POLITECHNIKA POZNAŃSKA

WYDZIAŁ INFORMATYKI I TELEKOMUNIKACJI
Instytut Informatyki

Praca dyplomowa licencjacka

INTERNETOWE ŚRODOWISKO METOD WSPOMAGANIA DECYZJI

Daniel Paszek, 148194
Klaudia Kowalska, 148184
Michał Zieliński, 148064
Aleksander Malcew, 147955

Promotor
dr inż. Bartłomiej Prędki

POZNAŃ 2024

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

Spis treści

Rozdział 1

Wstęp

Rozwój internetu oraz technik przetwarzania dużych zbiorów danych spowodował wzrost zapotrzebowania przedsiębiorstw na aplikacje internetowe i mobilne operujące na olbrzymich zestawach danych, zdolnych serwować ich klientom najlepiej dopasowane produkty. Rozpatrywana problematyka wymaga nie tylko pokaźnej mocy obliczeniowej, ale i inteligentnych algorytmów potrafiących dostosować się do przetwarzanych przez nie zbiorów danych i preferencji klientów. Posiadając te informacje, algorytmy powinny być w stanie zaprezentować użytkownikom końcowym rankingi odwzorowujące ich upodobania, na przykład "najstraszniejsze filmy i seriale z gatunku horrorów dla fanów dreszczyku emocji".

Nie tylko firmy i korporacje zmagają się jednak z tą problematyką. Z pewnością można założyć, że każdy użytkownik internetu spotkał się choć raz z problemem wyboru najlepszego produktu spośród setek dostępnych artykułów, czy porywającej książki z gatunku fantastyki.

W takich sytuacjach przydatne okazują się inteligentne systemy wspomagania decyzji, a w szczególności systemy wielokryterialnego wspomagania decyzji (w skrócie WWD), które specjalizują się we wspieraniu decyzji przy uwzględnieniu wielu kryteriów i dużej liczby analizowanych wariantów.

1.1 Cel i założenia projektu

W niniejszej pracy podjęto się próby utworzenia aplikacji internetowej do użytku w realnych zastosowaniach wspomagania decyzji, jak i do edukacji uniwersyteckiej. System posiada zaimplementowane metody WWD rozwiązujące różne problematyki wspomagania decyzji. Głównym założeniem projektu jest efektywny i intuicyjny interfejs użytkownika, który skutecznie wspomaga proces analizy działania zaimplementowanych metod, jak i wspomaga operacje porównywania analizowanych przez system wariantów. Ważnym aspektem jest także przemyślana implementacja systemu, ułatwiająca dodawanie nowych metod w sposób modularny.

Zaprojektowany system nadaje się do użytku lokalnego, jak i do szeroko pojętego udostępniania na dedykowanych serwerach. Aplikacja wspiera autoryzację użytkowników, udostępnianie projektów w trybie przeglądania oraz edycji, operacje wyjścia/wejścia (import, eksport danych, generowanie raportów), bogatą wizualizację działania metod.

1.2 Zakres projektu i zastosowane technologie

Opisany w niniejszej pracy projekt został zrealizowany przy użyciu nowoczesnych technik i narzędzi wykorzystywanych w inżynierii oprogramowania. W poniższym podrozdziale przedstawione zostały niektóre z najważniejszych programów i technologii wykorzystywanych w czasie tworzenia systemu.

1.2.1 Użyte technologie informatyczne

Architektura mikroserwisów, *Docker*

Wiele współczesnych systemów informatycznych bazuje na architekturze **mikroserwisów**. Struktura ta pozwala na odseparowanie warstw programu oraz zabezpieczenie systemu przed atakami. W niniejszym projekcie została zastosowana szczególna wersja architektury mikroserwisowej, zwaną **konteneryzacją**. Technologia ta wykorzystuje wirtualizację na poziomie systemu operacyjnego oraz system operacyjny Linux - te dwie cechy zapewniają wysoką wydajność, konfigurowalność, kompatybilność międzyplatformową i bezpieczeństwo tworzonego rozwiązania.

Aplikacja została podzielona na [wstawić finalną liczbę kontenerów] kontenery, hostowane przy użyciu platformy *Docker*:

- Silnik z metodami WWD,
- Aplikacja internetowa,
- Silnik bazodanowy.

