



МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ  
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. ІГОРЯ СІКОРСЬКОГО»  
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ  
КАФЕДРА ІНФОРМАЦІЙНОЇ БЕЗПЕКИ

## **Проектування розподілених систем**

### **Лабораторна робота №5**

**Мікросервіси з використанням Service Discovery**  
**та Config Server на базі Consul**

Перевірив:  
Родіонов А. М.

Виконав:  
студент I курсу  
групи ФБ-41мп  
Сахній Н. Р.

Київ 2025

**Мета роботи:** Це завдання базується на основі попередніх і є їх розвитком. Таким чином, до системи необхідно додати **Consul**, який буде виконувати роль *Service Register*, *Service Discovery* та *Config Server*. Це дасть змогу явно не вказувати адреси мікросервісів та конфігурації для Hazelcast та Message Queue.

**Архітектура** складається з 4-ох мікросервісів, реалізованих на мові **Python**:

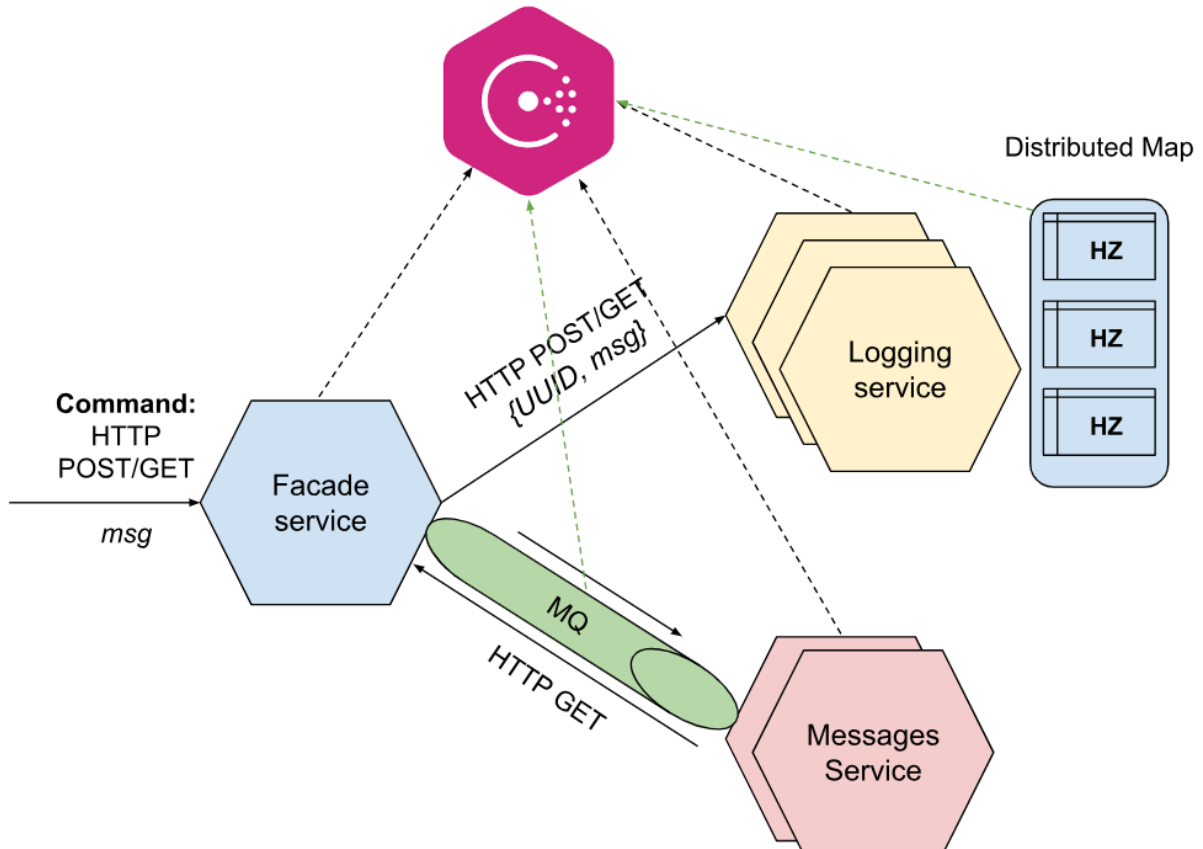
- **facade-service** – обробляє POST/GET-запити надіслані клієнтом.
- **logging-service** – зберігає у своїй пам'яті всі повідомлення із їхніми унікальними ідентифікаторами та надає до них доступ для їх перегляду.
- **messages-service** – отримує та зберігає повідомлення із черги.
- **consulver** – динамічно надає інформацію про конфігурації системи.

