



---

# POLITECHNIKA POZNAŃSKA

---

WYDZIAŁ INFORMATYKI I TELEKOMUNIKACJI  
Instytut Informatyki

Praca dyplomowa inżynierska

## **APLIKACJA INTERNETOWA SŁUŻĄCA DO GENEROWANIA PLANÓW LEKCJI DLA SZKÓŁ PODSTAWOWYCH ORAZ ŚREDNICH**

Mateusz Biernacki, 140681

Dominik Boła, 136524

Maciej Gorał, 132228

Grzegorz Piątkowski, 135868

Promotor

dr inż. Izabela Janicka-Lipska

POZNAŃ 2022



# Spis treści

<b>1</b>	<b>Wstęp</b>	<b>1</b>
<b>2</b>	<b>Podstawy teoretyczne</b>	<b>3</b>
2.1	Problem optymalizacyjny . . . . .	3
2.2	Problem NP-trudny . . . . .	3
2.3	Heurystyka . . . . .	3
2.4	Algorytm ewolucyjny . . . . .	3
<b>3</b>	<b>Analiza i porównanie możliwych rozwiązań</b>	<b>5</b>
3.1	Analiza problemu . . . . .	5
3.2	Aktualnie dostępne rozwiązania . . . . .	6
3.2.1	aSc TimeTables . . . . .	6
3.2.2	Prime Timetable . . . . .	6
3.2.3	SuperSaas . . . . .	6
3.3	Możliwe podejścia . . . . .	6
3.4	Wymagania funkcjonalne i нефункционалне . . . . .	7
<b>4</b>	<b>Przygotowanie infrastruktury informatycznej</b>	<b>8</b>
<b>5</b>	<b>Projekt i implementacja aplikacji internetowej w technologii Vue.js</b>	<b>9</b>
5.1	Narzędzia i technologie . . . . .	9
5.1.1	Node.js . . . . .	9
5.1.2	Vue.js . . . . .	9
5.1.3	Vuex . . . . .	9
5.1.4	Jest . . . . .	10
5.1.5	Json Web Tokens . . . . .	10
5.1.6	Postman . . . . .	10
5.1.7	Visual Studio Code . . . . .	11
5.1.8	Axios . . . . .	11
5.1.9	Bootstrap . . . . .	11
5.2	Widoki . . . . .	12
5.2.1	Strona główna . . . . .	12
5.2.2	Widok szkoły . . . . .	12
5.2.3	Rejestracja . . . . .	13
5.2.4	Logowanie . . . . .	13
5.2.5	Ankieta . . . . .	13
5.2.6	Dodawanie przedmiotów . . . . .	14
5.2.7	Dodawanie nauczycieli . . . . .	14

5.2.8	Dodawanie sali lekcyjnych . . . . .	15
5.2.9	Dodawanie klas . . . . .	15
5.2.10	Edycja danych . . . . .	16
<b>6</b>	<b>Projekt i implementacja strony serwerowej opartej na architekturze REST w technologii Django oraz bazy danych MySQL</b>	<b>17</b>
<b>7</b>	<b>Projekt i implementacja algorytmu generującego plan lekcji</b>	<b>18</b>
7.1	Działanie algorytmu . . . . .	18
<b>8</b>	<b>Instrukcja użytkownika</b>	<b>20</b>
<b>9</b>	<b>Zakończenie</b>	<b>21</b>
	<b>Literatura</b>	<b>22</b>

# Rozdział 1

## Wstęp

W dzisiejszych czasach ciężko znaleźć osobę która nigdy nie korzystała z internetu. Dla większości naszego społeczeństwa używanie go jest na porządku dziennym. Wykorzystywany jest prawie w każdej dziedzinie życia, służyć może np. do komunikacji w czasie rzeczywistym, dokonywania szybkich płatności, zakupów internetowych, nauki czy też pracy. Przykłady można wymieniać bez końca, jednak trzeba zwrócić szczególną uwagę na narzędzia, dzięki którym możemy korzystać z sieci w tak szerokim zakresie. Jednym z najpopularniejszych wykorzystywanych oraz dynamicznie rozwijanych rozwiązań są aplikacje webowe. Jedną z ich głównych zalet jest możliwość korzystania z nich niezależnie od używanego urządzenia, o ile ma ono dostęp do internetu oraz posiada przeglądarkę internetową. W przeciwieństwie do aplikacji desktopowych, wszelkie aktualizacje są dokonywane przez administratora, co w kontekście rozwoju oprogramowania jest bardzo wygodne zarówno dla programisty jak i użytkownika. Są to jedne z wielu zalet, które z pewnością przyczyniły się do szybkiego rozwoju aplikacji internetowych oraz związanych z nimi technologii.

Tematem podjętym w pracy jest aplikacja służąca do generowania planów zajęć. Zaprojektowanie tego typu rozwiązania, daje możliwość nauki rozmaitych technologii informatycznych, powszechnie wykorzystywanych w praktyce. Główną motywacją do podjęcia takiego tematu stanowią wady obecnie stosowanego przez większość szkół manualnego tworzenia planów zajęć. Ręczne tworzenie planu jest czasochłonne i wymaga dużego nakładu pracy. Dla osób odpowiedzialnych za ich przygotowanie (dalej zwanych planistami) jest to zadanie monotonne, a także przytłaczające. Planisci, nawet ci z dużym doświadczeniem, nie są zdolni do utworzenia planu, który optymalnie wykorzystywałby godziny uczniów, nauczycieli, a także dostępność sali lekcyjnych. Skutkuje to znaczną liczbą niewykorzystanego czasu w środku dnia lekcyjnego.

Celem pracy jest zaprojektowanie aplikacji, dzięki której po podaniu niezbędnych danych, możliwe byłoby automatyczne wygenerowanie planu zajęć dla szkoły. Aplikacja ma umożliwić planiście dodawanie danych o przedmiotach, nauczycielach, salach i klasach. Na podstawie podanych danych planista ma mieć możliwość generacji rozkładu zajęć dla wszystkich klas w szkole. Aplikacja ma być przeznaczona dla szkół podstawowych oraz średnich. Ograniczenie to wynika z założenia niepodzielności klasy. W przypadku uczelni wyższych niejednorodny podział na grupy znacząco zwiększa poziom skomplikowania rozwiązywanego problemu.

