Generator ruchu Google Analytics
Iteracja I architektura systemu

Bartłomiej Dalak Bartłomiej Karwowski
Bartosz Gromek Tomasz Kanas

18 kwietnia 2018

Wstęp

Dokument architektury systemu ma na celu przedstawienie wizji architektury. Opisana architektura może ulec zmianom w fazie implementacji.

Opis elementów architektury

Aplikacja

Aplikacją w naszym przypadku będzie skrypt **generate_ga_traffic.py**, generujący ruch według podanych przez użytkownika wartości. Będzie on uruchamiany przez konsolę.

Język

Wykorzystany zostanie Python w wersji 3.6.

Użyte biblioteki

- **requests**: wysyłanie zapytań do GA i Measurement Protocol Validation Server
- **pandas**: do obsługi pliku 'browser.csv', w którym mamy rozkład przeglądarek na terenie Polski.

send_requests_api

API służące do komunkacji z GA, sprawdza dane od użytkownika, generuje potrzebne dane oraz je wysyła.

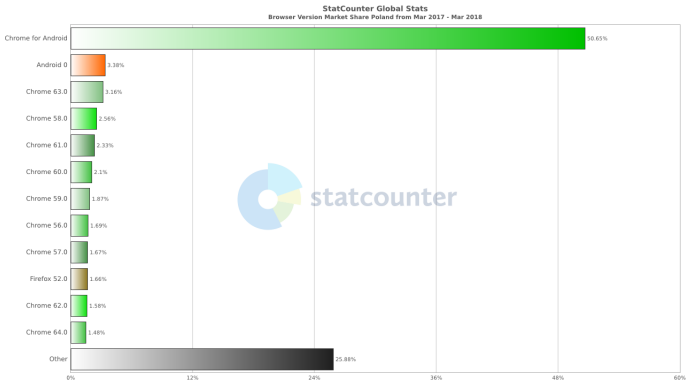
Metody

- **send(tracking_id, url, visits_no, time)**: sprawdza czy dane użytkownika mogą zostać wygenerowane, za pomocą funkcji **check_data**. Jeśli dane przejdą weryfikację to przekazuje je do niżej opisanej funkcji **generate_data**. Wpp. zwraca kod WRONG_DATA. Po odebraniu wygenerowanych danych, próbuje przesłać je bezpośrednio do GA. Przykładowe wysłanie danych: **requests.post("https://www.google-analytics.com/collect", data="v": 1, "t": "pageview", "tid": tracking_id, "cid": 1, "dp": url)**. W ten sposób będziemy wysyłać w pętli kolejne wejścia z wygenerowanych danych. Zwraca jeden z 2 możliwych kodów OK, CONNECTION_PROBLEM, w zależności od tego czy udało się wysłać wszystkie dane, czy utracono połączenie.

- **check_data(tracking_id, url)**: metoda, która wyśle requesta do Measurement Protocol Validation Server za pomocą **requests.post("https://www.google-analytics.com/debug/collect",data='tid': tracking_id, 'dp': url, 'v': 1)** i jeśli otrzymany response w formacie JSON w polu ["hitParsingResult"][0]["valid"] zawiera wartość true to zwróci tuple(true, []), wpp. zwróci tuple(false, parameters), gdzie parameters to lista błędnych parametrów, które otrzymamy z pól ["hitParsingResult"][0]["parserMessege"][i]["parameter"], gdzie i to indeks złego parametru.

- **generate_data(visits_no)**: metoda wołana przez **send()**, generujące odpowiednie dane do wysłania. Po odebraniu informacji przekazanych przez użytkownika, do odpowiedniej ilości zapytań przypisuje dane przygotowane z wiarygodnym rozkładem. Informacje do tego potrzebne zostaną zczytane z pliku 'browser.csv', który zostanie pobrany ze strony GlobalStats StatCounter (dane dot. oprogramowania użytkowników witryny). Informacje te zostaną przypisane na zmienną **distribution_informations** będącą typu DataFrame. Zostanie to wykonane tylko raz.

