

**Politechnika Warszawska**

W Y D Z I A Ł M A T E M A T Y K I  
I N A U K I N F O R M A C Y J N Y C H



# Praca dyplomowa inżynierska

na kierunku Informatyka i Systemy Informacyjne

Usosfix - aplikacja do dokonywania zamian studentów między grupami zajęciowymi

**Lidia Sługocka**

Numer albumu 286403

**Krzysztof Tabeau**

Numer albumu 290504

**Piotr Lewandowski**

Numer albumu 290534

Promotor

dr inż. Paweł Rzążewski

WARSZAWA 2021

.....  
.....

podpis promotora

.....  
.....

podpis autora

## **Streszczenie**

*Temat pracy dyplomowej:* Usosfix - aplikacja do dokonywania zamian studentów między grupami zajęciowymi

Poszukując tematu niniejszej pracy dyplomowej, chcieliśmy, aby była ona funkcjonalnym narzędziem odpowiadającym na istniejące problemy w otaczającym nas środowisku. W związku z tym naturalnym dla nas było, aby w pierwszej kolejności rozważyć problemy istniejące na Uczelni, a konkretniej na naszym Wydziale.

W trakcie swoich studiów realizowaliśmy się m.in. poprzez działalność na rzecz studentów – Krzysztof był starostą grupy dziekańskiej oraz wspólnie z Lidią preźnie działały w Wydziałowej Radzie Samorządu. Stworzyło nam to okazję do zapoznania się z codziennymi problemami studentów oraz zyskania szerszego spojrzenia na funkcjonowanie Wydziału. Dzięki temu zidentyfikowaliśmy problem regularnie pojawiający się w trakcie układania planu zajęć na kolejny semestr.

Administracja Wydziału patrzy na zadanie ułożenia planu zajęć globalnie, my natomiast popatrzyliśmy na nie z perspektywy studenta. Z tego punktu widzenia poza ogólnym harmonogramem grup i przedmiotów istotne jest również wzięcie pod uwagę, do jakich grup jest zapisana dana osoba.

Niniejszy dokument opisuje finalny produkt oraz proces tworzenia aplikacji Usosfix - aplikacji do dokonywania zamian studentów między grupami zajęciowymi.

UsosFix składa się z trzech komponentów - aplikacji mobilnej, sieciowej oraz serwera z bazą danych. Aplikacja ma pomóc studentom w dopasowaniu ich planów zajęć - przenoszeniu się między grupami oraz usunięciu kolizji zajęć lub okienek. Jej działanie opiera się na planie zajęć pobranym z USOSa, prośbach o wymiany zgłoszonych przez studentów oraz doborze w zespoły projektowe.

**Słowa kluczowe:** Usosfix, plan, wymiany, zamiany

## **Abstract**

*Diploma thesis topic:* Usosfix - an application for exchanging students between classes

When searching for the topic of this diploma thesis, we wanted it to be a functional tool responding to the existing problems in the surrounding environment. Therefore, it was natural for us to first consider the problems existing at the University, and more specifically at our Faculty.

During our studies, we acted for the benefit of students - Krzysztof was the leader of the dean's group and, together with Lidia, they worked in the Students' Union Faculty Council. It created us an opportunity to get to know everyday problems of students and gain a broader view of the functioning of the Faculty. Thanks to this, we have identified a problem that appears regularly during the preparation of the timetable for the next semester.

The administration of the Faculty looks at the task of compiling a timetable globally, while we looked at it from the student's perspective. From this point of view, in addition to the general schedule of groups and subjects, it is also important to consider which groups a student is enrolled in.

This document describes the final product and the process of creating the Usosfix application - an application for exchanging students between classes.

UsosFix consists of three components - a mobile and web application and a server with a database. The application is designed to help students adjust their timetable - moving between groups, removing class collisions or having a free period. Its operation is based on the timetable downloaded from USOS, requests for exchanges submitted by students and getting into project teams.

**Keywords:** Usosfix, timetable, exchanges





**Politechnika Warszawska**

Warszawa, 28.01.2022 r.

Krzysztof Tabeau  
290504  
Informatyka i Systemy Informacyjne

## OŚWIADCZENIE

Świadomy/-a odpowiedzialności karnej za składanie fałszywych zeznań oświadczam, że niniejsza praca dyplomowa została napisana przeze mnie samodzielnie, pod opieką kierującego pracą dyplomową.

Jednocześnie oświadczam, że:

- niniejsza praca dyplomowa nie narusza praw autorskich w rozumieniu ustawy z dnia 4 lutego 1994 roku o prawie autorskim i prawach pokrewnych (Dz.U. z 2021 r., poz. 1062) oraz dóbr osobistych chronionych prawem cywilnym,
- niniejsza praca dyplomowa nie zawiera danych i informacji, które uzyskałem/-am w sposób niedozwolony,
- niniejsza praca dyplomowa nie była wcześniej podstawą żadnej innej urzędowej procedury związanej z nadawaniem dyplomów lub tytułów zawodowych,
- wszystkie informacje umieszczone w niniejszej pracy, uzyskane ze źródeł pisanych i elektronicznych, zostały udokumentowane w wykazie literatury odpowiednimi odnośnikami,
- znam regulacje prawne Politechniki Warszawskiej w sprawie zarządzania prawami autorskimi i prawami pokrewnymi, prawami własności przemysłowej oraz zasadami komercjalizacji.

A handwritten signature in blue ink, appearing to read "Krzysztof Tabeau".

czytelny podpis studenta





**Politechnika Warszawska**

Warszawa, 28.01.2022 r.

Lidia Ślugocka  
286403  
Informatyka i Systemy Informacyjne

### **OŚWIADCZENIE**

Świadomy/-a odpowiedzialności karnej za składanie fałszywych zeznań oświadczam, że niniejsza praca dyplomowa została napisana przeze mnie samodzielnie, pod opieką kierującego pracą dyplomową.

Jednocześnie oświadczam, że:

- niniejsza praca dyplomowa nie narusza praw autorskich w rozumieniu ustawy z dnia 4 lutego 1994 roku o prawie autorskim i prawach pokrewnych (Dz.U. z 2021 r., poz. 1062) oraz dóbr osobistych chronionych prawem cywilnym,
- niniejsza praca dyplomowa nie zawiera danych i informacji, które uzyskałem/-am w sposób niedozwolony,
- niniejsza praca dyplomowa nie była wcześniej podstawą żadnej innej urzędowej procedury związanej z nadawaniem dyplomów lub tytułów zawodowych,
- wszystkie informacje umieszczone w niniejszej pracy, uzyskane ze źródeł pisanych i elektronicznych, zostały udokumentowane w wykazie literatury odpowiednimi odnośnikami,
- znam regulacje prawne Politechniki Warszawskiej w sprawie zarządzania prawami autorskimi i prawami pokrewnymi, prawami własności przemysłowej oraz zasadami komercjalizacji.

*Lidia Ślugocka*  
czytelny podpis studenta



Załącznik nr 1 do zarządzenia nr 109 /2021  
Rektora PW z dnia 9 listopada 2021 r.

„załącznik nr 5 do zarządzenia nr 42 /2020 Rektora PW

**Politechnika Warszawska**



Warszawa, 28.01.2022 r.

Piotr Lewandowski  
290534  
Informatyka i Systemy Informacyjne

**OŚWIADCZENIE**

Świadomy/-a odpowiedzialności karnej za składanie fałszywych zeznań oświadczam,  
że niniejsza praca dyplomowa została napisana przeze mnie samodzielnie, pod opieką kierującego pracą  
dyplomową.

Jednocześnie oświadczam, że:

- niniejsza praca dyplomowa nie narusza praw autorskich w rozumieniu ustawy z dnia 4 lutego 1994 roku o prawie autorskim i prawach pokrewnych (Dz.U. z 2021 r., poz. 1062) oraz dóbr osobistych chronionych prawem cywilnym,
- niniejsza praca dyplomowa nie zawiera danych i informacji, które uzyskałem/-am w sposób niedozwolony,
- niniejsza praca dyplomowa nie była wcześniej podstawą żadnej innej urzędowej procedury związanej z nadaniem dyplomów lub tytułów zawodowych,
- wszystkie informacje umieszczone w niniejszej pracy, uzyskane ze źródeł pisanych i elektronicznych, zostały udokumentowane w wykazie literatury odpowiednimi odnośnikami,
- znam regulacje prawne Politechniki Warszawskiej w sprawie zarządzania prawami autorskimi i prawami pokrewnymi, prawami własności przemysłowej oraz zasadami komercjalizacji.

A handwritten signature in black ink, appearing to read "lewandowski", is placed over a dotted line.

czytelny podpis studenta



**Politechnika Warszawska**



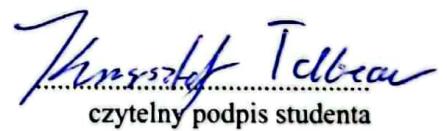
Krzysztof Tabreau  
290504  
Wydział Matematyki i Nauk Informacyjnych  
Informatyka i Systemy Informacyjne

Warszawa, 28.01.2022 r.

Oświadczenie studenta w przedmiocie udzielenia licencji  
Politechnice Warszawskiej

Oświadczam, że jako ~~autor~~/współautor\* pracy dyplomowej pt. *Usosfix – aplikacja do dokonywania zamian studentów między grupami zajęciowymi* udzielam/~~nie~~udzielam\* Politechnice Warszawskiej nieodpłatnej licencji na niewyłączne, nieograniczone w czasie, umieszczenie pracy dyplomowej w elektronicznych bazach danych oraz udostępnianie pracy dyplomowej w zamkniętym systemie bibliotecznym Politechniki Warszawskiej osobom zainteresowanym.

Licencja na udostępnienie pracy dyplomowej nie obejmuje wyrażenia zgody na wykorzystywanie pracy dyplomowej na żadnym innym polu eksploatacji, w szczególności kopiowania pracy dyplomowej w całości lub w części, utrwalania w innej formie czy zwielokrotniania.

  
czytelny podpis studenta

\* niepotrzebne skreślić



**Politechnika Warszawska**



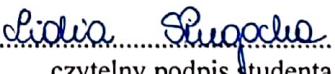
Lidia Slugocka  
286403  
Wydział Matematyki i Nauk Informacyjnych  
Informatyka i Systemy Informacyjne

Warszawa, 28.01.2022 r.

Oświadczenie studenta w przedmiocie udzielenia licencji  
Politechnice Warszawskiej

Oświadczam, że jako autor/współautor\* pracy dyplomowej pt. *Usosfix – aplikacja do dokonywania zamian studentów między grupami zajęciowymi* udzielam/nie udzielam\* Politechnice Warszawskiej nieodpłatnej licencji na niewyłączne, nieograniczone w czasie, umieszczenie pracy dyplomowej w elektronicznych bazach danych oraz udostępnianie pracy dyplomowej w zamkniętym systemie bibliotecznym Politechniki Warszawskiej osobom zainteresowanym.

Licencja na udostępnienie pracy dyplomowej nie obejmuje wyrażenia zgody na wykorzystywanie pracy dyplomowej na żadnym innym polu eksploatacji, w szczególności kopiowania pracy dyplomowej w całości lub w części, utrwalania w innej formie czy zwielokrotniania.

  
czytelny podpis studenta

\* niepotrzebne skreślić





## Politechnika Warszawska

Załącznik nr 3 do zarządzenia nr 109 /2021  
Rektora PW z dnia 9 listopada 2021 r.

„załącznik nr 9 do zarządzenia nr 42 /2020  
Rektora PW

Warszawa, 28.01.2022 r.

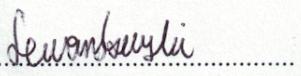
Piotr Lewandowski  
290534  
Wydział Matematyki i Nauk Informacyjnych  
Informatyka i Systemy Informacyjne

### Oświadczenie studenta w przedmiocie udzielenia licencji

Politechnice Warszawskiej

Oświadczam, że jako autor/współautor\* pracy dyplomowej pt. *Usosfix – aplikacja do dokonywania zamian studentów między grupami zajęciowymi* udzielam/nie udzielam\* Politechnice Warszawskiej nieodpłatnej licencji na niewyłączne, nieograniczone w czasie, umieszczenie pracy dyplomowej w elektronicznych bazach danych oraz udostępnianie pracy dyplomowej w zamkniętym systemie bibliotecznym Politechniki Warszawskiej osobom zainteresowanym.

Licencja na udostępnienie pracy dyplomowej nie obejmuje wyrażenia zgody na wykorzystywanie pracy dyplomowej na żadnym innym polu eksploatacji, w szczególności kopiowania pracy dyplomowej w całości lub w części, utrwalania w innej formie czy zwielokrotniania.

  
czytelny podpis studenta

\* niepotrzebne skreślić





# **Spis treści**

<b>1.</b>	<b>Wstęp</b>	<b>19</b>
<b>2.</b>	<b>Analiza wymagań</b>	<b>23</b>
2.1.	Słowniczek	23
2.2.	Ogólny zarys	24
2.3.	Specyfikacja wymagań	25
2.3.1.	Wymagania funkcjonalne	25
2.3.2.	Wymagania niefunkcjonalne	32
<b>3.</b>	<b>Projekt architektury</b>	<b>33</b>
3.1.	Główne komponenty	33
3.1.1.	Aplikacja sieciowa	33
3.1.2.	Aplikacja mobilna	34
3.1.3.	Aplikacja serwerowa	34
3.2.	Architektura	34
3.2.1.	Architektura aplikacji sieciowej i mobilnej	34
3.2.2.	Architektura aplikacji serwerowej	36
3.3.	Opis modułów	36
3.3.1.	Aplikacja mobilna i sieciowa	36
3.3.2.	Aplikacja serwerowa	38
3.4.	Komunikacja między komponentami	39
3.4.1.	Aplikacja sieciowa i mobilna z aplikacją serwerową	39
3.4.2.	Aplikacja serwerowa z bazą danych	39
3.4.3.	Aplikacja serwerowa z serwerem mailowym	39
3.4.4.	Aplikacja serwerowa z systemem USOS	39
3.4.5.	Moduły z interfejsami aplikacji sieciowej	40
3.4.6.	Klasy aktywności z interfejsami aplikacji mobilnej	40
<b>4.</b>	<b>Interfejs użytkownika</b>	<b>45</b>
4.1.	Logo	45

4.2.	Kolory aplikacji mobilnej . . . . .	45
4.3.	Kolory elementów planu zajęć . . . . .	46
4.4.	Czcionki . . . . .	47
4.5.	Interfejsy aplikacji mobilnej i sieciowej . . . . .	48
4.5.1.	Pasek nawigacji/Menu . . . . .	48
4.5.2.	Plan zajęć . . . . .	48
4.5.3.	Plan przedmiotu . . . . .	51
4.5.4.	Grupa zajęciowa . . . . .	52
4.5.5.	Zespoły użytkownika . . . . .	55
4.5.6.	Wymiany użytkownika . . . . .	57
4.5.7.	Wybór wymiany do nadania relacji . . . . .	60
4.5.8.	Panel użytkownika . . . . .	62
4.5.9.	Konwersacje/Wiadomości . . . . .	67
4.5.10.	Ekran logowania . . . . .	67
4.5.11.	Ekran do autoryzacji tokenu - aplikacja mobilna . . . . .	68
4.5.12.	Okna modalne - aplikacja mobilna . . . . .	68
4.6.	Interfejsy zewnętrzne . . . . .	68
4.7.	Procesy . . . . .	69
<b>5.</b>	<b>Implementacja i testy . . . . .</b>	<b>84</b>
5.1.	Algorytm wymian . . . . .	84
5.1.1.	Relacje . . . . .	85
5.1.2.	Implementacja algorytmu . . . . .	86
5.2.	Testy . . . . .	86
5.2.1.	Podejście do implementacji i testów . . . . .	86
5.2.2.	Aplikacja serwerowa . . . . .	86
5.2.3.	Aplikacja sieciowa . . . . .	86
5.2.4.	Aplikacja mobilna . . . . .	86
<b>6.</b>	<b>Użyte technologie . . . . .</b>	<b>88</b>
6.1.	Technologie ogólne . . . . .	88
6.2.	Aplikacja sieciowa . . . . .	88
6.2.1.	Technologie bezpośrednie . . . . .	88
6.2.2.	Technologie wspierające . . . . .	89
6.3.	Aplikacja serwerowa . . . . .	90
6.4.	Aplikacja mobilna . . . . .	90

<b>7. Plan tworzenia projektu</b>	<b>92</b>
7.1. Podział prac	92
7.2. Model wytwórczy	92
7.3. Harmonogram prac	93
7.4. Harmonogram końcowy	97



## 1. Wstęp

Problem, na który odpowiada nasza praca związany jest z układaniem planu zajęć.

Przed rozpoczęciem każdego semestru społeczności Wydziału udostępniany jest wstępny plan zajęć. Tworzy się go w oparciu o:

- kierunki i realizowane w ramach nich przedmioty,
- podział na grupy dziekańskie oraz laboratoryjne,
- dostępność prowadzących,
- dostępność sal.

Ułożenie optymalnego planu zajęć jest problemem trudnym obliczeniowo, zatem zrozumiałe jest, że ciężko jest stworzyć go w taki sposób, aby odpowiadał każdemu.

Uwagi zgłasiane do wstępnego planu dzielą się na dwa rodzaje:

- grupowe - pochodzące od większej grupy,
- jednostkowe - pochodzące od jednego studenta.

Do pierwszego typu zaliczmy m.in. kolizje zajęć, czyli np. sytuacje, w których dana grupa dziekańska w tym samym czasie ma zajęcia z dwóch przedmiotów. Rozwiążanie takich problemów wymaga modyfikacji planu i z reguły nie generuje większych trudności.

Kluczowe jest jednak spojrzenie z perspektywy jednostki. Wówczas rzeczy, które z daleka wydają się nie istnieć, z bliska stanowią poważną barierę na drodze do stworzenia planu odpowiadającego wszystkim. Rozważmy zadanie skonstruowania planu zajęć z perspektywy pojedynczego studenta.

Student może być zapisany na przedmioty obowiązkowe oraz obieralne. W ramach każdego przedmiotu przypisany jest do grup wykładowych, ćwiczeniowych i laboratoryjnych/projektowych. Dodatkowo może on również realizować przedmioty zaległe z poprzednich semestrów. Niektóre przedmioty wymagają dobrania się w zespoły projektowe, w których wszyscy członkowie muszą uczęszczać na te same zajęcia w ramach tego przedmiotu. Ponadto, układając swój plan, student musi znaleźć przestrzeń na zajęcia z języków obcych lub zajęcia wychowania

fizycznego oraz uwzględnić obowiązki pozauczelniane. Wymienione wyżej czynniki sprawiają, że plan studenta często wymaga zmian, ponieważ np. w tym samym czasie student jest zapisany na zajęcia z dwóch przedmiotów lub nie jest w stanie zapisać się na zajęcia z języków obcych. Doświadczenie pokazuje, że takie sytuacje zdarzają się u wielu studentów. Jest to zatem realny problem, którego rozwiążanie wymaga zaangażowania szerokiego grona osób.

Aktualnie najlepszym i najczęściej stosowanym rozwiązaniem powyższego zagadnienia jest proces organizowany przez starostów grup dziekańskich. Do jego realizacji wykorzystywane są popularny portal społecznościowy Facebook oraz arkusze kalkulacyjne Google.

Przebieg tego procesu można zobaczyć na rysunku 1.1.

Niejednokrotnie proces ten jest powtarzany ze względu na pojawienie się istotnych informacji, np. zmianę opublikowanego wcześniej planu zajęć. Powoduje to zmarnowanie czasu wszystkich uczestników procesu, a przede wszystkim niepotrzebnie wykonany ogrom pracy starostów związany z przygotowaniem arkusza, jego nadzorem i weryfikacją.

W obecnie stosowanym rozwiążaniu można znaleźć zdecydowanie więcej wad niż zalet. Jedyną zaletą, jaką potrafimy wskazać, jest fakt, że ten sposób działa i daje wzgleśnie zadowalające wyniki.

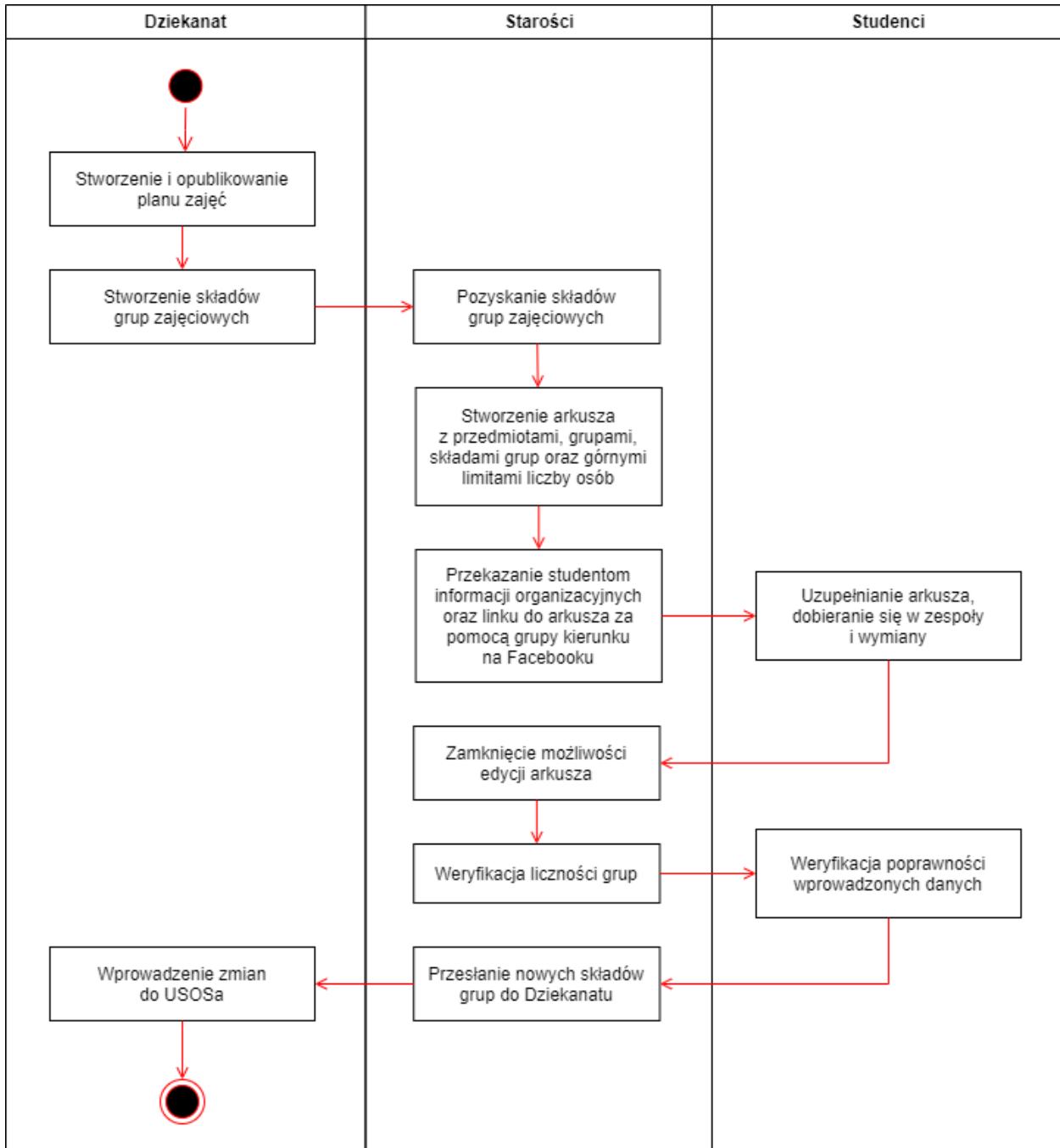
Z kolei do wad zaliczamy:

1. Wysoką praco- i czasochłonność,
2. Udostępnianie danych studentów bez ich wyraźnej zgody,
3. Brak nadzoru nad dostępem do arkusza (ze względu na udostępnienie linku użytkownicy nie identyfikują się imieniem i nazwiskiem ani nawet mailem),
4. Brak możliwości sprawdzenia czy informacja została rozpowszechniona wśród wszystkich zainteresowanych, np. wśród studentów powtarzających pojedyncze przedmioty,
5. Brak „przypisanego” starosty do przedmiotów obieralnych wspólnych dla różnych kierunków i lat,
6. Ręczne uzupełnianie danych w arkuszach oraz brak zautomatyzowania pobierania składów grup,
7. Brak jednego źródła, z którego można pobrać składy grup (każdy student w USOSie widzi tylko składy grup, do których należy),
8. Niebezpieczeństwo, że ktoś zmieni czyjeś przypisanie do grupy bez jego zgody,
9. Konieczność ręcznego weryfikowania czy w danej grupie nie jest za dużo studentów,

10. Brak jasnego oznaczenia, które składy grup obowiązują w przypadku, gdy proces nie został zakończony, semestr się rozpoczął i składy ostatecznych grup różnią się od składów wpisanych w USOSie.

Rozwiążanie, które my proponujemy ma odpowiedzieć na tak złożony problem oraz zlikwidować część wymienionych wyżej wad.

W aplikacji UsosFix dane o studencie udostępnione są tylko po wyrażeniu przez niego zgody, a składy grup uzupełniają się automatycznie. Dodatkowo do aplikacji mają dostęp tylko osoby, posiadające konto w USOSie, a do zgłoszonych wymian ma dostęp jedynie ich autor. Aplikacja zdejmuję obowiązek weryfikowania liczby studentów w grupach, sama przelicza zgłoszone prośby oraz umożliwia wysyłkę maili do Dziekanatu.



