Studia i Materiały Informatyki Stosowanej, Tom 5, Nr 13, 2013
str. 29-38

PROJEKT I IMPLEMENTACJA APLIKACJI INTERNETOWEJ WSPOMAGAJĄCEJ TWORZENIE PLANU ZAJĘĆ

Łukasz Rydzkowski

Uniwersytet Kazimierza Wielkiego
Wydział Matematyki, Fizyki i Techniki
Instytut Mechaniki i Informatyki Stosowanej
ul. Kopernika 1, 85-074 Bydgoszcz
e-mail: lukasz.rydzkowski@gmail.com

Streszczenie: *Generowanie planu zajęć jest problem złożonym obliczeniowo, który można spotkać w praktycznie każdej placówce edukacyjnej. Niniejszy artykuł przedstawia aplikację internetową wspomagającą tworzenie planu zajęć. Można tu znaleźć teoretyczne informacje dotyczące planu zajęć oraz opis poszczególnych modułów aplikacji..*

Słowa kluczowe: *Harmonogram, plan zajęć, aplikacja internetowa, PHP, JavaScript*

Project and implementation of a web application supporting creation schedule

Abstract: *Generating schedule is computationally complex problem that may find in all of education facilities. This paper contains the description of a web application supporting creation schedule. You can find there theoretical information about schedule generating and description program modules.*

Keywords: *schedule, plan, web application, PHP, JavaScript*

1. WSTĘP

Układanie planu zajęć jest złożonym problemem optymalizacyjnym, w którym pod uwagę trzeba wziąć bardzo wiele czynników. Dlatego też powszechną praktyką podczas układania planów w placówkach edukacyjnych jest korzystanie z aplikacji wspomagających ten proces. Programy tego typu można podzielić na aplikację w pełni automatyczne, które układają harmonogram bez konieczności ingerencji użytkownika oraz na aplikację interaktywne, w których to użytkownik sam układa plan poprzez ręczne umieszczanie elementów przedstawiających przedmioty w określonych okresach czasu. Dodatkowo można wyróżnić programy hybrydowe stanowiące połączenie obu przedstawionych typów i do tego rodzaju aplikacji należy opisywany w tym artykule program

2. UKŁADANIE PLANU ZAJĘĆ

Aplikacja opisywana w niniejszej publikacji przeznaczona jest do tworzenia planu zajęć uczelni, który stanowi pewien zbiór powiązań pomiędzy grupami, wykładowcami, przedmiotami oraz salami w jednostce czasu. Należy również zaznaczyć że w planie zajęć w odróżnieniu od planu lekcji poszczególne grupy nie są rozłączne.
Plan powinien spełnić szereg warunków, które można podzielić na dwie podstawowe grupy. Pierwsza z nich to warunki mocne, które muszą być spełnione aby stworzony harmonogram mógł zostać zastosowany w praktyce. Aby spełnić warunki mocne w danej sali oraz w danej jednostce czasu nie mogą jednocześnie odbywać się więcej niż jedne zajęcia. Wykładowca nie może w danej jednostce czasu prowadzić więcej niż jednych zajęć. Dana grupa studentów w danej jednostce czasu nie może mieć więcej niż jednych zajęć, grupy studentów nie muszą być rozłączne, a więc warunek ten dotyczy również grup powiązanych z daną grupą. Wykładowca nie może prowadzić zajęć w jednostce czasu, która oznaczona jest dla niego jako niedostępna. W sali nie mogą odbywać się zajęcia w jednostce czasu, która jest oznaczona dla niej jako niedostępna. Przed i po każdych zajęciach muszą odbywać się przerwy o z góry określonej dla danego przedmiotu długości. Przedmioty mogą odbywać się jedynie w salach, do których są przypisane w sposób bezpośredni lub pośredni poprzez rodzaj zajęć, do którego dany przedmiot należy. Dana grupa studentów nie może mieć zajęć w sali jeśli liczba miejsc w tej sali jest większa niż liczebność grupy.
Druga grupa warunków zawiera tzw. warunki słabe, których spełnienie nie jest konieczne ale decyduje o jakości planu. Aby spełnić warunki słabe zajęcia dla poszczególnych grup oraz wykładowców powinny być skonsolidowane, tzn. nie powinny występować pomiędzy nimi przerwy dłuższe niż jest to wymagane. Ponadto zajęcia dla poszczególnych grup oraz wykładowców powinny być równomiernie rozłożone w całym powtarzającym się okresie czasu, z reguły jest to tydzień.

