**AKADEMIA GÓRNICZO-HUTNICZA**

**IM. STANISŁAWA STASZICA W KRAKOWIE**



**Wydział Informatyki, Elektroniki i Telekomunikacji**

Katedra Informatyki

**Przewodnik po projekcie**

**Interaktywny system wspomagania układania rozkładu zajęć**

Roger Barlik, Krzysztof Nowakowski,

Krystian Ujma, Krzysztof Wróbel

Promotor: Mgr inż. Witold Rakoczy

OŚWIADCZENIE AUTORÓW PRACY

Oświadczamy, świadomi odpowiedzialności karnej za poświadczenie nieprawdy, że niniejszą pracę dyplomową wykonaliśmy osobiście i samodzielnie (w zakresie wyszczególnionym we wstępie) i że nie korzystaliśmy ze źródeł innych niż wymienione w pracy.

Spis treści

[1. Wstęp 5](#_Toc424252992)

[2. Cel i wizja projektu 5](#_Toc424252993)

[2.1. Charakterystyka problemu 5](#_Toc424252994)

[2.2. Wizja produktu 6](#_Toc424252995)

[2.3. Analiza zagrożeń 6](#_Toc424252996)

[3. Zakres funkcjonalności 7](#_Toc424252997)

[3.1. Kontekst użytkownika 7](#_Toc424252998)

[3.2. Wymagania funkcjonalne 7](#_Toc424252999)

[3.3. Wymagania niefunkcjonalne 7](#_Toc424253000)

[4. Wybrane aspekty realizacji 8](#_Toc424253001)

[4.1. Architektura systemu 8](#_Toc424253002)

[4.2. Moduł backend 8](#_Toc424253003)

[4.2.1. Moduł Spring 9](#_Toc424253004)

[4.2.2. Moduł Hibernate 9](#_Toc424253005)

[4.2.3. Moduł frontend 9](#_Toc424253006)

[4.3. Baza Danych 10](#_Toc424253007)

[5. Organizacja pracy 12](#_Toc424253008)

[5.1. Kontrola wersji, komunikacja 12](#_Toc424253009)

[5.2. Skrócony plan pracy 12](#_Toc424253010)

[5.3. Podział prac 13](#_Toc424253011)

[6. Narzędzia używane podczas tworzenia projektu 13](#_Toc424253012)

[7. Wynik projektu 14](#_Toc424253013)

[7.1. Powstała dokumentacja 15](#_Toc424253014)

[8. Testy 15](#_Toc424253015)

[9. Propozycja dalszych prac 15](#_Toc424253016)

[9.1. Raportowanie błędnych akcji 15](#_Toc424253017)

[9.2. Wczytywanie danych 15](#_Toc424253018)

[9.3. Ładowanie danych z pliku 15](#_Toc424253019)

[9.4. Widoki szczegółowe 16](#_Toc424253020)

[9.5. Skrzynka alertów 16](#_Toc424253021)

[9.6. Dostęp wykładowców 16](#_Toc424253022)

[9.7. Skrzynka wiadomości 16](#_Toc424253023)

[9.8. Dostęp mobilny 16](#_Toc424253024)

[10. Materiały źródłowe 17](#_Toc424253025)

[11. Spis ilustracji 17](#_Toc424253026)

# Wstęp

Celem tego dokumentu jest podsumowanie przebiegu prac na projektem inżynierskim – prowadzonego w roku akademickim 2014/2015 przez okres około 8 miesięcy. Zostały tu jedynie nakreślone pewne informacje związane z budową i funkcjonowaniem systemu, szczegółowe informacje zostały zawarte w dokumentacji technicznej oraz użytkownika

Tworzony projekt miał być samodzielnym, skończonym systemem wspomagania układania planu lekcji, tworzonym w systemie prototypowania. W trakcie prac z powodu dużej objętości projektu, jego zawansowania oraz wystąpienia zagrożenia niespełnienia wszystkich wymagań systemu z powodu ograniczeń technologii, zespół zmuszony był na zakończenie projektu zawansowanym systemem dowodzącym słuszności założeń projektu i technologii do jego realizacji.

