POLITECHNIKA POZNAŃSKA
WYDZIAŁ ELEKTRYCZNY, INFORMATYKA
SEMESTR VI, GRUPA BSI-2

Podstawy Teleinformatyki
WebScrapper / Metawyszukiwarka

Paweł Soja
Numer indeksu: 122031
pawel.soja@student.put.poznan.pl

Krzysztof Łuczak
Numer indeksu: 122008
krzysztof.t.luczak@student.put.poznan.pl

Dawid Wiktorski
Numer indeksu: 122056
dawid.wiktorski@student.put.poznan.pl

Poznań, 31 maja 2017

Spis treści

Spis tablic

1 Opis i uzasadnienie wyboru tematu

Celem projektu jest zbudowanie platformy do zbierania i prezentowania danych z różnych stron internetowych. Platforma składa się z serwisu internetowego prezentującego dane użytkownikom zalogowanym oraz z aplikacji zbierających te dane.

Na potrzeby tego projektu i dokumentacji utworzone zostało pojęcie 'scrapowanie', które oznaczać będzie zbieranie danych ze stron internetowych poprzez odwiedzenie jej i zapisanie wybranych informacji do bazy danych.

1.1 Ogólny proces zbierania i przetwarzania danych

Informacje zbierane są przez tzw. scrapery, a następnie zapisywane w bazie danych. Scraper zbiera tylko te dane, które zostaną ustalone przez programistę. Dalsze filtrowanie odbywa się na poziomie aplikacji internetowej w oparciu o profil użytkownika lub podane parametry. Wyszukiwanie wykonywane jest w bazie danych. Dane zapisywane w bazie usuwane są po ustalonym czasie. Dlatego też aplikacja umożliwia filtrowanie wstecz, ale tylko do pewnej granicy. Dane zapisane w bazie można określić jako 'newsy'. Jeżeli informacja zostaje usunięta to znaczy, że jest już nieaktualna. Dzięki takiemu systemowi gromadzenia danych, aplikacja jest w stanie serwować użytkownikom najnowsze materiały, przy jednoczesnym zachowaniu wydajności filtrowania różnych źródeł. Z założenia użytkownik regularnie korzysta z aplikacji.

1.2 Uzasadnienie wyboru tematu

Temat wybraliśmy, ponieważ interesuje nas dziedzina przetwarzania danych. Chcielibyśmy poznać technologie scrapowania, parsowania stron internetowych oraz język Python, framework Django i technologie front-endowe tj. HTML5, Javascript. Jednocześnie nie znaleźliśmy zadowalającego nas serwisu, który udostępniałby takie usługi, dlatego sami zdecydowaliśmy zrobić swój.

2 Organizacja pracy

Przy pracy nad projektem, korzystano z repozytorium GitHub. Link do repozytorium: WebSrapper/Metawyszukiwarka

2.1 Harmonogram prac

Orientacyjny harmonogram prac został przedstawiony w tablicy 1. Wyszczególniono zadania oraz osobę/osoby zajmujące się danym fragmentem projektu.

Tablica 1: Harmonogram prac

Lp.	Opis	Miesiąc
1.	Wybór technologii, modułów, podział pracy	Marzec
2.	Wstępna dokumentacja, planowanie serwisu	Kwiecień
3.	Zapoznanie z technologią, testy bibliotek	Kwiecień
4.	Baza danych, pierwszy moduł, interfejs	Maj
5.	Kolejne moduły	Maj
6.	Testowanie serwisu, poprawki	Czerwiec
7.	Zakończenie prac nad serwisem	Czerwiec

2.2 Podział prac pomiędzy członków zespołu

W tablicy 2 przedstawiono podział prac pomiędzy członków zespołu.

Tablica 2: Podział prac

Lp.	Opis	Osoby
1.	Baza danych	Wszyscy
2.	Projekt interfejsu	Paweł Soja
3.	Front-end serwisu	Paweł Soja
4.	Back-end serwisu	Krzysztof Łuczak, Dawid Wiktorski
5.	Moduł I	Krzysztof Łuczak
6.	Moduł II	Dawid Wiktorski
7.	Moduł III	Dawid Wiktorski
8.	Moduł IV	Dawid Wiktorski
9.	Moduł V	Dawid Wiktorski
10.	Testowanie	Wszyscy

2.3 Środowisko pracy

- IDE PyCharm,
- TeXstudio,
- przeglądarki internetowe: Google Chrome oraz Mozilla Firefox.

