

Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Інститут комп'ютерних наук та інформаційних технологій
Кафедра автоматизованих систем управління



Звіт
до лабораторної роботи № 4
з дисципліни
Паралельні обчислення і розподілені системи
на тему:
«Мікросервіси»

Виконала: студентка ОІ-32
Горяча І. В.
Прийняв: асистент каф. АСУ
Копильчак О. А.

Львів – 2025

Мета: Оволодіти практичними прийомами розробки RESTful веб сервісів.

Послідовність роботи:

1. Розбийте систему, розроблену вами у лабораторній роботі No3 на окремі мікросервіси.
2. Мікросервіси можна запускати локально на окремих портах, або використати Docker контейнери.
3. Для демонстрації роботи програми використайте будь-який http клієнт, наприклад, Postman
4. Оформити звіт про роботу за такою структурою:
 - назва роботи;
 - мета роботи;
 - послідовність роботи;
 - індивідуальне завдання;
 - текст програми;
 - скріншоти з http запитами та відповідями від сервісу;
 - висновки.

Результати:

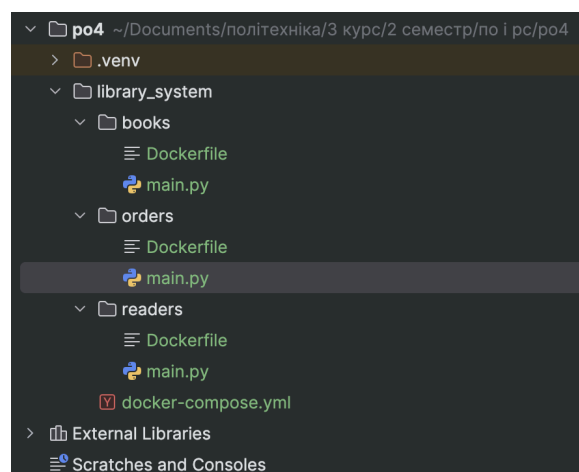
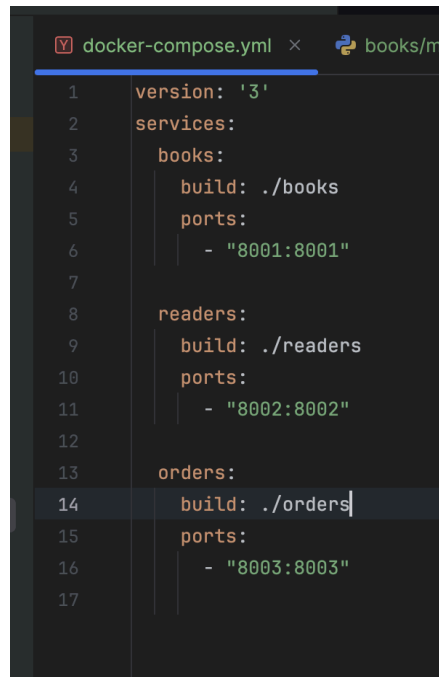


