

Projekt Zespołowy 2 Grupa 4M

Spis treści

1	Wprowadzenie. 2
1.1	Cel projektu. 2
1.2	Wstępna wizja projektu. 2
2	Metodologia wytwarzania. 2
3	Analiza wymagań. 2
3.1	Wymagania użytkownika i biznesowe. 2
3.2	Wymagania funkcjonalne i нефункционалне. 3
3.3	Przypadki użycia. 3
3.4	Potwierdzenie zgodności wymagań. 10
4	Architektura 10
4,1	Definicja architektury. 10
4.2	Specyfikacja analityczna i projektowa. 13
5	Dane trwale. 16
5.1	Model ER. 16
5.2	Model logiczny danych. 17
5.3	Przetwarzanie i przechowywanie danych. 17
6	Projekt standardu interfejsu użytkownika 18

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 oraz mentorem jest realizowana za pomocą platformy Teams i jest stała. W trakcie tworzenia projektu występują spotkania z ekspertami, 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 нефункционалне

Wymagania funkcjonalne:

Czyli za co odpowiada system:

- Prowadzi konta użytkowników
- Tworzy transkrypcje po spotkaniu
- Umie rozpoznawać użytkowników po ich głosie czytaj kto teraz mówi
- Potrafi 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 нефункционалне:

- 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.

