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,

Онищук Микола

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**

Лабораторну роботу було виконано на мові Golang та розгорнуто у Docker. Також було використано RabbitMQ в якості брокера повідомлень для комунікації між споживачем та постачальником та PostgreSQL у якості бази даних.

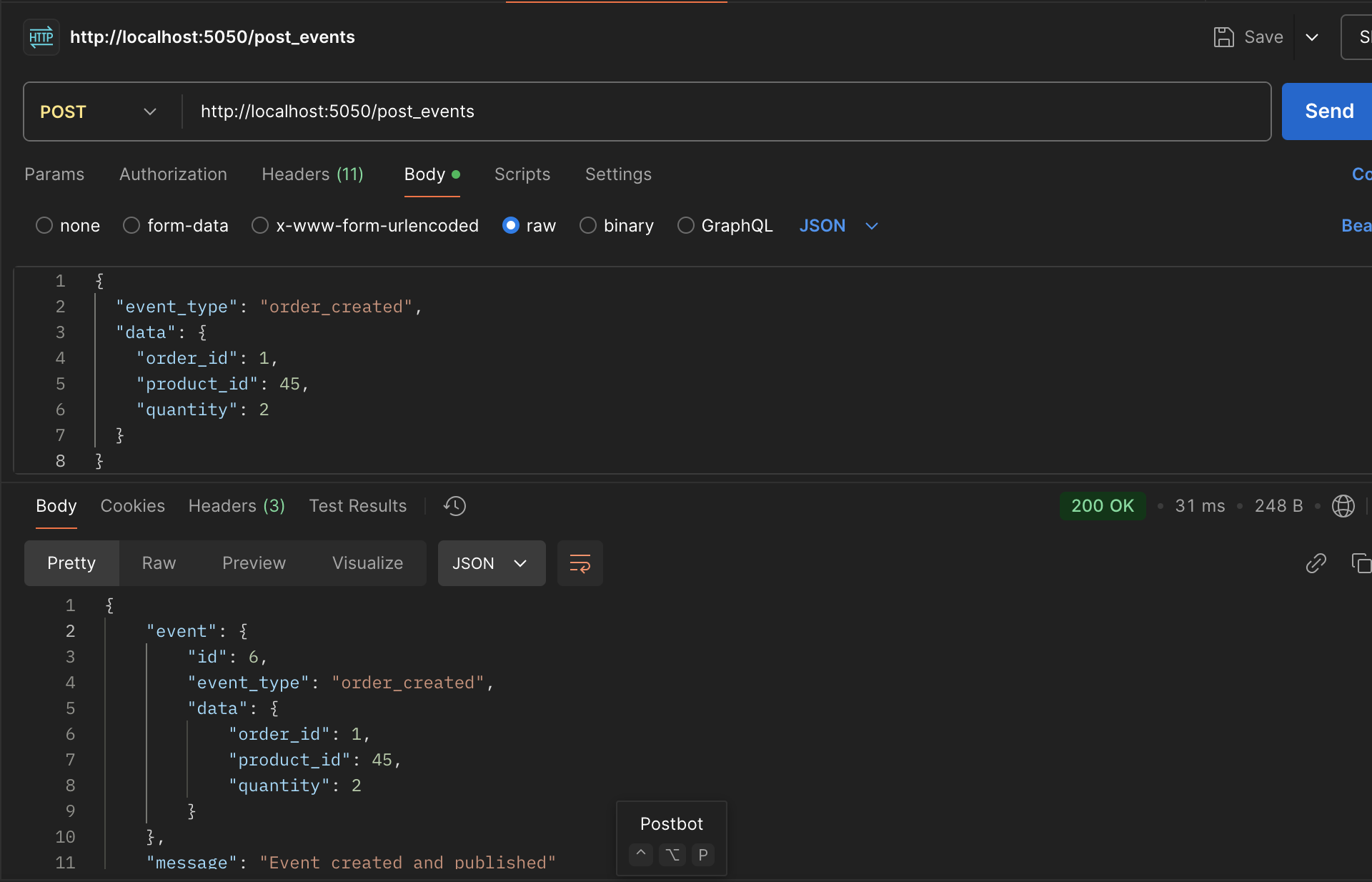
Розроблена система містить по одному інстансу споживача та постачальника, а також брокер повідомлень для спілкування між ними та сервіс бази даних.