3 Wymagania

3.1 Opis funkcjonalności

Aktorzy systemu:

- użytkownik
  - użytkownik zalogowany - posiada prawa do użytkowania serwisu,
  - użytkownik niezalogowany - może dokonać rejestracji,
  - administrator - zarządza serwisem,
- aplikacja internetowa - prezentuje dane,
- moduł zbierający dane (scraper) - zbiera i przetwarza dane.

Wymagania funkcjonalne systemu scharakteryzowano w tablicy 3.

Tablica 3: Funkcjonalności

Funkcja	Opis	Aktorzy
Przeglądanie strony głównej	Możliwość przeglądania strony głównej serwisu.	Użytkownicy
Rejestracja	Możliwość zarejestrowania konta w serwisie.	Użytkownik niezalogowany
Potwierdzenie rejestracji, zmiany hasła lub zmiany adresu e-mail konta	Możliwość potwierdzenia rejestracji, zmiany hasła lub zmiany adresu e-mail konta poprzez kliknięcie link aktywacyjny wysłany pocztą elektroniczną.	Użytkownik niezalogowany
Logowanie	Możliwość logowania się do serwisu.	Użytkownik niezalogowany
Wylogowanie	Możliwość wylogowania się z serwisu.	Użytkownik zalogowany, administrator
Zmiana hasła do konta	Możliwość zmiany hasła do aktywnego konta.	Użytkownik zalogowany, administrator

Tablica 3 – *Kontynuacja*

Funkcja	Opis	Aktorzy
Zmiana adresu e-mail konta	Możliwość zmiany adresu e-mail konta.	Użytkownik zalogowany, administrator
Ustawienie profilu źródeł	Możliwość wybrania źródeł, z których otrzymywane będą informacje.	Użytkownik zalogowany, administrator
Ustawienie profilu tagów	Możliwość wybrania tagów, na podstawie których filtrowane będą informacje.	Użytkownik zalogowany, administrator
Ustawienie filtra daty	Możliwość wybrania przedziału czasowego, na podstawie którego filtrowane będą informacje.	Użytkownik zalogowany, administrator
Zbieranie danych ze strony i parsowanie ich	Scraper zbiera dane ze strony, parsuje je oraz zapisuje do bazy danych. Jeden scraper zbiera dane z jednej strony.	Scraper

3.2 Wymagania pozafunkcjonalne

- zainstalowany interpreter języka Python w wersji 3.5 lub wyższej,
- zainstalowana biblioteka "BeautifulSoup",
- język interfejsu użytkownika: polski,
- bezpieczne przechowywanie haseł w formie zahaszowanej.