# Cel i wizja projektu

Tematem projektu jest **stworzenie interaktywnego systemu wspomagania układania planu lekcji**. System jest przeznaczony do uruchomiania na serwerze w sieci osiągalnym dla użytkowników. Udostępniane jest platforma do tworzenia, przechowywania, przeglądania oraz edytowania planów zajęć.

## Charakterystyka problemu

Problemem jakiemu jest poświęcona praca to zagadnienie wspomagania układania planu zajęć dla Wydziału Informatyki, Elektroniki i Telekomunikacji. Osoba tworząca plany zajęć na uczelni, w czasie tworzenia nowego planu, opiera się na już istniejącym rozkładzie zajęć, pochodzącym z poprzedniego roku. Poprzedni plan jest modyfikowany w celu wprowadzenia zmian uwzględniających nowe warunki.

Na dzień dzisiejszy poziom wspomagania układania planu zajęć przez systemy komputerowe jest niski, nie jest używane żadne rozwiązanie planistyczne, czy to aplikacja półkowa, czy system dedykowany. Praca nad planem opiera się na arkuszu kalkulacyjnym oraz notatkach w których planista przechowuje preferencje oraz ograniczenia dotyczące rozplanowania elementów.

Elementami jakie wchodzą w skład planowania planu są: terminy, przedmioty, grupy studenckie, wykładowcy oraz sale (wykładowe, laboratoryjne i inne). Duża złożoność problemu, ilość ograniczeń i preferencji jakie trzeba wziąć pod uwagę sprawia że dotychczasowe, niemal ręczne podejście do zagadnienia jest nieefektywne.

## Wizja produktu

Finalnym produktem powinien być system umieszczony na serwerze, pozwalający użytkownikowi na tworzenie planu zajęć i wykonywaniu na nim operacji.

Planista powinien mieć jak najłatwiejszy i najszerszy podgląd obecnego stanu planu, być może na różnych płaszczyznach (grupa studencka, wykładowca, sala), dzięki czemu będzie on mógł z łatwością kojarzyć elementy planu ze sobą i umiejscawiać je w konkretnych terminach

Równolegle planista potrzebuje mieć dostęp do informacji opisującej stopień wypełnienia planu (np. lista przedmiotów bez przydzielonego prowadzącego). Bardzo istotnym elementem jest rozwiązanie problemu braku wiedzy nt. powstających w czasie układania rozkładu konfliktów, błędów i niewypełnień preferencji wykładowców bądź nałożonych ograniczeń.

W celu ułatwienia układania planu lekcji, gdzie zajęcia przeprowadzane w różne dni są swoim lustrzanym odbiciem, powinny być wprowadzone szablony wiążące te dni ze sobą i przenoszące wszelkie zmiany jednego dnia na wszystkie powiązane tym samym szablonem.

Kolejnym funkcją jest usprawnienie komunikacji, najlepiej poprzez wprowadzenie możliwości śledzenia zmian przez wykładowców i wprowadzania przez nich do systemu uwag o ich preferencjach, a także obecnym stanie planu. Wiąże się to z udostepnieniem dostępu do systemu dla wykładowców

System powinien oferować wygodny interfejs dostępny dla użytkownika końcowego, biorący pod uwagę jego dotychczasowy dobór narzędzi oraz preferencje.

System powinien być przygotowany na rozszerzenia związane z dostępem dla studentów do przeznaczonych dla nich planów przy pomocy urządzeń mobilnych.

## Analiza zagrożeń

Wykonywalność założonego projektu jest obarczone ryzykiem. Głównymi zidentyfikowanymi czynnikami z którymi związane jest ryzyko są: nieznajomość obranej technologii oraz ograniczony czas na wykonanie projektu.

Nieznajomość obranej technologii generuje ryzyko opóźnień poszczególnych etapów związanych z potrzebą douczania się technologii w czasie trwania projektu, istnieje też ryzyko braku możliwości wykonania pewnych elementów aplikacji w danej technologii, związane jest z nim też ryzyko niespełnienia wszystkich wymagań związanych z produktem z powodu ograniczeń technologii

Ograniczenie czasowe wiąże się z ryzykiem nie dokończenia aplikacji na wyznaczony termin. System wspomagania pracy planisty jest złożonym systemem, w którym zachodzi potrzeba uwzględnienia szczegółowych wymagań klienta.