В даній роботі Enterprise патерн асинхронної комунікації було реалізовано за допомогою моделі Event Sourcing.

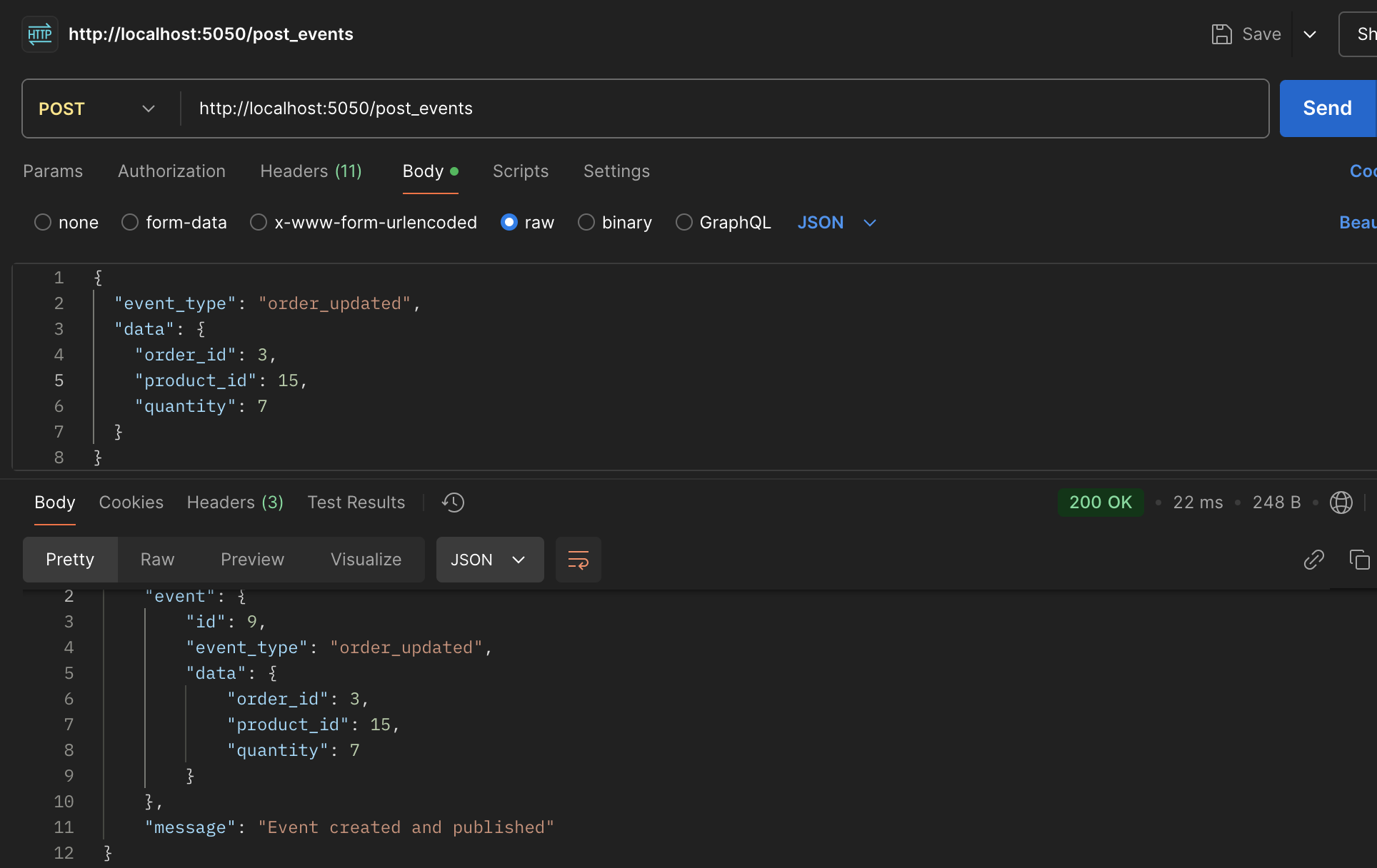
Сервіс споживача має 3 ендпоінти - “/post\_events” для створення та публікації нової події, “/get\_events” для отримання історії подій та “/get\_orders\_details” для перегляду актуального стану замовлень, створених чи оновлених на основі подій. Постачальник не має ендпоінтів, адже спілкування між ним та споживачем відбувається за допомогою Event Sourcing моделі: споживач записує події у базу даних і публікує їх у чергу RabbitMQ, а постачальник отримує ці події з черги, обробляє їх і оновлює стан своїх даних відповідно до отриманих змін.

