Jędrzej Blak
Grzegorz Majchrzak

TASS – projekt 2

Etap 2

1. Koncepcja

1.1.Temat

Sprawdź trasy i parametry łączy prowadzących do popularnych adresów internetowych w wybranym przez użytkownika kraju świata, za pomocą usługi *super-ping*. Wyniki przedstaw w postaci graficznej na podkładzie mapowym.
Koncepcja wykonania (maks. 10 pkt.): powinna zawierać wytypowanie źródeł danych i technologii ich pobrania i przechowania; wytypowanie klucza łączącego dane lub metodyki uzyskania zadowalającego złączenia; zdefiniowanie scenariusza użycia aplikacji, technologii jej wykonania, ergonomii, architektury (1) albo metodyki analizy danych (2). Termin: 17 grudnia; za opóźnienie w dostarczeniu koncepcji naliczana jest jednorazowa kara: 5 punktów. Kara ta uwzględniana jest przy obliczaniu ostatecznej oceny; nie ma ona wpływu na zaliczenie projektu 2. Koncepcje dostarczone po 3 stycznia nie będą oceniane ani opiniowane, a projekt nie zostanie zaliczony.

1.2.Źródła danych oraz technologie ich pobrania

Niestety, zaproponowana w temacie projektu usługa super-ping nie działa - po wpisaniu adresu badanej strony nigdy nie wyświetlają się wyniki. Postanowiliśmy więc poszukać rozwiązań alternatywnych. Po przeanalizowaniu innych usług napotkaliśmy następujące problemy związane z pobieraniem danych:

- usługi nie działają
- usługi są płatne
- usługi nie udostępniają swojego API (niezbędne stworzenie parserów)
- ograniczona liczba serwerów, z których można wysyłać pingi
- w większości brak opcji traceroute

Stworzyliśmy listę serwisów najbardziej odpowiadających tematyce, które mogą okazać się użyteczne jako źródło danych:
https://ping.pe/
https://wondernetwork.com/
https://www.locaping.com/
http://maplatency.com/

Jako podstawowe źródło danych postanowiliśmy wybrać stronę https://ping.pe/. Za jej pomocą można sprawdzić opóźnienia oraz drogę, jaką pokonują pakiety do wybranego adresu z 31 predefiniowanych serwerów znajdujących się na różnych kontynentach (Ameryka Północna, Europa, Azja, Australia). Przy jej użyciu przetestujemy połączenia z kilkunastoma wybranymi przez nas stronami internetowymi (najpopularniejszymi) z różnych państw.

1.3.Przechowywanie danych

Dane pobrane ze stron będziemy przechowywali w formie plików tekstowych (oddzielnych dla każdej strony), zawierających państwo, w którym się znajdują, nazwy serwerów źródłowych średni czas pingu wychodzącego z tych serwerów oraz trasę, którą pokonują pakiety od tego serwera, otrzymaną za pomocą polecenia *traceroute*. Każdy

element trasy pakietu będzie zawierał adres routera, nazwę hosta (jeśli dostępna), państwo, w którym się on znajduje oraz nazwę jego właściciela.

1.4.Łączenie i obróbka danych

Naturalnym wydaje się być połączenie wspólnych elementów ścieżki pomiędzy poszczególnymi serwerami. W ten sposób będzie można zlokalizować najpopularniejsze trasy pakietów pomiędzy poszczególnymi państwami. Na podstawie takiego powiązania danych będzie można przeanalizować sieci powiązań pomiędzy poszczególnymi państwami.

1.5.Scenariusze użycia aplikacji

Użytkownik aplikacji będzie miał możliwość wybrania z wcześniej przygotowanej listy państwa, dla którego chce sprawdzić powiązania z dostępnymi dla nas serwerami. Po załadowaniu danych zostanie wyświetlona mapa świata z naniesioną trasą od poszczególnych serwerów do stron popularnych w wybranym państwie.

Drugą opcją wyświetlania danych będzie mapa reprezentująca średni czas podróży pakietów z dostępnych dla nas serwerów do wybranego państwa. Zostanie to zrealizowane poprzez nałożenie na mapę warstwy, która za pomocą kolorów (od zielonego, przez żółty, do czerwonego) będzie prezentowała średnie opóźnienie.

Dodatkowo obok mapy wyświetlone zostaną dane statystyczne dotyczące części sieci, która jest aktualnie prezentowana na mapie.