Посилання на GitHub з проектом, що містить вихідні коди 4-ох мікросервісів:

[https://github.com/sazan24/KPI/tree/main/Master's%20degree/Distributed%20Systems%20Design/Task 5-Microservices with Consul/](https://github.com/sazan24/KPI/tree/main/Master's%20degree/Distributed%20Systems%20Design/Task%205-Microservices%20with%20Consul/)

---

- **Опис HTTP POST/GET Request Flow**



## Завдання до виконання:

### о. Налаштувати кластери Consul та Hazelcast через docker-compose

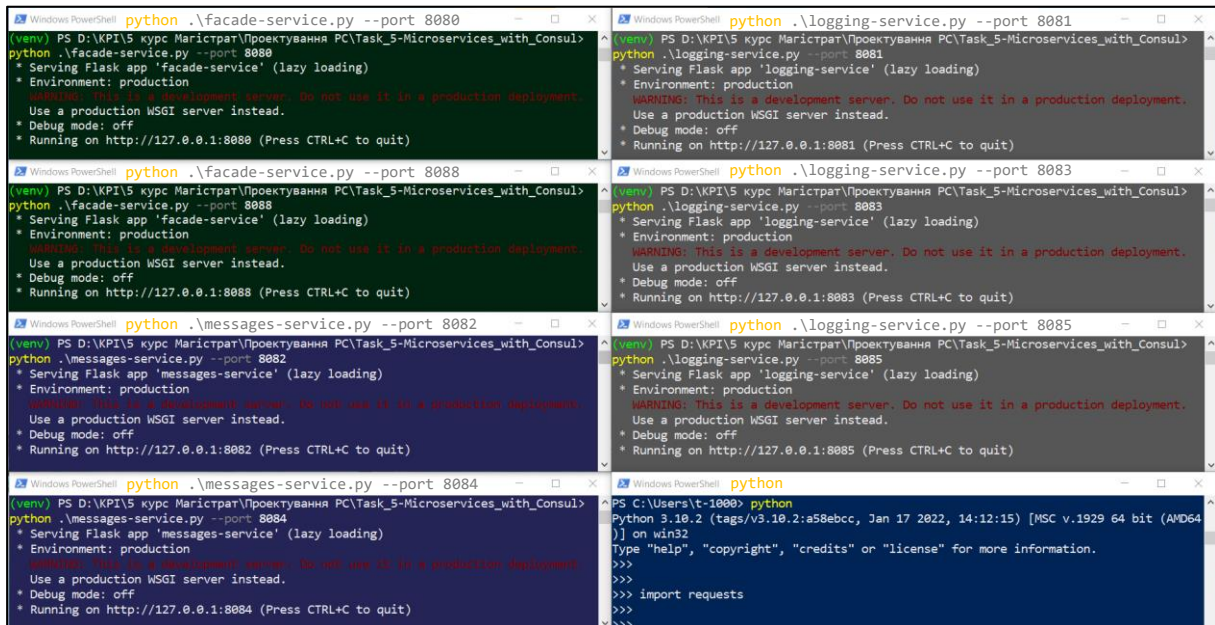
```
1  version: "3.2"
2
3  services:
4    consul:
5      container_name: 'consul-server'
6      image: 'hashicorp/consul'
7      ports:
8        - "8500:8500"
9        - "8600:8600/udp"
10     command: agent -server -ui -node=consul-server
11     -bootstrap-expect=1 -client=0.0.0.0
12
13
14     hazelcast-node1:
15       container_name: 'hazelcast-node1'
16       image: 'hazelcast/hazelcast:5.4.0'
17       environment:
18         - HZ_CLUSTERNAME=distributed-consul-cluster
19         - HZ_NETWORK_PUBLICADDRESS=192.168.88.164:5701
20       ports:
21         - '5701:5701'
22       networks:
23         - hazelcast-network
24
25     hazelcast-node2:
26       container_name: 'hazelcast-node2'
27       image: 'hazelcast/hazelcast:5.4.0'
28       environment:
29         - HZ_CLUSTERNAME=distributed-consul-cluster
30         - HZ_NETWORK_PUBLICADDRESS=192.168.88.164:5702
31       ports:
32         - '5702:5701'
33       networks:
34         - hazelcast-network
35
36     hazelcast-node3:
37       container_name: 'hazelcast-node3'
38       image: 'hazelcast/hazelcast:5.4.0'
39       environment:
40         - HZ_CLUSTERNAME=distributed-consul-cluster
41         - HZ_NETWORK_PUBLICADDRESS=192.168.88.164:5703
42       ports:
43         - '5703:5701'
44       networks:
45         - hazelcast-network
46
```

```
47
48     hazelcast-management:
49       container_name: 'hazelcast-management-center'
50       image: 'hazelcast/management-center:5.4.0'
51       depends_on:
52         - hazelcast-node1
53         - hazelcast-node2
54         - hazelcast-node3
55       ports:
56         - '8008:8080'
57       networks:
58         - hazelcast-network
59
60
61     networks:
62       hazelcast-network:
63         driver: bridge
64
```

```
i-10000DESKTOP-OR10PBB WINDOWS /d/KPI/5 курс Марістрат/Проектування PC/Task_5-Microservices_with_Consul
$ docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS
f034a728b41d   hazelcast/management-center:5.4.0   "bash ./bin/mc-start."  About a minute ago   Up About a minute   8081/tcp, 8443/tcp, 0.0.0.0:8008->8080/tcp
6b7d23d2f7d7   hazelcast/hazelcast:5.4.0           "hz start"              About a minute ago   Up About a minute   0.0.0.0:5702->5701/tcp
72ee58c3d00c   hazelcast/hazelcast:5.4.0           "hz start"              About a minute ago   Up About a minute   0.0.0.0:5703->5701/tcp
76a34579cc10   hazelcast/hazelcast:5.4.0           "hz start"              About a minute ago   Up About a minute   0.0.0.0:5701->5701/tcp
8e3f16ebb011   hashicorp/consul                    "docker-entrypoint.s..." About a minute ago   Up About a minute   8300-8302/tcp, 8600/tcp, 8301-8302/udp, 0.0.0.0:8500->8500/tcp, 0.0.0.0:8600->8600/udp
```