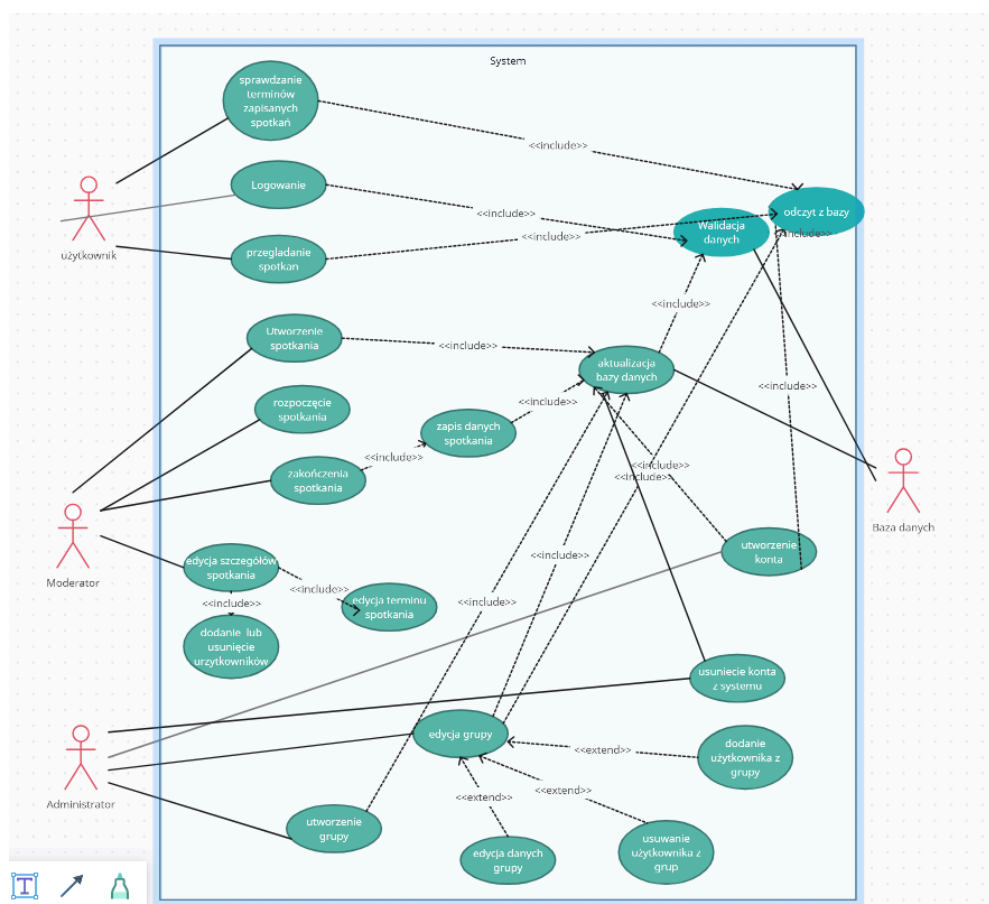
3.3 Przypadki użycia

Nasza aplikacja powstała z myślą, aby była stosowana dla uczestników takich ośrodków jak szkoła letnia, dlatego w dalszej części dokumentacji mogą pojawić się do niej odniesienia.

W naszej aplikacji użytkownicy mogą mieć różne role. Poniżej znajduje się podstawowy ich opis, natomiast wszystkie role użytkowników oraz ich możliwe działania w aplikacji są przedstawione w diagramie przypadków użycia na Rys. 3.1. W dalszej części rozdziału znajdują się historyjki użytkowników z podziałem na ich role, a następnie biznesowe i funkcjonalne przypadki użycia.

Podstawowy opis przypadków użycia:

- Uczestnik (uczestnik szkoły letniej) może uczestniczyć w spotkaniu i przeglądać transkrypcję i podsumowanie ze spotkania.
- Moderator (który by był traktowany jako nauczyciel szkoły letniej) tworzy, rozpoczyna i kończy spotkania oraz edytuje szczegóły spotkania takie jak przypisywanie użytkowników do mówców, edycja terminu spotkania lub edycja uczestników spotkania
- Administrator (kierownik szkoły letniej) tworzy i usuwa konta użytkowników i moderatorów oraz tworzy i edytuje grupy (np. klasa 2d w szkole letniej).



Rys. 3.1 Diagram przypadków użycia

Historyjki użytkowników:

- Uczestnik:
 - Jako uczestnik spotkania chcę móc się zalogować do aplikacji w łatwy sposób, podając nazwę użytkownika i hasło
 - Jako uczestnik chcę 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 chcę 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 chcę 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 chcę 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 chcę 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 nie spełniające wymagania

podpunkty 1-2 jak w głównym

3. Wyświetlenie komunikatu o zajętości nazwy lub niespełnianiu wymagań w zależności co zostało wykryte przez bazę 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. Przeglądanie 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 - Scenariusz główny

1. Wykonanie scenariusza FU2
2. Baza danych otrzymuje dane
3. Otrzymujemy potwierdzenie

Weryfikacja danych : scenariusz alternatywny - dane niepoprawne:

Podpunkty 1, 2 jak w scenariuszu głównym.

3. Otrzymujemy komunikat o niewłaściwych danych

3.4 Potwierdzenie zgodności wymagań

Zatwierdzam specyfikację wymagań, jako spełniających potrzeby Klienta. Data i podpis Właściciela tematu
Uwagi	

4 Architektura

Architekturę podzieliliśmy na 2 podrozdziały. Pierwszy podrozdział określa definicję architektury. Tam przedstawione są szablony, których użyliśmy oraz składniki diagramu C4 przedstawiające poszczególne warstwy aplikacji. Drugim podrozdziałem jest specyfikacja analityczna i projektowa. W tym rozdziale przedstawiamy nasze decyzje architektoniczne z perspektywy programistycznej oraz ich uzasadnienie.

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 oraz Google Translate.

