

Dokumentacja koncepcyjna

*| Projekt: Interaktywny system*

*wspomagania układania*

*rozkładu zajęć*

Roger Barlik

Krystian Ujma

Krzysztof Wróbel

Krzysztof Nowakowski

Spis treści

[1. Koncepcja 3](#_Toc417548692)

[1.1 Opis ogólny 3](#_Toc417548693)

[1.1.1 Ekran logowania 3](#_Toc417548694)

[1.1.1 Widok główny 4](#_Toc417548695)

[1.1.3 Kalendarz 5](#_Toc417548696)

[1.1.4 Obszar menu 5](#_Toc417548697)

[1.1.5 Alerty 5](#_Toc417548698)

[1.1.6 Zasobnik agregatów 6](#_Toc417548699)

[1.1.7 Agregat 6](#_Toc417548700)

[1.2 Opis szczegółowy 6](#_Toc417548701)

[1.2.1 Ekran logowania 6](#_Toc417548702)

[1.2.2 Widoki 7](#_Toc417548703)

[1.2.4 Agregat 9](#_Toc417548704)

[1.3 Funkcję dostępne dla użytkownika 15](#_Toc417548705)

[1.3.1 Wykonywanie operacji na wykładowcach 15](#_Toc417548706)

[1.3.2 Wykonywanie operacji na awatarach 15](#_Toc417548707)

[1.3.3 Wykonywanie operacji na grupach studenckich 15](#_Toc417548708)

[1.3.4 Wykonywanie operacji na liście agregatów 16](#_Toc417548709)

[1.3.5 Wyświetlanie widoków 16](#_Toc417548710)

[1.3.6 Wykonywanie operacji na szablonach 16](#_Toc417548711)

[1.3.7 Wykonywanie operacji na skrzynce widomości i alertów 16](#_Toc417548712)

[2. Moduł GWT 17](#_Toc417548713)

[2.1 Opis ogólny 17](#_Toc417548714)

[2.2 Budowa GWT 17](#_Toc417548715)

[2.3 Kompilator GWT 17](#_Toc417548716)

[2.4 GWT UI 18](#_Toc417548717)

[2.5 RPC 18](#_Toc417548718)

[2.6 Dodatkowe narzędzia 18](#_Toc417548719)

[2.7 GWT Development Mode 18](#_Toc417548720)

[3. Architektura systemu 19](#_Toc417548721)

[3.1 Przepływ danych 19](#_Toc417548722)

[3.2 Moduł backend 19](#_Toc417548723)

[3.2.1 Warstwa kontrolerów 20](#_Toc417548724)

[3.2.2 Warstwa Modelu 20](#_Toc417548725)

[3.3 Moduł frontend 21](#_Toc417548726)

[4. Diagramy przypadków użycia 22](#_Toc417548727)

[4.1. Zarządzanie planami zajęć 22](#_Toc417548728)

[1.1 Zarządzanie planami zajęć 22](#_Toc417548729)

[4.2 Zarządzanie przedmiotami 29](#_Toc417548730)

[4.3 Zarządzanie grupami studenckimi 30](#_Toc417548731)

[4.4 Tworzenie zajęcia wraz z terminem 30](#_Toc417548732)

[4.5. Tworzenie terminu 31](#_Toc417548733)

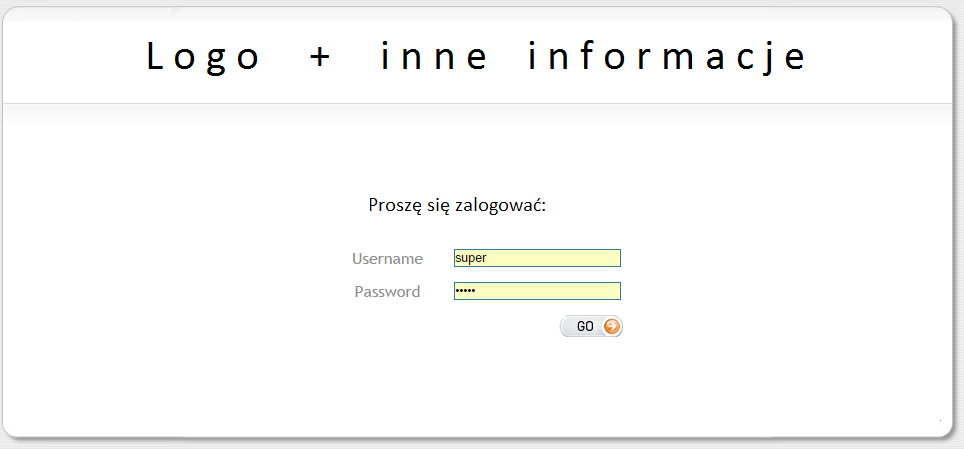
# 1. Koncepcja

## 1.1 Opis ogólny

Głównym zadaniem aplikacji jest wspomaganie planisty w jego zadaniu tworzenia planu zajęć dla poszczególnych grup studenckich uwzględniając dostępność poszczególnych sal oraz prowadzących. Po uwzględnieniu wymagań funkcjonalnych system został zbudowany z wybranych elementów: ekranu logowania, widoków, obszaru menu, alertów, kalendarza, zasobników agregatów oraz samych agregatów.

### 1.1.1 Ekran logowania

Ekran logowania jest punktem startowym całej aplikacji i to z nim użytkownik będzie miał pierwszy kontakt. Jego głównym celem jest potwierdzenie tożsamości użytkownika który próbuje uzyskać dostęp do systemu oraz jego uprawnień.

