Лабораторна робота №3

Золотов Іван ФБ-31мп

Посилання на GitHub

https://github.com/SideOctopus/Lab_3

Реалізація мікросервісів:

```python
from flask import Flask, request, jsonify
import requests
import random

app = Flask(__name__)

# Список URL-адрес, які вказують на сервіси журналювання
logging_services = [
    "http://localhost:5001",
    "http://localhost:5002",
    "http://localhost:5003"
]

@app.route('/data', methods=['POST', 'GET'])
def handle_data():
    if request.method == 'POST':
        # Обробка POST-запиту
        message = request.data.decode('utf-8')  # Отримання повідомлення з тіла запиту
        target_service = random.choice(logging_services)  # Вибір випадкового сервісу журналювання
        response = requests.post(f'{target_service}/data', json={'message': message})  # Відправка POST-запиту до вибраного сервісу
        return response.text  # Повернення текстової відповіді від сервісу журналювання

    elif request.method == 'GET':
        # Обробка GET-запиту
        target_service = random.choice(logging_services)  # Вибір випадкового сервісу журналювання
        response = requests.get(f'{target_service}/data')  # Відправка GET-запиту до вибраного сервісу
        return jsonify(response.json())  # Повернення JSON-відповіді від сервісу журналювання

if __name__ == '__main__':
    app.run(port=5000, debug=True)
```

```python
from flask import Flask, request, jsonify
import hazelcast
import sys

app = Flask(__name__)

# Підключення до кластера Hazelcast з вказаними учасниками
hz = hazelcast.HazelcastClient(cluster_members=[
    "127.0.0.1:5701",
    "127.0.0.1:5702",
    "127.0.0.1:5703"
])

# Отримання розподіленої мапи з назвою "messages"
distributed_map = hz.get_map("messages").blocking()

@app.route('/data', methods=['POST', 'GET'])
def data():
    if request.method == 'POST':
        # Обробка POST-запиту
        message = request.json['message']  # Витягнення повідомлення з JSON-тіла запиту
        key = str(len(distributed_map.key_set()) + 1)  # Генерація унікального ключа для повідомлення
        distributed_map.put(key, message)  # Збереження повідомлення у розподілену мапу
        return 'Message stored successfully'  # Повернення підтвердження успішного збереження

    elif request.method == 'GET':
        # Обробка GET-запиту
        # Отримання всіх повідомлень з розподіленої мапи
        messages = {key: distributed_map.get(key) for key in distributed_map.key_set()}
        return jsonify(messages)  # Повернення повідомлень у вигляді JSON-відповіді

if __name__ == '__main__':
    port = int(sys.argv[1])  # Отримання номера порту з аргументу командного рядка
    app.run(port=port, debug=True)  # Запуск додатка Flask на вказаному порту у режимі налагодження
```

```python
from flask import Flask, jsonify, request

app = Flask(__name__)

messages = {}

@app.route('/data', methods=['POST', 'GET'])
def data():
    if request.method == 'POST':
        # Обробка POST-запиту
        message = request.json['message']  # Отримання повідомлення з JSON-тіла запиту
        key = len(messages) + 1  # Генерація унікального ключа для повідомлення
        messages[key] = message  # Збереження повідомлення у словник
        return 'Message stored successfully'  # Повернення підтвердження успішного збереження

    elif request.method == 'GET':
        # Обробка GET-запиту
        return jsonify(messages)  # Повернення усіх збережених повідомлень у вигляді JSON

if __name__ == '__main__':
    app.run(port=5005, debug=True)
```

Клієнт

```python
import requests

def main():
    base_url = 'http://127.0.0.1:5000/data'

    while True:
        print("1) Send POST-request")
        print("2) Send GET-request")
        print("3) Exit")
        choice = int(input("Your choice: "))

        if choice == 1:
            message = input("Enter your message: ")
            response = requests.post(base_url, data=message)
            print('POST response:', response.text)
        elif choice == 2:
            response = requests.get(base_url)
            print('GET response:', response.json())
        elif choice == 3:
            break
        else:
            print("Invalid choice. Please try again.")

if __name__ == "__main__":
    main()
```

Запускаємо loggingService *3

```
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment.
 Use a production WSGI server instead.
 * Running on http://127.0.0.1:5001
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
```

```
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment.
 Use a production WSGI server instead.
 * Running on http://127.0.0.1:5003
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
```

```
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment.
 Use a production WSGI server instead.
 * Running on http://127.0.0.1:5002
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
```

facadeService

```
* Debug mode: on
ARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
ress CTRL+C to quit
* Restarting with stat
* Debugger is active!
```

messagesService

```
* Debug mode: on
ARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Running on http://127.0.0.1:5005
ress CTRL+C to quit
* Restarting with stat
* Debugger is active!
```

Далі я ввів 10 повідомлень і перевірив, що вони збереглись.

Переглянув ноди, після чого я відключив loggingService – це призвело до перемішання даних які я вводив до цього. Самі дані не були ушкодженні).