Języki programowania, frameworki, bazy danych

Konteneryzacja posiada również jeszcze jedną, ogromną zaletę - zastosowanie mikroserwisów pozwala na dowolność w wykorzystaniu przeróżnych języków programowania i bibliotek programistycznych. Na wymienionych powyżej kontenerach zostały użyte różne języki oraz frameworki, z których najważniejszymi są:

- ***PHP*** - język skryptowy, szeroko wykorzystywany w budowaniu stron internetowych i aplikacji webowych. W ramach języka użyte zostały:
  - ***Laravel*** - Framework implementujący model MVC (z ang. *Model-View-Controller*),
  - ***Livewire*** - Full-stackowy framework Laravela, służący do budowy interfejsów użytkownika,
  - ***Filament*** - Biblioteka wzorców i komponentów, wykorzystująca framework *Livewire*
- ***Kotlin*** - międzyplatformowy, statycznie typowany język programowania ogólnego przeznaczenia, stworzony i rozwijany przez firmę *JetBrains*. Język ten działa na maszynie wirtualnej *Javy*.
- ***Java*** - obiektowy, międzyplatformowy język programowania, aktualnie rozwijany przez korporację *Oracle*. Głównym frameworkiem wykorzystywanym w czasie tworzenia projektu był:
  - ***Spring Boot*** - narzędzie umożliwiające tworzenie mikrousług minimalnym nakładem pracy wejścia i konfiguracji.

Technologią niezbędną w każdym projekcie informatycznym, tworzonym na środowisko webowe, jest silnik bazodanowy. Z uwagi na silną zależność pomiędzy danymi oraz niezbędną międzyplatformowość, w ramach projektu użyty został silnik zarządzania relacyjnymi bazami danych *PostgreSQL*. Silnik ten jest natywnie wspierany przez rodzinę systemów operacyjnych *Linux* i pozwala na łatwe zarządzanie dużymi wolumenami składowanych danych.

1.2.2 Środowisko deweloperskie

Ze względu na skalę projektu oraz mnogość użytych technologii i języków programowania, bardzo ważnym narzędziem używanym w czasie tworzenia systemu było Zintegrowane Środowisko Programistyczne (ang. *IDE - integrated development environment*). Najczęściej używanymi *IDE* były programy oferowane przez firmę *JetBrains*, lidera tego typu rozwiązań na rynku informatycznym. Szczególnie przydatnymi *IDE* były ***PhpStorm*** (użyte przy pisaniu w języku PHP), ***IntelliJ IDEA*** (użyte przy pisaniu w języku Kotlin), ***DataGrip*** (budowa i zarządzanie bazą danych)

1.2.3 Komunikacja i praca równoległa

System kontroli wersji *GIT*, platforma *Github*

Niechybnie najważniejszym narzędziem użytym w czasie implementacji systemu jest system kontroli wersji *GIT*. Systemem kontroli wersji nazywamy narzędzie wspomagające proces tworzenia oprogramowania w sposób współbieżny przez wielu programistów jednocześnie. Narzędzie *GIT* umożliwia również tworzenie historii zmian, przydatnej w przypadku implementacji nowych, eksperymentalnych modułów projektu.

Kod źródłowy projektu umieszczony został w repozytorium platformy *Github*. Platforma ta poza udostępnianiem kodu zawiera wiele narzędzi, pomocnych w czasie realizowania projektu. Ważniejszymi z nich jest tablica zadań kanban, narzędzia automatycznego wdrażania zmian *Github Actions*, czy *Github Issues* umożliwiające planowanie zadań oraz recenzowanie zmian w kodzie.

Tablica zadań *Kanban*

Metodyka wykorzystywana w inżynierii oprogramowania do zarządzania zasobami i zadaniami projektowymi. Metodę zaadaptowało wiele platform i narzędzi zarządzania projektami, w ramach projektu użyte zostało narzędzie *Github Projects*, hostowane na platformie *Github*.

Platforma *Discord*

Chmurowa usługa internetowa pozwalająca na hostowanie serwerów głosowych oraz czatu, głównie wykorzystywana przez graczy oraz twórców treści. Discord używany jest również przez profesjonalne firmy programistyczne, jako komunikator internetowy.

1.3 Układ pracy

TODO

Rozdział 2

Podstawy teoretyczne

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

Rozwinięcie

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[1] 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.

[1] Uwagi o stylu pochodzą częściowo ze stron prof. Macieja Drozdowskiego [1].

Rozdział 4

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] Maciej Drozdowski. Jak pisać prace dyplomowe – uwagi o formie. [on-line] `http://www.cs.put.poznan.pl/mdrozdowski/dyd/txt/jak_mgr.html`, 2006.

[2] Donald E. Knuth. *The TEXbook*. Computers and Typesetting. Addison-Wesley, Reading, MA, USA, 1986.

[3] Leslie Lamport. *LATEX — A Document Preparation System — User's Guide and Reference Manual*. Addison-Wesley, Reading, MA, USA, 1985.

Dodatek A

Składanie dokumentu w systemie LaTeX

W tym rozdziale znajduje się garść informacji o tym, jak poprawnie składać tekst pracy w systemie LaTeX 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 zagnieżdżać (`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.