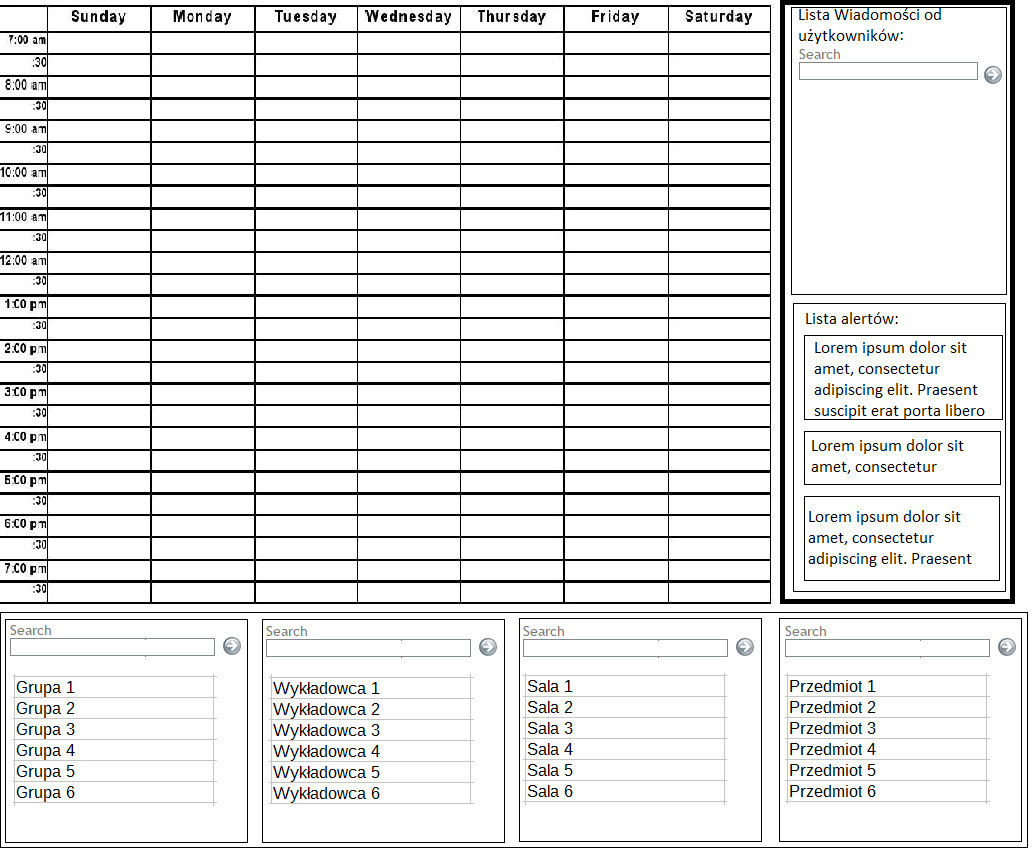


Rysunek : Przykładowy wygląd ekranu logowania

Ekran logowania nie odbiega od standardowych ekranów logowania istniejących w innych aplikacjach. Użytkownik wpisuje swój login (adres e-mail) i hasło. Dodatkowo użytkownik będzie miał możliwość przypomnienia swojego hasła oraz edycje swojego profilu.

### 1.1.1 Widok główny

Widok jest głównym obszarem roboczym na którym będzie operował użytkownik. Widoki dzielą się na widok główny oraz widoki szczegółowe. Na widoki składa się: kalendarz, alerty oraz zasobnik agregatów.



Rysunek : Przykładowy układ widoku głównego

Użytkownik ma możliwość otwarcia więcej niż jednego widoku naraz w celu usprawnienia przeprowadzania operacji. Operacja ta jest możliwa z poziomu widoku głównego. Poszczególne widoki mogą wyświetlać różne zakresy dat oraz rożne plany.

### 1.1.3 Kalendarz

Obszar planu przedstawia siatkę dni z podziałem na jednostki czasowe. Zakres wyświetlany na planie jest zmienny i jest ustawiany za pomocą kalendarza. Na siatce godzin i dni można umieszczać agregaty z zasobnika agregatów . Możliwe jest też przemieszczanie ich w zakresie danego planu oraz poza nim (inne widoki).

### 1.1.4 Obszar menu

Obszar menu zajmuje górny wiersz widoku, jego zadaniem jest do różnego typu ustawień związanych z ustawieniami programu, uzyskiwaniu o nim informacji oraz jego obsługą. Można w nim między innymi przełączać widoki czy otwierać nowe.

### 1.1.5 Alerty

Ta kolumna służy do komunikacji. Zawiera wiadomości od innych użytkowników jak i listę alertów. Alerty i wiadomości można filtrować po słowach kluczowych (np. fragmencie treści, dniu, nadawcy). Kolumna alertów zawiera informacje o konfliktach, niespełnionych preferencjach prowadzącego, niespełnione warunki sali, wakaty. Kliknięcie w konkretny alert przenosi użytkownika do okna planu. Kolumna wiadomości zawiera tylko wiadomości nieprzeczytane.

### 1.1.6 Zasobnik agregatów

Zasobnik agregatów służy do przechowywania stworzonych agregatów. Wyświetlane są tylko agregaty które nie zostały już wykorzystane (umieszczone na planie). Użytkownik ma możliwość wyciągania agregatów z zasobnika na plan poprzez przeciągania go za pomocą myszy. Zasobnik jest wyposażony w kategorie grupujące poszczególne agregaty oraz funkcje wyszukiwania pozwalającą na znalezienie poszukiwanego agregatu za pomocą nazwy.

### 1.1.7 Agregat

Agregat jest kolorowym prostokątem reprezentującym pojedyncze zajęcie(wykład, laboratorium lub projekt)dla danej grupy studenckiej, do którego przypisane są lub dopiero będą dodatkowe informację. Agregaty można przeciągać pomiędzy widokami oraz umieszczać na planie.

## 1.2 Opis szczegółowy

### 1.2.1 Ekran logowania

W czasie logowania użytkownik może zalogować się na jedną z kilku grup użytkowników mających różne uprawnienia oraz zakres dostępu.

#### Użytkownik – planista

Główny użytkownik system ma do dyspozycji dostęp do formularzy tworzenia agregatów oraz ich edycji. Widoczne są dla niego wszystkie widoki oraz ma możliwość ich edycji. Posiada możliwość odbierania i wyświetlania wiadomości oraz alertów. Ponadto może udostępniać gotowe (lub w budowie) plany zajęć innym użytkownikom

#### Użytkownik – student

System oferuje studentowi możliwość przeglądania kalendarzy pod kątem zbierania informacji:

* gdzie ma zajęcia
* kto prowadzi zajęcia
* kiedy i gdzie można spotkać konkretnego wykładowcę

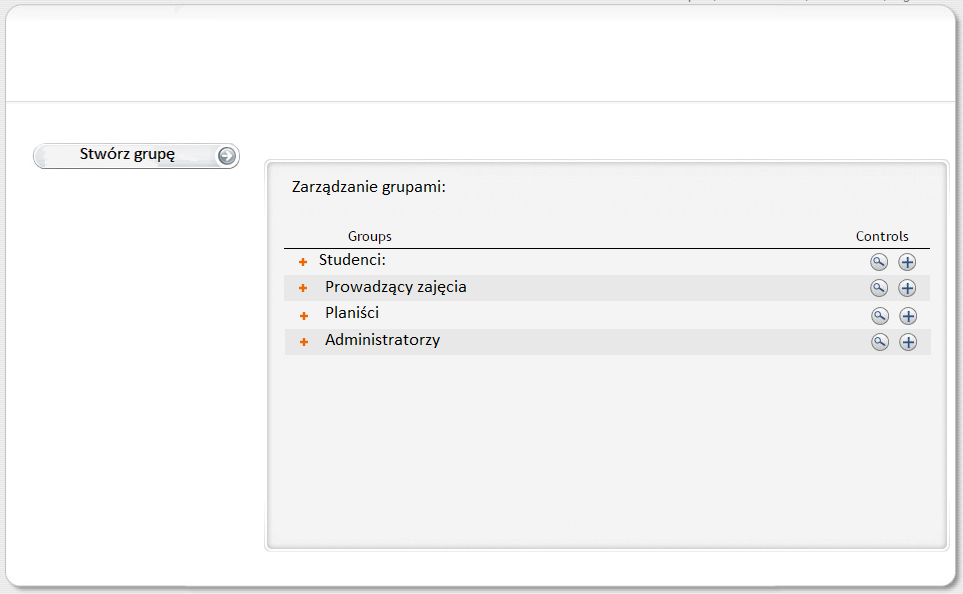
Te funkcje są realizowane na zasadzie: użytkownik wpisuje nazwisko wykładowcy do okienka wyszukiwania, wybiera odpowiednią osobę ze zwróconego wyniku i po kliknięciu na nią otwiera mu się kalendarz który może przeglądać.

#### Użytkownik – wykładowca

Wykładowca ma możliwość przeglądania udostępnionego mu planu zajęć oraz zgłaszania uwag poprzez system wiadomości.

#### Użytkownik – administrator

Administrator ma dostępy widok zarządzania grupami użytkowników, podglądania i edycji ich danych.



Rysunek : Zarzadzanie grupami

### 1.2.2 Widoki

Planista ma do wyboru cztery rodzaje widoków:

-widok główny

- widok grupy studenckiej

- widok wykładowcy

- widok sali

Za pomocą tych widoków planista może rozmieszczać elementy z dostępnego zasobnika na planie. W ten sposób „sczepiając” razem salę + wykładowcę + grupę studencką + przedmiot.

