



---

# POLITECHNIKA POZNAŃSKA

---

WYDZIAŁ INFORMATYKI I TELEKOMUNIKACJI  
Instytut Informatyki

Praca dyplomowa licencjacka

## **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

Tutaj będzie karta pracy dyplomowej;  
oryginał wstawiamy do wersji dla archiwum PP, w pozostałych kopiach wstawiamy ksero.

# 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 . . . . .	5
3.2.1	aSc TimeTables . . . . .	5
3.2.2	Prime Timetable . . . . .	5
3.2.3	SuperSaas . . . . .	5
3.3	Możliwe podejścia . . . . .	6
<b>4</b>	<b>Przygotowanie infrastruktury informatycznej</b>	<b>7</b>
<b>5</b>	<b>Projekt i implementacja aplikacji internetowej w technologii Vue.js</b>	<b>8</b>
5.1	Narzędzia i technologie . . . . .	8
5.1.1	Node.js . . . . .	8
5.1.2	Vue.js . . . . .	8
5.1.3	Vuex . . . . .	8
5.1.4	Jest . . . . .	9
5.1.5	Json Web Tokens . . . . .	9
5.1.6	Postman . . . . .	9
5.1.7	Visual Studio Code . . . . .	10
5.1.8	Axios . . . . .	10
5.1.9	Bootstrap . . . . .	10
5.2	Widoki . . . . .	11
5.2.1	Strona główna . . . . .	11
5.2.2	Widok szkoły . . . . .	11
5.2.3	Rejestracja . . . . .	12
5.2.4	Logowanie . . . . .	12
5.2.5	Ankieta . . . . .	13
5.2.6	Dodawanie przedmiotów . . . . .	13
5.2.7	Dodawanie nauczycieli . . . . .	14
5.2.8	Dodawanie sali lekcyjnych . . . . .	14

5.2.9	Dodawanie klas . . . . .	15
5.2.10	Edycja danych . . . . .	15
<b>6</b>	<b>Projekt i implementacja strony serwerowej opartej na architekturze REST w technologii Django</b>	<b>17</b>
<b>7</b>	<b>Projekt i implementacja algorytmu generującego plan lekcji</b>	<b>18</b>
<b>8</b>	<b>Testy</b>	<b>19</b>
<b>9</b>	<b>Wnioski</b>	<b>20</b>
<b>10</b>	<b>Zakończenie</b>	<b>21</b>
	<b>Literatura</b>	<b>22</b>
<b>A</b>	<b>Składanie dokumentu w systemie <math>\text{\LaTeX}</math></b>	<b>23</b>
A.1	Struktura dokumentu . . . . .	23
A.2	Akapity i znaki specjalne . . . . .	23
A.3	Wypunktowania . . . . .	23
A.4	Polecenia pakietu <code>ppfcmthesis</code> . . . . .	24
A.5	Rysunki . . . . .	24
A.5.1	Tablice . . . . .	25
A.5.2	Checklista . . . . .	25
A.6	Literatura i materiały dodatkowe . . . . .	26

# Rozdział 1

## Wstęp

(Źródła?) Tematem podjętym w pracy jest aplikacja służąca do generowania planów zajęć. Główną motywacją do podjęcia takiego (tego?..) 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 zwanymi planistami) jest to zadanie monotonne, a także przytłaczające. Planiści, 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 niejednolity 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 front-endową 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 ...

Wstęp do pracy powinien zawierać następujące elementy:

- krótkie uzasadnienie podjęcia tematu;
- cel pracy (patrz niżej);
- zakres (przedmiotowy, podmiotowy, czasowy) wyjaśniający, w jakim rozmiarze praca będzie realizowana;
- ewentualne hipotezy, które autor zamierza sprawdzić lub udowodnić;
- krótką charakterystykę źródeł, zwłaszcza literaturowych;

- układ pracy (patrz niżej), czyli zwięzłą charakterystykę zawartości poszczególnych rozdziałów;
- ewentualne uwagi dotyczące realizacji tematu pracy np. trudności, które pojawiły się w trakcie realizacji poszczególnych zadań, uwagi dotyczące wykorzystywanego sprzętu, współpraca z firmami zewnętrznymi.

**Wstęp do pracy musi się kończyć dwoma następującymi akapitami:**

Celem pracy jest opracowanie / wykonanie analizy / zaprojektowanie / .....

oraz:

Struktura pracy jest następująca. W rozdziale 2 przedstawiono przegląd literatury na temat ..... Rozdział 3 jest poświęcony ..... (kilka zdań). Rozdział 4 zawiera ..... (kilka zdań) ..... itd. Rozdział X stanowi podsumowanie pracy.

W przypadku prac inżynierskich zespołowych lub magisterskich 2-osobowych, po tych dwóch w/w akapitach musi w pracy znaleźć się akapit, w którym będzie opisany udział w pracy poszczególnych członków zespołu. Na przykład:

Jan Kowalski w ramach niniejszej pracy wykonał projekt tego i tego, opracował .....  
Grzegorz Brzęczyszczkiewicz wykonał ....., itd.

## Rozdział 2

# Podstawy teoretyczne

### 2.1 Problem optymalizacyjny

Problem optymalizacyjny [10] jest to problem obliczeniowy, który polega na znalezieniu maksymalnej/minimalnej wartości pewnego parametru. Wartość takiego parametru zazwyczaj opisywana jest funkcją, dzięki której wartość parametru zależna jest od przeszukiwanych danych wejściowych. W przypadku 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 rozwiązania 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 umożliwiający 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 gdy 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. **Inizjalizacja** - 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 ilość osobników o najwyższym parametrze oceny. Reszta osobników zostaje zabita.
3. **Ewolucja** - Ewolucja składa się z dwóch pomniejszych 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 utknęła by 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 populacji z najwyższym parametrem oceny i zwrócony jako rozwiązanie. Są dwie możliwości, w których zakończenie działania algorytmu może zostać wywołane. Gdy osiągnię on maksymalny czas działania (np. założona maksymalna ilość pokoleń) lub gdy osiągnięty zostanie poszukiwany pułap parametru oceny.