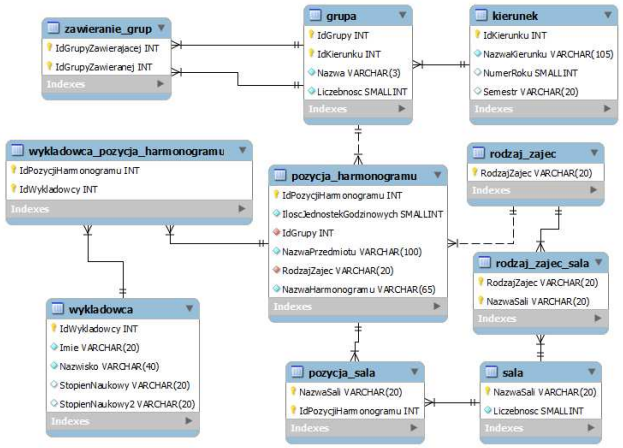
3. OGÓLNA STRUKTURA APLIKACJI

Program opisywany w niniejszym artykule można podzielić na kilka podstawowych modułów. Pierwszy z nich to baza danych MySQL przechowująca informację potrzebne do wygenerowania planu oraz kod wykonywany na serwerze WWW napisany w języku PHP odpowiadający za komunikację z bazą danych. Kolejny moduł stanowi algorytm automatycznie generujący plan zajęć oraz kod zarządzający plikami zawierającymi wygenerowane plany zajęć. Ten moduł również został napisany z wykorzystaniem PHP. Ostatnią część programu stanowi kod działający w przeglądarce użytkownika odpowiedzialny za wyświetlenie oraz działanie interfejsu użytkownika. W skład tego modułu wchodzi również kod umożliwiający modyfikowanie wygenerowanego planu. Opisywany moduł został napisany z wykorzystanie języka programowania JavaScript, języka znaczników HTML oraz arkuszy stylów CSS.
Program został przetestowany w następujących przeglądarkach: Firefox 17.0.1, Chrome 23, Safari 5.1.7, Opera 12.12 oraz Internet Explorer 9. Rozmiar wszystkich elementów znajdujących się na poszczególnych stronach wchodzących w skład aplikacji określony jest za pomocą wartości podanych w jednostce em, która jest wartością procentową w stosunku do wysokości tekstu ustawionego dla całego dokumentu. Dzięki takiemu podejściu po otwarciu strony program sprawdza rozdzielczość ekranu użytkownika i na jej podstawie ustawia rozmiar czcionki dokumentu co pozwala w łatwy sposób skalować wszystkie elementy na stronie.

9. BAZA DANYCH

W bazie danych są przechowywane jedynie informacje potrzebne do wygenerowania planu zajęć, natomiast wygenerowane plany zajęć są przechowywane w plikach.

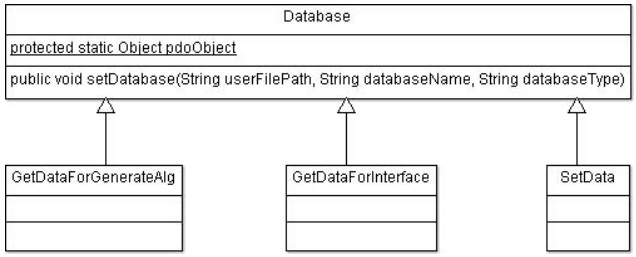
Baza została zaprojektowana poprzez stworzenie diagramu związków encji w programie MySQL Workbench 5.2.44, który umożliwia wygenerowanie gotowego kodu SQL na podstawie stworzonego diagramu. Gotowy model związków – encji wygląda następująco:



Rysunek 1 Diagram związków encji

Centralnym elementem bazy danych jest tabela `pozycja_harmonogramu`, która jak nazwa wskazuje zawiera informacje dotyczące poszczególnych pozycji harmonogramów. W skład tych informacji wchodzi nazwa przedmiotu, nazwa harmonogramu oraz ilość jednostek godzinowych jaka musi zostać zrealizowana w okresie, dla którego generowany jest plan. Ponadto tabela `pozycja_harmonogramu` powiązana jest związkiem wiele do wielu z tabelą `wykładowca`, dzięki czemu możliwe jest pobranie listy wykładowców powiązanych z konkretną pozycją harmonogramu. Struktura bazy umożliwia przypisanie dowolnej liczby wykładowców. Możliwa jest również sytuacja gdy do danej pozycji nie jest przypisany żaden wykładowca. Kolejnym związkiem opisywanej tabeli jest związek wiele do wielu z tabelą `sala`, dzięki czemu możliwe jest przypisanie konkretnych sal do pozycji harmonogramu, co wymusza odbycie się zajęć w jednej z tych sal. Przypisanie pozycji harmonogramu do sali niweluje powiązanie pozycji z typem zajęć, które przedstawione jest na diagramie jako związek jeden do wielu, ponieważ konkretna pozycja może mieć przypisany tylko jeden typ zajęć, natomiast typ zajęć może być powiązany z wieloma pozycjami harmonogramu. Powiązanie pomiędzy typem zajęć a pozycją harmonogramu oznacza, że konkretny przedmiot może odbywać się tylko w salach, które mają przypisany ten sam rodzaj zajęć. Połączenie pomiędzy rodzajem zajęć a salą jest oznaczone poprzez związek wiele do wielu pomiędzy tabelami `rodzaj_zajęć` oraz `sala`. Ostatnią informacją przechowywaną w tabeli `pozycja_harmonogramu` jest numer id grupy, która jest przypisana do konkretnych zajęć. Pomiędzy tabelami `grupa` oraz `pozycja_harmonogramu` występuje związek jeden do wielu, ponieważ do danej pozycji może być przypisana tylko jedna grupa, natomiast poszczególne grupy mogą być przypisane do wielu przedmiotów. Na Rysunku 1 można jeszcze znaleźć tabelę `kierunek`, która powiązana jest z tabelą `grupa` i związek ten oznacza przyporządkowanie grup do poszczególnych kierunków studiów oraz tabelę `zawieranie_grup`, w której znajdują się informacje dotyczące zależności pomiędzy poszczególnymi grupami.