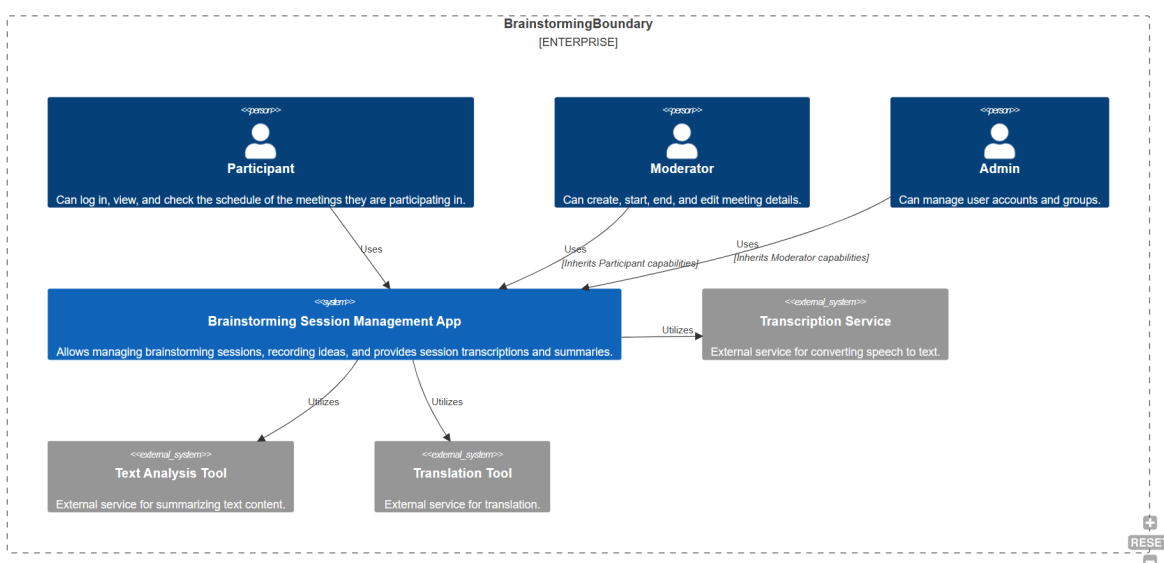
Poniżej znajdują się diagramy C4 poszczególnych warstw aplikacji. Pierwszy diagram dotyczy kontekstu systemu i jest przedstawiony na Rys. 4.1. Na tym poziomie diagramy dają

odbiorcom widok z lotu ptaka na cały system. Szczegóły dotyczące wewnętrznego działania systemu i technologii nie są ważne na tym poziomie. Diagramy te powinny zawierać ogólny opis systemu — co robi, kto go używa i z jakimi innymi systemami współdziała.

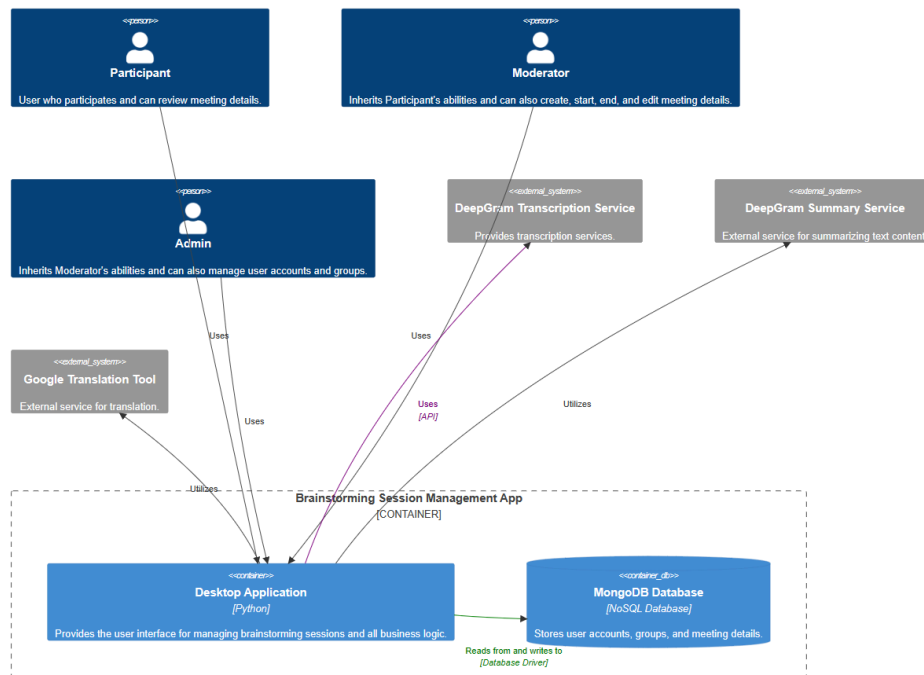
Następnie mamy diagram kontenerowy przedstawiony na Rys. 4.2. Ten diagram powiększa następny poziom systemu, aby szczegółowo opisać kontenery, których system będzie używał, oraz sposób, w jaki będą się komunikować i współdziałać ze sobą. Kontener MongoDB Database odnosi się tylko do naszej bazy danych, w której będziemy uzupełniać dane, a nie do usługi hostującej bazę danych. Umieszczenie danych w tej bazie jest częścią aplikacji.