# Zakres funkcjonalności

System powinien oferować funkcjonalność dopasowaną do potrzeb oraz wymagań planisty. Rozdział ten przedstawia listę najważniejszych wymagań funkcjonalnych i niefunkcjonalnych.

## Kontekst użytkownika

System przewiduje trzy typy aktorów.

**Planistę** - Celem użytkownika jest umieszczenie w systemie planu zajęć i jego modyfikacja. Użytkownik traktuje system jako narzędzie pracy, dostępne pod określonym adresem IP. System oferuje użytkownikowi możliwość skorzystania z graficznego interfejsu w celu wygodnej pracy.

**Wykładowcę –** Celem tego użytkownika jest dostęp do udostępnionego dla niego planu zajęć i ewentualne zgłaszanie preferencji oraz zastrzeżeń. Użytkownik traktuje system jako usługę dostępną pod określonym adresem IP.

**Administratora** – Celem tego użytkownika jest konfiguracja oraz zarządzanie kontami.

**Współpracujące systemy:** System jest samodzielną aplikacją. Nie wymaga współpracy z innymi systemami.

## Wymagania funkcjonalne

Zgodnie z założeniami projektu planista powinien mieć możliwość:

* tworzenia planu zajęć
* edycji planu zajęć
* przeglądania planu zajęć
* dostępu do systemu poprzez przeglądarkę
* metoda *drag and drop* do obsługi elementów kalendarza
* możliwość równoczesnej pracy na wielu oknach
* tworzenie szablonów dla dni
* dostęp do planów dla wykładowców

## Wymagania niefunkcjonalne

System powinien spełniać następujące wymagania niefunkcjonalne:

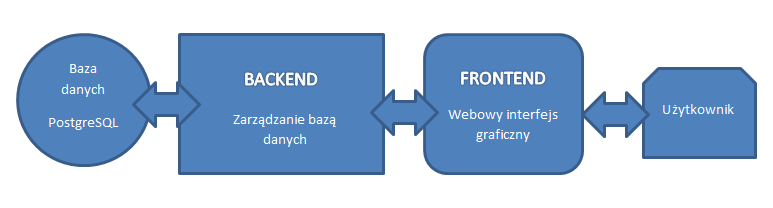
* skalowalny
* intuicyjny i wygodny interfejs graficzny
* niezawodny

# Wybrane aspekty realizacji

## 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, realizowany jest przez moduły Hibernate oraz Spring.
* Moduł frontend – interfejs webowy dostępny przez przeglądarkę, odpowiada za komunikację z użytkownikiem, realizowany jest przez moduł HTML.

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 REST Pozwala on dzięki jednorodnemu interfejsowi na późniejszą rozbudowę systemu o inne moduły oraz prostą oraz szybką komunikację pomiędzy częścią kliencką a serwerem przy pomocy obiektów JSON. Ogólny schemat opisujący budowę systemu pokazany jest na Rysunku 1. 

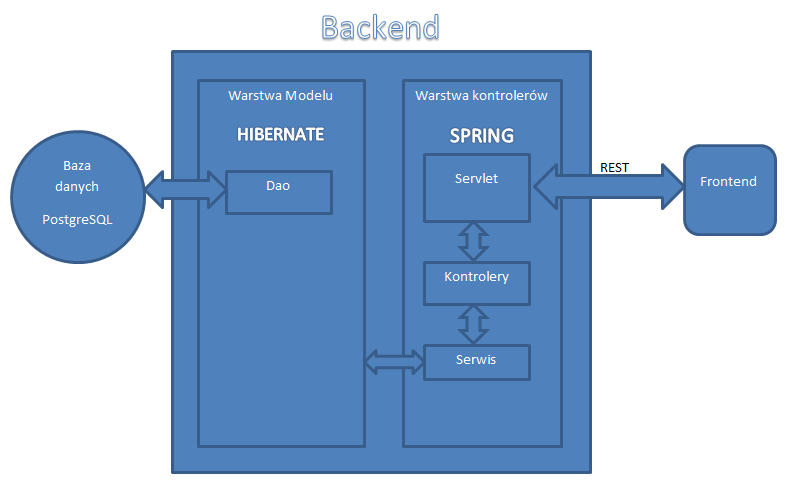
Rysunek 1: Uproszczony diagram budowy systemu

