Prezentacja nt. pracy inżynierskiej

System zarządzania treścią dla serwisu informacyjnego oparty na mikroserwisach

ang. Micro-services based content management system for news website

Autor: Dariusz Hatala

Politechnika Krakowska
im. Tadeusza Kościuszki

Wydział Inżynierii
Elektrycznej i Komputerowej

Cele pracy inżynierskiej - 1

- Opracowanie i przedstawienie przykładowej architektury systemu spełniającego poniższe warunki:
  - Wysoka skalowalność
  - Odporność na błędy
  - Architektura rozproszona oparta na mikroserwisach
  - Integracja z usługami chmury obliczeniowej
- Wykonanie przykładowego systemu na podstawie uprzednio ustalonego scenariusza

Cele pracy inżynierskiej - 2

- Zbadanie i analiza problemów występujących w trakcie tworzenia takiego systemu
- Wykonanie testów:
  - wydajnościowych związanych ze skalowaniem aplikacji
  - niezawodności związanej z wystąpieniem awarii w poszczególnych modułach aplikacji

Uzasadnienie wyboru tematu

- Tendencja na rynku do:
  - Odchodzenia od architektur monolitycznych
  - Optymalizacja kosztów poprzez rezygnacji z utrzymania infrastruktury "on premises" na rzecz chmury obliczeniowej
  - Wysoka konkurencja na rynku usług internetowych wymusza tworzenie usług w skali masowej (wymagana skalowalność)
- Niewielka ilość materiałów omawiających integrację całego stosu technologicznego usług umożliwiających zarządzanie instancjami systemu mikro-serwisów.

Wybrany scenariusz

- Utworzenie prostego systemu zarządzania treścią dla ogólnotematycznej strony informacyjnej.
- Treści tworzone w edytorze WYSIWYG.
- Zapewnienie wysokiej skalowalności systemu w sposób zautomatyzowany
- Zapewnienie odporności systemu na błędy poszczególnych usług
- Zapewnienie optymalizacji kosztów poprzez dynamiczną alokację zasobów w chmurze obliczeniowej
- Zaczerpnięcie przykładowych artykułów za pomocą API Guardiana