Kolejnym diagramem jest diagram komponentowy przedstawiony na Rys. 4.3. Schematy komponentów ilustrują szczegóły poszczególnych kontenerów. Diagramy te dzielą strukturalne bloki konstrukcyjne (kod, systemy plików, biblioteki) kontenera na poszczególne komponenty. Diagram szczegółowo opisuje, czym jest każdy komponent, jakie są jego obowiązki, jakie ma interakcje z innymi komponentami oraz szczegóły technologii/implementacji. Komponent Transcription Manager oprócz uzyskania transkrypcji zarządza także operacjami związanymi z transkrypcją, takimi jak podsumowanie lub translacja.

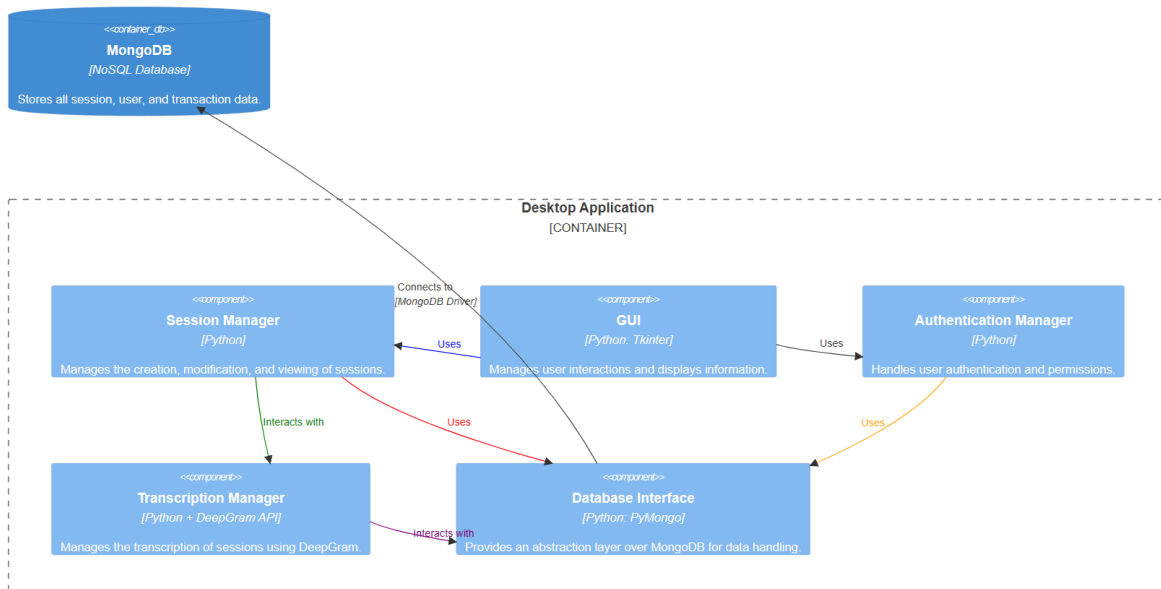
Na końcu mamy diagram wdrożeniowy zaprezentowany na Rys. 4.4. Diagram wdrożeniowy ilustruje, w jaki sposób architektura oprogramowania, zaprojektowana na poziomie koncepcyjnym, przekłada się na fizyczną architekturę systemu, w której oprogramowanie będzie działać jako węzły. Mapuje wdrażanie komponentów oprogramowania w węzłach sprzętowych i przedstawia ich relacje za pośrednictwem ścieżek komunikacyjnych, umożliwiając wizualną reprezentację środowiska wykonawczego oprogramowania w wielu węzłach.