Jak już wcześniej wspomniano za bezpośrednią komunikację z bazą danych odpowiada kod napisany w języku PHP, a konkretnie biblioteka PDO (ang. PHP Data Objects) wbudowana w ten język i stanowiąca interfejs różnych systemów zarządzania relacyjnymi bazami danych w tym m.in. MySQL. Uproszczona struktura klas odpowiadających za komunikację z bazą wygląda następująco:



Rysunek 2 Diagram klas odpowiedzialnych za komunikację z bazą

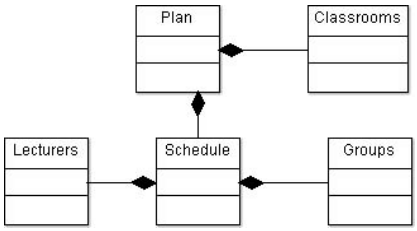
Kluczowym elementem diagramu na Rysunku 2 jest klasa `Database`, która udostępnia statyczną metodę `Database::setDatabase()` tworzącą obiekt PDO przechowywany w atrybucie `Database::pdoObject`. Konstruktor klasy `Database` jako parametr przyjmuję ścieżkę do pliku zawierającego informacje potrzebne do logowania do bazy, nazwę bazy danych oraz typ bazy danych. Informacje dotyczące logowania do bazy zapisane są w następującym formacie: `login;;;hasło;;;host`.

Wszystkie klasy, które mają wykonywać operacje związane z bazą danych muszą dziedziczyć po klasie `Database` w celu uzyskania dostępu do obiektu `Database::pdoObject`. Na powyższym diagramie można wyróżnić dwie klasy odpowiedzialne za pobieranie

danych z bazy i są to odpowiednio klasy: `GetDataForGenerateAlg` udostępniająca metody pobierające i odpowiednio przygotowujące dane potrzebne w algorytmie generującym plan zajęć oraz `GetDataForInterface` udostępniająca metody pobierające i odpowiednio przygotowujące dane wyświetlane na stronie generowania planu oraz na stronie modyfikowania danych. Ostatnia znajdująca się na diagramie klasa o nazwie `SetData` udostępnia metody dodające, modyfikujące oraz usuwające dane z bazy.

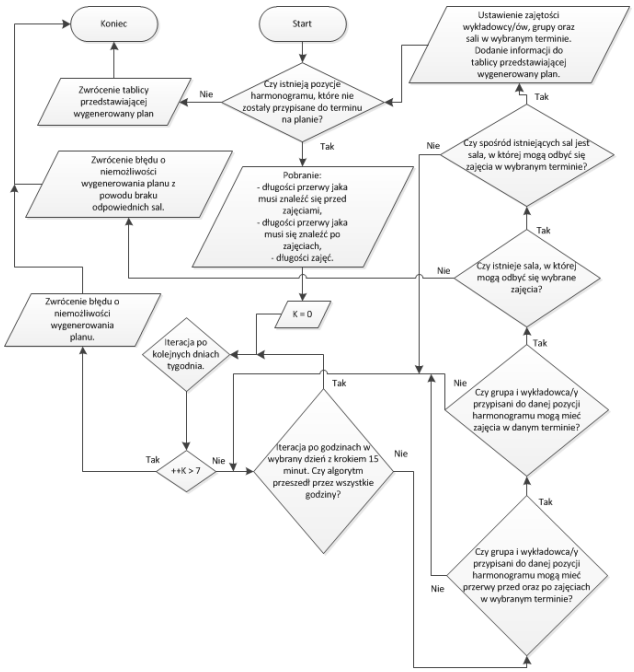
10. MODUŁY DZIAŁAJĄCE NA SERWERZE WWW

W skład opisanego w tej części artykułu modułu wchodzi algorytm generujący gotowy plan zajęć oraz kod odpowiedzialny za zarządzanie plikami. Zaimplementowany przez autora programu algorytm jest algorytmem typu brute force, ponieważ poszczególne zajęcia są wstawiane w pierwsze znalezione okresy czasu spełniające warunki konieczne do zrealizowania danych zajęć. Uproszczona struktura klas odpowiedzialnych za wygenerowanie planu zajęć wygląda następująco:



Rysunek 3 Struktura klas algorytmu generującego plan zajęć

Schemat blokowy przedstawiający poszczególne kroki wykonywane w celu wygenerowania planu przedstawiony został na Rysunku 4.