Jak wspomniano planista może otworzyć jeden z 4 widoków, w ten sposób np. otwierając widok wykładowcy łączy go z pozostałymi elementami: salą, grupą studencką, przedmiotem.

Widoki te nie różnią się zasadniczo w wyglądzie – istotny czynniki to z jakiej perspektywy jest przedstawiony kalendarz.

I tak gdy planista otworzy widok wykładowcy ukaże się mu poniższy obraz który składa się z 2 istotnych elementów: Kalendarza oraz zasobnika agregatów

#### 

Rysunek : Przykładowy widok wykładowcy

#### Widok wykładowcy

Widok wybranego wykładowcy, pokazuje plan dla danego wykładowcy wraz z agregatami które zostały do niego dołączone. Agregaty które zostały już połączone z danym dniem oraz godziną zostają wyświetlone na planie zajęć. Tymczasem agregaty niezagospodarowane ale przyłączone do danego prowadzącego przedstawione są w zasobniku agregatów.

Widok ten jest dostępny dla planisty oraz dla wykładowcy którego dotyczy. Planista ma pełny dostęp, podczas gdy wykładowca nie widzi zasobnika agregatów.

#### Widok grupy studenckiej

Widok wybranej grupy studenckiej. Tak jak w pozostałych widokach zawiera agregaty połączone z daną grupą studencką. Agregaty mogą zawierać różne ilości połączonych elementów: przedmiot + grupa studencka, przedmiot + grupa studencka + wykładowca, przedmiot + grupa studencka + wykładowca + sala.

Dostęp do tego widoku ma planista z pełnymi prawami oraz uczniowie należący do danej grupy studenckiej (bez zasobnika agregatów).

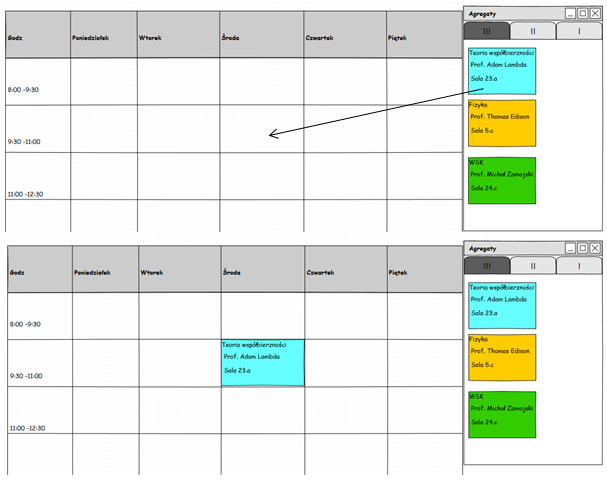
#### Widok Sali

Widok wybranej sali lekcyjnej pokazuje agregaty połączone z tą salą, które mogą być połączeniem są połączeniem kilku elementów: przedmiot + grupa studencka +sala , przedmiot + grupa studencka + wykładowca + sala.

Widok Sali jest dostępny tylko dla planisty.

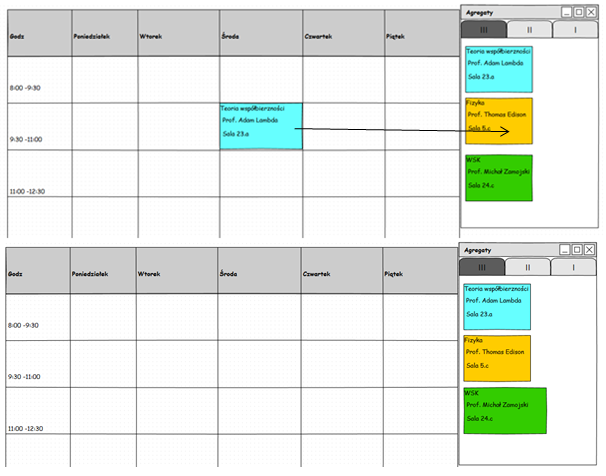
### 1.2.4 Agregat

Planista może dodawać nowe agregaty lub kolejne instancje tego samego typu agregatu do widoku przeciągając je z zasobnika na siatkę. Agregaty których ilość instancji osiągnie zamierzoną ilość godzinową zostaną usunięte z zasobnika.



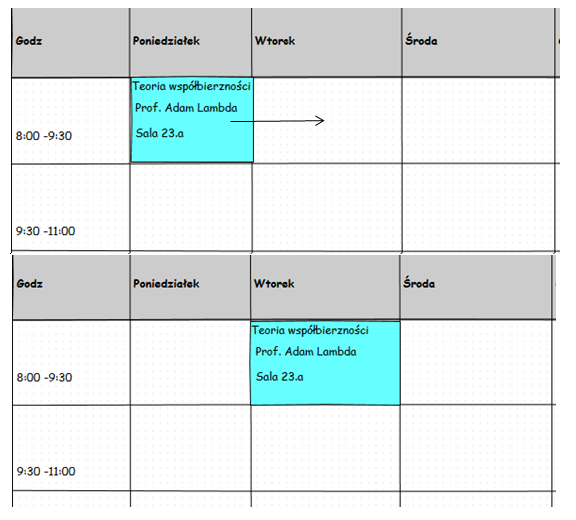
Rysunek 3: Ustawianie agregatu na siatce

Planista może usuwać agregaty z widoku przeciągając je z siatki do zasobnika. Agregaty które nie są już dostępne w zasobniku z powodu osiągnięcia zamierzonej liczby godzin zostaną ponownie do niego dodane

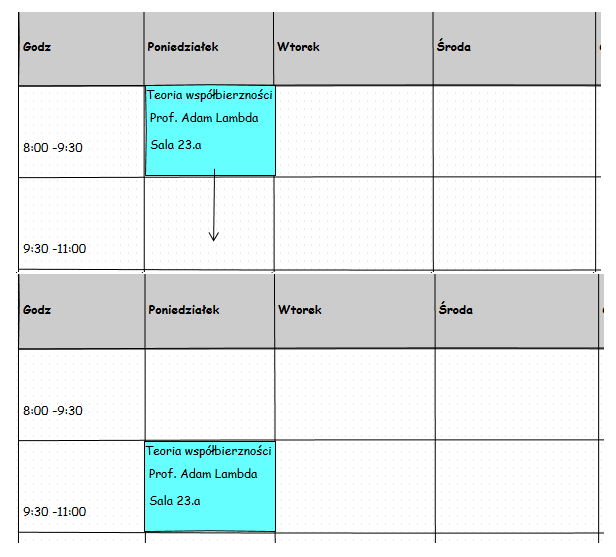


Rysunek 4: Usuwanie agregatu z siatki

Planista w ramach tego samego widoku będzie mógł przesuwać agregat w pionie (zmiana godziny) jak i poziomie(zmiana dnia) przeciągając agregat z jednego oczka siatki na drugi.



Rysunek 5 : Przesuwanie w poziomie



Rysunek 6: Przesuwanie w pionie

Przeciągnięcie jednego agregatu na drugi sprawi że te dwa agregaty zamienią się miejscami w przypadku gdy agregat jest przeciągany z innego oczka siatki. Drugą możliwością jest przeciągniecie elementu z zasobnika, wtedy przeciągnięty element zajmie miejsce poprzedniego agregatu podczas gdy ten zostanie wrócony do zasobnika.