## 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.

Moduł komunikuje się z modułem frontend za pomocą zapytań typu REST obsługiwanych przez moduł Spring, dane są mapowane oraz przekazywane do modułu Hibernate, który komunikuje się z bazą danych. Dane lub potwierdzenie wykonania akcji jest przekazywane z powrotem i opakowywane przez moduł Spring w zapytanie REST

Model modułu backend i jego sposobu komunikacji został przedstawiony na rysunku nr. 2.



Rysunek 2: Uproszczony model modułu backend

## Moduł Spring

Warstwa kontrolerów oparta na frameworku Spring odpowiedzialna jest za reagowanie na przychodzące zapytania od klienta, tworzenie odpowiedzi protokołu HTTP i odsyłanie jej do klienta. Rolę zarządcy pełni *RestContlorer* przekierowujący żądania HTTP do odpowiedniego kontrolera. Dla każdego z rodzaju zasobów używana jest oddzielna klasa kontrolera.

Każdy z kontrolerów posiada będzie odpowiednie metody odpowiedzialne za wykonanie operacji na zestawie danych. W kontrolerach konstruowane są odpowiedzi REST dla frontendu, na podstawie danych dostarczonych przez logikę biznesową.

## Moduł Hibernate

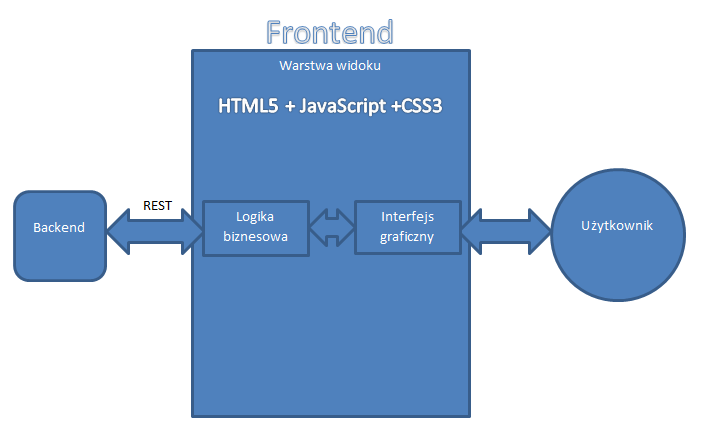
Warstwa modelu odpowiedzialna będzie za wydobycie z bazy danych odpowiedniego zasobu, wykonaniu na tych danych określonych operacji i zwrócenie ich do kontrolera. W tej warstwie działa framework Hibernate który jest odpowiedzialny za mapowanie bazy relacyjnej na obiekty oraz komponent DAO(Data Access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

## 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 standardu HTML5, który pozwala na stworzenie w pełni funkcjonalnej aplikacji internetowej używając przy jej tworzeniu języka HTML oraz języka skryptowego JavaScript. Główną zaletą standardu HTML5 jest obsługiwanie metody *drag and drop* pomiędzy oknami przeglądarki. Za wygląd interfejsu odpowiedzialny jest standard CSS3.

