

APLIKACJA DO WYSYŁKI INFORMACJI W MODELU ASYNCHRONICZNYM

Karol Piasecki 6126

Problem inżynieryjny i jego sposób rozwiązania

Problem:

Wysyłka informacji do użytkownika w czasie zbliżonym do rzeczywistego, korzystając z asynchronicznego przepływu danych

Próba jego rozwiązania:

Stworzenie systemu na bazie architektury mikroservisów oraz z użyciem asynchronicznego message brokera Apache Kafka.

Cel i zakres pracy

- Aplikacje typu mikroservis, serwer główny Core oraz serwer Persistence
- Aplikacja webowa, stworzona z użyciem biblioteki React
- Logowanie/Rejestracja dla każdego z użytkowników
- Możliwość podglądu wiadomości
- Synchronizacja wiadomości z użyciem protokołu websocket
- Asynchroniczny zapis wiadomości przy użyciu Kafki
- Infrastruktura Kafki, poprawne skonfigurowanie ZooKeepera oraz samego serwisu Apache Kafka

Wymagania aplikacji

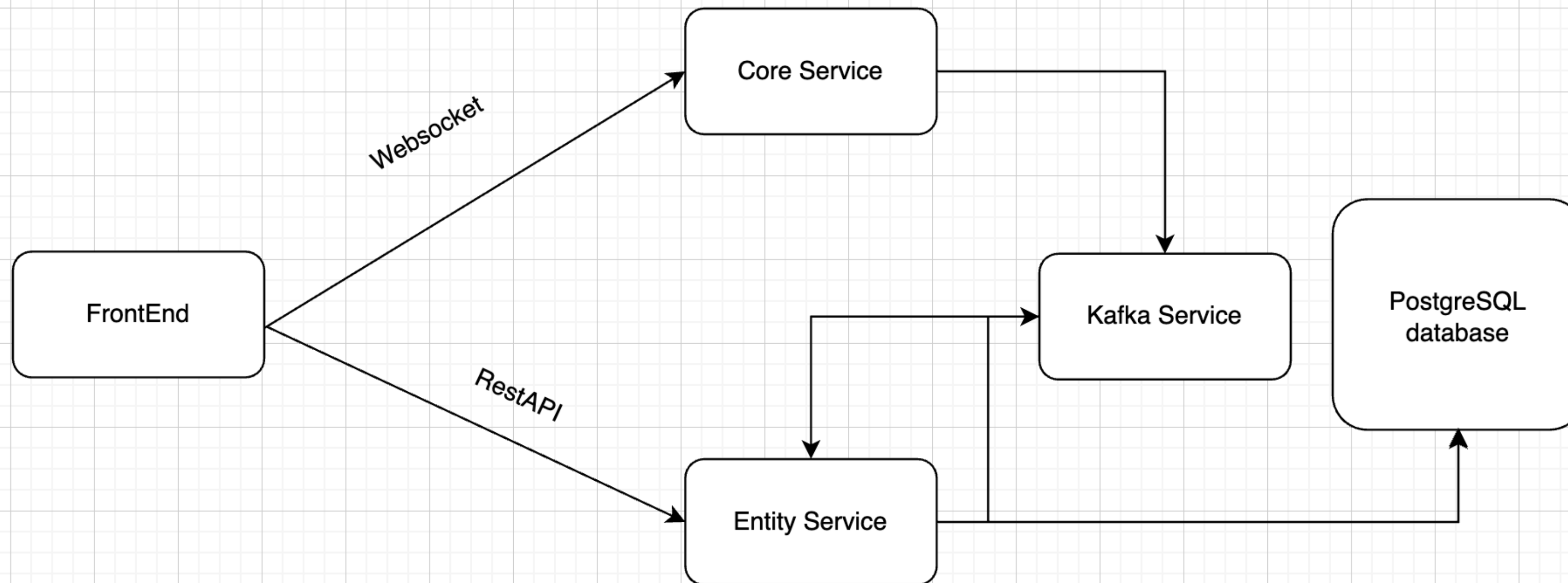
Wymagania funkcjonalne to między innymi:

- Implementacja połączenia Websocket z mikroserwisem*
- Wystawienie konkretnego API do pobrania wiadomości*
- Kompletne skonfigurowanie środowiska Apache Kafka wraz z serwerem biznesowym*
- Możliwość zapisu jak i odczytu wiadomości poprzez odrębny mikroserwis zajmujący się zarządzaniem relacyjną bazą danych*
- Wyszukiwanie użytkowników po nadanych im identyfikatorach*
- Rejestracja oraz możliwość zalogowania się do serwisu poprzez użycie odpowiednich danych uwierzytelniających*