1.6.Technologia wykonania

Dane do naszej aplikacji zostaną pobrane za pomocą parserów napisanych w języku Java, przy użyciu biblioteki służącej do pobierania oraz parsowania stron internetowych - jsoup.
Tworzona przez nas aplikacja, będzie aplikacją webową. Warstwa serwerowa będzie napisana za pomocą języka Java, w technologii Spring Framework. Interakcja z użytkownikiem będzie odbywała się poprzez przeglądarkę internetową. Na stronie internetowej za pomocą JavaScriptowej biblioteki Leaflet.js zostanie wyświetlona mapa, a dane z aplikacji naniesione przy pomocy JavaScript. Dodatkowo zostanie przez nas użyta biblioteka służąca do analizy grafów i sieci - neo4j. Zakładamy możliwość zmiany bibliotek podczas realizacji projektu w przypadku napotkania problemów (np. jpgapht, grph do analizy grafów, API Google do wyświetlania mapy).

2. Raport końcowy

2.1.Pozyskanie danych

Jako analizowane kraje zostały wybrane następujące kraje, wraz z 5 bardzo popularnymi witrynami według rankingu Amazon Alexa Top Sites:

- Niemcy – chip.de, gmx.net, spiegel.de, t-online.de, web.de
- Polska – allegro.pl, interia.pl, olx.pl, onet.pl, wp.pl
- Rosja – avito.ru, mail.ru, ok.ru, vk.com, yandex.ru

- USA – amazon.com, craigslist.org, facebook.com, google.com, youtube.com
- Chiny – baidu.com, jd.com, qq.com, sohu.com, taobao.com

Wybrane kraje są zróżnicowane pod względem lokalizacji, a także politycznie, dodatkowo w każdym z nich znajdują się serwery, z których badane były opóźnieina oraz trasy pakietów.

Wszystkie wykorzystane do działania programu dane zostały pozyskane ze strony http://ping.pe, która została wytypowana jako najbardziej użyteczna we wstępnej fazie projektu. Strona cechuje się mnogością serwerów do analizy, a dostęp do danych jest darmowy. Ponadto, z uwagi na prostą budowę strony, ułatwione jest pozyskiwanie z niej danych (parsowanie HTML). Serwery, dla których pobierane są dane znajdują się w następujących krajach: Kanada, USA, Holandia, Luksemburg, Niemcy, Włochy, Rosja, Singapur, Japonia, Australia oraz Chiny.

Dane są pozyskiwane przez parser napisany w języku Java z wykorzystaniem zapytań REST GET do serwera ping.pe w celu pozyskania opóźnień oraz za pomocą biblioteki jSoup do uzyskania tras pakietów. W toku pracy nad projektem okazało się, że w drugim wypadku zapytania GET nie były wystarczająco niezawodne – często nie udawało się otrzymać odpowiedzi z serwera. Zadecydowano o pobraniu statycznej strony po określonym czasie, a następnie wczytanie jej do parsera w celu uzyskania wyników. Gotowe dane są zapisywane w postaci plików JSON, w katalogach z nazwami krajów, w których znajdują się analizowane serwisy.

2.2.Przedstawienie wyników

Wyniki prezentowane są za pomocą aplikacji webowej, działającej w oparciu o środowisko uruchomieniowe Node.js z frameworkiem Express. Zdecydowano o skorzystaniu z API Google Maps:

- Geocoding API – do uzyskiwania przybliżonych współrzędnych badanych lokalizacji,
- Javascript API – do prezentacji wyników – przedstawienia opóźnień i tras pakietów na mapie

Aplikacja analizuje zawartość katalogu „countries”, znajdującego się w głównym katalogu serwera, a następnie wczytuje pliki JSON dla poszczególnych krajów. Główny widok aplikacji składa się z mapy oraz menu, za pomocą którego użytkownik może wybrać interesujący go kraj, witrynę oraz dane (opóźnienia dla wszystkich serwerów lub trasy pakietów dla serwera wybranego przez użytkownika).

2.3.Odstępstwa od pierwotnej koncepcji

Podczas realizacji projektu zdecydowano się wprowadzić kilka zmian w stosunku do pierwotnej koncepcji:

- Zdecydowano się skorzystać z jednego serwisu jako źródła danych (ping.pe), gdyż cechował się on ich największą różnorodnością, ilością, nie posiadał ograniczeń w dostępie do nich oraz cechował się łatwością ich pozyskania.