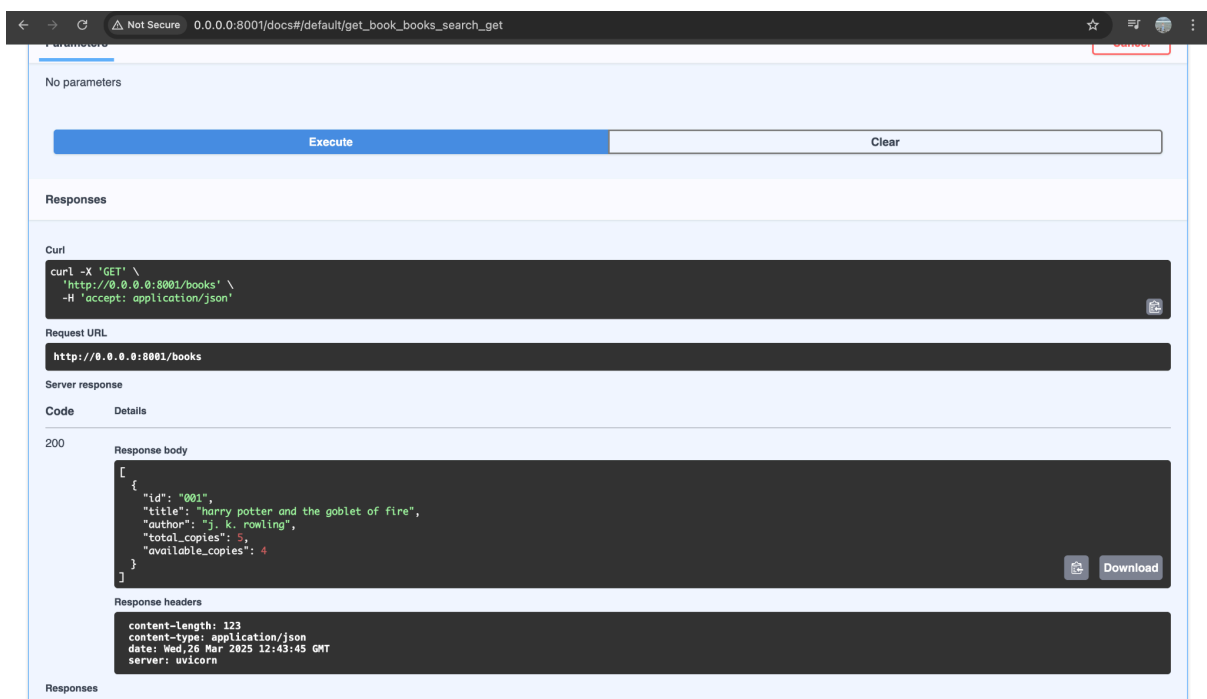
Рис.1. Отримана структура файлів із використанням Docker.

Основні сутності: книги, читачі та замовлення – діляться на три мікросервіси і запускаються на різних портах.



```
1 version: '3'
2 services:
3   books:
4     build: ./books
5     ports:
6       - "8001:8001"
7
8   readers:
9     build: ./readers
10    ports:
11      - "8002:8002"
12
13   orders:
14     build: ./orders
15    ports:
16      - "8003:8003"
17
```

Рис.2. Структура документу для запуску конфігурації мікросервісів за допомогою Docker.



Parameters

No parameters

Execute Clear

Responses

Curl

```
curl -X 'GET' \
  'http://0.0.0.0:8001/books' \
  -H 'accept: application/json'
```

Request URL

http://0.0.0.0:8001/books

Server response

Code Details

200

Response body

```
[
  {
    "id": "001",
    "title": "harry potter and the goblet of fire",
    "author": "j. k. rowling",
    "total_copies": 5,
    "available_copies": 4
  }
]
```

Response headers

```
content-length: 123
content-type: application/json
date: Wed, 26 Mar 2025 12:43:45 GMT
server: uvicorn
```

Responses

Рис.3. Створені книги, API запити на порті 8001.