Rysunek 1.1: Schemat aktualnego procesu dopasowywania planu zajęć

## **2. Analiza wymagań**

### **2.1. Słowniczek**

**USOS** - Uniwersytecki System Obsługi Studiów

**RODO** - Rozporządzenie Parlamentu Europejskiego i Rady (UE) 2016/679 z dnia 27 kwietnia 2016 r. w sprawie ochrony osób fizycznych w związku z przetwarzaniem danych osobowych i w sprawie swobodnego przepływu takich danych oraz uchylenia dyrektywy 95/46/WE (ogólne rozporządzenie o ochronie danych)

**Użytkownik** - student Wydziału MiNI PW od pierwszego zalogowania do aplikacji UsosFix

**Administrator** - osoba mająca uprawnienia administratora w systemie, czyli mogąca nadawać uprawnienia starosty oraz administratora użytkownikom systemu

**Grupa zajęciowa** - grupa wykładowa, ćwiczeniowa, laboratoryjna lub projektowa prowadzona z ramach danego przedmiotu

**Starosta** - starosta danej grupy dziekańskiej, któremu przysługują dodatkowe uprawnienia w aplikacji

**Wymiana** - przepisanie studenta z jednej grupy zajęciowej danego rodzaju do innej grupy zajęciowej tego samego rodzaju w ramach jednego przedmiotu

**Zespół** - grupa studentów, którzy wspólnie pracują nad projektem w ramach danego przedmiotu

**System** - wszystkie komponenty UsosFix - aplikacja webowa, sieciowa, serwerowa oraz baza danych

**Algorytm** - algorytm zamiany studentów między grupami

## 2.2. Ogólny zarys

Nasza praca inżynierska jest podzielona na aplikację sieciową dostępną poprzez popularne przeglądarki internetowe:

- Google Chrome
- Mozilla Firefox
- Opera
- Safari
- Edge

oraz aplikację mobilną, dedykowaną dla użytkowników urządzeń mobilnych z systemem operacyjnym Android.

Po zalogowaniu każdy student Wydziału może zobaczyć swój plan zajęć. Do wyboru ma plan, który jest wpisany do USOSa oraz plan, który zawiera zmiany, o które wnioskował. Klikając na poszczególne przedmioty w tym planie, student może przenieść się do widoków odpowiednich przedmiotów lub grup zajęciowych. W widoku grup zajęciowych znajdują się informacje o grupie. W tym miejscu student ma również możliwość zgłoszenia chęci zmiany grupy. Aplikacja UsosFix zbiera prośby o zmianę grupy zajęciowej i na ich podstawie generuje nowe składy grup, realizując możliwie jak największą liczbę zamian.

W widokach danych grup zajęciowych zawarta jest lista osób tam zapisanych uwzględniając RODO. Każdy zarejestrowany student otrzymuje unikalny identyfikator, który jest wykorzystywany do komunikacji pomiędzy użytkownikami. Student może wysłać wiadomość do innego użytkownika za pomocą komunikatora tekstowego oraz może go zaprosić do swojego zespołu z tego przedmiotu. Odbiorca wiadomości może ją przyjąć (jeśli tego nie zrobi, nadawca nie będzie mógł nadać kolejnej wiadomości) i odpowiedzieć na nią. Jeśli studenci utworzą własny zespół, będą widoczni dla pozostałych jako zajęci i będą rozważani jako jeden zespół w ramach zamian wewnętrz grup zajęciowych danego przedmiotu.

Zupełność danych polega na liczbie użytkowników korzystających z aplikacji. System zbiera informacje z USOSa od zalogowanych użytkowników.

## 2.3. SPECYFIKACJA WYMAGAŃ

### 2.3. Specyfikacja wymagań

#### 2.3.1. Wymagania funkcjonalne

##### Logowanie

Logowanie do aplikacji, zarówno dla aplikacji mobilnej jak i dla strony internetowej, przebiega za pośrednictwem platformy CAS USOS. Do zalogowania potrzebne są jedynie login oraz hasło do konta w systemie USOS. Korzystanie z aplikacji jest możliwe dla każdego studenta PW, jednak wstępna grupą docelową stanowią studenci Wydziału MiNI.

##### Wyświetlanie planu zajęć użytkownika

Po zalogowaniu do aplikacji wyświetlany jest plan zajęć użytkownika. Widoki planu zajęć zostały stworzone na wzór widoków na stronie internetowej USOS oraz w aplikacji Mobilny USOS. Użytkownik ma możliwość wyboru rodzaju wyświetlanego planu za pomocą odpowiednich przycisków. Do wyboru ma aktualny plan z USOSa oraz plan, który mógłby mieć po zrealizowaniu wszystkich zgłoszonych przez niego wymian.

##### Wyświetlanie danego przedmiotu

Użytkownik może kliknąć na dowolną grupę zajęciową w swoim planie zajęć. Takie kliknięcie przekieruje użytkownika do widoku danego przedmiotu. W widoku przedmiotu wyświetlane są wszystkie grupy zajęciowe realizowane w ramach danego przedmiotu. Dla każdej grupy w tym widoku wyświetlane są informacje o:

- typie zajęć,
- numerze grupy,
- prowadzących,
- terminie zajęć,
- miejscu odbywania się zajęć.

Jeśli zajęcia odbywają się w różnych terminach, to podawana jest informacja o najczęściej występującym terminie.

## **Wyświetlanie szczegółów grupy zajęciowej**

Użytkownik może kliknąć na dowolną grupę zajęciową w widoku danego przedmiotu. Po kliknięciu na ekranie wyświetla się widok ze szczegółowymi informacjami o danej grupie takimi jak:

- nazwa przedmiotu,
- numer grupy,
- prowadzący zajęcia,
- typ zajęć (wykład, ćwiczenia, laboratorium, projekt),
- terminy zajęć,
- miejsce zajęć,
- liczba studentów należących do grupy,
- limit studentów należących do grupy,
- lista studentów należących do danej grupy - jako nazwa studenta wyświetlany jest unikalny identyfikator studenta lub imię i nazwisko studenta w zależności od ustawień użytkownika.

Jeśli student nie należy do danej grupy i nie zgłosił prośby o dołączenie do niej, to w widoku grupy znajduje się przycisk z możliwością zgłoszenia prośby o dołączenie do niej. Jeśli student już zgłosił taką prośbę, to w widoku grupy znajduje się przycisk z możliwością rezygnacji z tej prośby.

W tym widoku użytkownik ma również możliwość zaproszenia do zespołu studentów należących do danej grupy.

## **Wymiany**

### 1. Wyświetlanie wymian

Aplikacja wyświetla wszystkie wymiany zgłoszone przez użytkownika. Każdy element prezentujący wymianę zawiera:

- nazwę przedmiotu,
- typ zajęć,
- numer aktualnej grupy użytkownika,
- numer docelowej grupy użytkownika,

## 2.3. SPECYFIKACJA WYMAGAŃ

- przycisk umożliwiający nadanie lub usunięcie relacji z inną wymianą,
- listę wymian, z którymi powiązana lub wykluczona jest wymiana.

### 2. Wysyłanie prośby o dołączenie do grupy zajęciowej

Użytkownik może wysłać prośbę o dołączenie do grupy zajęciowej poprzez naciśnięcie przycisku w widoku danej grupy zajęciowej. Taka możliwość występuje tylko w grupach, do których użytkownik nie jest zapisany i które są prowadzone w ramach przedmiotu, na który zapisany jest użytkownik.

Zgłoszenie takiej prośby jest równoznaczne ze stworzeniem prośby o wymianę między grupami oraz z rezygnacją z grupy zajęciowej w ramach danego przedmiotu, do której aktualnie zapisany jest użytkownik.

### 3. Anulowanie prośby o dołączenie do grupy zajęciowej

Użytkownik może zrezygnować z dołączania do grupy zajęciowej poprzez naciśnięcie odpowiedniego przycisku w widoku grupy zajęciowej, która jest grupą docelową w stworzonej przez użytkownika wymianie.

### 4. Nadawanie relacji

Każda wymiana może być w relacji powiązania lub wykluczania z innymi wymianami użytkownika. Nadanie relacji między wymianami następuje poprzez:

- (a) kliknięcie odpowiedniego przycisku przy danej wymianie,
- (b) wybór wymian, które mają być w relacji,
- (c) oraz zatwierdzenie wybranych wymian.

### 5. Anulowanie relacji

Relacja jest anulowana poprzez kliknięcie odpowiedniego przycisku przy danej relacji w widoku wymian.

## Zespoły

W algorytmie zamiany między grupami, zespoły traktowane są jak jedna jednostka. Albo do danej grupy zajęciowej trafiają wszyscy członkowie zespołu, albo nikt.

### 1. Wyświetlanie zespołów

Aplikacja wyświetla wszystkie zespoły, do których należy lub do których zaproszony jest użytkownik.

Każdy element reprezentujący zespół, do którego należy użytkownik, zawiera:

- nazwę przedmiotu,
- listę użytkowników należących do zespołu,
- listę użytkowników zaproszonych do zespołu,
- przycisk umożliwiający opuszczenie zespołu,
- przyciski umożliwiające usunięcie innych użytkowników z zespołu,
- przyciski umożliwiające anulowanie zaproszenia innych użytkowników do zespołu,
- pole tekstowe oraz przyciski umożliwiające wyszukanie oraz zaproszenie innych użytkowników do zespołu.

Każdy element reprezentujący zespół, do którego zaproszony jest użytkownik, zawiera:

- nazwę przedmiotu,
- listę użytkowników należących do zespołu,
- listę użytkowników zaproszonych do zespołu,
- przyciski umożliwiające zaakceptowanie lub odrzucenie zaproszenia do zespołu.

## 2. Zapraszanie do zespołu

Użytkownik może zaprosić innych użytkowników do zespołu z danego przedmiotu. Wysyłanie zaproszeń do zespołu jest możliwe na dwa sposoby:

- (a) Za pomocą widoku grupy zajęciowej - klikając przycisk przy nazwie danego użytkownika w liście studentów uczęszczających na daną grupę zajęciową.
- (b) Za pomocą widoku już istniejących zespołów - do każdego istniejącego już zespołu, studentów, którzy uczęszczają na dany przedmiot oraz którzy jeszcze nie należą do żadnego zespołu w danym przedmiocie, poprzez wcześniejsze wyszukanie ich za pomocą pola tekowego.

## 3. Akceptowanie próśb o dołączenie do zespołu

Każdy zaproszony do zespołu może zaakceptować lub odrzucić zaproszenie za pomocą odpowiedniego przycisku w widoku zespołu.

## 4. Wychodzenie z zespołu

Każdy użytkownik należący do zespołu ma możliwość zrezygnowania z należenia do zespołu za pomocą odpowiedniego przycisku w widoku zespołu.

## 5. Usuwanie innych użytkowników z zespołu

Każdy użytkownik należący do zespołu może usunąć z zespołu innych użytkowników do

## 2.3. SPECYFIKACJA WYMAGAŃ

niego należących poprzez naciśnięcie odpowiedniego przycisku przy nazwie użytkownika w widoku zespołu.

### 6. Anulowanie zaproszenia do zespołu

Każdy użytkownik należący do zespołu może anulować zaproszenie innych użytkowników do zespołu poprzez naciśnięcie odpowiedniego przycisku przy nazwie użytkownika w widoku zespołu.

## Wiadomości

### 1. Wyświetlanie konwersacji

Wszystkie konwersacje użytkownika - zarówno te, których jest członkiem jak i te, do których jest zaproszony - są wyświetlane w formie podobnej jak konwersacje w powszechnie znanej aplikacji Messenger razem z polem tekstowym do wyszukiwania użytkowników. Każda konwersacja zawiera nazwę użytkownika oraz ostatnią wiadomość wysłaną w ramach konwersacji wraz z czasem jej wysłania. Konwersacje są posegregowane od najmłodszej do najstarszej po ostatnich wiadomościach.

### 2. Wyświetlanie wiadomości w konwersacji

Wiadomości w ramach jednej konwersacji są wyświetlane w analogiczny sposób jak wiadomości w powszechnie znanej aplikacji Messenger.

### 3. Zapraszanie do czatu

Użytkownik ma możliwość zaproszenia do czatu każdego użytkownika aplikacji. Wyszukiwanie użytkowników odbywa się w widoku wszystkich konwersacji na podstawie wpisanego prefiksu w polu tekstowym z wyszukiwaniem. Po znalezieniu użytkownika, kliknięciu na niego oraz naciśnięciu przycisku zapraszającego do wyszukanego użytkownika zostaje wysłane zaproszenie do czatu. Użytkownik nie ma możliwości napisania wiadomości dopóki zaproszenie nie zostanie zaakceptowane.

### 4. Akceptowanie zaproszenia do czatu

Użytkownik może zaakceptować lub odrzucić zaproszenie do czatu poprzez naciśnięcie odpowiedniego przycisku w widoku wiadomości danej konwersacji. Jeśli zaproszenie zostanie zaakceptowane, to należący do konwersacji użytkownicy mogą od tej pory wymieniać się wiadomościami. Jeśli zaproszenie zostanie odrzucone, to użytkownicy nie mają możliwości wymieniania się wiadomościami ani wysłania do siebie ponownego zaproszenia.

### 5. Wysyłanie wiadomości

Mechanizm wysyłania wiadomości jest analogiczny jak w powszechnie znanej aplikacji Mes-

senger. Zawartością wiadomości może być tylko tekst. Użytkownik nie ma możliwości dołączenia do wiadomości żadnego załącznika. Wysyłanie wiadomości odbywa się w czasie rzeczywistym.

## Panel użytkownika

Aplikacja posiada moduł zwany Panelem użytkownika, który składa się z następujących segmentów:

- Wyświetlanie informacji o użytkowniku

Panel użytkownika dla każdego użytkownika prezentuje informacje o imieniu, nazwisku, numerze albumu, nazwie użytkownika.

- Zmiana nazwy użytkownika

Użytkownik może zmienić swoją nazwę użytkownika przy pomocy pola tekstowego oraz przycisku.

- Zmiana języka aplikacji

Użytkownik ma możliwość wyboru języka aplikacji - polskiego lub angielskiego. Wybór użytkownika jest zapisywany w bazie tak, aby przy kolejnym uruchomieniu aplikacji wszystkie dane zostały wyświetlane w odpowiednim języku.

- Panel starosty

Użytkownicy posiadający uprawnienia starosty w panelu użytkownika mają tak zwany Panel starosty. W panelu starosty znajdują się:

- tabela z nazwami przedmiotów, na które zapisany jest użytkownik oraz z informacjami ile zostało zgłoszonych wymian, ile z nich zostało zrealizowanych oraz ile z nich nie zostało zrealizowanych. Te same, skumulowane dla wszystkich przedmiotów, dane są wyświetlane u góry tabeli,
- przycisk umożliwiający uruchomienie algorytmu zamiany studentów między grupami,
- przycisk umożliwiający wysłanie składu grup po wymianach do Dziekanatu.

- Uruchamianie algorytmu zamiany studentów między grupami

Użytkownicy z uprawnieniami starosty mogą uruchomić algorytm dla przedmiotów na które są zapisani wybierając przedmioty z listy.

- Wysyłanie danych o dokonanych zmianach

Użytkownicy z uprawnieniami starosty mogą wysłać do Dziekanatu dane z finalnymi składami grup zajęciowych, po wykonaniu algorytmu zmiany studentów między zajęciami.

## 2.3. SPECYFIKACJA WYMAGAŃ

- Zarządzanie uprawnieniami

Użytkownicy posiadający uprawnienia administratora aplikacji w panelu użytkownika mają tak zwany Panel administratora. W panelu administratora znajdują się:

- lista użytkowników z uprawnieniami starosty,
- pole tekstowe oraz przycisk umożliwiające nadanie użytkownikom uprawnień starosty,
- lista użytkowników z uprawnieniami administratora,
- pole tekstowe oraz przycisk umożliwiające nadanie użytkownikom uprawnień administratora,
- przy każdym użytkowniku na obu listach znajduje się przycisk umożliwiający odebranie uprawnień,
- przycisk umożliwiający rozpoczęcie nowego semestru.

Nadawanie oraz odbieranie uprawnień jest potwierdzane odpowiednim przyciskiem na wyskakującym okienku pop-up.

Początkowo administratorami są twórcy aplikacji.

### Rozpoczęcie nowego semestru

Użytkownicy z uprawnieniami administratorów mogą rozpocząć nowy semestr, naciskając na odpowiedni przycisk w Panelu admina. Naciśnięcie tego przycisku powoduje wyczyszczenie danych w bazie oraz pobranie informacji o grupach zajęciowych wszystkich użytkowników w nowym semestrze.

### Wylogowanie

Użytkownik może wylogować się z aplikacji za pomocą odpowiedniego przycisku.

### Algorytm zamiany studentów między grupami

Wszystkie wymiany są realizowane zgodnie z algorytmem zamiany studentów między grupami. Celem algorytmu jest zmaksymalizowanie liczby zrealizowanych wymian, przy jednoczesnym uwzględnieniu następujących ograniczeń:

- liczba wolnych miejsc w grupie zajęciowej,
- składy zespołów, których członkowie muszą być w tych samych grupach w ramach danego przedmiotu,

- zmiany warunkowe - użytkownik może określić, że niektóre dotyczące go zmiany muszą zostać przeprowadzone jednocześnie, lub że pewne zmiany się wykluczają.

Kolejność zgłaszania wymian nie jest brana pod uwagę przy realizacji wymian.

### **2.3.2. Wymagania niefunkcjonalne**

#### 1. Ochrona danych osobowych

Aplikacja spełnia wymogi RODO.

#### 2. Konfigurowalność

Aplikacja umożliwia połączenie z dowolną instalacją USOSa, zależnie od konfiguracji bąkendu, co umożliwia korzystanie z aplikacji na wielu uczelniach.

#### 3. Stabilność

Aplikacja jak i jej stabilność jest zależna od serwisu USOS. Błąd w działaniu USOSWeb nie powinien mieć wpływu na działanie aplikacji dla już wcześniej zalogowanych użytkowników. Jednak dla nowych użytkowników aplikacji błąd w działaniu USOSa poskutkuje brakiem możliwości zalogowania się do aplikacji.

#### 4. Dostępność

Aplikacja jest łatwa w użyciu oraz posiada przejrzysty interfejs.

#### 5. Utrzymanie

Aplikacja umożliwia diagnozowanie i zapobieganie błędom, poprzez logowanie błędów oraz zautomatyzowane testy.

#### 6. Rozszerzalność

Aplikacja umożliwia dodanie nowych funkcjonalności bez konieczności wprowadzania istotnych zmian w już istniejących częściach kodu.

## **3. Projekt architektury**

### **3.1. Główne komponenty**

System UsosFix będzie działać w dwóch językach: angielskim i polskim. UsosFix dzieli się na 3 główne komponenty:

- aplikację mobilną,
- aplikację sieciową,
- aplikację serwerową.

#### **3.1.1. Aplikacja sieciowa**

Aplikacja sieciowa została zrealizowana za pomocą abstrakcyjnej idei Single-Page-App, która zakłada ściągnięcie wszystkich plików potrzebnych do działania aplikacji podczas pierwszego załadowania strony oraz przemieszczania się między widokami bez odświeżania. Przyśpiesza to działanie aplikacji, zwiększa dynamikę oraz dodaje wartość wizualną dla użytkownika.

Kolejną abstrakcyjną ideą jest kontroler stanu. Ta idea wymusza skoncentrowanie logiki aplikacji w miejscu odosobnionym od interfejsu użytkownika. Zwiększa również bezpieczeństwo oraz organizację projektu podczas potencjalnej rozbudowy. Aplikacja jest dostępna dla użytkowników następujących przeglądarek internetowych:

- Google Chrome dla wersji 58 i wyższych,
- Mozilla Firefox dla wersji 52 i wyższych,
- Opera dla wersji 45 i wyższych,
- Safari dla wersji 10 i wyższych,
- Edge dla wersji 12 i wyższych.

### 3.1.2. Aplikacja mobilna

Aplikacja mobilna została zrealizowana w środowisku Android Studio przy użyciu języka Java do opisu logiki jej funkcjonowania. Jest ona dostępna dla użytkowników Android wersji 4.3 (Jelly Bean) i nowszych, co zapewnia funkcjonowanie aplikacji na 98,4% urządzeń z systemem operacyjnym Android. Aplikacja została zaprojektowana tak, aby być jak najbardziej intuicyjną. Projekt części widoków wzorowany był na widokach aplikacji Mobilny USOS.

Aplikacja stworzona jest w architekturze 'klasycznej' z podziałem na dwie warstwy. Podzielona jest na widoki, prezentujące informacje i odpowiadające za interakcje z użytkownikiem, oraz klasy aktywności, opisujące logikę aplikacji. Oba te komponenty są nierozerwalnie ze sobą skorelowane. Każdy widok ma swoją klasę aktywności.

W przeciwieństwie do aplikacji webowej, w aplikacji mobilnej dane pobierane są dopiero wtedy, gdy są potrzebne - przy uruchamianiu poszczególnych widoków lub gdy użytkownik dokona zmian wymagających aktualizacji danych. Dane są przekazywane między klasami aktywności tylko wtedy, gdy jest to niezbędne do poprawnego działania aplikacji.

### 3.1.3. Aplikacja serwerowa

Aplikacja serwerowa jest odpowiedzialna za udostępnienie serwera z REST API, z którego korzystają oba opisane powyżej moduły. Serwer jest połączony do bazy danych, w której przechowywane są informacje na temat planów zajęć użytkowników, aktualnych grup oraz możliwych zamian. To właśnie po stronie serwera dokonywane są zamiany. Aplikacja serwerowa komunikuje się także z serwerem SMTP, który umożliwia wysyłanie maili z wynikami przeprowadzonych zamian.

## 3.2. Architektura

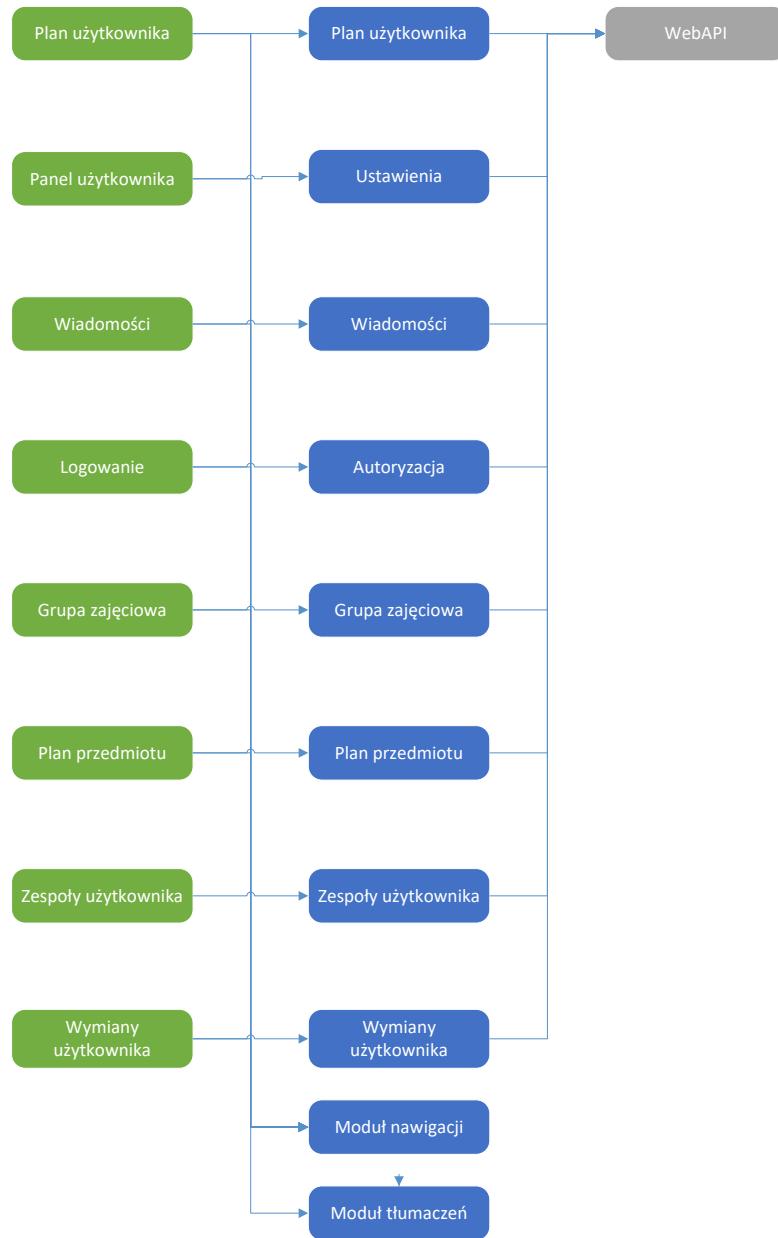
### 3.2.1. Architektura aplikacji sieciowej i mobilnej

Na diagramie 3.1 została zaprezentowana architektura aplikacji sieciowej oraz aplikacji mobilnej z następującym podziałem kolorystycznym:

- na zielono zostały zaznaczone widoki,
- na niebiesko zostały zaznaczone moduły/klasy aktywności zajmujące się danymi w aplikacji,
- na szaro została zaznaczona aplikacja serwerowa, z którą obie aplikacje się komunikują.

### 3.2. ARCHITEKTURA

Widoki zajmują się tylko wyświetlaniem danych, które znajdują się w modułach oraz przekazywaniem żądań między użytkownikiem a modułami/klasami aktywności. Moduły i klasy aktywności zajmują się przechowywaniem danych oraz szeroko pojętą logiką aplikacji. Jeśli moduł/klasa aktywności zarządza danymi, które nie są statyczne, to kontaktuje się za pomocą protokołu HTTP z aplikacją serwerową. Szczególnym przypadkiem jest moduł Wiadomości, który łączy się z API za pomocą WebSocketów.



Rysunek 3.1: Zależności pomiędzy modułami, interfejsami i API

W przypadku implementacji kontrolera stanu w aplikacji sieciowej nie posiadamy klas, ponieważ

nie są one potrzebne i optymalne w obliczu całego mechanizmu tej idei. Na rysunku 3.2 znajduje się diagram interfejsów i ich połączenia z kontrolerem stanu.

Logika aplikacji mobilnej zbudowana jest na podstawie klas aktywności (Activity). Wszystkie klasy, którym odpowiadają widoki, są klasami pochodnymi od klasy BaseActivity. Każda klasa aktywności zarządza logiką w ramach jednego widoku.

Na rysunku 3.3 znajduje się diagram klas aplikacji mobilnej.

### 3.2.2. Architektura aplikacji serwerowej

W tym komponencie możemy wyróżnić sześć modułów widocznych na diagramie 3.4. Pierwszym z nich jest **Baza danych**, w której przechowywane są dane o planach zajęć użytkowników aplikacji. Drugim jest **Serwer mailowym** który odpowiada jedynie za wysyłanie maili z wynikami wymian. Kolejne cztery moduły odpowiadają za logikę biznesową serwera. Są ze sobą ściśle związane, jednak możliwe jest wyodrębnienie następujących obszarów: **Komunikator**, **Użytkownicy**, **Plan**, **Autoryzacja**. Te cztery części korzystają z zewnętrznego API udostępnianego przez system USOS. Możliwe jest podłączenie serwera do dowolnej instancji USOS API, jednak w naszej pracy skupiamy się jedynie na instancji utrzymywanej przez Politechnikę Warszawską. Ostatnim z modułów serwera jest **Rest API**, czyli interfejs, dzięki któremu pozostałe komponenty UsosFix mogą korzystać z funkcjonalności serwera.

