МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Лабораторна робота №2

з дисципліни «Проектування розподілених систем»

Виконав:

Студент групи ІО – 31мн

Дмитришин Андрій Дмитрович

Перевірив:

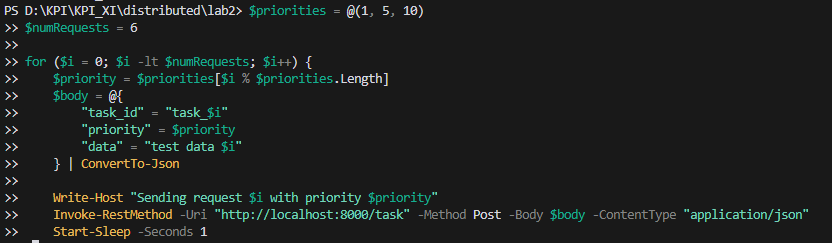
Обозний Д. М.

Київ 2024

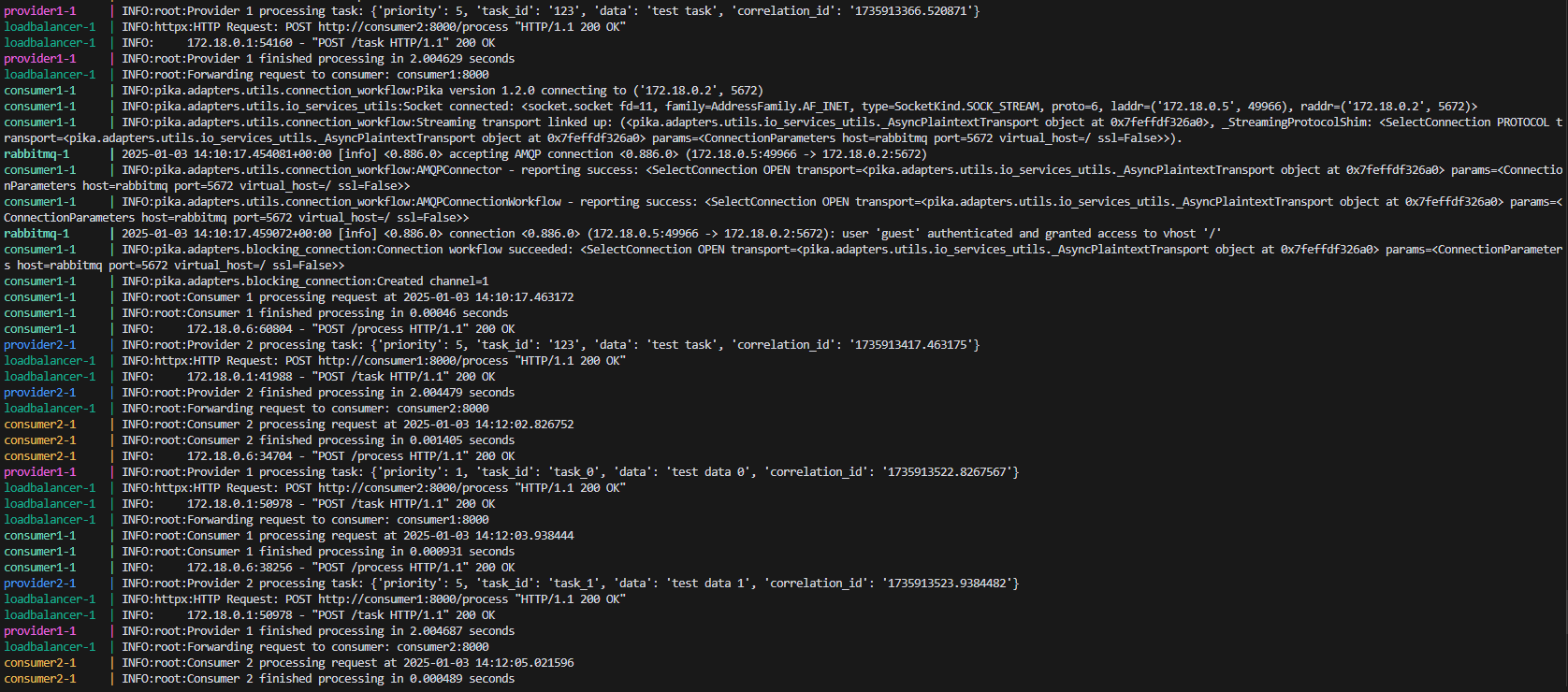
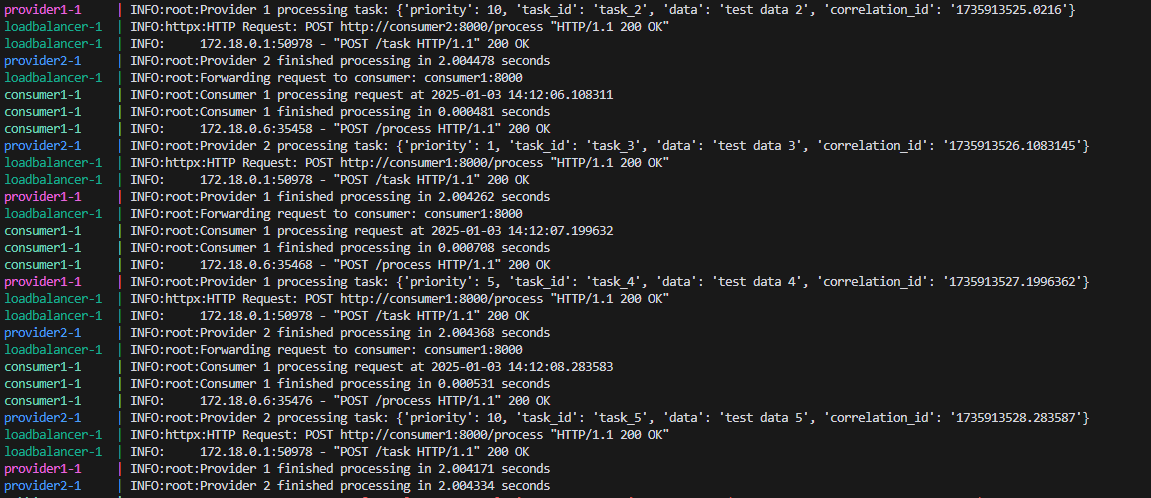
Лабораторна робота 2

