Politechnika Wrocławska
Wydział Informatyki i Telekomunikacji

Kierunek: IST

ZESPOŁOWE PRZEDSIĘWZIĘCIE INFORMATYCZNE

System Optymalizacji Ewolucyjnej Harmonogramów

Piotr Bonar
Jakub Borsuk
Aleksander Stepaniuk
Kacper Zakrzewski

Opiekun pracy
dr hab. inż. Michał Przewoźniczek

Słowa kluczowe: harmonogramowanie, algorytmy genetyczne, optymalizacja, plany zajęć

WROCŁAW (2025)

Spis treści

DOKUMENTACJA PROJEKTOWA

Cel i zakres przedsięwzięcia

Cel przedsięwzięcia

Celem przedsięwzięcia jest stworzenie systemu umożliwiającego firmom automatyczne tworzenie harmonogramów. Stworzone harmonogramy będą tworzone na podstawie preferencji uczestników, poprawiając jednocześnie efektywność pracowników tworzących harmonogramy jak i zadowolenie tworzących

Zakres przedsięwzięcia

W celu stworzenia systemu zespół stworzył 3 główne moduły:

- Frontend (Next.js) – obsługuje użytkowników, pozwala im komunikować się z resztą systemu.
- REST API (Django Python) – odbiera dane od frontendu, formatując je i tworząc powiązane encje w bazie danych
- Optymalizator (C++) – tworzy harmonogramy na podstawie zformatowanych danych odebranych od REST API

Oraz moduły pomocnicze:

- Bazę danych PostgreSQL
- Zarządcę wiadomości (Redis) – otrzymuje gotowe dane i przydziela je do instancji optymalizatora
- Progress listener – zwraca harmonogramy wytworzone przez optymalizator do bazy danych
- Optimization scheduler – sprawdza czy dana rekrutacja powinna zacząć optymalizację

Założenia wstępne

Dobór technologii

Technologie zostały wybrane na podstawie ogólnej wydajności i komfortu zespołu z danym językiem/frameworkiem. Optymalizator został napisany w C++ z powodu wydajności języka, jako że algorytmy genetyczne wymagają efektywnego użycia pamięci i dużej siły obliczeniowej. REST API został napisany w Django Python, kierując się znajomością systemu i wygodnej implementacji wymaganych technologii. Frontend został napisany w Next.js – frameworku znanego z wysokiej wydajności i modularności.

Przyjęte ograniczenia

- System musi działać jako aplikacja webowa dostępna z poziomu przeglądarki, bez tworzenia wersji natywnej.
- Harmonogram może być generowany tylko dla jednej rekrutacji naraz, aby uniknąć nadmiernego obciążenia serwera.
- Zajęcia są wielokrotnością 15 minut, aby zmniejszyć szerokość obszaru przeszukiwania do realistycznej wielkości.
- System nie przewiduje integracji z kalendarzami zewnętrznymi (np. Kalendarz Google.)
- Generowanie harmonogramu jest wywoływane przez system, gdy spełnione są wymagania, użytkownik nie ma możliwości zacząć optymalizacji bezpośrednio.

Słownik pojęć

Pojęcie	Nazwa w systemie	Definicja
Administrator firmy / Sekretariat	Office	Użytkownik zalogowany z rolą ‘office’ jest administratorem swojej firmy. Ma uprawnienia do tworzenia i zarządzania użytkownikami, pokojami i rekrutacjami firmy.
Cecha	Tag	Użytkownicy administracji firmy mogą tworzyć i dodawać do pokojów dowolne cechy. Cechy określają posiadane przez pokój zasoby wymagane do prowadzenia spotkań. Przy tworzeniu rekrutacji temat może mieć wymagane cechy, które ograniczą zakres pokojów tylko do tych posiadających wymagane cechy

Firma	Organization	System obsługuje wiele firm, każdy użytkownik, pokój, cecha i grupa posiada przypisaną firmę (ustalaną przy tworzeniu przez użytkownika administracji firmy).
Grupa	Group	Grupa użytkowników (na przykład kierunek, zespół). Grupy są tworzone przez użytkowników administracji firmy którzy mogą potem dodawać do nich uczestników. Dany uczestnik może należeć do wielu grup.
Harmonogram	Schedule	Harmonogram to grupa spotkań zwracana przez proces optymalizacji jako wynik rekrutacji.
Jakość	Fitness	Jakość to ocena harmonogramu. Obliczana jest ogólna jakość (jak plan wpasowuje się w preferencje wszystkich użytkowników), ogólna jakość użytkownika (jak spotkania, do którego należy użytkownik wpasowuje się w jego wszystkie preferencje) oraz dokładna jakość użytkownika (jak plan wpasowuje się w każdą oddzielną preferencje użytkownika)
Ograniczenia	Constraints	Zbiór reguł ustalanych przy tworzeniu rekrutacji, które muszą być zachowane, aby dany harmonogram był prawidłowy. Są to między innymi początek i koniec dnia, wymagane cechy dla pokoju.
Optymalizator	Optimizer	Moduł otrzymujący ograniczenia i preferencje, zwracający harmonogramy z wykorzystaniem algorytmu genetycznego.
Pokój	Room	Pokój to miejsce odbywania się spotkań. Posiada pojemność, identyfikator cechy i tabelę niedostępności, która gwarantuje, że harmonogramy nie będą nachodzić na siebie
Preferencje	Preferences	Zbiór reguł dla danej rekrutacji i użytkownika, który zwiększa jego zadowolenie, jeśli zwracany harmonogram spełnia je. Preferencje w odróżnieniu od ograniczeń harmonogram może łamać, ale obniży to jakość planu.
Przewodniczący	Host	Użytkownik prowadzący zajęcia. Podczas rekrutacji może zaznaczać preferencje tak samo jak uczestnik.
Rekrutacja	Recruitment	Proces tworzenia harmonogramów, w którym przypisani do niego użytkownicy wybierają swoje preferencje. Podczas tworzenia rekrutacji administracja firmy określa ograniczenia.
Spotkanie	Meeting	Spotkanie to blok czasu, prowadzący oraz grupa uczestników który jest częścią harmonogramu zwracanego poprzez proces optymalizacji.
Temat	Subject	Temat to ogólne dane spotkań, które mają zostać stworzone podczas optymalizacji. Temat posiada minimalną i maksymalną liczbę uczestników, wymagane cechy, grupy muszące uczestniczyć w spotkaniach oraz przewodniczących, którzy określają, ile spotkań powstanie
Tura	Round Length	Ustalany podczas tworzenia rekrutacji czas między turami optymalizacji. W każdej turze optymalizator otrzymuje najnowsze preferencje. Dłuższe okresy tury zwracają wyższej jakości harmonogramy.
Uczestnik	Participant	Uczestnik zajęć który może zaznaczać swoje preferencje do rekrutacji których jest członkiem
Użytkownik	User	Zalogowany użytkownik systemu
Waga	Weight	Waga to wartość przypisana dla każdego użytkownika określające jego wartość podczas tworzenia harmonogramu. Gdy dany użytkownik ma większą wagę jego preferencje mają większy wpływ na obliczaną jakość stworzonego harmonogramu.

Zadanie	Job	Zadanie to dane i rozwiązanie optymalizacji podczas jednej tury. Jest przechowywane w systemie dla dalszej analizy i wglądu administracji firmy

Specyfikacja wymagań na produkt programowe

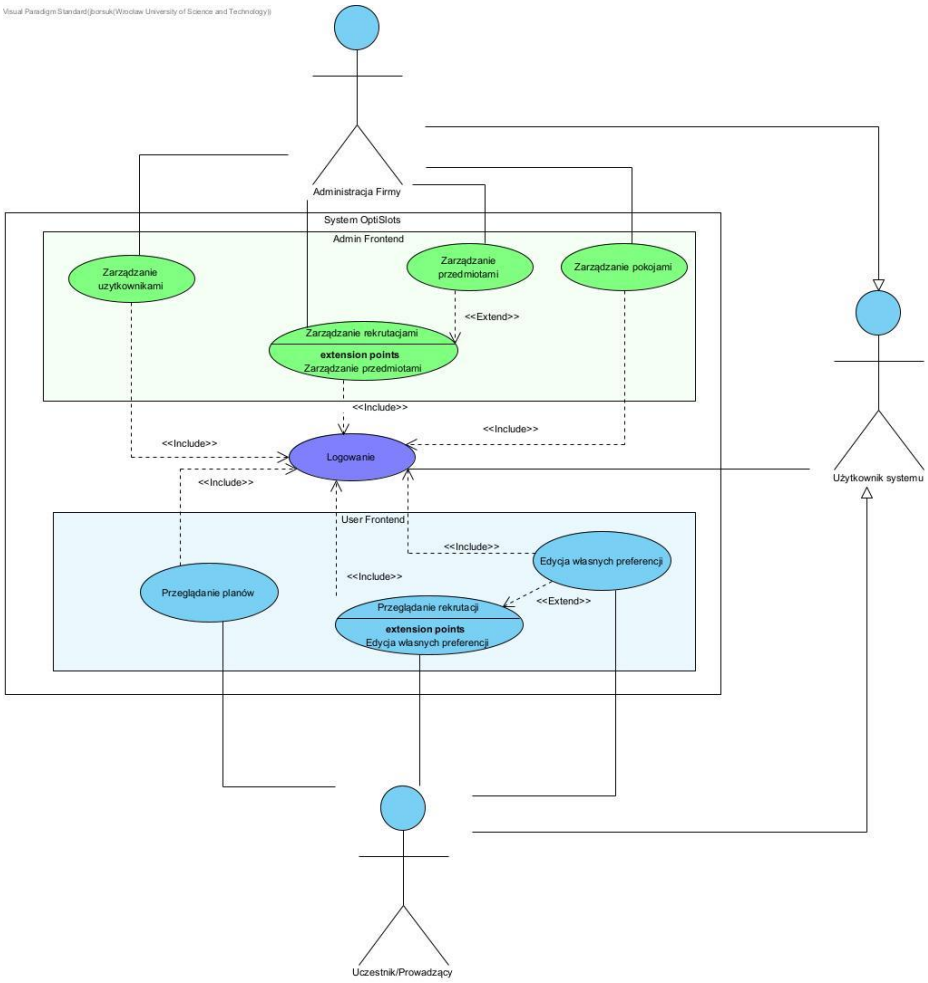
Wymagania funkcjonalne

- System umożliwia tworzenie rekrutacji wraz z parametrami (użytkownikami, przedmiotami, okresem czasowym itd.)
- Optymalizator generuje harmonogram na podstawie danych otrzymanych z backendu.
- System udostępnia wygenerowany harmonogram użytkownikom powiązanym z planem i zapisuje go w bazie.
- Użytkownik może się zalogować i wylogować.
- Administrator firmy może zarządzać kontami użytkowników (CRUD).
- Uczestnicy i prowadzący mogą definiować swoje preferencje dostępności (godziny, dni).
- System waliduje poprawność wprowadzonych danych (np. koniec zajęć nie może być przed początkiem).
- Uczestnicy i prowadzący mogą przeglądać historię wygenerowanych harmonogramów.

