Uniwersytet Gdański
Wydział Matematyki, Fizyki i Informatyki
Instytut Informatyki

Facebook Clone

Łukasz Nowosielski

Projekt z przedmiotu technologie chmurowe na kierunku informatyka profil praktyczny na Uniwersytecie Gdańskim.

Gdańsk

30 maja 2024

Spis treści

1 Opis projektu

Projekt "facebook-clone"powstał z myśla o dynamicznie rozwijajacej sie firmie **ConnectSphere**, która specjalizuje sie w dostarczaniu innowacyjnych rozwiazań technologicznych. **ConnectSphere** zauważyła, że wiele małych i średnich przedsiebiorstw potrzebuje prostych, ale skutecznych narzedzi do budowania społeczności i komunikacji z klientami. Jednak popularne platformy społecznościowe, takie jak Facebook, czesto sa zbyt obciażone nadmiarem funkcji, co prowadzi do wolnej i nieefektywnej komunikacji.

ConnectSphere potrzebowała prostego, ale skutecznego narzedzia do budowania społeczności wokół swojej misji. Chcieli stworzyć miejsce, gdzie firmy mogłyby łatwo komunikować sie ze swoimi klientami, dzielić sie aktualnościami i informacjami o produktach, a także otrzymywać bezpośrednie opinie.

Dlatego powstał pomysł na stworzenie "facebook-clone platformy, która oferuje podstawowe funkcje społecznościowe, takie jak tworzenie profilu, udostepnianie postów, komentowanie i reagowanie na posty innych użytkowników. Projekt ma na celu dostarczenie tych funkcji w prosty i intuicyjny sposób, umożliwiajac **ConnectSphere** skuteczna komunikacje i budowanie społeczności wokół ich produktów i usług.

1.1 Opis architektury - 8 pkt

"facebook-clone"jest zbudowane z interfejsu użytkownika (frontend), serwera aplikacji (backend), bazy danych oraz serwisu uwierzytelniania Keycloak.

Aplikacja jest hostowana na klastrze Kubernetes, który zapewnia skalowalność, niezawodność i wydajność. Interfejs użytkownika jest dostepny dla użytkowników za pośrednictwem przegladarki internetowej, podczas gdy serwer aplikacji komunikuje sie z baza danych mongodb i obsługuje żadania użytkowników. Elementy te sa zabezpieczone dzieki integracji z Keycloak.

Aplikacja korzysta z Ingress w Kubernetes, który zarzadza dostepem do usług w klastrze, zapewniajac wydajne i niezawodne przekierowanie ruchu. Dzieki Ingress, aplikacja jest dostepna pod adresem https://facebook-clone.com (frontend), https://api.facebook-clone.com (backend) i https://keycloak.facebook-clone.com (keycloak). Ingress umożliwia także łatwe zarzadzanie ruchem sieciowym, co jest kluczowe dla utrzymania wydajności i niezawodności aplikacji.

1.2 Opis infrastruktury - 6 pkt

Aplikacja "facebook-clone"jest zaprojektowana do działania w środowisku Kubernetes, wykorzystujac Minikube. Minikube to narzedzie, które umożliwia uruchomienie jednego wezła klastra Kubernetes na lokalnym komputerze. Jest to idealne rozwiazanie dla deweloperów, którzy chca testować swoje aplikacje w środowisku Kubernetes bez konieczności tworzenia pełnego klastra. Specyfikacja klastra Minikube to 4GB pamieci RAM, 2 vCPU i 10GB miejsca na dysku.

1.3 Opis komponentów aplikacji - 8 pkt

Aplikacja składa sie z:

- **Frontend:** Interfejs użytkownika został zbudowany za pomoca biblioteki React. Do serwowania statycznych plików używamy serwera Nginx. Kod źródłowy frontendu jest zawarty w kontenerze Docker, który jest konfigurowany i uruchamiany za pomoca Dockerfile.
- **Backend:** Backend został zbudowany za pomoca frameworka Express.js. Backend komunikuje sie z baza danych MongoDB, która jest hostowana na platformie MongoDB Atlas. Podobnie jak frontend, backend jest również zawarty w kontenerze Docker, który jest konfigurowany i uruchamiany za pomoca Dockerfile.
- **Baza danych:** Aplikacja korzysta z MongoDB jako bazy danych. Jest ona hostowana w chmurze, co zapewnia dodatkowa niezawodność i elastyczność. Baza danych jest skonfigurowana do komunikacji z serwerem aplikacji za pośrednictwem bezpiecznego połaczenia.
- **Keycloak:** Do zarzadzania uwierzytelnianiem i autoryzacja w aplikacji używamy Keycloak. Keycloak jest skonfigurowany do pracy w trybie "client credentials", gdzie autoryzacja jest oparta na identyfikatorze i tajnym kluczu klienta, a nie na danych użytkownika. Tokeny uwierzytelniajace sa generowane przez Keycloak i przekazywane do backendu, gdzie sa używane do zabezpieczania naszego API.
- **PostgreSQL:** Aplikacja korzysta z bazy danych PostgreSQL, która jest zarzadzana przez Keycloak. Baza danych jest przechowywana na stałym woluminie (PVC), co pozwala na zachowanie danych oraz ustawień dla Keycloak nawet po zatrzymaniu i ponownym uruchomieniu kontenera.

