

Проектування розподілених систем
Task 1 - Базова архітектура мікросервісів
Мартиненко Денис ФБ-42мп

facade-service

У випадку POST-запиту facade сервіс отримує текст повідомлення, генерує id та надсилає POST-запит з повідомленням logging сервісу.

У випадку GET-запиту facade сервіс виконує два GET-запити до logging та messages сервісів.

Також цей код спробує зробити POST-запит до logging-service 3 рази з інтервалом 2 секунди, якщо виникне помилка.

```
import requests, uuid
from flask import Flask, request, jsonify
from tenacity import retry, stop_after_attempt, wait_fixed

app = Flask(__name__)
logging_service = 'http://localhost:8881/logging'
messages_service = 'http://localhost:8882/messages'

@retry(stop=stop_after_attempt(3), wait=wait_fixed(2))
def send_with_retry(url, data):
    response = requests.post(url, data=data)
    if response.status_code != 201:
        raise Exception('Logging service did not accept message.')
    return response

@app.route('/', methods=['POST', 'GET'])
def home():
    if request.method == 'POST':
        txt = request.form.get('txt')
        if not txt:
            return jsonify('No message'), 400
        id = str(uuid.uuid4())
        msg = {'id': id, 'txt': txt}
        try:
            send_with_retry(logging_service, msg)
            return jsonify(msg), 200
        except Exception as e:
            return jsonify({'error': str(e)}), 500

    elif request.method == 'GET':
        try:
            logging_response = requests.get(logging_service)
            messages_response = requests.get(messages_service)
```

```

        combined_response = [
            f'logging-service: {logging_response.text}',
            f'message-service: {messages_response.text}'
        ]
        return jsonify(combined_response), 200
    except Exception as e:
        return jsonify({'error': str(e)}), 500

if __name__ == '__main__':
    app.run(port=8880)

```

logging-service

У випадку POST-запиту logging сервіс отримує id та текст повідомлення і зберігає його.

У випадку GET-запиту logging сервіс повертає текст всіх збережених повідомлень.

Також цей код запобігає повторному логуванню повідомлень із тим самим UUID.

```

from flask import Flask, request, jsonify

app = Flask(__name__)
msg = {}

@app.route('/logging', methods=['POST', 'GET'])
def home():
    if request.method == 'POST':
        id, txt = request.form.get('id'), request.form.get('txt')
        if not (id and txt):
            return jsonify('No id or text'), 400

        if id in msg:
            print(f'Duplicate message detected: {id}')
            return jsonify('Duplicate detected'), 200

        msg[id] = txt
        print('Message:', txt)
        return jsonify('Message logged'), 201

    elif request.method == 'GET':
        return jsonify(list(msg.values()))

if __name__ == '__main__':
    app.run(port=8881)

```

messages-service

У випадку GET-запиту messages сервіс просто повертає текст 'Message-service not implemented yet'.

```
from flask import Flask, request, jsonify

app = Flask(__name__)

@app.route('/messages', methods=['GET'])

def home():
    if request.method == 'GET':
        return 'Message-service not implemented yet'
    else:
        return jsonify('Method not allowed'), 405

if __name__ == '__main__':
    app.run(port=8882)
```

logging-service через gRPC:

```
import grpc
from concurrent import futures
import logging_pb2, logging_pb2_grpc

msg = {}

class LoggingService(logging_pb2_grpc.LoggingServiceServicer):
    def LogMessage(self, request, context):
        if request.id in msg:
            return logging_pb2.LogReply(status='Duplicate')
        msg[request.id] = request.txt
        print('Message:', request.txt)
        return logging_pb2.LogReply(status='Logged')

    def GetMessages(self, request, context):
        return logging_pb2.MessagesReply(messages=list(msg.values()))

def serve():
    server = grpc.server(futures.ThreadPoolExecutor(max_workers=10))
    logging_pb2_grpc.add_LoggingServiceServicer_to_server(LoggingService(),
server)
    server.add_insecure_port('[::]:50051')
    server.start()
    server.wait_for_termination()

if __name__ == '__main__':
    serve()
```

facade-service із gRPC-клієнтом:

```
import grpc, uuid
from flask import Flask, request, jsonify
import logging_pb2, logging_pb2_grpc

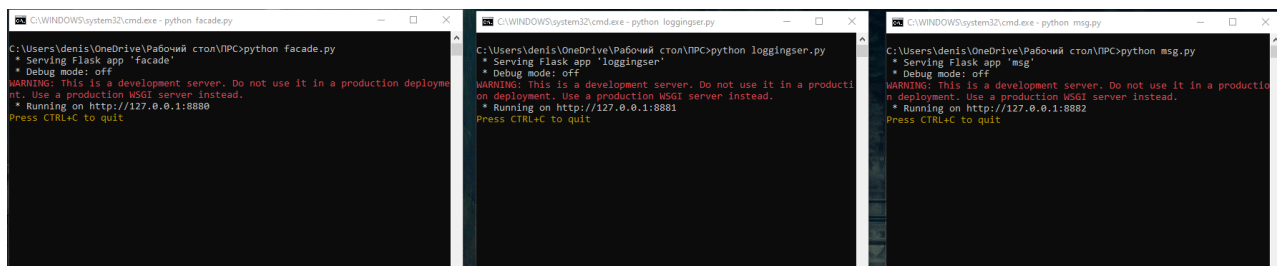
app = Flask(__name__)
channel = grpc.insecure_channel('localhost:50051')
client = logging_pb2_grpc.LoggingServiceStub(channel)

@app.route('/', methods=['POST', 'GET'])
def home():
    if request.method == 'POST':
        txt = request.form.get('txt')
        if not txt:
            return jsonify('No message'), 400
        id = str(uuid.uuid4())
        response = client.LogMessage(logging_pb2.LogRequest(id=id, txt=txt))
        return jsonify({'id': id, 'status': response.status}), 200

    elif request.method == 'GET':
        response = client.GetMessages(logging_pb2.Empty())
        return jsonify(list(response.messages)), 200

if __name__ == '__main__':
    app.run(port=8880)
```