Wymagania niefunkcjonalne

- System musi być dostępny przez 99% czasu w skali miesiąca.
- Interfejs użytkownika musi być responsywny.
- Czas odpowiedzi backendu na standardowe zapytania nie powinien przekraczać 300 ms.
- Generowane plany muszą być ‘poprawne’ (nie naruszać zasad np.: sala/użytkownik zajęta/y w tym samym czasie)

Diagram przypadków użycia



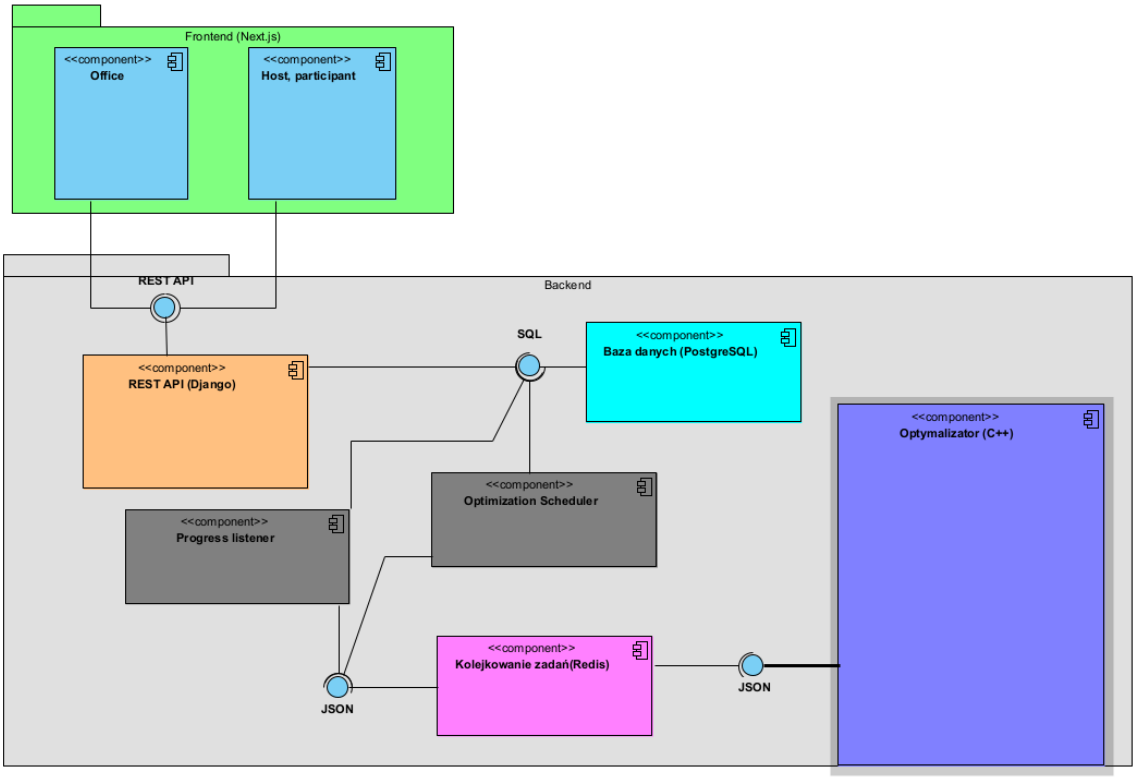
Rysunek 1: Diagram przypadków użycia

Projekt produktu programowego

Decyzje architektoniczne

System stosuje architekturę wielowarstwową dzieląc system na frontend, i backend. Komunikacja między frontendem a backendem odbywa się za pomocą REST API, które przesyła dane optymalizatorowi z użyciem modułu pomocniczego ‘scheduler’ który sprawdza spełnienie warunków do rozpoczęcia pracy nad planem. Gotowe plany zwracane są do bazy danych za pomocą modułu pomocniczego ‘progress listener’.

Projekt architektury



Rysunek 2: Diagram komponentów systemu OptiSlots

- **Frontend** – Przesyła zapytania API do backendu, który zwraca wymagane do dalszych działań dane.
- **REST API** – Wysyła i odbiera dane poprzez protokół HTTP, które po konwersji zapisuje do bazy danych.
- **Baza danych** – Przechowuje dane oraz wykonuje przesłane zapytania SQL
- **Optimization Scheduler** – Co minutę sprawdza czy zapisane harmonogramy mogą być przekazane do optymalizacji. Jeżeli warunki są spełnione, przesyła dane do optymalizacji.
- **Progress Listener** – Otrzymuje dane o ostatniej optymalizacji od Redis, po czym przekazuje je do bazy danych.
- **Redis** – Zarządza wiadomościami, kolejkując je do optymalizatora, odebrane po rundzie optymalizacji przekazuje do komponentu progress listener
- **Optymalizator** – Po otrzymaniu danych wejściowych zaczyna optymalizację dzieloną na tury (ustalane przez użytkownika). Zwraca ułożony harmonogram lub informacje o braku możliwości rozwiązania problemu.

Zastosowane wzorce projektowe

W projekcie zastosowano szereg wzorców projektowych i architektonicznych wspierających modularność oraz skalowalność. Poniżej przedstawiono najistotniejsze z nich oraz ich rolę w rozwiązaniu.

Wzorce architektoniczne

- **Client–Server** - System opiera się na klasycznej architekturze klient–serwer. Aplikacja frontendowa (Next.js) komunikuje się z serwerem poprzez REST API, co umożliwia separację logiki prezentacji od logiki biznesowej.
- **Komponentowa architektura** - Warstwa backendowa została podzielona na niezależne komponenty: REST API, Scheduler, Listener postępu, kolejka Redis, moduł optymalizatora w C++, oraz baza danych. Ułatwia to skalowanie i rozbudowę systemu.
- **Message-Driven Architecture** - Komunikacja pomiędzy komponentami backendu realizowana jest asynchronicznie za pomocą kolejki Redis. Poszczególne moduły wysyłają i odbierają komunikaty w formacie JSON, co odciąża API i poprawia wydajność.
- **Model-View-Template –** REST API zostało napisane przy użyciu Frameworku Django, opierającego się na architekturze Model-View-Template, która jest wariantem architektury Model-View-Controller.

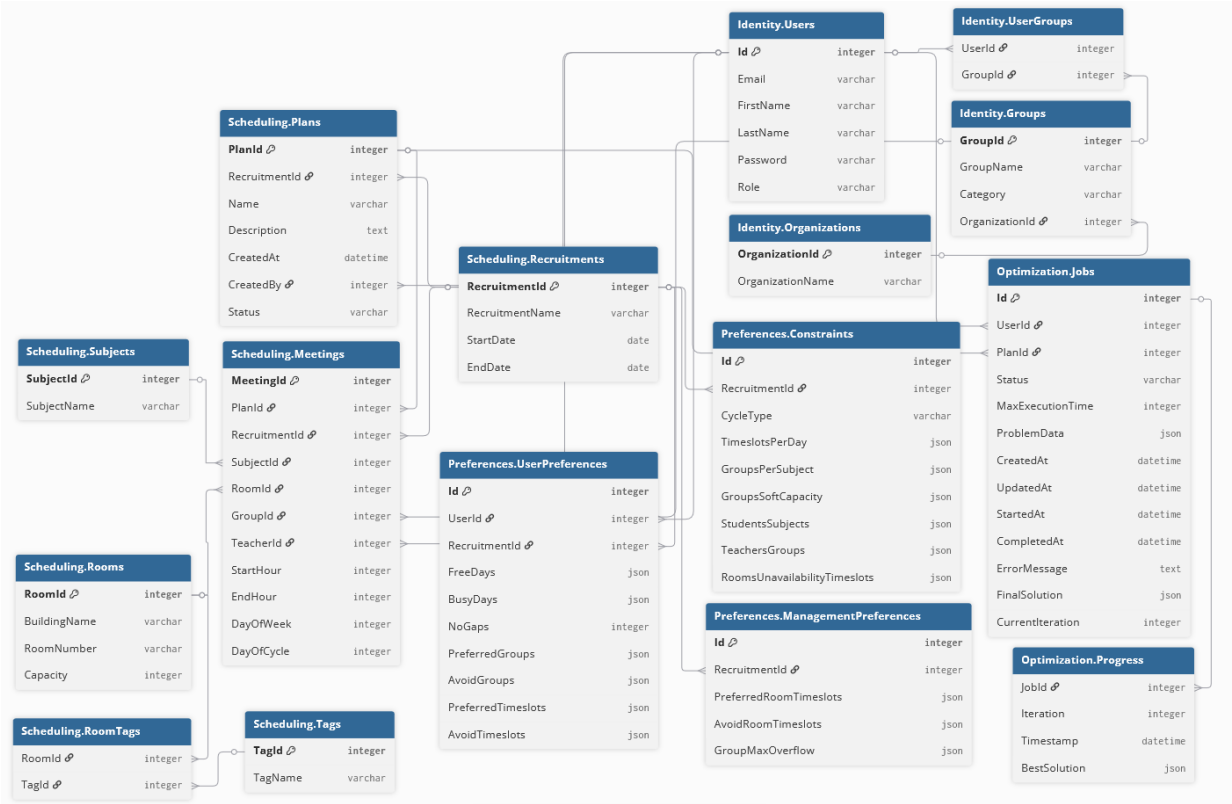
Wzorce komunikacyjne i integracyjne

- **REST API jako Facade** - Warstwa API pełni funkcję fasady, ukrywając złożoną logikę backendową (harmonogram, optymalizator, monitorowanie postępu) i dostarczając jednolity interfejs dla klienta.
- **Producer–Consumer (Redis)** - Zadania optymalizacyjne są umieszczane w kolejce (producer) i przetwarzane przez optymalizator (consumer). Gwarantuje to odporność systemu na przeciążenia.
- **Publish–Subscribe -Listener** nasłuchuje komunikatów wysyłanych po zakończeniu optymalizacji lub zmianie jej statusu. Umożliwia to asynchroniczne przesyłanie informacji zwrotnych do systemu.