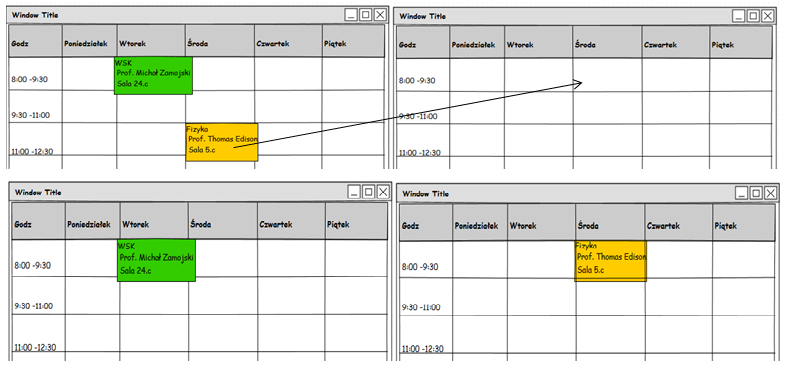


Rysunek 7: Zamiana agregatów miejscami



Rysunek 8: Zamiana przy użyciu agregata z zasobnika

Możliwe będzie także przeciąganie agregatów pomiędzy oknami wyświetlającymi widoki różnych siatek, spowoduje to taką samą reakcje jak przy przeciąganiu na tej samej siatce. Dodatkowo przy przeciąganiu pomiędzy widokami różnych typów np. z prowadzącego na grupę, zmienione zostaną informacje wyświetlane na agregacie.



Rysunek 9: Przeciąganie pomiędzy różnymi oknami

## 1.3 Funkcję dostępne dla użytkownika

### 1.3.1 Wykonywanie operacji na wykładowcach

Planista ma możliwość wykonywania operacji na liście wykładowców umieszczonych w systemie. Możliwe operacje do przeprowadzenia na wykładowcach:

•wyświetlenie listy wykładowców

• stworzenie nowego wykładowcy

• zmiana danych wykładowcy

• usunięcie wykładowcy

### 1.3.2 Wykonywanie operacji na awatarach

Planista ma możliwość wykonywania operacji na liście awatarów umieszczonych w systemie. Możliwe operacje do przeprowadzenia na awatarach:

•wyświetlenie listy awatarów

• stworzenie nowego awatara

• dodanie wykładowcy do awatara

• usunięcie wykładowcy z awatara

• usunięcie awatara

### 1.3.3 Wykonywanie operacji na grupach studenckich

Planista ma możliwość wykonywania operacji na liście grup studenckich umieszczonych w systemie. Możliwe operacje do przeprowadzenia na grupach studenckich:

•wyświetlenie listy grup studenckich

• stworzenie nowej grupy studenckiej

• usunięcie grupy studenckiej

### 1.3.4 Wykonywanie operacji na liście agregatów

Planista ma możliwość wykonywania operacji na liście agregatów umieszczonych w systemie. Możliwe operacje do przeprowadzenia na grupach studenckich:

• wyświetlenie listy agregatów

• stworzenie nowego agregatu

• edycja danych agregatu

• usunięcie grupy studenckiej

### 1.3.5 Wyświetlanie widoków

Planista ma możliwość wyświetlania różnych widoków kalendarza w różnych konfiguracjach. Możliwe interakcje z widokiem.

• wyświetlenie widoku

• wyświetlenie widoku w nowym oknie

• przełączenie widoku

• zmiana zakresu wyświetlanych dat

### 1.3.6 Wykonywanie operacji na szablonach

Planista ma możliwość wykonywania operacji na szablonach stworzonych w systemie. Możliwe operacje do przeprowadzenia na szablonach:

• stworzenie szablonu

• edycja zakresu szablonu

• usunięcie szablonu

### 1.3.7 Wykonywanie operacji na skrzynce widomości i alertów

Planista ma możliwość wykonywania operacji na wiadomościach i alertach wyświetlonych w skrzynce. Możliwe operacje do przeprowadzenia:

• przeczytanie wiadomości

• przeczytanie szczegółów alertu

• zignorowanie alertu

# 2. Moduł GWT

## 2.1 Opis ogólny

GWT(Google Web Toolkit) jest to Framework służący do tworzenia aplikacji webowych w oparciu o język Java. Biblioteka pozwala tworzyć strony internetowe bez potrzeby dużej znajomości języka HTML, CSS, JavaScript. Google Web Toolkit jest wyposażony w kilka modułów służących do budowy aplikacji w technice AJAX: GTW kompilator, warstwę UI, system RPC, powłokę GTW oraz kilka dodatkowych narzędzi ułatwiających zarządzanie webowym poziomem aplikacji. GWT jest używane przez wiele produktów w Google, włączając Google AdWords i Orkut. Jest open Source, czyli całkowicie za darmo i używane przez tysiące programistów Java na całym świecie. Pierwsza wersja GWT została wypuszczona w maju 2006 roku

Toolkit jest zintegrowany z bliblioteką testową JUnit.

GWT nie jest związane z żadnym konkretnym IDE dlatego można go używać z każdym środowiskiem programistycznym obsługującym Jave.

## 2.2 Budowa GWT

Toolkit składa się z dwóch głównych komponentów:

* GWT SDK - zawiera biblioteki Java API, kompilator, i serwer rozwoju. To pozwala pisać aplikacje po stronie klienta w języku Java i rozmieścić je jako JavaScript.
* Plugin dla Eclipse IDE - zapewnia wsparcie środowiska IDE dla Google Web Toolkit i projekty aplikacji internetowych Engine.

## 2.3 Kompilator GWT

Kompilator frameworku zajmuje się przekształceniem kodu źródłowego napisanego w Javie do języka JavaScript. Z definicji kompilator przekształca zadania w moduły. Kompilator może rozbić kod końcowy w różne przekompilowane artefakty w zależności w jakie przeglądarki jest celowana aplikacja.

W przeciwieństwie do większości toolkitów budujących aplikacje webowe w technice AJAX które mają skomplikowaną logikę potrzebną by współpracować I dostosowywać się do różnych przeglądarek, GWT stosuje metodę podmieniania głównych klas w zależności od przeglądarki która jest celem. Wynikiem takiej polityki jest zmiejszenie ilości kodu który musi pobrać przeglądarka aby wyświetlić aplikacje, ponieważ potrzebny jej jest tylko kod przeznaczony dla danej przeglądarki zamiast dużej klasy zawierającej w sobie kod potrzebny do obsługi każdej możliwej przeglądarki.

## 2.4 GWT UI

Interfejs użytkownika GTW zbudowany na inteligentnym systemie kompilacji jest multi-przeglądarkowy. Elementu UI są implementowane w Javie a następnie używana jest przygotowana pod daną przeglądarkę implementacja obiektowego modelu dokumentu (DOM), aby zbudować natywne elementy przeglądarki kiedy są potrzebne przez wyższą warstwę Javy.

W przeciwieństwie do innych bibliotek AJAX które skupiają się na gadżetach UI, GWT udostępnia podstawowe elementy UI przy pomocy których użytkownik może budować swoją aplikacje. W skład tych elementów wchodzą różnego rodzaju panele dotyczące układu, modele przedstawiające dane, grafiki, elementy przyjmujące dane od użytkownika oraz inne.

## 2.5 RPC

Zdalne wywołanie procedury (RPC) jest jednym z głównych aspektów które korzystają z inteligentnego kompilatora. System ten pozwala na serializacje oraz de- serializacje obiektów Javy z usług ze strony serwera które mogą być wywoływane asynchronicznie przez klienta.