Rysunek 4 Schemat blokowy algorytmu generującego plan zajęć

Podstawową, niepodzielną jednostką czasu wykorzystywaną podczas generowania planu jest 15 minut, a więc wszystkie wartości związane z czasem takie jak np. długość przerw, długość zajęć czy godzina wystąpienia zajęć są wielokrotnością 15 minut. Kluczowym elementem kodu generującego plan zajęć jest klasa `Plan`. Konstruktor klasy `Plan` jako parametr przyjmuje tablicę asocjacyjną, która zawiera wszystkie informacje potrzebne do wygenerowania planu zajęć. W skład tych informacji wchodzą takie dane jak:

- nazwa generowanego planu,
- długość jednostki godzinowej podana w minutach,
- domyślna długość przerwy oraz długość przerwy przed i po zajęciach dla poszczególnych pozycji harmonogramu jeśli są one różne od domyślnej wartości,
- domyślna liczba jednostek godzinowych jaka może wystąpić obok siebie bez przerwy oraz liczba jednostek godzinowych bez przerwy dla poszczególnych pozycji harmonogramu jeśli wartość ta ma być różna od wartości domyślnej,
- dni tygodnia, w które mogą odbywać się zajęcia wraz z liczbą dni tygodnia w całym okresie, w którym ma być realizowany generowany plan,
- godziny, pomiędzy którymi mogą odbywać się zajęcia w poszczególne dni tygodnia,

- nazwy harmonogramów, dla których ma zostać wygenerowany plan,
- nazwy sal, dla których ma zostać wygenerowany plan,
- opcjonalna niedostępność wykładowców lub sal w konkretne dni tygodnia pomiędzy ustalonymi godzinami.

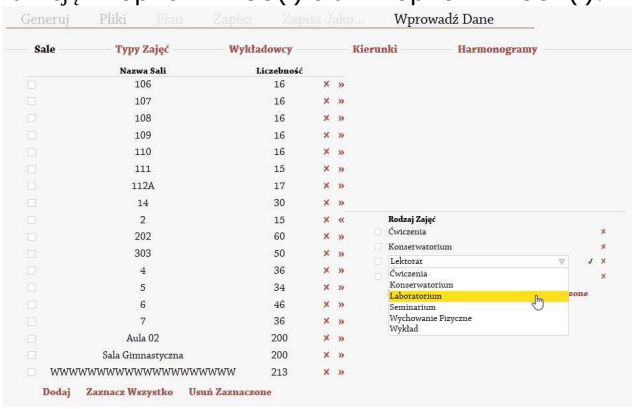
Wewnątrz konstruktora poszczególne elementy tablicy przypisywane są do atrybutów klasy oraz generowane są obiekty klas `Schedule` oraz `Classrooms`. Klasa `Plan` oprócz konstruktora posiada również metodę `Plan::GeneratePlan()`, która poprzez wykonanie kolejnych kroków przedstawionych na Rysunku 4 generuję i zwraca tablicę asocjacyjną stanowiącą odwzorowanie wygenerowanego planu zajęć.

Wygenerowane plany zajęć po zapisaniu przez użytkownika przechowywane są na serwerze w plikach tekstowych o rozszerzeniu *.plan. Pliki te zawierają tablice, zwracane przez metodę `Plan::generatePlan()`, zapisane w formacie JSON. Aplikacja umożliwia eksport planów do plików PDF, które następnie pakowane są w archiwum o rozszerzeniu *.zip i udostępniane użytkownikowi do pobrania. Pliki PDF tworzone są za pomocą udostępnionej w Internecie klasy mPDF napisanej w języku PHP, która generuje pliki w formacie PDF na podstawie kodu HTML. Podczas wykonywania operacji eksportu planów zajęć poszczególne pliki PDF zawierające plany zapisywane są na bieżąco w folderze tymczasowym, a następnie po zapisaniu wszystkich plików pakowane do archiwum zip za pomocą klasy ZipArchive, która wbudowana jest w język PHP.