Projekt można podzielić na pięć głównych części: konfigurację infrastruktury informatycznej, implementację back-end, implementację front-end, implementację algorytmu oraz testy.

Praca ma następującą strukturę. Rozdział drugi poświęcony jest podstawom teoretycznym. Rozdział trzeci zawiera analizę problemu i dostępnych rozwiązań. Rozdział czwarty to opis infrastruktury informatycznej. Rozdział piąty omawia część frontendową aplikacji. Rozdział szósty charakteryzuje backend aplikacji. Rozdział siódmy wyjaśnia działanie algorytmu generacji planu.

Rozdział ósmy opisuje testy. Rozdział dziewiąty stanowią wnioski. Rozdział dziesiąty jest podsumowaniem pracy.

Implementacja aplikacji została wykonana przez cztery osoby. Mateusz Biernacki wykonał ... Dominik Boła wykonał ... Maciej Goral wykonał ... Grzegorz Piątkowski wykonał ...

## Rozdział 2

# Podstawy teoretyczne

### 2.1 Problem optymalizacyjny

Problem optymalizacyjny [11] jest to problem obliczeniowy, który polega na znalezieniu maksymalnej/minimalnej wartości pewnego parametru. Wartość takiego parametru zazwyczaj opisywana jest funkcją, dzięki której wartość parametru zależy od przeszukiwanych danych wejściowych. Jeśli poszukiwana jest jak najmniejsza wartość parametru, mówimy o problemie minimalizacyjnym i odpowiednio w przypadku poszukiwania największej wartości parametru, mówimy o problemie maksymalizacyjnym.

### 2.2 Problem NP-trudny

Problem NP-trudny [9] jest problemem obliczeniowym, dla którego nie jest możliwym znalezienie rozwiązania w czasie wielomianowym przy wykorzystaniu niedeterministycznej maszyny Turinga, a sprawdzenie znalezionej odpowiedzi jest co najmniej tak trudne jak każdego innego problemu z grupy NP. Problem optymalizacyjny jest jednym z problemów należących do grupy NP-trudnych.

### 2.3 Heurystyka

Heurystyka [4] jest techniką rozwiązywania problemów w przypadku, gdy znalezienie dokładnego rozwiązania jest zbyt kosztowne. Metoda heurystyczna oferuje zmniejszenie kosztów rozwiązania problemu, jednak ceną takiego podejścia jest spadek dokładności rozwiązania czy nawet jego poprawności. Przy wykorzystaniu metody heurystycznej otrzymanie optymalnego rozwiązania możliwe jest tylko w szczególnych przypadkach. Tego typu podejście wykorzystuje się również, w przypadku, gdy algorytm dokładny umożliwiające znalezienie rozwiązania optymalnego nie jest znany, w celu zawężenia pola badań.

### 2.4 Algorytm ewolucyjny

Algorytm ewolucyjny [5] jest heurystycznym podejściem do rozwiązywania problemów, które nie mogą zostać rozwiązane w czasie wielomianowym, takie jak grupa problemów NP-trudnych, czy po prostu w celu zmniejszenia kosztów znalezienia rozwiązania problemu. Algorytmy ewolucyjne stosowane samodzielnie używane są zazwyczaj do rozwiązywania problemów optymalizacyjnych. Zastosowanie i działanie algorytmu ewolucyjnego jest bardzo proste do zrozumienia ze względu na

to, że mamy do czynienia na co dzień z podobnym zjawiskiem w naturze czyli z selekcją naturalną. Przebieg działania algorytmu ewolucyjnego składa się z 4 głównych kroków.

1. **Inicjalizacja** – W celu rozpoczęcia działania algorytmu, potrzebna jest pierwsza grupa rozwiązań (dalej nazywana populacją). Populacja zawierać będzie założoną liczbę możliwych rozwiązań (dalej nazywaną osobnikami). Zazwyczaj podczas inicjalizacji osobniki tworzone są w sposób losowy. Takie podejście jest wręcz zalecane, ponieważ umożliwia to przebadanie dużej różnorodności osobników, dzięki czemu finałowe rozwiązanie będzie lepsze.
2. **Selekcja** – Gdy pierwotna populacja jest gotowa, jej osobniki trzeba poddać ocenie. Funkcja oceny powinna składać się ze ściśle opisanych warunków opisujących środowisko, do którego osobniki muszą się przystosować. Im dokładniej środowisko zostanie opisane w funkcji oceny, tym lepsze będzie finałne rozwiązanie. Gdy funkcja jest poprawnie przygotowana, każdy z osobników musi zostać poddany ocenie, po której otrzymuje parametr oceny. Dzięki temu można wyróżnić rozwiązania lepsze od reszty. Z populacji zostaje wybrana założona liczba osobników o najwyższym parametrze oceny. Reszta osobników zostaje zabita.
3. **Ewolucja** – Ewolucja składa się z dwóch kroków: krzyżowania oraz mutacji.
  - a) Krzyżowanie – po otrzymaniu wybranych osobników z selekcji, użyte są one do stworzenia nowego pokolenia dla algorytmu, stając się osobnikami-rodzicami. Wykorzystując charakterystyki osobników-rodziców, utworzona zostaje populacja osobników-dzieci poprzez wymieszanie ze sobą charakterystyk osobników-rodziców. Po utworzeniu nowego pokolenia osobników-dzieci, osobniki-rodzice zostają zabite.
  - b) Mutacja – jest to prawdopodobnie najważniejszy krok ewolucji. Bez niego cała populacja bardzo szybko utknęłaby w miejscu, nie oferując żadnego sensownego rozwiązania. W tym kroku charakterystyka każdego osobnika-dziecka z nowego pokolenia poddana jest małym losowym zmianom w celu zróżnicowania ich od osobników-rodziców. Na końcu tego kroku osobniki-dzieci stają się nowym pokoleniem osobników w populacji, która może ponownie zostać poddana selekcji.
4. **Finalizacja** – Ostatecznie działanie algorytmu musi dobiec końca. W tym kroku z populacji zostaje wybrany osobnik z najwyższym parametrem oceny i zwrócony jako rozwiązanie. Są dwie możliwości, w których zakończenie działania algorytmu może zostać wywołane. Gdy osiągnie on maksymalny czas działania (np. założona maksymalna liczba pokoleń) lub gdy osiągnięty zostanie poszukiwany pułap parametru oceny.