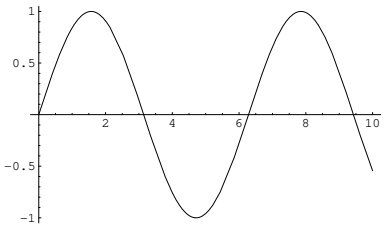
Proszę również zwrócić uwagę na znak myślnika, który może być pauzą „—" lub półpauzą: „–". Należy stosować je konsekwentnie. Do łączenia wyrazów używamy zwykłego „-" (*północno-wschodni*), do myślników — pauzy lub półpauzy. Inne zasady interpunkcji i typografii można znaleźć w słownikach.

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 uspójnić 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	`we call this a \termdef{Database Management System} (\acronym{DBMS})`
docelowo	we call this a *Database Management System* (*DBMS*)

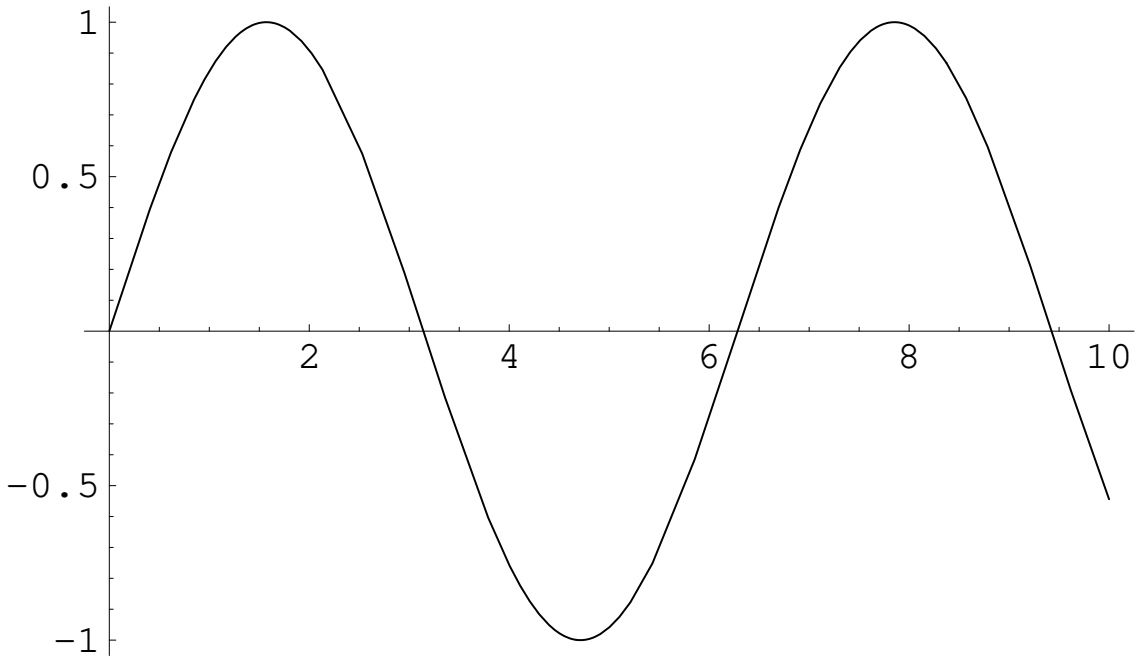
Makra zdefiniowane dla języka polskiego. Podobnie jak dla języka angielskiego zdefiniowano odpowiedniki polskie: `definicja`, `akronim` oraz `english` dla tłumaczeń angielskich terminów. Przykłady poniżej obrazują ich przewidywane użycie w tekście.

źródło	`nazywamy go \definicja{systemem zarządzania bazą danych} (\akronim{DBMS}, \english{Database Management System})`
docelowo	nazywamy go *systemem zarządzania bazą danych* (*DBMS*, ang. *Database Management System*)

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. Dostepne 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 LaTeXu 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.

Tabela 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. Dostepne są one poleceniem `fcmtcaption` (zob. tablicę A.2).

Tabela 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 LaTeXu dywiz pełen (—) albo półpauza (–), przykład: A niech to jasna cholera — wrzasnąłem.

- Połączenie między wyrazami to zwykły myślnik, przykład: północno-zachodni
- Sprawdź czy tutuł pracy ma maksymalnie dwa wiersze i czy stanowią one pełne frazy (czy nie ma przeniesienia bez sensu).
- Sprawdź ostrzeżenia o 'overfull' i 'underful' 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.[1]
- 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.[2]
- 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 LaTeXa do własnych potrzeb.
  `http://www.ctan.org/tex-archive/macros/latex/contrib/memoir/memman.pdf`
- 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`

[1] Here is a footnote.
[2] Here is a footnote.

- Dla łaknących więcej wiedzy o systemie LaTeX podstawowym źródłem informacji jest książka Lamporta [3]. Prawdziwy *hardcore* to oczywiście *The TEXbook* profesora Knutha [2].

Instytut Informatyki, Wydział Informatyki i Telekomunikacji
Politechnika Poznańska

Skład przy użyciu systemu LaTeX na platformie Overleaf.