## 3.3. Opis modułów

### 3.3.1. Aplikacja mobilna i sieciowa

Wszystkie moduły pobierają dane oraz zmieniają je poprzez wysłanie żądania do API. Odpowiadają również za dynamiczne wyświetlanie danych otrzymanych od API - zarówno uzupełnianie ich jak i dodawanie poszczególnych elementów. Te operacje są efektem akcji wykonanych przez użytkownika w odpowiednich interfejsach. Ze względu na większość ten sam cel i sposób działania w aplikacji mobilnej i sieciowej, poniższa lista przedstawia moduły obu komponentów.

**Panel użytkownika** - odpowiada za pobranie i przechowywanie danych o użytkowniku tzn. imienia, nazwiska, numeru albumu, nazwy użytkownika, identyfikatora, ustawień wyświetlania użytkownika, języka aplikacji. Za jego pomocą można zmienić ustawienia takie jak nazwa użytkownika, wyświetlanie użytkownika oraz język aplikacji. W zależności od uprawnień użytkownika odpowiada on również za:

- dla starosty:

### 3.3. OPIS MODUŁÓW

pobranie i przechowywanie danych o wymianach zgłoszonych w ramach 'należących' do niego przedmiotów, wysyłanie żądania o uruchomienie algorytmu oraz wysyłanie danych do Dziekanatu,

- dla administratora:

pobieranie danych o użytkownikach z uprawnieniami starosty oraz administratora, nadawanie i odbieranie uprawnień oraz rozpoczęwanie nowego semestru.

**Autoryzacja** - odpowiada za przekierowanie do strony USOSa oraz autoryzację tokenu

**Konwersacje** - odpowiada za pobieranie, przechowywanie oraz dynamiczne wyświetlanie danych o konwersacjach użytkownika, za wyszukiwanie użytkownika oraz za zapraszanie użytkowników do czatu. Dla każdej konwersacji przechowuje dane o dacie wysłania ostatniej wiadomości, treści ostatniej wiadomości oraz drugim uczestniku konwersacji.

**Wiadomości** - odpowiada za pobieranie, przechowywanie oraz dynamiczne wyświetlanie wiadomości w ramach konwersacji. Poprzez API ma pełen dostęp do wiadomości użytkownika. Za pomocą WebSockets umożliwia wysyłanie oraz otrzymywanie wiadomości od innego użytkownika w czasie rzeczywistym. W przypadku gdy użytkownik otrzyma zaproszenie do czatu, moduł umożliwia zaakceptowanie lub odrzucenie zaproszenia. Wiadomości zawierają treść oraz czas ich wysłania, a informacje o nadawcy pozwalają na poprawne ich wyświetlenie.

**Plan użytkownika** - odpowiada za pobieranie, przechowywanie oraz dynamiczne wyświetlanie planu użytkownika z USOSa oraz po uwzględnieniu wymian. Dane wyświetlane są na podstawie wybranego rodzaju planu oraz wybranego dnia tygodnia.

**Plan przedmiotu** - odpowiada za pobieranie, przechowywanie oraz dynamiczne wyświetlanie planu danego przedmiotu. Grupy zajęciowe zawierają informacje o numerze grupy, typie zajęć, prowadzących zajęcia, miejscu odbywania zajęć oraz najczęściej występującym terminie zajęć.

**Nawigacja aplikacji mobilnej** - odpowiada za przełączanie między modułami

**Nawigacja aplikacji sieciowej** - zajmuje się przemieszczaniem użytkownika między widokami. Przetrzymuje dane o historii przeglądania strony w postaci stosu oraz zarządza przekierowaniami między nimi. Sprawdza czy użytkownik jest zalogowany przed wyświetleniem strony dostępnej tylko dla zalogowanego użytkownika. Obsługuje również przeciwny przypadek.

**Grupa zajęciowa** - odpowiada za pobieranie, przechowywanie oraz dynamiczne wyświetlanie danych o grupie zajęciowej. Dodatkowo umożliwia wysłanie i anulowanie prośby o wymianę oraz zaproszenie innych studentów do zespołu.

**Zespoły użytkownika** - odpowiada za pobieranie, przechowywanie oraz dynamiczne wyświetlanie zespołów. Umożliwia wysłanie żądania o opuszczenie zespołu, usunięcie innych członków z zespołu, wyszukiwanie i zapraszanie studentów do zespołu oraz zaakceptowanie i odrzucanie zaproszenia. Zespół zawiera informacje o identyfikatorze zespołu, nazwie przedmiotu, członkach zespołu oraz zaproszonych do zespołu.

**Wymiany użytkownika** - odpowiada za pobieranie, przechowywanie oraz dynamiczne wyświetlanie wymian. Umożliwia wysłanie żądania nadanie i usunięcie relacji miedzy wymianami. Wymiana zawiera informacje o identyfikatorze wymiany, nazwie przedmiotu, typie zajęć, numerach grupy obecnej i pożąданej, danych wymian, z którym wymiana jest w relacji.

### 3.3.2. Aplikacja serwerowa

**Baza danych** - Przechowuje informacje o użytkownikach, składach grup zajęciowych oraz możliwych wymianach. Jest zarządzana z wykorzystaniem biblioteki Entity Framework przy podejściu *Code First*. Schemat bazy jest przedstawiony na rysunku 3.5

**Serwer mailowy** - Korzystający z platformy SendGrid serwer, umożliwiający wysyłanie maili.

**Rest API** - Interfejs udostępniający pozostałym komponentom na odczyt części danych przechowywanych w bazie oraz ich przetwarzanie w sposób zdefiniowany w poniższych modułach.

**Użytkownicy** - Część serwera odpowiedzialna za zarządzanie informacjami o danym użytkowniku.

**Plan** - Moduł zajmujący się koordynacją zamian i tworzeniem nowych składów grup zajęciowych.

**Komunikator** - Moduł odpowiedzialny za funkcjonalności, uwzględniające bezpośrednią interakcję pomiędzy użytkownikami.

**Autoryzacja** - Moduł pozwalający na autoryzację użytkowników naszej aplikacji przy pomocy CAS USOS.

### 3.4. Komunikacja między komponentami

Do pełnego działania każdego komponentu potrzebna jest komunikacja między nimi. W projekcie występują następujące ścieżki komunikacji.

#### 3.4.1. Aplikacja sieciowa i mobilna z aplikacją serwerową

Z zasady API powinno być uniwersalne, dlatego komunikacja jest utrzymywana za pomocą uniwersalnego protokołu HTTP. Aplikacje wysyłają żądania o dane lub wysyłają dane do aplikacji serwerowej za pomocą metody GET, POST, PUT lub DELETE. W odpowiedzi serwer zwraca informacje o powodzeniu operacji związanej z żądaniem, o tym czy użytkownik wykonujący żądanie posiada odpowiednie uprawnienia, a jeśli jest potrzeba, zwraca dane w formacie JSON. Do zarządzania wiadomościami został użyty dwukierunkowy protokół WebSocket, dzięki któremu jeden komponent może informować na bieżąco o zmianach w drugim komponencie. Dokumentacja udostępnianego przez aplikację serwerową API została wygenerowana przy użyciu narzędzia Swagger. Uzyskane w ten sposób pliki udostępnione przez aplikację serwerową są dostępne w repozytorium z kodem.

#### 3.4.2. Aplikacja serwerowa z bazą danych

Komunikacja z bazą danych korzysta z biblioteki *Npgsql Entity Framework Core Provider*, która umożliwia korzystanie z Entity Framework w połączeniu z bazą danych PostgreSQL. Entity Framework umożliwia automatyczne stworzenie i aktualizowanie schematu bazy danych na podstawie kodu w języku C#. W naszej aplikacji zostało zastosowane podejście „code-first”, co oznacza że zarządzanie bazą danych jest przeprowadzane pośrednio, z wykorzystaniem biblioteki EF dla języka C# i powiązanych z nią narzędzi, bez konieczności bezpośredniego łączenia się z bazą danych.

#### 3.4.3. Aplikacja serwerowa z serwerem mailowym

Komunikacja z platformą SendGrid przebiega przy użyciu dedykowanej dla języka C# biblioteki o tej samej nazwie, która pozwala w prosty sposób wysłać maila z poziomu aplikacji serwerowej.

#### 3.4.4. Aplikacja serwerowa z systemem USOS

Komunikacja z systemem USOS przebiega za pośrednictwem udostępnianego przez ten system REST API. Autoryzacja całego systemu UsosFix jest powiązana z autoryzacją w systemie CAS

USOS. Użytkownik naszego systemu po zalogowaniu zezwala aplikacji serwerowej na dostęp do danych dotyczących obecnych planów zajęć, grup i przedmiotów. Dostęp ten jest ograniczony do danych dotyczących danego użytkownika, więc aplikacja UsosFix nie jest w stanie uzyskać dostępu do całościowego oglądu na plan wszystkich studentów Wydziału, a jedynie przedmioty i grupy, na które uczęszcza co najmniej jeden użytkownik UsosFix.

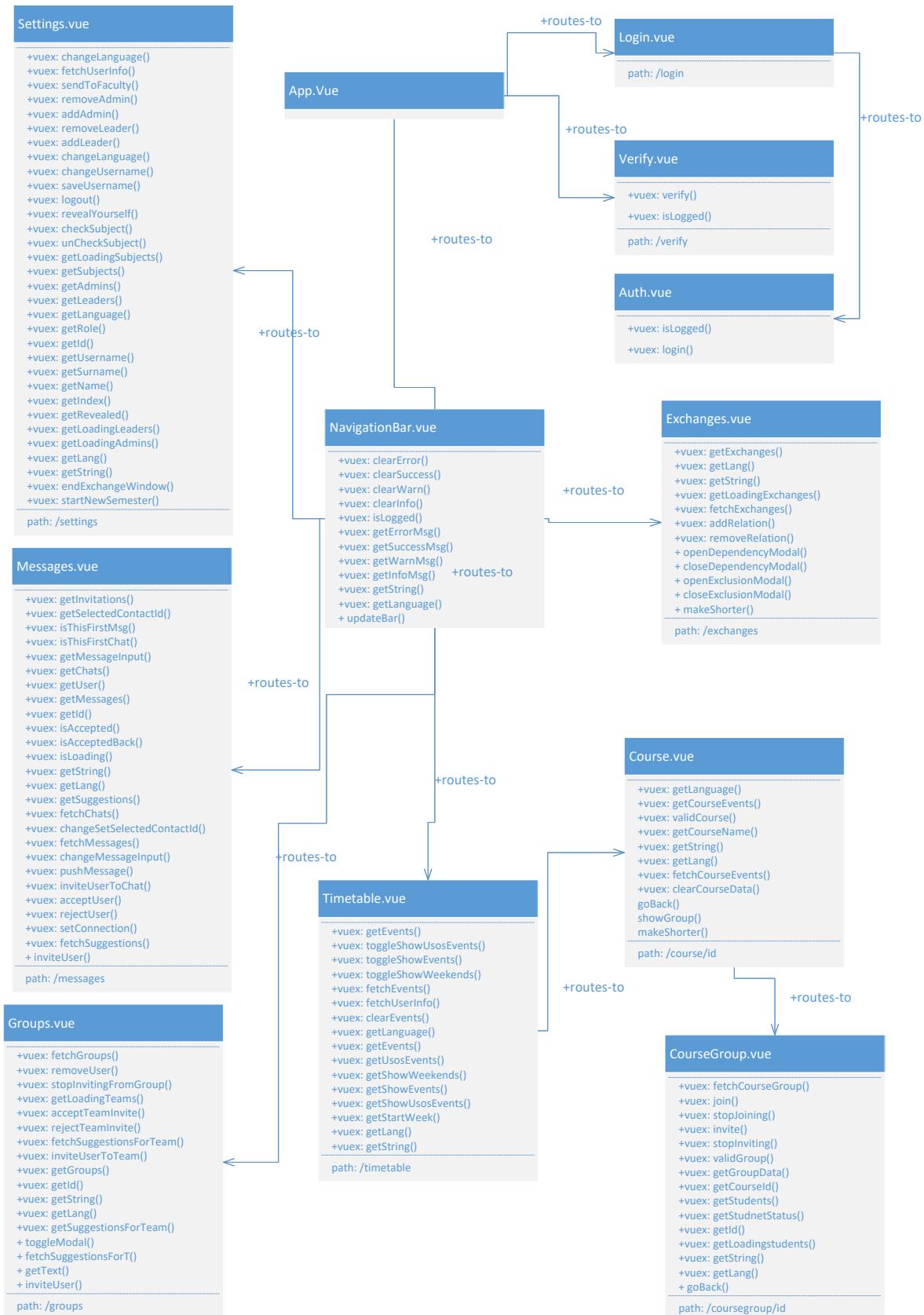
#### **3.4.5. Moduły z interfejsami aplikacji sieciowej**

Aby zachować bezpieczeństwo i schemat kontrolera stanu, do komunikacji zostały użyte specjalne funkcje, które definiują dany moduł. Wszelkie dane, które dany moduł przetrzymuje można otrzymać poprzez zbiór funkcji **getters**. Wszystkie akcje związane z modułem, które użytkownik może wykonać są zdefiniowane w grupie funkcji **actions**.

#### **3.4.6. Klasy aktywności z interfejsami aplikacji mobilnej**

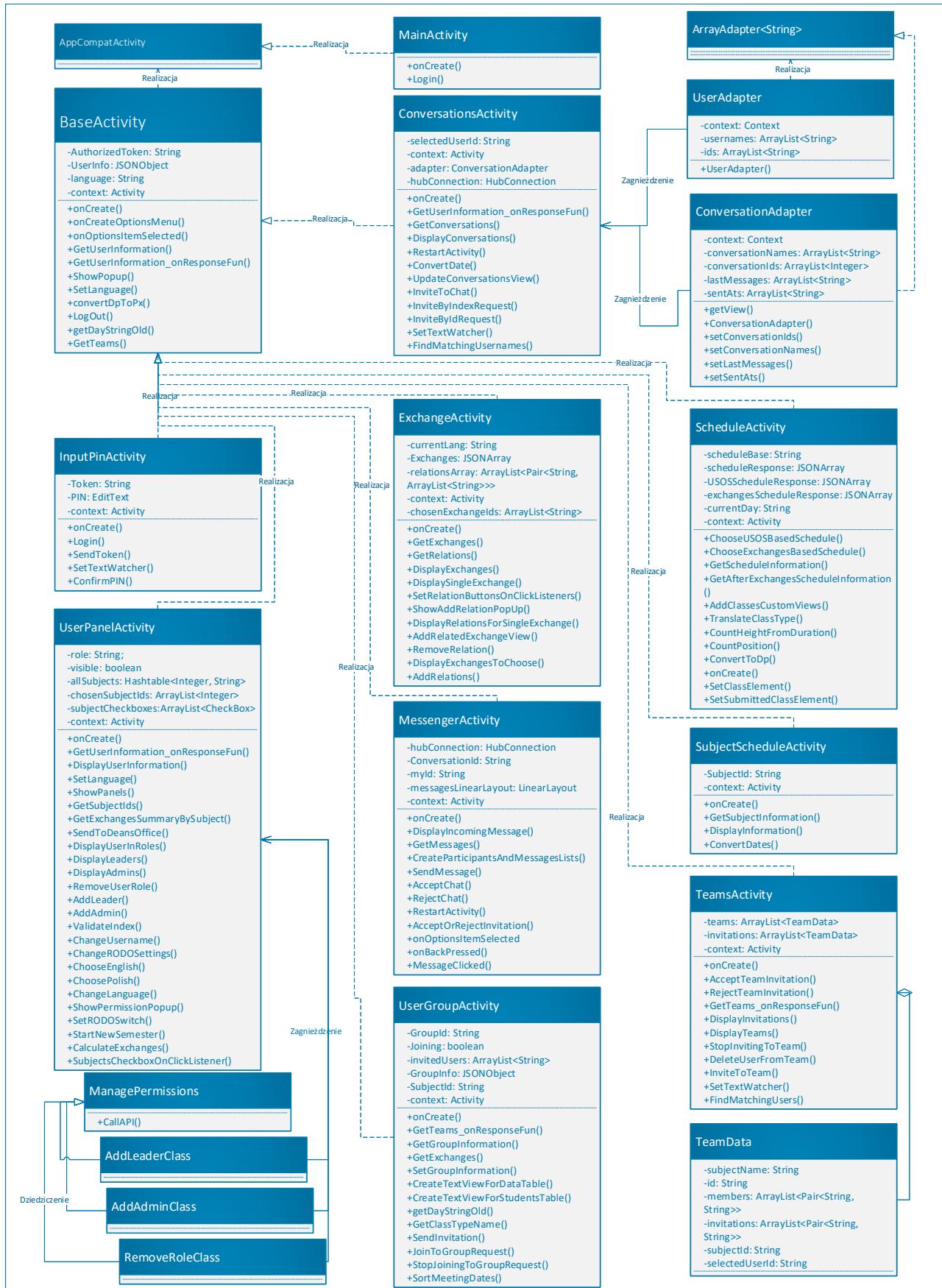
Komunikacja między klasami aktywności, a interfejsami aplikacji mobilnej występuje na dwa sposoby. Komunikacja pomiędzy klasami aktywności, a interfejsami działa na podstawie modyfikowania poszczególnych instancji widoków lub fragmentów widoków. Z kolei komunikacja na linii interfejsy → klasy aktywności działa na podstawie obserwatorów zdarzeń, ustawianych albo bezpośrednio w interfejsie, albo na odpowiednich elementach poprzez klasy aktywności.

### 3.4. KOMUNIKACJA MIEDZY KOMPONENTAMI



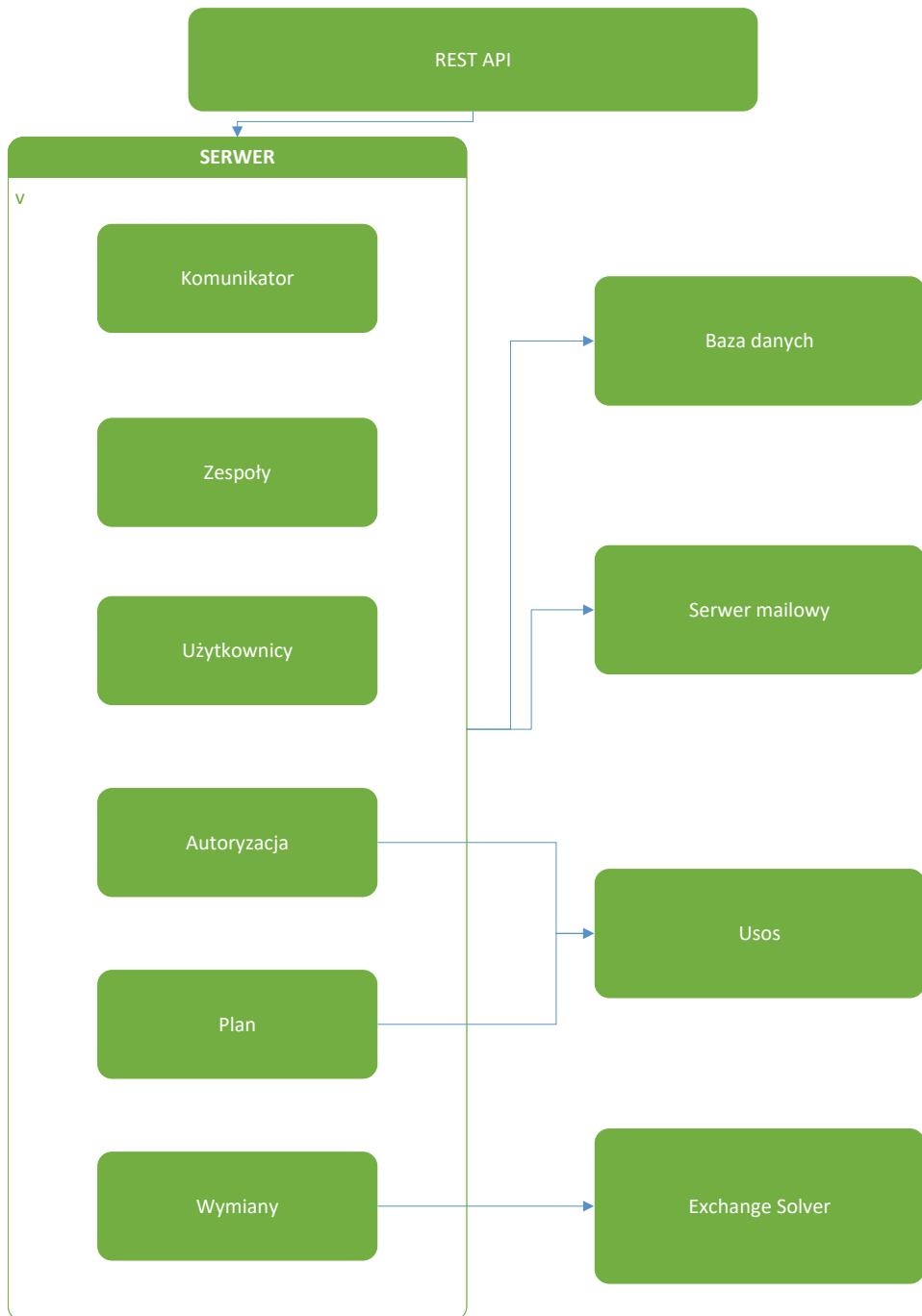
Rysunek 3.2: Diagram interfejsów

### 3. PROJEKT ARCHITEKTURY



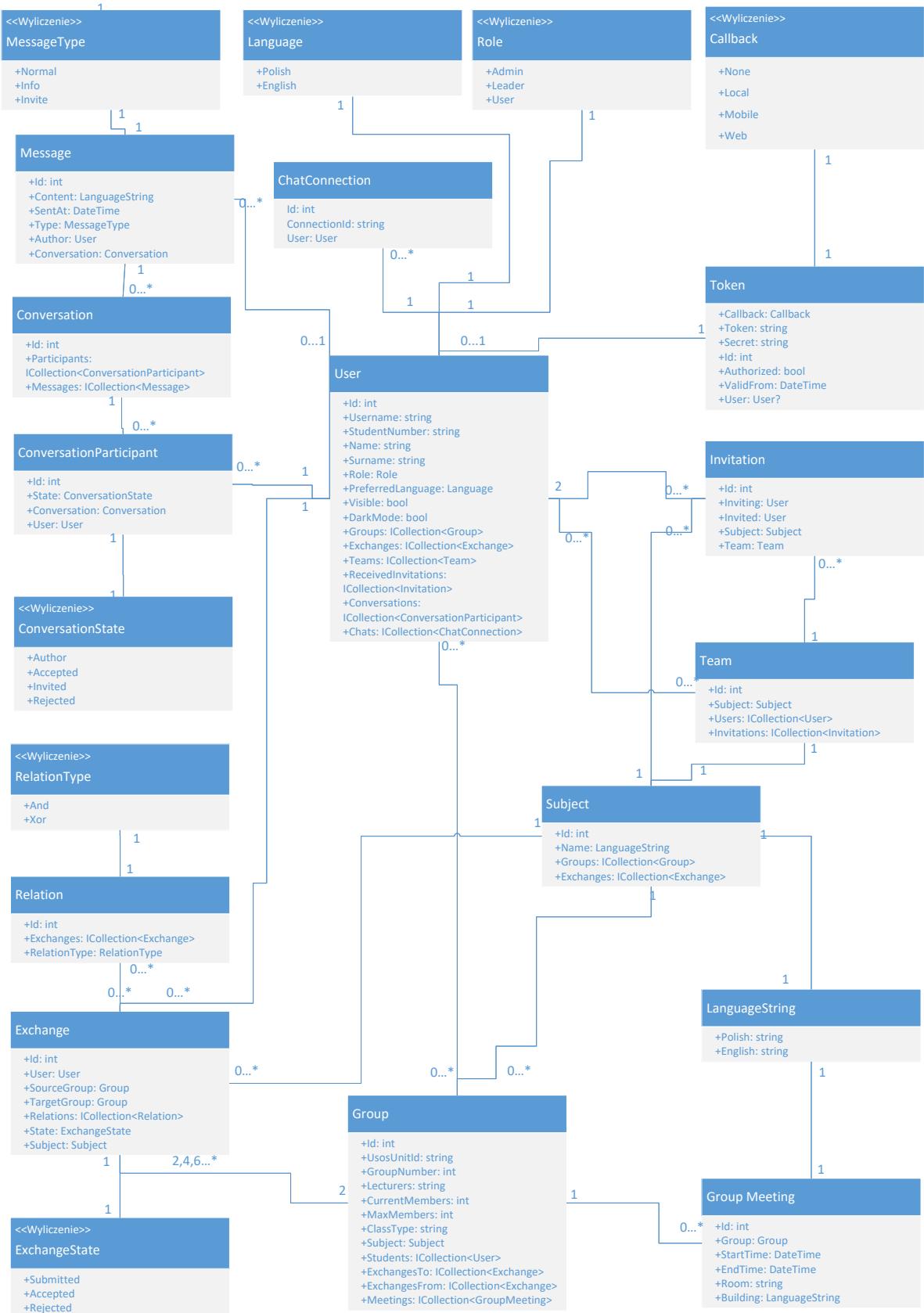
Rysunek 3.3: Diagram klas aplikacji mobilnej

### 3.4. KOMUNIKACJA MIEDZY KOMPONENTAMI



Rysunek 3.4: Zależności pomiędzy modułami

### 3. PROJEKT ARCHITEKTURY



Rysunek 3.5: Klasy i zależności, z których utworzony został schemat bazy danych.

## **4. Interfejs użytkownika**

### **4.1. Logo**



Rysunek 4.1: Logo aplikacji UsosFix

Autorką użytego logo, widocznego na rysunku 4.1, oraz projektantką kilku widoków jest Marta Kozłowska, studentka Architektury wnętrz na Wydziale Sztuk Pięknych Uniwersytetu Mikołaja Kopernika w Toruniu oraz Graphics Designer w Phibox SA.

### **4.2. Kolory aplikacji mobilnej**

Aplikacja mobilna stworzona została na sześciu podstawowych kolorach. Kolory przedstawione na rysunku 4.2 stosowane są jako odpowiednio:

- tło przycisków służących do usuwania odpowiednich elementów,
- tło przycisków służących do opuszczania zespołu,
- podstawowy kolor aplikacji używany jako tło ramki większości elementów,

- podstawowy kolor tła przycisków,
- podstawowy kolor tła pól tekstowych,
- podstawowy kolor tła tabel.