## Rozdział 3

# Analiza i porównanie możliwych rozwiązań

### 3.1 Analiza problemu

Podstawowym problemem w automatycznym tworzeniu planu zajęć jest dobór warunków wykorzystywanym przy generacji. Warunki te można podzielić na niezbędne do utworzenia poprawnego planu oraz warunki dodatkowe, których spełnienie zwiększa użyteczność planu z punktu widzenia planisty.

Wśród warunków niezbędnych należy wyróżnić warunek braku konfliktów. Konflikt ma miejsce, gdy występuje jedna z następujących sytuacji:

- w jednej godzinie lekcyjnej, jednej klasie została przyporządkowana więcej niż jeden przedmiot,
- w jednej godzinie lekcyjnej, jednemu nauczycielowi została przyporządkowana więcej niż jedna klasa,
- w jednej godzinie lekcyjnej jednej sali została przyporządkowana więcej niż jedna klasa.

W przypadku szkół podstawowych oraz średnich do warunków niezbędnych należy również zaliczyć brak niewykorzystanych godzin w środku dnia lekcyjnego dla uczniów. Dodatkowo niektóre zajęcia, takie jak wychowanie fizyczne, mogą być przeprowadzone tylko w specjalnie przeznaczonych do tego salach.

Warunki dodatkowe mogą różnić się w zależności od czynników, które należy wziąć pod uwagę przy pod uwagę przy generacji plany wynikające ze specyfikacji szkoły oraz wymagań personelu dydaktycznego. Do tych czynników można zaliczyć:

- ograniczenia dostępności nauczycieli, wynikające z pracy w innych placówkach oświatowych lub innych powodów,
- ograniczenia wynikające z odległości między salami,
- obecność zajęć nieobowiązkowych, które muszą w danym dniu lekcyjnym być skrajnie pierwsze lub ostatnie,
- minimalizację niewykorzystanych godzin w środku dnia lekcyjnego dla uczniów,
- konieczność grupowania zajęć w przypadku kilku godzin lekcyjnych tego samego przedmiotu jednego dnia – w takim przypadku zajęcia te powinny następować bezpośrednio po sobie oraz w tej samej sali,

- możliwie jak najbardziej równomierne rozłożenie przedmiotów trakcie tygodnia lekcyjnego.

## 3.2 Aktualnie dostępne rozwiązania

### 3.2.1 aSc TimeTables

aSc TimeTables [1] to aplikacja desktopowa wspomagająca przygotowywanie planów zajęć. Narzędzie umożliwia generowanie planów na podstawie zdefiniowanych wymagań, wprowadzenie do nich ręcznych poprawek oraz wyszukiwanie konfliktów we wprowadzonych zmianach. aSc TimeTables jest najbardziej rozbudowanym rozwiązaniem tego typu dostępnym na rynku, pozwalającym na tworzenie planów zajęć dla szkół i uczelni. Do dodatkowych funkcji programu należy możliwość importu danych z pliku, zdolność mapowania szkoły oraz udostępnienia planów uczniom i nauczycielom za pomocą aplikacji mobilnej. Z wszechstronnością i bogactwem funkcji wiąże się wysoki poziom umiejętności potrzebny do poprawnego wykorzystania aplikacji. Do pozostałych wad programu należy brak regularnych aktualizacji, podatność na błędy w generacji planu, wysoka cena oraz dostępność ograniczona do systemu Windows.

### 3.2.2 Prime Timetable

Prime Timetables [13] to aplikacja internetowa przeznaczona dla organizacji edukacyjnych umożliwiająca zarówno ręczne jak i automatyczne układanie planów lekcji. Prime Timetables pozwala na wspólne tworzenie planów przez kilku użytkowników oraz udostępnianie gotowych planów dla uczniów i nauczycieli posiadających konta w serwisie. Aplikacja posiada rozbudowany zestaw narzędzi umożliwiających określanie ograniczeń związanych z automatyczną generacją planu. Główną wadą rozwiązania jest wysoka opłata miesięczna, której wysokość dodatkowo zależy od liczby nauczycieli w szkole.

### 3.2.3 SuperSaas

SuperSaas [15] to program do zarządzania szkołami i innymi instytucjami, którego głównym atutem jest wbudowany system rezerwacji. Przy pomocy konta WordPress użytkownicy aplikacji mogą umawiać terminy wizyt, a także dokonywać za nie płatności. SuperSaas cechuje niska cena oraz dostępność z poziomu przeglądarki. Duża część funkcjonalności aplikacji nie jest przeznaczona dla szkół. Pomimo możliwości wspomagania ręcznego układania planów zajęć, program nie pozwala na automatyczną ich generację, ani nawet wykrywanie konfliktów.

## 3.3 Możliwe podejścia

Możliwe rozwiązania można podzielić w zależności od kilku aspektów. Pierwszym z nich jest wybór rodzaju aplikacji – desktopowej, mobilnej lub internetowej. Ze względu na fakt, że korzystanie z aplikacji wymagać ma wprowadzania dużej ilości danych można założyć, że z punktu widzenia użytkownika najwygodniejsze będzie użycie w tym celu fizycznej klawiatury. Powoduje to odrzucenie wyboru aplikacji mobilnej. Zaletami wyboru aplikacji desktopowej jest możliwość korzystania z niej bez dostępu do internetu oraz bezpieczeństwo związane z lokalnym przechowywaniem danych. Pomimo tych korzyści rozwiązanie to nie oferuje zalet związanych z wyborem aplikacji internetowej – dostępu z dowolnego urządzenia wyposażonego w kompatybilną przeglądarkę, braku wymagań systemowych związanych z obliczeniami i przechowywaniem danych oraz braku konieczności aktualizowania aplikacji przez użytkownika.

### **3.4 Wymagania funkcjonalne i нефункционалне**

## Rozdział 4

# Przygotowanie infrastruktury informatycznej

## Rozdział 5

# Projekt i implementacja aplikacji internetowej w technologii Vue.js

### 5.1 Narzędzia i technologie

#### 5.1.1 Node.js

Node.js [8] jest środowiskiem uruchomieniowym umożliwiającym używanie języka JavaScript poza przeglądarką. Środowisko to charakteryzuje asynchroniczność oraz sterowanie zdarzeniami. Asynchroniczność umożliwia wykonywanie wielu czynności w tym samym czasie bez względu na jednowątkowość wynikającą z ograniczenia języka JavaScript. Sterowanie zdarzeniami jest rozwiązaniem typowym dla interfejsów graficznych. Zapewnia ono elastyczność oraz możliwość tworzenia bardziej interaktywnych elementów GUI. Ponadto Node.js udostępnia menadżera pakietów środowiska Node (NPM - Node Package Manager) dającego możliwość zarządzania zainstalowanymi funkcjonalnościami w prosty i przejrzysty sposób.

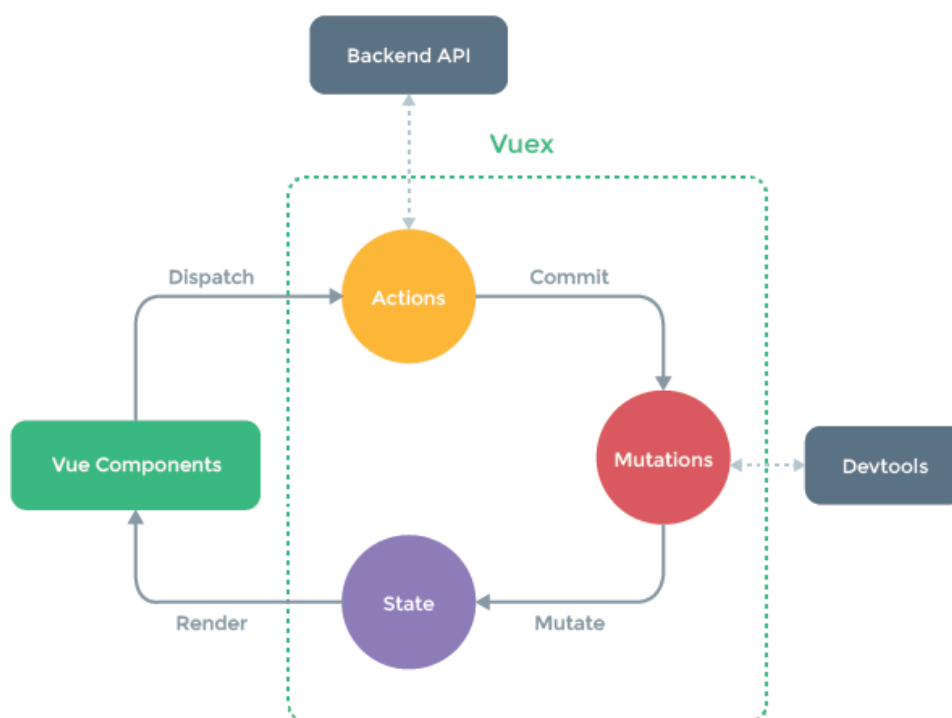
#### 5.1.2 Vue.js

Vue.js [17] to platforma programistyczna języka JavaScript służąca do budowania interfejsów użytkownika. W stosunku do dwóch najpopularniejszych alternatyw - platform programistycznych React oraz Angular - wyróżnia się prostotą, szybkością działania oraz niewielkim rozmiarem. Platforma programistyczna Vue.js została zaprojektowana tak, aby zapewnić jak największą elastyczność. Przy jej użyciu możliwe jest tworzenie nie tylko prostych komponentów, ale i aplikacji typu single-page-application oraz multi-page-application.

Cechą charakterystyczną Vue.js jest wykorzystanie szablonów jako sposobu na powiązanie języka znaczników HTML z warstwą logiki JavaScript. Powiązanie to umożliwia wykorzystywanie w prosty sposób instrukcji warunkowych oraz pętli do wyświetlania zawartości aplikacji.

#### 5.1.3 Vuex

Vuex [18] to biblioteka oferująca scentralizowany magazyn danych dostępny dla wszystkich komponentów w aplikacji. Stan danych w magazynie Vuex jest zmieniany poprzez mutacje wykonywane w reakcji na działanie dyspozytora (zob. rysunek 5.1). Takie podejście sprawia, że dane z części backendowej aplikacji mogą zostać pobrane tylko raz, a później będą one dostępne bezpośrednio w części frontendowej za pośrednictwem magazynu.



RYSUNEK 5.1: Schemat przepływu danych w Vuex [14]

#### 5.1.4 Jest

W projekcie wykorzystano testową platformę programistyczną Jest [6] będącą częścią Vue Test Utils. Vue Test Utils to zestaw funkcjonalności upraszczających testowanie komponentów Vue.js. Zestaw ten zapewnia metody umożliwiające symulowanie działań użytkownika w aplikacji oraz przechwytywanie i porównywanie rezultatów tych interakcji z oczekiwanymi. Jest cechuje brak konieczności konfiguracji, izolacja testów oraz szybkość i bezpieczeństwo działania.

#### 5.1.5 Json Web Tokens

Json Web Token [7] jest otwartym standardem przesyłania zabezpieczonych danych. Dane w formacie Json są podpisywane cyfrowo, co umożliwia weryfikację uprawnień. W aplikacjach internetowych JWT stosowane są głównie do autoryzacji użytkowników oraz zapewnienia bezpieczeństwa przesyłanie informacji pomiędzy frontendem a backendem. Niewielki rozmiar tokenu sprawia, iż możliwe jest przesyłanie go w treści zapytania HTTP lub nawet w jego nagłówku. Ta cecha sprawia również, że token może być przechowywany w pamięci przeglądarki, eliminując konieczność ponownego uwierzytelniania po rozpoczęciu nowej sesji.