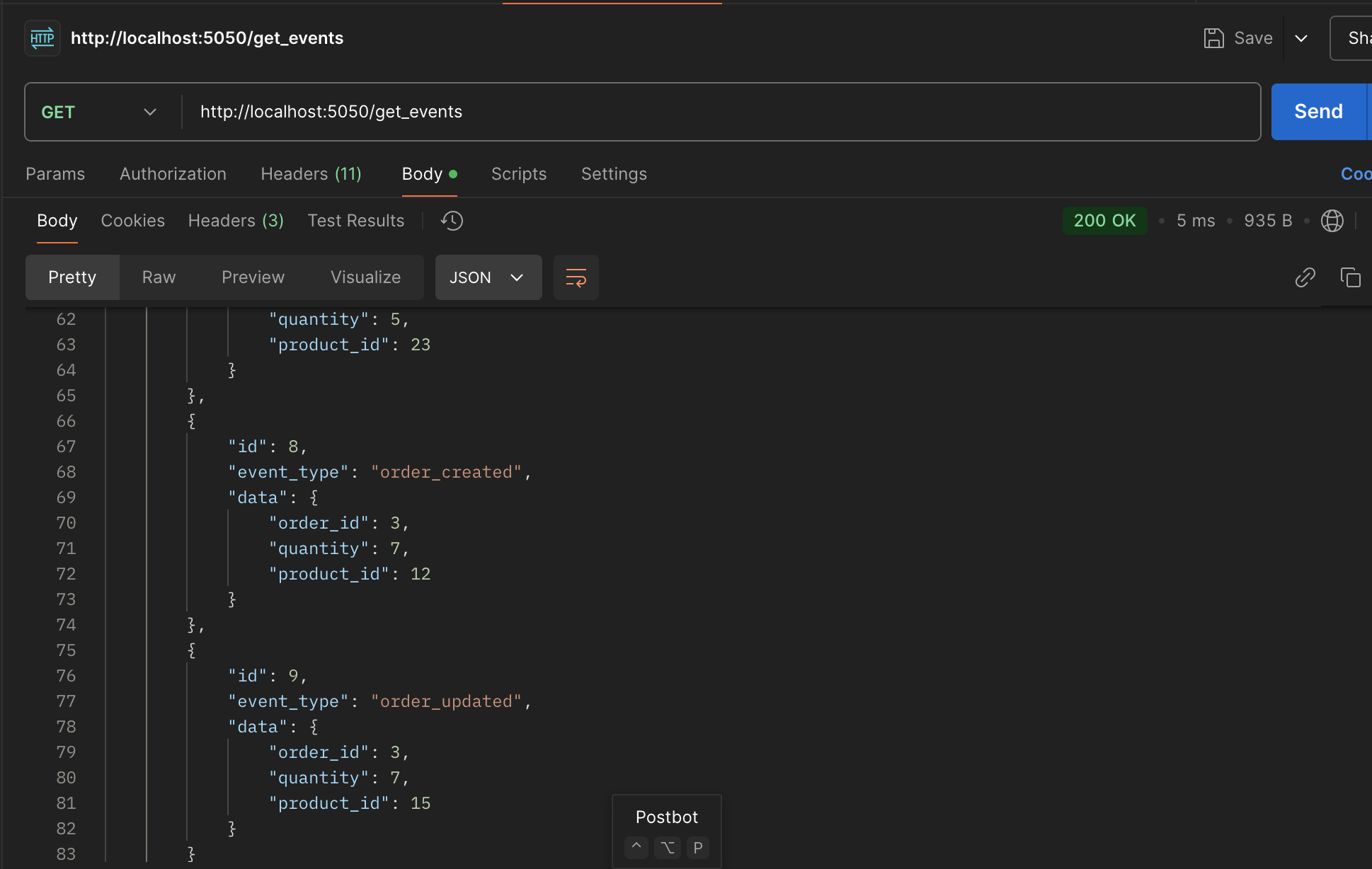
Протестуємо роботу системи, надсилаючи запити за допомогою “Postman”.