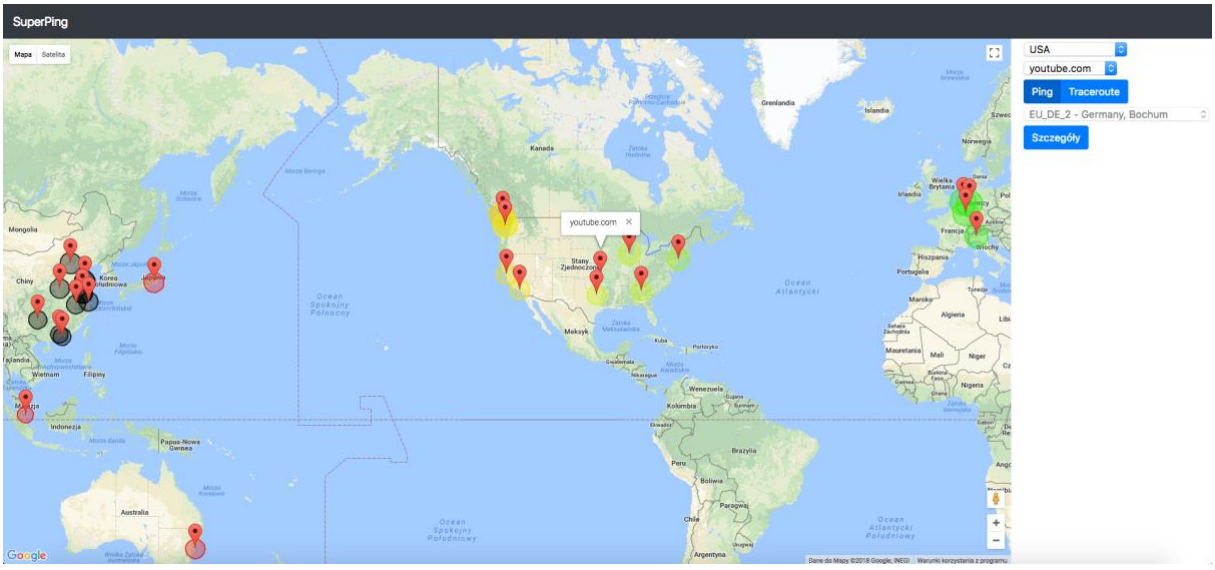
- Nie skorzystano z frameworka Spring Boot, gdyż okazał się on zbyt rozbudowany na potrzeby projektu – jego funkcjonalności nie byłyby wykorzystane. W celu ułatwienia fazy implementacji oraz użytkowania zdecydowano się na przejście na język Javascript i skorzystanie ze środowiska Node.js, które zużywa zdecydowanie mniej zasobów sprzętowych.
- Nie wykorzystano biblioteki neo4j do analizy grafów i sieci, gdyż okazała się ona zbędna.
- Biblioteka Leaflet.js do nanoszenia informacji na mapę nie została użyta, dane wyświetlane są jedynie w oparciu o narzędzia udostępnione przez firmę Google, w celu ujednolicenia projektu i nieobciążaniu go zbędnymi bibliotekami.
- Ścieżki pakietów są wyświetlane na mapie dla pojedynczego serwera, a nie wszystkich jednocześnie z uwagi na czytelność danych.

2.4.Skrócona instrukcja instalacji i użytkowania

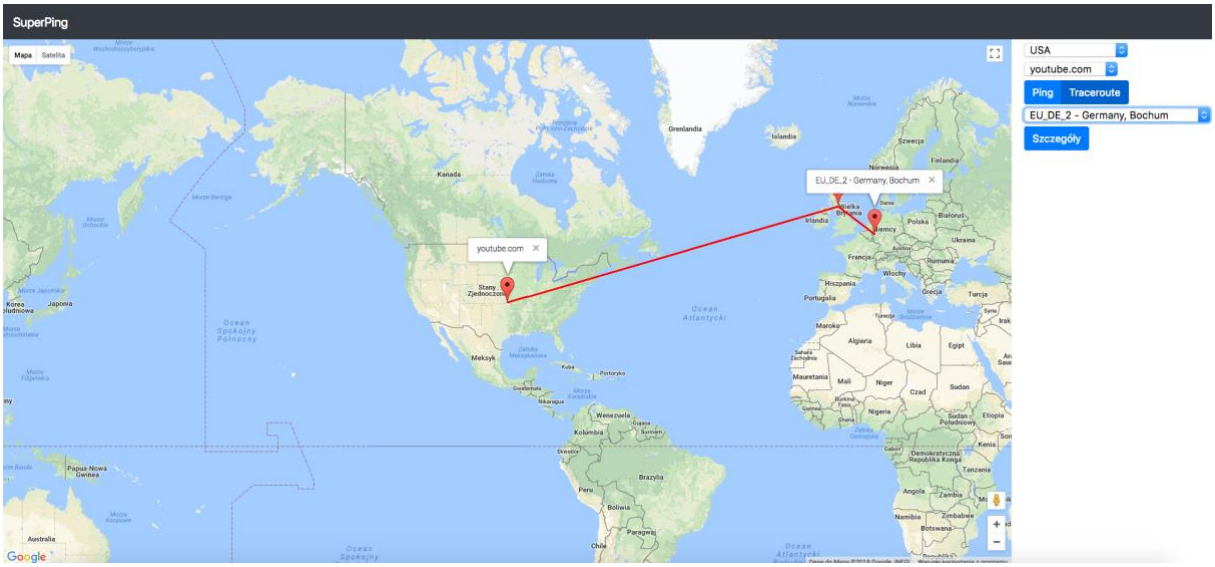
W celu uruchomienia parsera należy dysponować komputerem z zainstalowanym środowiskiem Java SE Development Kit w wersji 8 lub nowszej. Należy skompilować projekt mavenowy, a następnie uruchomić plik powstały w wyniku kompilacji. Statyczne strony pobrane w celu analizy traceroute'a powinny znajdować się w podkatalogu resource w katalogu głównym projektu mavenowego. Dane w postaci JSON powstałe w wyniku działania programu będą znajdować się w podkatalogu data.