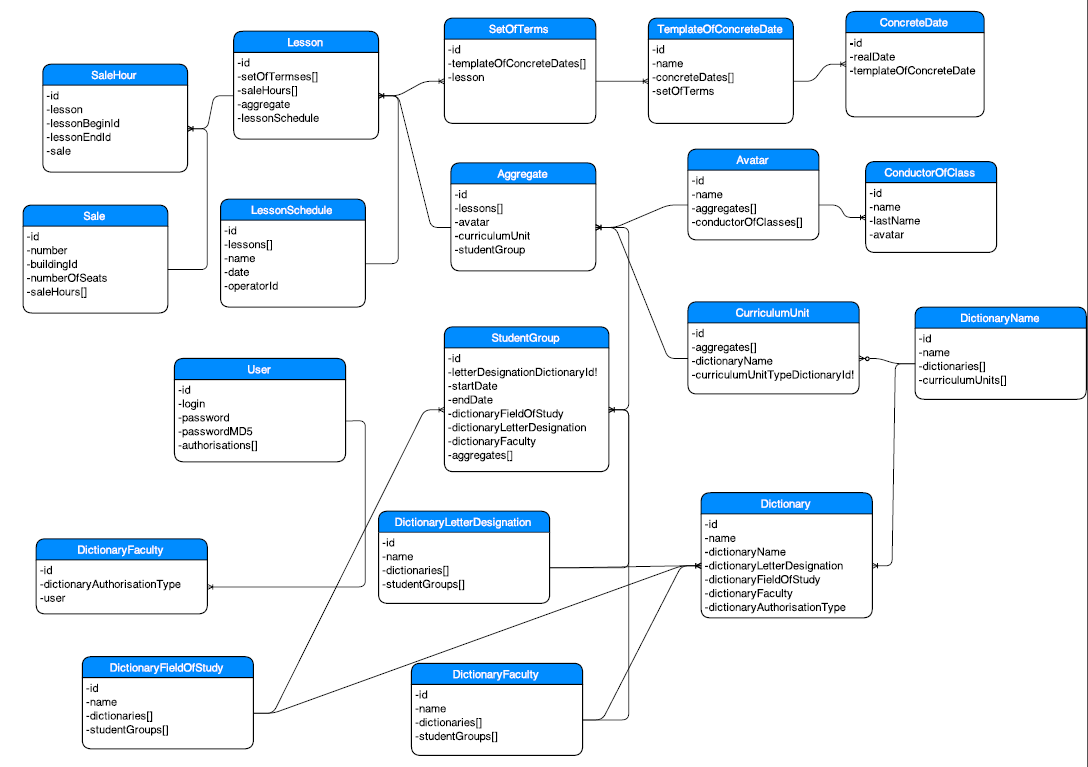
Moduł komunikuje się z częścią serwerową przy pomocy protokołu REST. Model modułu został przedstawiony na rysunku 3.



Rysunek 3: Uproszczony model modułu frontend

## Baza Danych

Baza danych została stworzona przy użyciu systemu bazodanowego PostgreSQL. Stworzone zostały w niej tabele odpowiadające za przechowywania danych na temat poszczególnych elementów na które składa się kalendarz oraz szablonów. Baza danych została zaprojektowania pod wpływem pełnej specyfikacji systemu i zawiera tabele przygotowane pod wszystkie planowane funkcję i może być podstawą do dalszych prac na programem. Układ tabel przedstawiony jest na rysunku 4.



Rysunek 4: Diagram relacyjnej bazy danych

# Organizacja pracy

Prace zostały zaplanowane na okres niecałych dwóch semestrów (8 miesięcy). W czasie tego okresu powinien zostać stworzony gotowy produkt wraz z pełną dokumentacją

Pracę nad projektem została podzielona na dwie fazy:

* Projektowanie bazy oraz interfejsu graficznego
* Właściwa implementacja

System został podzielony na trzy moduły, które zostały rozdzielone pomiędzy członków zespołu. Lista modułów:

* Moduł frontend
* Moduł backend
* Baza danych

## Kontrola wersji, komunikacja

Ze względu na ilość osób w grupie, ich oddalenie od siebie oraz dostępność wybraną formą komunikacji zostały konferencje w programie Skype odbywające się w tygodniowych odstępstwach czasowych.

Zespół ustalił że kod źródłowy zarządzany będzie przez system kontroli wersji git. Wszelkie zmiany w głównym repozytorium będą konsultowane w czasie cotygodniowego spotkania na platformie Skype.

