Міністерство освіти і науки України

Національний технічний університет України «КПІ ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Проектування розподілених систем

**ЗВІТ**

до лабораторної роботи №1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Виконали**  **студенти** |  | ІП-31мн Кобельський Вадим |  |  |
|  |  | (№ групи, прізвище, ім’я, по батькові ) |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  | (№ групи, прізвище, ім’я, по батькові ) |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  | (№ групи, прізвище, ім’я, по батькові ) |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| **Прийняв** |  | Обозний Д. М. |  |  |
|  |  | (посада, прізвище, ім’я, по батькові ) |  |  |

Київ 2024

**Завдання**

* Реалізувати синхронну комунікація між 2ма сервісами. Cпоживач Cервісу генерує завдання на обчислення і чекає відповіді від Постачальник Сервісу.
* 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
* Розгорнути Load Balancer перед Споживачем Сервісу і/або Постачальником сервісу
* *Опціонально: реалізувати протокол gRPC*
* *Опціонально: авторизація на рівні Споживача Сервісу*
* *Опціонально: авторизація на рівні Постачальника Сервісу*



Хід роботи

Синхронна комунікація між 2ма сервісами реалізовувалась через HTTP протокол на мові програмування Python за допомогою бібліотеки FastAPI

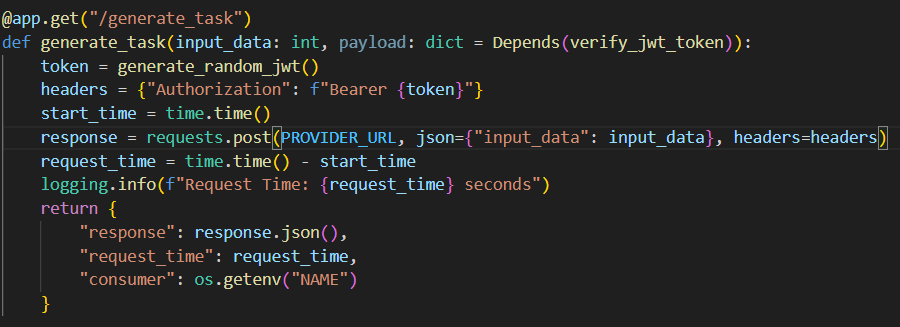


Рисунок 1 – Метод для генерації завдання в Consumer

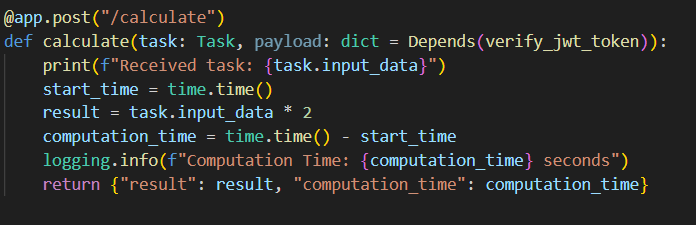


Рисунок 2 – метод для опрацювання завдання в Provider

Також з рисунків 1 та 2 можна побачити лінії з логуванням, Consumer сервіс логує час виконання запиту а Producer логує час обчислення завдання.

Load Balancer було розгорнуто перед Consumer сервісом, і архітектура в такому випадку мала наступний вигляд.

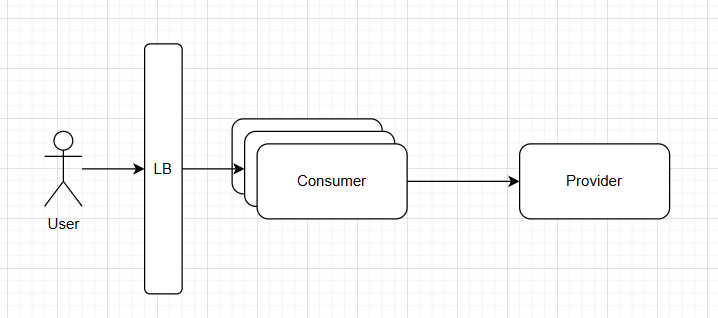


Рисунок 3 – архітектура розробленого застосунку

Також було реалізовано авторизацію на рівні споживача сервісу та постачальника сервісу.

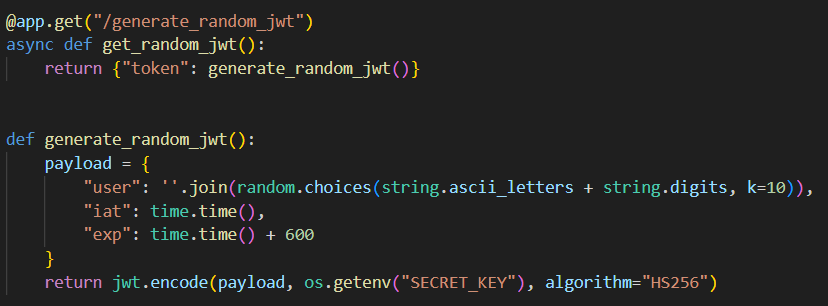


Рисунок 4 – метод для створення jwt токену

Також з рисунків 1 та 2 можна побачити що метод для створення завдання та для опрацювання завдання потребують авторизації, яка саме відбувається за допомогою jwt токену.

Сам застосунок піднімається за допомогою Docker контейнера і Load Balancing відбувається за допомогою nginx.

На рисунках 5-7 можна побачити процес виконання програми:

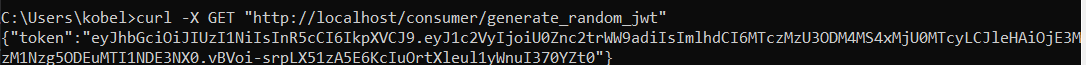


Рисунок 5 – Отримання jwt токену

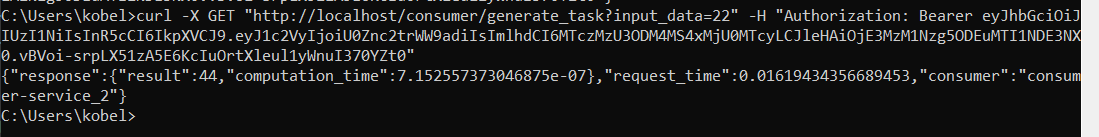
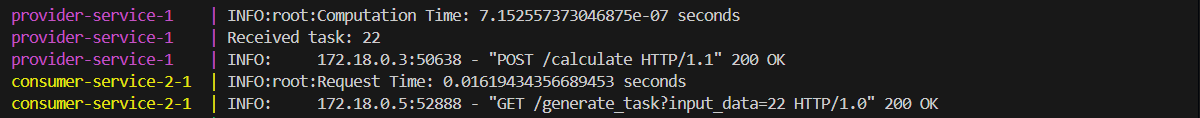


Рисунок 6 – Виконання запиту для генерування завдання



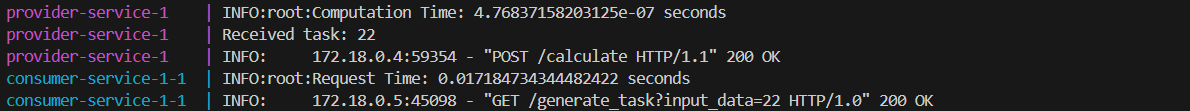


Рисунок 7 – логи з сервісів

Код застосунку доступний за посиланням: