

Techniki multimedialne

**Post mortem projektu „Auction
Scraper”**

Alan Wielguszewski

L7

1. Opis projektu.

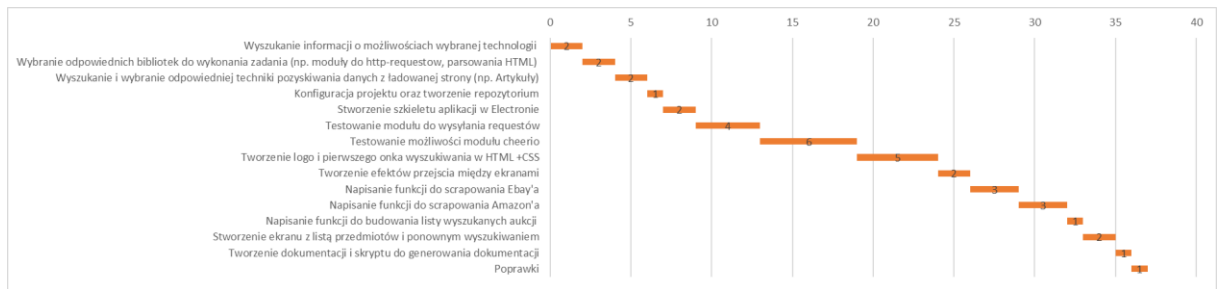
Tematem projektu było stworzenie prostego programu wyciągającego dane z określonych stron internetowych. Do projektu wybrałem pobieranie aukcji z popularnych serwisów aukcyjnych takich jak Ebay i Amazon. Program zdecydowałem się zbudować w oparciu o język Javascript oraz pomocniczo HTML i CSS, przy wykorzystaniu Node.js oraz Electron.

Node.js jest popularnym środowiskiem wykonawczym dla języka Javascript zbudowanym na silniku V8 z przeglądarki Google Chrome. Zapewnia on całą funkcjonalność języka dając możliwość budowania aplikacji webowych, desktopowych oraz od niedawna systemów wbudowanych.

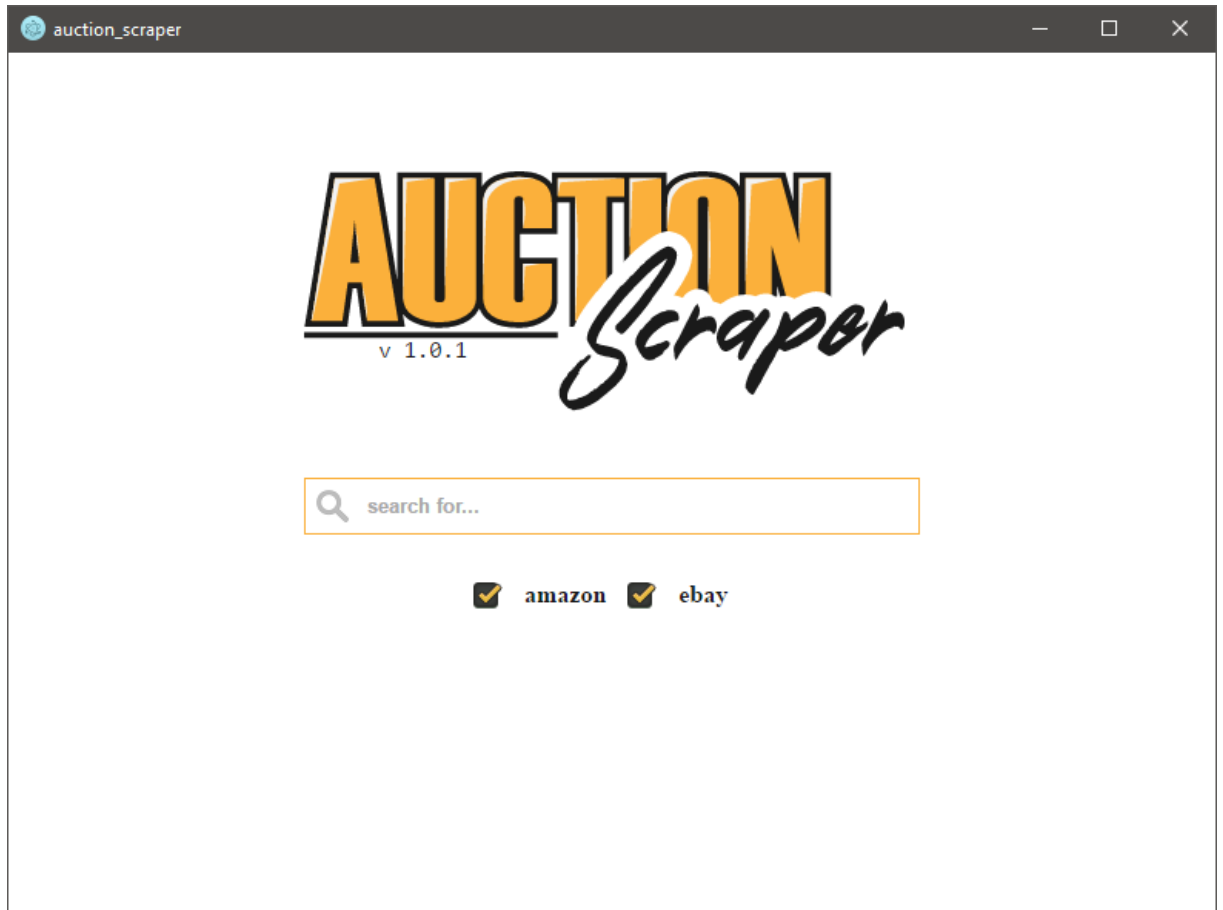
Electron jest środowiskiem opartym na Node.js implementującym swojego rodzaju sanbox'ową przeglądarkę, która pozwala na budowanie zaawansowanych aplikacji desktopowych przy użyciu Javascript, HTML i CSS oraz wszelkich innych bibliotek i frameworków napisanych w Javascript takich jak ReactJS, Angular itp.