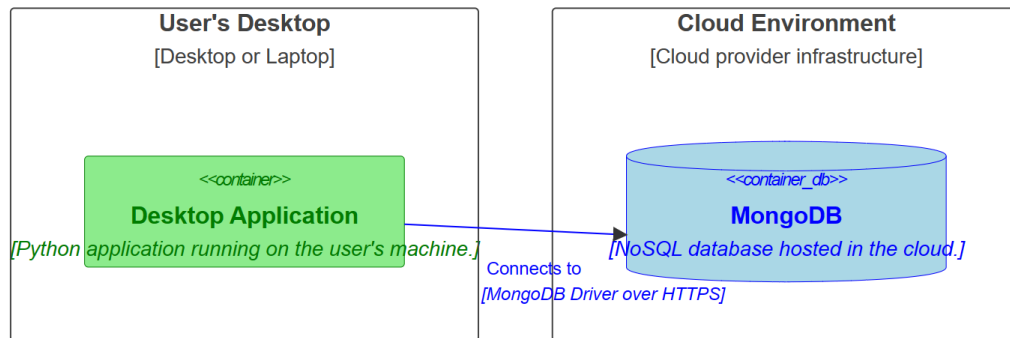
Rys. 4.1 Diagram kontekstowy



Rys. 4.2 Diagram kontenerowy



Rys. 4.3 Diagram komponentowy



Rys. 4.4 Diagram wdrożeniowy

4.2 Specyfikacja analityczna i projektowa

Link do repozytorium kodu: 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 Java Scripcie 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:

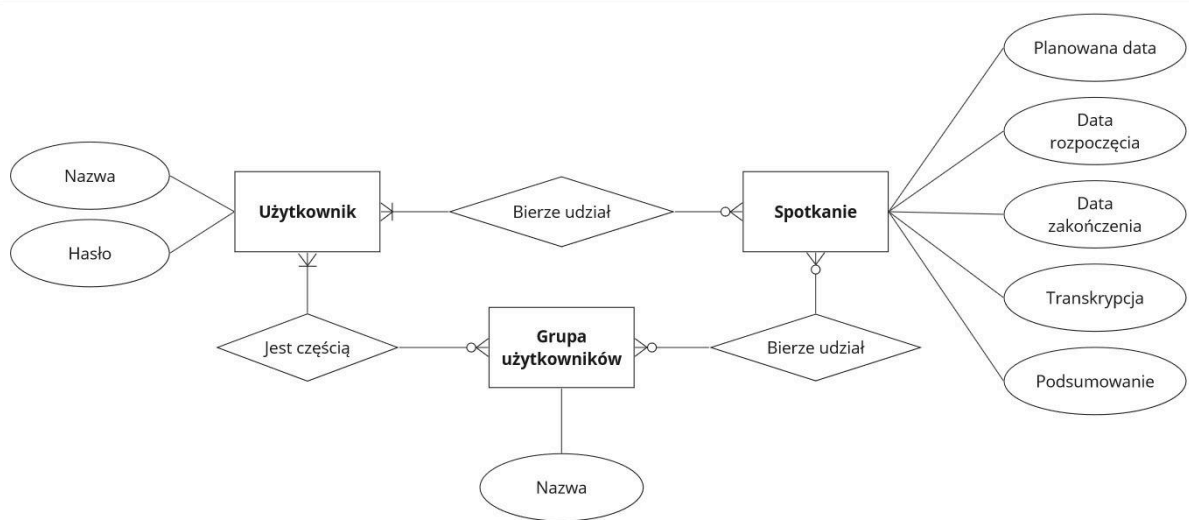
Wybraliśmy bazę danych MongoDB ponieważ umożliwia korzystanie z bazy dokumentowej, która w naszym przypadku wydawała się najlepszą opcją. Do bazy danych miały być zapisywane między innymi dane z transkrypcji oraz podsumowania w języku takim w jakim przebiegała rozmowa. Nie wybraliśmy bazy relacyjnej, 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