Rysunek 4.2: Podstawowe kolory aplikacji mobilnej

### 4.3. Kolory elementów planu zajęć

W celu zwiększenia czytelności różne rodzaje zajęć zostały różnie oznaczone w myśl kolorystyki analogicznej do kolorystyki aplikacji Mobilny USOS. Kolory są zastosowane odpowiednio do wykładów, ćwiczeń, laboratorium oraz zajęć projektowych i są zaprezentowane odpowiednio na rysunku 4.3a.

W planie uwzględniającym zgłoszone wymiany zostały również zastosowane oznaczenia wizualne w celu rozróżnienia stanu wymiany:

- Wymiana zgłoszona - ciemniejsze paski pokrywające element w 50% pod kątem 45 stopni do pionu w kolorach pokazanych na rysunku 4.3b,
- Wymiana udana - brak oznaczenia,
- Wymiana nieudana - szare tło z czerwonym X na pierwszym planie.

#### 4.4. CZCIONKI



(a) Standardowe kolory elementów planu zajęć



(b) Ciemniejsze kolory elementów planu zajęć

Rysunek 4.3: Porównanie kolorów elementów planu zajęć

#### 4.4. Czcionki

Wszystkie teksty w aplikacji mobilnej oraz aplikacji sieciowej są napisane przy użyciu rodziny fontów Barlow (Rysunek 4.4).

Barlow Bold używane jest do nagłówków, Barlow Medium jest podstawową czcionką używanym w większości napisów, Barlow Regular jest używany w edytowalnych polach tekstowych, a Barlow Semibold w niektórych nazwach danych.

**Barlow Bold**      **Barlow Medium**  
**Barlow Regular**    **Barlow Semibold**

Rysunek 4.4: Czcionki aplikacji mobilnej

## 4.5. Interfejsy aplikacji mobilnej i sieciowej

### 4.5.1. Pasek nawigacji/Menu

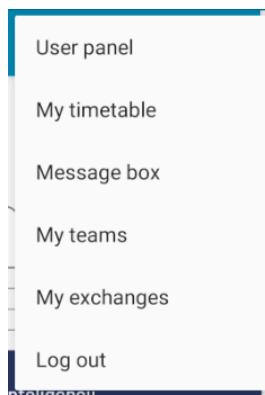
Pasek nawigacji aplikacji sieciowej oraz menu aplikacji mobilnej służą do nawigacji między pięcioma głównymi widokami: Planem użytkownika, Zespołami, Panelem użytkownika, Wymianami oraz Wiadomościami. Po wybraniu elementu odpowiadającego danemu widokowi, aplikacja przełączy się na wybrany widok.

#### Pasek nawigacji - tylko aplikacja sieciowa

Pasek nawigacji widoczny jest dla zalogowanych użytkowników w formie paska na górze ekranu. Po wybraniu danego elementu paska zostaje zmieniony URL aplikacji na odpowiadający wybranemu widokowi oraz zostanie dodany wpis do historii przeglądarki o przejściu na nowy adres. Jak na szczeźcie rysunku 4.6, każdy element tego interfejsu składa się z ikony i opisu. Aktywny widok jest dodatkowo podświetlony. W przypadku małego ekranu (Rysunek 4.7) elementy interfejsu są oznaczone tylko ikonami dla lepszej czytelności.

#### Menu - tylko aplikacja mobilna

Menu jest widoczne dla zalogowanych użytkowników w formie trzech kropek na pasku u góry ekranu i rozwijane na kliknięcie użytkownika. Poza wcześniejszymi wymienionymi już elementami posiada przycisk służący do wylogowania (Rysunek 4.5).



Rysunek 4.5: Rozwinięte menu aplikacji mobilnej

### 4.5.2. Plan zajęć

Widok, który wyświetla plan użytkownika w taki sam sposób jak USOS w przypadku aplikacji sieciowej oraz jak Mobilny USOS w przypadku aplikacji mobilnej. Możliwe jest wyświetlanie obecnego planu z USOSa oraz planu z uwzględnieniem potencjalnych zmian w grupach i zmianami

#### 4.5. INTERFEJSY APLIKACJI MOBILNEJ I SIECIOWEJ

już dokonanymi. Po kliknięciu na grupę w planie, następuje przekierowanie do widoku planu przedmiotu, w ramach którego prowadzona jest grupa.

##### Aplikacja sieciowa

W oknie pełnoekranowym plan jest wyświetlany jako pełny tydzień (Rysunek 4.6). Po lewej stronie na pionowej osi są umieszczone godziny w przedziałach 30-minutowych. Możliwe jest określenie czy w dniach tygodnia ma się wyświetlać również sobota i niedziela czy tylko zakres dni poniedziałek - piątek. Dodatkowo aktualny dzień tygodnia jest lekko wyróżniany kolorem w czasie rzeczywistym.

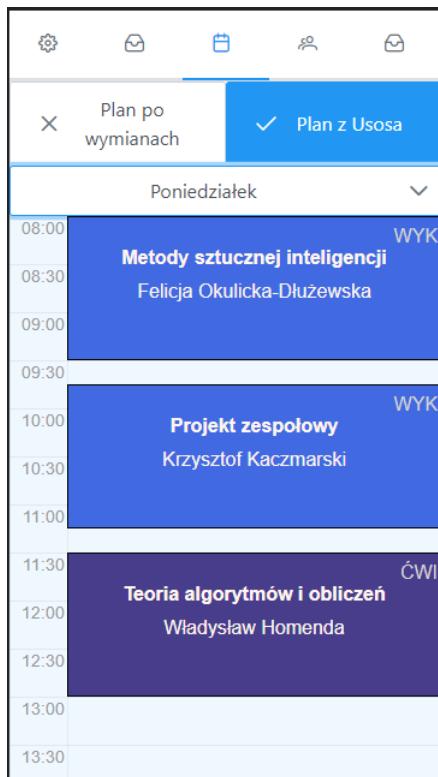
Na urządzeniach mobilnych jednocześnie może być wyświetlony tylko jeden z siedmiu dni w tygodniu (Rysunek 4.7). Im większy ekran tym większy poziom szczegółowości elementów na planie. Szczególny, które są przy większych ekranach to:

- godzina rozpoczęcia i zakończenia zajęć w formacie od-do,
- rodzaj zajęć jako skrót trzyliterowy,
- nazwa przedmiotu,
- prowadzący zajęcia,
- miejsce przeprowadzenia zajęć.

Przy mniejszych ekranach nie jest wyświetlane miejsce przeprowadzenia zajęć oraz dane dotyczące godzin rozpoczęcia i zakończenia.

Panel użytkownika	Moja wymiana	Mój plan	Moje zespoły	Wiadomości			
Poniedziałek	Wtorek	Sroda	Czwartek	Piątek	Sobota	Niedziela	Weekendy
08:00 - 09:30 <b>Metody sztucznej inteligencji</b> Felicia Okulicko-Dłużewska Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE	08:00 - 09:30 <b>Metody sztucznej inteligencji</b> Felicia Okulicko-Dłużewska Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE						
09:45 - 11:15 <b>Projekt zespołowy</b> Krysztof Kaczmarski Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE	10:00 - 11:30 <b>WYK</b>						
11:30 - 13:00 <b>Teoria algorytmów i obliczeń</b> Włodzimierz Homenda Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE	11:30 - 13:00 <b>Teoria algorytmów i obliczeń</b> Agnieszka Jastrzębska Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE	11:30 - 13:00 <b>Teoria algorytmów i obliczeń</b> Włodzimierz Homenda Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE	12:15 - 14:45 <b>Seminarium dyplomowe</b> Włodzimierz Homenda Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE	12:15 - 14:45 <b>Projekt zespołowy</b> Paweł Kotowski Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE	14:00 - 15:30 <b>Seminarium dyplomowe</b> Włodzimierz Homenda Kampus Przychodnia TEAMS	14:00 - 15:30 <b>Projekt zespołowy</b> Paweł Kotowski Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE	15:30 - 17:00 <b>Teoria algorytmów i obliczeń</b> Włodzimierz Homenda Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE
13:00 - 14:45 <b>Seminarium dyplomowe</b> Włodzimierz Homenda Kampus Przychodnia TEAMS	13:00 - 14:45 <b>Seminarium dyplomowe</b> Włodzimierz Homenda Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE	13:00 - 14:45 <b>Seminarium dyplomowe</b> Włodzimierz Homenda Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE	15:30 - 17:00 <b>Projekt zespołowy</b> Paweł Kotowski Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE	15:30 - 17:00 <b>Projekt zespołowy</b> Paweł Kotowski Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE	17:00 - 18:30 <b>Teoria algorytmów i obliczeń</b> Włodzimierz Homenda Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE	17:00 - 18:30 <b>Teoria algorytmów i obliczeń</b> Włodzimierz Homenda Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE	18:30 - 19:45 <b>Metody sztucznej inteligencji</b> Felicia Okulicko-Dłużewska Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE
14:45 - 16:15 <b>Metody sztucznej inteligencji</b> Felicia Okulicko-Dłużewska Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE							
16:30							
18:00							
18:30							
20:00							
20:30							
22:00							
23:30							

Rysunek 4.6: Pasek nawigacji i plan zajęć w aplikacji sieciowej na dużym ekranie

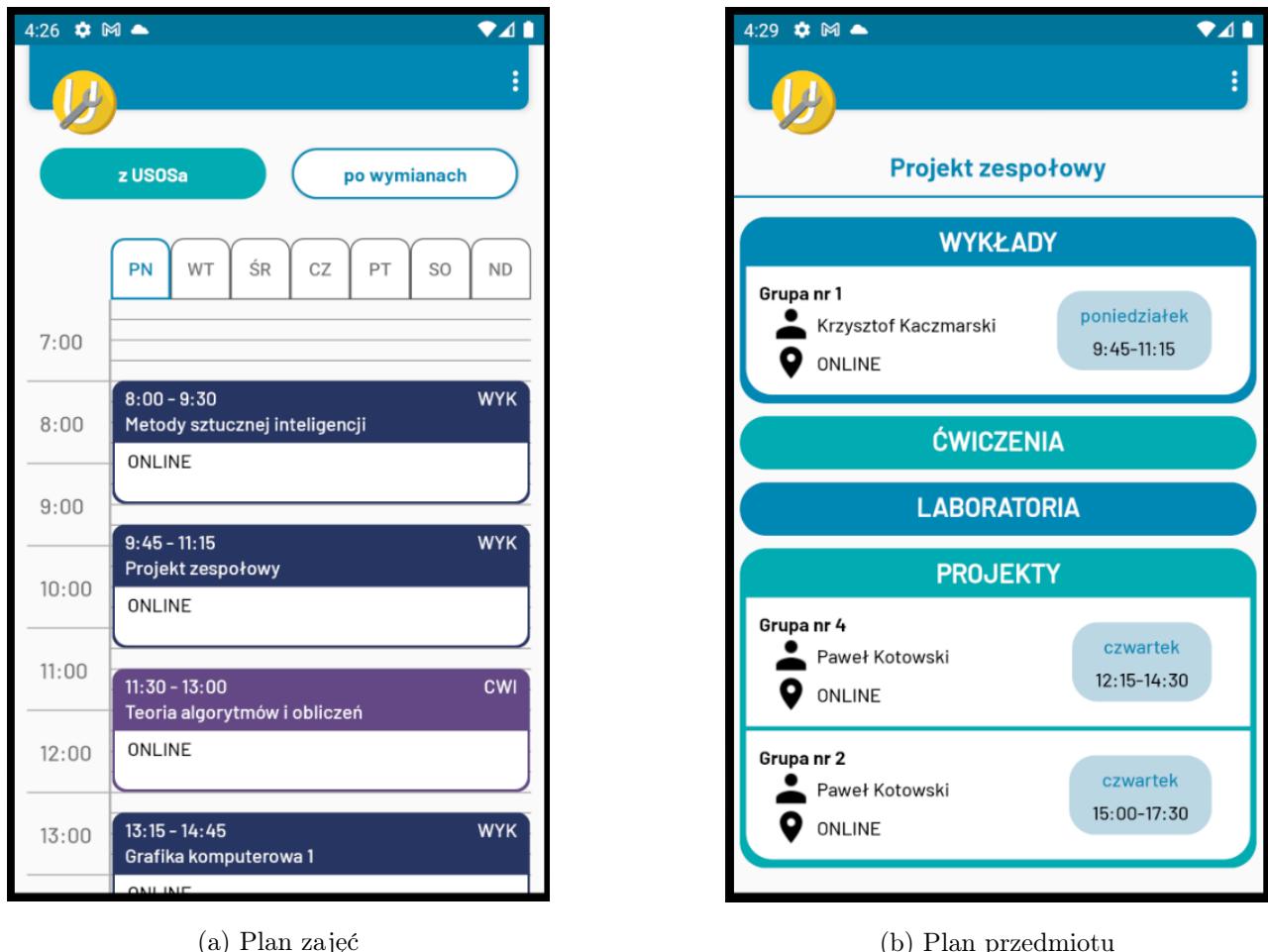


Rysunek 4.7: Pasek nawigacji i plan zajęć w aplikacji sieciowej na małym ekranie

### Aplikacja mobilna

W aplikacji mobilnej plan zajęć wyświetlany jest podobnie jak w aplikacji sieciowej na małych ekranach (Rysunek 4.8a). Jednocześnie widoczny jest tylko jeden z siedmiu dni tygodnia. Przyciski znajdujące się na górze ekranu umożliwiają zmianę rodzaju wyświetlanego planu. Przycisk odpowiadający rodzajowi aktualnie wyświetlanego planu jest wy pełniony kolorem oraz zawiera biały tekst. Zakładki ze skrótami dni tygodnia służą do zmiany wyświetlanego dnia. W kolumnie po lewej stronie znajdują się kolejne godziny w ciągu dnia. Przestrzeń, w której umieszczone są zajęcia, podzielona jest poziomymi liniami w odstępach odpowiadającym 15 minutom. Poszczególne rodzaje zajęć mają różne kolory tła zgodnie z tym, co zostało napisane w rozdziale 4.3. Informacje wyświetlane w pojedynczym elemencie planu są takie same jak w aplikacji sieciowej z wyjątkiem informacji o prowadzącym zajęcia.

#### 4.5. INTERFEJSY APLIKACJI MOBILNEJ I SIECIOWEJ



Rysunek 4.8: Widoki planów w aplikacji mobilnej

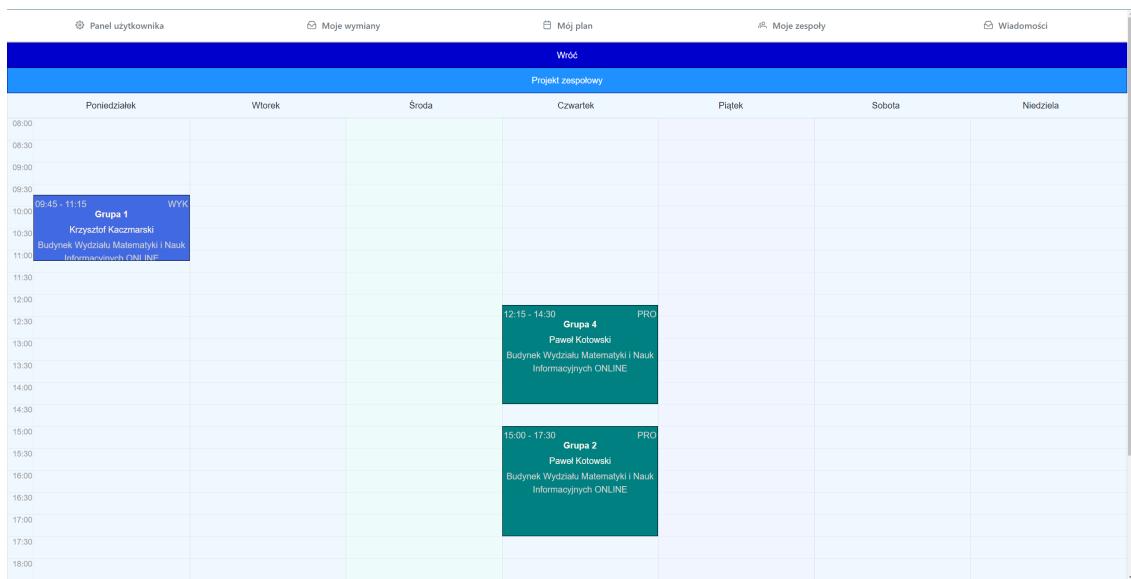
#### 4.5.3. Plan przedmiotu

##### Aplikacja sieciowa

Działanie interfejsu (Rysunek 4.9) jest analogiczne do planu zajęć z następującymi różnicami :

- między planem, a paskiem nawigacji znajduje się szeroki przycisk służący do powrotu do planu użytkownika,
- między planem, a paskiem nawigacji znajduje się pasek z nazwą przedmiotu,
- w elementach planu zamiast nazwy przedmiotu jest wypisany numer grupy ćwiczeniowej dla tych zajęć,
- po kliknięciu na grupę, zostaje wysłana do modułów prośba o przekierowanie na widok danej grupy.

## 4. INTERFEJS UŻYTKOWNIKA



Rysunek 4.9: Plan przedmiotu w aplikacji sieciowej

### Aplikacja mobilna

Widok planu przedmiotu jest podzielony na 4 listy - każda odpowiadająca jednemu rodzajowi zajęć (Rysunek 4.8b). Jeśli w ramach danego przedmiotu nie odbywają się zajęcia jednego rodzaju, to lista pozostaje pusta. Każdy element reprezentujący grupę zajęciową zawiera informacje o:

- numerze grupy,
- prowadzącym zajęcia,
- miejscu prowadzenia zajęć,
- terminie odbywania się zajęć - jeśli zajęcia odbywają się w różnych terminach, przedstawowany jest przeważający termin.

Naciśnięcie na grupę zajęciową powoduje przekierowanie na widok wybranej grupy zajęciowej.

#### 4.5.4. Grupa zajęciowa

Widok pokazujący dane na temat danej grupy zajęciowej (nazwa przedmiotu, numer grupy, rodzaj zajęć, osoba prowadząca zajęcia, miejsce, daty spotkań, liczbę osób maksymalnie mogących uczęszczać na zajęcia, liczbę osób obecnie zapisanych na zajęcia).

## 4.5. INTERFEJSY APLIKACJI MOBILNEJ I SIECIOWEJ

### Aplikacja sieciowa

Dane są wyświetlane w formie tabelki dwukolumnowej (Rysunek 4.10) (Rysunek 4.11).

Analogicznie do planu przedmiotu, w górnej części widoku znajduje się przycisk do powrotu do planu przedmiotu.

Pod tabelką z danymi znajduje się przycisk, który pozwala na zgłoszenie chęci uczęszczania do tej grupy, cofnięcia takiego zgłoszenia lub informujący użytkownika o tym, że już jest zapisany do danej grupy.

Wyświetlana jest również lista osób zapisanych do tej grupy w postaci unikalnych pseudonimów. Danego uczestnika można zaprosić do zespołu klikając na przycisk w tym samym wierszu tabelki, w którym znajduje się dana osoba. W przypadku osoby już zaproszonej, można przyciskiem zaprzestać jej zapraszania. W przypadku osoby już znajdującej się w zespole użytkownika, jest wyświetlony o tym komunikat zamiast przycisku. Użytkownik aktualnie zalogowany zostanie wyróżniony podświetleniem i nie będzie mógł zaprosić samego siebie do zespołu.

The screenshot shows a user interface for a network application. At the top, there is a navigation bar with links: 'Panel użytkownika', 'Moje wymiany', 'Mój plan', 'Moje zespoły', and 'Wiadomości'. Below the navigation bar, there is a 'Powrót' (Return) button. The main content area is divided into two columns. The left column contains a sidebar with the following items: 'Przedmiot' (Projekt zespołowy), 'Grupa' (Grupa 1), 'Rodzaj zajęć' (Wykład), 'Prowadzący' (Krzysztof Kaczmarski), 'Miejsce' (Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE), 'Data spotkania' (with a list of dates from 21-12-2020 to 25-01-2021), 'Limit studentów' (0), and 'Liczba studentów' (3). The right column displays information about the group: 'Projekt zespołowy', 'Grupa 1', 'Wykład', 'Krzysztof Kaczmarski', 'Budynek Wydziału Matematyki i Nauk Informacyjnych ONLINE', and a list of meeting dates. Below this, a blue button says 'Jesteś w tej grupie'. At the bottom, there is a section titled 'Uczestnicy' (Participants) with three entries: '1 Remigiusz#0001' with a 'Przejdź do zainteresowania' (Go to interest) button, '2 Lidka#0002' with a 'Zapraszaj użytkownika do zespołu' (Invite user to team) button, and '3 Axolotls2#0003(ty)' with a 'Już jest w twoim zespole' (Already in your team) button.

Rysunek 4.10: Widok grupy zajęciowej w aplikacji sieciowej na dużym ekranie

### Aplikacja mobilna

Widoku w aplikacji mobilnej (Rysunek 4.12) nie można przewijać. Wszystkie elementy są umiejscowione statycznie, jednak ekran nie jest w stanie pomieścić wszystkich informacji

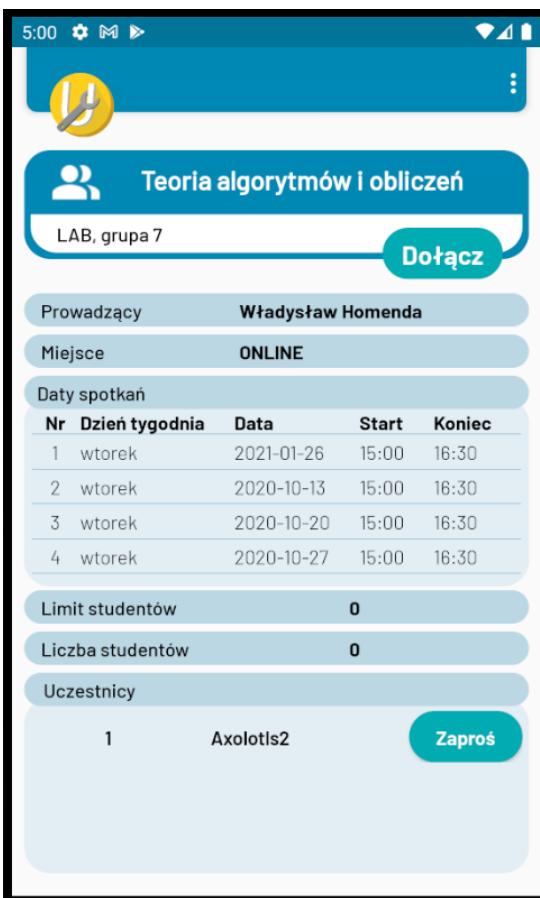
#### 4. INTERFEJS UŻYTKOWNIKA

Rysunek 4.11: Widok grupy zajęciowej w aplikacji sieciowej na małym ekranie

na raz. Dlatego też tabela z terminami spotkań oraz tabela z uczestnikami zajęć można przewijać wewnątrz nich. Poza wcześniej opisanymi informacjami w widoku znajdują się dwa rodzaje przycisków:

1. Przycisk służący do dołączenia do grupy (stworzenia wymiany) oraz przycisk anulujący wymianę. Oba przyciski znajdują się w tym samym miejscu u góry ekranu. Jeśli użytkownik już należy do danej grupy w USOSie, żaden z tych przycisków nie jest wyświetlany. Jeśli użytkownik nie należy do danej grupy w USOSie i nie zgłosił jeszcze prośby o dołączenie do tej grupy, wyświetlany jest przycisk służący stworzeniu wymiany. Jeśli użytkownik nie należy do danej grupy w USOSie i zgłosił już prośbę o dołączenie do tej grupy, wyświetlany jest przycisk służący anulowaniu wymiany. Kliknięcie w te przyciski znajduje swoje odzwierciedlenie w widoku wymian.
2. Przyciski pozwalające na zaproszenie uczestników do zespołu znajdujące się przy każdym uczestniku w tabeli z uczestnikami (z wyjątkiem zalogowanego użytkownika). Jeśli dany uczestnik jest już zaproszony do zespołu z danego przedmiotu, to na przycisku widnieje odpowiedni napis, a przycisk jest nieaktywny.

## 4.5. INTERFEJSY APLIKACJI MOBILNEJ I SIECIOWEJ



Rysunek 4.12: Widok grupy zajęciowej w aplikacji mobilnej

### 4.5.5. Zespoły użytkownika

Widok, w którym użytkownik może zobaczyć listę zespołów, do których należy lub do których został zaproszony. Interfejs zespołów składa się z listy kafelków. Każdy kafelek prezentuje zespół, do którego należy użytkownik lub zespół, do którego zaproszony jest użytkownik.

#### Zespół, którego uczestnikiem jest zalogowany użytkownik

Każdy taki kafelek zawiera listę osób należących do grupy oraz listę osób zaproszonych. Osoby zaproszone są wyróżnione wyszarzeniem i informacją o ich stanie podaną w nawiasie za ich pseudonimem. U góry kafelka znajduje się nazwa przedmiotu w ramach którego stworzony jest zespół. Poniżej znajduje się pole tekstowe oraz przycisk, które służą do zapraszania do zespołu dodatkowych osób. Po wpisaniu do tego pola prefiksu pojawia się lista z użytkownikami, których nazwa zaczyna się od wpisanego prefiksu, którzy uczęszczają na dany przedmiot oraz którzy jeszcze nie należą do żadnego zespołu w ramach tego przedmiotu. Po wybraniu opcji z listy, przycisk do zapraszania staje się aktywny, a po jego naciśnięciu zaprasza wybranego użytkownika do zespołu.

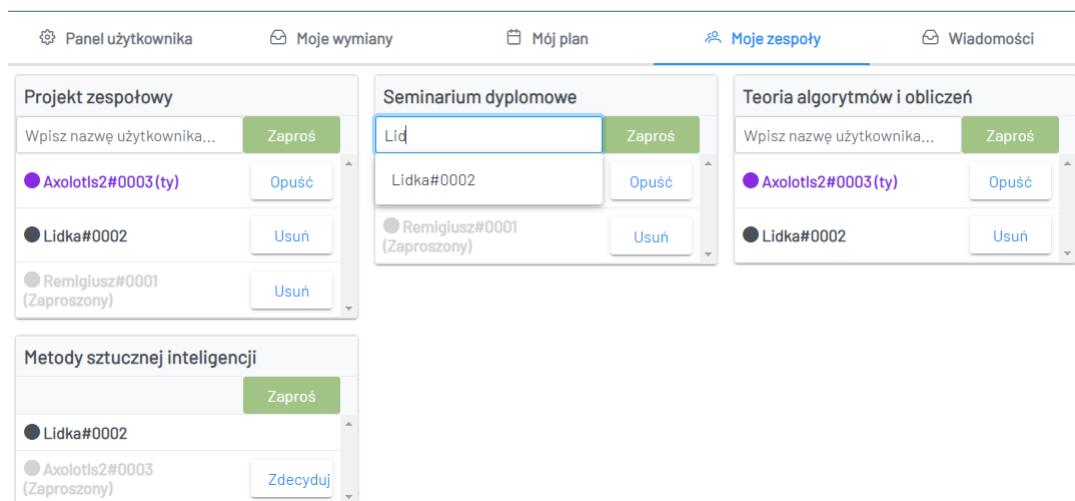
### Zespół, do którego zaproszony jest zalogowany użytkownik

Ten rodzaj kafelka zawiera elementy, które zostały opisane w kafelku pierwszego rodzaju, jednak pole tekstowe oraz przycisk do zapraszania w aplikacji sieciowej są nieaktywne, a w aplikacji mobilnej w ogóle ich nie ma.