Do działania aplikacji webowej, za pomocą której prezentowane są dane, jest niezbędne środowisko Node.js w wersji 8.9.4. W celu uruchomienia serwera należy przejść do katalogu zawierającego plik o nazwie index.js, pobrać potrzebne pakiety za pomocą polecenia *„npm install”*, a następnie uruchomić aplikację poleceniem *„node index.js”*. Serwer WWW ze stroną jest uruchamiany na porcie 3000, w przeglądarce należy więc wpisać adres *„loalhost:3000”*.

Za pomocą menu po prawej stronie użytkownik może wybrać interesujący go kraj, następnie jedną z popularnych witryn w wybranym kraju. Kolejnym krokiem jest wybranie typu wyświetlanych danych:



Przycisk „Ping” do prezentacji opóźnień do wybranej przez użytkownika strony ze wszystkich serwerów. Na mapie pojawia się pinezka z orientacyjną lokalizacją serwera ze stroną internetową oraz pinezki serwerów wraz z kolorowymi kołami, reprezentującymi opóźnienie. Po kliknięciu na pinezkę użytkownik ma dostęp do wartości średniego opóźnienia z danego serwera do strony.



Przycisk „Traceroute” do prezentacji graficznej tras pakietów z wybranego serwera do strony internetowej. Menu pozwalające na wybór serwera jest aktywne jedynie po wybraniu tej opcji. Na mapie pojawia się pinezka z orientacyjną lokalizacją serwera ze stroną internetową oraz połączone liniami pinezki reprezentujące kolejność lokalizacji, przez które przechodzą pakiety w drodze od serwera do wybranej strony internetowej.

SuperPing

Nazwa serwera	Minimum	Średni	Maksimum	Procent utraconych pakietów
Canada, BC, Vancouver	166.47 ms	166.79 ms	167.24 ms	0 %
Canada, BC, Vancouver	171.08 ms	173.63 ms	181.39 ms	0 %
USA, WA, Seattle	160.04 ms	160.12 ms	160.4 ms	0 %
USA, WA, Seattle	154.76 ms	154.78 ms	154.8 ms	0 %
USA, CA, Fremont	160.08 ms	160.31 ms	161.54 ms	0 %
USA, CA, Los Angeles	158.56 ms	158.62 ms	158.71 ms	0 %
USA, TX, Dallas	132.58 ms	132.68 ms	132.77 ms	0 %
USA, IL, Chicago	118.14 ms	119.15 ms	123.41 ms	0 %
USA, GA, Atlanta	116.7 ms	116.76 ms	116.8 ms	0 %
USA, NY, New York	99.78 ms	99.82 ms	99.95 ms	0 %
Netherlands, Nuland	21.95 ms	21.99 ms	22.08 ms	0 %
Luxembourg, Roost	33.9 ms	34.32 ms	36.77 ms	0 %
Germany, Bochum	21.49 ms	21.55 ms	21.68 ms	0 %
Italy, Milan	23.1 ms	23.22 ms	24.45 ms	0 %
Russia, Moscow	33.4 ms	34.03 ms	35.87 ms	0 %
Russia, Tomsk	103.21 ms	103.38 ms	103.71 ms	0 %
Singapore	311.3 ms	311.36 ms	311.65 ms	0 %
Japan, Tokyo	296.04 ms	296.13 ms	296.25 ms	0 %
Australia, Sydney	296.08 ms	296.16 ms	296.21 ms	0 %
China, Guizhou	- ms	- ms	- ms	100 %
China, Henan	- ms	- ms	- ms	100 %
China, Guangzhou	- ms	- ms	- ms	100 %
China, Beijing	- ms	- ms	- ms	100 %

Dla obu typów danych jest dostępna tabela ze szczegółami, znajdująca się poniżej mapy, dostępna także po kliknięciu na przycisk „Szczegóły”.