



Kierunki Rozwoju Oprogramowania

Krzysztof Kopeć

Cel projektu

Przeglądanie różnych stron internetowych w poszukiwaniu interesujących informacji bywa uciążliwe. Trzeba pamiętać które dane są dostępne w jakim serwisie oraz jak do nich dotrzeć. W niektórych przypadkach interfejs strony jest nieintuicyjny, albo niezbyt przystępny, co utrudnia odbiór informacji.

Celem projektu było stworzenie aplikacji webowej agregującej interesujące treści za pomocą techniki zwanej webscrappingiem. Zdecydowano się na przedstawienie zestawiania dotyczącego kwestii finansowych, tj. notowań na giełdzie, kursów walut oraz bieżącej sytuacji na rynku kryptowalut.

Założeniem było napisanie rozwiązania które prezentuje dane w zuniifikowany i przejrzysty sposób. Zgromadzenie danych z wielu źródeł i wyświetlenie ich w podobny sposób może ułatwić ich odbiór oraz usprawnić cały proces ich śledzenia. Dzięki temu użytkownik musi pamiętać tylko jeden adres, a intuicyjny interfejs usprawnia proces nawigacji do interesujących treści.

Aplikacja

Aplikacja serwerowa

Do napisania usługi serwerowej wykorzystano język python oraz framework tornado. Zdecydowano się na taki zestaw narzędzi z uwagi na jego prostotę, dostępność niezbędnych funkcjonalności oraz elastyczność. Z uwagi na sposób działania aplikacji, nie było potrzeby instalowania bardziej kompleksowych rozwiązań.

Po uruchomieniu skryptu inicjalizowany jest serwer HTTP. Generowany jest routing aplikacji i główna pętla nasłuchująca przychodzących połączeń. Oprócz tego pobierane są także dane i umieszczane są w pamięci podręcznej aplikacji. Tworzony jest również licznik, który odświeża cache w określonych interwałach. Obecnie został on ustawiony na 30s.

Serwer został zdefiniowany tak, aby umożliwiać jedynie odczyt danych. Wystawione API obsługuje tylko metodę GET. Po otrzymaniu zapytania przez serwer, kierowane jest ono do odpowiedniego handlera i dane z cache zwracane są użytkownikowi w formie JSON'a.

Dane zbierane w aplikacji pochodzą z 3 źródeł:

- notowania giełdowe: https://www.money.pl/gielda/indeksy_gpw/wig/
- kursy walut: <https://www.money.pl/pieniadze/nbp/srednie/>
- kryptowaluty: <http://www.cryptocurrencyprices.net/>

Zawartość strony pobierana jest z użyciem biblioteki requests. Następnie przetwarzana jest przez narzędzie beautifulsoup. Za jego pomocą treść jest odpowiednio filtrowana w taki sposób, aby uzyskać jedynie interesujące informacje. Wybierane są poszczególne znaczniki HTML, a ich treść tekstowa jest umieszczana w ramce danych biblioteki pandas. Zdecydowano się na jej użycie ze względu na łatwość organizacji i manipulacji danymi tabelarycznymi. W ostatnim kroku etykiety danych są zamieniane na prostsze odpowiedniki i cały obiekt jest serializowany do formatu JSON.

Aplikacja kliencka

Część front-endową aplikacji napisano z wykorzystaniem frameworka Angular. Zdecydowano się na niego ze względu na łatwość i szybkość budowania aplikacji, a także ze względu na modułowe podejście do jej architektury. Wygląd aplikacji podzielono na 4 widoki: po jednym szczegółowym dla każdego zestawienia danych oraz jeden zbiorczy agregujący dane w skróconej formie.

Aplikacja komunikuje się z częścią serwerową wykorzystując protokół HTTP. Dla każdej parsowanej witryny został stworzony osobny serwis, który po otrzymaniu danych tworzy obiekty odpowiedniego, wcześniej zdefiniowanego typu. Wszystkie serwisy rozszerzają jednak tę samą klasę, więc działają w zunifikowany sposób.

Do zdefiniowania wyglądu aplikacji wykorzystano framework Bootstrap. Zdecydowano się na jego użycie ze względu na jego wszechstronność oraz elastyczność. Style zostały w aplikacji zdefiniowane w taki sposób, aby spełniać założenia 'Responsive web design'. Dzięki temu strona może być wyświetlona w przejrzysty sposób na ekranach o różnej wielkości.

Wnioski

Aplikację można byłoby rozbudować i wzbogacić o kolejne zestawienia danych, niekoniecznie z kategorii związanej z finansami. Należałoby jednak zwrócić uwagę na sposób ładowania treści, ponieważ narzędzie użyte do pozyskiwania zawartości dokumentu strony ma pewne ograniczenia. Jeżeli zawartość strony jest generowana dynamicznie za pomocą javascriptu, pobrana zostanie jedynie jej pierwotna struktura. Aby odczytać treść takiej strony można byłoby skorzystać na przykład z biblioteki selenium, która umożliwia pobranie strony z odpowiednio zdefiniowanym opóźnieniem lub nawigowanie po załadowanej stronie w naturalny sposób.

Napisane rozwiązanie nie przechowuje danych historycznych. Skupia się jedynie na bieżących wartościach. Można byłoby rozbudować ją o tę funkcjonalność. Należałoby wtedy jednak rozważyć zmianę wykorzystywanych narzędzi, tak aby ułatwić implementację połączenia z bazą danych oraz dodanie nowych endpointów w API. Jedną z możliwości byłby framework Django.

Obecnie jedynym używanym w aplikacji językiem jest język polski. Możliwym usprawnieniem byłaby także obsługa wielojęzyczności.