Rozproszona aplikacja umożliwiająca przeglądanie fraktali

Autor:
Tymon Tobolski (181037)
Jacek Wieczorek (181043)
Mateusz Lenik (181142)

Prowadzący:
dr inż. Marek Woda

Wydział Elektroniki
III rok
Pn 12.15 - 14.00

Spis treści

1 Opis projektu

Celem projektu jest stworzenie aplikacji internetowej pozwalającej użytkownikowi na przeglądanie fraktali renderowanych na klastrze. Aplikacja pozwalałaby na przybliżanie i oddalanie widocznej części fraktala, zmianę kolorów oraz zapisywanie ostatniej przeglądanej pozycji przez danego użytkownika.

2 Uzasadnienie biznesowe

Projekt może zostać wykorzystany jako proof of concept dla aplikacji przetwarzających duże ilości danych i prezentację ich w formie lub na mapie. Jest to proste do rozbudowy demo technologiczne, na podstawie którego można stworzyć bardziej skomplikowane aplikacje.

3 Technologie

Aplikacja będzie oparta o technologie wykorzystujące JVM, dzięki czemu możliwe będzie uruchomienie jej na kilku platformach bez konieczności modyfikacji. Dodatkowo ułatwia to korzystanie z różnych języków programowania podczas implementacji aplikacji. Za interfejs webowy będzie odpowiadała aplikacja w Ruby on Rails, która będzie się komunikowała z rozproszonym backendem zaimplementowanym w Scali.

- Platforma: JVM
- Język implementacji: Ruby, Scala, CoffeeScript
- Baza danych: SQLite lub MySQL
- Framework: Ruby on Rails

4 Funkcjonalności systemu

- Obsługa użytkowników
  - Utworzenie konta,
  - Autentykacja za pomocą OAuth,
  - Pobieranie awatarów,
  - Zapisanie ustawień i ostatniej przeglądanej pozycji.
- Wyświetlanie mapy
  - Nawigacja po mapie za pomocą myszy,
  - Wybór renderowanego fraktala,
  - Wybór kolorów dla renderowanego fraktala,
  - Wybór metody podziału obrazu do renderowania.
- Renderowanie
  - Podział obrazu na mniejsze fragmenty do renderowania na backendzie,
  - Przesyłanie żądań z aplikacji do backendu,
  - Komunikacja między węzłami backendu.

5 Implementacja

Praca nad projektem została podzielona na dwie części, aplikację internetową oraz rozproszone API pozwalające na renderowanie części fraktala.
W obecnej wersji system pozwala na wybór renderowanego fraktala spośród dwóch algorytmów, Mandelbrota i zbioru Julii. Oba obrazy można uzyskać zarówno w wersji monochromatycznej, jak i kolorowej. Dla każdego z fraktali można również zdefiniować ilość iteracji algorytmu oraz wymiary płytek.

6 Aplikacja internetowa

Aplikacja internetowa została utworzona przy użyciu frameworka Ruby on Rails. Do wyświetlania fraktala wykorzystana została biblioteka Leaflet. Aplikacja pobiera poszczególne części fraktala łącząc się z rozproszonym API. Obecnie zaimplementowane jest logowanie za pomocą Twitter OAuth. Dla zalogowanego użytkownika zapisywana jest ostatnie przeglądane współrzędne oraz wybrane parametry renderowanego fraktala.

Aplikacja internetowa została zaimplementowana w architekturze MVC z wykorzystaniem architektury REST.

6.1 Kontrolery

6.1.1 PagesController

Kontroler odpowiedzialny za wyświetlanie stron statycznych.