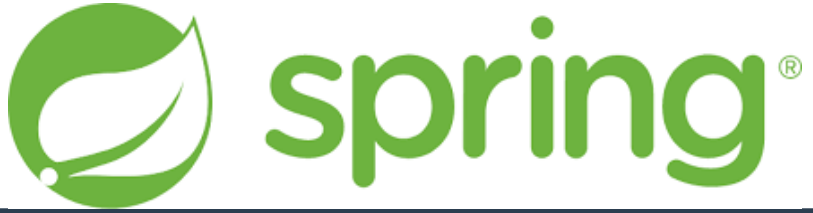
Uzasadnienie wyboru technologii - 1

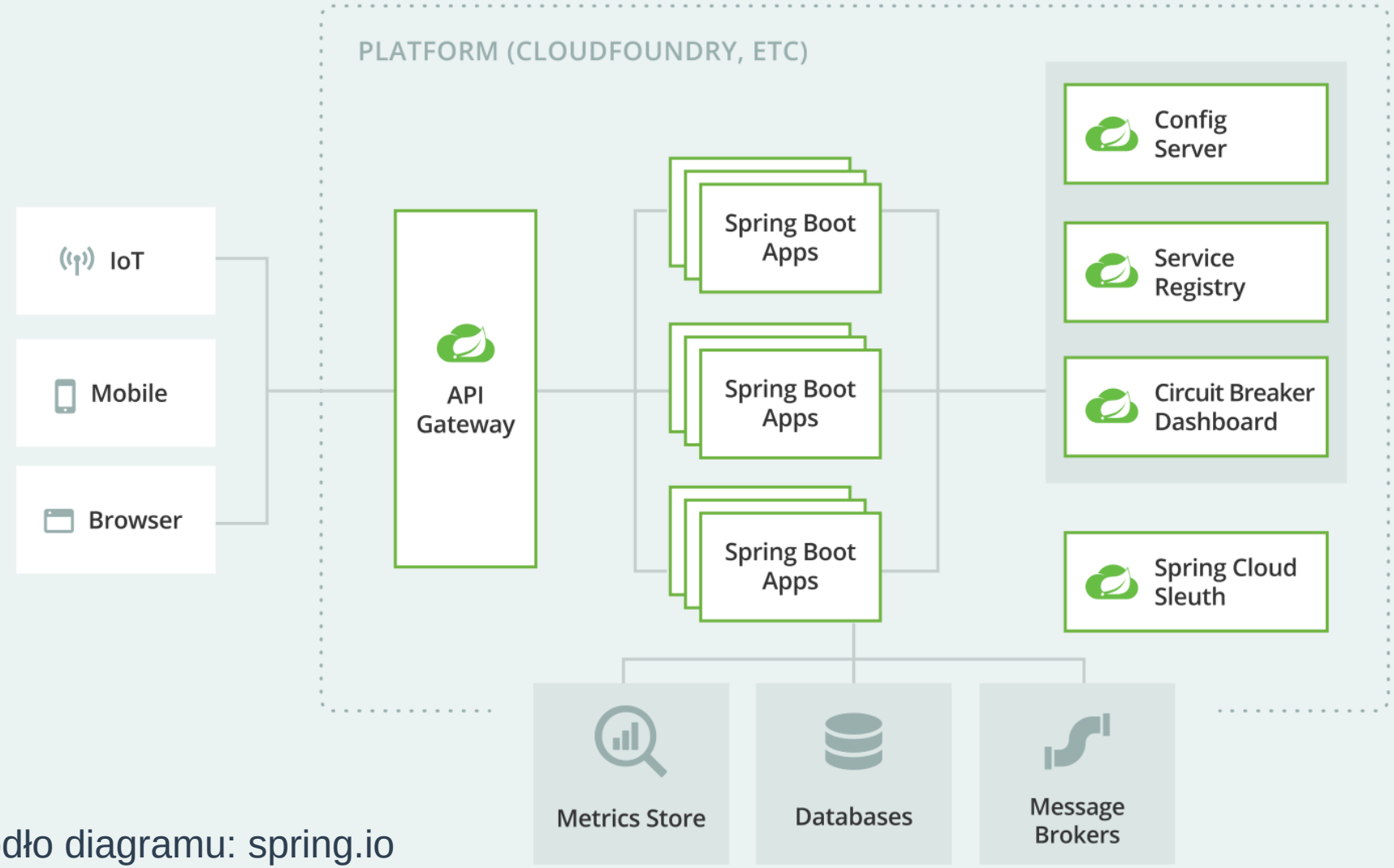
Spring Cloud

- "Projekt parasolowy"
- Pełny stos technologiczny do obsług mikro-serwisów
- Warstwa abstrakcji nad poszczególnymi technologiami

Spring Boot do tworzenia mikro-serwisów

- Zintegrowany serwer webowy - Tomcat
- Konwencja zamiast konfiguracji
- Redukcja tzw. "boilerplate code"





źródło diagramu: spring.io

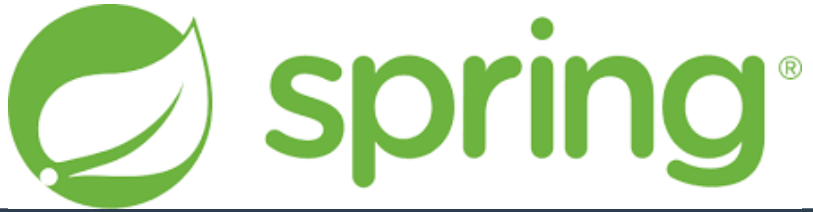
Uzasadnienie wyboru technologii - 2

Spring Data jako warstwa dostępu do danych

- Abstrakcja ponad technologią przechowywania danych
- Automatyzacja zarządzania transakcjami