11. MODUŁY DZIAŁAJĄCE W PRZEGLĄDARCE UŻYTKOWNIKA

W niniejszej części artykułu przedstawiono moduł aplikacji napisany z wykorzystanie języka programowania JavaScript. Na początkowym etapie pisania programu okazało się że konieczne jest napisanie w kodzie JavaScript kilku bibliotek, które wykorzystywane są w części odpowiedzialnej za wyświetlanie oraz działanie interfejsu użytkownika.

Pierwszą stworzoną bibliotekę stanowi obiekt `MyEvent`, który udostępnia metody umożliwiające przypisywanie oraz usuwanie zdarzeń związanych z elementami drzewa DOM. Konieczność napisania tej biblioteki wynikała z tego że w przeglądarce Internet Explorer przypisywanie oraz usuwanie zdarzeń odbywa się w nieco inny sposób niż w pozostałych przeglądarkach. Kolejną napisaną przez autora aplikacji biblioteką jest funkcja `InputForm()` zwracająca obiekty odpowiedzialne za tworzenie oraz wykonywanie działań związanych z polami formularzy. Opisywana funkcja jako parametr przyjmuje obiekt zawierający informacje potrzebne do wygenerowania formularza. Oprócz funkcji `InputForm()` zostały napisane jeszcze dwie biblioteki odpowiedzialne za tworzenie i wykonywanie działań związanych z formularzami. Pierwsza z nich to biblioteka odpowiedzialna za tworzenie i wykonywanie działań związanych z listami rozwijanymi w skład której wchodzą funkcję `DropDownList()` oraz `DropDownList2()`.

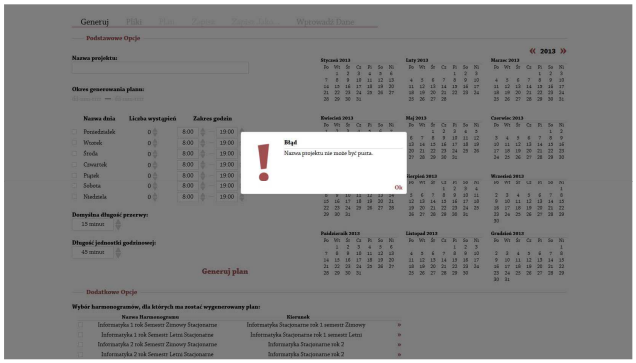


Rysunek 5 Fragment zrzutu ekranu aplikacji z wyświetloną listą rozwijaną

Drugą bibliotekę stanowi funkcja `ScrollingForm()`, która odpowiada za tworzenie obiektów odpowiedzialnych za generowanie oraz wykonywanie zdarzeń związanych z formularzami, w których wartość liczbowa zmieniana jest poprzez przyciski znajdujące się po prawej stronie formularza. Zmiana wartości w formularzu tworzonym przez opisywaną funkcję następuje zarówno w przypadku pojedynczego przyciśnięcia przycisku strzałki znajdującej się obok formularza jak również poprzez ciągłe naciskanie tego elementu. Różnica polega na tym, że podczas pojedynczego naciśnięcia wartość zmieniana jest tylko jeden raz. Natomiast w drugim przypadku wartość zmieniana jest przez cały czas trwania przyciśnięcia klawisza myszy i kolejne wartości zmieniane są coraz szybciej aż do osiągnięcia pewnego minimalnego odstępu czasowego pomiędzy kolejnymi zmianami wyświetlanej wartości.

Rysunek 6 Formularz stworzony za pomocą funkcji `ScrollingForm()`

Kolejną napisaną przez autora pracy bibliotekę stanowi funkcja `Info()`, która odpowiada za wyświetlanie okna zawierającego treść błędu lub ostrzeżenia związanego z działaniami wykonywanymi przez użytkownika. Wyświetlenie okna polega na przyciemnieniu tła i wyświetleniu elementu div w centralnej części ekranu. Przyciemnienie tła realizowane jest za pomocą kontenera div o szerokości i wysokości równej rozmiarom całego wyświetlanego dokumentu. Element ten posiada czarne tło i przezroczystość o wartości 0,5.

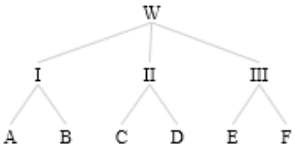


Rysunek 7 Błąd wyświetlony za pomocą funkcji `Info()`

Następną opisywaną biblioteką jest funkcja `GraphDrawer()` odpowiadająca za tworzenie grafów reprezentujących powiązania pomiędzy grupami znajdującymi się na poszczególnych kierunkach. Grafy tworzone są za pomocą elementu canvas, który dostępny jest w hipertekstowym języku znaczników HTML i generowane są na podstawie obiektów o następującej strukturze:

```
{
  W: {
     I: {
        A: 0,
        B: 0
     },
     II: {
        C: 0,
        D: 0
     },
     III: {
        E: 0,
        F: 0
     }
  }
}
```

Graf wygenerowany na podstawie powyższego obiektu wygląda następująco:

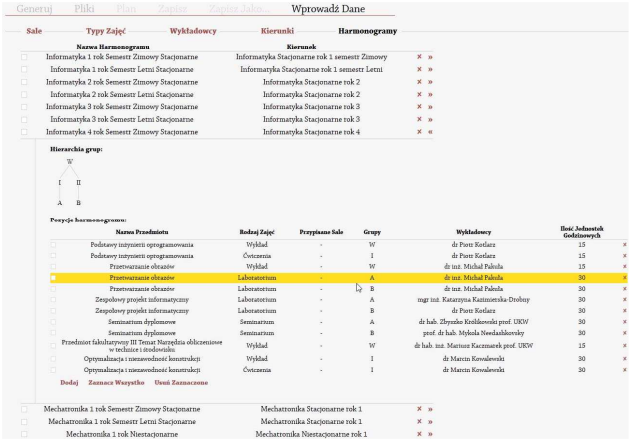


Rysunek 8 Graf przedstawiający powiązania pomiędzy grupami

Kolejnym opisywanym w tym podrozdziale elementem jest funkcja `Ajax()`, która wykorzystywana jest podczas komunikacji pomiędzy przeglądarką użytkownika, a serwerem WWW. W odróżnieniu od pozostałych opisywanych bibliotek funkcja `Ajax()` nie zwraca żadnych obiektów, a przesyłanie danych na serwer odbywa się poprzez wywołanie funkcji. W opisywanej bibliotece za wysyłanie żądań http odpowiada obiekt XMLHttpRequest, a domyślnym typem żądania HTTP jest POST. Oprócz powyższych bibliotek autor napisał również funkcję `Position()` odpowiedzialną za pobieranie pozycji kursora lub elementu drzewa DOM oraz funkcję `dispatchMouseEvent()` wywołującą zdarzenie kliknięcia na podany w parametrze element. W aplikacji wykorzystywane są również biblioteki znalezione przez autora pracy w Internecie. Jest to funkcja `VerticalText()` tworząca przy pomocy elementu canvas tekst obrócony o 90 stopni oraz biblioteki jQuery i jQuery UI.

Interfejs użytkownika składa się z trzech podstawowych sekcji. Pierwsza z nich to sekcja umożliwiająca przeglądanie oraz modyfikowanie danych znajdujących się w bazie danych. Dane znajdujące się w bazie wyświetlane są w postaci tabel generowanych za pomocą kilku funkcji zwracających obiekty. Sekcja podzielona jest na pięć podstron, które umożliwiają modyfikowanie informacji o salach, rodzajach zajęć, wykładowcach, kierunkach oraz harmonogramach. Jednocześnie może być wyświetlona tylko jedna podstrona i w momencie otwierania sekcji ładowana jest podstrona zawierająca informację o salach. Pozostałe podstrony ładowane są w momencie, gdy użytkownik pierwszy raz kliknie w odnośniki przenoszące go do nich, dzięki czemu ładowanie całej sekcji przebiega

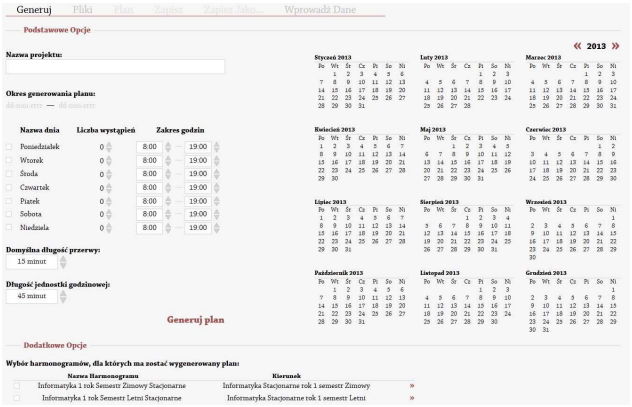
znacznie szybciej niż gdyby wszystkie podstrony miały by być od razu generowane. Oprócz sytuacji, gdy pierwszy raz wchodzimy na daną podstronę są one przeładowywane również w momencie, gdy zostały usunięte lub zmodyfikowane dane na podstronie zawierającej dane powiązane z właśnie otwieraną podstroną. W przypadku dodawania nowych danych nie ma konieczności przeładowywania powiązanych podstron.



Rysunek 9 Fragment zrzutu ekranu zawierający sekcję wprowadzania danych

Drugą sekcję stanowi strona umożliwiająca wygenerowanie planu zajęć. Opisywana sekcja podzielona jest na dwie części. W pierwszej części ustawiana jest nazwa projektu, wybierane są dni tygodnia, w które mają odbywać się zajęcia, wybierany jest zakres godzin dla poszczególnych dni tygodnia pomiędzy którymi mogą odbywać się zajęcia, ustawiana jest domyślna długość przerwy pomiędzy zajęciami oraz ustawiana jest domyślna długość jednostki godzinowej. Na kilka osobnych słów zasługuje całoroczny interaktywny kalendarz umożliwiający wybór okresu, dla którego ma zostać wygenerowany plan.