Надішлемо POST-запит за адресою <http://localhost:5050/post_events> та отримаємо результат у вигляді створеної відповідної події та повідомлення про її створення (рис. 1). Тіло запиту повинно містити тип події, а також ідентифікатори, замовлення, товару та кількість одиниць товару.

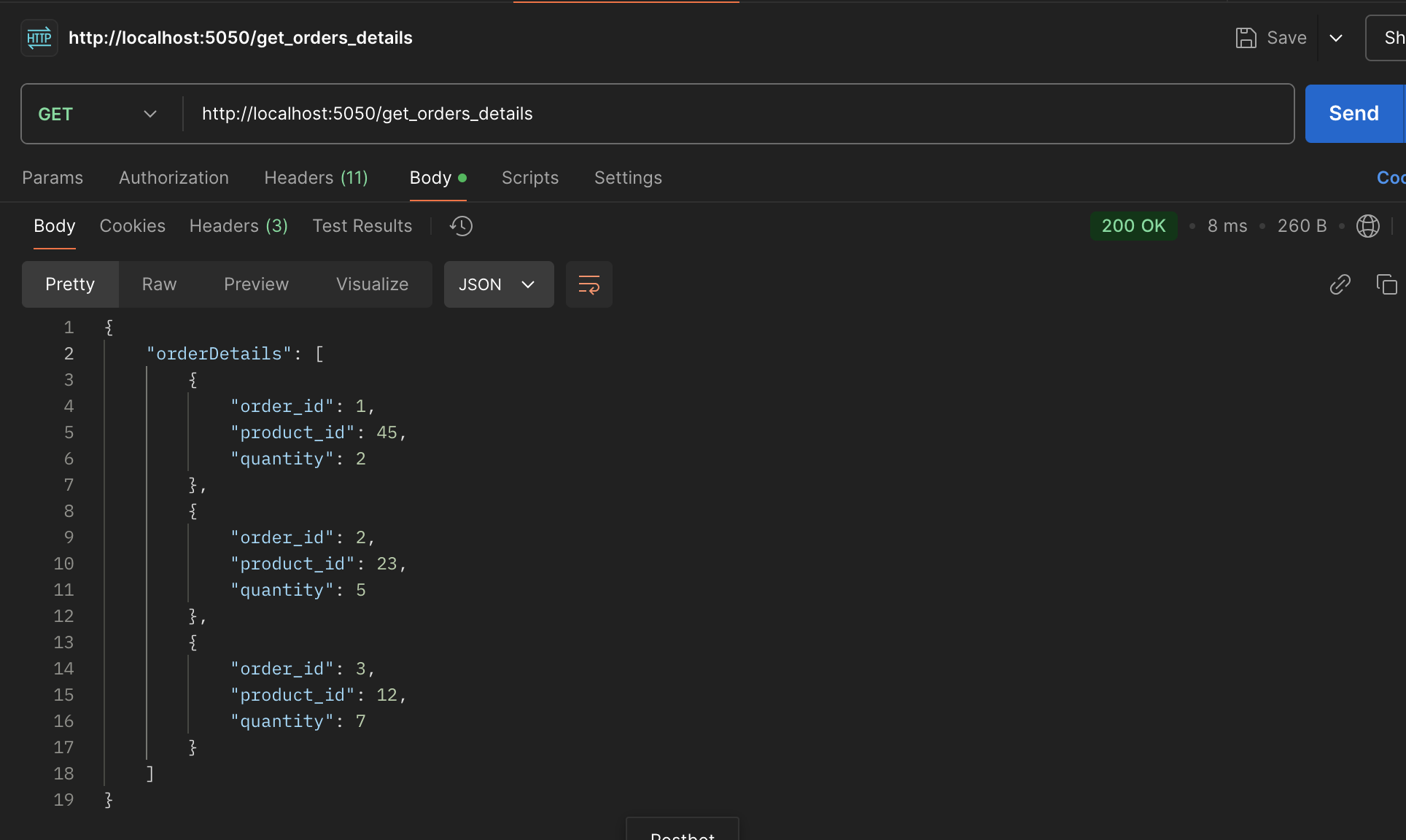
Рисунок 1 - Відправка запиту про завдання.