Spring Security do autoryzacji i uwierzytelnienia

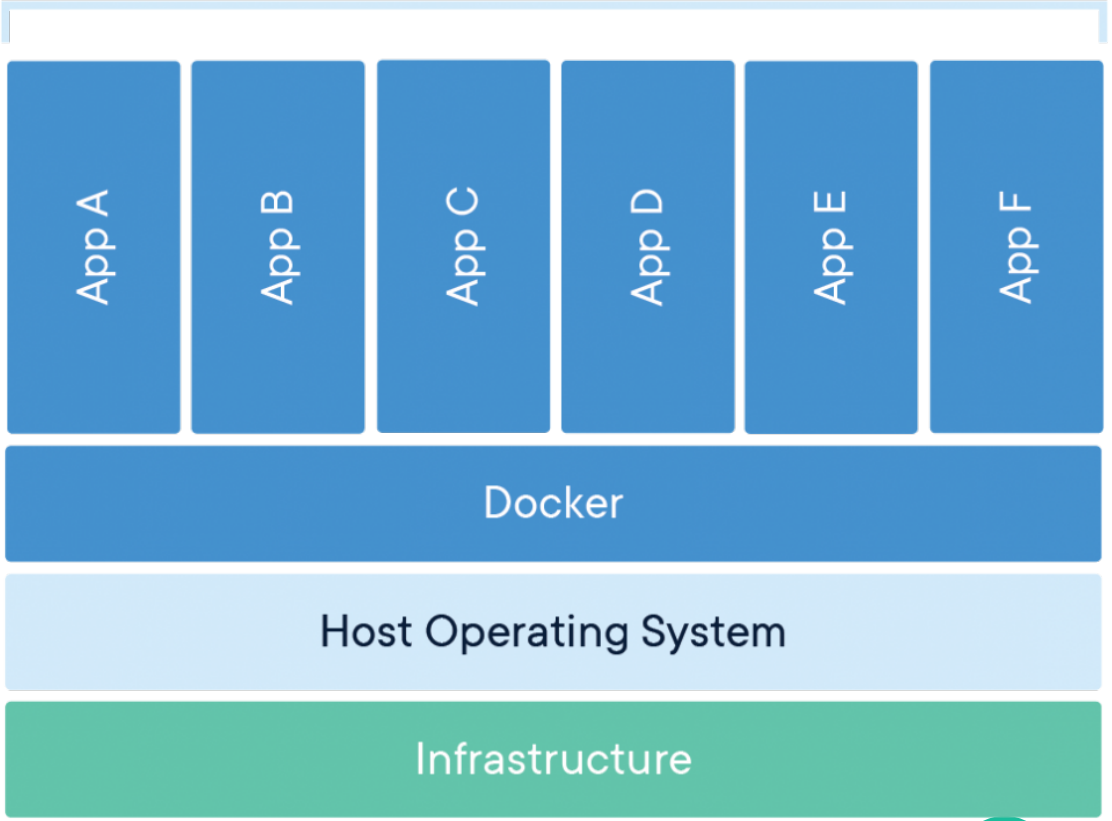
- Integracja ze Spring Boot
- Integracja z wieloma dostawcami tożsamości



Uzasadnienie wyboru technologii - 3

Docker do konteneryzacji instancji aplikacji

- Poziom abstrakcji ponad systemem operacyjnym
- Zachowanie wysokiej wydajności poprzez uniknięcie wirtualizacji
- Eliminacja konfiguracji środowiska w czasie wdrożenia



Uzasadnienie wyboru technologii - 4

Spring Cloud Gateway jako "brama" dla API

- Routowanie, filtrowanie, równoważenie ruchu od klientów

Netflix Eureka jako rejestr usług

- Integracja z projektem Spring Cloud
- Odnajdywanie usług od strony serwera

Netflix Hystrix pełniący funkcję "bezpiecznika"

- Integracja z projektem Spring Cloud
- Odpowiednie zarządzanie ruchem w przypadku nieresponsywnej usługi

Uzasadnienie wyboru technologii - 5

Spring Cloud Config Server do konfiguracji poszczególnych usług

- Scentralizowana konfiguracja
- Zmiana konfiguracji bez potrzeby restartu

Spring Cloud Sleuth do zarządzania logami oraz śledzenia operacji w systemie rozproszonym

Netflix Hystrix Dashboard oraz **Spring Actuator** do monitoringu usług

Uzasadnienie wyboru technologii - 6

Amazon Web Services jako dostawca chmury obliczeniowej

- Dojrzała chmura obliczeniowa
- Szeroka gama dostępnych usług
- Elastyczne zarządzanie kosztami

Uzasadnienie wyboru technologii - 7

Amazon DocumentDB jako silnik dla baz danych

- Zarządzana usługa bazy danych typu NoSQL
- Kompatybilna z popularnym MongoDB
- Wsparcie dla replikacji w celu uzyskania wysokiej dostępności

Amazon Virtual Private Cloud do izolacji publicznej oraz prywatnej części systemu

Uzasadnienie wyboru technologii - 8

Amazon Cognito dla zarządzania tożsamością użytkowników

AWS Fargate jako serwer kontenerów

- Abstrakcja ponad sprzętem
- Wsparcie dla obrazów Docker'a
- Automatyczne skalowanie w zależności od obciążenia maszyn

Uzasadnienie wyboru technologii - 9

Angular Framework do utworzenia interfejsu użytkownika

- Interfejs w przeglądarce internetowej typu Single Page Application
- Wsparcie języka TypeScript – statyczne typowanie w JavaScript
- Wstrzykiwanie zależności
- Leniwe ładowanie komponentów

Podsumowanie

- Zasadnicza trudność - integracja wielu technologii
- Główny cel – zbadanie oraz omówienie problemów związanych z taką architekturą aplikacji

Dziękuję za uwagę.

Politechnika Krakowska
im. Tadeusza Kościuszki

Wydział Inżynierii
Elektrycznej i Komputerowej