4 Wybrane technologie i uzasadnienie

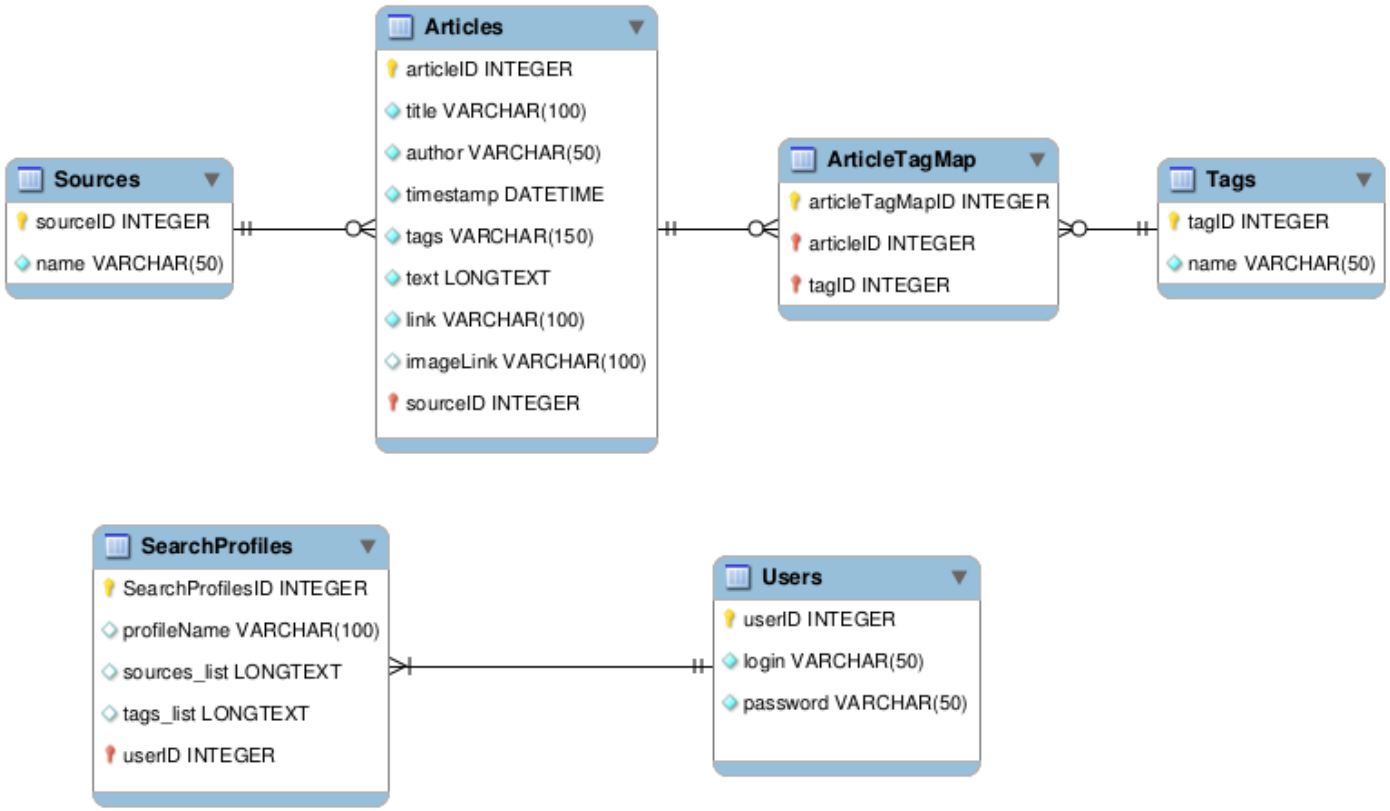
- Back-end - `Python, Django`
  - stosunkowo krótki czas tworzenia aplikacji przy jednoczesnym zachowaniu:
    * pełnej funkcjonalości,
    * stabilności,
    * wydajności
- Front-end - `HTML5, Javascript`
  - uniwersalna technologia, która jest wspierana przez wszystkie popularne przeglądarki internetowe
- Baza danych - `SQLite`
  - prosta integracja z językiem Python,
  - dobra we wstępnej fazie projektu

4.1 Biblioteka BeautifulSoup

Bibliotek do wyciągania danych ze stron internetowych jest wiele, są to między innymi: `Scrapy`, `BeautifulSoup`, `Urllib2`, `MarkupSafe` oraz `feedparser`. W projekcie wykorzystaliśmy bibliotekę BeautifulSoup, która przeznaczona jest dla języka Python. Umożliwia ona wyciąganie danych z plików HTML oraz XML. BeautifulSoup w wersji 4 współracuje z takimi parserami jak: `Python html.parser`, `lxml HTML parser`, `lxml XML parser` i `htm5lib`. Po przeprowadzonych testach, lxml HTML parser okazał się najszybszym oraz najbardziej niezawodym parserem spośród wyżej wymienionych. Dużą zaletą biblioteki BeautifulSoup jest łatwa implemetacja w architekturze modułowej.

5 Architektura rozwiązania

W projekcie została wykorzystana relacyjna baza danych SQLite. Na rysunku 1 przedstawiono schemat relayjny bazy danych.



Rysunek 1: Schemat bazy danych

W tablicy 4 scharakteryzowano bazę danych.