Rozdział teoretyczny — przegląd literatury naświetlający stan wiedzy na dany temat.

Przegląd literatury naświetlający stan wiedzy na dany temat obejmuje rozdziały pisane na podstawie literatury, której wykaz zamieszczany jest w części pracy pt. *Literatura* (lub inaczej *Bibliografia*, *Piśmiennictwo*). W tekście pracy muszą wystąpić odwołania do wszystkich pozycji zamieszczonych w wykazie literatury. **Nie należy odnośników do literatury umieszczać w stopce strony.** Student jest bezwzględnie zobowiązany do wskazywania źródeł pochodzenia informacji przedstawianych w pracy, dotyczy to również rysunków, tabel, fragmentów kodu źródłowego programów itd. Należy także podać adresy stron internetowych w przypadku źródeł pochodzących z Internetu.



## Rozdział 3

# Analiza i porównanie możliwych rozwiązań

### 3.1 Analiza problemu

### 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 [12] 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 [13] 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**

## Rozdział 4

# Przygotowanie infrastruktury informatycznej

Rozdziały dokumentujące pracę własną studenta: opisujące ideę, sposób lub metodę rozwiązania postawionego problemu oraz rozdziały opisujące techniczną stronę rozwiązania — dokumentacja techniczna, przeprowadzone testy, badania i uzyskane wyniki.

Praca musi zawierać elementy pracy własnej autora adekwatne do jego wiedzy praktycznej uzyskanej w okresie studiów. Za pracę własną autora można uznać np.: stworzenie aplikacji informatycznej lub jej fragmentu, zaproponowanie algorytmu rozwiązania problemu szczegółowego, przedstawienie projektu np. systemu informatycznego lub sieci komputerowej, analizę i ocenę nowych technologii lub rozwiązań informatycznych wykorzystywanych w przedsiębiorstwach, itp.

Autor powinien zadbać o właściwą dokumentację pracy własnej obejmującą specyfikację założeń i sposób realizacji poszczególnych zadań wraz z ich oceną i opisem napotkanych problemów. W przypadku prac o charakterze projektowo-implementacyjnym, ta część pracy jest zastępowana dokumentacją techniczną i użytkową systemu.

W pracy **nie należy zamieszczać całego kodu źródłowego** opracowanych programów. Kod źródłowy napisanych programów, wszelkie oprogramowanie wytworzone i wykorzystane w pracy, wyniki przeprowadzonych eksperymentów powinny być umieszczone np. na płycie CD, stanowiącej dodatek do pracy.

### Styl tekstu

Należy<sup>1</sup> stosować formę bezosobową, tj. *w pracy rozważono .....*, *w ramach pracy zaprojektowano ....*, a nie: *w pracy rozważyłem*, *w ramach pracy zaprojektowałem*. Odwołania do wcześniejszych fragmentów tekstu powinny mieć następującą postać: „Jak wspomniano wcześniej, ....”, „Jak wykazano powyżej ....”. Należy unikać długich zdań.

Niedopuszczalne są zwroty używane w języku potocznym. W pracy należy używać terminologii informatycznej, która ma sprecyzowaną treść i znaczenie.

Niedopuszczalne jest pisanie pracy metodą *cut&paste*, bo jest to plagiat i dowód intelektualnej indolencji autora. Dane zagadnienie należy opisać własnymi słowami. Zawsze trzeba powołać się na zewnętrzne źródła.

---

<sup>1</sup>Uwagi o stylu pochodzą częściowo ze stron prof. Macieja Drozdowskiego [17].

## 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 menedżer pakietów środowiska Node (NPM - Node Package Manager) dający możliwość zarządzania zainstalowanymi funkcjonalnościami w prosty i przejrzysty sposób.

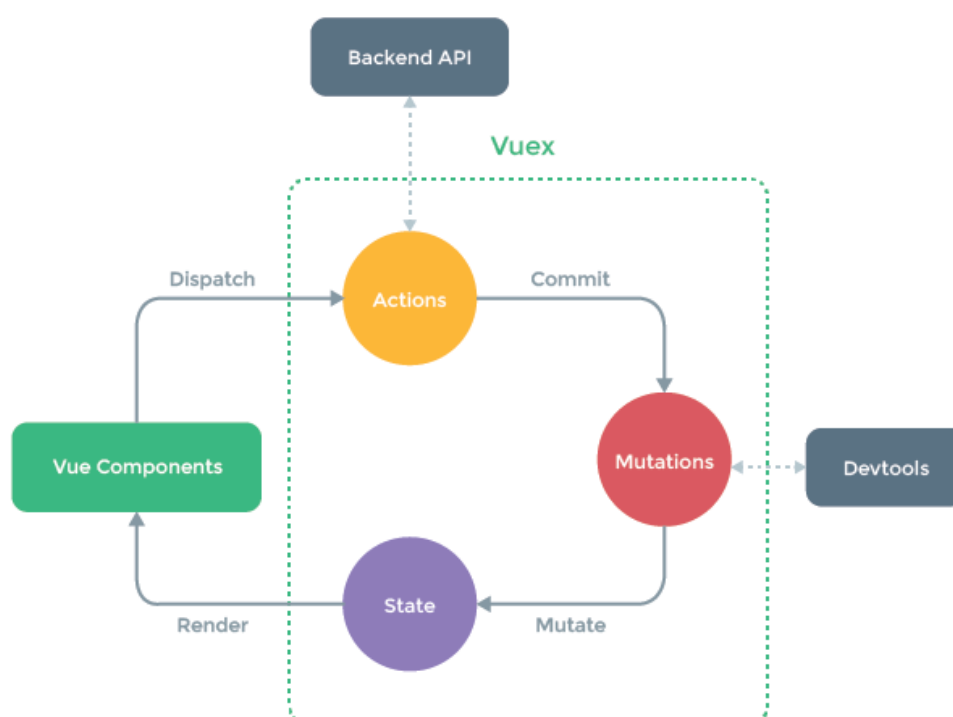
#### 5.1.2 Vue.js

Vue.js [15] to framework języka JavaScript służący do budowania interfejsów użytkownika. W stosunku do dwóch najpopularniejszych alternatyw - frameworków React oraz Angular - wyróżnia się prostotą, szybkością działania oraz niewielkim rozmiarem. Framework Vue.js został zaprojektowany tak, aby zapewnić jak największą elastyczność. Przy jego 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 [16] 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, 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.