### Aplikacja sieciowa

W aplikacji sieciowej lista kafelków jest wyświetlana w jednej kolumnie (Rysunek 4.15a) lub w rzędach (Rysunek 4.13) w zależności od wielkości ekranu. Osoba obecnie zalogowana jest wyróżniona na liście członków zespołu kolorem fioletowym. Przy każdym użytkowniku znajduje się przycisk z akcją, jaką osoba zalogowana może wykonać względem danej osoby. Akcje są opisane w tablicy 4.1. Jeśli zalogowany użytkownik jest zaproszony do zespołu, to nie może wykonać akcji względem innych uczestników, ani zapraszać nowych osób, dopóki nie zaakceptuje zaproszenia.

Aby wykonać daną akcję, użytkownik musi potwierdzić jej wykonanie w oknie modalnym (Rysunek 4.14).

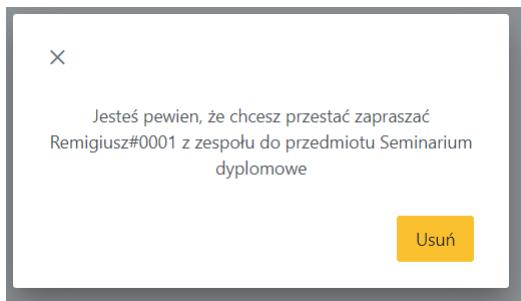


Rysunek 4.13: Widok zespołów użytkownika na dużym ekranie

Stan osoby względem której jest wykonywana akcja	Akcja
Osoba zalogowana	Opuścić grupę
Osoba zaproszona	Usunąć zaproszenie do zespołu
Inny uczestnik	Usunąć uczestnika z zespołu

Tablica 4.1: Akcje, jakie może wykonać zalogowany użytkownik względem innych uczestników zespołu

#### 4.5. INTERFEJSY APLIKACJI MOBILNEJ I SIECIOWEJ



Rysunek 4.14: Potwierdzenie wykonania akcji

#### Aplikacja mobilna

W aplikacji mobilnej lista kafelków zawsze wyświetlna jest w jednej kolumnie, która zajmuje całą szerokość ekranu (Rysunek 4.15b). Wspomniane wyżej wyróżnienie osób zaproszonych do zespołu w aplikacji mobilnej uwypuklone jest jeszcze bardziej poprzez podzielenie członków zespołu oraz osób zaproszonych na dwie tabele. Kafelki dzielą się na dwa rodzaje:

##### **Zespół, którego uczestnikiem jest zalogowany użytkownik**

Każdy taki kafelek, poza wcześniej opisanymi elementami, w prawym górnym rogu posiada przycisk pozwalający na opuszczenie zespołu. Przy nazwie zalogowanego użytkownika nie znajduje się żaden przycisk. Jeśli suma liczby członków i liczby osób zaproszonych jest równa 2, to przy żadnym członku ani zaproszonym do zespołu nie jest wyświetlany przycisk. W pozostałych przypadkach przy wszystkich członkach i zaproszonych poza zalogowanym użytkownikiem, znajduje się przycisk służący odpowiednio do usunięcia członka z zespołu oraz do anulowania zaproszenia do zespołu.

##### **Zespół, do którego zaproszony jest zalogowany użytkownik**

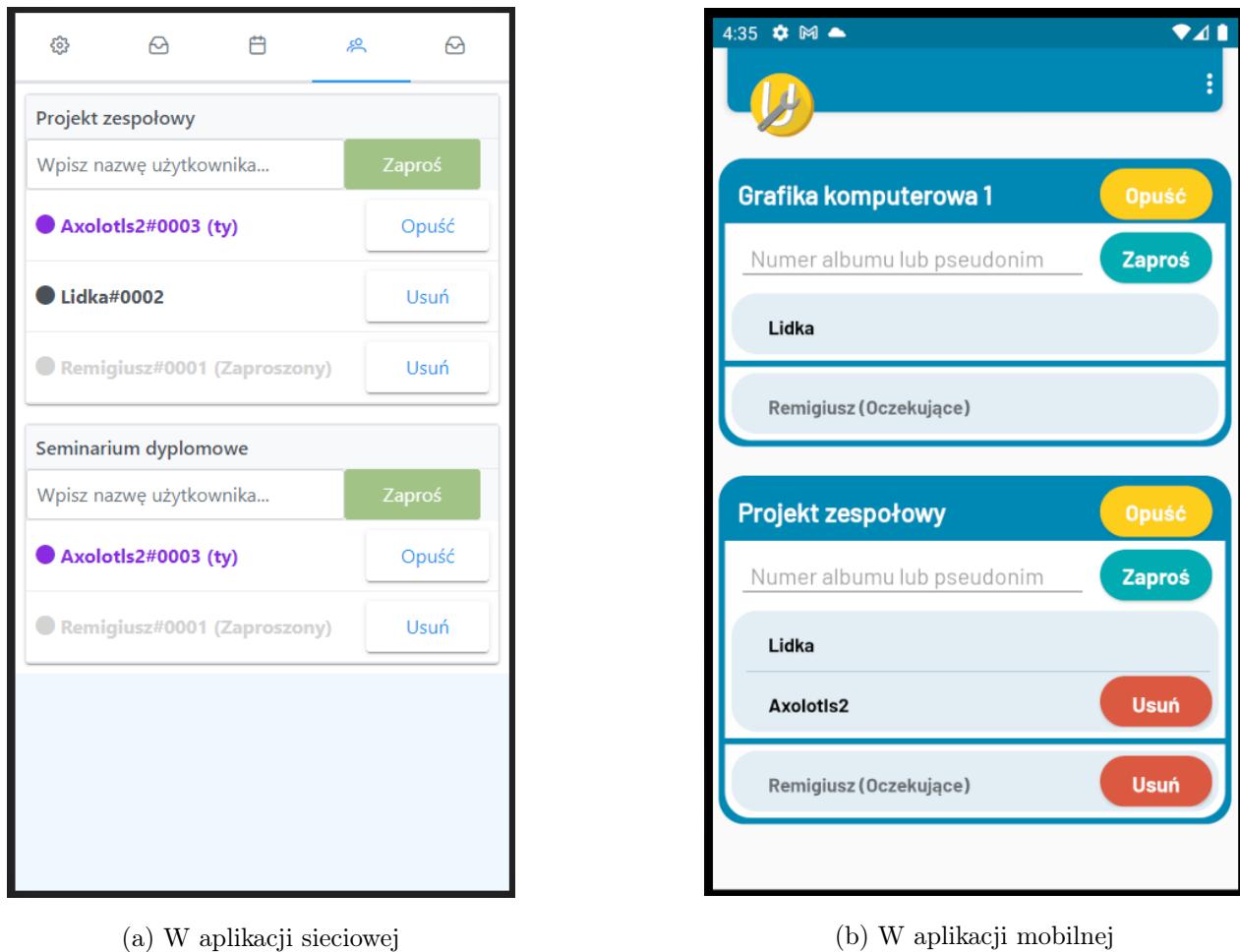
W tym rodzaju kafelka nie znajduje się żaden opisany wcześniej przycisk (Rysunek 4.16). U dołu kafelka znajdują się dwa przyciski - jeden służący do zaakceptowania zaproszenia oraz jeden służący do odrzucenia zaproszenia. Po zaakceptowaniu zaproszenia kafelek zmienia się na kafelek prezentujący zespół, którego uczestnikiem jest zalogowany użytkownik. Po odrzuceniu zaproszenia - znika.

#### 4.5.6. Wymiany użytkownika

#### Aplikacja sieciowa

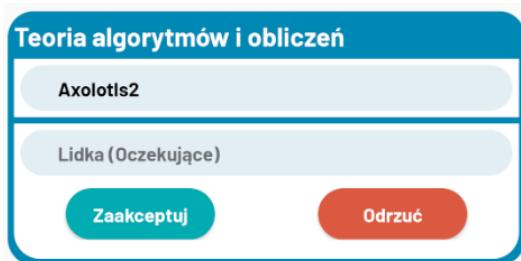
Interfejs składa się z rozwijanej tabeli z wymianami użytkownika. Elementy tabeli składają się z przycisku do rozwinięcia elementu, obecnej grupy zajęciowej użytkownika, docelowej grupy zajęciowej, przycisku do nadania relacji powiązania oraz przycisku do nadania

#### 4. INTERFEJS UŻYTKOWNIKA



Rysunek 4.15: Widoki zespołów użytkownika na małych ekranach

#### 4.5. INTERFEJSY APLIKACJI MOBILNEJ I SIECIOWEJ



Rysunek 4.16: Widok zespołu, do którego zaproszony jest użytkownik w aplikacji mobilnej

relacji wykluczenia (Rysunek 4.17). W widoku na mniejszym ekranie nazwy przedmiotów i grup zostały skrócone do pierwszych liter dla lepszej czytelności (Rysunek 4.18).

Po rozwinięciu elementu pokazują się relacje nadane dla danej wymiany w postaci relacji oraz wymiany z którą nastąpiło powiązanie. W przypadku mniejszych ekranów nazwa relacji została zastąpiona ikoną.

Z	Do		
▼ Metody sztucznej inteligencji - Grupa 5	→ Metody sztucznej inteligencji - Grupa 5	Dodaj relację powiązania	Dodaj relację wykluczenia
Powiązane z:	Projekt zespołowy - Grupa 4	→ Projekt zespołowy - Grupa 2	Anuluj relację
Powiązane z:	Seminarium dyplomowe - Grupa 3	→ Seminarium dyplomowe - Grupa 1	Anuluj relację
▶ Projekt zespołowy - Grupa 4	→ Projekt zespołowy - Grupa 4	Dodaj relację powiązania	Dodaj relację wykluczenia
▶ Teoria algorytmów i obliczeń - Grupa 7	→ Teoria algorytmów i obliczeń - Grupa 7	Dodaj relację powiązania	Dodaj relację wykluczenia
▶ Seminarium dyplomowe - Grupa 3	→ Seminarium dyplomowe - Grupa 3	Dodaj relację powiązania	Dodaj relację wykluczenia

Rysunek 4.17: Widok wymian użytkownika na dużym ekranie

## Aplikacja mobilna

Widok prezentuje listę wymian użytkownika wraz z relacjami między wymianami (Rysunek 4.19). Element prezentujący wymianę może być przedstawiany w kilku wariantach w zależności od tego w jakich relacjach z innymi wymianami znajduje się prezentowana wymiana.

- Jeśli wymiana nie jest w żadnej relacji, wyświetlany jest podstawowy widok taki jak w przypadku wymiany w ramach przedmiotu '*Metody sztucznej inteligencji*' na rysunku 4.19. W widoku podstawowym na samej górze znajduje się nazwa przedmiotu oraz typ zajęć, których dotyczy stworzona wymiana. Na białym tle znajdują się numery



Rysunek 4.18: Widok wymian użytkownika na małym ekranie

grup, w której obecnie jest użytkownik oraz w której chce być. Dodatkowo po prawej stronie znajdują się przyciski otwierające okno wyboru wymian, które mają być w relacji z tą wymianą. Rodzaj relacji definiuje rodzaj naciśniętego przycisku.

- Jeśli wymiana jest w relacjach tylko jednego rodzaju, pod elementami z widoku podstawowego wyświetlane jest pole tekstowe informujące o rodzaju relacji (np. '*W relacji powiązania z*'), a pod nim wymiany, które są w relacji z daną wymianą. Wymiany w relacji zawierają informacje o nazwie przedmiotu, numerze obecnej grupy i numerze pożąданiej grupy oraz przycisk pozwalający na usunięcie relacji.
- Jeśli wymiana jest w relacjach różnych rodzajów, pod elementami z widoku podstawowego wyświetlane są dwie listy wymian w relacji z daną wymianą - lista wymian w relacji powiązania oraz lista wymian w relacji wykluczenia, tak jak w przypadku wymiany w ramach przedmiotu '*Seminarium dyplomowe*' na rysunku 4.19.

#### 4.5.7. Wybór wymiany do nadania relacji

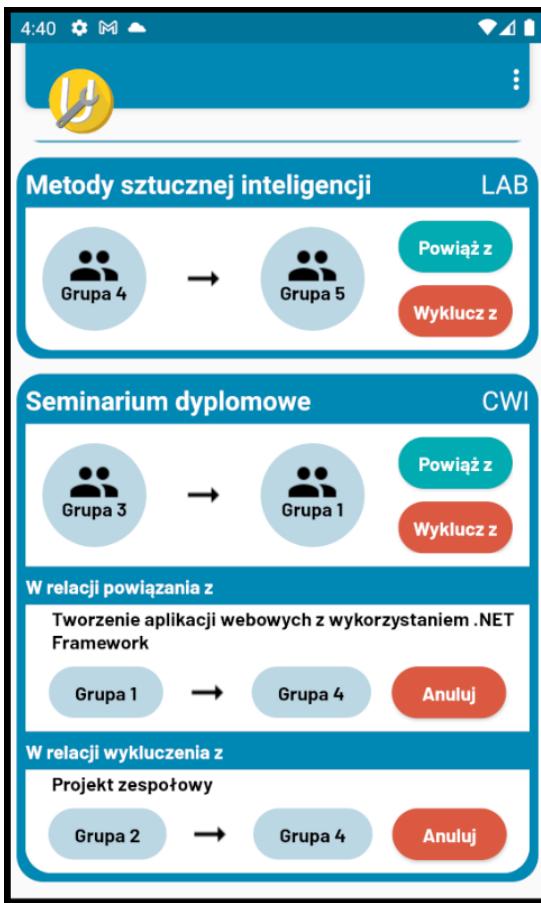
##### Aplikacja sieciowa

Po naciśnięciu dowolnego przycisku nadania relacji pojawia się okno modalne (Rysunek 4.20), w którym można wybrać wymianę do nadania relacji. W widoku dla mniejszych ekranów okno modalne zostało analogicznie dostosowane (Rysunek 4.21)

##### Aplikacja mobilna

Okno wyboru wymian prezentuje wymiany (Rysunek 4.22), które nie są wymianą,

#### 4.5. INTERFEJSY APLIKACJI MOBILNEJ I SIECIOWEJ



Rysunek 4.19: Wyświetlanie wymian w aplikacji mobilnej

z którą nadajemy relacje oraz które nie są jeszcze w relacji z tą wymianą. Wygląd każdego elementu, prezentującego wymianę, jest analogiczny do widoku podstawowego wymiany z tą różnicą, że nie ma w nim przycisków do otwierania okna wyboru wymian. Poniżej listy wymian znajdują się dwa przyciski - jeden służący do nadania relacji z wybranymi wymianami i jeden służący do anulowania wybierania wymian do relacji.

Po naciśnięciu wymiany, co jest równoznaczne z jej wyborem, wybrany element zmienia kolor oraz jest wyróżniony poprzez kolorową obwódkę, tak jak wymiana w ramach przedmiotu '*Seminarium dyplomowe*'. Ponowne jej naciśnięcie, które jest równoznaczne z anulowaniem wyboru, zmienia kolor elementu na kolor początkowy, tak jak wymiana w ramach przedmiotu '*Tworzenie aplikacji webowych z wykorzystaniem .NET Framework*'.

## 4. INTERFEJS UŻYTKOWNIKA

Dodaj relację wykluczenia do Teoria algorytmów i obliczeń - Grupa 7 → Teoria algorytmów i obliczeń - Grupa 6

Z

Metody sztucznej inteligencji - Grupa 5 → Metody sztucznej inteligencji - Grupa 4

Projekt zespołowy - Grupa 4 → Projekt zespołowy - Grupa 2

Zamknij

Rysunek 4.20: Dodawanie relacji między wymianami na dużym ekranie

### 4.5.8. Panel użytkownika

Interfejs pozwalający użytkownikowi na sprawdzenie swoich danych oraz na zmianę ustawień dotyczących całej aplikacji. Podstawowy użytkownik zobaczy w widoku swoje imię, nazwisko, numer albumu, nazwę użytkownika od identyfikator. Tutaj użytkownik może również zmienić pseudonim, język oraz ujawnić swoje dane osobowe.

#### Panel podstawowy

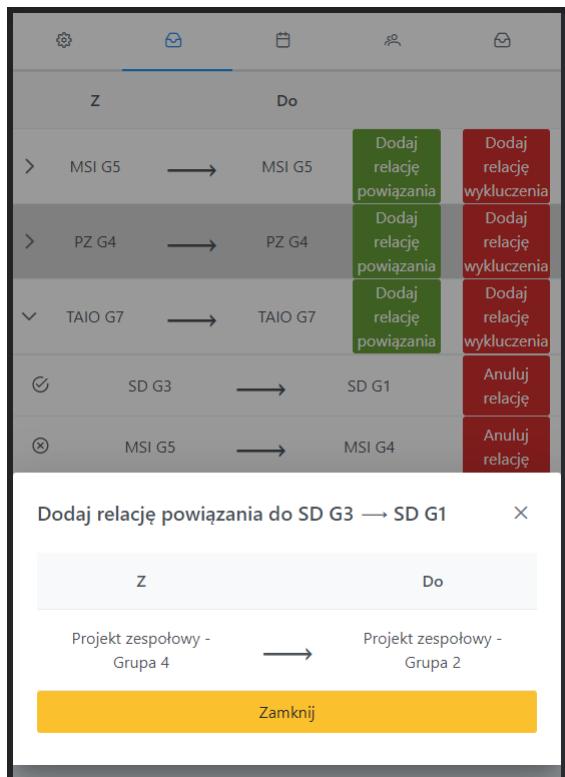
##### Aplikacja sieciowa

W aplikacji sieciowej (Rysunek 4.23) zmiana języka zrealizowana jest za pomocą przełącznika, natomiast ujawnienie danych osobowych za pomocą przycisku. Po udanej zmianie pseudonimu, z wykorzystaniem pola tekstowego oraz przycisku, zostanie wyświetlone powiadomienie o powodzeniu operacji (Rysunek 4.24).

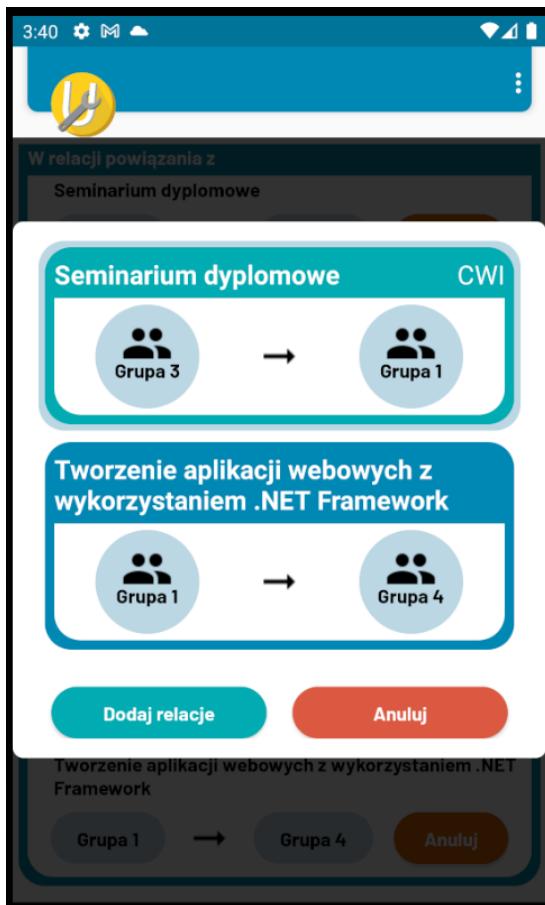
##### Aplikacja mobilna

W aplikacji mobilnej informacje o użytkowniku oddzielone są od ustawień (Rysunek 4.25). Te dwa segmenty wyświetlane są w dwóch kafelkach. Ustawienia RODO reguleowane są za pomocą przełącznika. Z kolei język można zmienić za pomocą przycisków radiowych (eng. *radio button*).

#### 4.5. INTERFEJSY APLIKACJI MOBILNEJ I SIECIOWEJ

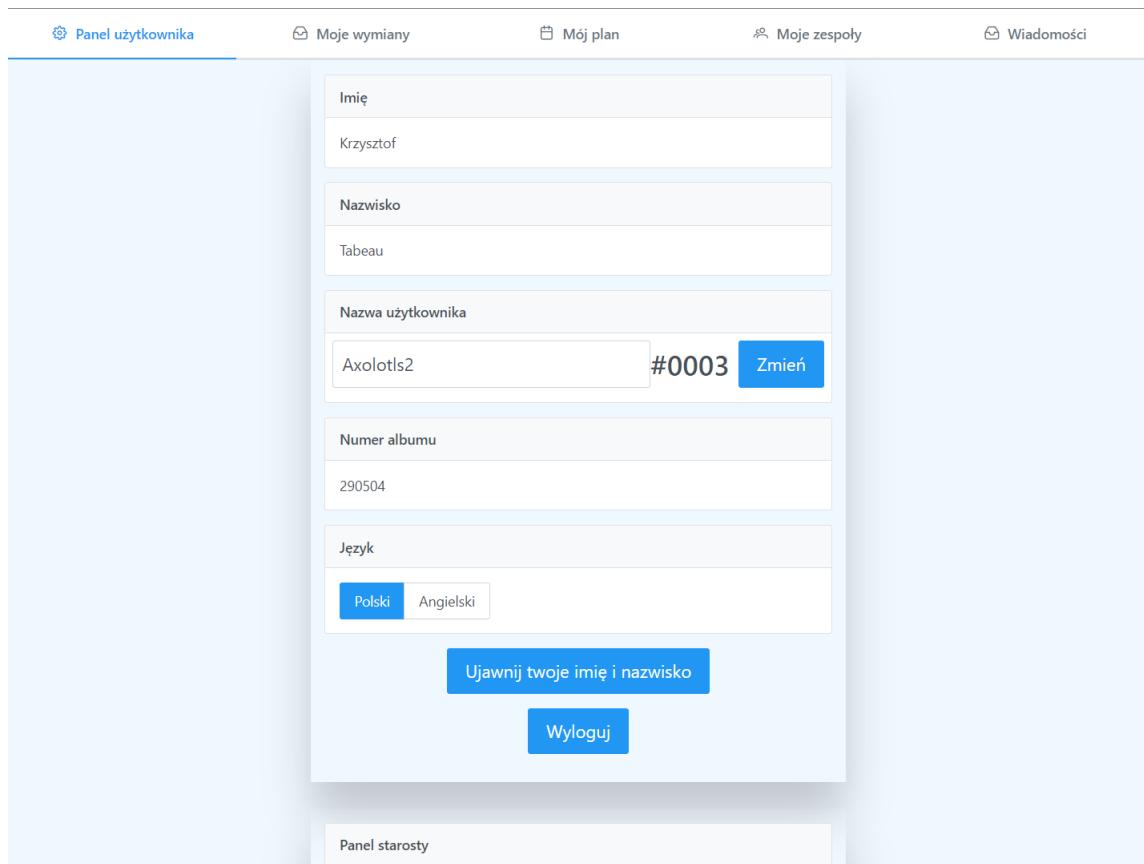


Rysunek 4.21: Dodawanie relacji miedzy wymianami na małym ekranie

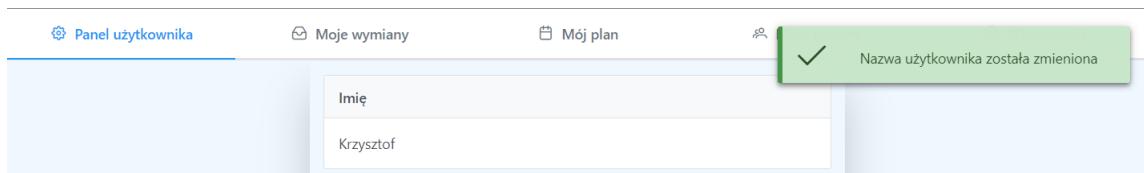


Rysunek 4.22: Wybieranie wymian do nadania relacji w aplikacji mobilnej

#### 4. INTERFEJS UŻYTKOWNIKA

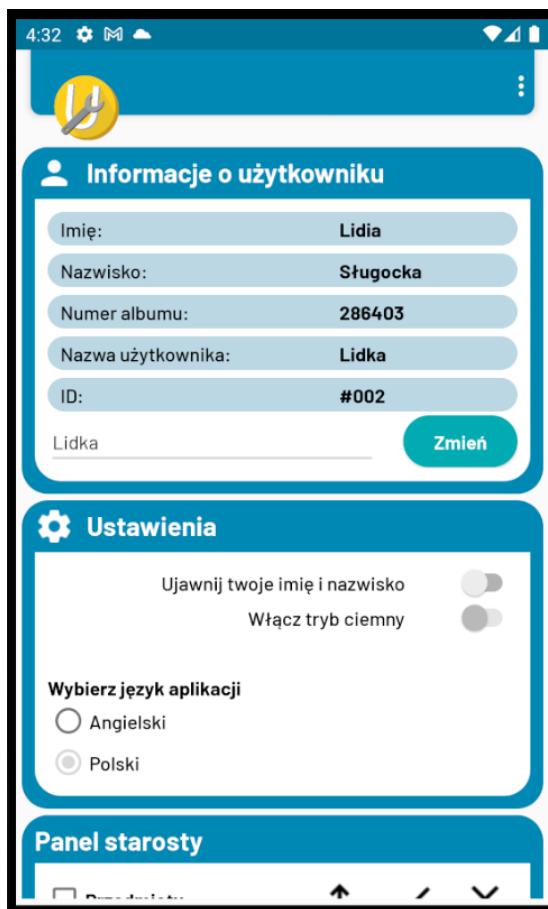


Rysunek 4.23: Widok panelu użytkownika (1/3)



Rysunek 4.24: Powiadomienie o powodzeniu

#### 4.5. INTERFEJSY APLIKACJI MOBILNEJ I SIECIOWEJ



Rysunek 4.25: Panel użytkownika aplikacji mobilnej

#### Panel starosty

Dla użytkowników o uprawnieniach starosty wyświetlony zostanie również panel starosty (rysunki 4.26 oraz 4.27). W tym panelu znajduje się tabela ze statystykami, zawierająca informacje o liczbie zgłoszonych wymian, liczbie zaakceptowanych wymian oraz liczbie odrzuconych wymian w podziale na przedmioty.