Kompilator generuje kod podczas kompilacji aby obsłużyć serializacje na niskim poziomie. Obiekty podlegające serializacji są mapowane oraz zostaje im przypisana wersja podczas kompilacji. Uzyskiwane są w ten sposób dwie korzyści: można zagwarantować że serwer i klient porozumieją się kiedy pojawi się mowa wersja oraz implementacja serwera może skompresować stan obiektów Java do prymitywów w JavaScript, co zwiększa prostotę i efektywność przesyłania danych.

## 2.6 Dodatkowe narzędzia

Poza głównymi modułami składającymi się na GTW, wyposażony też jest w grupę narzędzi służących do ułatwienia budowania aplikacji na poziomie webowym. W ich skład wchodzą narzędzia dodające możliwość innej komunikacji z serwerem, wspomagające umiędzynarodowienie, oraz testowanie.

Biblioteki zawarte w GTW pozwalają na porzucenie systemu RPC i porozumiewanie się z serwerem przy pomocy XML oraz usług bazujących na JSON. Pozwala to na zintegrowanie aplikacji z wieloma istniejącymi webowymi API lub nie opierających się na Javie backend-em.

Biblioteka działająca na poziomie kompilacji pozwala w prosty sposób na wspieranie wielojęzykowych aplikacji.

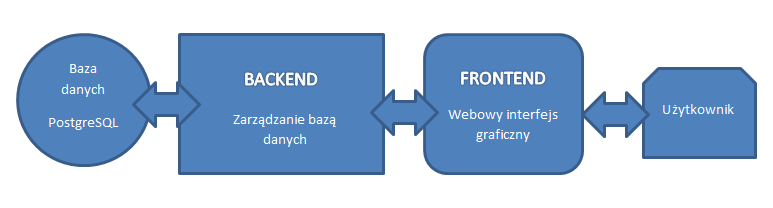
## 2.7 GWT Development Mode

GWT udostępnia opcję nazwaną Development Mode. Pozwala ona na wystartowanie aplikacji webowej na komputerze programisty przez wirtualną maszynę Javy. W tym trybie każda zmiana w kodzie jest natychmiast implementowana do włączonej aplikacji co ułatwia debugowanie programu.

# 3. Architektura systemu

Aplikacja składa się z dwóch głównych modułów odpowiadających za obsługę użytkownika oraz obsługę bazy danych.

* Moduł backend - odpowiada za obsługę bazy danych i komunikację z front endem, umieszczona została tu logika biznesowa
* Moduł frontend – interfejs webowy dostępny przez przeglądarkę, odpowiada za komunikację z użytkownikiem

System jest zbudowany zgodnie z modelem MVC(Model - View - Controller) który zakłada podział aplikacji na trzy Moduły: model zawierający logikę biznesową, widok odpowiedzialny za oraz wyświetlania danych, sterownik który ma zapewnić interfejs do komunikacji z użytkownikiem. Moduły do komunikacji używają interfejsu RPC Pozwala on na prostą oraz szybką komunikację pomiędzy częścią kliencką a serwerem przy pomocy obirktów POJO. Poniżej przedstawiamy ogólny schemat systemu: 

## 3.1 Przepływ danych

Przepływ danych pomiędzy modułami i bazą danych wygląda następująco:

1. Informacja o wybranej akcji trafia do kontrolera menu który w razie wybrania operacji dodania wyświetla odpowiedni formularz w celu zebrania danych.

2. Wysyłane jest asynchroniczne zapytanie do serwera

3. Servlet po stronie serwerowej tworzy obiekt Teacherentity i przekazuje go do kontrolera wykładowców

4. Kontroler przesyła żądanie do odpowiedniej usługi

5. Usługa wywołuje odpowiednią metodę z warstwy DAO

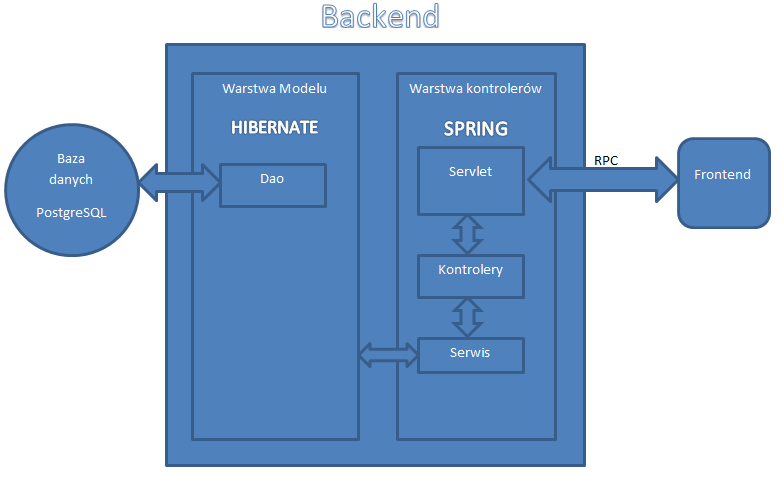
6. Warstwa Dao operuje na bazie danych

7. Generowany jest widok dla użytkownika

## 3.2 Moduł backend

Moduł serwerowy składa z logiki biznesowej napisanej w języku Java zbudowanej przy użyciu frameworku hibernate. Drugą częścią modułu jest warstwa kontrolerów zbudowana zgodnie z frameworkiem Spring. Dane użytkowane przez system są przechowywane w relacyjnej bazie danych PostgreSQL.

Sposób działania modułu backend wygląda następująco: Od strony frontendu przesyłane jest zapytanie klienckie typu RPC (Remote Procedure Call*)*. Dane przesyłane są za pomocą wywołania asynchronicznego. Zapytanie trafia do warstwy kontrolerów(Spring), gdzie jest kierowana do konkretnego kontrolera przez servlet. Kontroler przekazuje następnie odpowiednią akcję do serwisu, który komunikuje się z warstwą modelu (hibernate), gdzie przy pomocy komponentu dao wykonywane są odpowiednie operacje na bazie. Ostatecznie dane powrotne są odpowiednio opakowywane przez serwer i odsyłane do frontendu. Poniżej został przedstawiony model backendu i jego sposobu komunikacji.



### 3.2.1 Warstwa kontrolerów

Warstwa kontrolerów oparta na frameworku Spring odpowiedzialna jest za reagowanie na przychodzące zapytania od klienta i odsyłanie dpowiednich danych do klienta. Głównymi możliwościami oferowanymi przez framework Spring jakie są używane w aplikacji są: Spring bean, są to dowolne obiekty produkowane przez fabrykę Springa i zarządzane przez kontener Springa. Spring Context, który pozwala na połączenie ze sobą beansów aby mogły pracować razem. Dependency injeciton (wstrzykiwanie zależności) która oferuje możliwość separacji interfejsów, modularyzacji aplikacji oraz możliwość pokrycia funkcjonalności naprawdę dużą porcją testów jednostkowych.