Після відправки кількох таких запитів (рис. 2), надішлемо GET-запит на [http://localhost:5050/get\_events](http://localhost/%get_events) (рис. 3). Так ми отримаємо список усіх подій, що відбулися.

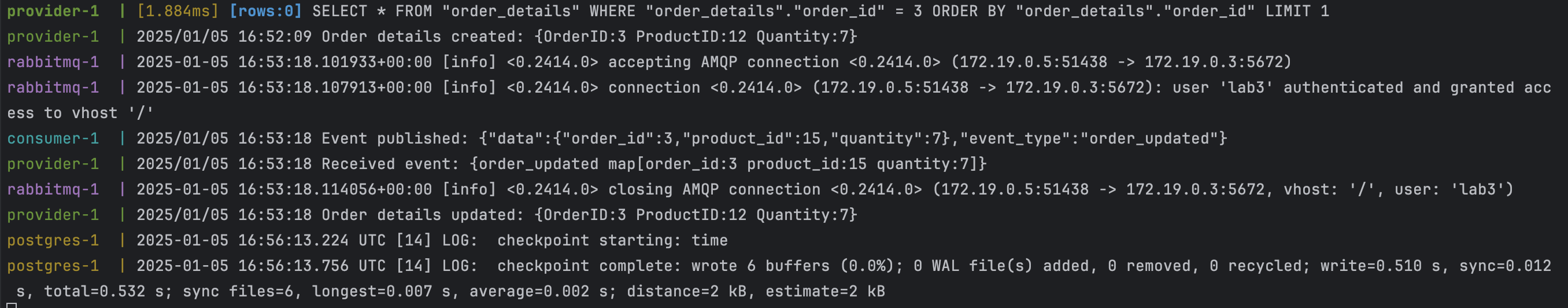
Рис. 3 - Відправка події типу ”order\_updated”.

Рисунок 4 - Відправка запиту про отримання подій.

Тепер відправимо GET-запит на http://localhost:5050/get\_orders\_details для отримання поточних даних про всі замовлення (рис. 5).

Рисунок 5 - Відправка запиту для отримання поточної інформації про замовлення.

Внаслідок надісланих запитів було отримано наступні логи (рис. 6).

Рисунок 6 - Логи при виконанні запитів.