#### 5.1.4 Jest

W projekcie wykorzystano framework testowy Jest [6] będący 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 przesyłania 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 [11] 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 [14] 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 frameworków, w tym Vue.js, najbardziej istotnym 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 framework taki 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.

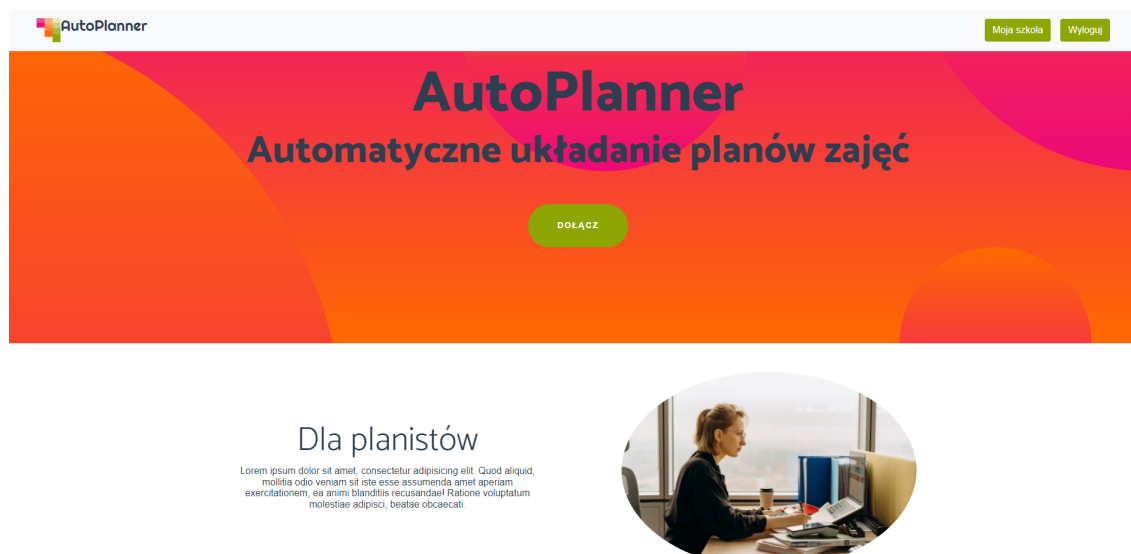
### 5.1.9 Bootstrap

Bootstrap [3] jest frameworkiem CSS (Cascading Style Sheets) upraszczającym 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 frameworku elementy strony internetowe 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

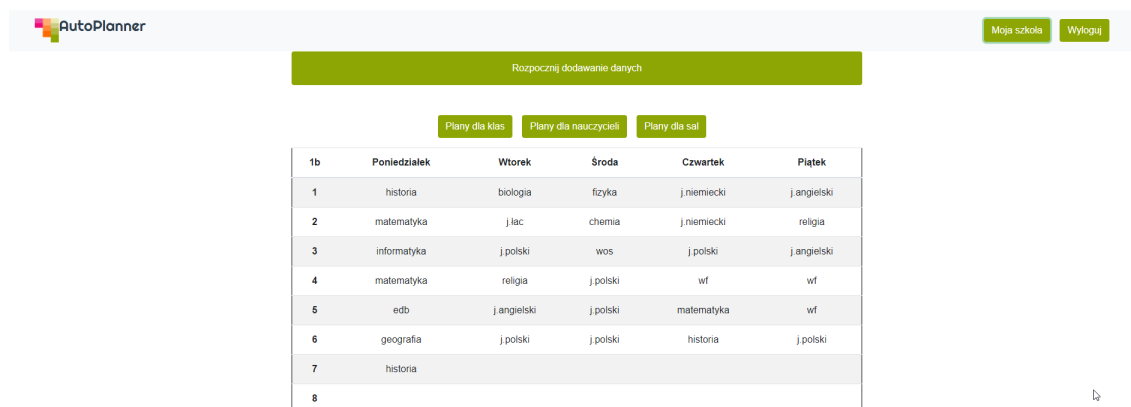
Strona główna (zob. rysunek 5.2) aplikacji została stworzona na bazie szablonu Bootstrap o nazwie One Page Wonder(źródło). Zawiera ona krótki opis aplikacji, a także korzyści płynących z wykorzystania jej dla planistów, nauczycieli oraz uczniów. Pasek menu znajdujący się zawsze na górze strony jest stałym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

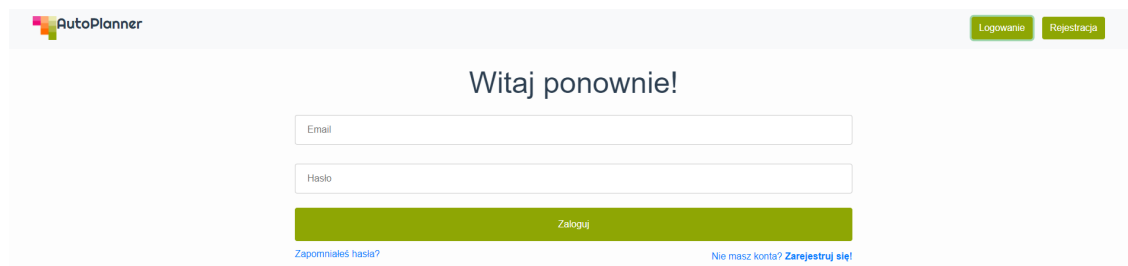


The screenshot shows the registration page of the AutoPlanner application. At the top left is the 'AutoPlanner' logo. At the top right are two buttons: 'Logowanie' and 'Rejestracja'. The main heading is 'Zarejestruj się'. Below it are three input fields: 'Email', 'Nazwa użytkownika', and 'Hasło'. At the bottom is a large green button labeled 'Zarejestruj'.