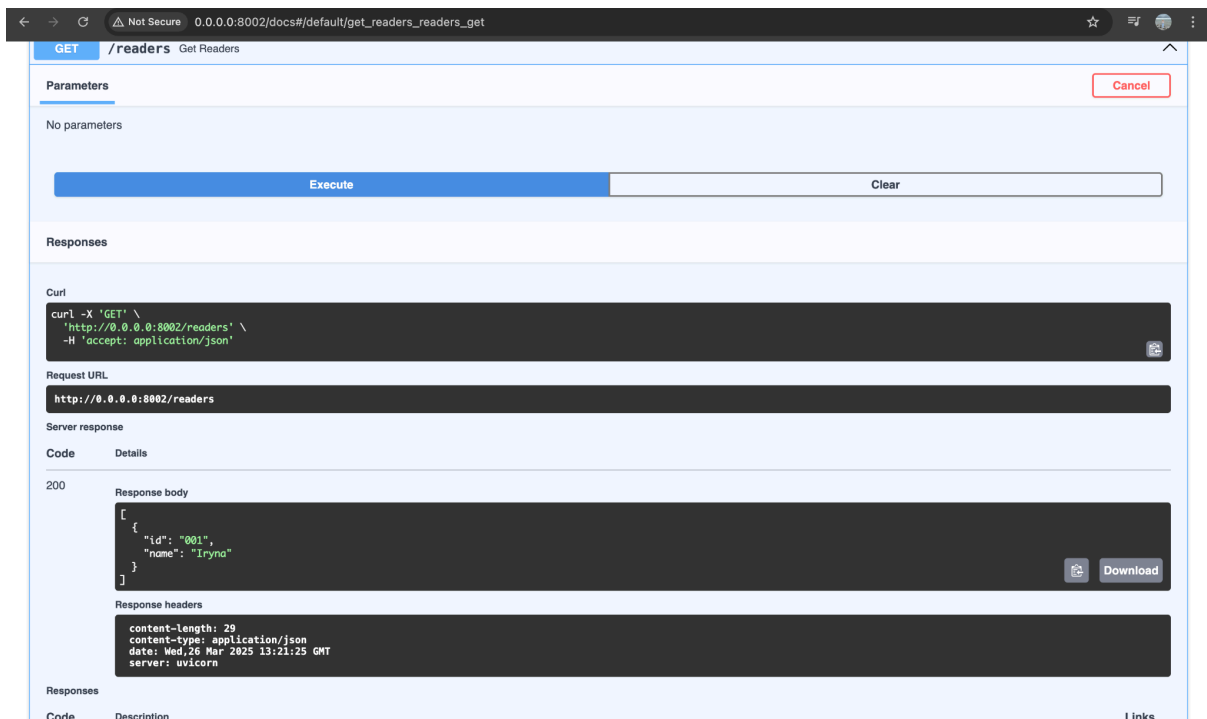


Рис.4. Створені користувачі, API запити на порті 8002.