* Реалізувати асинхронну комунікацію між Постачальником Сервісу і Споживачем Сервісу за допомогою Брокера Повідомлень
* Постачальник Сервісу має підраховувати час обчислення і логувати його для подальшого аналізу
* Cпоживач Cервісу має підраховувати час виконання запиту і логувати його для подальшого аналізу
* Реалізувати горизонтальне масштабування засобами Брокера Повідомлень
* Реалізувати чергу с пріоритетами
* Реалізувати request-reply паттерн в асинхронній комунікації
* Порівняти результати синхронної і асинхронної комунікації

Запит



Логи

**1. Реалізувати асинхронну комунікацію між Постачальником Сервісу і Споживачем Сервісу за допомогою Брокера Повідомлень**

Реалізація асинхронної комунікації здійснена через RabbitMQ з використанням бібліотеки pika. У логах видно, що використовуються асинхронні з'єднання:

consumer2-1 | INFO:pika.adapters.utils.connection\_workflow:Streaming transport linked up: ...

Це підтверджує використання асинхронного транспорту \_AsyncPlaintextTransport.

**2. Постачальник Сервісу має підраховувати час обчислення і логувати його для подальшого аналізу**

Постачальники логують час виконання завдань:\

provider1-1 | INFO:root:Provider 1 finished processing in 2.004629 seconds

provider2-1 | INFO:root:Provider 2 finished processing in 2.004479 seconds

Час обчислення вимірюється коректно.

**3. Cпоживач Cервісу має підраховувати час виконання запиту і логувати його для подальшого аналізу**

Споживачі також логують час обробки запитів:

consumer2-1 | INFO:root:Consumer 2 finished processing in 0.00048 seconds

consumer1-1 | INFO:root:Consumer 1 finished processing in 0.00046 seconds

Це дозволяє аналізувати ефективність обробки на рівні споживачів.

**4. Реалізувати горизонтальне масштабування засобами Брокера Повідомлень**

Горизонтальне масштабування реалізовано через декілька споживачів (consumer1, consumer2), які отримують повідомлення з однієї черги RabbitMQ. Розподіл завдань між споживачами відображається у логах:

loadbalancer-1 | INFO:root:Forwarding request to consumer: consumer1:8000

loadbalancer-1 | INFO:root:Forwarding request to consumer: consumer2:8000

**5. Реалізувати чергу з пріоритетами**

Пріоритети завдань вказані у даних логів. Наприклад:

provider1-1 | INFO:root:Provider 1 processing task: {'priority': 5, 'task\_id': '123', 'data': 'test task', ...}

provider1-1 | INFO:root:Provider 1 processing task: {'priority': 10, 'task\_id': 'task\_2', ...}

Завдання обробляються відповідно до їх пріоритету.

**6. Реалізувати request-reply паттерн в асинхронній комунікації**

Request-reply паттерн реалізований. Логи показують, що запити пересилаються від load balancer до відповідних споживачів, а відповіді повертаються до постачальників:

loadbalancer-1 | INFO:httpx:HTTP Request: POST http://consumer1:8000/process "HTTP/1.1 200 OK"

provider1-1 | INFO:root:Provider 1 finished processing in 2.004687 seconds

**7. Порівняти результати синхронної і асинхронної комунікації**

**Синхронна комунікація:**

* Кожна задача виконується послідовно.
* Час очікування відповіді блокує виконання інших задач.
* Підходить для простих систем з низьким рівнем навантаження.

**Асинхронна комунікація (результати тестування):**

* Завдання розподіляються між споживачами, що дозволяє обробляти їх паралельно.
* Виміряний час обробки споживачів:Consumer 1: 0.00046 seconds

Consumer 2: 0.00048 seconds

* Час обробки постачальників:Provider 1: 2.004629 seconds

Provider 2: 2.004479 seconds

Асинхронна комунікація демонструє значно кращу продуктивність у високонавантажених сценаріях завдяки паралельній обробці.

**Загальний висновок**

Реалізація відповідає всім зазначеним вимогам:

1. Використана RabbitMQ для асинхронної комунікації.
2. Логування часу виконання реалізовано для всіх компонентів.
3. Підтримується горизонтальне масштабування та обробка завдань з пріоритетами.
4. Паттерн request-reply функціонує коректно.

Якщо потрібна додаткова оптимізація або тестування, уточніть деталі.