1. Усі мікросервіси мають самостійно динамічно реєструватись при старті у **Consul**, і кожного з сервісів може бути запущено декілька екземплярів.

- **facade-service** (x2: 8080 та 8088)
- **logging-service** (x3: 8081, 8083 та 8085)
- **messages-service** (x2: 8082, 8083)



```
python .\facade-service.py --port 8080
(venv) PS D:\KPI\5 курс Маріцпар\Проектування PC\Task_5-Microservices_with_Consul>
python .\facade-service.py --port 8080
* Serving Flask app 'facade-service' (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:8080 (Press CTRL+C to quit)

python .\facade-service.py --port 8088
(venv) PS D:\KPI\5 курс Маріцпар\Проектування PC\Task_5-Microservices_with_Consul>
python .\facade-service.py --port 8088
* Serving Flask app 'facade-service' (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:8088 (Press CTRL+C to quit)

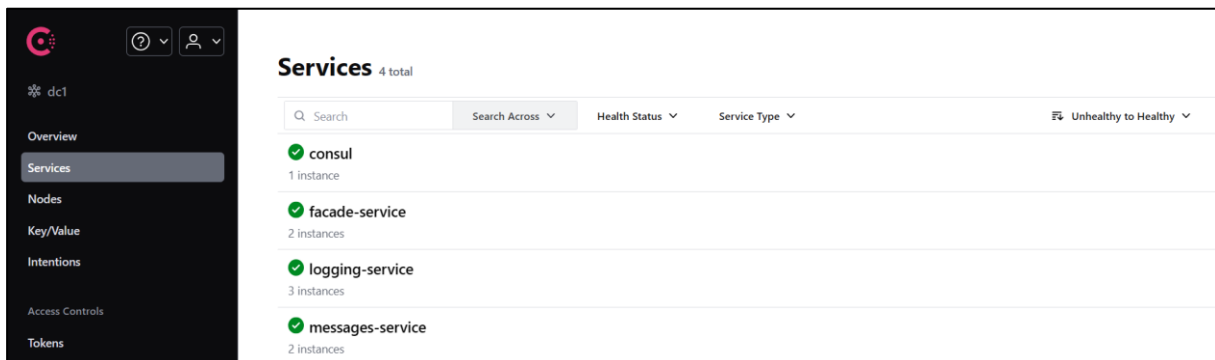
python .\messages-service.py --port 8082
(venv) PS D:\KPI\5 курс Маріцпар\Проектування PC\Task_5-Microservices_with_Consul>
python .\messages-service.py --port 8082
* Serving Flask app 'messages-service' (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:8082 (Press CTRL+C to quit)

python .\messages-service.py --port 8084
(venv) PS D:\KPI\5 курс Маріцпар\Проектування PC\Task_5-Microservices_with_Consul>
python .\messages-service.py --port 8084
* Serving Flask app 'messages-service' (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:8084 (Press CTRL+C to quit)

python .\logging-service.py --port 8081
(venv) PS D:\KPI\5 курс Маріцпар\Проектування PC\Task_5-Microservices_with_Consul>
python .\logging-service.py --port 8081
* Serving Flask app 'logging-service' (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:8081 (Press CTRL+C to quit)

python .\logging-service.py --port 8083
(venv) PS D:\KPI\5 курс Маріцпар\Проектування PC\Task_5-Microservices_with_Consul>
python .\logging-service.py --port 8083
* Serving Flask app 'logging-service' (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:8083 (Press CTRL+C to quit)

python .\logging-service.py --port 8085
(venv) PS D:\KPI\5 курс Маріцпар\Проектування PC\Task_5-Microservices_with_Consul>
python .\logging-service.py --port 8085
* Serving Flask app 'logging-service' (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:8085 (Press CTRL+C to quit)
```