Tablica 4: Opis bazy danych

Tabela	Opis
Articles	Zawiera wszystkie sparsowane strony.
Tags	Zawiera wszystkie dostępne tagi. Dodanie nowego taga odbywa się automatycznie, gdy scraper podczas parsowania wykryje, że danego taga jeszcze nie ma w bazie.
ArticleTagMap	Łączy daną stronę z odpowiednim tagiem.

Tablica 4 – *Kontynuacja*

Tabela	Opis
Sources	Zawiera wszystkie dostępne źródła, czyli strony internetowe, z których zbieramy dane. Dodanie odbywa się ręcznie. Administrator musi napisać moduł dla danej strony.
Users	Zawiera wszystkich użytkowników serwisu.
TagsProfile	Łączy użytkownika z tagami, które wybrał.
SourceProfile	Łączy użytkownika z źródłami danych, które wybrał.

6 Interesujące problemy i ich rozwiązania

Podczas implementacji modułów, w każdym z nich, spotkano się z problemem wyciągania potrzebnych danych ze stron internetowych. Przy parsowaniu stron, najczęściej nieznajdywanym elementem w artykule okazał się link do obrazka. Rozwiązaniem problemu było najpierw ustawienie domyślego obrazka dla danego modułu oraz wykorzystanie go, gdy parser nie poradził sobie ze znalezieniem obrazka w artykule.

6.1 Moduł scrapujący sekurak.pl

Serwis sekurak.pl zawiera dwa rodzaje artykułów, które pobieramy. Pierwszy rodzaj znajduje się w kategorii "teksty", a drugi "w biegu". Scraper został podzielony na dwa wątki, każdy dla jednej z tych kategorii. To spowodowało przyspieszenie procesu zbierania o około 50%. Algorytm sekwencyjny dla jednej strony artykułów wykonywał się około 24 s. Algorytm wielowątkowy około 14 s.

6.2 Moduł scrapujący altcontroldelete.pl

W serwisie altcontroldelete.pl data publikacji artykułu może być podana w języku polskim lub angielskim. Rozwiązaniem problemu było utworzenie dwóch słowników, które w odpowiedni sposób zinterpretują podaną datę.

7 Opis stron internetowych, z których zbierane są informacje

Strony internetowe, z których zbierane są dane to:

- www.sekurak.pl,
- www.dobreprogramy.pl/Blog.html
- www.niebezpiecznik.pl
- www.pclab.pl/news.html
- www.altcontroldelete.pl

Poniżej w podrozdziałach przedstawiono dane, które są zbierane z każdej ze stron internetowych.

7.1 sekurak.pl

W tablicy 5 pokazano dane, które są zbierane ze strony sekurak.pl

Tablica 5: Parametry artykułów - sekurak.pl

Tytuł	Data opublikowania	Tagi	Obrazek	Fragment tekstu	Link

7.2 dobreprogramy.pl/Blog.html

W tablicy 6 pokazano dane, które są zbierane ze strony dobreprogramy.pl/Blog.html

Tablica 6: Parametry artykułów - dobreprogramy.pl

Tytuł	Data opublikowania	Tagi	Autor	Fragment tekstu	Link

7.3 niebezpiecznik.pl

W tablicy 7 pokazano dane, które są zbierane ze strony niebezpiecznik.pl

Tablica 7: Parametry artykułów - niebezpiecznik.pl

Tytuł	Data opublikowania	Tagi	Autor	Obrazek	Fragment tekstu	Link

7.4 pclab.pl/news.html

W tablicy 8 pokazano dane, które są zbierane ze strony pclab.pl/news.html

Tablica 8: Parametry artykułów - pclab.pl/news.html

Tytuł	Data opublikowania	Tagi	Autor	Obrazek	Fragment tekstu	Link

7.5 altcontroldelete.pl

W tablicy 9 pokazano dane, które są zbierane ze strony altcontroldelete.pl

Tablica 9: Parametry artykułów - altcontroldelete.pl

Tytuł	Data opublikowania	Tagi	Autor	Obrazek	Fragment tekstu	Link

8 Instrukcja użytkowania aplikacji