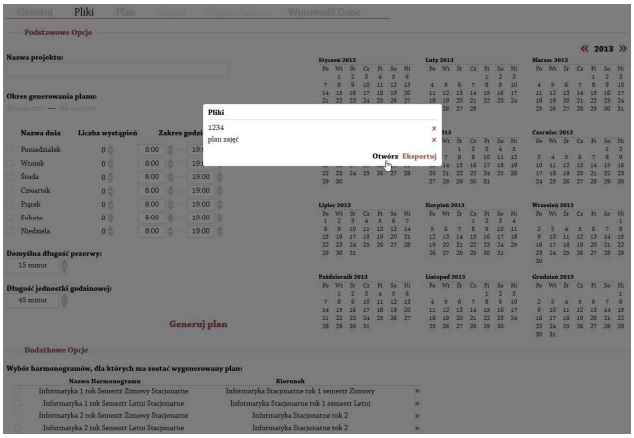
Kalendarz zbudowany jest w oparciu o tabele, a przechodzenie pomiędzy poszczególnymi latami odbywa się poprzez przyciski znajdujące się nad kalendarzem. Kod HTML kolejnych lat generowany jest w momencie pierwszego wyświetlania tych lat. Interaktywność kalendarza polega na możliwości zaznaczania na nim poszczególnych dni, lub całych okresów, w których mogą odbywać się zajęcia. Możliwe jest również ustawienie aby w dany dzień tygodnia np. poniedziałek odbywały się zajęcia przewidziane na np. środę.



Rysunek 10 Fragment zrzutu ekranu przedstawiający pierwszą część sekcji generowania planu

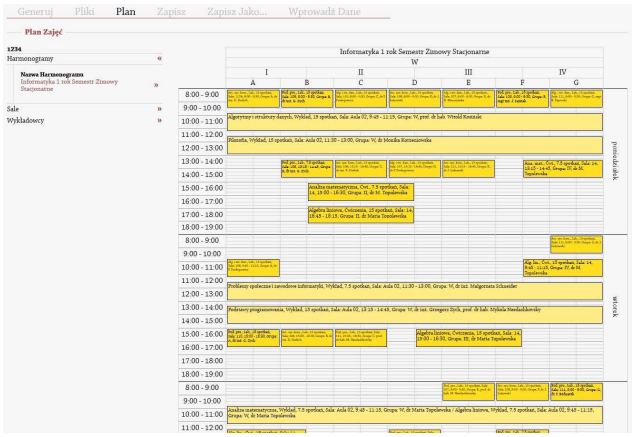
Drugą część opisywanej sekcji stanowią trzy tabele zawierające informację o znajdujących się w bazie harmonogramach, wykładowcach oraz salach. Podstawową funkcją opisywanej części jest wybór, dla których harmonogramów oraz sal ma zostać wygenerowany plan. Możliwe jest również ustalenie liczby jednostek godzinowych, jaka może wystąpić bez przerwy oraz długość przerwy przed i po dla poszczególnych pozycji harmonogramu. Tabele zawierające dane o wykładowcach i salach zawierają również formularze umożliwiające ustawianie niedostępności w poszczególne dni tygodnia, pomiędzy poszczególnymi godzinami. Na podstawie wybranych ustawień generowany jest obiekt, który następnie przesyłany jest za pomocą funkcji `Ajax()` na serwer gdzie zamieniany jest na tablicę asocjacyjną za pomocą wbudowanej w PHP funkcji `json_decode()`. Stworzona tablica stanowi parametr potrzebny do wygenerowania obiektu klasy `Plan`, która jest szerzej opisana w punkcie 5 Moduły działające na serwerze WWW.

Trzecią i ostatnią sekcje interfejsu użytkownika stanowi strona przeglądania oraz wprowadzania zmian w planie zajęć. Przejście do opisywanej sekcji następuje poprzez wygenerowanie nowego planu zajęć lub poprzez okno umożliwiające zarządzanie zapisanymi planami, które można wyświetlić klikając w odnośnik znajdujący się w głównym menu.



