НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені ІГОРЯ СІКОРСЬКОГО”

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №3

з предмету «Проектування розподілених систем»

Виконав:

студент групи ІМ-31мн

Рекечинський Дмитро

Київ 2024

Завдання

* Реалізувати Enterprise паттерн асинхронної комунікації:
  + Event Sourcing модель з механізмом матеріалізації сутності-

АБО

* + 2-phase / 3-phase commit з використанням Брокера Повідомлень і 2ма Постачальниками Сервісу

Виконання завдання

Проект було створено з реалізацію паттерну Enterprise на основі моделі Event Sourcing.

Для виконання завдання було створено 4 сервіси:

* Споживач сервісу consumer-service
* Постачальник сервісу provider-service
* Брокер повідомлень RabbitMQ (rabbitmq)
* Сервіс БД PostgreSQL (postgres)

Проблема з відстеженням статусу сервісу, яка виникла у 2 лабораторній роботі з RabbitMQ, поширилась також на PostgreSQL. Це вирішується за допомогою додавання healthcheck, і в Інтернеті радили використати для цього команду pg\_isready.

Таким чином, конфігурація для healthcheck в PostgreSQL виглядає так:

healthcheck:

test: pg\_isready

interval: 10s

timeout: 5s

retries: 10

start\_period: 5s

Залежні контейнери налаштовуються у такий же спосіб, як описано у 2 лабораторній роботі:

depends\_on:

rabbitmq:

condition: service\_healthy

postgres:

condition: service\_healthy

Втім, цей healthcheck працює не так, як слід... Кожні секунд 5 виникає помилка «FATAL: role "root" does not exist».

Подальше дослідження показало ось що:

* Якщо в команді pg\_isready не вказано користувача явно, тоді перевірка виконується для поточного користувача.
* Команди в середовищі контейнера запускаються від користувача root, який має розширені права.
* За умовчуванням, PostgreSQL має користувача postgres. Але оскільки ми вказали змінну середовища POSTGRES\_USER: user, то створюється користувач user. З цього випливає, що користувача root не передбачено жодним чином.

Це можна виправити за допомогою явного вказування користувача та бази даних у команді healthcheck:

healthcheck:

test: "pg\_isready -d $${POSTGRES\_DB} -U $${POSTGRES\_USER}"