### 3.2.2 Warstwa Modelu

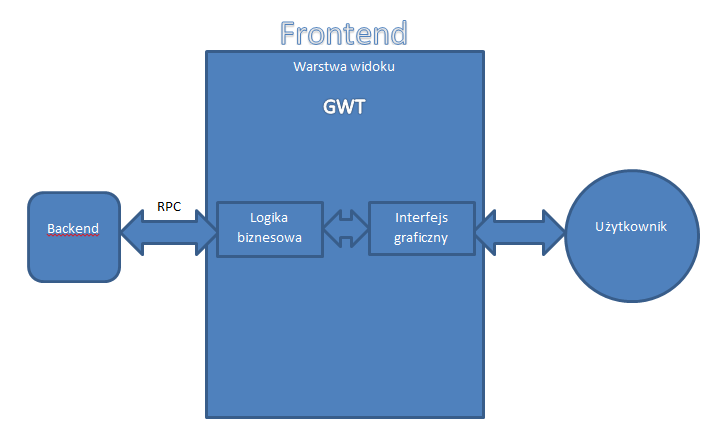
Warstwa modelu odpowiedzialna będzie za wydobycie z bazy danych odpowiedniego

zasobu, wykonaniu na tych danych określonych operacji i zwrócenie ich do kontrolera. W tej warstwie działa framework Hibernate który jest odpowiedzialny za mapowanie bazy relacyjnej na obiekty oraz komponent DAO(Data Access Object) zapewniający jednolity interfejs służący do komunikacją pomiędzy aplikacją a źródłem danych.

## 3.3 Moduł frontend

Moduł od strony klienckiej jest interfejsem webowym dostępny przez przeglądarkę przeznaczonym dla użytkownika końcowego. Podstawowymi jego zadaniami są: zapewnieni interfejsu graficznego pobieranie danych od użytkowników, reagowanie na działania użytkownika oraz wyświetlanie danych otrzymanych od serwera.

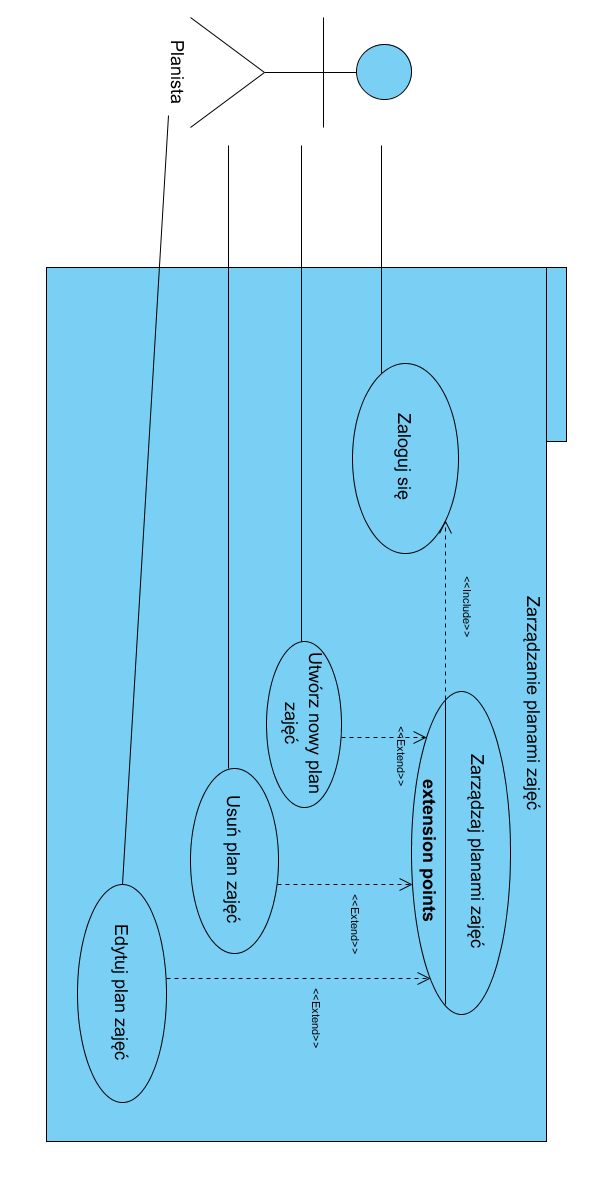
Moduł ten został stworzony przy pomocy biblioteki SDK gwt, który pozwala na stworzenie w pełni funkcjonalnej aplikacji internetowej używając przy jej tworzeniu języka Java, którego kod jest następnie kompilowany do języka Javascript oraz html. Możliwe jest dodawanie komponentów html oraz uatrakcyjnianie wyglądu za pomocą technologii CSS. Zarys modułu:



# 

# 4. Diagramy przypadków użycia

## 4.1. Zarządzanie planami zajęć



### Zarządzanie planami zajęć

LOGOWANIE:

Aktorzy: Planista

Wyzwalacz: Planista naciska na przycisk „Zaloguj się”

Warunek początkowy: Planista nie jest zalogowany

Przepływ podstawowy:

1. Wprowadzanie adresu e-mail i hasła

2. Komunikat o udanym logowaniu

3. Udostępnienie interfejsu użytkownika

Przepływ alternatywny:

1.1. Błędne dane logowania

1.2. Komunikat o błędzie

1.2.1.1. Ponowne logowanie

TWORZENIE NOWEGO PLANU ZAJĘĆ:

Aktorzy: Planista

Wyzwalacz: Planista naciska na przycisk „Stwórz nowy plan zajęć”

Warunek początkowy: Planista jest zalogowany

Przepływ podstawowy

1. Wybranie nazwy oraz okresu obowiązywania planu.

2. Planista przypisuję odpowiednią grupę studencką.

3. Planista klika „Zapisz” w celu zapisania planu do którego potem będzie mógł przypisywać

terminy z zajęciami.

Przepływ alternatywny:

1.1 Błędna nazwa pliku

1.2 Błędne dane- komunikat o błędzie

EDYCJA PLANU ZAJĘĆ:

Aktorzy: Planista

Wyzwalacz: Planista naciska na przycisk „Edytuj plan zajęć”

Warunek początkowy: Planista jest zalogowany

Przepływ podstawowy

1. Wybranie planu do edycji.

2. Zatwierdzenie zmian.

Przepływ alternatywny:

1.1 Wybranie planu którego nie można edytować

1.2 Wprowadzenie błędnych danych- komunikat o błędzie

USUWANIE PLANU ZAJĘĆ:

Aktorzy: Planista

Wyzwalacz: Planista naciska na przycisk „Usuń plan zajęć”

Warunek początkowy: Planista jest zalogowany

Przepływ podstawowy:

1. Wybranie planu do usunięcia

2. Zatwierdzenie usunięcia

Przepływ alternatywny:

1.1 Wybranie planu który już nie istnieje(jest usunięty a strona jest nie odświeżona).

1.2 Komunikat o błędzie.

1.2 Zarządzanie przedmiotami

WYBIERANIE PLANU ZAJĘĆ:

Aktorzy: Planista

Wyzwalacz: Planista naciska na przycisk „Wybierz plan zajęć”

Warunek początkowy: Planista jest zalogowany

Przepływ podstawowy:

1. Wybranie planu do którego będą dodawane przedmioty.

EDYTOWANIE PRZEDMIOTU:

Aktorzy: Planista

Wyzwalacz: Planista naciska na przycisk „Wybierz plan zajęć”

Warunek początkowy: Planista jest w oknie edycji danego planu zajęć oraz zalogowany.

Przepływ podstawowy:

1. Wybranie przedmiotu

2. Kliknięcie „Edytuj przedmiot”

USUWANIE PRZEDMIOTU:

Aktorzy: Planista

Wyzwalacz: Planista naciska na przycisk „Wybierz plan zajęć”

Warunek początkowy: Planista jest w oknie edycji danego planu zajęć oraz zalogowany.

Przepływ podstawowy:

1. Wybranie przedmiotu

2. Kliknięcie „usuń przedmiot”

Przepływ alternatywny:

Brak możliwości usunięcia przedmiotu- komunikat o błędzie

DODAWANIE NOWEGO PRZEDMIOTU:

Aktorzy: Planista

Wyzwalacz: Planista naciska na przycisk „Wybierz plan zajęć”

Warunek początkowy: Planista jest w oknie edycji danego planu zajęć oraz zalogowany.