2. При звертанні **facade-service** до **logging-service** та **messages-service**, IP-адреси (і порти) мають зчитуватись **facade-service** з **Consul**.

6748904c-f6d6-4f95-b190-82df0c6cccf5

No service checks

All node checks passing

consul-server

localhost:8080

b724bfd1-d491-49b7-a855-2f24c7ac0300

No service checks

All node checks passing

consul-server

localhost:8080

facade-service

04f6bccd-71c5-4e50-9b25-8f4918edbad5	<input type="checkbox"/> No service checks <input checked="" type="checkbox"/> All node checks passing consul-server localhost:8083	
b4ba0be8-fc73-4f60-a934-6cd54bbffbc2	<input type="checkbox"/> No service checks <input checked="" type="checkbox"/> All node checks passing consul-server localhost:8081	logging-service
e6a5f54c-64ed-416e-9c9a-c0a19559e71f	<input type="checkbox"/> No service checks <input checked="" type="checkbox"/> All node checks passing consul-server localhost:8085	

0efb8419-f3a4-4a2f-800e-362d1b4d275c	<input type="checkbox"/> No service checks <input checked="" type="checkbox"/> All node checks passing consul-server localhost:8082	
340c8162-ce6c-467d-804a-66d4d4efe2f5	<input type="checkbox"/> No service checks <input checked="" type="checkbox"/> All node checks passing consul-server localhost:8084	messages-service

3. Налаштування для клієнтів Hazelcast мають зберігатись як key/value у **Consul** і зчитуватись **logging-service**.

4. Налаштування для “Message Queue” (адреса, назва черги, ) мають зберігатись як key/value у **Consul** і зчитуватись **facade-service** та **messages-service**.

Key / Values

### lab\_settings

Value

```

{
  "cluster_name": "distributed-consul-cluster",
  "queue_name": "message-distributed-queue",
  "map_name": "message-distributed-map",
  "cluster_nodes": {
    "hazelcast-node1": 1,
    "hazelcast-node2": 1,
    "hazelcast-node3": 1,
    "distque-management-center": 1
  },
  "api-endpoints": {
    "fcd": "/facade",
    "log": "/logging",
    "msg": "/messages"
  }
}

```

Якщо ноди Hazelcast-кластера будуть вимкнені, то зберігатиметься “0”, а не “1”:

```

8  "cluster_nodes": {
9    "hazelcast-node1": 0,
10   "hazelcast-node2": 0,
11   "hazelcast-node3": 0,
12   "distque-management-center": 0
13 },