Wymagania niefunkcjonalne to między innymi:

- Napisana oprawa graficzna w języku JavaScript korzystając z frameworku React*
- Skalowalne okienko chatu dla tabletów oraz aplikacji mobilnych*
- Panel logowania oraz rejestracji powszechnie dostępny dla użytkowników*

Architektura aplikacji



FrontEnd - aplikacja służąca do wyświetlania graficznego interfejsu użytkownika

CoreService - jeden z mikroservisów który nawiązuje połączenie z klientem poprzez protokół WebSocket

EntityService - jeden z mikroservisów który zarządza zapisem oraz przesyłaniem (starszych) wiadomości

KafkaService - serwis infrastrukturalny przez który przechodzą informacje i trafiają do konkretnego mikroservisów aby następnie zostały zapisane w bazie danych PostgreSQL

Narzędzia i technologie

Technologie

- Spring, Kafka, Spring Boot, Webflux, Hibernate, JPA
- ReactJS, Material UI
- PostgreSQL


Narzędzia

- IntelliJ Idea
- Postman

Interfejs oraz funkcjonalności

ChatChannel

O PROJEKCIEZALOGUJ



Zaloguj się

Login *

Hasło *


ZALOGUJ SIĘ

[Zarejestruj się](#)

Interfejs oraz funkcjonalności

ChatChannel

O PROJEKCIEZALOGUJ



Zarejestruj sie

Login *

Hasło *

Imię *

REJESTRACJA

[Zaloguj sie](#)