Ten rozdział skupia się na sposobie wykorzystania bazy danych z MongoDB. Przedstawiony został model ER oraz relacyjny bazy danych

5.1 Model ER

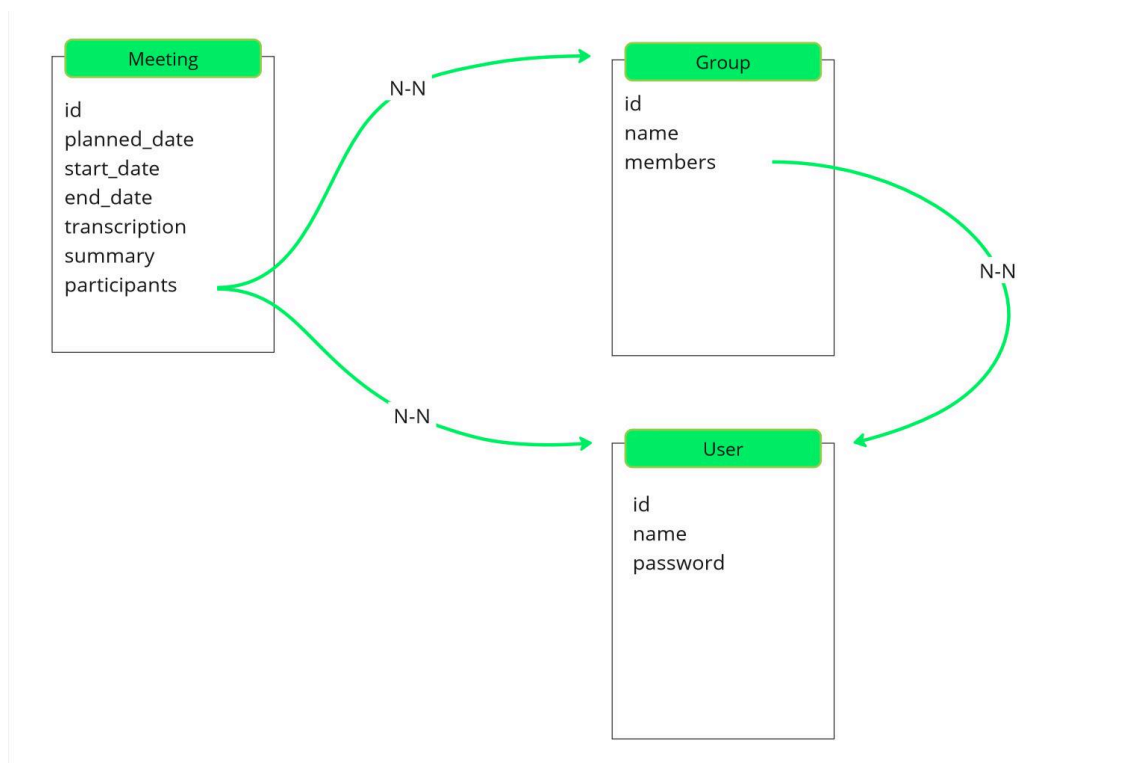
Diagram modelu ER (Rys 5.1) zawiera trzy encje występujące w projekcie, ich atrybuty oraz relacje między nimi przedstawione są poniżej.



Rys. 5.1 Diagram modelu ER

5.2 Model logiczny danych

Diagram modelu logicznego danych został przedstawiony na Rys. 5.2. Widać na nim sposób osadzania (embedding) dokumentów.