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.



The screenshot shows the login page of the AutoPlanner application. At the top left is the 'AutoPlanner' logo. At the top right are two buttons: 'Logowanie' and 'Rejestracja'. The main heading is 'Witaj ponownie!'. Below it are two input fields: 'Email' and 'Hasło'. At the bottom is a large green button labeled 'Zaloguj'. Below the button, there are two links: 'Zapomniałeś hasła?' on the left and 'Nie masz konta? Zarejestruj się!' on the right.

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"/>

[Wyslij](#)

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.

[Moja szkoła](#)
[Wyloguj](#)

## Krok 1 - Dodaj przedmioty

- biologia
- chemia
- fizyka
- j.polski
- matematyka
- religia
- wf

Nazwa przedmiotu

Dodaj

Przejdź dalej

Poprzedni krok

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.

The screenshot shows the 'Krok 2 - Dodaj nauczycieli' (Step 2 - Add teachers) form. On the left, there is a vertical list of teacher names: Janusz Walczuk, Krystyna Pawłowicz, Robert Lewandowski, Waldemar Klepski, Ksiądz Robak, Snoop Dog, and Albert Einstein. The main form area contains input fields for 'Imię i Nazwisko' (First and Last Name) and 'Email'. Below these is a section titled 'Prowadzone przedmioty' (Subjects taught) with a dropdown menu labeled '-- wybierz przedmiot --'. At the bottom of the form are three buttons: 'Dodaj' (Add), 'Przejdź dalej' (Go next), and 'Poprzedni krok' (Previous step). The top of the page features the 'AutoPlanner' logo and two buttons: 'Moja szkoła' (My school) and 'Wyloguj' (Logout).

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.

The screenshot shows the 'Krok 3 - Dodaj sale lekcyjne' (Step 3 - Add lecture halls) form. On the left, there is a vertical list of room numbers: 100, 101, 102, 103, 104, 105, and 200. The main form area contains an input field for 'Nazwa sali' (Room name). Below this is a checkbox labeled 'Preferowane przedmioty' (Preferred subjects). At the bottom of the form are three buttons: 'Dodaj' (Add), 'Przejdź dalej' (Go next), and 'Poprzedni krok' (Previous step). The top of the page features the 'AutoPlanner' logo and two buttons: 'Moja szkoła' (My school) and 'Wyloguj' (Logout).

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.

The screenshot displays the 'Krok 4 - Dodaj klasy' (Step 4 - Add classes) interface of the AutoPlanner application. On the left, there is a sidebar with four green buttons labeled '1A', '1B', '1C', and '1D'. The main area features a header with the 'AutoPlanner' logo and 'Moja szkoła' / 'Wyloguj' buttons. Below the header, the title 'Krok 4 - Dodaj klasy' is centered. The form includes a 'Nazwa klasy' input field. The 'Lista przedmiotów' (List of subjects) section contains a dropdown menu for selecting a subject, a 'Liczba godzin tygodniowo' (Weekly hours) input, a 'liczba godzin' (hours) input, and a checkbox for selecting a teacher. At the bottom of the form are three green buttons: 'Dodaj' (Add), 'Przejdź dalej' (Go next), and 'Poprzedni krok' (Previous step).

RYСУNEK 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 omówiony zostanie ich struktura 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.

The screenshot shows the 'Edycja danych nauczyciela' (Teacher Data Edit) interface. At the top, there is a header bar with the 'AutoPlanner' logo on the left and 'Moja szkoła' and 'Wyloguj' buttons on the right. The main title 'Edycja danych nauczyciela' is centered below the header. On the left side, there is a sidebar with a button labeled 'AA BB'. The main content area contains two input fields: the first for the name 'AA BB' and the second for the email 'AABB@CC.com'. Below these is a section titled 'Prowadzone przedmioty' (Subjects Taught) with a dropdown menu currently showing 'Matematyka'. Under the dropdown are two small buttons, '+' and '-'. At the bottom of the form are three large, dark blue buttons: 'Zapisz zmiany' (Save changes), 'Wróć bez zapisywania' (Return without saving), and 'Usuń nauczyciela' (Delete teacher).

RYSUNEK 5.11: Aplikacja internetowa - Widok edycji danych nauczyciela

## Rozdział 6

# Projekt i implementacja strony serwerowej opartej na architekturze REST w technologii Django

Rozdziały dokumentujące pracę własną studenta: opisujące ideę, sposób lub metodę rozwiązania postawionego problemu oraz rozdziały opisujące techniczną stronę rozwiązania — dokumentacja techniczna, przeprowadzone testy, badania i uzyskane wyniki.

Praca musi zawierać elementy pracy własnej autora adekwatne do jego wiedzy praktycznej uzyskanej w okresie studiów. Za pracę własną autora można uznać np.: stworzenie aplikacji informatycznej lub jej fragmentu, zaproponowanie algorytmu rozwiązania problemu szczegółowego, przedstawienie projektu np. systemu informatycznego lub sieci komputerowej, analizę i ocenę nowych technologii lub rozwiązań informatycznych wykorzystywanych w przedsiębiorstwach, itp.

Autor powinien zadbać o właściwą dokumentację pracy własnej obejmującą specyfikację założeń i sposób realizacji poszczególnych zadań wraz z ich oceną i opisem napotkanych problemów. W przypadku prac o charakterze projektowo-implementacyjnym, ta część pracy jest zastępowana dokumentacją techniczną i użytkową systemu.