#### 5.1.6 Postman

Postman [12] jest zestawem narzędzi do testowania API (Application Programming Interface). Zapewnia on możliwość wysyłania zapytań HTTP dowolnego typu oraz podgląd odpowiedzi i kodów błędów, jeśli takie wystąpiły. Główną zaletą Postmana jest możliwość tworzenia kolekcji

zapytań, które ułatwiają organizację pracy podczas planowania połączeń pomiędzy częścią frontendową i backendową aplikacji. Dodatkowo narzędzie pozwala na współdzielenie kolekcji z zaproszonymi użytkownikami, co znacząco upraszcza proces testowania manualnego. Poza testowaniem manualnym Postman umożliwia tworzenie automatycznych testów przy pomocy języka JavaScript. Dzięki generatorowi losowych danych możliwa jest symulacja działań nawet kilku tysięcy różnych użytkowników w systemie.

### 5.1.7 Visual Studio Code

Visual Studio Code [16] jest edytorem kodu, którego głównymi zaletami jest wsparcie dla debugowania, inteligentnego uzupełniania kodu, refaktoryzacji oraz kontroli wersji. Dużą korzyścią płynącą z korzystania z programu Visual Studio Code jest dostęp do rozszerzeń, usprawniających pracę z kodem w dowolnym języku programowania. Rozszerzenia zapewniają również wsparcie dla platform programistycznych, w tym Vue.js, najbardziej istotnego dla tej części projektu. Mały rozmiar oraz wysoka wydajność znacznie przyspieszają korzystanie z aplikacji i sprzyjają intensywnej iteracji rozwiązań.

### 5.1.8 Axios

Axios [2] jest biblioteką języka JavaScript służącą do wykonywania zapytań HTTP z poziomu Node.js lub przeglądarki. W aplikacjach internetowych wykorzystywany jest do uzyskiwania danych z części backendowej aplikacji. Axios bazuje na obietnicach (promise), co pozwala na obsługiwanie akcji asynchronicznie. Biblioteka może być użyta poprzez zwykły Javascript lub platformę programistyczną taką jak Vue.js. W porównaniu z innymi bibliotekami służącymi do wykonywania zapytań HTTP Axios oferuje wsparcie dla starszych przeglądarek, możliwość ustawienia ograniczenia czasowego dla zapytań, ochronę przed CSRF (Cross-Site Request Forgery), a także automatyczną transformację danych JSON.

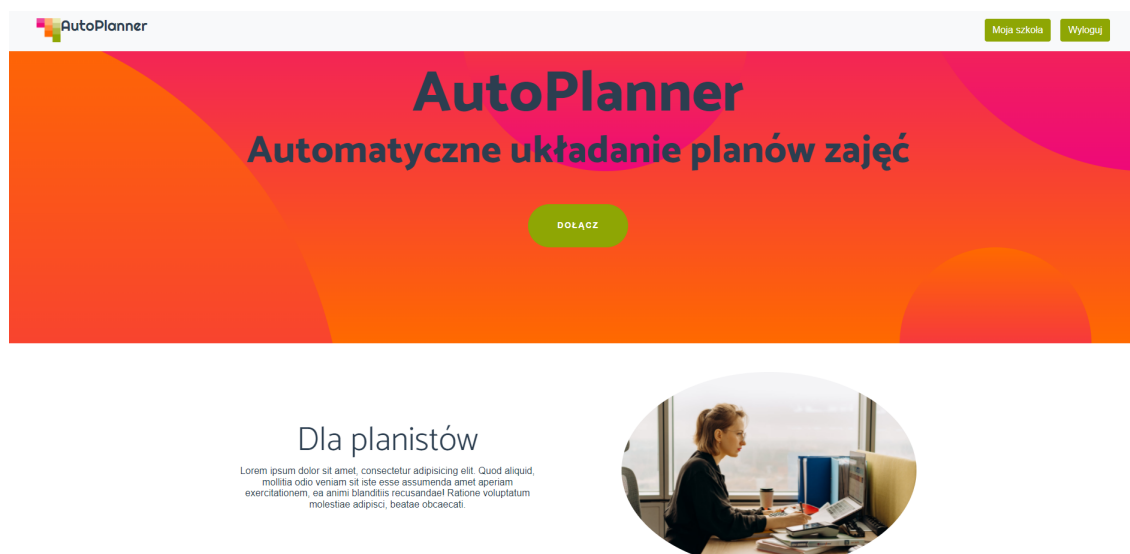
### 5.1.9 Bootstrap

Bootstrap [3] jest platformą programistyczną CSS (Cascading Style Sheets) upraszczającą projektowanie interfejsu graficznego aplikacji internetowych. Bootstrap pomaga zapewnić responsywność stron, a więc poprawne ich wyświetlanie na urządzeniach mobilnych. Przed pojawieniem się tego rozwiązania często występowała konieczność przygotowywania oddzielnych stylów dla ekranów o różnych rozdzielczościach. Dzięki zastosowaniu platformy programistycznej elementy strony internetowej zostają przeskalowane i przemieszczone tak, aby pomieścić się na ekranie niezależnie od jego wielkości i proporcji. Dodatkowo Bootstrap pozwala na zastosowanie zaawansowanych komponentów takich jak paski nawigacji, wskaźniki postępu czy miniatury.