Przepływ podstawowy:

1. Planista klika „Dodaj nowy przedmiot”

2. Uzupełnienie potrzebnych danych przedmiotu

3. Zatwierdzenie

Przepływ alternatywny:

1.2 Błędna nazwa przedmiotu- komunikat o błędzie.

1.3 Zarządzanie grupami

DODAWANIE NOWEJ GRUPY:

Aktorzy: Planista

Wyzwalacz: Planista naciska na przycisk „Dodaj nową grupę”

Warunek początkowy: Planista jest w oknie edycji grup oraz zalogowany.

Przepływ podstawowy:

1. Planista klika „Dodaj nową grupę”

2. Uzupełnienie potrzebnych danych grupy

3. Zatwierdzenie

Przepływ alternatywny:

1.2 Błędna dane grupy np. grupa już może istnieć- komunikat o błędzie.

EDYTOWANIE GRUPY:

Aktorzy: Planista

Wyzwalacz: Planista naciska na przycisk „Edytuj grupy”

Warunek początkowy: Planista jest w oknie edycji grup oraz zalogowany.

Przepływ podstawowy:

1. Wybranie grupy do edycji

2. Kliknięcie „Edytuj grupę”

3. Zmiana danych grupy

4. Zatwierdzenie

Przepływ alternatywny:

1.4 Błędna dane grupy np. grupa już może istnieć- komunikat o błędzie.

USUWANIE GRUPY:

Aktorzy: Planista

Wyzwalacz: Planista naciska na przycisk „Usuń grupę”

Warunek początkowy: Planista jest w oknie edycji grup oraz zalogowany.

Przepływ podstawowy:

1. Wybranie grupy do usunięcia

2. Kliknięcie „usuń grupę”

Przepływ alternatywny:

1.2 Brak możliwości usunięcia grupy- komunikat o błędzie

1.4 Tworzenie zajęcia

TWORZENIE ZAJĘCIA:

Aktorzy: Planista

Wyzwalacz: Planista naciska na przycisk „Stwórz zajęcie”

Warunek początkowy: Planista jest w oknie edycji danego planu zajęć oraz zalogowany.

Przepływ podstawowy:

1. Planista wybiera prowadzącego

2. Wybranie przedmiotu

3. Wybranie grupy studenckiej

4. Wybranie Sali

5. Wybranie terminu

6. Kliknięcie „Stwórz”

Przepływ alternatywny:

1.1 Prowadzący ma w tym czasie inne zajęcia

1.4 Sala jest zajęta

1.6 Komunikaty o ewentualnych błędach.1.5 Tworzenie terminu

TWORZENIE TERMINU:

Aktorzy: Planista

Wyzwalacz: Planista naciska na przycisk „Stwórz termin”

Warunek początkowy: Planista jest w oknie zarządzania terminami oraz zalogowany

Przepływ podstawowy:

1. Planista wybiera zestaw dni dla których chce stworzyć termin

2. Planista wybiera godziny dla danego terminu

3. Wybranie Sali

4. Zatwierdzenie poprzez kliknięcia na „Zatwierdź”

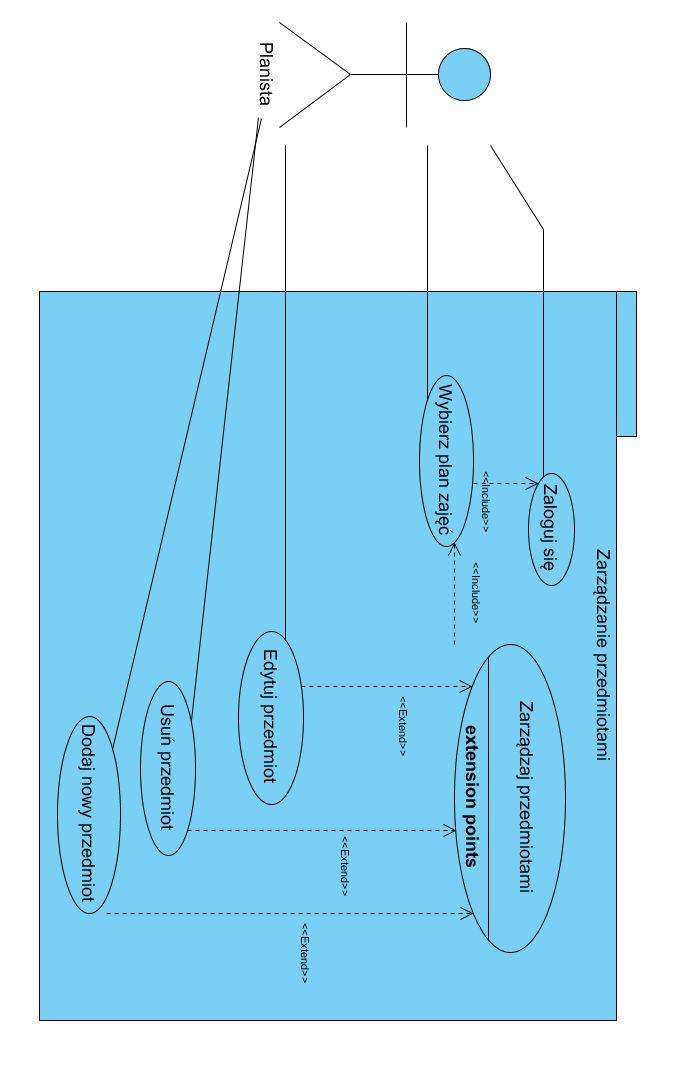
Przepływ alternatywny:

1.1 Termin jest niemożliwy do wybrania – komunikat o błędzie

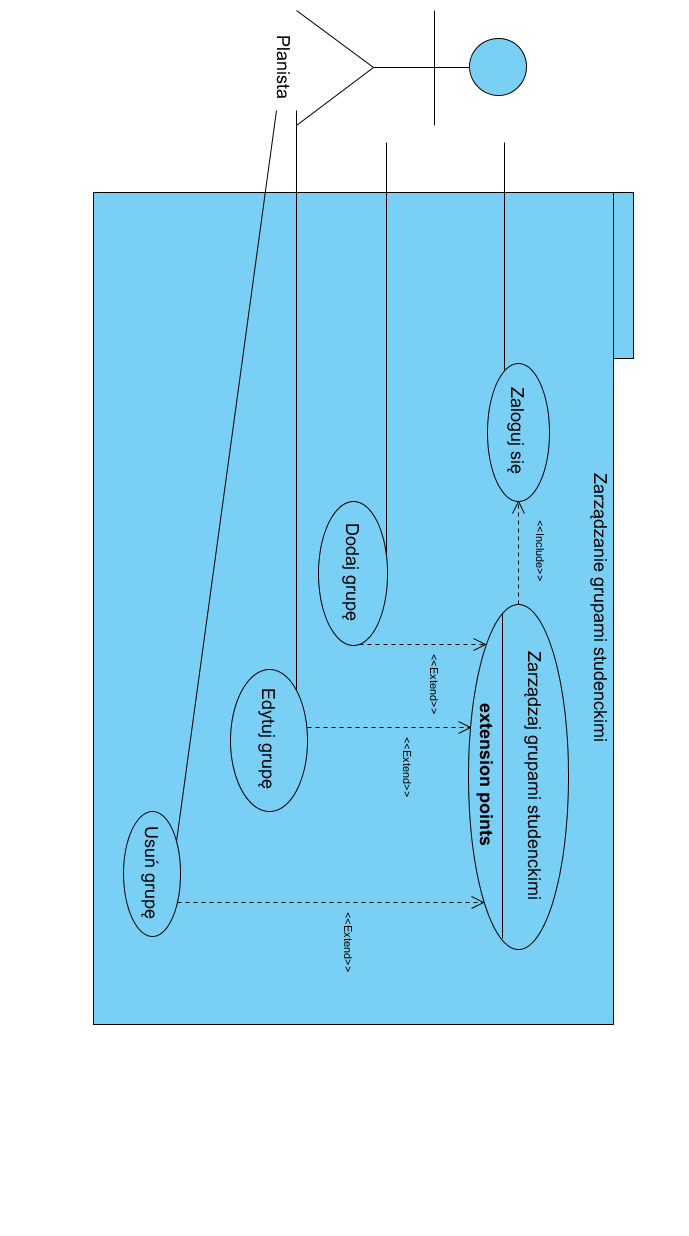
1.2 Godziny są już zajęte – komunikat o błędzie