Podsumowanie

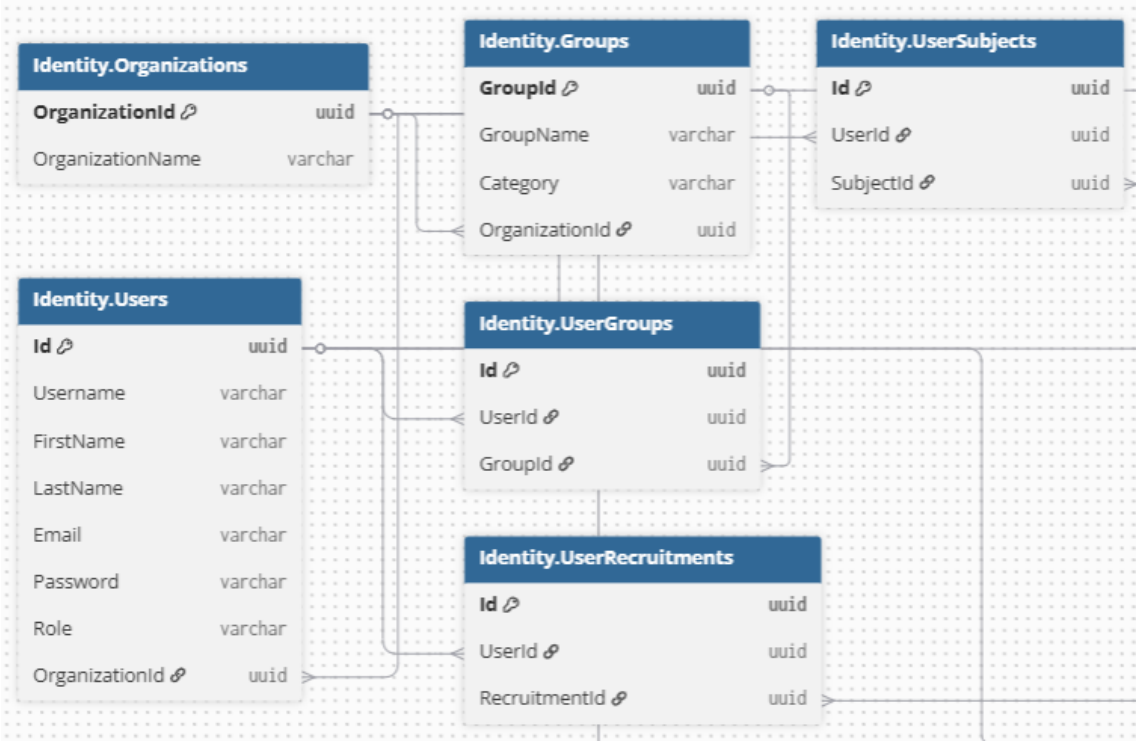
Zastosowanie powyższych wzorców pozwoliło na stworzenie systemu modularnego, łatwego w utrzymaniu, a jednocześnie wydajnego dzięki asynchronicznej komunikacji i izolacji poszczególnych komponentów.

Projekt bazy danych



Rysunek 3: Diagram pełnej bazy danych

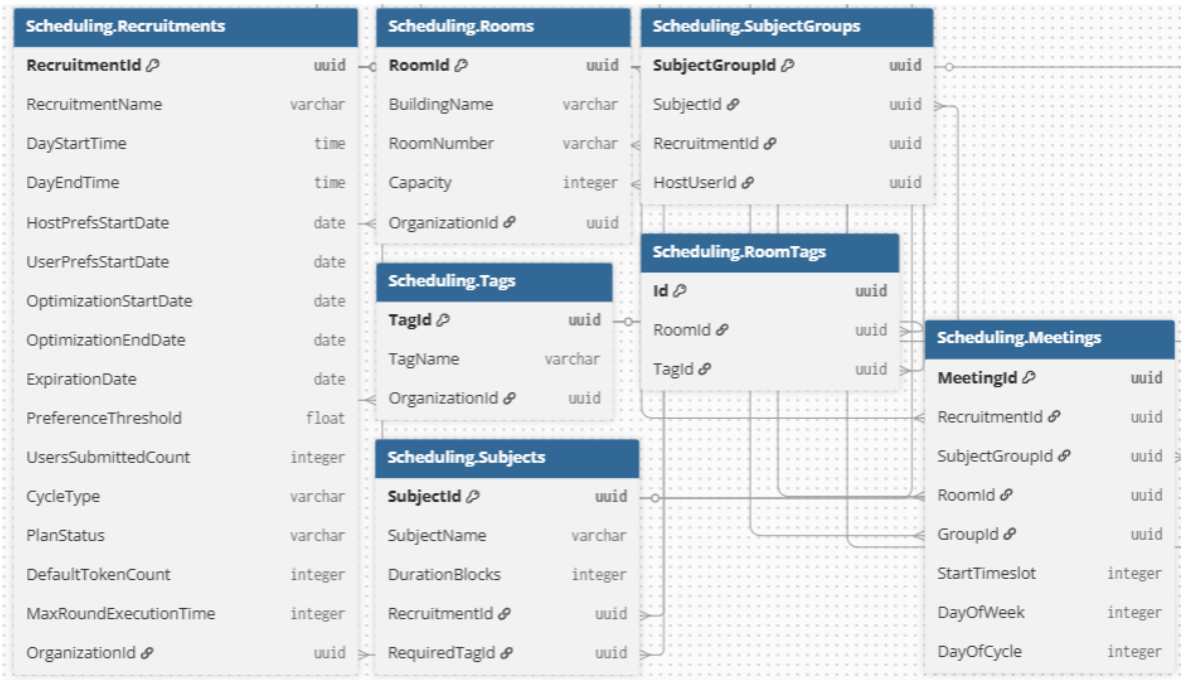
Szczegółowy diagram bazy danych



Rysunek 4: Tabele Identity w bazie danych

Część bazy danych przechowująca dane o użytkownikach.

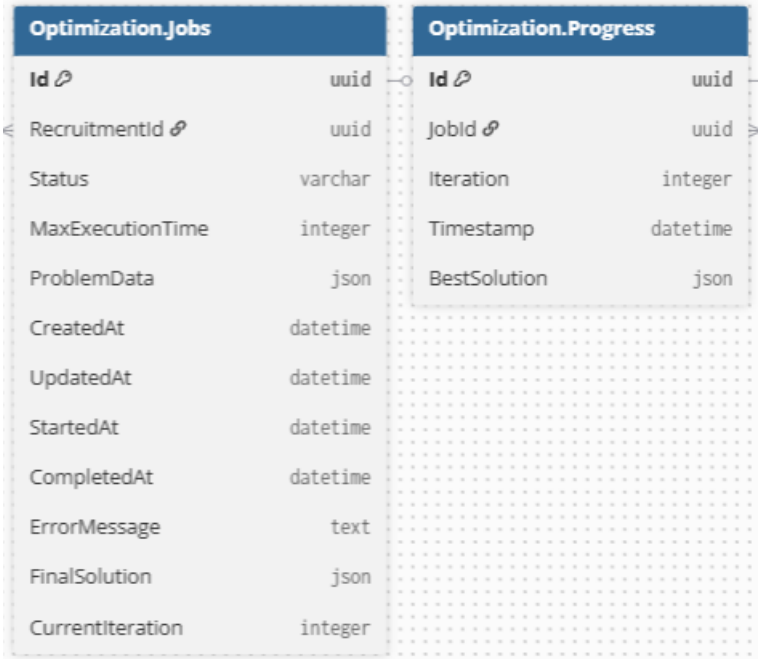
- **Identity.Users**: Tabela przechowujące dane użytkowników. Użytkowników mogą dodawać inni użytkownicy z rolą „Office” – członkowie administracji firmy.
- **Identity.Groups**: Tabela określająca grupy użytkowników – użytkownicy są przechowywani w grupach aby ułatwić dodawanie ich do rekrutacji
- **Identity.Organizations**: Tabela zawierająca organizacje – użytkownicy mogą edytować i przeglądać tylko części systemu należące do ich firmy.
- **Identity.UserGroups**: Tabela many-to-many określająca którzy użytkownicy należą do której grupy.
- **Identity.UserRecruitments**: Tabela many-to-many określająca którzy użytkownicy należą do której rekrutacji.