Rys. 5.2 Diagram modelu logicznego

5.3 Przetwarzanie i przechowywanie danych

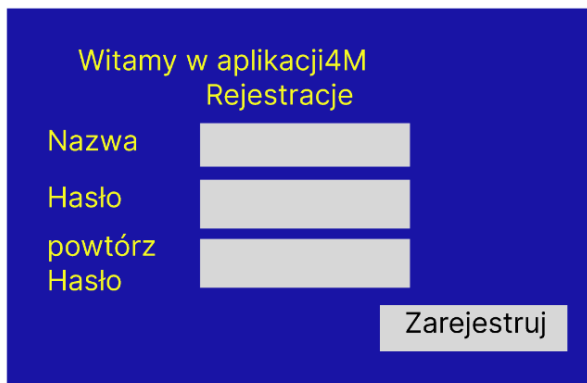
Łączenie z bazą danych MongoDB realizowane jest w aplikacji za pomocą biblioteki PyMongo. Nie planujemy projektować wyzwalaczy ani korzystać z bardziej zaawansowanych funkcjonalności dostępnych w bazie danych.

6 Projekt standardu interfejsu użytkownika

Szkice jak ma wyglądać GUI naszej aplikacji zostały zrobione w programie figma.com i dostępne są pod poniższym linkiem:

<https://www.figma.com/file/0C47bqVu2zI7MakocFkiwC/wizja-artystyczna?type=design&node-id=8-83&mode=design&t=sc4VR5RpVzpxq5VN-0>

Szkice te przedstawiają jak będzie wyglądała aplikacja z perspektywy typowego użytkownika. Ekrany rejestracji i logowania przedstawione są odpowiednio na Rys. 6.1 oraz Rys. 6.2. Po zalogowaniu użytkownik ma możliwość przeglądania dostępnych dla niego spotkań w formie kafelkowej (Rys. 6.3), gdzie będzie mógł wybrać filtry spotkań takie jak okres czasu (użytkownik podaje dwie daty od, do), grupy biorące udział w spotkaniach oraz stan spotkania (odbyte, w trakcie czy zaplanowane). Może również przejrzeć spotkania w formie miesięcznego kalendarza (Rys. 6.5). Po kliknięciu na spotkanie użytkownik przechodzi na ekran szczegółów spotkania (Rys. 6.6). Ten ekran uczestnik spotkania może jedynie przeglądać natomiast Moderator może go edytować. Podczas przebiegu spotkania wyświetlany jest ekran widoczny na Rys. 6.4.



Rys 6.1 Ekran rejestracji użytkowników

Witamy w aplikacji
4M- Logowanie

Nazwa

Hasło

Zarejestruj Zaloguj

Rys 6.2 Ekran logowania użytkowników

Grupa 1 wybrana
odbyte
zaplanowane
wszystkie

grupa2

grupa3

swobodne
spotkania

wszystkie

Filtr od do

Filtr zaplanowane

Nazwa użytkownika

Spotkanie z
08.04.2024 o 10:00

Spotkanie zaplanowane
na 11.04.2024 o 15:00

Spotkanie zaplanowane
na 15.04.2024 o 20:00

Rys 6.3 Ekran do przeglądania grup

Nazwa czas i wygaszacz ekranu (tło lub bąbelki)

Transkrypcja

Tekst próbny

Rys 6.4 Ekran przebiegu spotkania

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 13-14 10 15-18	17	18	19
20	21	22	23	24	25	26

Rys 6.5 Ekran kalendarza

Nazwa spotkania	nazwa spotkania
data i czas rozpoczęcia(wpierw zaplanowana następnie rzeczywista)	data i czas rozpoczęcia
lista członków	lista członków
grupy z dostępem	grupy z dostępem
data i czas zakończenia(po spotkaniu)	data i czas zakończenia
transkrypcja (po spotkaniu)	otwórz
Podsumowanie (po spotkaniu)	tekst

Rys 6.6 Ekran szczegółów spotkania

Zatwierdzam dokumentację.	
	Data i podpis Mentora