1.4 Konfiguracja i zarzadzanie - 4 pkt

Konfiguracja i zarzadzanie aplikacja na poziomie klastra Kubernetes odbywa sie na kilka sposobów:

- **Deploymenty:** Każdy komponent aplikacji (frontend, backend, Keycloak, baza danych PostgreSQL) jest uruchamiany jako osobny Deployment w Kubernetes. Deploymenty pozwalaja na łatwe skalowanie i aktualizacje komponentów aplikacji.
- **Serwisy:** Komunikacja miedzy komponentami aplikacji jest zarzadzana za pomoca Serwisów Kubernetes, które zapewniaja odkrywanie i balansowanie obciażenia.
- **ConfigMaps i Secrets:** Konfiguracja aplikacji jest przechowywana w ConfigMaps (Ingress, Keycloak, PostgreSQL) i Secrets (Backend, PostgreSQL). Pozwala to na centralne zarzadzanie konfiguracja i bezpieczne przechowywanie poufnych danych.
- **Persistent Volume Claims:** Dane bazy danych PostgreSQL sa przechowywane na Persistent Volume za pomoca Persistent Volume Claim, co zapewnia trwałość danych.
- **Ingress:** Dostep do aplikacji z zewnatrz klastra jest zarzadzany za pomoca Ingress, który kieruje ruch sieciowy do odpowiednich Serwisów.

1.5 Zarzadzanie błedami

- **Obsługa błedów na poziomie aplikacji:** W backendzie i frontendzie korzystam z bloków ‘try catch ‘ do przechwytywania i obsługi błedów. Dzieki temu, nawet w przypadku nieoczekiwanych błedów, aplikacja nie przestaje działać.
- **Zarzadzanie błedami na poziomie infrastruktury:** W konfiguracji Deployment dla backendu, korzystam z mechanizmu replikacji Kubernetes. Dzieki temu, jeśli jeden z podów aplikacji napotka bład i przestanie działać, Kubernetes automatycznie przekieruje ruch do drugiego poda.

1.6 Skalowalność - 4 pkt

Skalowalność jest kluczowa w architekturze aplikacji opartej na Kubernetes.

- **Skalowanie horyzontalne:** Aplikacja jest skalowana horyzontalnie poprzez zwiekszanie liczby replik podów na backendzie. Obecnie liczba replik na backendzie została zwiekszona do 2, co pozwala na lepsze rozłożenie obciażenia i zwieksza przepustowość. Kubernetes automatycznie rozprowadza ruch miedzy replikami, zapewniajac równomierne obciażenie.

1.7 Wymagania dotyczace zasobów - 2 pkt

- **Frontend:** 300MB pamieci RAM, 0.2 rdzenia CPU
- **Backend:** 600MB pamieci RAM, 0.2 rdzenia CPU (na replike)
- **PostgreSQL:** 256MB pamieci RAM, 0.15 rdzenia CPU
- **Keycloak:** 2000MB pamieci RAM, 1 rdzeń CPU

1.8 Architektura sieciowa - 4 pkt

Architektura sieciowa aplikacji opiera sie na modelu sieciowym Kubernetes, który umożliwia komunikacje miedzy podami oraz z zewnetrznymi usługami sieciowymi.

- **Ingress:** Wszystkie usługi (frontend, backend, Keycloak) sa dostepne poprzez Ingress, który zarzadza dostepem do usług w klastrze z zewnatrz. Ingress umożliwia routowanie ruchu sieciowego do odpowiednich usług na podstawie adresu URL i metody HTTP.
- **Protokoły:** Aplikacja korzysta z protokołu HTTP do komunikacji miedzy frontendem a backendem. Do zarzadzania stanem sesji i autoryzacji używany jest protokół OAuth2 z Keycloak.