```

Через фасад-сервіси почергово відправимо 10 **POST**-запитів із повідомленнями:

```
python .\facade-service.py --port 8080
- Message Was Queued: 'msg1'
logging-service 8083
127.0.0.1 - - [10/Apr/2025 12:56:52] "POST /facade HTTP/1.1" 201 -
- Message Was Queued: 'msg3'
logging-service 8083
127.0.0.1 - - [10/Apr/2025 12:56:52] "POST /facade HTTP/1.1" 201 -
- Message Was Queued: 'msg5'
logging-service 8083
127.0.0.1 - - [10/Apr/2025 12:56:52] "POST /facade HTTP/1.1" 201 -
- Message Was Queued: 'msg7'
logging-service 8085
127.0.0.1 - - [10/Apr/2025 12:56:52] "POST /facade HTTP/1.1" 201 -
- Message Was Queued: 'msg9'
logging-service 8083
127.0.0.1 - - [10/Apr/2025 12:56:52] "POST /facade HTTP/1.1" 201 -

python .\facade-service.py --port 8088
- Message Was Queued: 'msg2'
logging-service 8081
127.0.0.1 - - [10/Apr/2025 12:56:52] "POST /facade HTTP/1.1" 201 -
- Message Was Queued: 'msg4'
logging-service 8083
127.0.0.1 - - [10/Apr/2025 12:56:52] "POST /facade HTTP/1.1" 201 -
- Message Was Queued: 'msg6'
logging-service 8081
127.0.0.1 - - [10/Apr/2025 12:56:52] "POST /facade HTTP/1.1" 201 -
- Message Was Queued: 'msg8'
logging-service 8081
127.0.0.1 - - [10/Apr/2025 12:56:52] "POST /facade HTTP/1.1" 201 -
- Message Was Queued: 'msg10'
logging-service 8085
127.0.0.1 - - [10/Apr/2025 12:56:52] "POST /facade HTTP/1.1" 201 -

python .\messages-service.py --port 8082
- Message Was Received: 'msg1'
- Message Was Received: 'msg2'
- Message Was Received: 'msg3'
- Message Was Received: 'msg4'
- Message Was Received: 'msg5'
- Message Was Received: 'msg6'
- Message Was Received: 'msg7'
- Message Was Received: 'msg8'
- Message Was Received: 'msg9'
- Message Was Received: 'msg10'

python .\messages-service.py --port 8084
- Message Was Received: 'msg1'
- Message Was Received: 'msg2'
- Message Was Received: 'msg3'
- Message Was Received: 'msg4'
- Message Was Received: 'msg5'
- Message Was Received: 'msg6'
- Message Was Received: 'msg7'
- Message Was Received: 'msg8'
- Message Was Received: 'msg9'
- Message Was Received: 'msg10'
```

Через фасад-сервіси почергово відправимо пару **GET**-запитів і отримаємо “msg”:

```
python .\facade-service.py --port 8080
logging-service 8081
127.0.0.1 - - [10/Apr/2025 12:57:19] "GET /facade HTTP/1.1" 200 -
127.0.0.1 - - [10/Apr/2025 12:57:38] "GET /facade HTTP/1.1" 200 -

python .\facade-service.py --port 8088
127.0.0.1 - - [10/Apr/2025 12:57:19] "GET /messages HTTP/1.1" 200 -

python .\facade-service.py --port 8084
127.0.0.1 - - [10/Apr/2025 12:57:38] "GET /messages HTTP/1.1" 200 -

python .\logging-service.py --port 8081
127.0.0.1 - - [10/Apr/2025 12:57:19] "GET /logging HTTP/1.1" 200 -
127.0.0.1 - - [10/Apr/2025 12:57:38] "GET /logging HTTP/1.1" 200 -

python .\logging-service.py --port 8083
python .\logging-service.py --port 8085

python
>>> print(res.json())
['logging-service: ["msg3","msg7","msg2","msg6","msg10","msg8","msg5","msg1","msg4","msg9"]\n', 'messages-service: ["msg2","msg4","msg6","msg8","msg10"]\n']
>>>
>>> res = requests.get("http://localhost:8080/facade")
>>> print(res.json())
['logging-service: ["msg3","msg7","msg2","msg6","msg10","msg8","msg5","msg1","msg4","msg9"]\n', 'messages-service: ["msg1","msg3","msg5","msg7","msg9"]\n']
>>> exit
Use exit() or Ctrl-Z plus Return to exit
```