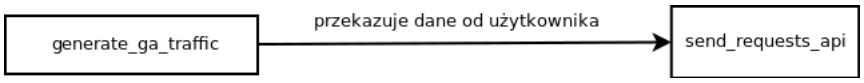
Wykres użytkowania przeglądarek na terenie Polski



Schemat działania

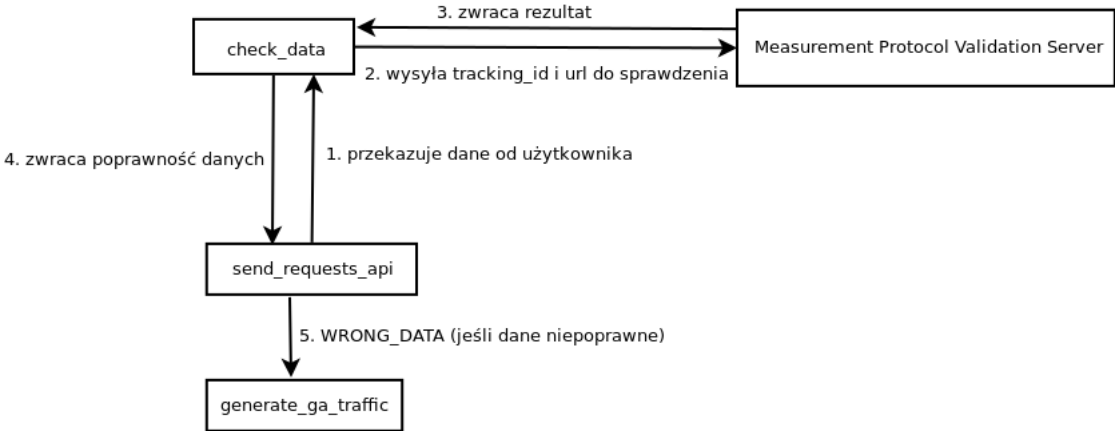
Przekazanie danych

Użytkownika wywołuję funkcję **send(tracking_id, url, visits_no, time)** udostępnioną przez **send_requests_api**. W rezultacie otrzymuje komunikat tego czy udało się pomyślnie wysłać żądanie.



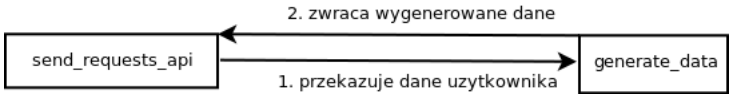
Sprawdzenie poprawności danych

API po uzyskaniu danych od użytkownika wywołuje funkcję **check_data** sprawdzającą czy podany tracking_id i url są poprawne. Jeśli nie są to API zwróci komunikat WRONG_DATA.



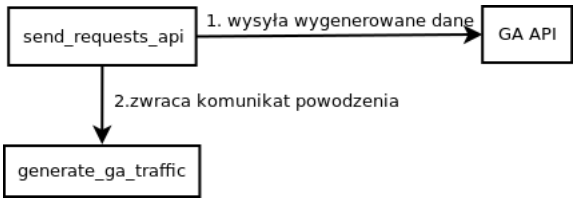
Wygenerowanie danych

Jeśli dane od użytkownika są poprawne to zostaje wywołana funkcja **generate_data**, która generuje i zwraca dane.



Wysyłanie danych i zwrócenie komunikatu

Po otrzymaniu wygenerowanych danych zostają one wysyłane do Google Analytics, a następnie zostaje zwrócony komunikat powodzenia.



Wykrywanie braku połączenia z internetem

Jeśli **send_requests_api** przez 10 min. nie uda się wysłać żadnego requesta do GA API, bądź nie uzyskamy odpowiedzi, w takim wypadku zostaje zatrzymane wysyłanie oraz zwrócony do **generate_ga_traffic** komunikat CONNECTION_PROBLEM. Wtedy użytkownik zostaje o tym poinformowany i może wybrać jedną z trzech możliwości co chce robić dalej.