Użytkownik może wybrać poszczególne przedmioty, klikając w odpowiednie checkboxy. Wybranie checkboxa w nagłówku tabeli powoduje jednoczesne zaznaczenie lub odznaczenie wszystkich checkboxów w tabeli. Naciśnięcie przycisku **Policz wymiany** spowoduje wyświetlenie okna (rys. 4.28), w którym należy potwierdzić lub anulować policzenie wymian.

Po potwierdzeniu zostanie zgłoszona prośba o policzenie wymian dla danych przedmiotów. Potwierdzenie w postaci okna modalnego jest wymuszane przy każdej akcji starosty lub admina w tym interfejsie.

Przycisk **Wyślij dane do dziekanatu** pozwala na wysłanie do Dziekanatu maila z nowymi składami grup zajęciowych.

	Przedmioty	↑	↙	×
<input type="checkbox"/>	Projekt zespołowy	3	0	0
<input type="checkbox"/>	Metody sztucznej inteligencji	1	0	1
<input type="checkbox"/>	Seminarium dyplomowe	2	0	0
<input type="checkbox"/>	Teoria algorytmów i obliczeń	1	0	0

**Policz wymiany**

**Wyślij dane do dziekanatu**

Rysunek 4.26: Widok panelu użytkownika (2/3)

W aplikacji sieciowej na mniejszych ekranach nazwy przedmiotów zostały skrócone do pierwszych liter słów tworzących daną nazwę (Rysunek 4.29).

### Panel administratora

Dla użytkowników z uprawnieniami administratora poza panelem starosty wyświetlony zostanie dodatkowo panel administratora (Rysunki 4.30 oraz 4.31). W tym widoku można zobaczyć listę starostów oraz listę administratorów. Przy elementach tych list znajdują się przyciski do odbierania uprawnień danej osobie. Pod każdą z tych list znajduje się pole tekstowe, akceptujące tylko liczby, z przyciskiem **Dodaj**, które pozwala nadać odpowiednie uprawnienia osobie, której numer albumu został wpisany w pole tekstowe. Pole tekstowe zawiera mechanizm walidacji, który przy wpisaniu nie poprawnego indeksu pokazuje odpowiednią wiadomość (Rysunek 4.32).

Przycisk **Rozpocznij nowy semestr** służy do rozpoczęcia nowego semestru.

Każda operacja nadająca lub odbierająca uprawnienia musi zostać potwierdzona w wyskakującym oknie modalnym (Rys. 4.33).

## 4.5. INTERFEJSY APLIKACJI MOBILNEJ I SIECIOWEJ



Rysunek 4.27: Panel starosty aplikacji mobilnej

### 4.5.9. Konwersacje/Wiadomości

Widok skrzynki z wiadomościami jest możliwie najbardziej zbliżony do widoków w aplikacji Messenger (Rysunek 4.34). Interfejs składa się z dwóch części, wyboru czatu po lewej stronie oraz samego chatu po prawej stronie. W wyborze czatu na szczycie znajduje się mechanizm do zapraszania użytkowników analogiczny do mechanizmu zapraszania do zespołu w widoku zespołów (Rysunek 4.35).

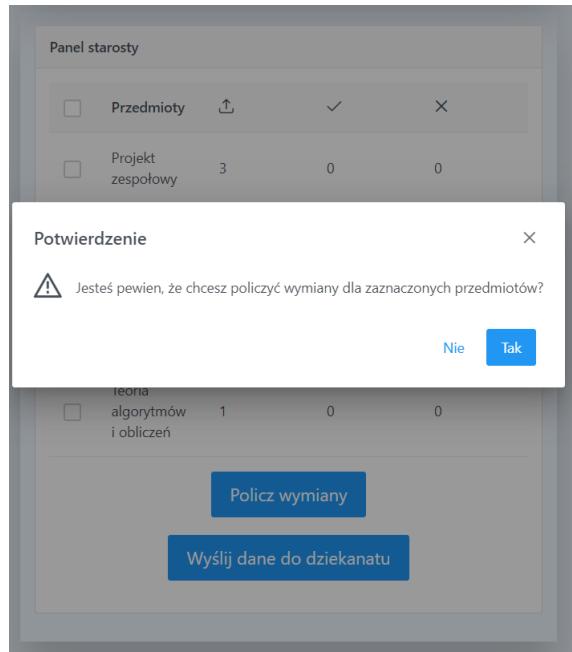
W widoku na mniejszym ekranie mechanizm został przeniesiony do okna modalnego (Rysunek 4.36). W liście czatów każdy czat zawiera informacje o: pseudonimie użytkownika, dacie ostatniej wiadomości oraz treści i adresacie ostatniej wiadomości. W samym czacie znajduje się lista wysyłanych wiadomości z datami oraz rozróżnieniem kolorystycznym adresatów wiadomości. Porządek wiadomości jest taki, aby najnowsze wiadomości były na dole okna. Na samym dole znajduje się pole tekstowe z przyciskiem do wysyłania wiadomości.

W aplikacji mobilnej interfejs ten podzielony jest na dwa ekrany. Podział jest analogiczny do podziału w powszechnie znanej aplikacji Messenger. Ekran konwersacji zawiera pole i przycisk do wyszukiwania użytkowników oraz listę wszystkich konwersacji (Rys. 4.37a) analogiczną do tej części widoku w aplikacji sieciowej.

Ekran wiadomości (Rys. 4.37b) również jest analogiczny do tej części widoku aplikacji sieciowej oraz do analogicznego widoku w aplikacji Messenger.

### 4.5.10. Ekran logowania

Prosty widok zawierający tylko przycisk **Zaloguj** w wersji mobilnej aplikacji mobilnej (Rys. 4.38) i wersji aplikacji sieciowej (Rys. 4.39).



Rysunek 4.28: Okno modalne z potwierdzeniem akcji

#### 4.5.11. Ekran do autoryzacji tokenu - aplikacja mobilna

Na środku ekranu znajduje się pole tekstowe do wpisania PINu (Rysunek 4.40a), służącego do autoryzacji tokenu użytkownika. Użytkownik otrzymuje PIN podczas logowania przez system USOS (Rysunek 4.40b). W dolnej części ekranu znajduje się przycisk do potwierdzenia PINu, a co za tym idzie - do autoryzacji tokenu użytkownika.

#### 4.5.12. Okna modalne - aplikacja mobilna

W aplikacji mobilnej występuje kilka rodzajów okien modalnych:

1. okno informujące o sukcesie operacji (Rysunek 4.41),
2. okno informujące o błędzie operacji (Rysunek 4.41),
3. okno do wyboru wymian, z którymi użytkownik chce nadać relacje (Rysunek 4.22),
4. okno do potwierdzania nadania lub odbioru uprawnień użytkownikom (Rysunek 4.33).

### 4.6. Interfejsy zewnętrzne

Aplikacja UsosFix jest zintegrowana z już istniejącą infrastrukturą systemu USOS. Pozwala to na wykorzystanie już istniejących kont studenckich oraz przypisanych do nich informacji w

## 4.7. PROCESY

		↑	✓	×
<input type="checkbox"/>	PZ	3	0	0
<input type="checkbox"/>	MSI	1	0	1
<input type="checkbox"/>	SD	2	0	0
<input type="checkbox"/>	TAIO	1	0	0

**Policz wymiany**

**Wyślij dane do dziekanatu**

Rysunek 4.29: Widok panelu starosty na małym ekranie

naszym systemie. Ułatwia to rozwiązywanie dwóch problemów. Po pierwsze, logowanie w naszej aplikacji może zostać przeprowadzone z pomocą systemu CAS USOS (Rys. 4.42). Po drugie, podczas logowania możemy poprosić użytkownika o udostępnienie nam danych na temat jego planu zajęć, co rozwiązuje problem wypełnienia naszej bazy danych informacjami na temat istniejących już grup przedmiotowych. Połączenie z systemem USOS, z którego korzystamy, to publicznie dostępne USOS API. Jego dokumentację można znaleźć na stronie [1].

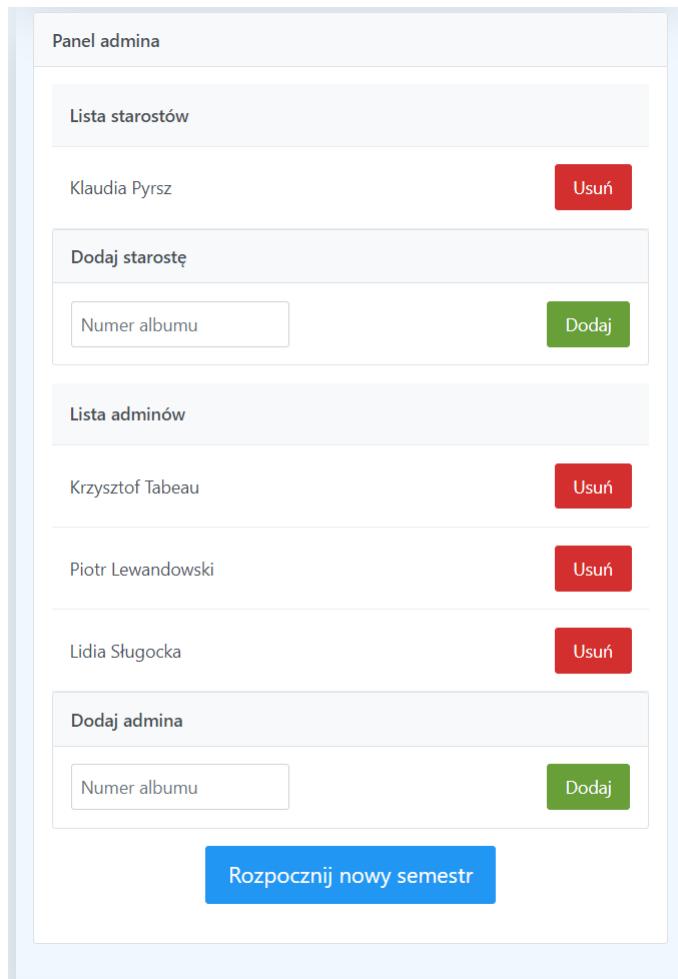
### 4.7. Procesy

#### Ogólny schemat procesów:

1. Wykonanie akcji w aplikacji mobilnej lub serwerowej (np. naciśnięcie odpowiedniego przycisku).
2. Wysłanie żądania do API.
3. Obsłużenie żądania przez API, aktualizacja bazy danych.
4. Odpowiedź API o powodzeniu lub niepowodzeniu żądania.
5. Zaktualizowanie odpowiedniego widoku/widoków.

**Logowanie** - pierwszy proces, z jakim spotyka się użytkownik podczas korzystania z aplikacji.

Po jego rozpoczęciu przez użytkownika kliknięciem przycisku **Zaloguj**, wykonywane są następujące operacje:



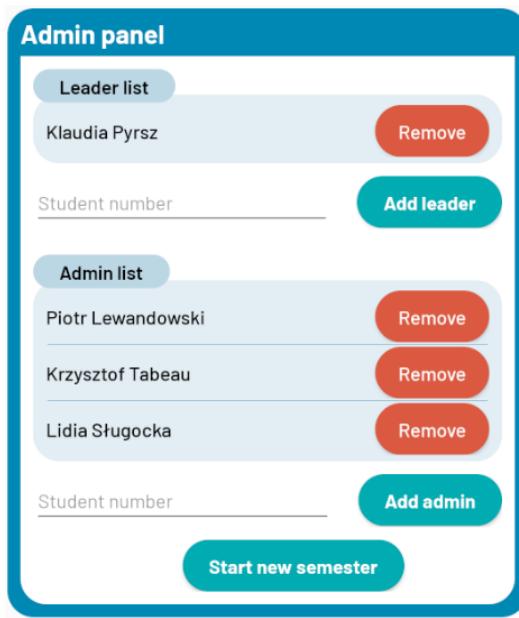
Rysunek 4.30: Widok panelu użytkownika (3/3)

1. Za pomocą dostarczonego przez API endpointa pobierany jest token.
2. Token jest używany do skonstruowania adresu URL przekierowującego do strony logowania USOSa.
3. Użytkownik loguje się do USOSa za pomocą swoich danych logowania.
4. Po zalogowaniu, USOS pyta użytkownika czy chce on pozwolić naszej aplikacji na dostęp do jego danych. (Tylko podczas pierwszego logowania w aplikacji)

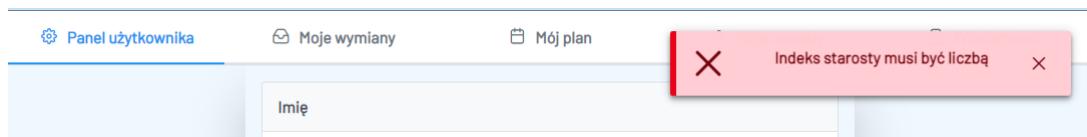
Dla aplikacji sieciowej:

5. Po wyrażeniu zgody na udostępnienie danych, aplikacja pobiera autoryzowany token za pomocą endpointu, dostarczonego przez API oraz przekierowuje do widoku planu zajęć. Ten autoryzowany token jest później wykorzystywany do pobierania danych na temat użytkownika z USOSa.

#### 4.7. PROCESY



Rysunek 4.31: Panel administratora aplikacji mobilnej



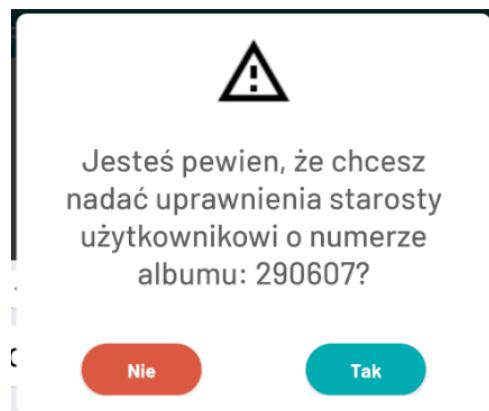
Rysunek 4.32: Błąd walidacji

Dla aplikacji mobilnej:

5. Po wyrażeniu zgody na udostępnienie danych, USOS wyświetla użytkownikowi PIN, który ma posłużyć do dalszej autoryzacji.
6. Po powrocie do aplikacji, użytkownik musi wpisać otrzymany PIN w wyznaczonym miejscu.
7. Po wpisaniu prawidłowego PINu, aplikacja pobiera autoryzowany token za pomocą endpointu, dostarczonego przez API. Ten autoryzowany token jest później wykorzystywany do pobierania danych na temat użytkownika z USOSa.

#### Zaproszenie użytkownika do zespołu

1. Jeśli użytkownik już należy do zespołu z danego przedmiotu - kliknięcie przy zapraszanym użytkowniku przycisku, służącego do zapraszania do zespołu lub wyszukanie zapraszanego użytkownika w widoku zespołów oraz kliknięcie przycisku.
2. Jeśli użytkownik jeszcze nie należy do zespołu z danego przedmiotu - kliknięcie przy zapraszanym użytkowniku przycisku, służącego do zapraszania do zespołu.
3. Wysłanie żądania do API.



Rysunek 4.33: Okno modalne do potwierdzenia zmiany uprawnień

4. Zaproszenie do zespołu.
5. Dodanie lub zaktualizowanie zespołu w widoku zespołów.

#### Zaakceptowanie/Odrzucenie zaproszenia do zespołu

1. Naciśnięcie przycisku akceptującego/odrzucającego w odpowiednim zespole w widoku zespołów.
2. Wysłanie żądania do API.
3. Zaakceptowanie/Odrzucenie zaproszenia.
4. Zaktualizowanie widoku zespołów w przypadku akceptacji lub usunięcie zespołu z widoku zespołów w przypadku odrzucenia.

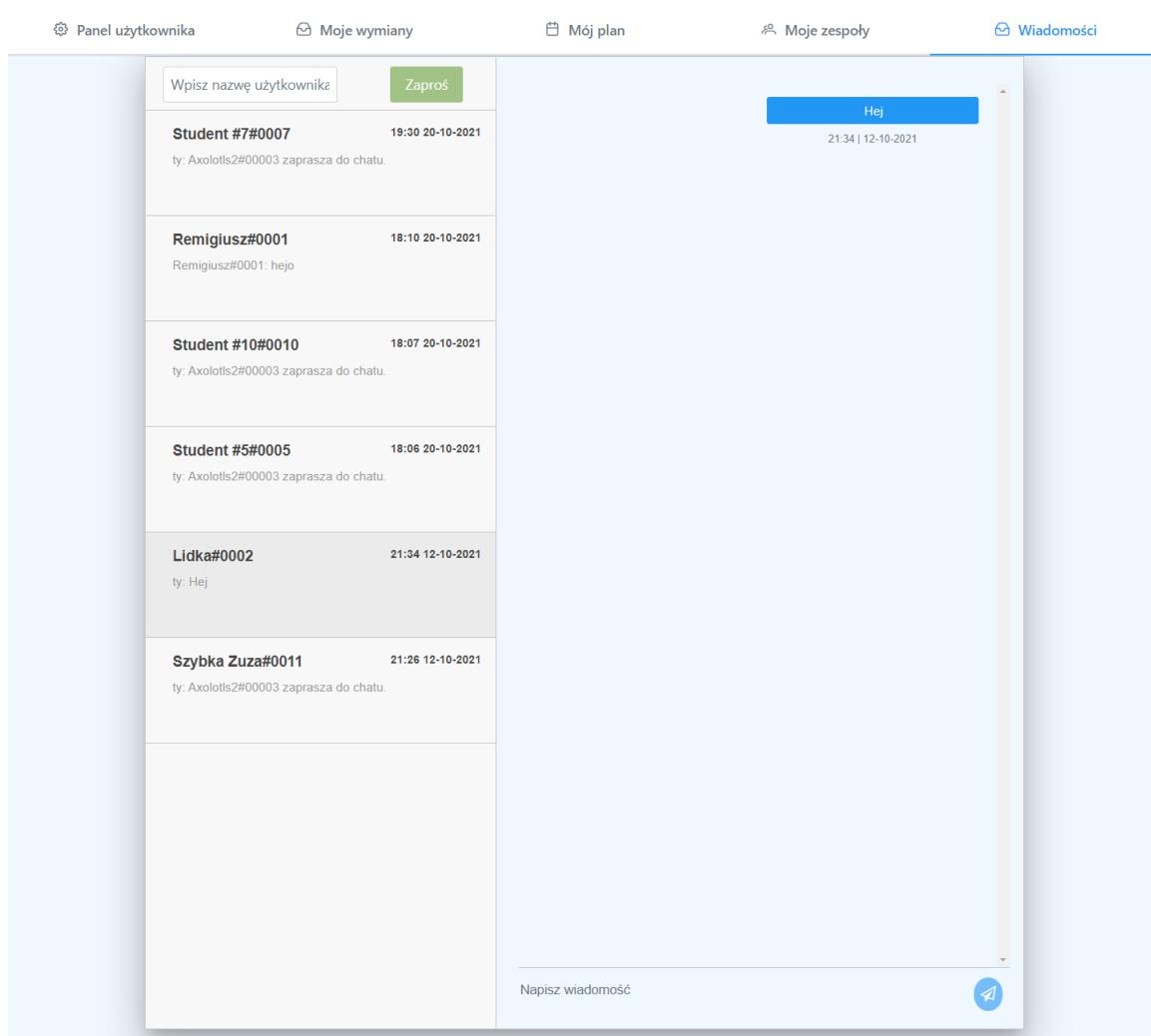
#### Opuszczenie zespołu

1. Naciśnięcie przycisku służącego do opuszczania zespołu w widoku odpowiedniego zespołu.
2. Wysłanie żądania do API.
3. Opuszczenie zespołu.
4. Usunięcie zespołu z widoku zespołów.

#### Usunięcie członka zespołu/Anulowanie zaproszenia do zespołu

1. Naciśnięcie przycisku służącego do usunięcia członka zespołu/anulowania zaproszenia do zespołu.
2. Wysłanie żądania do API.
3. Usunięcie z zespołu/Anulowanie zaproszenia do zespołu.

#### 4.7. PROCESY



Rysunek 4.34: Widok wiadomości na dużym ekranie

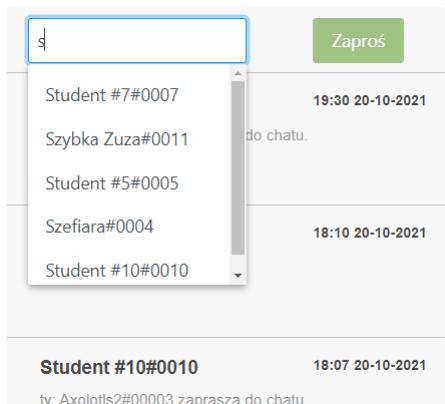
4. Zaktualizowanie widoku zespołu.

#### Zgłoszenie wymiany

1. Naciśnięcie przycisku, służącego do dołączania do grupy, w widoku grupy zajęciowej.
2. Wysłanie żądania do API.
3. Stworzenie prośby o wymianę.
4. Dodanie wymiany do widoku wymian oraz uwzględnienie wymiany w planie zajęć.

#### Nadanie relacji

1. Naciśnięcie przycisku do nadania relacji (powiązania lub wykluczenia) w widoku wymiany.
2. Otworzenie okna modalnego z innymi wymianami użytkownika, z którymi dana wymiana nie jest w relacji.



Rysunek 4.35: Wybór użytkowników do konwersacji

3. Wybór innej wymiany (innych wymian) w oknie modalnym.
4. Naciśnięcie przycisku służącego do nadania relacji w oknie modalnym.
5. Wysłanie żądania do API.
6. Nadanie relacji.
7. Zaktualizowanie widoku wymian.

### **Usunięcie wymiany**

1. Usunięcie wymianu odbywa się na tej samej zasadzie, co jej zgłoszenie.

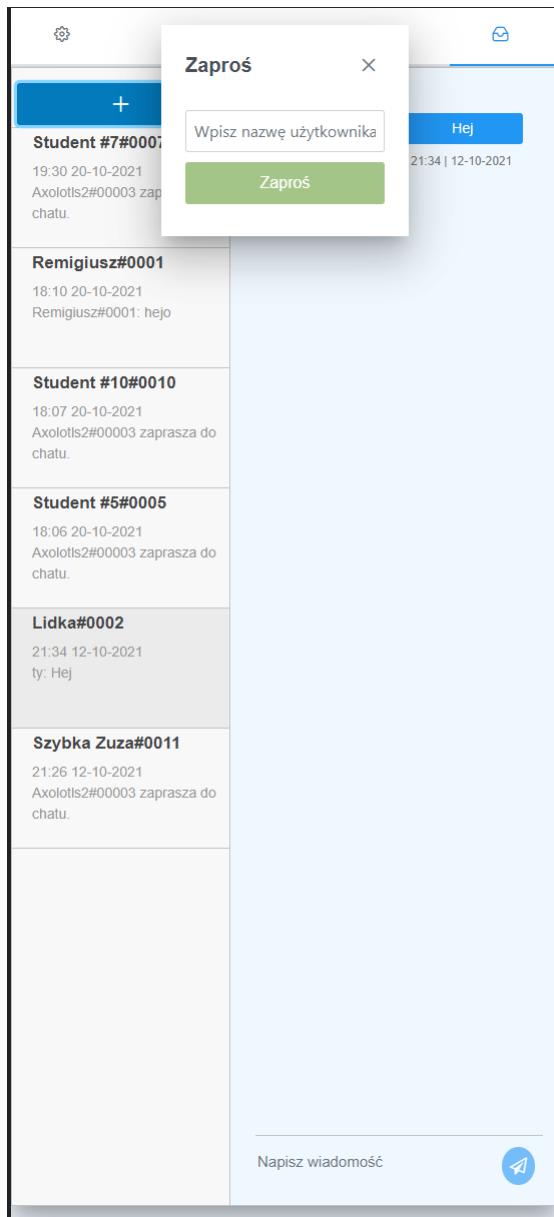
### **Anulowanie relacji**

1. Naciśnięcie przycisku służącego do anulowania relacji przy odpowiedniej relacji w widoku wymian.
2. Wysłanie żądania do API.
3. Usunięcie relacji.
4. Zaktualizowanie widoku wymian.

### **Zaproszenie do czatu**

1. Wyszukanie użytkownika.
2. Naciśnięcie przycisku zapraszającego wybranego użytkownika w widoku wiadomości/konwersacji.
3. Wysłanie żądania do API.
4. Zaproszenie użytkownika do czatu.
5. Zaktualizowanie widoku wiadomości/konwersacji.