Rysunek 5: Tabele Scheduling w bazie danych

Część bazy danych zajmująca się tworzeniem harmonogramów

- **Scheduling.Subjects**: Tabela zawierająca tematy spotkań w rekrutacji, określająca ich cechy.
- **Scheduling.SubjectGroups**: Tabela określająca prowadzących, którzy są przypisani do danego tematu i prowadzą spotkanie. Ten sam prowadzący może być zapisany do tego samego przedmiotu wiele razy – oznacza to że prowadzi wiele grup uczestników.
- **Scheduling.Recruitments**: Tabela określająca daną rekrutacje. Użytkownicy mają dostęp do wielu opcji np.: określanie czy plan jest tygodniowy, dwutygodniowy, czy miesięczny, liczba punktów preferencji lub czasy wybierania preferencji jak i okres trwania planu.
- **Scheduling.Rooms**: Tabela określająca pokoje – dany pokój posiada też tablicę niedostępności wypełnianą przez optymalizator.
- **Scheduling.Tags**: Tabela określająca cechy które może posiadać pokój. Przy tworzeniu rekrutacji uzytkownik może określać jakie cechy pokoju są wymagane do prowadzenia tematu (np.: „Posiada projektor”, „Dostęp do komputerów”).
- **Scheduling.RoomTags**: Tabela określająca cechy danego pokoju.
- **Scheduling.Meetings**: Tabela określająca spotkania po wykonanej rekrutacji.



Rysunek 6: Tabele Optimization w bazie danych

Część bazy danych używanej przez optymalizator do tworzenia i zwracania wyników rekrutacji:

- **Optimization.Jobs**: Tabela zawierająca zadania do wykonania przez optymalizator. Backend tworzy zadanie gdy można zacząć tworzyć plan (gdy skończy się czas na wybieranie preferencji lub wystarczający procent użytkowników wypełni swoje preferencje).
- **Optimization.Progress**: Ostatnie rozwiązanie dla danego zadania, które zwrócił optymalizator.

Rysunek 7: Tabele Preferences w bazie danych

Część bazy danych używanej przetrzymująca dane o preferencjach użytkowników oraz zasadach rekrutacji podawanych przy tworzeniu.

- **Preferences.Constraints**: Tabela zawierająca zasady rekrutacji – dany plan nie może złamać zawartych w tej tabeli danych.
- **Preferences.UserPreferences**: Tabela zawierające preferencje danego użytkownika. Dotyczy specyficznego harmonogramu.

Tabele powiązań

Identity

LP	Tabela źródłowa	Kolumna FK	Tabela docelowa	Kolumna PK	Typ relacji
1	Identity.Users	OrganizationId	Identity.Organizations	OrganizationId	1-N
2	Identity.Groups	OrganizationId	Identity.Organizations	OrganizationId	1-N
3	Identity.UserGroups	UserId	Identity.Users	Id	M-N
4	Identity.UserGroups	GroupId	Identity.Groups	GroupId	M-N
5	Identity.UserRecruitments	UserId	Identity.Users	Id	M-N
6	Identity.UserRecruitments	RecruitmentId	Scheduling.Recruitments	RecruitmentId	M-N
7	Identity.UserSubjects	UserId	Identity.Users	Id	M-N
8	Identity.UserSubjects	SubjectId	Scheduling.Subjects	SubjectId	M-N

Scheduling

LP	Tabela źródłowa	Kolumna FK	Tabela docelowa	Kolumna PK	Typ relacji
1	Scheduling.Subjects	RecruitmentId	Scheduling.Recruitments	RecruitmentId	1-N
2	Scheduling.Subjects	RequiredTagId	Scheduling.Tags	TagId	1-N
3	Scheduling.Recruitments	OrganizationId	Identity.Organizations	OrganizationId	1-N
4	Scheduling.SubjectGroups	SubjectId	Scheduling.Subjects	SubjectId	1-N
5	Scheduling.SubjectGroups	RecruitmentId	Scheduling.Recruitments	RecruitmentId	1-N
6	Scheduling.SubjectGroups	HostUserId	Identity.Users	Id	1-N
7	Scheduling.Meetings	RecruitmentId	Scheduling.Recruitments	RecruitmentId	1-N
8	Scheduling.Meetings	SubjectGroupId	Scheduling.SubjectGroups	SubjectGroupId	1-N
9	Scheduling.Meetings	RoomId	Scheduling.Rooms	RoomId	1-N
10	Scheduling.Meetings	GroupId	Identity.Groups	GroupId	1-N
11	Scheduling.RoomTags	RoomId	Scheduling.Rooms	RoomId	M-N
12	Scheduling.RoomTags	TagId	Scheduling.Tags	TagId	M-N
13	Scheduling.Rooms	OrganizationId	Identity.Organizations	OrganizationId	1-N
14	Scheduling.Tags	OrganizationId	Identity.Organizations	OrganizationId	1-N

Preferences

LP	Tabela źródłowa	Kolumna FK	Tabela docelowa	Kolumna PK	Typ relacji
1	Preferences.UserPreferences	UserId	Identity.Users	Id	1-N
2	Preferences.UserPreferences	RecruitmentId	Scheduling.Recruitments	RecruitmentId	1-N
3	Preferences.Constraints	RecruitmentId	Scheduling.Recruitments	RecruitmentId	1-N

Optimization

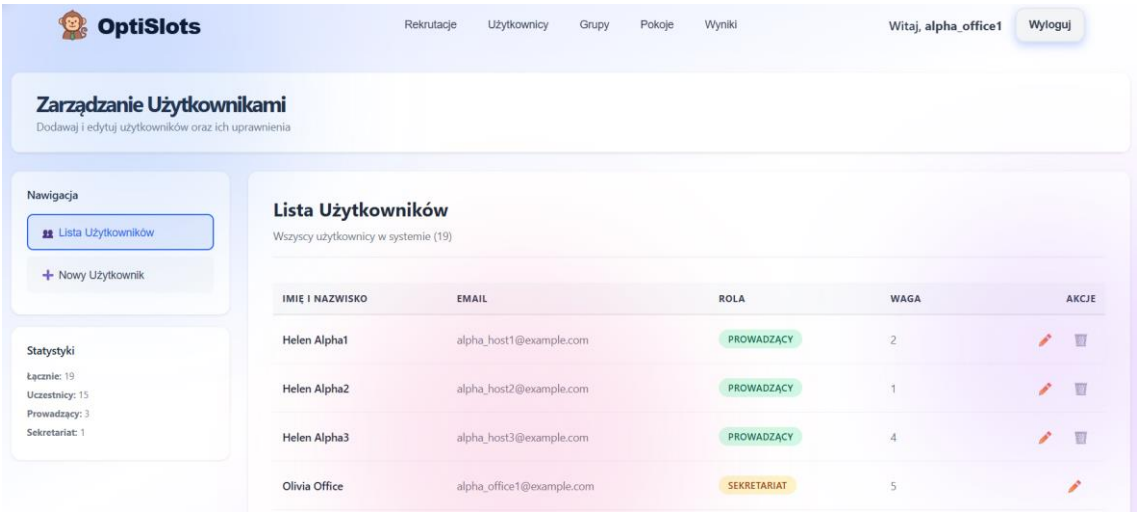
LP	Tabela źródłowa	Kolumna FK	Tabela docelowa	Kolumna PK	Typ relacji
1	Optimization.Jobs	RecruitmentId	Scheduling.Recruitments	RecruitmentId	1-N
2	Optimization.Progress	JobId	Optimization.Jobs	Id	1-N

Demonstracja produktu programowego

System obsługuje funkcjonalności związane z zarządzaniem użytkownikami, preferencjami i rekrutacjami. Poniżej znajdują się opisy i diagramy sekwencji związane z najczęściej wykonywanych zadań w aplikacji webowej.

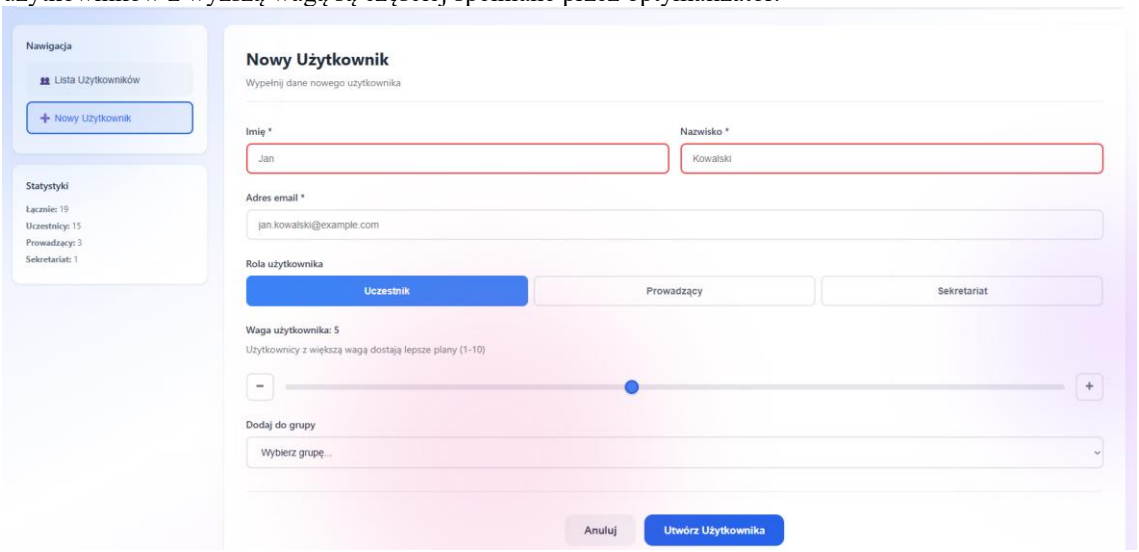
Dodawanie użytkownika

System pozwala dla członków administracji firmy dodawać użytkowników należących do tej samej firmy. Dani użytkownicy mogą należeć do grup uczestników, w celu łatwiejszego dodawania dużych grup uczestników do rekrutacji.



