło

2 sprawdzenie poprawności z bazą danych

3 sukces udało się zalogować

logowanie : scenariusz alternatywny - niepoprawne hasło lub nazwa użytkownika:

podpunkty 1 i 2 jak w scenariuszu głównym,

3 .dane niewłaściwe wyświetlenie komunikatu i powrót do punktu 1

logowanie : scenariusz nieudany: Nawet po wpisaniu nieistniejącej nazwy użytkownika i odpowiedniego hasła nie udało się zalogować

FU3: utworzenie grupy użytkowników :scenariusz główny

1 wykonanie scenariusza FU2

2 wypełnia nazwe i dodaje użytkowników do grupy

3. próba dodania do bazy danych

utworzenie grupy użytkowników :scenariusz alternatywny - zajęta nazwa grupy:

podpunkty 1-2-3 jak w głównym

4 wyświetlenie komunikatu o zajętości nazwy lub niespełnianiu wymagań w zależności co zostało wykryte przez baze danych i powrót do 2

utworzenie grupy użytkowników : scenariusz nieudany - Nie udało się stworzyć grupy mimo spełnionych wymagań

FU4: dodanie użytkownika do istniejącej grupy :Scenariusz główny

1 wykonanie scenariusza FU2

2 wybranie grupy

3 wybranie użytkownika którego chcesz do niej dodać

FU5: usunięcie użytkownika : Scenariusz główny

1 wykonanie scenariusza FU2

2 wybranie konta do usunięcia i jego likwidacja

FU6: usunięcie grup użytkownika Scenariusz główny

1 wykonanie scenariusza FU2

2 wybranie grupy i usunięcie jej i wszystkich spotkań które się w niej odbyły

FU7: zarządzanie organizacją konta Scenariusz główny

1 wykonanie scenariusza FU2

2 wybranie opcji zarządzania organizacją

FU8: przeglądanie spotkania Scenariusz główny

1 wykonanie scenariusza FU2

2 wybranie grupy

3 wybranie spotkania

4 przejrzanie szczegółów i interesujących danych

FU9: utworzenie spotkania Scenariusz główny

1 wykonanie scenariusza FU2

2 wybranie opcji utwórz spotkanie

3 wybranie grup i luźnych użytkowników do danego spotkania

4 wybór daty spotkania

FU10: edycja parametrów spotkania Scenariusz główny

1 wykonanie scenariusza FU2

2 wybór spotkania

3 zmiana godziny lub dodanie/usunięcie użytkownika lub całego spotkania

FU11: rozpoczęcie i przebieg spotkania Scenariusz główny

1 wykonanie scenariusza FU2

2 wybór spotkania

3 rozpoczęcie generacji transkrypcji

4 zakończenie spotkania

5 pojawienie się podsumowania

FU12 :weryfikacja danych

1 wykonanie scenariusza FU2

2 baza danych otrzymuje danych

3 otrzymujemy potwierdzenie

weryfikacja danych : scenariusz alternatywny - dane nie poprawne:

2 baza otrzymuje dane

3 otrzymujemy komunikat o niewłaściwych danych

3.4 Potwierdzenie zgodności wymagań

Zatwierdzam specyfikację wymagań, jako spełniających potrzeby Klienta.	<div>.....</div> <div>Data i podpis Właściciela tematu</div>
<i>Uwagi</i>	

4 Architektura

4.1 Definicja architektury

Zastosowaliśmy szablony:

- MVC

Interakcja klienta z widokiem powoduje modyfikację stanu modelu przez kontroler. Kontroler pobiera dane z modelu aby widok mógł je wyświetlić.

U nas model reprezentuje dane sesji burzy mózgów, użytkowników i grup.

Widok to GUI, a kontroler to systemy zarządzania danymi (zarządzanie sesjami burzy mózgów, uwierzytelnianiem, kontakt z bazą danych).

-klient serwer.

Klientem jest u nas aplikacja desktopowa, a rolę serwera pełni MongoDB w chmurze i zewnętrzne usługi takie jak Deepgram lub Google Translate.