## Skrócony plan pracy

Listopad –grudzień 2014

- Wybranie i sprawdzenie możliwości relacyjnej bazy danych

- Wybranie technologii do implementacji modułu frontend

- Wybranie technologii do implementacji modułu backend

- Zaprojektowanie bazy danych

- Zaprojektowanie wyglądu interfejsu graficznego

- Ustalenie wymagań systemu

Grudzień – styczeń 2014-2015

- Umieszczenie projektu na repozytorium na github.com

- Stworzenie bazy danych

- Rozpoczęcie prac nad modułem backend

Luty – marzec 2015

- Zakończenie prac na modułem backend

- Rozpoczęcie prac nad modułem frontend

Kwiecień – maj 2015

- Zakończenie prac nad modułem frontend

- Testy całości systemu

Czerwiec 2015

- Dostarczenie finalnego produktu

- Dostarczenie finalnej dokumentacji

## Podział prac

Roger Barlik - moduł frontend

* Implementacja kalendarza
* Tworzenie obiektów składowych kalendarza
* Tworzenie dokumentacji projektowej

Krzysztof Nowakowski - moduł frontend

* Wdrożenie mechaniki *drag and drop*
* Konfiguracja komunikacji z modułem backend
* Tworzenie dokumentacji projektowej

Krystian Ujma – moduł backend

* Stworzenie bazy danych
* Implementacja modułu Spring
* Komunikacja wewnątrz modułu backend
* Tworzenie konfiguracji projektowej

Krzysztof Wróbel - moduł backend

* Implementacja modułu Hibernate
* Komunikacja między modułem backend a bazą danych
* Czuwanie nad całością dokumentacji

# Narzędzia używane podczas tworzenia projektu

Podczas tworzenia oraz dokumentacji projektu zespół korzystał z kilku dodatkowych narzędzi do usprawnienia, kontroli oraz ułatwienia tego procesu. Poniżej wymienione zostają narzędzia:

**Github.com** – System kontroli wersji git pozwolił na kontrole przyrostu projektu. Repozytorium umieszczone na github.com pozwalało na szybkie rozpowszechniania kolejnych wersji system, rozdzieleniem pracy pomiędzy członków zespołów, kontrole zmian w projekcie.

**awwapp.com –** Usługa internetowa udostepniająca tablicę do rysowania dostępną dla kilku użytkowników. Usługa okazała się bardzo pomocna w czasie wczesnego projektowania struktury systemu oraz jego wyglądu.

# Wynik projektu

System którego zaprojektowanie oraz stworzenia podjęła się grupa miał duży poziom skomplikowania i wymagał znacznych nakładów pracy.

Do stworzenia systemu potrzebne były trzy moduły stworzone w trzech technologiach HTML5, Hibernate, Spring w trzech językach: Java, Javaspring, Html oraz bazy danych stworzonej w technologii PostreSQL.

Kolejne warstwy złożoności dokładały szczegółowe wymagania i sposób działania systemu jakie przedstawił klient, które musiały być spełnione aby praca nad systemem została zakończona sukcesem. Dane wymagania okazały się nie do spełnienia w większości branych pod uwagę technologii. W połowie pracy nad systemem wybrana technologia GWT, też okazała się niewystarczająca co wymogło dołożenie dodatkowych nakładów pracy.

Ostatecznie zespół jest zadowolony z wyniku prac nad systemem biorąc pod uwagę ograniczony czas, złożoność zagadnienia, napotkane problemy oraz duży nakład pracy jaki został w nie zainwestowany.

System mimo że nie doszedł do fazy prototypu ma przemyślany sposób działania kluczowego elementu produktu jakim jest system *drag and drop*, działający pomiędzy kilkoma oknami przeglądarki, który okazał się nieosiągalny w innych technologiach.

Obecny system funkcjonujący jako zawansowany system dowodzący słuszności założeń projektu prezentuje kluczowe elementy funkcjonalności systemu takie jak: wspominane *drag and drop*, gotowość do działania na serwerze, dostęp przez przeglądarkę, działania na agregatach - w zakresie ich tworzenia, edycji, umieszczania na kalendarzu, oraz funkcjonalność dodatkowa jaką jest zapisywanie ustawionych widoków.

Niestety z powodów opóźnień nie zostały zaimplementowane funkcję: widoki szczegółowe ze względu na każdy podmiot, czy wczytywanie podmiotów agregatów z spreparowanego pliku, tworzenie podmiotów agregatów za pomocą formularzy, tworzenie i zapisywanie szablonów dni. Elementy te mogą być przedmiotem pracy kolejnej grupy która będzie kontynuować rozbudowę tego systemu.