Rysunek 11 Fragment zrzutu ekranu przedstawiający okno zarządzania zapisanymi planami zajęć

Sposób wyświetlania okna jest identyczny jak w przypadku okien zawierających treść błędów lub ostrzeżeń, które wyświetlane są za pomocą funkcji `Info()`. Pliki planów zajęć znajdujące się na serwerze zawierają tekst w formacie JSON, który podczas otwierania planów jest pobierany i zamieniany na obiekt JavaScript za pomocą funkcji `eval()`. Okno zarządzania planami, oprócz otwierania zapisanych planów, umożliwia również eksport planów do formatu PDF. Sekcja wyświetlająca wygenerowane plany zajęć składa się z dwóch kolumn. Węższa kolumna, znajdująca się z lewej strony ekranu, zawiera rozwijane menu podzielone na trzy części zawierające odnośniki do poszczególnych planów. Pierwsza część zawiera odnośniki do planów dla całych harmonogramów oraz poszczególnych grup, druga do planów dla sal, trzecia natomiast do planów zajęć wykładowców. Drugą kolumnę stanowi plan wyświetlany na bazie tabeli. Oznacza to, że tło stanowiące siatkę dla bloków przedstawiających poszczególne przedmioty stanowi wygenerowana tabela, w której wiersze opisane są poprzez godziny w poszczególnych dniach tygodnia a kolumny poprzez nazwy grup, nazwę sali lub informację o wykładowcy. Przedmioty na planie przedstawione są za pomocą bloków div z ustawionym atrybutem CSS `position: absolute`. Do pobrania pozycji, w jakiej mają znaleźć się poszczególne bloki div wykorzystywana jest biblioteka jQuery a dokładniej funkcja `position()`. Pobierana jest pozycja komórki tabeli, której atrybut id odpowiada godzinie oraz dniu tygodnia rozpoczęcia zajęć i pozycja ta jest wykorzystywana podczas ustawiania dla kontenera div atrybutów CSS left oraz top. Następnie pobierana jest komórka, w której kończą się zajęcia i na tej podstawie obliczana jest wysokość oraz szerokość kontenera. Oczywiście możliwa jest sytuacja, w której zajęcia odbywają się naprzemiennie i kontenery div przedstawiające takie zajęcia są łączone w jeden element, a informacje dotyczące odbywających się przedmiotów są oddzielone znakiem /. Modyfikowanie wygenerowanego planu zajęć odbywa się poprzez przenoszenie za pomocą kursora myszy poszczególnych kontenerów symbolizujących zajęcia na planie. Przenoszenie bloków div jest realizowane za pomocą funkcjonalności dostępnych w bibliotece jQuery UI. Elementy mogą być przenoszone tylko po osi Y o odległość będącą wielokrotnością wysokości pojedynczego wiersza tabeli przedstawiającej plan i nie mogą wykraczać poza obszar tabeli. W momencie rozpoczęcia przenoszenia elementu div kolorem szarym podświetlane są obszary, w których przenoszone zajęcia nie mogą się odbywać z powodu występowania konfliktów. W sytuacji gdy użytkownik spróbuje przenieść kontener w takie miejsce automatycznie wróci on na pozycję, z której został przeniesiony.



Rysunek 12 Fragment zrzutu ekranu przedstawiający sekcję wprowadzania zmian w wygenerowanym planie zajęć

12. PODSUMOWANIE I WNIOSKI

W niniejszym artykule przedstawiono aplikację internetową przeznaczoną do wspomagania generowania planu zajęć. Opisywany program dotyka problemu powszechnie spotykanego w placówkach naukowych ale jego praktyczne wykorzystanie mogłoby się wiązać z koniecznością jego dalszej rozbudowy. Ewentualny kierunek dalszego rozwoju jest podyktowany niedoskonałością algorytmu generującego plan zajęć, która objawia się tym, że część

wygenerowanych planów zajęć charakteryzuje się sporą liczbą „okienek", czyli przerw pomiędzy zajęciami, które są dłuższe niż jest to konieczne. W celu zniwelowania tej wady należałoby zmodyfikować istniejący algorytm w taki sposób aby zoptymalizować proces układania planu zajęć. Jedną z dróg do osiągnięcia tego celu jest zaimplementowanie algorytmu poszukiwania z tabu, który został opracowany przez Freda Glovera i zaprezentowany w publikacjach przedstawionych przez niego w latach 1989 i 1990.

Literatura

1. Norberciak M., Przegląd metod automatycznego planowania – przykład wykorzystania algorytmu genetycznego w rozwiązywaniu prostego problemu planowania, Prace Naukowe Wydziałowego Zakładu Informatyki Politechniki Wrocławskiej – Zeszyt Sztuczna Inteligencja nr 1
2. Stefanov S., JavaScript Programowanie Obiektowe, Helion, Wydanie Pierwsze, 2010
3. Welling L., Thomson L., PHP i MySQL Tworzenie stron WWW Vademecum profesjonalisty, Helion, Wydanie Czwarte, 2009/10
4. Witczak E., Zastosowanie algorytmów poszukiwania z tabu do optymalizacji układania planu zajęć, 2009
5. Zandstra M., PHP Obiekty, wzorce, narzędzia, Helion, Wydanie Trzecie, 2011.

Studia i Materiały Informatyki Stosowanej, Tom 5, Nr 12, 2013
str. 29-38