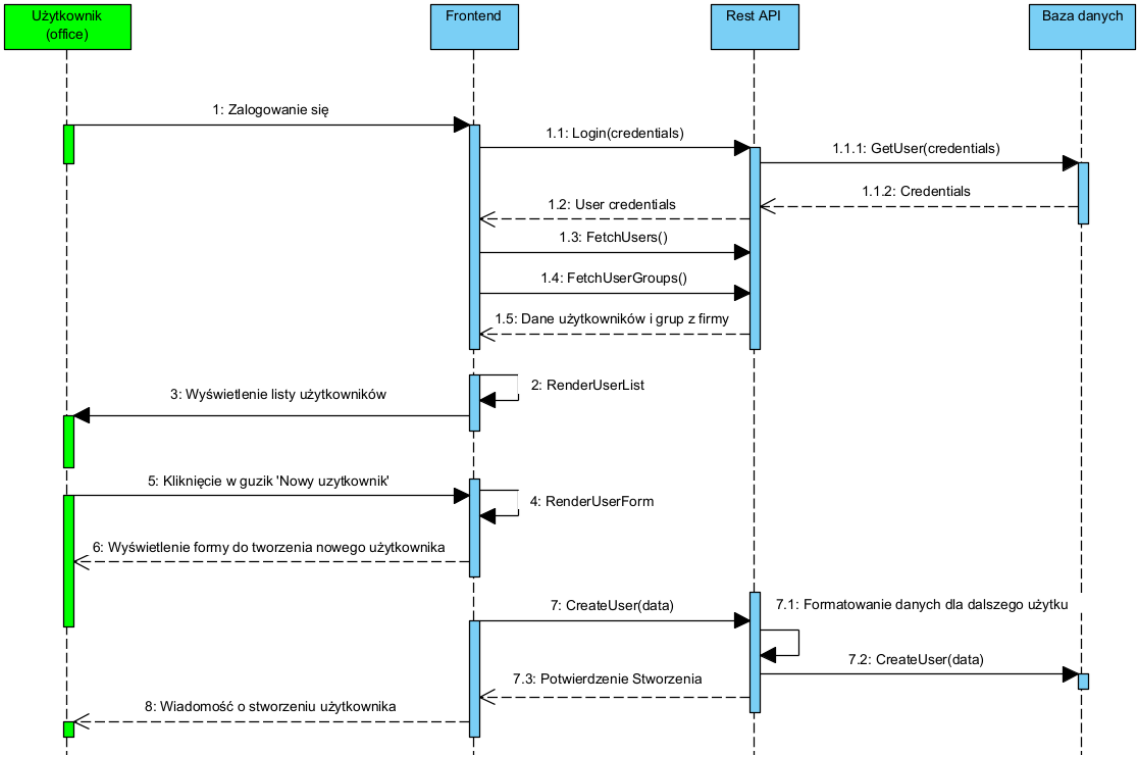
Rysunek 8: Widok listy użytkowników

W trakcie dodawania użytkownika system wymaga danych osobistych, ale hasło i nazwa użytkownika (wymagane do logowania się do systemu) są tworzone automatycznie i losowo przez system. Jeżeli w systemie istnieją już grupy, system pozwala na stworzenie użytkownika należącego do danych grup. Waga użytkownika dyktuje wpływ jego preferencji na stworzony harmonogram – preferencje użytkowników z wyższą wagą są częściej spełniane przez optymalizator.



Rysunek 9: Widok formularza dodawania użytkownika

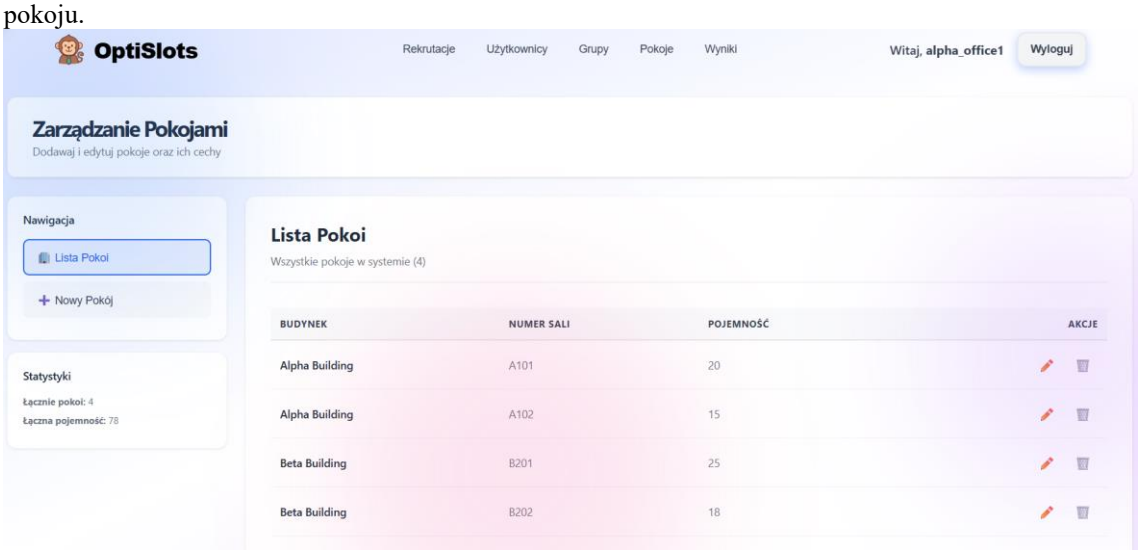
Tworzenie użytkownika może być opisane diagramem sekwencji.



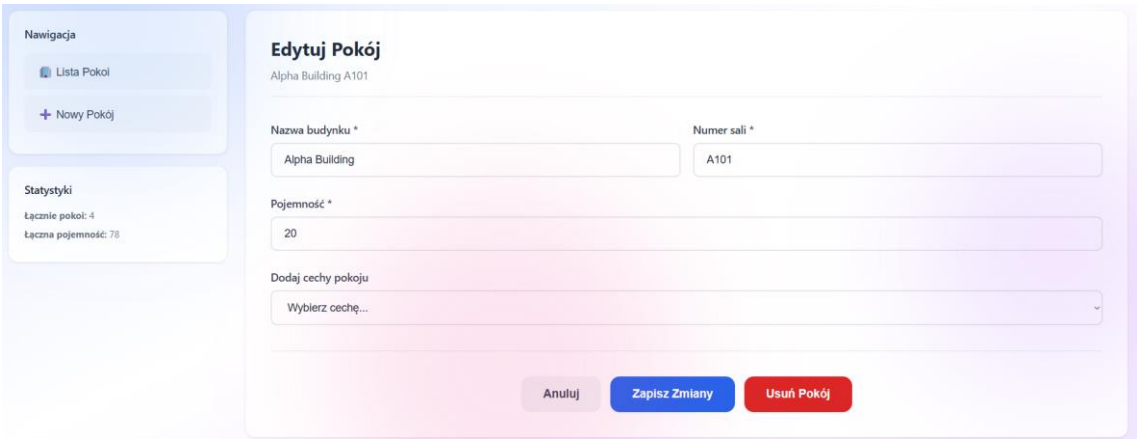
Rysunek 10: Diagram sekwencji - dodawanie użytkownika

Edycja pokoju

Administracja może zarządzać pokojami – dodawać, usuwać oraz edytować cechy danego pokoju.