#### 4.7. PROCESY

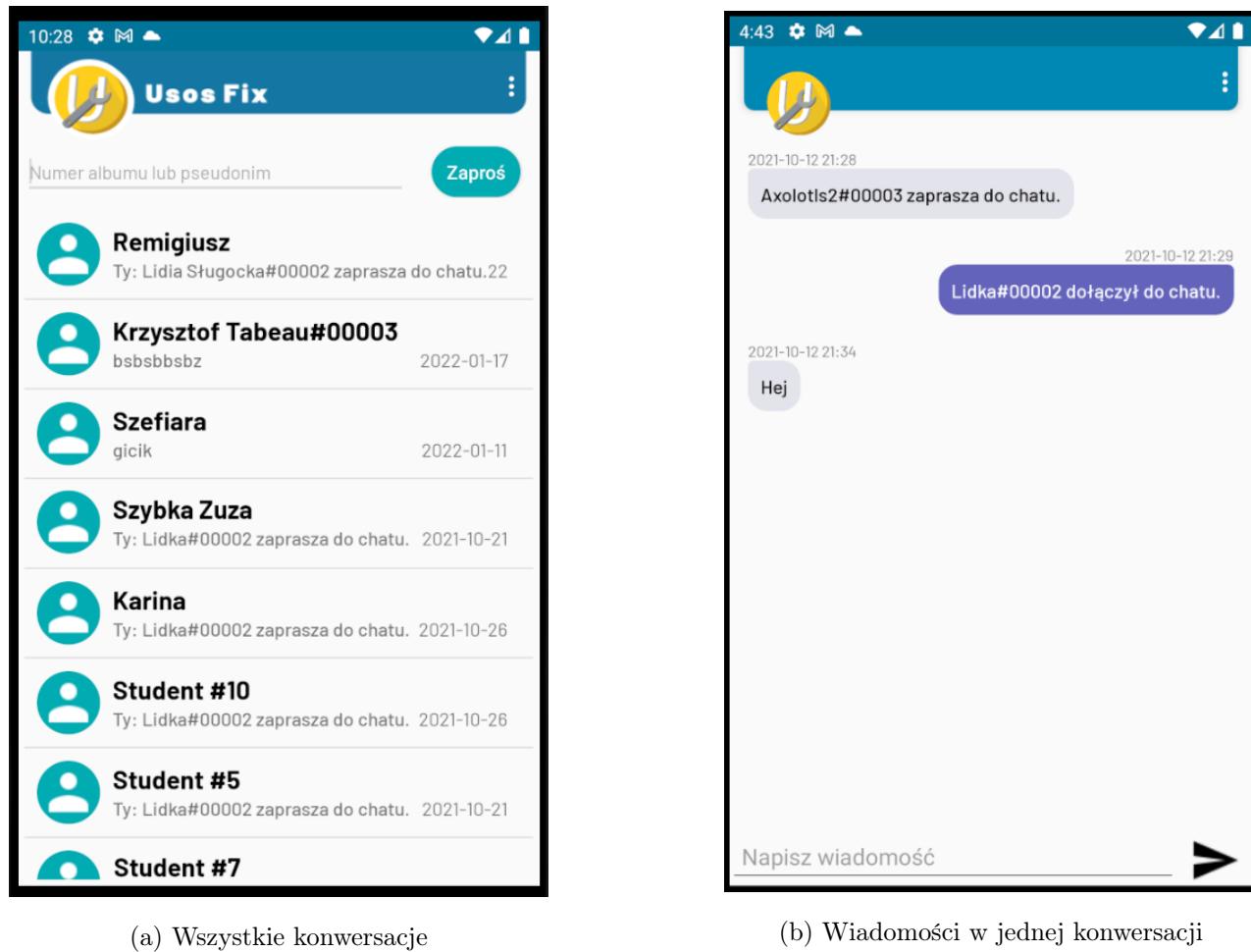


Rysunek 4.36: Widok wiadomości na małym ekranie

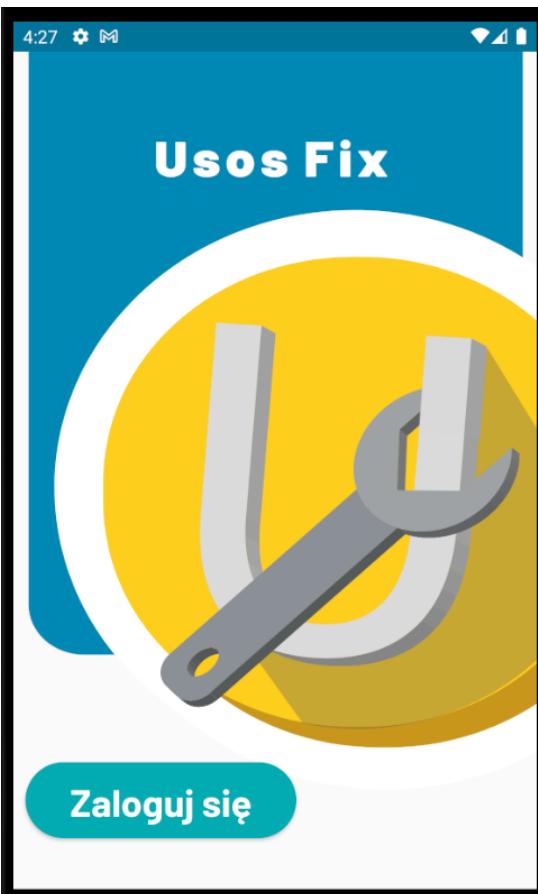
#### Zaakceptowanie/Odrzucenie zaproszenia do czatu

1. Wybór odpowiedniego czatu w widoku wiadomości/konwersacji.
2. Otworzenie widoku konwersacji (aplikacja mobilna).
3. Naciśnięcie przycisku akceptującego lub odrzucającego w widoku konwersacji.
4. Zaakceptowanie/Odrzucenie czatu.
5. Wysłanie wiadomości o zaakceptowaniu/odrzuceniu
6. Zaktualizowanie widoku wiadomości/konwersacji.

#### 4. INTERFEJS UŻYTKOWNIKA



Rysunek 4.37: Widoki modułu wiadomości aplikacji mobilnej



Rysunek 4.38: Ekran logowania aplikacji mobilnej

### Wysyłanie wiadomości

1. Nawiązanie stałego połączenia aplikacji z API za pomocą SignalR przy rozpoczęciu działania aplikacji.
2. Wpisanie treści wiadomości w odpowiednim polu tekstowym w widoku wiadomości.
3. Naciśnięcie przycisku do wysłania wiadomości lub naciśnięcie klawisza 'Enter' w przypadku aplikacji sieciowej.
4. Użycie SignalR do przesłania wiadomości do API.
5. Wpisanie wiadomości do bazy danych przez API.
6. Wysłanie odpowiedzi o powodzeniu/niepowodzeniu operacji do nadawcy.
7. Jeśli odbiorca wiadomości jest połączony za pomocą SignalR do API, wysłanie sygnału do aplikacji odbiorcy z treścią wiadomości.
8. Zaktualizowanie widoku wiadomości/konwersacji odbiorcy i nadawcy.



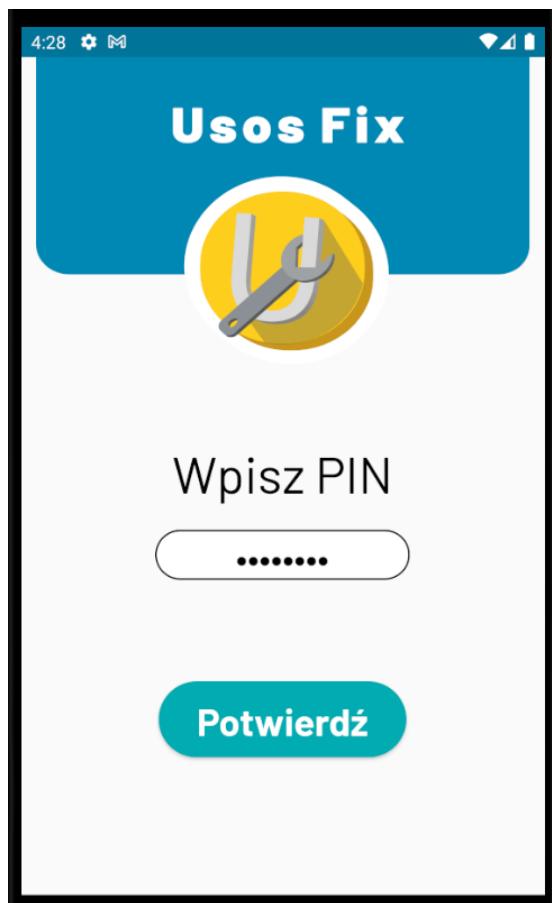
Rysunek 4.39: Ekran logowania aplikacji sieciowej

#### Nadawanie uprawnień – dotyczy tylko użytkowników z uprawnieniami admina

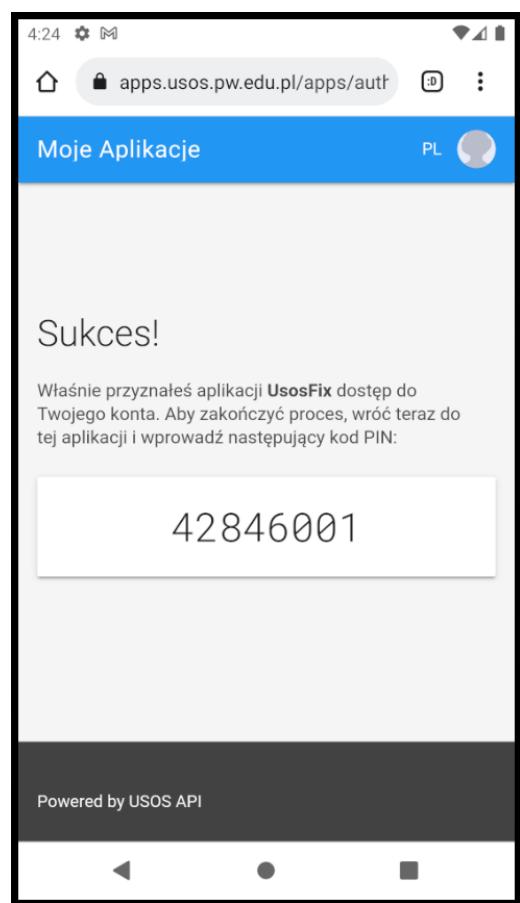
1. Wpisanie numeru albumu użytkownika w polu tekstowym do dodawania starostów/administratorów.
2. Naciśnięcie przycisku służącego do nadania uprawnień.
3. Sprawdzenie, czy wpisany tekst może być numerem albumu (6 cyfr).
4. Otworzenie okna modalnego do potwierdzenia chęci nadania uprawnień.
5. Potwierdzenie nadania uprawnień w oknie modalnym.
6. Wysłanie żądania do API.
7. Nadanie uprawnień wskazanemu użytkownikowi.
8. Aktualizacja panelu administratora w widoku panelu użytkownika z informacją o powodzeniu lub niepowodzeniu operacji.

#### Usuwanie uprawnień – dotyczy tylko użytkowników z uprawnieniami admina

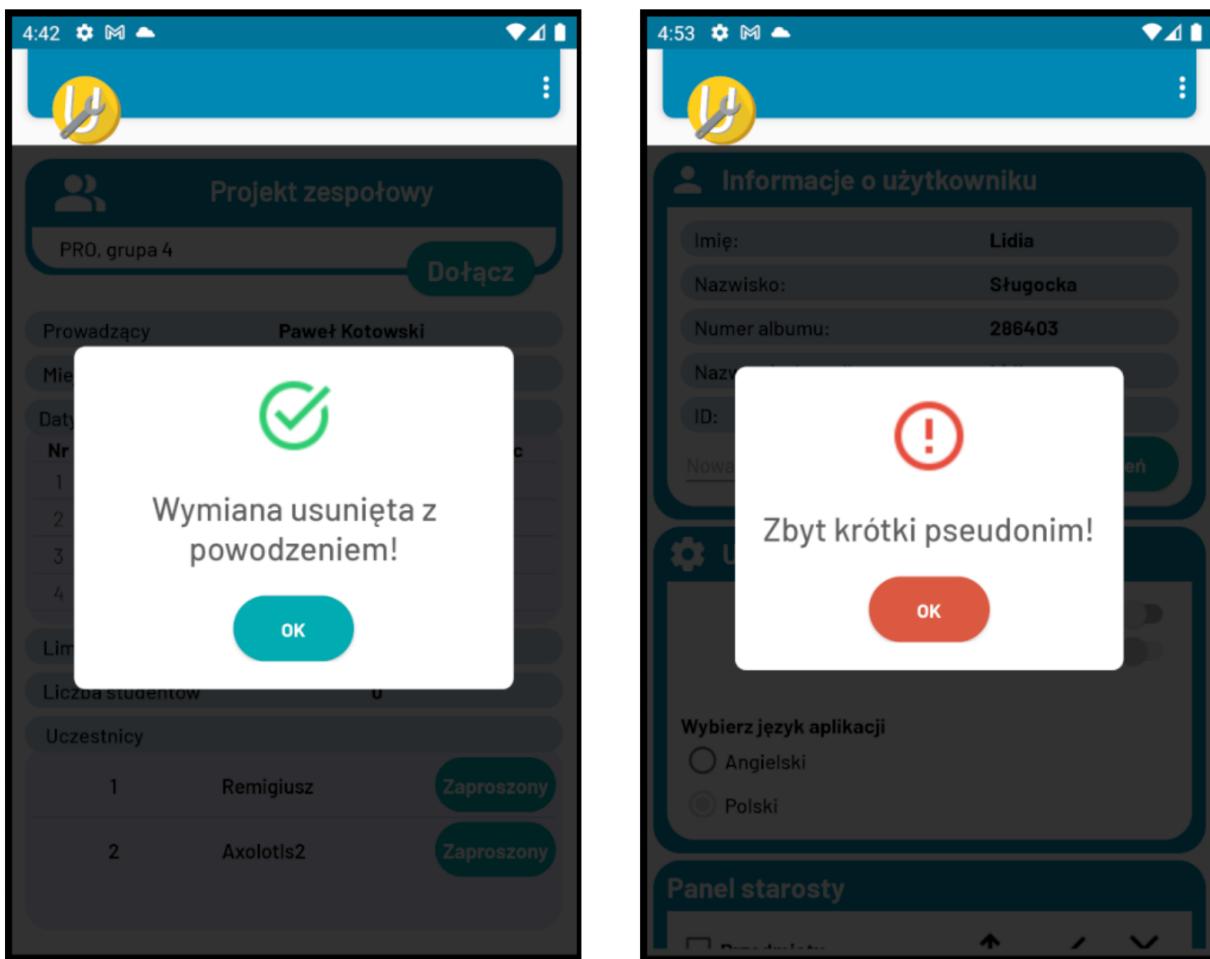
1. Naciśnięcie przycisku do odbierania uprawnień przy odpowiednim użytkowniku na liście starostów lub na liście administratorów.
2. Otworzenie okna modalnego do potwierdzenia chęci odebrania uprawnień.



(a) Ekran do potwierdzania PINu i autoryzacji tokenu



(b) Ekran do potwierdzania PINu i autoryzacji tokenu



Rysunek 4.41: Okna modalne informujące o rezultacie operacji

3. Potwierdzenie odebrania uprawnień w oknie modalnym.
4. Wysłanie żądania do API.
5. Usunięcie uprawnień wskazanemu użytkownikowi.
6. Aktualizacja panelu administratora w widoku panelu użytkownika z informacją o powodzeniu lub niepowodzeniu operacji.

#### **Wysłanie danych do dziekanatu – dotyczy starostów i adminów**

1. Naciśnięcie przycisku do wysyłki danych w panelu starosty.
2. Otworzenie okna modalnego do potwierdzenia wysłania danych.
3. Potwierdzenie wysłania danych w oknie modalnym.
4. Wysłanie żądania do API.
5. Wygenerowanie pliku .csv, zawierającego proponowane składy grup, po uwzględnieniu zrealizowanych wymian.

#### 4.7. PROCESY



Rysunek 4.42: Logowanie do USOSa na urządzeniu mobilnym

6. Wygenerowanie wiadomości e-mail, zawierających plik .csv jako załącznik.
7. Wysłanie wiadomości z wykorzystaniem serwera SendGrid.

#### Rozpoczęcie nowego semestru/Zaktualizowanie danych – dotyczy adminów

1. Naciśnięcie przycisku do rozpoczęcia nowego semestru w panelu starosty.
2. Otworzenie okna modalnego do potwierdzenia zaktualizowania danych.
3. Potwierdzenie rozpoczęcia nowego semestru.
4. Wysłanie żądania do API.
5. Zmiana aktywnego semestru na kolejny.
6. Oznaczenie istniejących danych na temat przedmiotów i grup jako historycznych.
7. Pobranie danych dotyczących nowego semestru.

### Zmiana ustawień – nazwa użytkownika, język aplikacji, RODO

1. Wpisanie nowej nazwy użytkownika w pole tekstowe oraz naciśnięcie przycisku do zmiany./Naciśnięcie przycisku służącego do zmiany języka/zrezygnowania z RODO.
2. Wysłanie żądania do API.
3. Wpisanie do bazy danych informacji o preferencji użytkownika.
4. Aktualizacja panelu użytkownika z informacją o powodzeniu lub niepowodzeniu operacji.

### Wymiana użytkowników

1. Zgłoszenie wymian przez użytkowników.
2. Kliknięcie przycisku do liczenia wymian przez użytkownika z uprawnieniami starosty lub administratora.
3. Wysłanie żądania do API.
4. Wykonanie operacji na serwerze:
  - (a) Odczytanie zgłoszonych wymian z bazy danych.
  - (b) Wygenerowanie sieci wraz z początkowym przepływem o strukturze, która pozwala na respektowanie relacji pomiędzy wymianami przy szukaniu maksymalnego przepływu.
  - (c) Znalezienie maksymalnego przepływu. Szukanie maksymalnego przepływu odpowiada szukaniu sposobu na jednoczesne przeprowadzenie jak największej liczby zamian.
  - (d) Jeśli istnieje kilka maksymalnych przepływów, wybrany zostaje ten, w którym wymiany zostały zgłoszone wcześniej.
  - (e) Stworzenie nowych składów grup zajęciowych.
5. Zaktualizowanie widoków.
6. Pierwszym etapem tego procesu jest wysyłanie przez użytkowników prośb o zmianę grupy, które zostają zapisane w bazie danych do późniejszego przetworzenia.
7. Użytkownik z uprawnieniami starosty lub admina zaznacza, wymiany dla których przedmiotów mają zostać wyliczone i wysyła takie żądanie do serwera.
8. Możliwe zamiany wprowadzone przez użytkowników zostają odczytane z bazy danych. Część z nich zostanie zrealizowana i wygenerowane zostaną nowe składy grup zajęciowych. Na podstawie zgromadzonych propozycji wymian, zostaje wygenerowana sieć

#### 4.7. PROCESY

wraz z początkowym przepływem o strukturze, która pozwala na respektowanie relacji pomiędzy wymianami przy szukaniu maksymalnego przepływu. Wierzchołkami w tym grafie są grupy zajęciowe, a krawędzi pomiędzy nimi reprezentują możliwe zamiany. Po znalezieniu maksymalnego przepływu, w zależności od tego które krawędzie zostały w nim wykorzystane możemy odtworzyć nowe składy grup. W tym modelu, szukanie maksymalnego przepływu odpowiada szukaniu sposobu na jednoczesne przeprowadzenie jak największej liczby zamian. W przypadku kiedy więcej niż jeden przepływ jest maksymalny, zostaje wybrany ten, w którym wymiany zostały zgłoszone wcześniej.

9. Po wykonanych zamianach użytkownicy widzą czy do dostały się do zamierzonej grupy w widoku planu użytkownika poprzez odpowiednie oznaczenie grup docelowych.

Prośby o wymiany mogą posiadać między sobą następujące relacje:

1. Prośba A jest związana z prośbą B:

Jeśli prośba A zostanie spełniona, to musi zostać również spełniona prośba B. Jeśli jedna z prośb nie zostanie spełniona, to druga też nie może zostać spełniona.

2. Prośba A wyklucza się z prośbą B:

Jeśli prośba A zostanie spełniona, to prośba B nie może zostać spełniona.

Obie te relacje są symetryczne. Relacja powiązania prośb jest dodatkowo przechodnia.

## 5. Implementacja i testy

### 5.1. Algorytm wymian

Algorytm wymian redukuje problem maksymalizacji liczby przeprowadzonych wymian do problemu szukania przepływu o minimalnym koszcie. Wejściem do algorytmu jest lista grup oraz lista wymian. Naszym celem jest wyznaczenie na ich podstawie sieci przepływu, w której minimalizacja kosztu będzie odpowiadała maksymalizacji liczby przeprowadzonych wymian. Konstrukcja sieci wygląda następująco:

1. Zacznij od grafu z pustym zbiorem wierzchołków i krawędzi.
2. Dodaj dwa wierzchołki, źródło i ujście.
3. Dla każdej grupy: dodaj jeden odpowiadający jej wierzchołek oraz po jednym wierzchołku dla każdego należącego do niej studenta. Następnie połącz wierzchołek grupy i ujście krawędzią o koszcie 0 i przepustowości równej liczbie wolnych miejsc w tej grupie.
4. Dla każdego użytkownika dodaj trzy krawędzie. Pierwsza ma przepustowość 1, koszt 0 i łączy źródło z wierzchołkiem użytkownika. Druga ma przepustowość 1, koszt 2 i łączy użytkownika z jego obecną grupą. Trzecia ma przepustowość 1, koszt 0 i łączy grupę z ujściem.
5. Dla każdej wymiany zgłoszonej przez tego użytkownika, dodaj krawędź od wierzchołka odpowiadającego użytkownikowi do wierzchołka grupy, do której użytkownik chciałby należeć. Krawędź ta ma przepustowość 1 i koszt 1. Ponadto dodaj krawędź od grupy docelowej do ujścia o przepustowości 1 i koszcie 0. Pomijane są wymiany znajdujące się w relacji powiązania z wymianą, która została wcześniej odrzucona, lub które są w relacji wykluczenia z wymianą, która została już zaakceptowana.

Stworzona sieć spełnia poniższe właściwości:

1. Maksymalna wartość przepływu jest równa liczbie studentów - krawędzi wychodzących ze źródła jest dokładnie tyle co studentów.

## 5.1. ALGORYTM WYMIAN

2. Istnieje przepływ osiągający maksymalną wartość - możemy go skonstruować korzystając jedynie z krawędzi odpowiadających obecny grupom studentów, co oznacza również, że w otrzymanym rozwiązaniu, każdy student zostanie przypisany do pewnej grupy.
3. Zamiana krawędzi odpowiadającej obecnej grupie na krawędź odpowiadającą zamianie zmniejsza koszt przepływu, ze względu na przypisane im wagi.

Jesteśmy więc w stanie, licząc maksymalny przepływ o minimalnym koszcie, wyznaczyć największy zbiór możliwych wymian. Wyliczony w ten sposób przepływ jest następnie tłumaczony na nowe składы grup w następujący sposób. Jeśli przez krawędź łączącą wierzchołek użytkownika z wierzchołkiem grupy został puszczyony przepływ, to zostaje on przypisany do tej grupy. Wszystkie niezrealizowane wymiany zostają odrzucone.

### 5.1.1. Relacje

Stworzone przez użytkowników relacje są uwzględniane przy realizacji wymian z pomocą następującej heurystyki. Opisaną w punkcie 5.1 metodę wyznaczania możliwych do zrealizowania wymian przeprowadzamy dla każdego przedmiotu po kolei. Po wyliczeniu wyniku w ramach danego przedmiotu dla każdej relacji  $r = (e, f)$ , która zawiera wymianę w ramach obecnie rozważanego przedmiotu (bez straty ogólności założymy, że jest to  $e$ ), wykonujemy:

1. Jeśli wymiany  $e, f$  są ze sobą w relacji powiązania i  $e$  została zaakceptowana, to dokonujemy wymiany  $e$ , tylko jeśli możemy jednocześnie dokonać wymiany  $f$ .
2. Jeśli wymiany  $e, f$  są ze sobą w relacji powiązania i  $e$  została odrzucona, to jednocześnie odrzucamy obie wymiany.
3. Jeśli wymiany  $e, f$  są ze sobą w relacji wykluczania i  $e$  została zaakceptowana, to jednocześnie akceptujemy  $e$  i odrzucamy  $f$ .
4. Jeśli wymiany  $e, f$  są ze sobą w relacji wykluczania i  $e$  została odrzucona, to nie jesteśmy jeszcze w stanie nic stwierdzić na temat wymiany  $f$ , więc zostanie ona normalnie rozważona na późniejszym etapie.

Uwzględnienie relacji może sprawić, że nie wszystkie wymiany wyznaczone przez przeliczenie przepływu zostaną zrealizowane. Oznacza to, że użytą przez nas metodą może wyznaczyć nieoptimalny pod względem liczby zrealizowanych wymian wynik.

### 5.1.2. Implementacja algorytmu

Dla tego grafu przepływ o minimalnym koszcie jest znajdowany przy pomocy zaimplementowanego w bibliotece Google OR-Tools „cost-scaling push-relabel algorithm”. Oznaczmy liczbę wierzchołków w rozważanym grafie przez  $V$ , liczbę krawędzi przez  $E$ , oraz maksymalny koszt krawędzi przez  $C$ . Wtedy złożoność wykorzystanej metody to:

$$O(V^2 \cdot \log(V))$$

## 5.2. Testy

### 5.2.1. Podejście do implementacji i testów

Nasza aplikacja została przetestowana przy pomocy testów jednostkowych oraz manualnych. Testy manualne sprawdzały przede wszystkim działanie całości aplikacji oraz integrację pomiędzy frontendem i backendem, a testy jednostkowe testowały przede wszystkim kluczowe elementy aplikacji serwerowej, takie jak algorytm przeprowadzania wymian.

### 5.2.2. Aplikacja serwerowa

Największe pokrycie testami po stronie aplikacji serwerowej mają dwa kluczowe obszary, które są również najbardziej podatnymi na błędy fragmentami aplikacji. Są to moduły odpowiedzialne za integrację z API platformy USOS oraz realizację wymian.

### 5.2.3. Aplikacja sieciowa

Aplikacja sieciowa od początku była w bezpiecznej wersji roboczej utrzymywana na zewnętrznym serwerze w celach testów na prawdziwych użytkownikach. Wersja testowa była aktualizowana po ukończeniu prac nad daną częścią aplikacji.

Pod kątem wizualnym aplikacja była testowana na lokalnym serwerze w trakcie prac nad komponentami i widokami.

### 5.2.4. Aplikacja mobilna

Aplikacja mobilna została przetestowana na emulatorze udostępnionym przez Android Studio oraz na fizycznych urządzeniach. Testy na emulatorze były wykonywane przez cały proces tworzenia aplikacji. Pozwoliły one na szybkie sprawdzanie jak będzie wyglądać oraz działać aplikacja na etapie tworzenia poszczególnych widoków i funkcjonalności. Możliwość debugowania aplikacji

## 5.2. TESTY

uruchomionej w emulatorze pozwoliła również na szybsze i bardziej precyzyjne rozpoznanie i naprawienie błędów.

Testy na fizycznych urządzeniach pozwoliły zweryfikować poprawność wyświetlania i działania aplikacji. Zostały one przeprowadzone przez ok. 10 testerów znalezionych wśród studentów PW. Każdy tester do uruchomienia aplikacji używał swojego smartfona. Mimo osiągnięcia oczekiwanej działania aplikacji na emulatorze, jej faktyczne uruchomienie nie urządzeniu wykazało dodatkowe błędy. Pojawiły się funkcje, które poprawnie działały na emulatorze, ale na urządzeniu nie. Przykładowo, do pobrania obecnej daty została użyta funkcja, która na emulatorze zwracała datę w pożądanym formacie, jednak na urządzeniu już nie.