## 5.2 Widoki

### 5.2.1 Strona główna

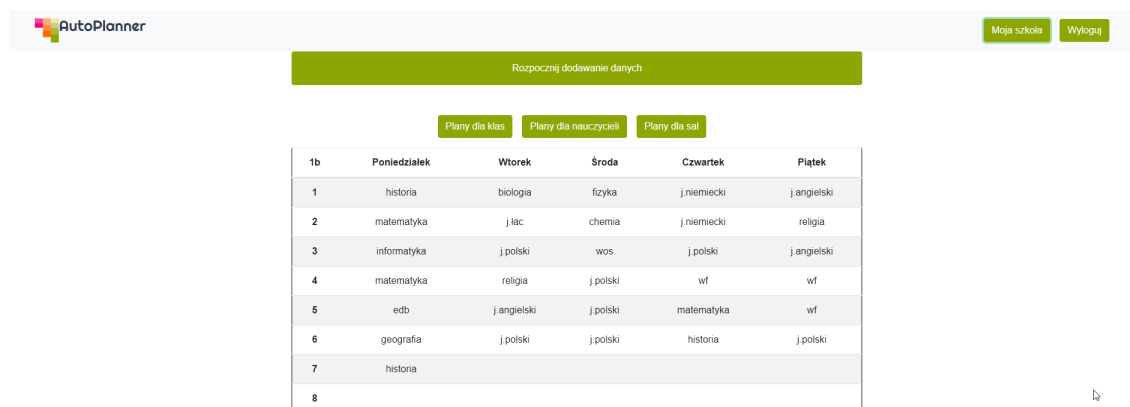
Strona główna (zob. rysunek 5.2) aplikacji została stworzona na bazie szablonu Bootstrap o nazwie One Page Wonder [10]. Zawiera ona krótki opis aplikacji, a także korzyści płynących z wykorzystania jej dla planistów, nauczycieli oraz uczniów. Pasek menu znajdujący się zawsze na górze strony jest stałym elementem aplikacji pojawiającym się w każdym z widoków. Pozwala on na przejście do widoków logowania i rejestracji, a w przypadku gdy użytkownik jest już zalogowany na wylogowanie lub przejście do widoku szkoły.



RYSUNEK 5.2: Aplikacja internetowa – Strona główna

### 5.2.2 Widok szkoły

Widok szkoły (zob. rysunek 5.3) pozwala na przejście do dodawania danych potrzebnych do wygenerowania planu, a przypadku gdy plan został już wygenerowany jest również miejscem, w którym jest on wyświetlany. Rozkład zajęć jest możliwy do wyświetlenia na trzy sposoby – z podziałem na klasy, nauczycieli lub sale lekcyjne.



RYSUNEK 5.3: Aplikacja internetowa – Widok szkoły



### 5.2.3 Rejestracja

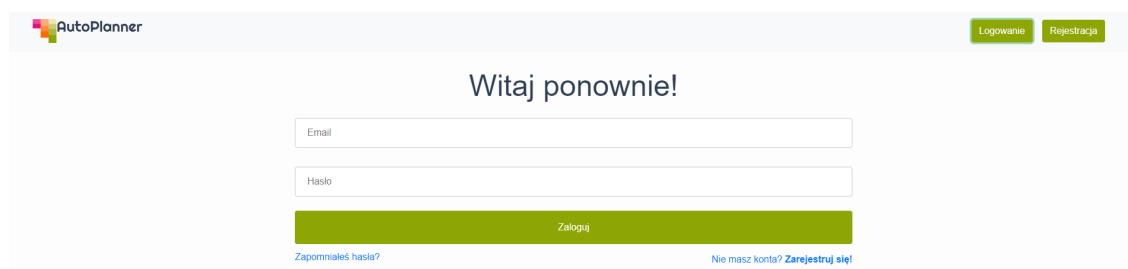
Widok rejestracji (zob. rysunek 5.4) umożliwia utworzenie konta w serwisie. Od użytkownika wymaga się podania adresu e-mail, nazwy użytkownika oraz hasła. Adres e-mail musi być unikatowy. Wynika to z konieczności weryfikacji konta poprzez wiadomość wysłaną przy pomocy serwera SMTP. Rozwiązanie to ma na celu zapobieganie atakom na stronę poprzez masowe tworzenie nowych kont.



RYSUNEK 5.4: Aplikacja internetowa – Widok rejestracji

### 5.2.4 Logowanie

Widok logowania (zob. rysunek 5.5) pozwala na dostęp do konta i zapisanych na nim danych z dowolnego urządzenia. Do uwierzytelnienia użytkownika wykorzystywany jest adres e-mail oraz hasło podane w procesie rejestracji. Powodzenie procesu logowania powoduje otrzymanie przez aplikację tokenu JWT, zapisywanego w pamięci przeglądarki. W przypadku utraty hasła użytkownik posiada możliwość odzyskania go po podaniu adresu e-mail powiązanego z istniejącym kontem.



RYSUNEK 5.5: Aplikacja internetowa – Widok logowania

### 5.2.5 Ankieta

Widok ankiety (zob. rysunek 5.6) pozwala nauczycielowi na podanie preferencji godzinowych pracy. W przeciwieństwie do pozostałych widoków jest on dostępny jedynie poprzez bezpośredni link wysyłany w wiadomości e-mail. Takie rozwiązanie sprawia, że jedynym użytkownikiem, od którego wymagane jest posiadanie konta jest planista. Nauczyciele jako użytkownicy bez konta są identyfikowani dzięki unikalności adresu URL.

#	Poniedziałek	Wtorek	Środa	Czwartek	Piątek
1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

RYSUNEK 5.6: Aplikacja internetowa – Widok ankiet dla nauczycieli

### 5.2.6 Dodawanie przedmiotów

Widok dodawania przedmiotów (zob. rysunek 5.7) stanowi pierwszy krok w procesie podawania danych koniecznych do wygenerowania planu zajęć. W celu dodawania przedmiotu należy podać jedynie jego nazwę. Powiązania z nauczycielami, salami lekcyjnymi i klasami będą mogły być wprowadzone w kolejnych krokach. Podanie nazw przedmiotów na początku procesu dodawania danych pozwala na to, aby w późniejszych etapach mogły być one wybierane z listy rozwijanej. Zapobiega to konieczności wielokrotnego ręcznego wprowadzania tych samych informacji i ułatwia tworzenie powiązań w bazie danych. W lewej części ekranu znajduje się lista już wprowadzonych przedmiotów. Wybranie z nich jednego powoduje przejście do ekranu edycji. Analogiczne rozwiązanie zostało zastosowane we wszystkich kolejnych ekranach dodawania danych.