Interfejs oraz funkcjonalności

ChatChannel

WIADOMOSCI

O PROJEKCIE

WYLOGUJ

Karol

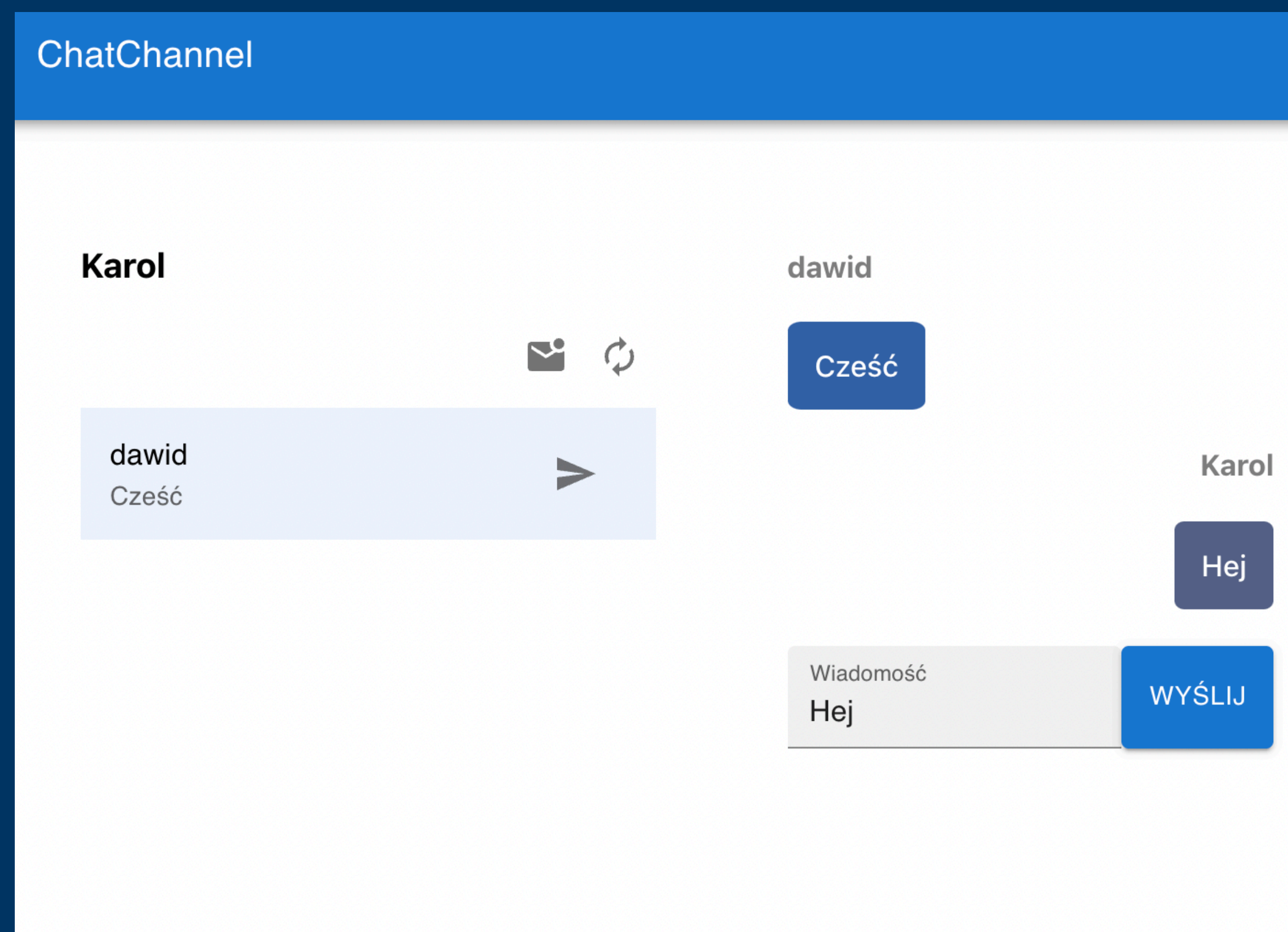
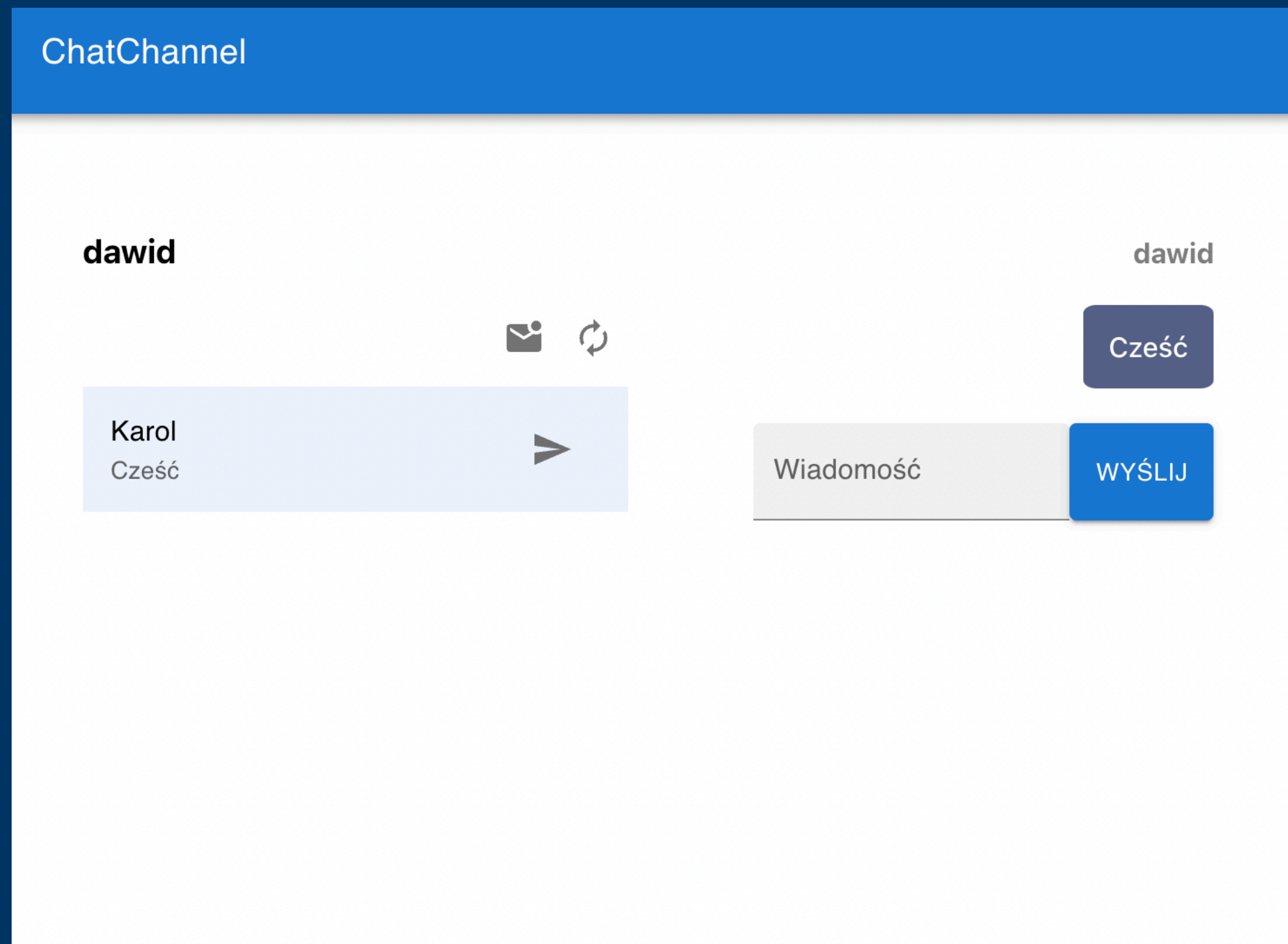
dawid