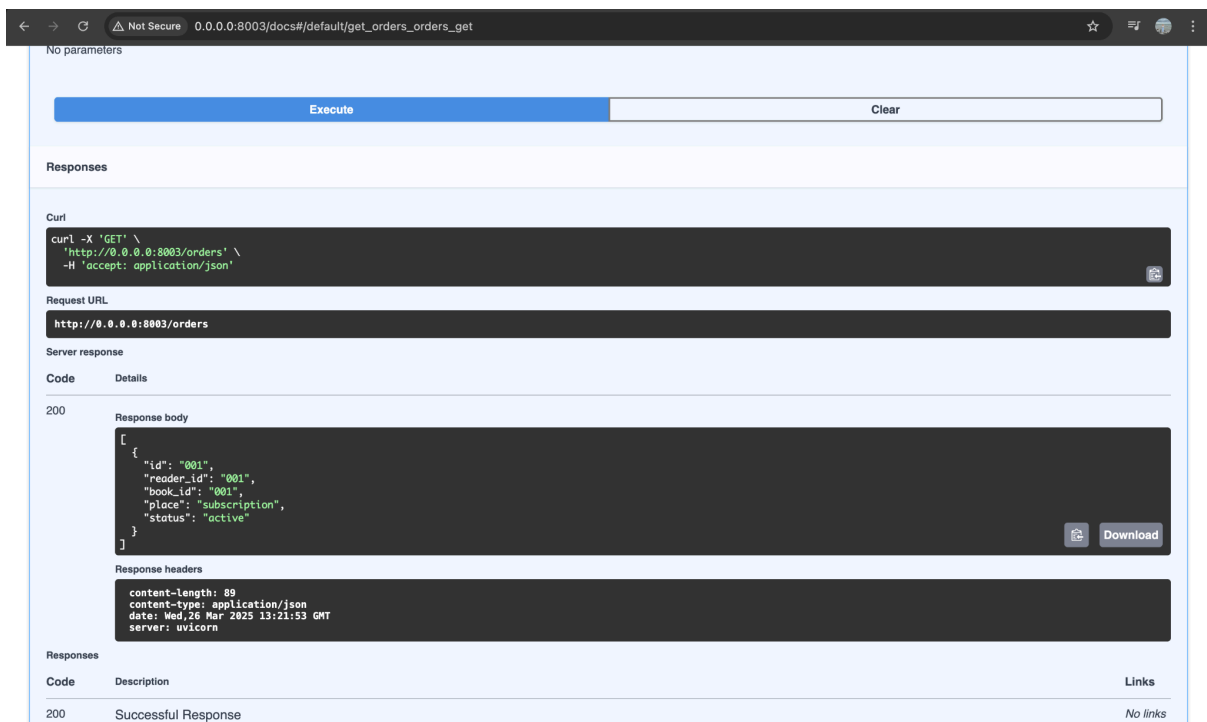


Рис.4. Створені замовлення, API запити на порті 8003.

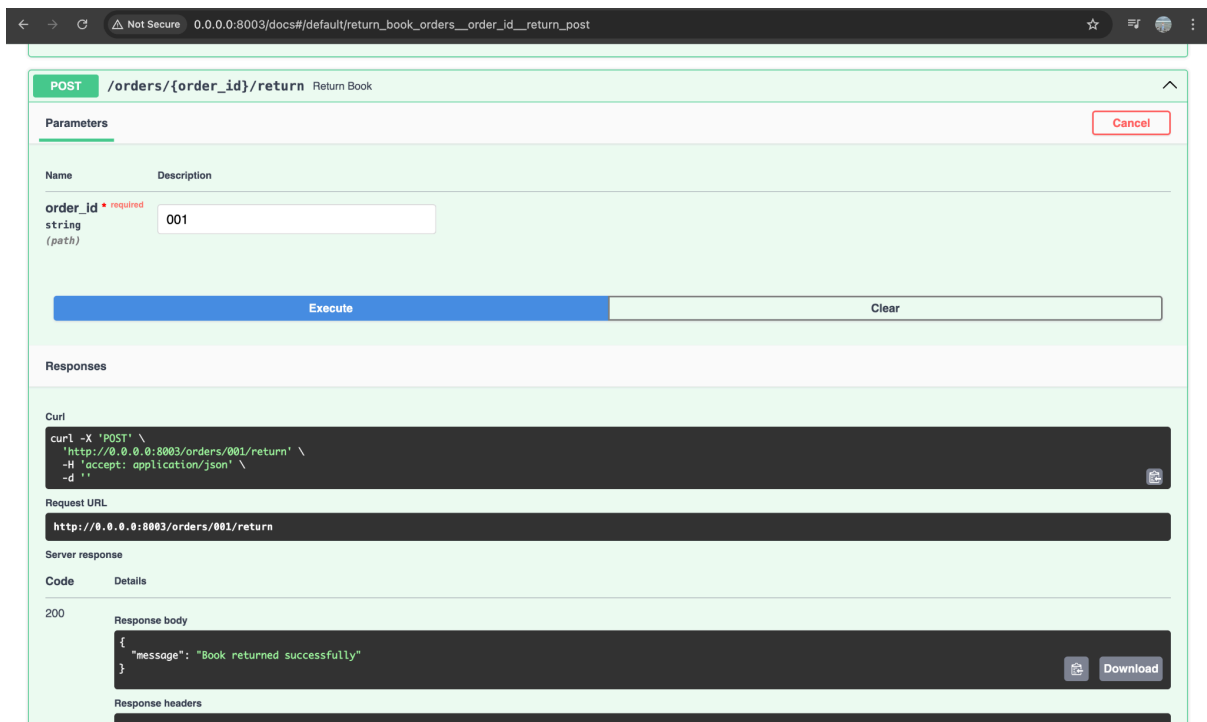


Рис.5. Повернення книги, API запити на порті 8003.