W pracy **nie należy zamieszczać całego kodu źródłowego** opracowanych programów. Kod źródłowy napisanych programów, wszelkie oprogramowanie wytworzone i wykorzystane w pracy, wyniki przeprowadzonych eksperymentów powinny być umieszczone np. na płycie CD, stanowiącej dodatek do pracy.

### Styl tekstu

Należy<sup>1</sup> stosować formę bezosobową, tj. *w pracy rozważono .....*, *w ramach pracy zaprojektowano ....*, a nie: *w pracy rozważyłem*, *w ramach pracy zaprojektowałem*. Odwołania do wcześniejszych fragmentów tekstu powinny mieć następującą postać: „Jak wspomniano wcześniej, ....”, „Jak wykazano powyżej ....”. Należy unikać długich zdań.

Niedopuszczalne są zwroty używane w języku potocznym. W pracy należy używać terminologii informatycznej, która ma sprecyzowaną treść i znaczenie.

Niedopuszczalne jest pisanie pracy metodą *cut&paste*, bo jest to plagiat i dowód intelektualnej indolencji autora. Dane zagadnienie należy opisać własnymi słowami. Zawsze trzeba powołać się na zewnętrzne źródła.

---

<sup>1</sup>Uwagi o stylu pochodzą częściowo ze stron prof. Macieja Drozdowskiego [17].

## Rozdział 7

# Projekt i implementacja algorytmu generującego plan lekcji

Rozdziały dokumentujące pracę własną studenta: opisujące ideę, sposób lub metodę rozwiązania postawionego problemu oraz rozdziały opisujące techniczną stronę rozwiązania — dokumentacja techniczna, przeprowadzone testy, badania i uzyskane wyniki.

Praca musi zawierać elementy pracy własnej autora adekwatne do jego wiedzy praktycznej uzyskanej w okresie studiów. Za pracę własną autora można uznać np.: stworzenie aplikacji informatycznej lub jej fragmentu, zaproponowanie algorytmu rozwiązania problemu szczegółowego, przedstawienie projektu np. systemu informatycznego lub sieci komputerowej, analizę i ocenę nowych technologii lub rozwiązań informatycznych wykorzystywanych w przedsiębiorstwach, itp.

Autor powinien zadbać o właściwą dokumentację pracy własnej obejmującą specyfikację założeń i sposób realizacji poszczególnych zadań wraz z ich oceną i opisem napotkanych problemów. W przypadku prac o charakterze projektowo-implementacyjnym, ta część pracy jest zastępowana dokumentacją techniczną i użytkową systemu.

W pracy **nie należy zamieszczać całego kodu źródłowego** opracowanych programów. Kod źródłowy napisanych programów, wszelkie oprogramowanie wytworzone i wykorzystane w pracy, wyniki przeprowadzonych eksperymentów powinny być umieszczone np. na płycie CD, stanowiącej dodatek do pracy.

### Styl tekstu

Należy<sup>1</sup> stosować formę bezosobową, tj. *w pracy rozważono .....*, *w ramach pracy zaprojektowano ....*, a nie: *w pracy rozważyłem*, *w ramach pracy zaprojektowałem*. Odwołania do wcześniejszych fragmentów tekstu powinny mieć następującą postać: „Jak wspomniano wcześniej, ....”, „Jak wykazano powyżej ....”. Należy unikać długich zdań.

Niedopuszczalne są zwroty używane w języku potocznym. W pracy należy używać terminologii informatycznej, która ma sprecyzowaną treść i znaczenie.

Niedopuszczalne jest pisanie pracy metodą *cut&paste*, bo jest to plagiat i dowód intelektualnej indolencji autora. Dane zagadnienie należy opisać własnymi słowami. Zawsze trzeba powołać się na zewnętrzne źródła.

---

<sup>1</sup>Uwagi o stylu pochodzą częściowo ze stron prof. Macieja Drozdowskiego [17].

## Rozdział 8

# Testy

Rozdziały dokumentujące pracę własną studenta: opisujące ideę, sposób lub metodę rozwiązania postawionego problemu oraz rozdziały opisujące techniczną stronę rozwiązania — dokumentacja techniczna, przeprowadzone testy, badania i uzyskane wyniki.

Praca musi zawierać elementy pracy własnej autora adekwatne do jego wiedzy praktycznej uzyskanej w okresie studiów. Za pracę własną autora można uznać np.: stworzenie aplikacji informatycznej lub jej fragmentu, zaproponowanie algorytmu rozwiązania problemu szczegółowego, przedstawienie projektu np. systemu informatycznego lub sieci komputerowej, analizę i ocenę nowych technologii lub rozwiązań informatycznych wykorzystywanych w przedsiębiorstwach, itp.

Autor powinien zadbać o właściwą dokumentację pracy własnej obejmującą specyfikację założeń i sposób realizacji poszczególnych zadań wraz z ich oceną i opisem napotkanych problemów. W przypadku prac o charakterze projektowo-implementacyjnym, ta część pracy jest zastępowana dokumentacją techniczną i użytkową systemu.

W pracy **nie należy zamieszczać całego kodu źródłowego** opracowanych programów. Kod źródłowy napisanych programów, wszelkie oprogramowanie wytworzone i wykorzystane w pracy, wyniki przeprowadzonych eksperymentów powinny być umieszczone np. na płycie CD, stanowiącej dodatek do pracy.

## Styl tekstu

Należy<sup>1</sup> stosować formę bezosobową, tj. *w pracy rozważono .....*, *w ramach pracy zaprojektowano ....*, a nie: *w pracy rozważyłem*, *w ramach pracy zaprojektowałem*. Odwołania do wcześniejszych fragmentów tekstu powinny mieć następującą postać: „Jak wspomniano wcześniej, ....”, „Jak wykazano powyżej ....”. Należy unikać długich zdań.