1.3 Sala jest już zarezerwowana – komunikat o błędzie

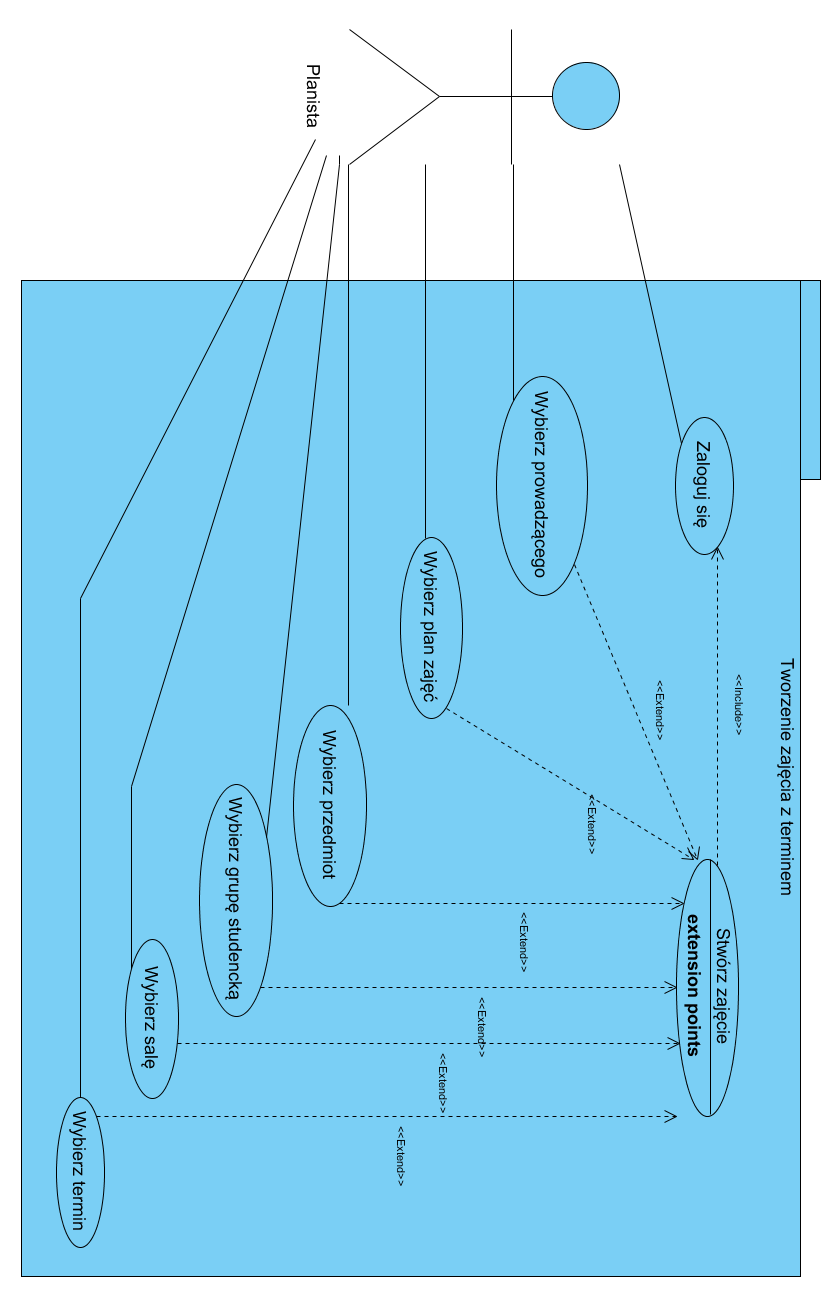
## 4.2 Zarządzanie przedmiotami



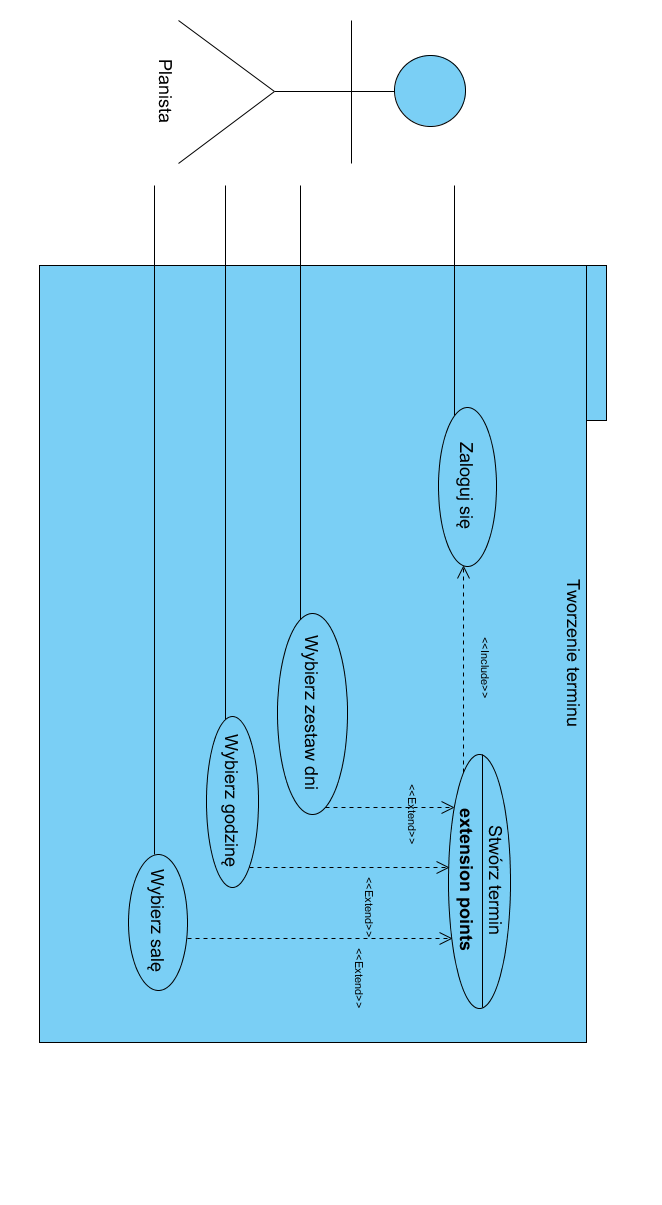
## 4.3 Zarządzanie grupami studenckimi



## 4.4 Tworzenie zajęcia wraz z terminem



## 4.5. Tworzenie terminu



Scenariusze systemowych przypadków użycia