Inicjuje chat

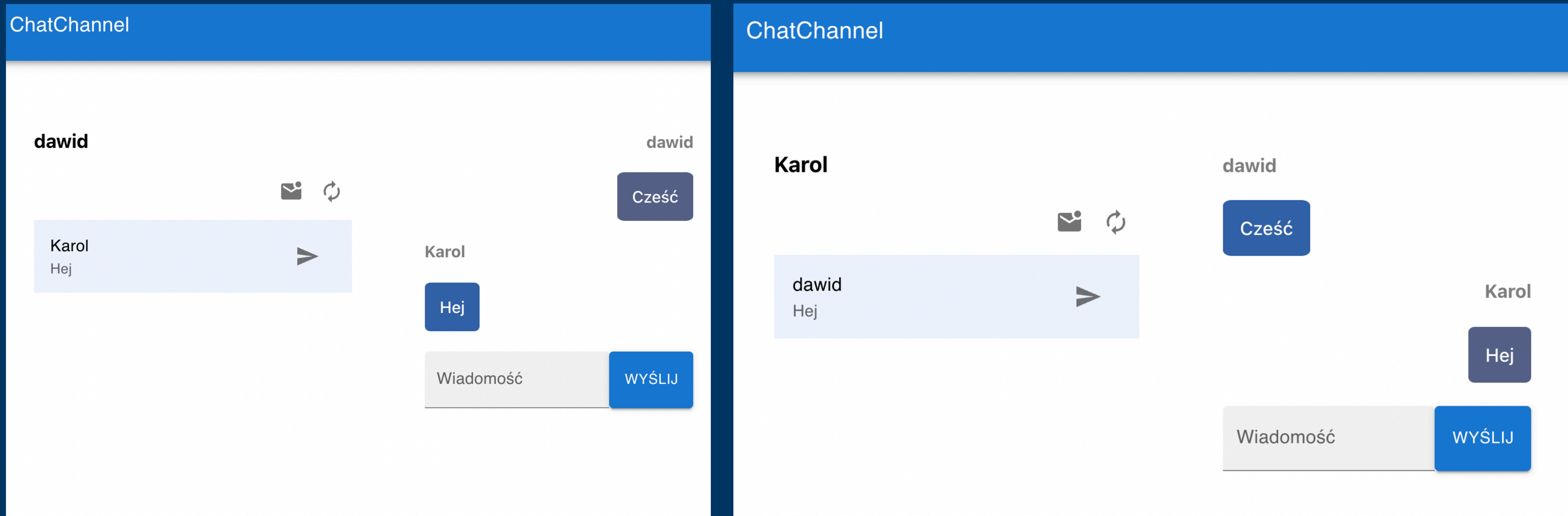
Kafka consumer-0-1

partition-revoked-latency-max	NaN
fetch-throttle-time-max	0
count	140
connection-close-rate	0
incoming-byte-rate	5.601664023724695
time-between-poll-max	5002
outgoing-byte-total	5340
response-rate	0.3501040014827934
preferred-read-replica	-1
rebalance-total	1
request-total	526
outgoing-byte-total	30681
commit-rate	0

Interfejs oraz funkcjonalności



Interfejs oraz funkcjonalności



INFO 80926 --- [ntainer#0-0-C-1] c.r.server.service.SaveMessagesService :
Message text Hej to 4 from 3
INFO 80926 --- [ntainer#0-0-C-1] c.r.server.service.SaveMessagesService :
Save next message, from user 4 to user 3

Podsumowanie

System do wysyłki informacji przedstawiony przez autora pracy realizuje wszystkie postawione w niej wymagania oraz rozwiązuje określony problem.