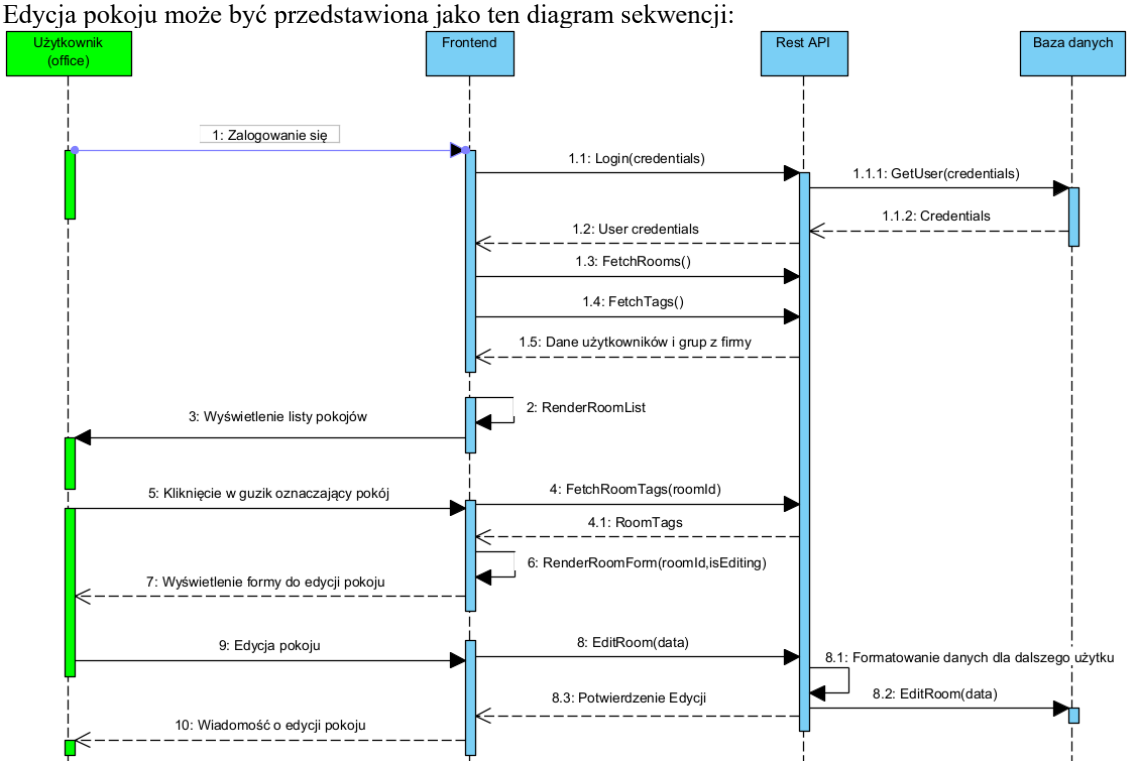


Rysunek 11: Widok listy pokoju



Rysunek 12: Widok formularza edycji pokoju

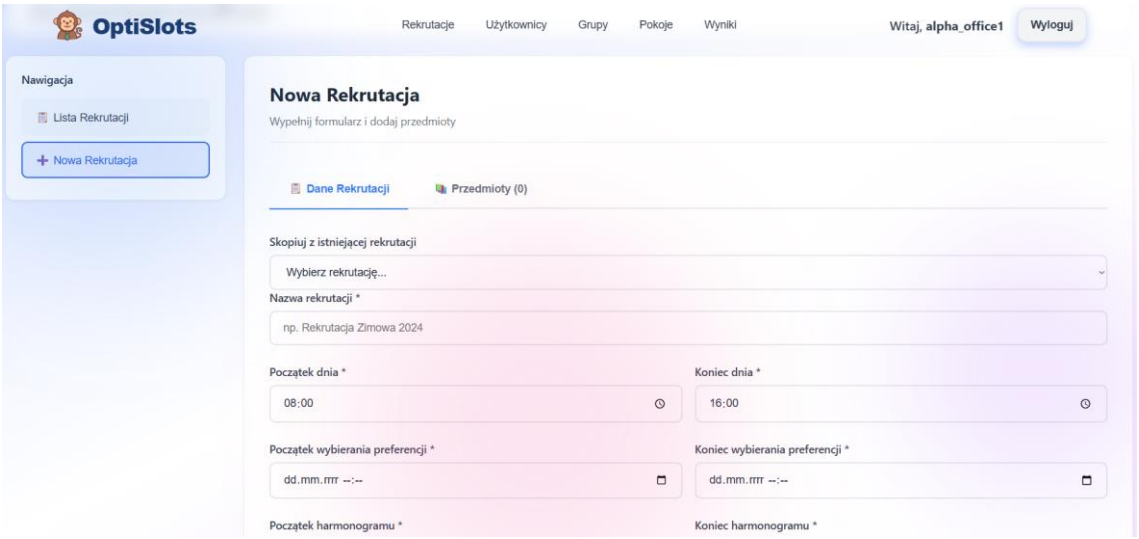
Edycja pokoju może być przedstawiona jako ten diagram sekwencji:



Rysunek 13: Diagram sekwencji - Edycja pokoju

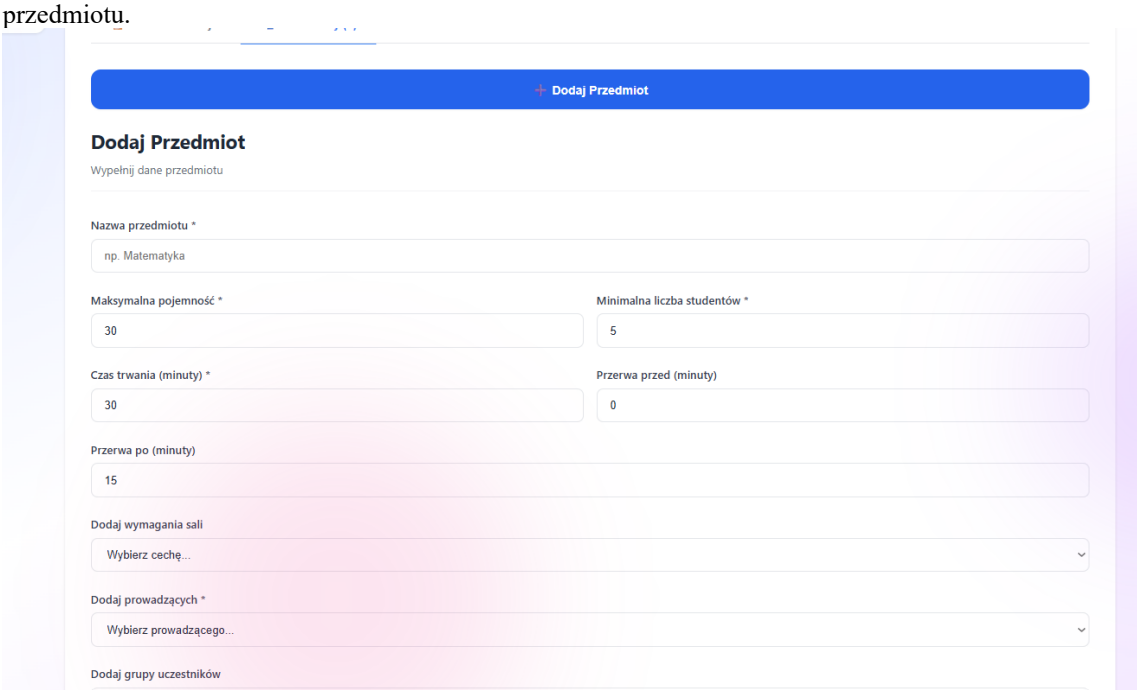
Tworzenie rekrutacji

Administracja firmy ma możliwość tworzenia rekrutacji, których zwrotem jest harmonogram. Rekrutacja jest w pełni dostosowywalne – administracja może ustalać czas rozpoczęcia i zakończenie dnia, daty początku i końca wybierania preferencji przez użytkowników oraz okres aktywności powstałego harmonogramu.

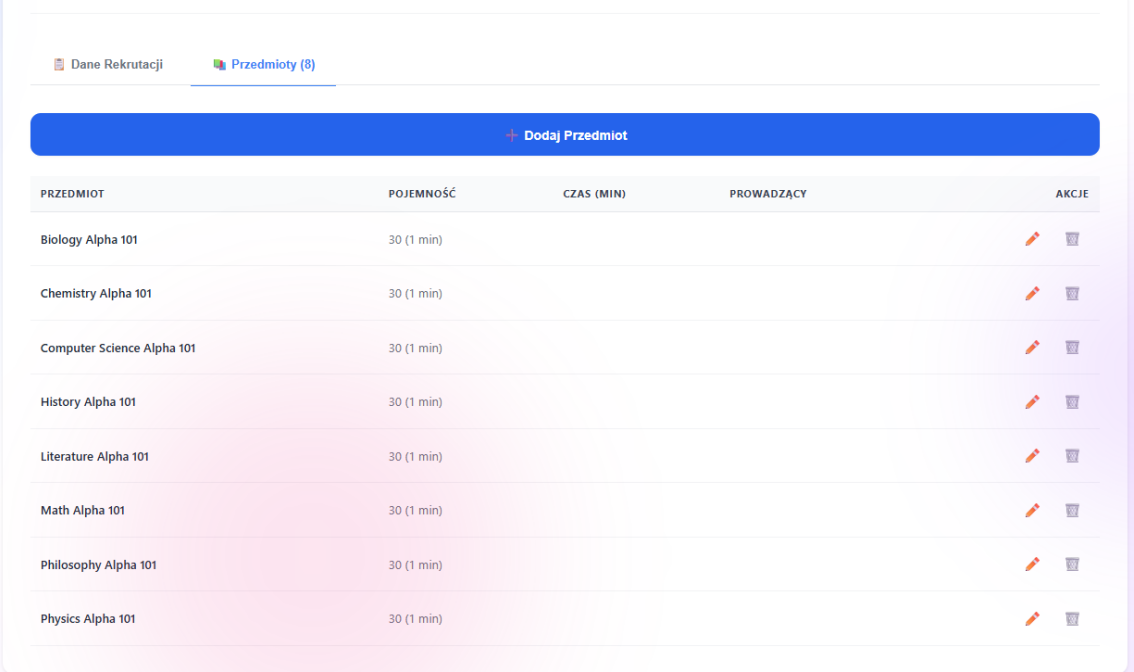


Rysunek 14: Widok formularza tworzenia rekrutacji

Rekrutacja składa się z przedmiotów. Każdy przedmiot posiada własne dane określające maksymalną i minimalną liczbę uczestników do stworzenia spotkania, czas trwania spotkania oraz czas przerwy przed i po spotkaniu. Każdy przedmiot posiada również co najmniej jedną grupę użytkowników która musi uczęszczać na spotkanie, oraz prowadzącego. Dany przedmiot może posiadać wymagane cechy – będzie przypisywany jedynie do pokojów posiadających wszystkie cechy wymagane do prowadzenia przedmiotu.