Diagram kontekstowy:

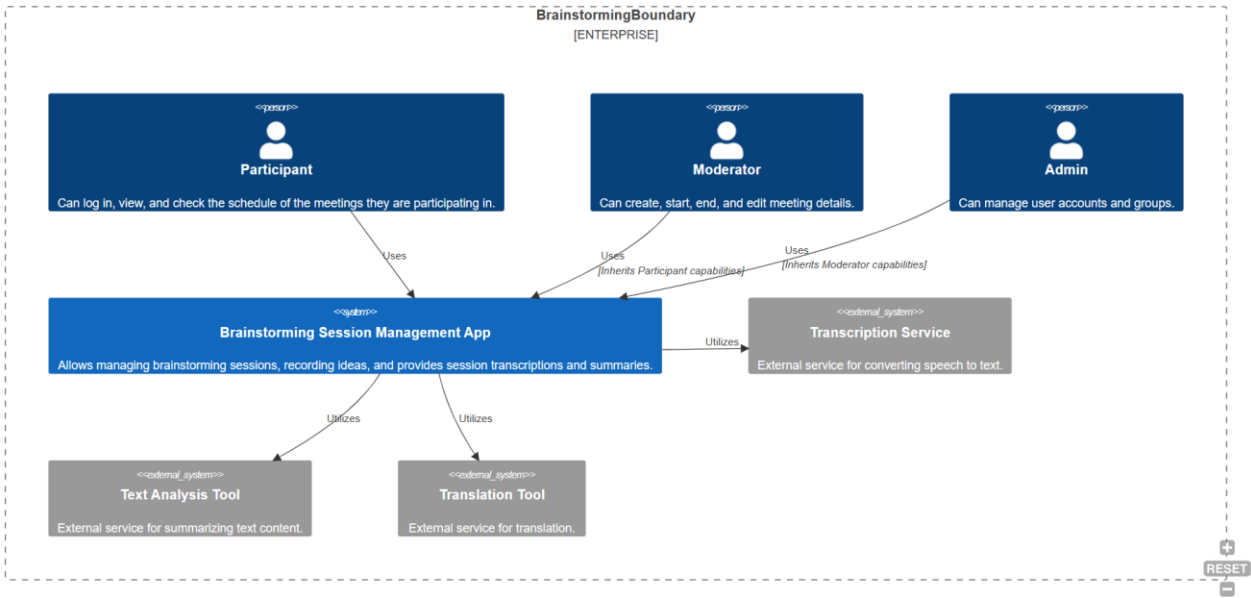
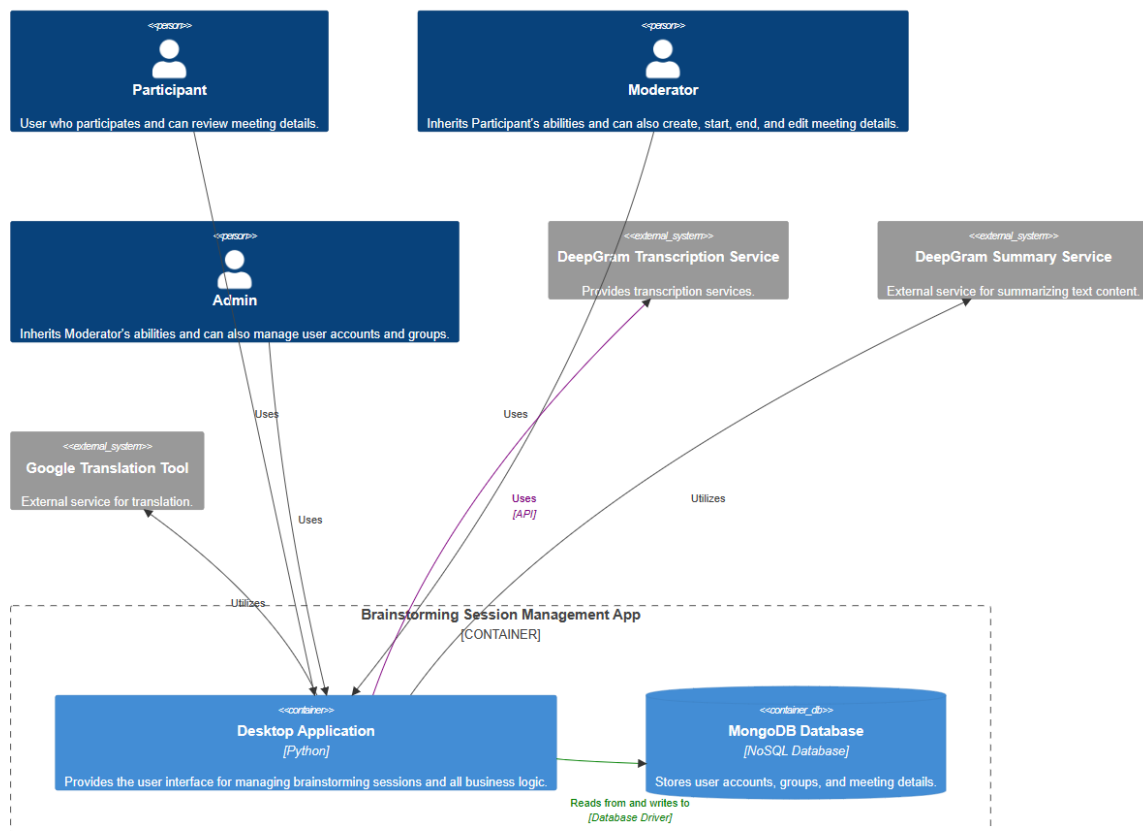
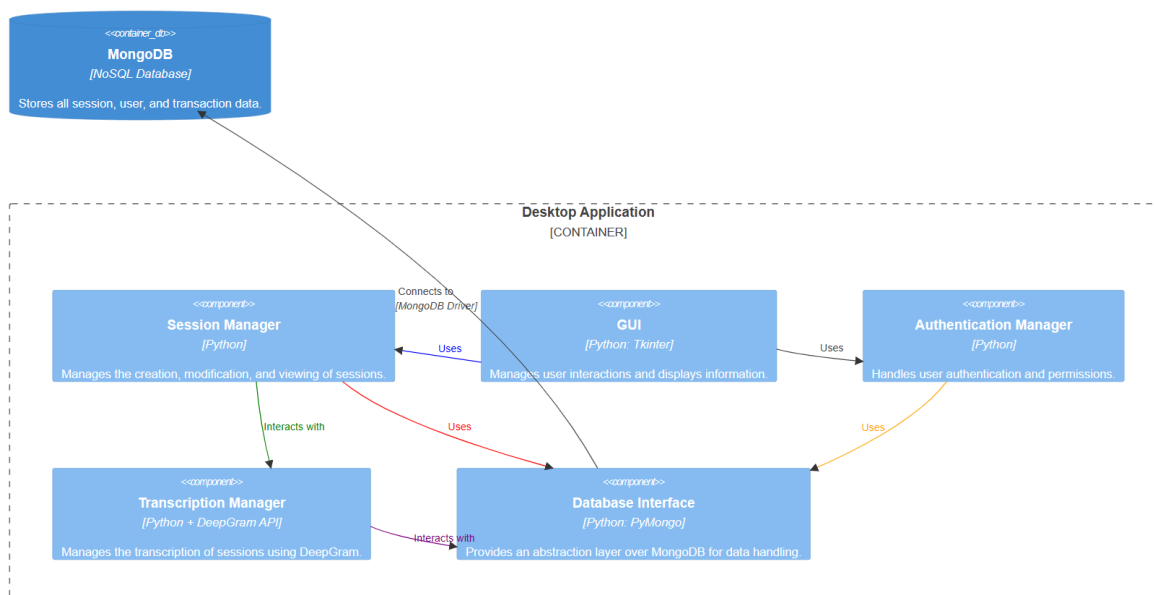