## Powstała dokumentacja

Podczas procesu tworzenia systemu powstała pełna dokumentacja sytemu. Lista powstałych dokumentów:

* **Dokumentacja koncepcyjna**
* **Dokumentacja użytkownika**
* **Dokumentacja procesowa**
* **Dokumentacja techniczna**
* **Przewodnik po projekcie**

# Testy

W czasie tworzenia projektu wszystkie moduły testowane były testami jednostkowymi a także testowana była na różnych stadiach projektu komunikacja pomiędzy poszczególnymi modułami. Wynikowy system był testowany przez członków zespołu pod kontem funkcjonalności od strony użytkownika.

# Propozycja dalszych prac

Umieszczone w tym rozdziale są propozycje dalszych prac mających na celu wzbogacenie funkcjonalności oraz rozszerzenie ilości użytkowników, którzy korzystali by z systemu.

## Raportowanie błędnych akcji

W obecnej wersji systemu przy używaniu funkcji *drag and drop* pomiędzy przeglądarkami jeżeli nad dany obiekt zostanie przeciągnięty obiekt, może z nim wchodzić w interakcję, zostaje wywołana zaprogramowana funkcja i baza jest odpowiednio zmieniana.

W przypadku gdy obiekt zostanie błędnie przeciągnięty nie następuje żadna akcja ponieważ przeglądarki w takim przypadku nie widzą siebie nawzajem. Aby zakomunikować błędną akcję należy rozwiązać ten problem. Może to być osiągnięte za pomocą *web socketów*, po stronie backendu mogą być to *websockety* Java.

## Wczytywanie danych

W obecnej wersji systemu wszystkie dane wczytywane są metodą *eager loading*. Zaimplementowany jest obiekt Dozer, który mapuje encje na obiekty DTO. Ta metoda wczytywania jest wygodna przy małej ilości danych, jednak wraz z ich wzrostem znacząco spowalnia system. Należy zaimplementować wczytywanie danych poprzez metodę *lazy-loading.*

## Ładowanie danych z pliku

W obecnej wersji nie została zaimplementowany mechanizm wczytywania danych niezmieniających się w planie: wykładowców, grup studenckich, przedmiotów z spreparowanego pliku. Wczytywanie z pliku wymaga wypracowania sposobu organizacji pliku oraz odpowiedniego sposobu parsowania pliku.

## Widoki szczegółowe

W systemie nie została zaimplementowana opcja wyboru i otworzenia widoku szczegółowego dla danego podmiotu, np. dla danego wykładowcy widok ze wszystkimi przedmiotami które prowadzi na siatce kalendarza.

## Skrzynka alertów

W systemie nie została stworzona skrzynka wyświetlająca widomości alertowe związane z konfliktami agregatów na kalendarzu

## Dostęp wykładowców

Dostęp dla wykładowców w celu zapoznania się z widokiem szczegółowym dostępnym dla danego wykładowcy

## Skrzynka wiadomości

Udostępnienie wykładowcą komunikacji z planistą w ramach systemu w celu wyrażenia opinii na temat planu zajęć.

## Dostęp mobilny

Dodatkową możliwością rozwoju systemu, może być stworzenie API dla urządzeń mobilnych, z którego będą korzystać studenci w celu uzyskania dostępu do swoich planów zajęć.

# Materiały źródłowe

<http://dev.w3.org/html5/html-author/> - Dokumentacja standardu html5

<http://www.w3.org/TR/#tr_CSS> – Dokumentacja standardu CSS3

<http://hibernate.org/> - Główna strona frameworku Hibernate

<https://spring.io/> - Główna strona frameworku Spring

<https://github.com/> - Główna strona repozytoriów git

<http://www.postgresql.org.pl/> - Główna strona systemu bazodanowego PosgreSQL

# Spis ilustracji

[Rysunek 1: Uproszczony diagram budowy systemu 8](#_Toc424252979)

[Rysunek 2: Uproszczony model modułu backend 9](#_Toc424252980)

[Rysunek 3: Uproszczony model modułu frontend 10](#_Toc424252981)

[Rysunek 4: Diagram relacyjnej bazy danych 11](#_Toc424252982)