6.1.2 SessionsController

Kontroler odpowiedzialny za tworzenie i kończenie sesji użytkownika

6.1.3 UsersController

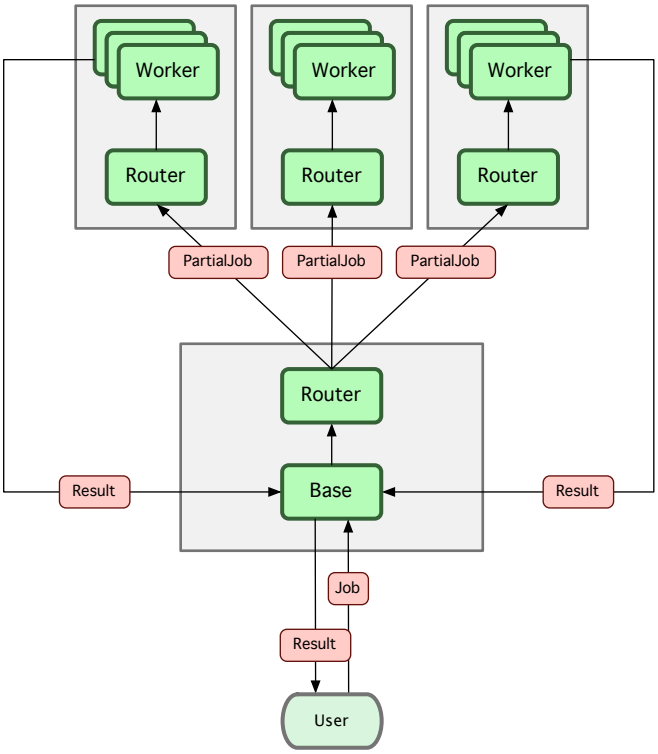
Kontroler odpowiedzialny za aktualizację ustawień fraktala dla zalogowanego użytkownika

6.2 Widoki

Wszelkei widoki zostały napisane przy uzyciu języka znaczników *Haml*. W celu generowania formularzy użyty został gem *SimpleForm*, a cały layout oparty o bibliotekę styli css - *Bootstrap*. Skrypty wykorzystujące *jQuery*, napisane zostały przy pomocy języka *CoffeScript*, kompilowalnego do kodu javascript.

7 Rozproszone API

Biblioteka Leaflet pobiera z API obrazki o zdefiniowanych przez użytkownika rozmiarach w pikselach korzystając z URL w formacie `http://{serwer}/img/{x}/{y}/{z}` wypełniając pola `x`, `y`, `z` odpowiednimi wartościami. Następnie, zgodnie z rysunkiem 1, zapytanie jest przydzielane do odpowiednich węzłów, renderowane, a rezultat zwracany jest do klienta.

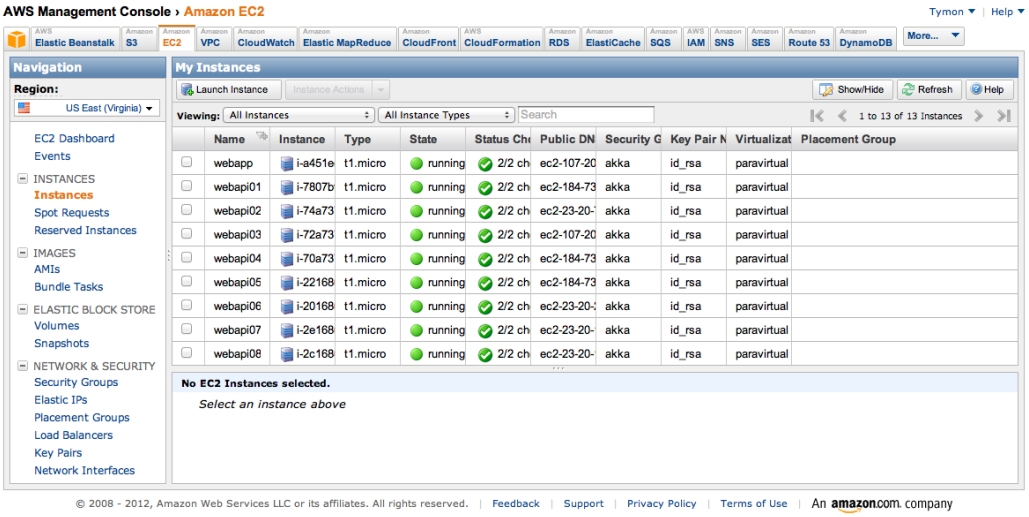


Rysunek 1: Load balancing.

API zostało zaimplementowane w języku Scala wykorzystując bibliotekę Akka.io umożliwiającą łatwe użycie Actor pattern.

8 Wdrożenie

Aplikacja została wdrożona na klastrze obliczeniowym składającym się z dziewięciu maszyn wirtualnych. Ze względu na łatwą konfigurację, jako host maszyn wirtualnych została wybrana platforma Amazon EC2.



Rysunek 2: Klaster obliczeniowy.

9 Testy

T ODO

10 Podsumowanie

Zaimplementowana zostały podstawowe funkcjonalności aplikacji. W przyszłości dodane zostaną pozostałe funkcjonalności, takie jak wybór dodatkowych opcji w aplikacji. Do projektu dodana zostanie również dokumentacja projektowa oraz testy behawioralne.