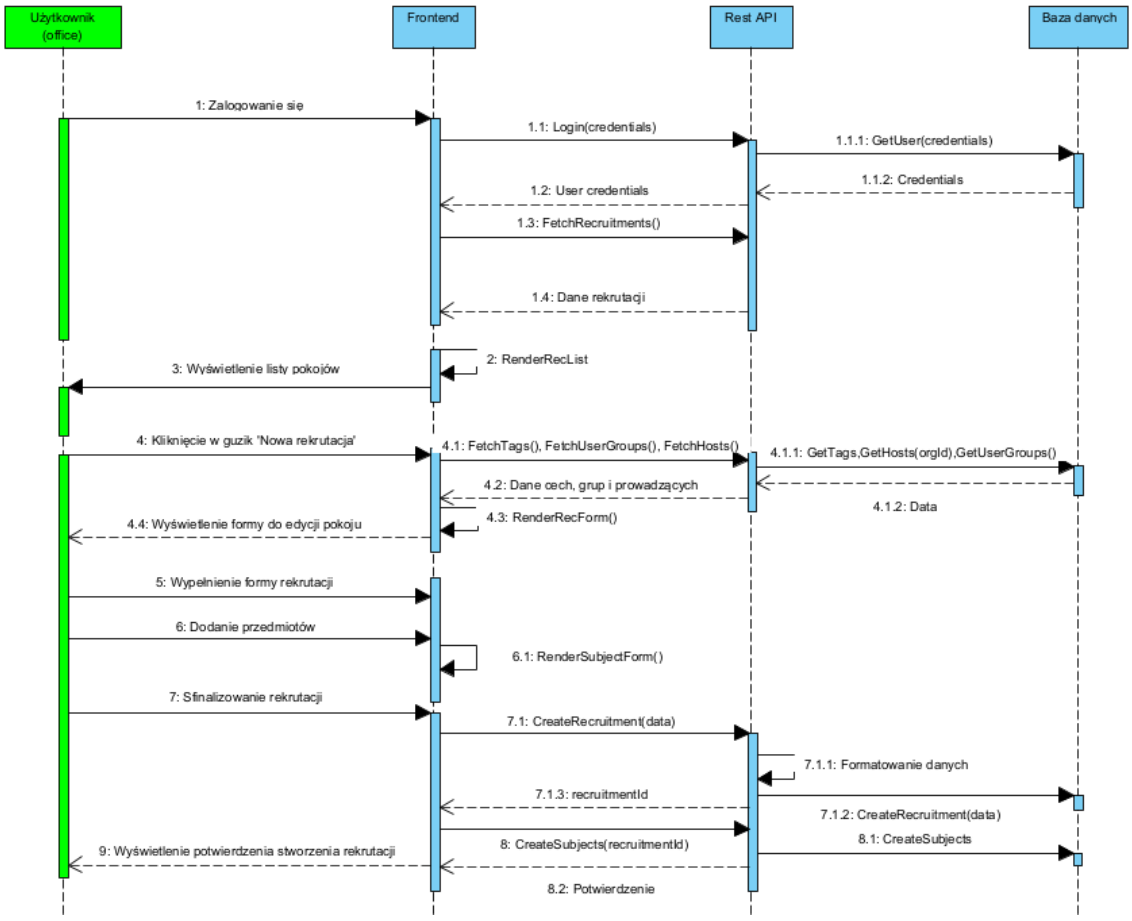


Rysunek 15: Widok formularza tworzenia przedmiotów



Rysunek 16: Przykładowy widok listy przedmiotów dla rekrutacji

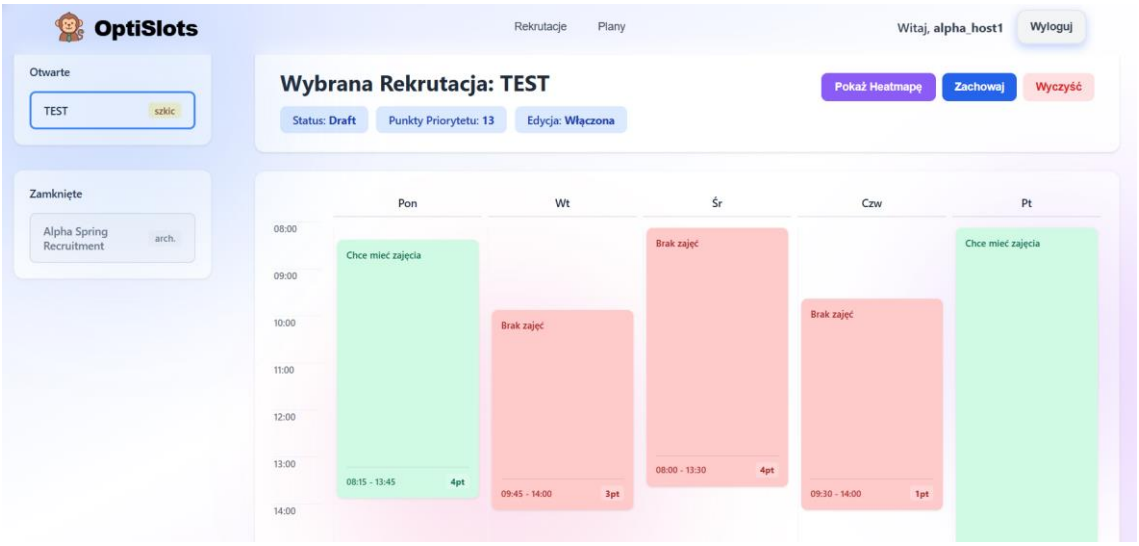
Tworzenie rekrutacji może być przedstawione tym diagramem sekwencji



Rysunek 17: Diagram sekwencji - tworzenie rekrutacji

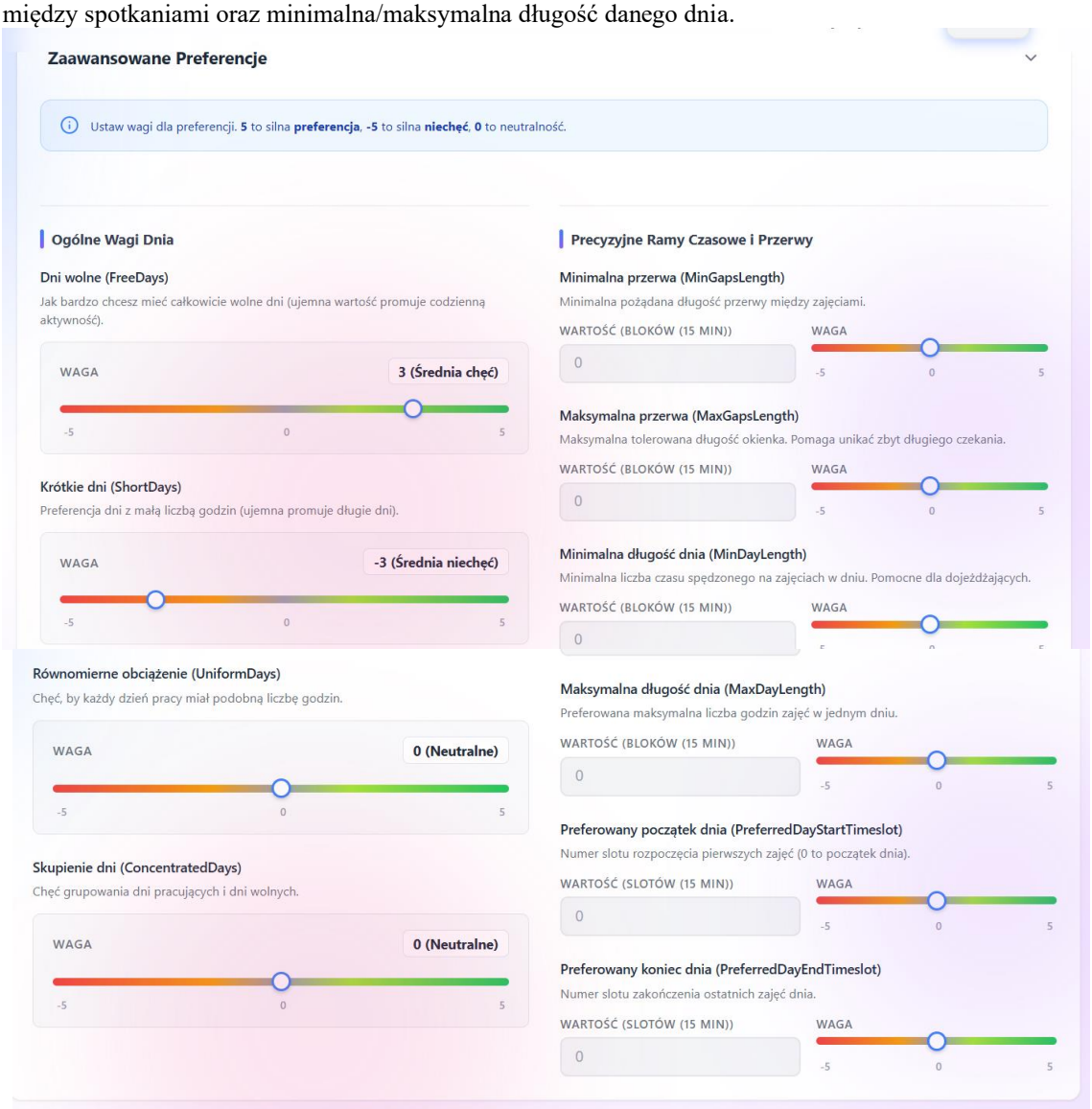
Wybieranie preferencji

Użytkownik zapisany na rekrutacje ma możliwość wybierania i edytowania swoich preferencji. Po zalogowaniu się ma możliwość wybrać dowolną aktywną rekrutacje (której jest członkiem) i zaznaczyć pasujące mu czasy spotkań oraz dokładne metryki określające styl harmonogramu.



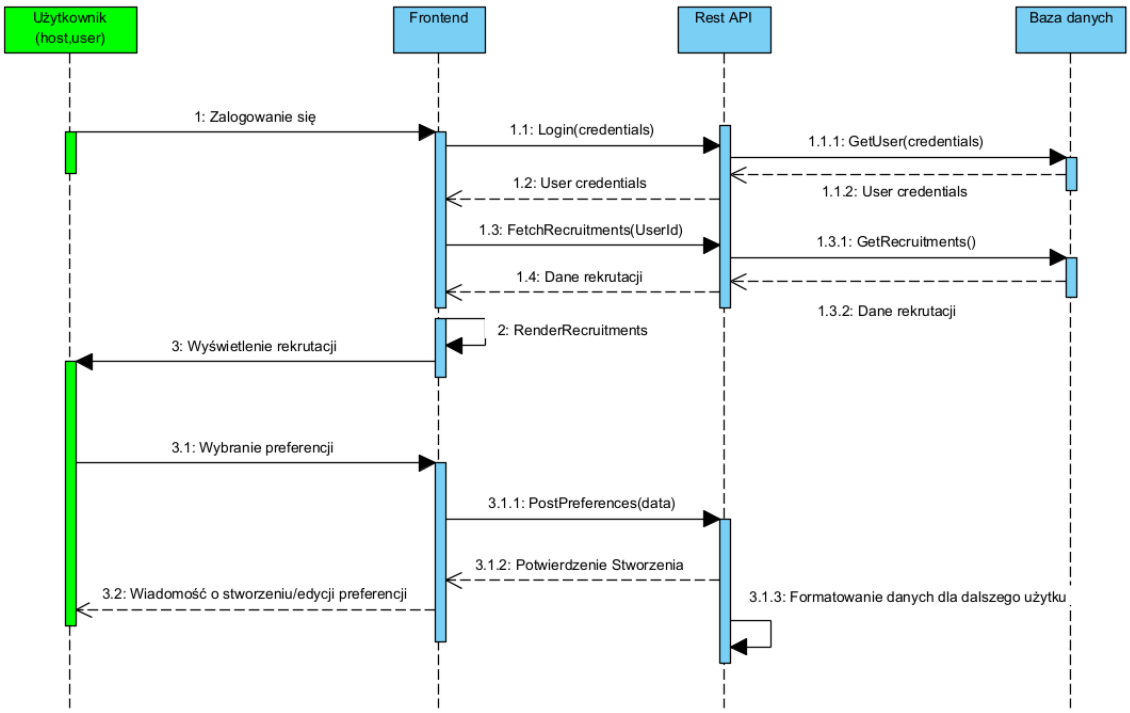
Rysunek 18: Widok formularza wybierania preferencji czasowych

Użytkownicy mają też dostęp do bardziej skomplikowanych preferencji, realizowanych w formie pasków przesuwania. Obsługiwane preferencje to między innymi: Liczba wolnych dni, maksymalna przerwa między spotkaniami oraz minimalna/maksymalna długość danego dnia.