RYSUNEK 5.7: Aplikacja internetowa – Widok dodawania przedmiotów

### 5.2.7 Dodawanie nauczycieli

W widoku dodawania nauczycieli (zob. rysunek 5.8) planista ma możliwość wprowadzenia danych personelu dydaktycznego oraz jego powiązań z przedmiotami. Każdy nauczyciel musi posiadać imię i nazwisko, unikalny w skali szkoły adres e-mail oraz przynajmniej jeden prowadzony przedmiot. Ekran umożliwia dodanie kolejnych prowadzonych przedmiotów w przypadku, gdy nauczyciel prowadzi więcej niż jeden.

**AutoPlanner** Moja szkoła Wyloguj

### Krok 2 - Dodaj nauczycieli

Janusz Walczuk  
Krystyna Pawłowicz  
Robert Lewandowski  
Waldemar Kiepski  
Ksiądz Robak  
Snoop Dog  
Albert Einstein

**Prowadzone przedmioty**  
  

+

-

Dodaj

Przejdź dalej

Poprzedni krok

RYSUNEK 5.8: Aplikacja internetowa – Widok dodawania nauczycieli

### 5.2.8 Dodawanie sali lekcyjnych

Dodanie informacji o salach lekcyjnych (zob. rysunek 5.9) stanowi trzeci krok dodawania danych. Sala lekcyjna musi posiadać nazwę, a opcjonalnie także listę przedmiotów, które mogą być w niej prowadzone. W przypadku, gdy nie zostanie wybrany żaden preferowany przedmiot, sala zostaje uznana za salę zwykłą, co oznacza, że będzie mógł być w niej prowadzony dowolny przedmiot.

**AutoPlanner** Moja szkoła Wyloguj

### Krok 3 - Dodaj sale lekcyjne

100  
101  
102  
103  
104  
105  
200

☐ Preferowane przedmioty  

Dodaj

Przejdź dalej

Poprzedni krok

RYSUNEK 5.9: Aplikacja internetowa – Widok dodawania sali lekcyjnych

### 5.2.9 Dodawanie klas

Ostatnim etapem w procesie dodawania niezbędnych danych jest wprowadzenie parametrów klas (zob. rysunek 5.10). Każda klasa musi posiadać nazwę oraz listę przedmiotów, które mają się pojawić w jej planie zajęć. Każdy element listy musi zawierać nazwę przedmiotu, liczbę godzin lekcyjnych w tygodniu przeznaczonych na przedmiot oraz opcjonalnie prowadzącego przedmiot. Brak wyboru nauczyciela umożliwia przypisanie zajęć dowolnemu prowadzącemu dany przedmiot.

AutoPlanner

Moja szkoła Wyloguj

### Krok 4 - Dodaj klasy

1A 1B 1C 1D

Nazwa klasy

#### Lista przedmiotów

-- wybierz przedmiot -- Liczba godzin tygodniowo liczba godzin ☐ -- wybierz prowadzącego --

+ -

Dodaj

Przejdź dalej

Poprzedni krok

RYSUNEK 5.10: Aplikacja internetowa – Widok dodawania klas

### 5.2.10 Edycja danych

Dla czterech powyższych ekranów istnieją odpowiadające im ekrany edycji danych. Ze względu na ich analogiczną budowę ich struktura zostanie omówiona na bazie widoku edycji danych nauczyciela (zob. rysunek 5.11). Przejście do tego ekranu umożliwiają przyciski znajdujące się po prawej stronie widoku dodawania nauczycieli. Przyciski te są wciąż obecne w widoku edycji i pozwalają na przechodzenie pomiędzy danymi poszczególnych osób bez zapisywania wprowadzonych zmian. W chwili przejścia do ekranu edycji pola z danymi zostają uzupełnione pierwotnie wprowadzonymi informacjami o nauczycielu. Takie rozwiązanie ma celu zapobieganie konieczności ponownego wprowadzania wszystkich danych, w przypadku gdy tylko niektóre z nich wymagają zmian. Przyciski u dołu pozwalają na zapisanie wprowadzonych zmian, powrót do ekranu dodawania nauczycieli lub całkowite usunięcie nauczyciela z bazy danych.

AutoPlanner

Moja szkoła Wyloguj

### Edycja danych nauczyciela

AA BB

AA BB

AABB@CC.com

#### Prowadzone przedmioty

Matematyka

+ -

Zapisz zmiany

Wróć bez zapisywania

Usuń nauczyciela

RYSUNEK 5.11: Aplikacja internetowa – Widok edycji danych nauczyciela

## Rozdział 6

# Projekt i implementacja strony serwerowej opartej na architekturze REST w technologii Django oraz bazy danych MySQL

## Rozdział 7

# Projekt i implementacja algorytmu generującego plan lekcji

### 7.1 Działanie algorytmu

Problem ułożenia najlepszego planu zajęć jest problemem Np-Zupełnym. W projekcie zostało zaimplementowane podejście ewolucyjne. Na rysunek X została zobrazowana struktura implementowanego planu zajęć.

Rysunek struktury danych

Zagadnienie implementacji algorytmu można podzielić na cztery główne części:

- przygotowanie danych wejściowych,
- ułożenie planu zajęć,
- ocena planu zajęć,
- ewolucja planu zajęć.

Przygotowanie danych wejściowych jest kluczowe w działaniu algorytmu. Na podstawie danych otrzymanych od back-end, zostaje ułożona lista czteroelementowych krotek, gdzie pierwszym elementem jest nazwa grupy, drugim elementem jest nazwa przedmiotu, trzecim elementem jest nazwa nauczyciela, a czwartym elementem jest nazwa sali. Wartość elementu sali jest na początku wartością pustą null, a wartość elementu nauczyciela, jest wartością pustą null, wtedy i tylko wtedy kiedy nie został wskazany nauczyciel dla konkretnej klasy. W takiej liście znajdują się wszystkie jednostki lekcyjne występujące w całej szkole. Przykładowo jeżeli pewna klasa IIC ma mieć 5 matematyk w tygodniu z nauczycielem Jan Kowalski, to do listy zostanie dodane pięć krotek postaci (IIC, matematyka, Jan Kowalski, null). Struktura wspomnianej listy znajduje się na rysunku 7.1.