interval: 10s

timeout: 5s

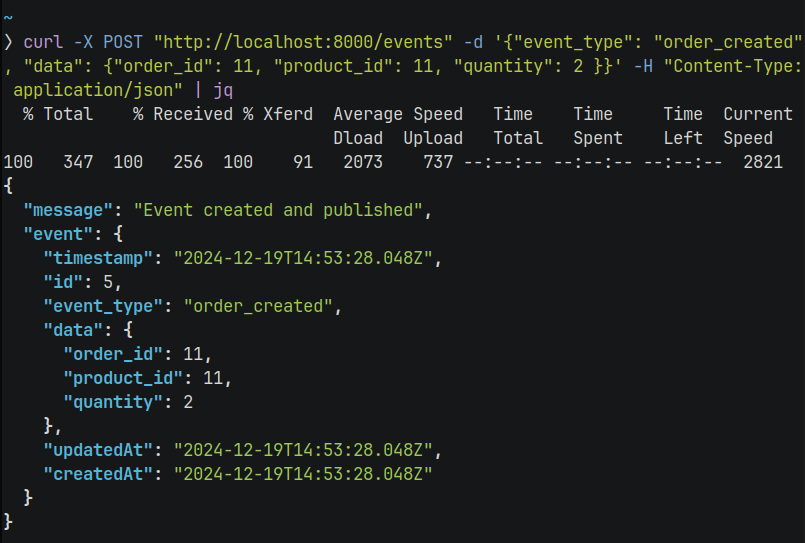
retries: 10

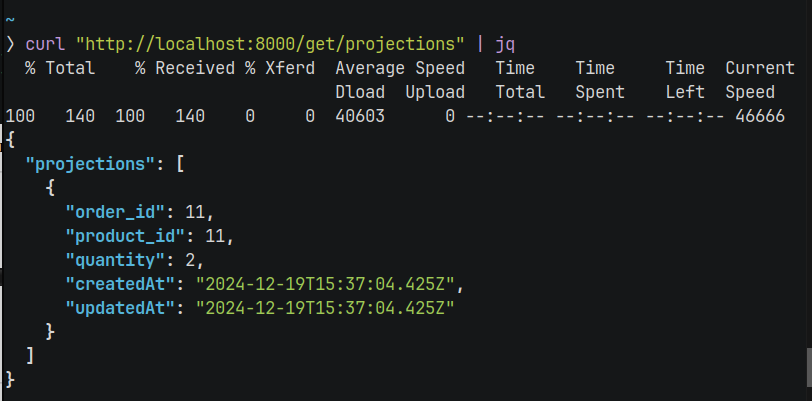
start\_period: 5s

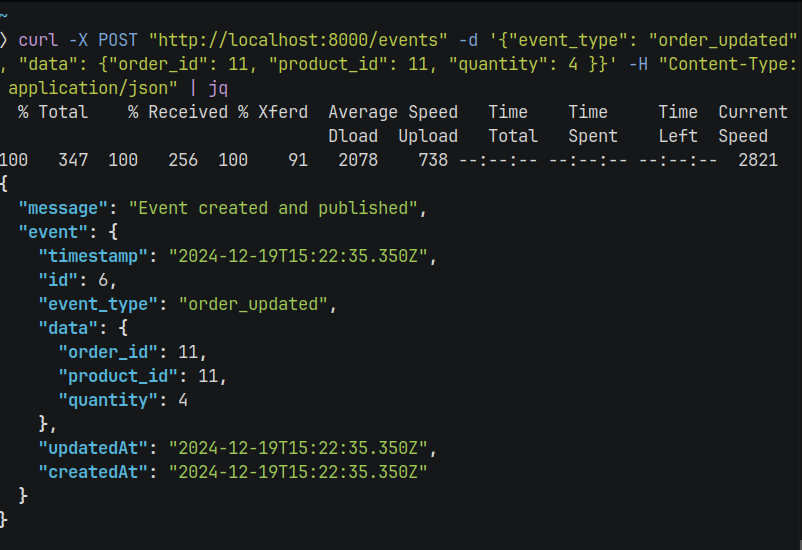
Демонстрація результатів

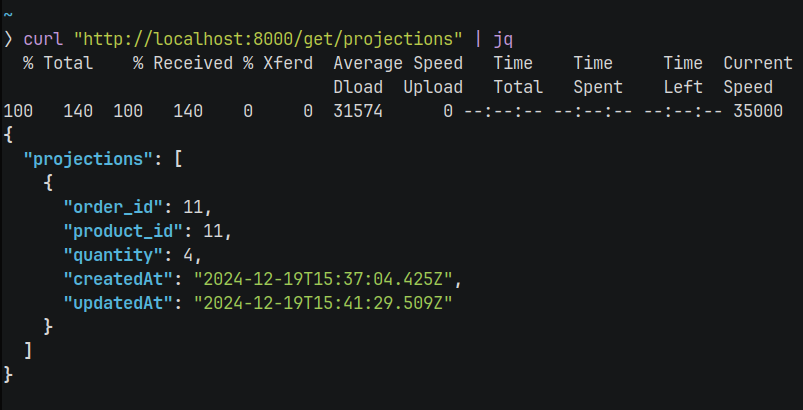
Рис. 1 — Відображення всіх подій

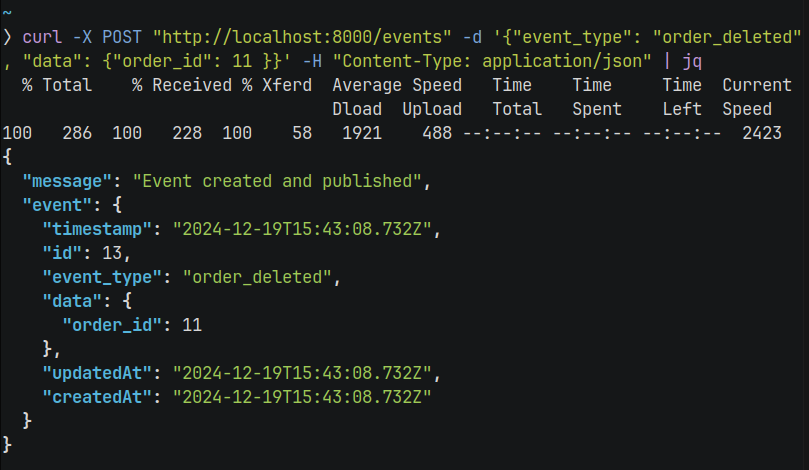
Рис. 2 — Відображення всіх сутностей

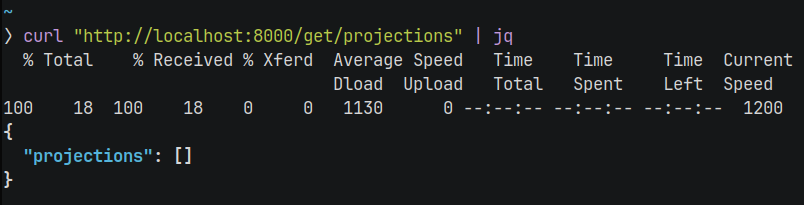
Рис. 3.1 — Створення нової сутності

Рис. 3.2 — Результат виконання

Рис. 4.1 — Оновлення сутності

Рис. 4.2 — Результат виконання

Рис. 5.1 — Видалення сутності

Рис. 5.2 — Результат виконання

Повна версія коду проекту розміщена за веб-адресою: <https://github.com/rocket111185/distribution-systems/tree/release/lab3>