Diagram kontenerowy:



MongoDB Database odnosi się tylko do naszej bazy danych, w której będziemy uzupełniać dane, a nie do usługi hostującej bazę danych. Umieszczenie danych w tej bazie jest częścią aplikacji.

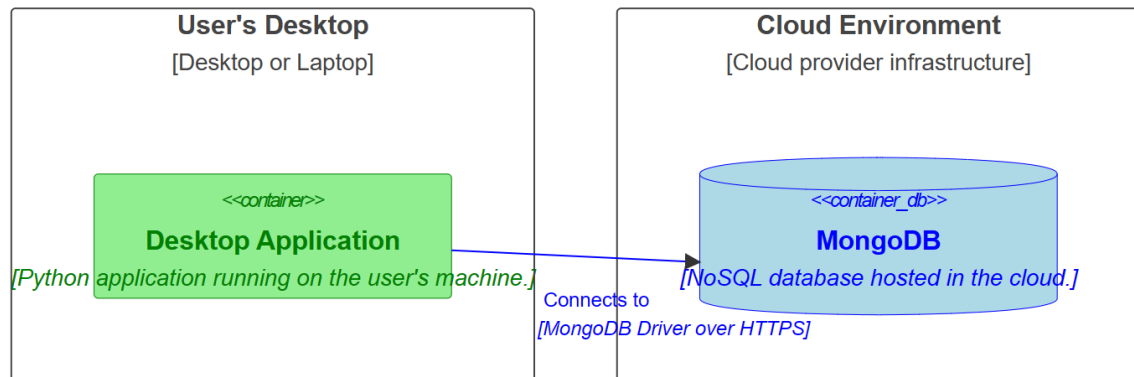
Diagram komponentowy:



Transcription Manager oprócz uzyskania transkrypcji zarządza także operacjami związanymi z transkrypcją, takimi jak podsumowanie lub translacja.

Diagram wdrożeniowy

Deployment Diagram for Brainstorming Session Management Application



4.2 Specyfikacja analityczna i projektowa

Link do repozytorium : https://gitlab-stud.elka.pw.edu.pl/mpakulsk/pzsp2_24/

Podstawowe informacje:

- Aplikacja desktopowa
- Backend – Python
- Frontend – Python (biblioteka Tkinter)
- Transkrypcja – Deepgram
- Podsumowania – Deepgram
- Translacja – Tłumacz Google
- Baza danych – MongoDB

Decyzje wyboru:

Aplikacja desktopowa z backendem i frontendem w Pythonie:

- Wybraliśmy aplikację desktopową, ponieważ została nam ona polecona przez naszego mentora.
- Zastanawialiśmy się jeszcze nad aplikacją webową, w której backend napisany byłby w Pythonie z frameworkiem Flask, a frontend w JavaScriptcie z frameworkiem ReactJS.
- Myśleliśmy również o stworzeniu aplikacji desktopowej z backendem w Pythonie z frameworkiem Flask, a frontendem również w JavaScriptcie z użyciem frameworku Elektron, ale z tego zrezygnowaliśmy, ponieważ mentor nam zalecił napisanie frontendu w języku Python.
- Myśleliśmy jeszcze o aplikacji mobilnej z backendem w Pythonie za pomocą frameworku kivy i frontendem w JavaScriptcie za pomocą React Native lub backendem i frontendem w JavaScriptcie za pomocą React Native.
- Wybraliśmy Pythona zarówno również dlatego, że język ten posiada dobrą integrację z Deepgramem, jest kompatybilny z bazą danych MongoDB i jest stosunkowo prostym językiem.
- Przy tworzeniu GUI wykorzystamy bibliotekę Tkinter, ponieważ jest wbudowana w Pythona i również jest stosunkowo prosta w użyciu.

Transkrypcja:

- Wybraliśmy Deepgram jako narzędzie do transkrypcji, ponieważ posiada model Nova-2 z opcją meeting, która potrafi generować transkrypcję w wielu językach w stosunkowo dobrej jakości.
- Również ma wbudowaną możliwość diaryzacji (podziału wypowiedzianego tekstu na mówców).
- Deepgram nie jest bezpłatnym narzędziem, ale posiada darmowe \$200 kredytu, które starcza na długi czas. W przypadku wykorzystania kredytów, można założyć drugie konto i ponownie operować na \$200 darmowych kredytów.
- Testowaliśmy też inne opcje modelu Nova-2 jak general oraz inne modele jak Nova, Enhanced, Base.

Podsumowania:

Do podsumowania również wybraliśmy model Nova-2 z narzędzia Deepgram, ponieważ w porównaniu z innymi modelami prezentował dobrą jakość tworzenia podsumowań.

Podsumowania, które są tworzone są abstrakcyjne, dzięki czemu opisują przebieg rozmowy własnymi słowami zamiast cytowania pewnych zdań z transkrypcji rozmowy.

Za pomocą biblioteki Transformers z HuggingFace testowaliśmy również inne modele do tworzenia podsumowań takie jak:

- allegro/herbert-base-cased,

- z-dickson/bart-large-cnn-climate-change-summarization – jedyny model z biblioteki Transformers, który potrafi podsumowywać bezpośrednio na język polski,
- facebook/bart-large-cnn.
- google-bert/bert-base-multilingual-cased

Stworzone podsumowanie nie jest idealne, ponieważ skupiałem się na tym, aby używać modeli, które są darmowe żeby aplikacja również była darmowa.

Translacja:

Do translacji użyliśmy narzędzia Tłumacz Google, ponieważ jest całkowicie bezpłatny oraz zapewnia może nie najdokładniejsze ale najbardziej kompletne tłumaczenie tekstu.