Głównym założeniem układania planu zajęć jest to, że na podstawie tej samej listy krotek, zawsze zostanie wygenerowany konkretny plan zajęć. Układanie planu zajęć przebiega następująco:

NAZWA GRUPY	NAZWA PRZEDMIOTU	NAZWA NAUCZYCIELA	NAZWA SALI
IIC	matematyka	Jan Kowalski	null
IIC	matematyka	Jan Kowalski	null
IIC	matematyka	Jan Kowalski	null
IIC	język polski	Andrzej Nowak	null
IIC	język polski	Andrzej Nowak	null
IA	język polski	null	null

RYSUNEK 7.1: Struktura listy krotek

1. z listy krotek zostaje zabrana pierwsza z brzegu krotka,
2. wybrana krotka zostaje przydzielona do pierwszej możliwej konkretnej godziny w
3. konkretnym dniu (czyli do takiej jednostki godzinowej, której są spełnione
4. wprowadzone przez użytkownika założenia oraz jest wolna sala),
5. poprzednie czynności zostają powtarzane tak długo, aż cała lista zostanie przeiterowana.

Jeżeli po zakończeniu działania procesu układania planu zajęć lista krotek nie będzie pusta, to ułożony plan zajęć jest niekompletny i niepoprawny. Proces układania planu zajęć został przedstawiony na rysunek X.

Rysunek proces układania planu zajęć

Funkcja oceny przyznaje planu zajęć pewną ilość punktów. Ocena może mieć wartość z zakresu od minus nieskończoności do 0, gdzie im wartość większa, tym lepsza ocena.

TODO LEPSZY AKAPIT FUNKCJI OCENY

Część ewolucyjna algorytmu wykorzystuje wszystkie poprzednie części. Na początku algorytm generuje pewną populację planów zajęć, która będzie nazywana dalej pierwszą generacją. Zawartość listy krotek jest identyczna dla każdej jednostki z populacji, ale kolejność krotek w liście jest pseudo losowo zmieniona. Dzięki takiemu rozwiązaniu, każda jednostka w populacji może wygenerować zupełnie inny plan zajęć. Za pomocą funkcji oceny, do każdego z planów zostaje przydzielona pewna ilość punktów.

Połowa najgorzej ocenionych planów zostaje zabita. Z każdego planu zajęć, któremu udało się przeżyć, generowany jest kolejny plan. Plan, z którego został wygenerowany nowy plan, będzie nazywany w dalszej części pracy rodzicem, natomiast nowo wygenerowany plan będzie nazywany potomkiem. Lista krotek potomka, będzie zawierała tę samą kolejność co rodzic, ale losowo wybrane rekordy zamienia się w sposób pseudolosowy pozycjami w liście, taka zamiana będzie nazywana dalej mutacją. Nowo wygenerowane jednostki zostają ocenione oraz dodane do populacji, w ten sposób powstaje druga generacja planów zajęć.

Czynność z poprzedniego akapitu powtarza się jeszcze  $g$ -razy, gdzie  $g$  to liczba wszystkich generacji. Działanie całego algorytmu zostało przedstawione na diagramie poniżej.

DIAGRAM ALGORYTM

AKAPIT O ZŁOŻONOŚCI

AKAPIT O WIELOWĄTKOWOŚCI

AKAPIT O POTRZEBNYCH ZASOBACH

## Rozdział 8

# Instrukcja użytkowania



## Rozdział 9

# Zakończenie

# Literatura

- [1] aSc TimeTables. [on-line] <https://www.asctimetables.com/#!/home>. (29.12.2021).
- [2] Axios. [on-line] <https://vuejs.org/v2/cookbook/using-axios-to-consume-apis.html>. (29.12.2021).
- [3] Bootstrap. [on-line] <https://getbootstrap.com/docs/5.1/getting-started/introduction/>. (29.12.2021).
- [4] Heuristic. [on-line] <https://www.khanacademy.org/computing/ap-computer-science-principles/algorithms-101/solving-hard-problems/a/using-heuristics>. (29.12.2021).
- [5] Heuristic. [on-line] <https://towardsdatascience.com/introduction-to-evolutionary-algorithms-a8594b484ac>. (29.12.2021).
- [6] Jest. [on-line] <https://vue-test-utils.vuejs.org/guides/>. (29.12.2021).
- [7] Json web tokens. [on-line] <https://jwt.io/introduction>. (29.12.2021).
- [8] Node.js. [on-line] <https://nodejs.org/en/about/>. (29.12.2021).
- [9] Np-hard problem. [on-line] <https://xlinux.nist.gov/dads/HTML/nphard.html>. (29.12.2021).
- [10] One page wonder. [on-line] <https://startbootstrap.com/theme/one-page-wonder>. (29.12.2021).
- [11] Optimization problem. [on-line] <https://xlinux.nist.gov/dads/HTML/optimization.html>. (29.12.2021).
- [12] Postman. [on-line] <https://www.postman.com/product/what-is-postman/>. (29.12.2021).
- [13] Prime Timetable. [on-line] <https://primetimetable.com/help/#overview&q>. (29.12.2021).
- [14] Schemat przepływu danych w vuex. [on-line] <https://vuex.vuejs.org/vuex.png>. (29.12.2021).
- [15] SuperSaaS. [on-line] [https://www.supersaas.com/info/doc/getting\\_started](https://www.supersaas.com/info/doc/getting_started). (29.12.2021).
- [16] Visual studio code. [on-line] <https://code.visualstudio.com/docs>. (29.12.2021).
- [17] Vue.js. [on-line] <https://vuejs.org/v2/guide/>. (29.12.2021).
- [18] Vuex. [on-line] <https://vuex.vuejs.org/>. (29.12.2021).



© 2022 Mateusz Biernacki, Dominik Boła, Maciej Goral, Grzegorz Piątkowski

Instytut Informatyki, Wydział Informatyki i Telekomunikacji  
Politechnika Poznańska

Skład przy użyciu systemu  $\text{\LaTeX}$  na platformie Overleaf.