Rysunek 19: Widok formularza zaawansowanych preferencji

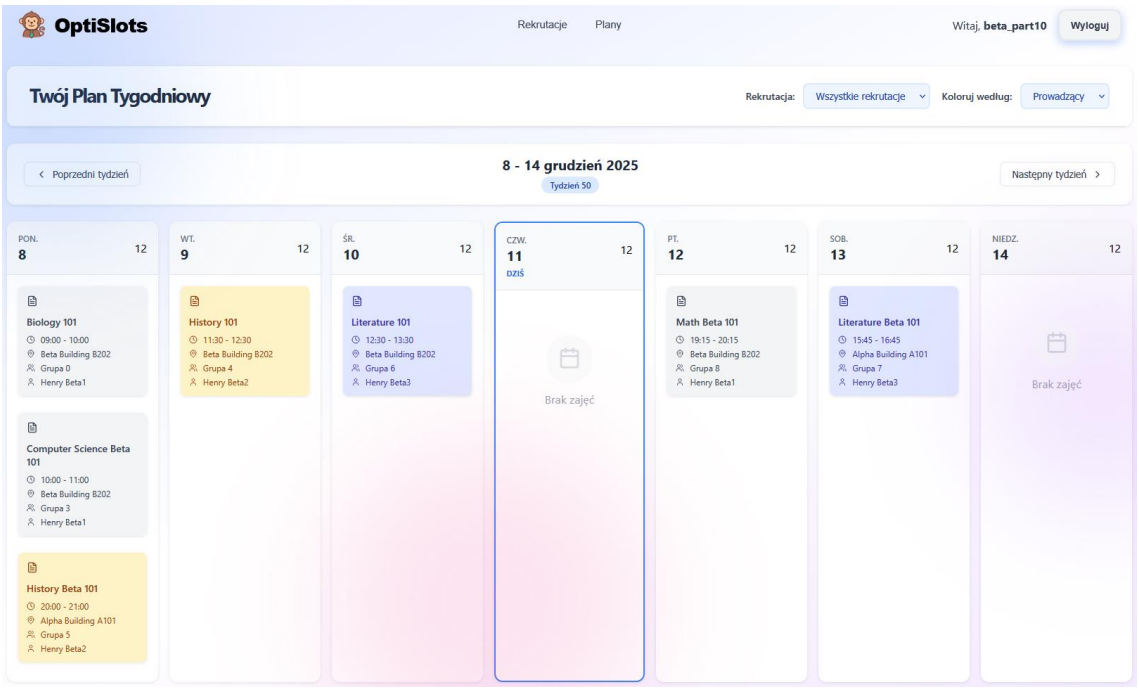
Proces wybierania preferencji może być opisany diagramem sekwencji.



Rysunek 20: Diagram sekwencji - wybieranie preferencji

Przeglądanie planu

Użytkownicy po zakończeniu rekrutacji mogą przeglądać wygenerowane plany. Plany rozłożone są tygodniowo, aby prawidłowo wyświetlać zajęcia z rekrutacji dwutygodniowych (raz na dwa tygodnie) i miesięcznych (raz na cztery tygodnie).

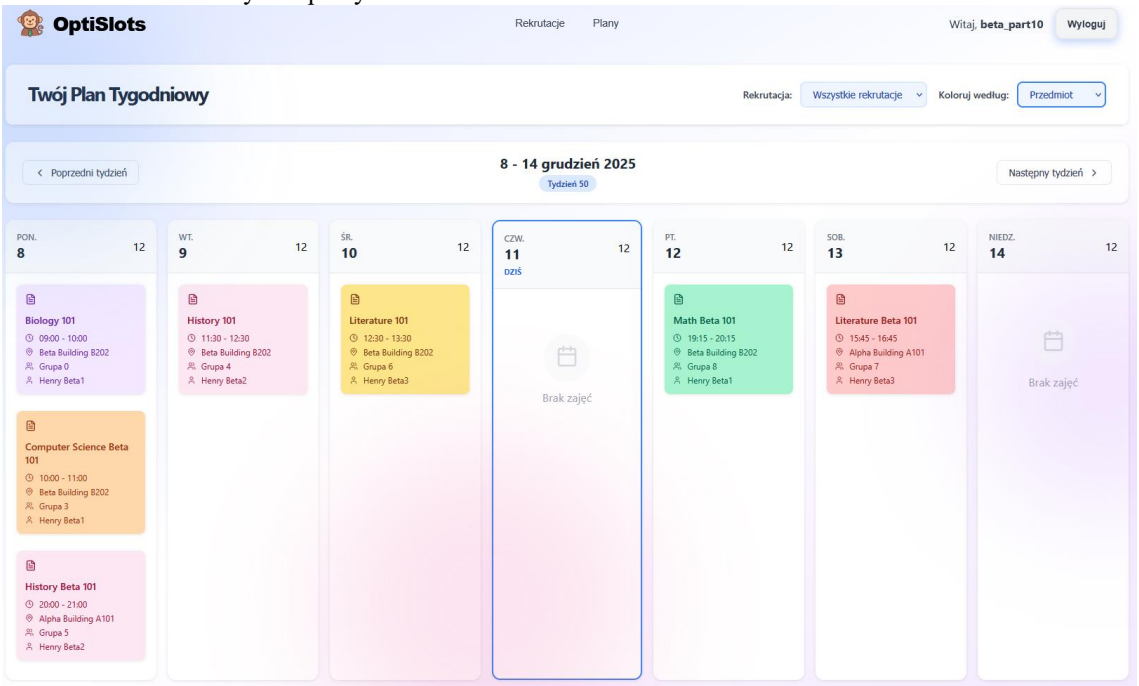


Rysunek 21: Widok planu

Użytkownicy mają dostęp do filtrowania przedmiotów po wielu metrykach. Po zaznaczeniu metryki spotkania zmienią wygląd reprezentując które spotkania dzielą tę samą metrykę. Dostępne metryki:

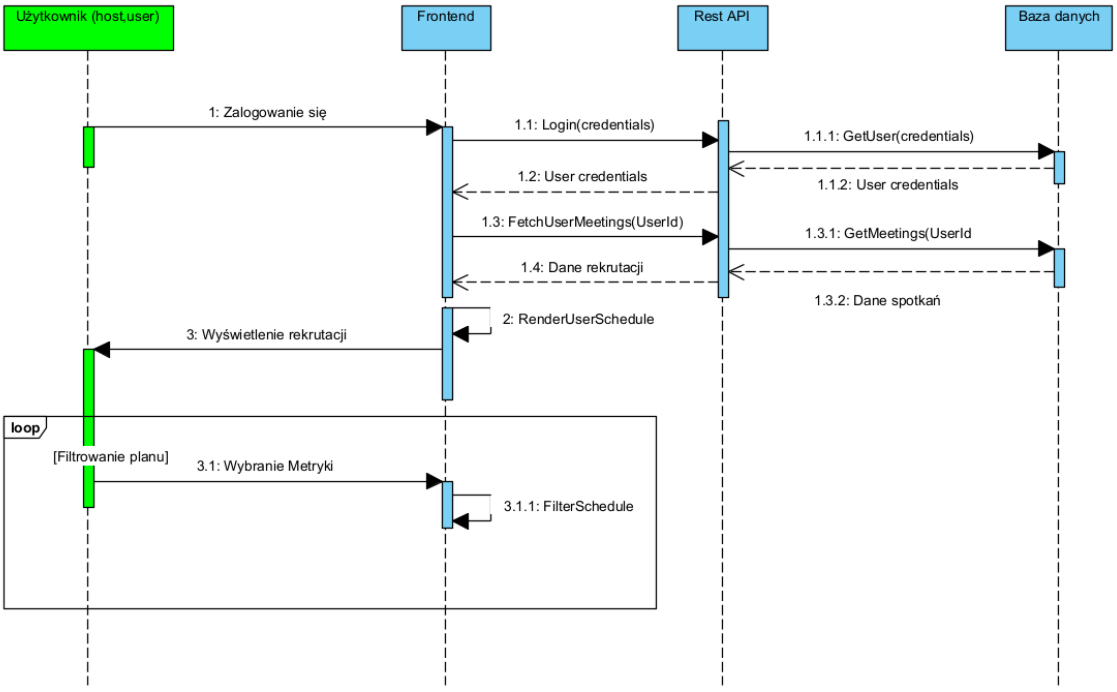
- Temat spotkania
- Prowadzący spotkania
- Pokój odbywania się spotkania
- Grupa użytkowników odbywająca spotkanie

- Aktualnie aktywne plany



Rysunek 22: Widok planu - posortowany po przedmiotach

Przeglądanie planu może być reprezentowane diagramem sekwencji



Rysunek 23: Diagram sekwencji - Przeglądanie planu