2. Diagram Gantt'a

Task	start	duration	end
Wyszukanie informacji o możliwościach wybranej technologii	0	2	2
Wybranie odpowiednich bibliotek do wykonania zadania (np. moduły do http-requestow, parsowania HTML)	2	2	4
Wyszukanie i wybranie odpowiedniej techniki pozyskiwania danych z ładowanej strony (np. Artykuły)	4	2	6
Konfiguracja projektu oraz tworzenie repozytorium	6	1	7
Stworzenie szkieletu aplikacji w Electronie	7	2	9
Testowanie modułu do wysyłania requestów	9	4	13
Testowanie możliwości modułu cheerio	13	6	19
Tworzenie logo i pierwszego onka wyszukiwania w HTML +CSS	19	5	24
Tworzenie efektów przejścia między ekranami	24	2	26
Napisanie funkcji do scrapowania Ebay'a	26	3	29
Napisanie funkcji do scrapowania Amazon'a	29	3	32
Napisanie funkcji do budowania listy wyszukanych aukcji	32	1	33
Stworzenie ekranu z listą przedmiotów i ponownym wyszukiwaniem	33	2	35
Tworzenie dokumentacji i skryptu do generowania dokumentacji	35	1	36
Poprawki	36	1	37






3. Wygląd aplikacji.






auction_scraper

search again...

	Generic Usb Flash Drives 8GB Bulk 10 Pack in Green - Budget and Do the Job	\$45.99	amazon
	Belkin 3.0 USB-C (USB Type C) to USB-A Adapter [USB-IF Certified]	\$19.37	amazon
			on

auction_scraper

search again...

	Drive Memory Stick lot		e
	2TB 1TB 512GB Swivel USB 2.0 Flash Drive Memory Stick Pen Storage Thumb U Disk	\$8.88 to \$14.88	ebay
	USB Flash Drive Storage Memory Stick For Apple IOS Android Smartphone Computer	\$16.59	ebay

4. Problemy.

Pierwszym problemem było dostosowanie Electrona do projektu, jego konfiguracja oraz dobranie odpowiednich bibliotek, np. Wykorzystując jQuery w projekcie stwarzało dużo problemów, ze względu że Electron działa trochę inaczej niż normalna przeglądarka i 'this' w obiekcie jQuery nie wskazywało na aktualnie zwrócony obiekt, tak jak powinno tylko na obiekt okna w Electronie.

Drugim problemem było wyciąganie samych danych ze stron, np. Początkowo chciałem dołączyć do tej listy stron Allegro, ale wysyłając requesta z aplikacji, przykładowo na adres 'https://allegro.pl/listing?string=steam' z określonym słowem kluczowym do wyszukania, który w przeglądarce wyświetla pełną listę przedmiotów, do aplikacji zwraca jedynie szkielet pliku HTML a reszta jest budowana ze skryptów dynamicznie. Jest to bardzo popularne w teraźniejszych stronach internetowych wykorzystując frameworki frontendowe takie jak ReactJS. Z tego względu po wielu próbach zdecydowałem się porzucić Allegro i skupić się na Ebay i Amazon.

Kolejną przeszkodą były zabezpieczenia Amazona przed webcrawlerami i innymi tego typu, które od razu wykrywały próbę pobierania strony przez aplikacje, które nie są pełnoprawnymi przeglądarkami. Zamiast szkieletu strony, w odpowiedzi dostawałem propozycje z Amazona, aby wykupić ich oficjalne API...

Po wielu próbach z wieloma różnymi nagłówkami requesta, udało się w końcu pobierać kilka pozycji z wyszukanych aukcji.

Jeżeli miałbym tworzyć taką aplikację jeszcze raz, to przede wszystkim wykorzystałbym np React'a do budowy wyświetlanych treści w aplikacji, z tego względu, że Electron rozdziela wykonywanie programu na dwa procesy. Główny w którym działa aplikacja oraz gdzie tworzy się okna. W głównym procesie jednak nie można wykonywać skryptów dla aktualnie działającego okna. Oknem aplikacji zajmuje się drugi proces, który bezpośrednio z kolei nie ma dostępu do procesu głównego. Dlatego aby zainicjować z procesu pobocznego załadowanie nowego pliku HTML do bieżącego okna trzeba wysłać między procesowe komunikaty itp. co bywa kłopotliwe, dlatego też przy wykorzystaniu React'a zbudowałbym skrypty, które budują aplikacje dynamicznie przy wykorzystaniu tylko jednego pliku HTML.