Za pomocą biblioteki Transformers z HuggingFace testowaliśmy również inne modele do tworzenia transkrypcji takie jak:

- gsarti/opus-mt-tc-en-pl
- facebook/m2m100_418M
- Helsinki-NLP/opus-mt-pl-en,

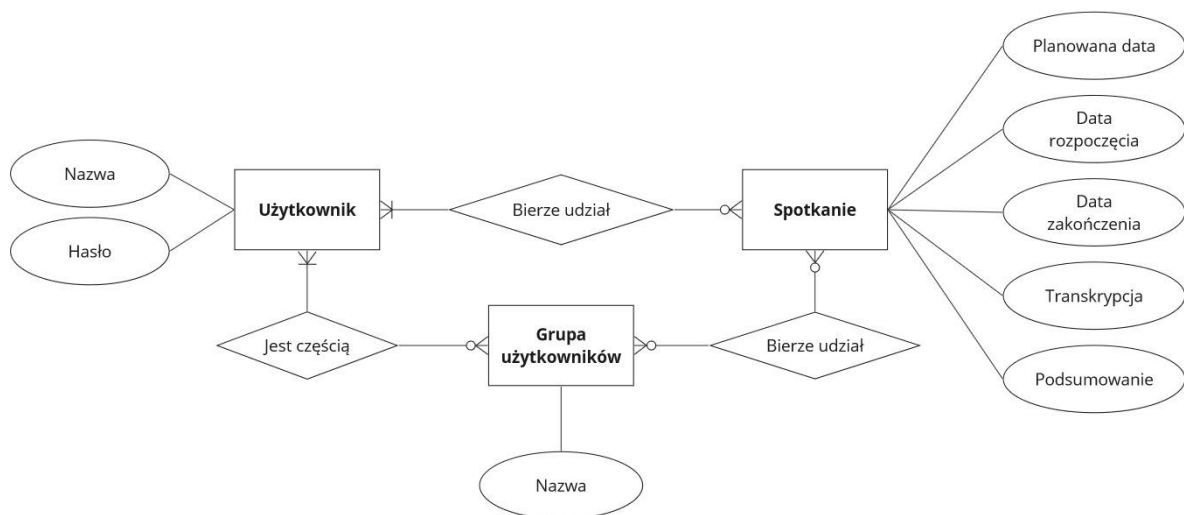
Niestety w większości przypadków tekst albo nie tłumaczył się do końca albo ten sam tekst był powtarzany wielokrotnie.

Baza danych:

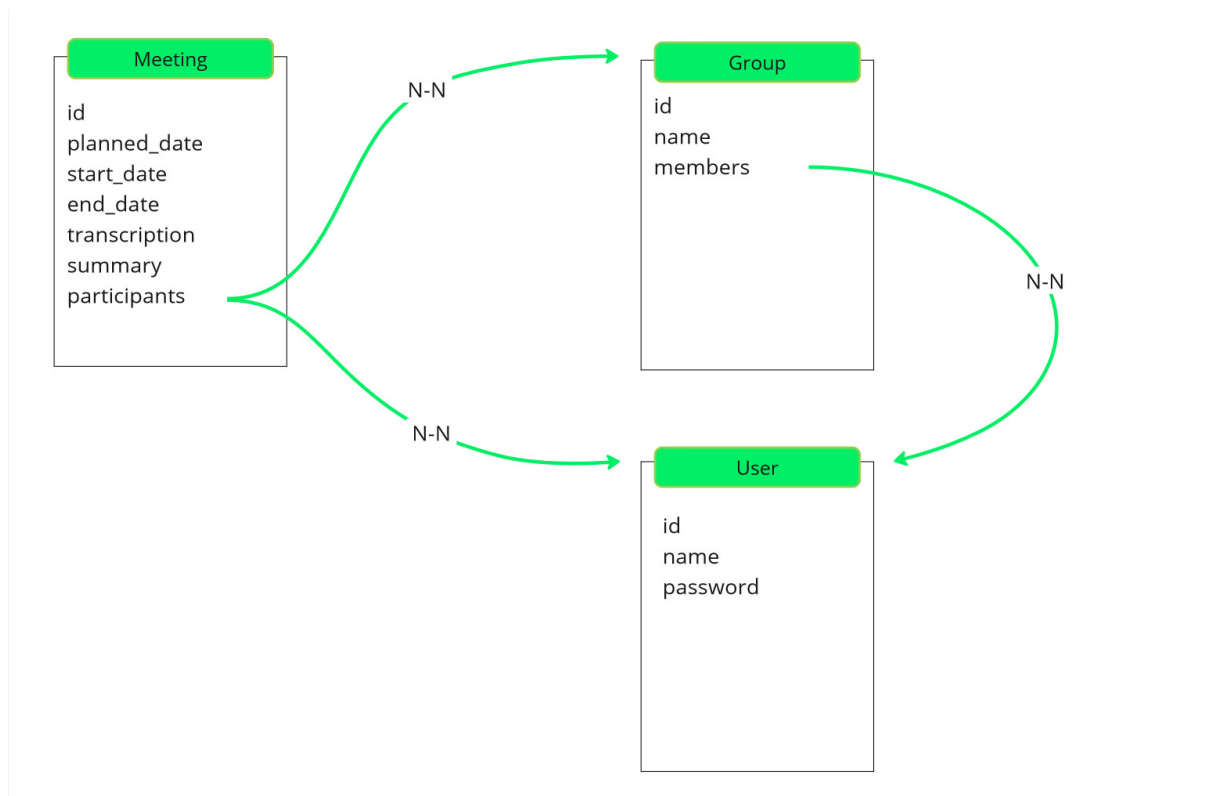
Wybraliśmy bazę danych MongoDB ponieważ umożliwia korzystanie z bazy dokumentowej, która w naszym przypadku wydawała się najlepszą opcją. Do bazy danych miały być zapisywane między innymi dane z transkrypcji oraz podsumowania w języku takim w jakim przebiegała rozmowa. Nie wybraliśmy bazy relacyjnej, ponieważ baza danych MongoDB jest prostsza w obsłudze i może pomieścić obszerne dane z transkrypcji rozmowy oraz nie ma potrzeby zadawania do bazy danych złożonych zapytań. Dodatkowo liczba uczestników spotkania w tabeli spotkań może być różna, a w bazie SQL przez to że tabele mają podział na kolumny trzeba by było trzymać identyfikator każdego uczestnika spotkania w oddzielnej kolumnie lub stworzyć dodatkową tabelę realizującą relację wiele do wielu.

5 Dane trwałe

5.1 Model ER:



5.2 Model logiczny danych



5.3 Przetwarzanie i przechowywanie danych

W ramach bazy danych wybraliśmy bazę dokumentową mongo ponieważ dane które będziemy przechowywać to tabela użytkowników, tabele grup oraz tabela spotkań. Spotkanie może zawierać zarówno luźnych użytkowników jak i całe ich grupy

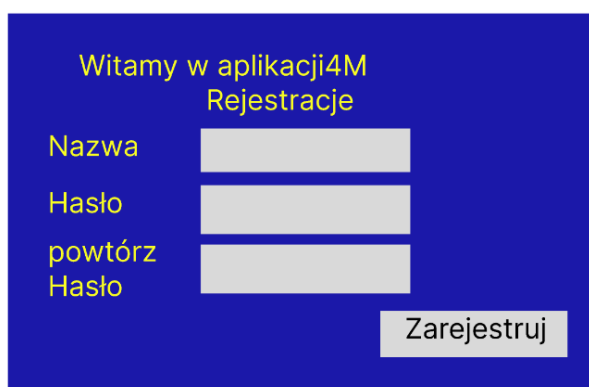
6 Projekt standardu interfejsu użytkownika

Do stworzenia pierwszych szkiców GUI użyliśmy strony figma.com.

Poniżej zamieszczamy link do tej strony:

<https://www.figma.com/file/0C47bqVu2zl7MakocFkiwC/wizja-artystyczna?type=design&node-id=8-83&mode=design&t=sc4VR5RpVzpxq5VN-0>

Ekran rejestracji użytkowników:



Witamy w aplikacji 4M
Rejestracja

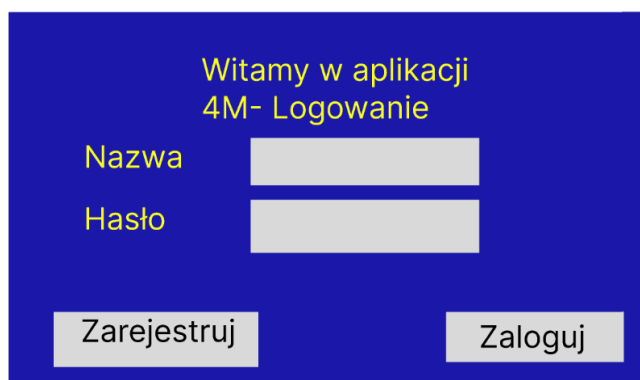
Nazwa

Hasło

powtórz
Hasło

Zarejestruj

Ekran logowania użytkowników:



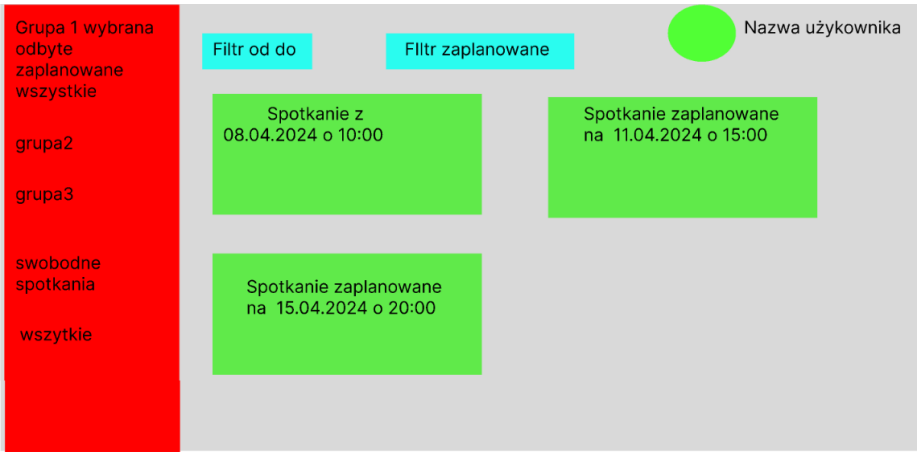
Witamy w aplikacji
4M- Logowanie

Nazwa

Hasło

Zarejestruj Zaloguj

Ekran do przeglądania grup:



Ekran przebiegu spotkania:



Widok kalendarza:

Miesiąc i rok						
Poniedziałek	Wtorek	środa	czwartek	piątek	sobota	niedziela
		1	2	3	4	5
6	7 PZP2 15-18	8 mako 15-18 ANMA 18-20	9	10	11	12
13	14	15	16 9-12 10 13-14 15-18	17	18	19
20	21	22	23	24	25	26

Widok szczegółów spotkania:

Nazwa spotkania	<input type="text" value="nazwa spotkania"/>
data i czas rozpoczęcia(wpierw zaplanowana następnie rzeczywista)	<input type="text" value="data i czas rozpoczęcia"/>
lista członków	<input type="text" value="lista członków"/>
grupy z dostępem	<input type="text" value="grupy z dostępem"/>
data i czas zakończenia(po spotkaniu)	<input type="text" value="data i czas zakończenia"/>
transkrypcja (po spotkaniu)	<input type="text" value="otwórz"/>
Podsumowanie (po spotkaniu)	<input type="text" value="tekst"/>

7 Specyfikacja testów

[standardy obsługi błędów i sytuacji wyjątkowych

rodzaje testów, specyfikacja i opis sposobu realizacji poszczególnych rodzajów testów, scenariusze testowe

miary jakości testów]

8 Wirtualizacja/konteneryzacja

9 Bezpieczeństwo

10 Podręcznik użytkownika

[instrukcja użycia funkcjonalności systemu]

11 Podręcznik administratora

[- instrukcja budowy systemu z kodu źródłowego

- instrukcja instalacji i konfiguracji systemu

- instrukcja aktualizacji oprogramowania

- instrukcja zarządzania użytkownikami i uprawnieniami

- instrukcja tworzenia kopii zapasowych i odtwarzania systemu

- instrukcja zarządzania zasobami systemu]

12 Podsumowanie

[Krytyczna analiza osiągniętych wyników, mocne i słabe strony

Możliwe kierunki rozwoju]

13 Bibliografia

[Wykaz materiałów źródłowych, opis zgodny ze standardem sporządzania opisów bibliograficznych - <https://bg.pw.edu.pl/index.php/przypisy-i-bibliografia>]

Zatwierdzam dokumentację.	<p>.....</p> <p>.....</p> <p style="text-align: right;">Data i podpis Mentora</p>
---------------------------	-----------------------------------------------------------------------------------