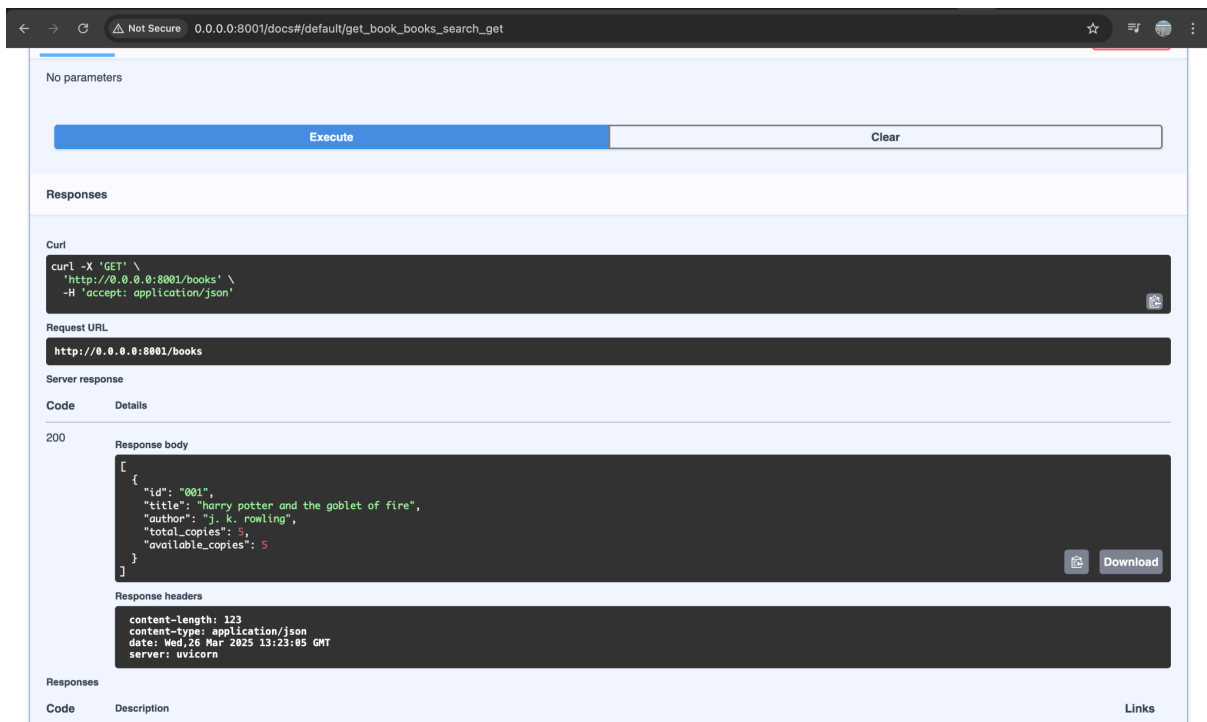


Рис.6. Успішний результат повернення книги, результат API запиту на порті 8001, що стосується книг.

Код програми – посилання на репозиторій github:

<https://github.com/ira-horiacha/po-rs/tree/main>

Висновок: Отже, під час виконання цієї лабораторної роботи, код програми було розбито на три мікросервіси, три різні порти, де кожен порт стосується окремої сутності: книги, читачі, замовлення. Завдяки Docker кожен мікросервіс був контейнеризований, що забезпечило легке розгортання, ізоляцію середовища для кожного сервісу та масштабованість. Docker Compose використовується для запуску всіх контейнерів в єдиній мережі, що дозволяє сервісам взаємодіяти один з одним через HTTP-запити за допомогою внутрішніх адрес Docker.

В результаті було реалізовано інтегровану систему з можливістю обробки замовлень книг, перевірки доступності читачів та книг, а також автоматичного оновлення кількості доступних копій книг при їхньому замовленні та поверненні.