Map Statistics (In-Memory Format: BINARY)									
RESET TIME 1 minute ago → now									
Default View									
Member ^	^ Entries	^ Gets	^ Puts	^ Rem...	^ Sets	^ Entry Memory	^ Events	^ Hits	
192.168.88.164:5701	3	6	3	0	0	492.00 B	0	6	
192.168.88.164:5702	3	6	3	0	0	493.00 B	0	6	
192.168.88.164:5703	4	8	4	0	0	656.00 B	0	8	
TOTAL	10	20	3	0	0	1.60 kB	0	20	



5. Продемонструвати, що у випадку відключення екземпляру певного мікросервісу, це буде відображатись у **Consul** (відключений екземпляр сервісу перестане існувати), а виклики будуть перенаправлятись до інших працюючих екземплярів.

```
Windows PowerShell python .\facade-service.py --port 8080
logging-service 8081
messages-service 8082
127.0.0.1 - - [10/Apr/2025 12:57:19] "GET /facade HTTP/1.1" 200 -
logging-service 8081
messages-service 8084
127.0.0.1 - - [10/Apr/2025 12:57:38] "GET /facade HTTP/1.1" 200 -

Windows PowerShell python .\facade-service.py --port 8088
(venv) PS D:\KPI\5 курс Марістрат\Проектування PC\Task_5-Microservices_with_Consul>
(venv) PS D:\KPI\5 курс Марістрат\Проектування PC\Task_5-Microservices_with_Consul>

Windows PowerShell python .\messages-service.py --port 8082
127.0.0.1 - - [10/Apr/2025 12:57:19] "GET /messages HTTP/1.1" 200 -
[ERROR] Queue read failed: 'TargetDisconnectedError'

Windows PowerShell python .\messages-service.py --port 8084
(venv) PS D:\KPI\5 курс Марістрат\Проектування PC\Task_5-Microservices_with_Consul>
(venv) PS D:\KPI\5 курс Марістрат\Проектування PC\Task_5-Microservices_with_Consul>

Windows PowerShell python .\logging-service.py --port 8081
127.0.0.1 - - [10/Apr/2025 12:57:19] "GET /logging HTTP/1.1" 200 -
127.0.0.1 - - [10/Apr/2025 12:57:38] "GET /logging HTTP/1.1" 200 -

Windows PowerShell python .\logging-service.py --port 8083
+++++++[STOP-NODE]+++++++
hazelcast-node3
(venv) PS D:\KPI\5 курс Марістрат\Проектування PC\Task_5-Microservices_with_Consul>

Windows PowerShell python .\logging-service.py --port 8085
+++++++[STOP-NODE]+++++++
hazelcast-node1
(venv) PS D:\KPI\5 курс Марістрат\Проектування PC\Task_5-Microservices_with_Consul>

Windows PowerShell python
>>>
>>>
```

dc1

Overview

Services

Nodes

Key/Value

Intentions

Access Controls

Tokens

Services 4 total

Q Search

Search Across

Health Status

Service Type

Unhealthy to Healthy

consul

1 instance

facade-service

1 instance

logging-service

1 instance

messages-service

1 instance

```
Windows PowerShell python .\facade-service.py --port 8080
logging-service 8081
messages-service 8082
127.0.0.1 - - [10/Apr/2025 14:19:27] "GET /facade HTTP/1.1" 200 -

Windows PowerShell python .\facade-service.py --port 8088
Windows PowerShell python .\logging-service.py --port 8081
127.0.0.1 - - [10/Apr/2025 14:19:27] "GET /logging HTTP/1.1" 200 -

Windows PowerShell python .\messages-service.py --port 8082
127.0.0.1 - - [10/Apr/2025 14:19:27] "GET /messages HTTP/1.1" 200 -

Windows PowerShell python .\messages-service.py --port 8084
Windows PowerShell python
>>> res = requests.get("http://localhost:8080/facade")
>>> print(res.json())
{'logging-service: ['msg6','msg5','msg4','msg3','msg1','msg2','msg7','msg8','msg9','msg10'],'messages-service: ['msg1','msg3','msg5','msg7','msg9']\n'}
```

Map Statistics (In-Memory Format: BINARY)

RESET TIME 1 minute ago → now

Default View

Member	Entries	Gets	Puts	Rem...	Sets	Entry Memory	Events	Hits
192.168.88.164:5702	10	16	3	0	0	1.60 kB	0	16
TOTAL	10	16	3	0	0	1.60 kB	0	16