Wykryte zostały również funkcje, które nie działały poprawnie na emulatorze, ale działały poprawnie na urządzeniu. W tym wypadku za przykład można podać wyświetlanie aplikacji w trybie ciemnym. Tryb ciemny na urządzeniu wyświetla się zgodnie z oczekiwaniami. Z kolei na emulatorze wszystkie elementy są widoczne, ale ich kolorystyka pozostawia wiele do życzenia.

## 6. Użyte technologie

### 6.1. Technologie ogólne

Ze względu na różny cel działania, każda z aplikacji składającej się na przedstawiany projekt została napisana z innym podejściem używając innych technologii. Są natomiast technologie, które były używane przez każdego z autorów do wsparcia zarządzania całością projektu oraz organizacji pracy.

Podstawową technologią użytą do wersjonowania kodu jest **Git** [2].

Kolejną technologią, której użyliśmy była **Jira** [3]. Dzięki niej byliśmy w stanie na bieżąco monitorować rozwój wszystkich aplikacji oraz reorganizować priorytety w trakcie ich tworzenia.

Do przetrzymywania kodu użyliśmy technologii **Bitbucket** [4].

Dodatkowo do udostępniania aplikacji użytkownikom na zewnętrznych serwisach użyliśmy serwisu **Heroku** [5].

### 6.2. Aplikacja sieciowa

Do rozwoju aplikacji sieciowej przyczyniają się dwa typy technologii. Te, które pozwoliły lub ułatwiły stworzenie czystego kodu oraz te, które bezpośrednio można uznać za część kodu.

#### 6.2.1. Technologie bezpośrednie

Najważniejszym narzędziem użyтыm w aplikacji sieciowej jest **Javascript** [7] oraz **Vue.js 3.0.0** [6]. Pierwsze jest językiem użytym do napisania aplikacji, a drugie jest **frameworkiem**, czyli dodatkiem do języka organizującym całość aplikacji z własną składnią, schematami oraz sposobem działania.

Biblioteki do **Vue.js** użyte w kodzie oraz ich przeznaczenie:

- @microsoft/signalr [8] - narzędzie użyte do biernej komunikacji z aplikacją serwerową.

Dzięki utrzymywaniu stałego połączenia z API, aplikacja jest w stanie pokazywać nowe wiadomości w czasie rzeczywistym.

## 6.2. APLIKACJA SIECIOWA

- axios [9] - narzędzie użyte do czynnej komunikacji z serwerem poprzez podstawowy protokół HTTP
- primevue / primeng / primeicons / primeflex [10] - biblioteki z gotowymi komponentami możliwymi do użycia w kodzie. Na użyte komponenty składają się takie rzeczy jak: przyciski, tabele, ikony, kolumny tabel, powiadomienia, menu, lista wyboru, okno modalne, okno skalowane, pola autouzupełniane, panele, paski ładowania, odznaki, pola numeryczne.
- vuex [11] - biblioteka do implementacji kontrolera stanu. Organizuje i rozdziela logikę aplikacji od interfejsu użytkownika.
- vue-cal [12] - narzędzie oferujące elastyczny komponent kalendarza użyty w widokach planów użytkownika i przedmiotów.
- vue-router [13] - narzędzie do organizacji ruchu i przejść między widokami.
- vue-cookies [14] - narzędzie do organizacji ciasteczek przeglądarki. W ciasteczkach aplikacja przechowuje token sesji.
- npm [15] - narzędzie oferujące dostęp do wszystkich bibliotek **Javascript'owych** użytych w aplikacji.

### 6.2.2. Technologie wspierające

- eslint [16] - biblioteka do monitorowania stylu kodu **Javascript'owego**. Sam język z powodu braku komplikacji jest elastyczny co do kodowania i pozwala na różne style programowania, przez co przy nieuwadze kod może stać się bardzo szybko bardzo nieczytelny. To narzędzie temu zapobiega, robiąc wstępną analizę kodu i pokazując lub naprawiając style niepoprawne z jego ustaloną wcześniej konfiguracją.
- stylelint [17] - narzędzie analogiczne do **eslint'a**, tylko zamiast analizować kod **JavaScript'owy**, analizuje kod **CSS** [18].
- VS Code [19] - edytor tekstowy z wieloma dodatkami ułatwiający pracę nad kodem używany podczas prac deweloperskich.
- serve-static / express [20] - biblioteki użyte do statycznego udostępniania działającej aplikacji na zdalnym serwerze.
- jest [21] - biblioteka do testów interfejsów.
- vue-cli-service [22] - narzędzie do zarządzania projektem z poziomu terminala, ułatwiające pracę z technologią **Vue**. Narzędzie było używane podczas prac deweloperskich.

### 6.3. Aplikacja serwerowa

Aplikacja serwerowa została stworzona w technologii **ASP.NET 6.0** [23]. Jest to **framework** stworzony przez firmę **Microsoft**, który umożliwia tworzenie serwerów i aplikacji sieciowych w języku **C#** [24]. Kod serwera korzysta także z poniższych bibliotek:

- Entity Framework Core [25] - biblioteka oraz zestaw narzędzi odpowiadających za zarządzanie bazą danych.
- Google OR-Tools [26] - zestaw implementacji algorytmów rozwiązujących ogólne problemy optymalizacyjne i kombinatoryczne.
- SignalR [27] - z jego pomocą której została zaimplementowana dwustronna komunikacja pomiędzy serwerem oraz aplikacją sieciową i mobilną.
- Serilog [28] - został wybrany jako implementacja logowania w aplikacji serwerowej.
- SendGrid [29] - biblioteka umożliwiająca tworzenie i wysłanie wiadomości mailowych.
- NUnit [30] - biblioteka, w której zostały napisane testy jednostkowe.

### 6.4. Aplikacja mobilna

Aplikacja mobilna została stworzona przy pomocy **Android Studio** [31], natywnego środowiska dla aplikacji mobilnych dla systemu operacyjnego **Android**. Język programowania **Java** [32] został użyty do zaimplementowania logiki aplikacji - zarządzania działaniem poszczególnych akcji i widoków. Natomiast język **XML** [33] umożliwił stworzenie interfejsu użytkownika.

Większość potrzebnych funkcjonalności jest dostarczone przez **Android Studio** bez konieczności dołączania dodatkowych bibliotek, pakietów czy framework'ów.

Do stworzenia pozostałych funkcjonalności zostały użyte następujące technologie:

- com.google.android.material:material [34] - biblioteka pozwalającej na użycie elementów Google Material Design jako komponentów interfejsów,
- com.android.volley:volley [35] - biblioteka HTTP pozwalająca na łatwiejsze i szybsze wykonywanie żądań HTTP,
- com.microsoft.signalr:signalr [36] - pakiet pozwalający na użycie klienta SignalR dla języka Java, który został wykorzystany do zaimplementowania mechanizmu wysyłania i odbierania wiadomości w czasie rzeczywistym,

#### 6.4. APLIKACJA MOBILNA

- org.slf4j:slf4j-jdk14 - biblioteka SLF4J [37] - interfejs API rejestrowania wysokiego poziomu, który jest niezbędny do poprawnego działania klienta SignalR dla języka Java,
- com.google.code.gson:gson [38] - biblioteka pozwalająca na tworzenie obiektów JSON - w aplikacji wykorzystana do konwertowania obiektów na obiekty JSON,
- JUnit 4 [39] - framework służący do implementacji testów,
- androidx.test oraz androidx.text.espresso [40] - pakiety umożliwiające wykonanie testów aplikacji mobilnych na system Android napisanych w języku Java.

## 7. Plan tworzenia projektu

### 7.1. Podział prac

Podzieliliśmy się pracą jeszcze zanim rozpoczęliśmy tworzenie aplikacji. Piotr Lewandowski był odpowiedzialny za bazę danych, serwer i szeroko pojęty backend aplikacji. Pozostali członkowie zespołu zajęli się tzw. frontendem aplikacji. Lidia Ślągocka podjęła się stworzenia aplikacji mobilnej - zarówno interfejsów jak i logiki. Z kolei Krzysztof Tabeau zrealizował aplikację sieciową.

Na początku projektu kierowaniem projektu zajął się Krzysztof. W kompetencje jego roli wchodziło planowanie spotkań, wyznaczanie celów, podejmowanie decyzji projektowych, dopilnowywanie ich realizacji oraz zarządzanie organizacją projektu w Jirze. Po stworzeniu podstaw projektu z roli kierownika, została wydzielona rola właściciela projektu (eng. Product Owner), który podejmował decyzje jak projekt końcowy ma wyglądać. Rolę tą objęła Lidia.

### 7.2. Model wytwórczy

Ogólnym modelem wytwórczym był **Waterfall** z elementami **agile**. Cel naszej aplikacji mieliśmy określony od samego początku. Wiedzieliśmy jakie efekty chcemy osiągnąć i jakie przypadki użycia chcemy rozwiązać, tak więc podejście **Waterfall**, było jak najbardziej na miejscu.

Elementy metodyki **agile** użyliśmy do spraw czysto organizacyjnych. Podział pracy na **Sprinty** oraz cotygodniowe podsumowanie i planowanie kolejnych prac pomogło nam trzymać aktualne szacowanie terminu skończenia poszczególnych elementów aplikacji oraz informacje na temat tempa prac, które powinniśmy utrzymać, aby mieć gotowe etapy projektu w założonym wcześniej czasie.

## 7.3. HARMONOGRAM PRAC

### 7.3. Harmonogram prac

	Piotr Lewandowski	Lidia Ślugocka	Krzysztof Tabeau
Sprint 0 28.09 - 04.10.2020	<ul style="list-style-type: none"> <li>• Zapoznanie się z API USOSa</li> <li>• Zbadanie możliwości API USOSa</li> <li>• Zbadanie możliwości autoryzacji za pomocą USOSa</li> <li>• Przetestowanie odkrytych funkcjonalności</li> <li>• Przedstawienie możliwości reszcie zespołu</li> <li>• Wybór bazy danych</li> </ul>	<ul style="list-style-type: none"> <li>• Wybór technologii</li> <li>• Wybór języka programowania do stworzenia aplikacji mobilnej</li> <li>• Przetestowanie wybranego języka</li> <li>• Zainstalowanie i przetestowanie Android Studio</li> </ul>	<ul style="list-style-type: none"> <li>• Zapoznanie się z Vue.js</li> <li>• Zainstalowanie odpowiednich komponentów i programów</li> <li>• Nauczenie się schematu programowania w Vue.js</li> <li>• Znalezienie sposobu na routing w Single-Page-App'ie w Vue.js</li> <li>• Znalezienie metody na wykonywanie żądań HTTP w Vue.js</li> <li>• Nauczenie się Vuex (kontroler stanu)</li> </ul>
Sprint 1 05.10 - 11.10.2020	<ul style="list-style-type: none"> <li>• Napisanie kodu odpowiedzialnego za autoryzację użytkownika przez USOSa</li> <li>• Przetestowanie kodu z Krzysztofem</li> <li>• Wstępna konfiguracja bazy danych</li> </ul>	<ul style="list-style-type: none"> <li>• Napisanie strony logowania w wybranym języku</li> <li>• Podjęcie stylistycznych decyzji związanych z UX z Krzysztofem</li> </ul>	<ul style="list-style-type: none"> <li>• Napisanie strony logowania</li> <li>• Przetestowanie kodu logowania z Piotrem</li> <li>• Podjęcie stylistycznych decyzji związanych z UX z Lidią</li> </ul>

	Piotr Lewandowski	Lidia Sługocka	Krzysztof Tabeau
Sprint 2 12.10 - 18.10.2020	<ul style="list-style-type: none"> <li>• Zaprojektowanie bazy danych</li> <li>• Zautomatyzowanie pobierania danych z USOSa</li> <li>• Zautomatyzowanie wpisywania danych do bazy danych</li> </ul>	<ul style="list-style-type: none"> <li>• Stworzenie szkieletów widoków,</li> <li>• Napisanie komponentu odpowiedzialnego za nawigację (menu)</li> </ul>	
Sprint 3 19.10 - 25.10.2020	<ul style="list-style-type: none"> <li>• Prowadzenie sesji użytkownika</li> <li>• Obsługa żądań od zalogowanego użytkownika</li> </ul>	<ul style="list-style-type: none"> <li>• Implementacja widoku własnego planu użytkownika: <ul style="list-style-type: none"> <li>– Możliwość wyświetlania weekendów w planie</li> <li>– Wyświetlenie stanu z USOSa</li> <li>– Naniesienie aktywnych ogłoszeń na plan</li> </ul> </li> </ul>	
Sprint 4 26.10 - 01.11.2020	<ul style="list-style-type: none"> <li>• Obsługa żądań od zalogowanego użytkownika</li> <li>• Algorytm usuwania koliżji</li> </ul>	<ul style="list-style-type: none"> <li>• Implementacja widoków przedmiotów i grup: <ul style="list-style-type: none"> <li>– Grupy zajęciowe: <ul style="list-style-type: none"> <li>* Panele z częścią informacjami o grupie</li> <li>* Możliwość dołączenia do grupy np. za pomocą przycisku, jeśli użytkownik nie należy do danej grupy</li> </ul> </li> <li>– Moje grupy: <ul style="list-style-type: none"> <li>* Jedna grupa na przedmiot</li> <li>* Lista osób należących do grupy</li> <li>* Informacja kto jest w jakiej grupie zajęciowej</li> <li>* Możliwość wyświetlenia informacji o osobie</li> <li>* Możliwość zrezygnowania z RODO</li> </ul> </li> </ul> </li> </ul>	

### 7.3. HARMONOGRAM PRAC

	Piotr Lewandowski	Lidia Ślągocka	Krzysztof Tabeau
Sprint 5 02.11 - 08.11.2020	<ul style="list-style-type: none"> <li>• Obsługa żądań od zalogowanego użytkownika</li> <li>• Algorytm usuwania koliżji</li> <li>• Automatyczne generowanie wiadomości</li> </ul>	<ul style="list-style-type: none"> <li>• Implementacja skrzynki z wiadomościami (na wzór messengera): <ul style="list-style-type: none"> <li>– Skrzynki od zaakceptowanych użytkowników</li> <li>– Skrzynki od niezaakceptowanych użytkowników (użytkownik nadający wiadomość nie może napisać kolejnej?)</li> <li>– Możliwość akceptowania lub odrzucania i blokowania piszących użytkowników</li> <li>– Możliwość odpisania na wiadomość</li> <li>– Powiadomienia</li> <li>– Mały panel do informacji o osobie piszącej (nr albumu lub imię i nazwisko oraz wspólne zajęcia/kierunki/wydziały etc..)</li> </ul> </li> </ul>	
Sprint 6 09.11 - 15.11.2020	<ul style="list-style-type: none"> <li>• Obsługa żądań od zalogowanego użytkownika</li> <li>• System do automatycznych wyszukiwań zgodnych ogłoszeń i powiadamiania o tym użytkowników</li> </ul>	<ul style="list-style-type: none"> <li>• Implementacja panelu ustawień/kontrol: <ul style="list-style-type: none"> <li>– Lista zablokowanych użytkowników</li> <li>– Lista miejsc, w których zrezygnowano z RODO</li> </ul> </li> </ul>	
Sprint 7 16.11 - 22.11.2020	<ul style="list-style-type: none"> <li>• Obsługa żądań od zalogowanego użytkownika</li> <li>• System do automatycznych wyszukiwań zgodnych ogłoszeń i powiadamiania o tym użytkowników</li> </ul>	<ul style="list-style-type: none"> <li>• Implementacja ról: <ul style="list-style-type: none"> <li>– Admina (ustalanie starostów)</li> <li>– Starosty (eksport danych, możliwość zmiany limitu miejsc w grupie zajęciowej)</li> <li>– Podstawowego użytkownika</li> </ul> </li> </ul>	

## 7. PLAN TWORZENIA PROJEKTU

	Piotr Lewandowski	Lidia Sługocka	Krzysztof Tabeau
Sprint 8 23.11 - 29.11.2020	<ul style="list-style-type: none"> <li>• Dopracowanie modułów</li> <li>• Stworzenie Tutorialu (podświetlanie poszczególnych elementów z krótkim opisem co robią)</li> <li>• Przygotowanie do pisania testów jednostkowych</li> </ul>		
Sprint 9 30.11 - 6.12.2020	<ul style="list-style-type: none"> <li>• Testy jednostkowe</li> </ul>		
Sprint 10 7.12 - 13.12.2020	<ul style="list-style-type: none"> <li>• Dokonanie pisania testów jednostkowych</li> <li>• Przygotowanie do testów integracyjnych i połączenia</li> </ul>		
Sprint 11 14.12 - 20.12.2020	<ul style="list-style-type: none"> <li>• Połączenie aplikacji w całość</li> <li>• Testy integracyjne</li> </ul>		
Sprint 12 21.12.2021 - 3.01.2022	<ul style="list-style-type: none"> <li>• Poprawki</li> <li>• Doszlifowanie aplikacji</li> </ul>		

#### 7.4. Harmonogram końcowy

Ze względu na ograniczony czas na napisanie projektu w trakcie jednego semestru, niektóre moduły były wstępnie robione niedokładnie. Z oryginalnego planu udało nam się w całości zrealizować założenia do sprintu 7 włącznie. Zostały pominięte małe założenia, ze względu na większy cel w postaci ogólnego działania spójnej aplikacji.

Po zakończonym semestrze zajęliśmy się projektem nie układając harmonogramu. Definiowaliśmy nasze cele na podstawie ulepszania istniejących komponentów do poziomu wizualnie satysfakcjonującego. Cele ustalaliśmy przy rozpoczęciu sprintu oraz uznawaliśmy cel za zakończony, jeśli każdy członek zespołu nie miał uwag, podczas podsumowania sprintu.

Z oryginalnego planu nie zostały zrealizowane:

- testy integracyjne - podczas rozwoju aplikacji, utrzymywaliśmy ciągłą integrację komponentów, dzięki czemu mieliśmy pewność, że moduły dobrze współpracują ze sobą.
- tutorial obsługi aplikacji - postanowiliśmy włożyć więcej pracy w przejrzystość i intuicyjność aplikacji zamiast w instrukcję obsługi.
- system do automatycznych wyszukiwań zgodnych ogłoszeń i powiadamiania o tym użytkowników - zrezygnowaliśmy z ogłoszeń i skupiliśmy nasz system na podstawie zgłoszenia chęci bycia w danej grupie.
- lista zablokowanych użytkowników - podczas prac doszliśmy do wniosku, że ta funkcjonalność będzie przydatna użytkownikowi w minimalny sposób.
- lista miejsc, w których zrezygnowano z RODO - dla uspójnienia aplikacji podjęliśmy decyzję, aby użytkownik miał opcję wszędzie zrezygnować z RODO lub nigdzie.
- możliwość wyświetlenia informacji o osobie - funkcjonalność nie wnosiłaby nic przydatnego w kontekście obranych przez nas Use Case'ów do rozwiązania

## Bibliografia

- [1] Dokumentacja API USOSa  
<https://apps.usos.pw.edu.pl/developers/api/>
- [2] Dokumentacja systemu kontroli wersji GIT  
<https://git-scm.com/>
- [3] Dokumentacja programu Jira  
<https://www.atlassian.com/software/jira>
- [4] Dokumentacja systemu do przechowywania repozytorium Bitbucket  
<https://bitbucket.org/product/>
- [5] Dokumentacja systemu do udostępniania aplikacji Heroku  
<https://www.heroku.com/what>
- [6] Dokumentacja framework'u Vue 3.0  
<https://v3.vuejs.org/guide/introduction.html>
- [7] Dokumentacja języka programowania JavaScript  
<https://developer.mozilla.org/en-US/docs/Web/JavaScript>
- [8] Dokumentacja biblioteki SignalRJS  
<https://www.npmjs.com/package/signalrjs>
- [9] Dokumentacja biblioteki Axios  
<https://www.npmjs.com/package/axios>
- [10] Dokumentacja biblioteki PrimeVue  
<https://www.primefaces.org/primevue/showcase/>
- [11] Dokumentacja biblioteki Vuex  
<https://www.npmjs.com/package/vuex>
- [12] Dokumentacja biblioteki Vue-Cal  
<https://antoniandre.github.io/vue-cal/>

## BIBLIOGRAFIA

- [13] Dokumentacja biblioteki Vue-Router  
<https://router.vuejs.org/>
- [14] Dokumentacja biblioteki Vue-Cookies  
<https://www.npmjs.com/package/vue-cookies>
- [15] Dokumentacja systemu NPM  
<https://www.npmjs.com/>
- [16] Dokumentacja biblioteki Eslint  
<https://eslint.org/>
- [17] Dokumentacja biblioteki Stylelint  
<https://stylelint.io/>
- [18] Dokumentacja języka CSS  
<https://developer.mozilla.org/en-US/docs/Web/CSS>
- [19] Dokumentacja programu VSCode  
<https://code.visualstudio.com/>
- [20] Dokumentacja biblioteki Express  
<https://expressjs.com/>
- [21] Dokumentacja biblioteki Jest  
<https://jestjs.io/>
- [22] Dokumentacja biblioteki Vue-Cli-Service  
<https://cli.vuejs.org/guide/cli-service.html>
- [23] Dokumentacja framework'a ASP .NET Core 6  
<https://docs.microsoft.com/en-us/aspnet/core/release-notes/aspnetcore-6.0?view=aspnetcore-6.0>
- [24] Dokumentacja języka programowania C# 9.0  
<https://docs.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-9>
- [25] Dokumentacja biblioteki Entity Framework  
<https://docs.microsoft.com/en-us/ef/core/>
- [26] Dokumentacja biblioteki GoogleOR  
<https://developers.google.com/optimization>

- [27] Dokumentacja biblioteki SignalR  
<https://docs.microsoft.com/en-us/aspnet/core/signalr/introduction>
- [28] Dokumentacja biblioteki Serilog  
<https://serilog.net/>
- [29] Dokumentacja systemu SendGrid  
<https://sendgrid.com/>
- [30] Dokumentacja biblioteki NUnit  
<https://nunit.org/>
- [31] Dokumentacja programu Android Studio  
<https://developer.android.com/studio>
- [32] Dokumentacja języka programowania Java  
<https://docs.oracle.com/en/java/>
- [33] Dokumentacja języka XML  
<https://www.xml.com/>
- [34] Dokumentacja biblioteki Material  
<https://material.io/develop/android/docs/getting-started>
- [35] Dokumentacja biblioteki Volley  
<https://developer.android.com/training/volley>
- [36] Dokumentacja biblioteki SignalRJava  
<https://docs.microsoft.com/en-us/aspnet/core/signalr/java-client?view=aspnetcore-6.0>
- [37] Dokumentacja biblioteki Slf4j  
<https://www.slf4j.org/manual.html>
- [38] Dokumentacja formatu GSON  
<https://github.com/google/gson>
- [39] Dokumentacja biblioteki JUnit  
<https://junit.org/junit4/>
- [40] Dokumentacja biblioteki AndroidTest  
<https://developer.android.com/training/testing/instrumented-tests/androidx-test-libraries/test-setup>

## Spis rysunków

1.1	Schemat aktualnego procesu dopasowywania planu zajęć . . . . .	22
3.1	Zależności pomiędzy modułami, interfejsami i API . . . . .	35
3.2	Diagram interfejsów . . . . .	41
3.3	Diagram klas aplikacji mobilnej . . . . .	42
3.4	Zależności pomiędzy modułami . . . . .	43
3.5	Klasy i zależności, z których utworzony został schemat bazy danych. . . . .	44
4.1	Logo aplikacji UsosFix . . . . .	45
4.2	Podstawowe kolory aplikacji mobilnej . . . . .	46
4.3	Porównanie kolorów elementów planu zajęć . . . . .	47
4.4	Czcionki aplikacji mobilnej . . . . .	47
4.5	Rozwinięte menu aplikacji mobilnej . . . . .	48
4.6	Pasek nawigacji i plan zajęć w aplikacji sieciowej na dużym ekranie . . . . .	49
4.7	Pasek nawigacji i plan zajęć w aplikacji sieciowej na małym ekranie . . . . .	50
4.8	Widoki planów w aplikacji mobilnej . . . . .	51
4.9	Plan przedmiotu w aplikacji sieciowej . . . . .	52
4.10	Widok grupy zajęciowej w aplikacji sieciowej na dużym ekranie . . . . .	53
4.11	Widok grupy zajęciowej w aplikacji sieciowej na małym ekranie . . . . .	54
4.12	Widok grupy zajęciowej w aplikacji mobilnej . . . . .	55
4.13	Widok zespołów użytkownika na dużym ekranie . . . . .	56
4.14	Potwierdzenie wykonania akcji . . . . .	57
4.15	Widoki zespołów użytkownika na małych ekranach . . . . .	58
4.16	Widok zespołu, do którego zaproszony jest użytkownik w aplikacji mobilnej . . .	59
4.17	Widok wymian użytkownika na dużym ekranie . . . . .	59
4.18	Widok wymian użytkownika na małym ekranie . . . . .	60
4.19	Wyświetlanie wymian w aplikacji mobilnej . . . . .	61
4.20	Dodawanie relacji miedzy wymianami na dużym ekranie . . . . .	62

4.21 Dodawanie relacji między wymianami na małym ekranie . . . . .	63
4.22 Wybieranie wymian do nadania relacji w aplikacji mobilnej . . . . .	63
4.23 Widok panelu użytkownika (1/3) . . . . .	64
4.24 Powiadomienie o powodzeniu . . . . .	64
4.25 Panel użytkownika aplikacji mobilnej . . . . .	65
4.26 Widok panelu użytkownika (2/3) . . . . .	66
4.27 Panel starosty aplikacji mobilnej . . . . .	67
4.28 Okno modalne z potwierdzeniem akcji . . . . .	68
4.29 Widok panelu starosty na małym ekranie . . . . .	69
4.30 Widok panelu użytkownika (3/3) . . . . .	70
4.31 Panel administratora aplikacji mobilnej . . . . .	71
4.32 Błąd walidacji . . . . .	71
4.33 Okno modalne do potwierdzenia zmiany uprawnień . . . . .	72
4.34 Widok wiadomości na dużym ekranie . . . . .	73
4.35 Wybór użytkowników do konwersacji . . . . .	74
4.36 Widok wiadomości na małym ekranie . . . . .	75
4.37 Widoki modułu wiadomości aplikacji mobilnej . . . . .	76
4.38 Ekran logowania aplikacji mobilnej . . . . .	77
4.39 Ekran logowania aplikacji sieciowej . . . . .	78
4.41 Okna modalne informujące o rezultacie operacji . . . . .	80
4.42 Logowanie do USOSa na urządzeniu mobilnym . . . . .	81

## **Spis tabelic**

4.1 Akcje, jakie może wykonać zalogowany użytkownik względem innych uczestników zespołu . . . . .	56
---	----