Niedopuszczalne są zwroty używane w języku potocznym. W pracy należy używać terminologii informatycznej, która ma sprecyzowaną treść i znaczenie.

Niedopuszczalne jest pisanie pracy metodą *cut&paste*, bo jest to plagiat i dowód intelektualnej indolencji autora. Dane zagadnienie należy opisać własnymi słowami. Zawsze trzeba powołać się na zewnętrzne źródła.

---

<sup>1</sup>Uwagi o stylu pochodzą częściowo ze stron prof. Macieja Drozdowskiego [17].

## Rozdział 9

# Wnioski

Rozdziały dokumentujące pracę własną studenta: opisujące ideę, sposób lub metodę rozwiązania postawionego problemu oraz rozdziały opisujące techniczną stronę rozwiązania — dokumentacja techniczna, przeprowadzone testy, badania i uzyskane wyniki.

Praca musi zawierać elementy pracy własnej autora adekwatne do jego wiedzy praktycznej uzyskanej w okresie studiów. Za pracę własną autora można uznać np.: stworzenie aplikacji informatycznej lub jej fragmentu, zaproponowanie algorytmu rozwiązania problemu szczegółowego, przedstawienie projektu np. systemu informatycznego lub sieci komputerowej, analizę i ocenę nowych technologii lub rozwiązań informatycznych wykorzystywanych w przedsiębiorstwach, itp.

Autor powinien zadbać o właściwą dokumentację pracy własnej obejmującą specyfikację założeń i sposób realizacji poszczególnych zadań wraz z ich oceną i opisem napotkanych problemów. W przypadku prac o charakterze projektowo-implementacyjnym, ta część pracy jest zastępowana dokumentacją techniczną i użytkową systemu.

W pracy **nie należy zamieszczać całego kodu źródłowego** opracowanych programów. Kod źródłowy napisanych programów, wszelkie oprogramowanie wytworzone i wykorzystane w pracy, wyniki przeprowadzonych eksperymentów powinny być umieszczone np. na płycie CD, stanowiącej dodatek do pracy.

## Styl tekstu

Należy<sup>1</sup> stosować formę bezosobową, tj. *w pracy rozważono .....*, *w ramach pracy zaprojektowano ....*, a nie: *w pracy rozważyłem*, *w ramach pracy zaprojektowałem*. Odwołania do wcześniejszych fragmentów tekstu powinny mieć następującą postać: „Jak wspomniano wcześniej, ....”, „Jak wykazano powyżej ....”. Należy unikać długich zdań.

Niedopuszczalne są zwroty używane w języku potocznym. W pracy należy używać terminologii informatycznej, która ma sprecyzowaną treść i znaczenie.

Niedopuszczalne jest pisanie pracy metodą *cut&paste*, bo jest to plagiat i dowód intelektualnej indolencji autora. Dane zagadnienie należy opisać własnymi słowami. Zawsze trzeba powołać się na zewnętrzne źródła.

---

<sup>1</sup>Uwagi o stylu pochodzą częściowo ze stron prof. Macieja Drozdowskiego [17].



## Rozdział 10

# Zakończenie

Zakończenie pracy zwane również Uwagami końcowymi lub Podsumowaniem powinno zawierać ustosunkowanie się autora do zadań wskazanych we wstępie do pracy, a w szczególności do celu i zakresu pracy oraz porównanie ich z faktycznymi wynikami pracy. Podejście takie umożliwia jasne określenie stopnia realizacji założonych celów oraz zwrócenie uwagi na wyniki osiągnięte przez autora w ramach jego samodzielnej pracy.

Integralną częścią pracy są również dodatki, aneksy i załączniki zawierające stworzone w ramach pracy programy, aplikacje i projekty.

# Literatura

- [1] aSc TimeTables. [on-line] <https://www.asctimetables.com/#!/home>.
- [2] Axios. [on-line] <https://vuejs.org/v2/cookbook/using-axios-to-consume-apis.html>.
- [3] Bootstrap. [on-line] <https://getbootstrap.com/docs/5.1/getting-started/introduction/>.
- [4] Heuristic. [on-line] <https://www.khanacademy.org/computing/ap-computer-science-principles/algorithms-101/solving-hard-problems/a/using-heuristics>.
- [5] Heuristic. [on-line] <https://towardsdatascience.com/introduction-to-evolutionary-algorithms-a8594b484ac>.
- [6] Jest. [on-line] <https://vue-test-utils.vuejs.org/guides/>.
- [7] Json web tokens. [on-line] <https://jwt.io/introduction>.
- [8] Node.js. [on-line] <https://nodejs.org/en/about/>.
- [9] Np-hard problem. [on-line] <https://xlinux.nist.gov/dads/HTML/nphard.html>.
- [10] Optimization problem. [on-line] <https://xlinux.nist.gov/dads/HTML/optimization.html>.
- [11] Postman. [on-line] <https://www.postman.com/product/what-is-postman/>.
- [12] Prime Timetable. [on-line] <https://primetimetable.com/help/#overview&q>.
- [13] SuperSaaS. [on-line] [https://www.supersaas.com/info/doc/getting\\_started](https://www.supersaas.com/info/doc/getting_started).
- [14] Visual studio code. [on-line] <https://code.visualstudio.com/docs>.
- [15] Vue.js. [on-line] <https://vuejs.org/v2/guide/>.
- [16] Vuex. [on-line] <https://vuex.vuejs.org/>.
- [17] Maciej Drozdowski. Jak pisać prace dyplomowe – uwagi o formie. [on-line] [http://www.cs.put.poznan.pl/mdrozdowski/dyd/txt/jak\\_mgr.html](http://www.cs.put.poznan.pl/mdrozdowski/dyd/txt/jak_mgr.html), 2006.
- [18] Donald E. Knuth. *The T<sub>E</sub>Xbook*. Computers and Typesetting. Addison-Wesley, Reading, MA, USA, 1986.
- [19] Leslie Lamport. *L<sup>A</sup>T<sub>E</sub>X — A Document Preparation System — User’s Guide and Reference Manual*. Addison-Wesley, Reading, MA, USA, 1985.

