BO·TT·EGA
IT minds

Program szkolenia:

Apache Kafka - niezbędnik programisty Python

Informacje:

Nazwa:	**Apache Kafka - niezbędnik programisty Python**
Kod:	**python-kafka**
Kategoria:	Python
Odbiorcy:	developerzy, architekci
Czas trwania:	4 dni
Forma:	50% wykłady / 50% ćwiczenia

W trakcie tego praktycznego warsztatu zapoznamy się z narzędziami i bibliotekami, które pozwolą programistom Pythona w efektywny sposób implementować oraz testować systemy wykorzystujące Apache Kafka.

Przedstawione zostaną innowacyjne sposoby implementacji, w których Kafka to nie tylko broker wymiany wiadomości, ale też narzędzie do trwałego zapisywania i odczytywania danych, ich agregacji oraz strumieniowania. Omówimy sposoby integracji danych z klastra z popularnymi bazami danych oraz sposoby łączenia ze sobą strumieni danych.

W części warsztatowej położymy nacisk na testowanie automatyczne wytworzonego kodu, zgodność z wzorcami projektowymi oraz dobrymi praktykami wytwarzania oprogramowania.

Szkolenie to niezbędnik programistyczny dla każdego dewelopera rozpoczynającego swoją przygodę z implementacją aplikacji posługujących się Kafką. Jest to zbiór najbardziej efektywnych praktyk zebranych na bazie doświadczenia w pracy z systemami wykorzystującymi Apache Kafka.

BOTTEGA IT minds

Szczegółowy program:

1. Architektura Apache Kafka

- 1.1. Wprowadzenie do komunikacji asynchronicznej
  - 1.1.1. Komunikacja w monolicie vs mikrousługi
  - 1.1.2. Przykłady zastosowań komunikacji asynchronicznej
  - 1.1.3. Systemy oparte o zdarzenia
- 1.2. Architektura Apache Kafka
  - 1.2.1. Wprowadzenie do podstawowych pojęć
    - 1.2.1.1. Topics, Partitions, Offsets
    - 1.2.1.2. Brokers
    - 1.2.1.3. Topics Replication
    - 1.2.1.4. Producers
    - 1.2.1.5. Consumers, Consumers Groups
    - 1.2.1.6. Offsets
    - 1.2.1.7. Acks
  - 1.2.2. Kolejność wiadomości
  - 1.2.3. Replikacja
  - 1.2.4. Jak dobrać odpowiednią ilość partycji?
  - 1.2.5. Poziomy potwierdzenia dostarczenia wiadomości
  - 1.2.6. Co zapewnia Apache Kafka
  - 1.2.7. Skąd bierze się wydajność?
- 1.3. Architektura Confluent Platform
  - 1.3.1. Kafka Streams
  - 1.3.2. Kafka Connect

BOTTEGA IT minds

- 1.3.3. KSQL
- 1.3.4. Rest Proxy
- 1.3.5. Schema Registry

1.4. Kafka CLI

- 1.4.1. Zarządzanie Topic'ami
- 1.4.2. Wysyłanie wiadomości
- 1.4.3. Odbieranie wiadomości
- 1.4.4. Monitorowanie i zarządzanie Consumer Groups
- 1.4.5. Zarządzanie Offsetami

1.5. Pythonowy klient confluent_kafka

- 1.5.1. Kafka Producer
  - 1.5.1.1. Implementacja niestandardowego partycjonowania
  - 1.5.1.2. Serializacja wiadomości
  - 1.5.1.3. Idempotentność
- 1.5.2. Kafka Consumer
  - 1.5.2.1. Delivery semantics
  - 1.5.2.2. Deserializacja wiadomości
  - 1.5.2.3. Partition assignment strategies
  - 1.5.2.4. Static membership
  - 1.5.2.5. Implementacja cache w pamięci opartego o Kafka Consumer
- 1.5.3. Transakcje
  - 1.5.3.1. Jak działają transakcje?
  - 1.5.3.2. Jak dobrać parametr transaction.id?
  - 1.5.3.3. Do czego wykorzystać transakcje na Apache Kafka?
  - 1.5.3.4. Implementacja aplikacji biznesowej przetwarzającej wiadomości w sposób transakcyjny

BOTTEGA IT minds

1.5.4. Jak skonfigurować system do bezstratnego dostarczania wiadomości?

2. Przetwarzanie strumieni zdarzeń, Event Streaming

2.1. Koncepcja zastosowania w ekosystemie Confluent

2.2. Architektura aplikacji opartych o pythonową implementację Kafka Streams, Faust-streaming

2.3. Dualizm strumień-tablica

2.3.1. Lokalna baza danych RocksDB

2.3.2. Repartition topic

2.3.3. Changelog logic

2.4. Analiza topologii strumieni

2.5. Reset tool

2.6. Operacje bezstanowe

2.7. Operacje stanowe

2.7.1. Tablice

2.7.2. Agregowanie danych

2.7.3. Łączenie strumieni

2.7.4. Agregacja danych w oknach czasowych

2.8. Przetwarzanie zgodnie z podejściem Stateful Record-By-Record

2.9. Exactly once processing

2.10. Testowanie aplikacji opartych o pythonową implementację Kafka Streams, Faust-streaming

2.10.1. Testy jednostkowe

2.10.2. Testy integracyjne

2.11. Implementacja wzorców

2.11.1. Saga

2.11.2. Read-Process-Write

BOTTEGA IT minds

2.11.3. Sortowanie i deduplikacja zdarzeń

3. Wykorzystanie Apache Kafki z frameworkami webowymi na przykładzie FastAPI

3.1. Podstawowe operacje

3.1.1. Wysyłka wiadomości

3.1.1.1. tryby wysyłania wiadomości

3.1.1.2. obsługa potwierdzeń i błędów

3.1.2. Odbiór wiadomości

3.1.2.1. Wielowątkowość

3.1.2.2. Walidacja wiadomości

3.1.2.3. Zarządzanie potwierdzeniami

3.1.2.4. Ponawianie

3.1.2.5. Wysyłka odpowiedzi

3.1.3. Transakcje

3.1.3.1. Publikowanie transakcyjne

3.1.3.2. Transakcje inicjowane przez odczyt wiadomości

3.1.3.3. Zarządzanie transaction.id

3.2. Prawidłowa konfiguracja aplikacji

3.2.1. Zarządzanie błędami

3.2.2. Ustawienia klienta Kafki

3.2.3. Serializacja i deserializacja

3.2.4. Nagłówki

3.3. Testowanie

3.3.1. Przegląd możliwości testowania asynchronicznego z pytest

3.3.2. Uruchamianie Kafka do testów z wykorzystaniem docker i docker compose

3.3.3. Uruchamianie Kafka do testów z wykorzystaniem Testcontainers

BOTTEGA IT minds

3.4. Implementacja aplikacji biznesowej z wykorzystaniem FastAPI, Apache Kafka i relacyjnej bazy danych

3.4.1. Modelowanie procesów biznesowych w podejściu asynchronicznym

3.4.2. Implementacja mechanizmów zapewniających idempotentność

3.4.3. Wpływ asynchroniczności na modelowanie REST API

3.4.4. Różne poziomy spójności

3.4.5. Implementacja wzorca Outbox