## Dodatek A

# Składanie dokumentu w systemie L<sup>A</sup>T<sub>E</sub>X

W tym rozdziale znajduje się garść informacji o tym, jak poprawnie składać tekst pracy w systemie L<sup>A</sup>T<sub>E</sub>X wraz z przykładami, które mają służyć do przeklejania do własnych dokumentów.

### A.1 Struktura dokumentu

Praca składa się z rozdziałów (`chapter`) i podrozdziałów (`section`). Ewentualnie można również rozdziały zagnieżdzać (`subsection`, `subsubsection`), jednak nie powinno się wykraczać poza drugi poziom hierarchii (czyli `subsubsection`).

### A.2 Akapity i znaki specjalne

Akapity rozdziela się od siebie przynajmniej jedną pustą linią. Podstawowe instrukcje, które się przydają to *wyróżnienie pewnych słów*. Można również stosować **styl pogrubiony**, choć nie jest to generalnie zalecane.

Należy pamiętać o zasadach polskiej interpunkcji i ortografii. Po spójnikach jednoliterowych warto wstawić znak tyldy (`~`), który jest tak zwaną „twardą spacją” i powoduje, że wyrazy nią połączone nie będą rozdzielane na dwie linie tekstu.

Polskie znaki interpunkcyjne różnią się nieco od angielskich: to jest „polski”, a to jest “angielski”. W kodzie źródłowym tego tekstu będzie widać różnicę.

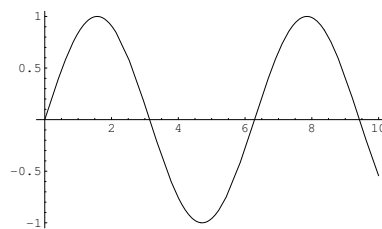
Proszę również zwrócić uwagę na znak myślnika, który może być pauzą „—” lub półpauzą: „-”. Należy stosować je konsekwentnie. Do łączenia wyrazów używamy zwykłego „-” (*północno-wschodni*), do myślników — pauzy lub półpauzy. Inne zasady interpunkcji i typografii można znaleźć w słownikach.

### A.3 Wypunktowania

Wypunktowanie z cyframi:

1. to jest punkt,
2. i to jest punkt,
3. a to jest ostatni punkt.

Po wypunktowaniach czasem nie warto wstawiać wcięcia akapitowego. Wtedy przydatne jest polecenie `noindent`. Wypunktowanie z kropkami (tzw. *bullet list*) wygląda tak:



RYSUNEK A.1: Wykres.

- to jest punkt,
- i to jest punkt,
- a to jest ostatni punkt.

Wypunktowania opisowe właściwie niewiele się różnią:

**elementA** to jest opis,

**elementB** i to jest opis,

**elementC** a to jest ostatni opis.

## A.4 Polecenia pakietu *ppfcmthesis*

Parę poleceń zostało zdefiniowanych aby uspołnić styl pracy. Są one przedstawione poniżej (oczywiście nie trzeba się do nich stosować).

**Makra zdefiniowane dla języka angielskiego.** Są nimi: `termdef` oraz `acronym`. Przykłady poniżej obrazują ich przewidywane użycie w tekście.

źródło	<code>we call this a \termdef{Database Management System} (\acronym{DBMS})</code>
docelowo	we call this a <i>Database Management System (DBMS)</i>

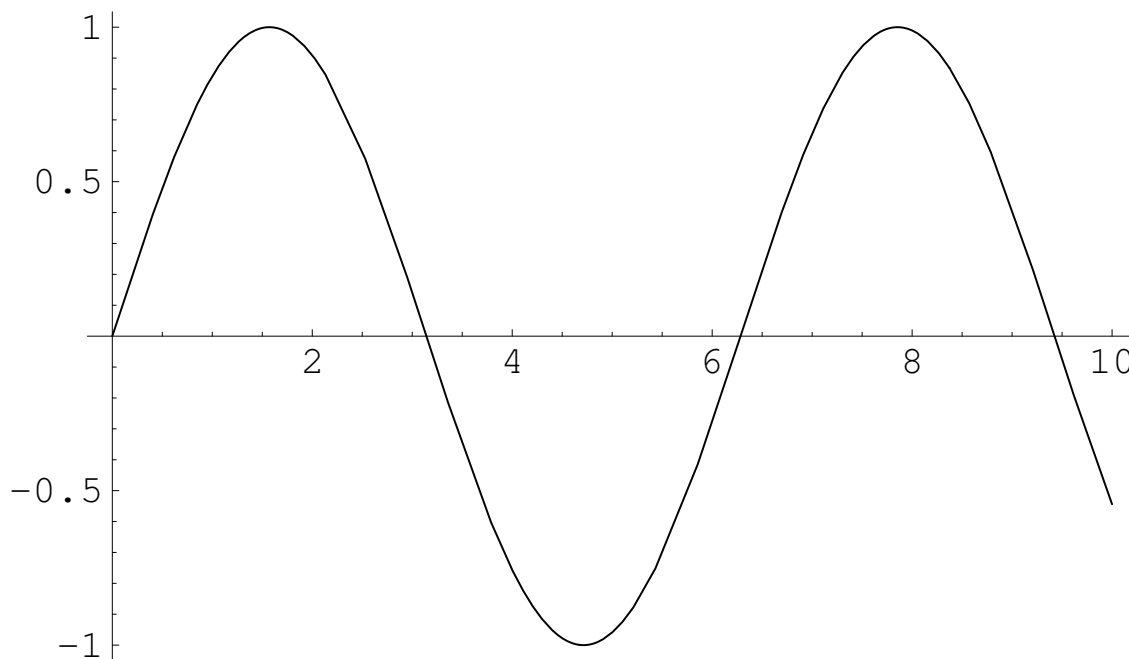
**Makra zdefiniowane dla języka polskiego.** Podobnie jak dla języka angielskiego zdefiniowano odpowiedniki polskie: `definicja`, `akronim` oraz `english` dla tłumaczeń angielskich terminów. Przykłady poniżej obrazują ich przewidywane użycie w tekście.

źródło	<code>nazywamy go \definicja{systemem zarządzania bazą danych} (\akronim{DBMS}, \english{Database Management System})</code>
docelowo	nazywamy go <i>systemem zarządzania bazą danych (DBMS, ang. Database Management System)</i>

## A.5 Rysunki

Wszystkie rysunki (w tym również diagramy, szkice i inne) osadzamy w środowisku `figure` i umieszczamy podpis *pod* rysunkiem, w formie elementu `caption`. Rysunki powinny zostać umieszczone u góry strony (osadzone bezpośrednio w treści strony zwykle utrudniają czytanie tekstu). Rysunek A.1 zawiera przykład pełnego osadzenia rysunku na stronie.

Styl FCMu to nieco inne nagłówki rysunków. Dostępne są one poleceniem `fcmfcaption` (zob. rysunek A.2).



**Rysunek A.2.** Ten sam wykres ale na szerokość tekstu. Formatowanie podpisu zgodne z wytycznymi FCMu.

### A.5.1 Tablice

Tablice to piękna rzecz, choć akurat ich umiejętne tworzenie w  $\text{\LaTeX}$ u nie jest łatwe. Jeśli tablica jest skomplikowana, to można ją na przykład wykonać w programie OpenOffice, a następnie wyeksportować jako plik *PDF*. W każdym przypadku tablice wstawia się podobnie jak rysunki, tylko że w środowisko `table`. Tradycja typograficzna sugeruje umieszczenie opisu tablicy, a więc elementu `caption` ponad jej treścią (inaczej niż przy rysunkach).

Tablica A.1 pokazuje pełen przykład.

TABLICA A.1: Przykładowa tabela. Styl opisu jest zgodny z rysunkami.

artykuł	cena [zł]
bułka	0,4
masło	2,5

Zasady FCMu sugerują nieco inne nagłówki tablic. Dostępne są one poleceniem `fcmtcaption` (zob. tablicę A.2).

**Tablica A.2**

Przykładowa tabela. Styl opisu jest zgodny z wytycznymi FCMu.

artykuł	cena [zł]
bułka	0,4
masło	2,5

### A.5.2 Checklista

- Znakiem myślnika jest w  $\text{\LaTeX}$ u dywiz pełen (`—`) albo półpauza (`-`), przykład: A niech to jasna cholera — wrzasnąłem.

- Połączenie między wyrazami to zwykły myślnik, przykład: północno-zachodni
- Sprawdź czy tytuł pracy ma maksymalnie dwa wiersze i czy stanowią one pełne frazy (czy nie ma przeniesienia bez sensu).
- Sprawdź ostrzeżenia o 'overfull' i 'underfull' boxes. Niektóre z nich można zignorować (spójrz na wynik formatowania), niektóre trzeba poprawić; czasem przeformułować zdanie.  
item Przypisy stawia się wewnątrz zdań lub za kropką, przykład: Footnote is added after a comma.<sup>1</sup>
- Nie używaj przypisów zbyt często. Zobacz, czy nie lepiej będzie zintegrować przypis z tekstem.
- Tytuły tabel, rysunków powinny kończyć się kropką.
- Nie używaj modyfikatora [h] (here) do rysunków i tabel. Rysunki i tabele powinny być justowane do góry strony lub na stronie osobnej.
- Wyróżnienie w tekście to polecenie *wyraz*, nie należy używać czcionki pogrubionej (która wystaje wizualnie z tekstu i rozprasza).
- Nazwy plików, katalogów, ścieżek, zmiennych środowiskowych, klas i metod formatujemy poleceniem `plik_o_pewnej_nazwie`.
- Po ostatniej zmianie do treści, sprawdź i przenieś wiszące spójniki wstawiając przed nie znak tyldy (twardej spacji), przykład: Ala i kotek nie lubią mleczka, a Stasiu lubi.
- Za i.e. (id est) i e.g. (exempli gratia) stawia się zwyczajowo przecinek w typografii amerykańskiej.
- Przed i za pełną pauza nie ma zwyczajowo spacji w typografii amerykańskiej, przykład: Darn, this looks good—said Mary.
- Zamykający cudzysłów oraz footnote wychodzą za ostatni znak interpunkcji w typografii amerykańskiej, przykłady: It can be called a “curiosity,” but it’s actually normal. Footnote is added after a comma.<sup>2</sup>
- Odwołania do tabel i rysunków zawsze z wielkiej litery, przykład: In Figure A.1 we illustrated XXX and in Table A.1 we show detailed data.

## A.6 Literatura i materiały dodatkowe

Materiałów jest mnóstwo. Oto parę z nich:

- *The Not So Short Introduction...*, która posiada również tłumaczenie w języku polskim.  
<http://www.ctan.org/tex-archive/info/lshort/english/lshort.pdf>
- Klasy stylu memoir posiadają bardzo wiele informacji o składzie tekstów anglosaskich oraz sposoby dostosowania L<sup>A</sup>T<sub>E</sub>Xa do własnych potrzeb.  
<http://www.ctan.org/tex-archive/macros/latex/contrib/memoir/memman.pdf>

---

<sup>1</sup>Here is a footnote.

<sup>2</sup>Here is a footnote.

- Nasza grupa dyskusyjna i repozytorium Git są również dobrym miejscem aby zapytać (lub sprawdzić czy pytanie nie zostało już zadane).  
`https://github.com/politechnika/put-latex`
- Dla łaknących więcej wiedzy o systemie LaTeX podstawowym źródłem informacji jest książka Lamporta [19]. Prawdziwy *hardcore* to oczywiście *The T<sub>E</sub>Xbook* profesora Knutha [18].



© 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.