Запуск:



POST request:

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.19045.5608]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\denis>curl -X POST -d "txt=Hello" http://localhost:8880
{"id":"ef3d9636-ea71-4b11-9031-6cac64af20fc","txt":"Hello"}

C:\Users\denis>curl -X POST -d "txt=Its me" http://localhost:8880
{"id":"54c460a1-dddc-4ca7-a92d-3c748c2d625c","txt":"Its me"}

C:\Users\denis>curl -X POST -d "txt=Den4ik" http://localhost:8880
{"id":"0ee72fc2-00af-4259-b364-a9745e617c75","txt":"Den4ik"}

C:\Users\denis>
```

```
C:\WINDOWS\system32\cmd.exe - python facade.py
C:\Users\denis\OneDrive\Рабочий стол\ПРС>python facade.py
* Serving Flask app 'facade'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:8880
Press CTRL+C to quit
127.0.0.1 - - [28/Mar/2025 14:40:54] "POST / HTTP/1.1" 200 -
127.0.0.1 - - [28/Mar/2025 14:41:22] "POST / HTTP/1.1" 200 -
127.0.0.1 - - [28/Mar/2025 14:41:34] "POST / HTTP/1.1" 200 -

C:\WINDOWS\system32\cmd.exe - python loggingser.py
C:\Users\denis\OneDrive\Рабочий стол\ПРС>python loggingser.py
* Serving Flask app 'loggingser'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:8881
Press CTRL+C to quit
Message: Hello
127.0.0.1 - - [28/Mar/2025 14:40:54] "POST /logging HTTP/1.1" 201 -
Message: Its me
127.0.0.1 - - [28/Mar/2025 14:41:22] "POST /logging HTTP/1.1" 201 -
Message: Den4ik
127.0.0.1 - - [28/Mar/2025 14:41:34] "POST /logging HTTP/1.1" 201 -
```

```
C:\WINDOWS\system32\cmd.exe - python msg.py
C:\Users\denis\OneDrive\Рабочий стол\ПРС>python msg.py
* Serving Flask app 'msg'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:8882
Press CTRL+C to quit
```

GET request:

```
C:\WINDOWS\system32\cmd.exe

C:\Users\denis>curl http://localhost:8880
["logging-service: [\\"Hello\\",\\"Its me\\",\\"Den4ik\\"]\n","message-service: Message-service not implemented yet"]

C:\Users\denis>
```

```
C:\WINDOWS\system32\cmd.exe - python facade.py

C:\Users\denis\OneDrive\Рабочий стол\ПРС>python facade.py
* Serving Flask app 'facade'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:8880
Press CTRL+C to quit
127.0.0.1 - - [28/Mar/2025 14:56:21] "GET / HTTP/1.1" 200 -
```

```
C:\WINDOWS\system32\cmd.exe - python loggingser.py

C:\Users\denis\OneDrive\Рабочий стол\ПРС>python loggingser.py
* Serving Flask app 'loggingser'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:8881
Press CTRL+C to quit
Message: Hello
127.0.0.1 - - [28/Mar/2025 14:40:54] "POST /logging HTTP/1.1" 201 -
Message: Its me
127.0.0.1 - - [28/Mar/2025 14:41:22] "POST /logging HTTP/1.1" 201 -
Message: Den4ik
127.0.0.1 - - [28/Mar/2025 14:41:34] "POST /logging HTTP/1.1" 201 -
127.0.0.1 - - [28/Mar/2025 14:56:21] "GET /logging HTTP/1.1" 200 -
```

```
C:\WINDOWS\system32\cmd.exe - python msg.py

C:\Users\denis\OneDrive\Рабочий стол\ПРС>python msg.py
* Serving Flask app 'msg'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:8882
Press CTRL+C to quit
127.0.0.1 - - [28/Mar/2025 14:56:21] "GET /messages HTTP/1.1" 200 -
```

Перевірка retry:

```
C:\WINDOWS\system32\cmd.exe - python facade.py

C:\Users\denis\OneDrive\Рабочий стол\ПРС>python facade.py
* Serving Flask app 'facade'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:8880
Press CTRL+C to quit
127.0.0.1 - - [28/Mar/2025 14:56:21] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [28/Mar/2025 15:00:24] "POST / HTTP/1.1" 500 -
```

```
C:\WINDOWS\system32\cmd.exe - python loggingser.py

C:\Users\denis\OneDrive\Рабочий стол\ПРС>python loggingser.py
* Serving Flask app 'loggingser'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:8881
Press CTRL+C to quit
Message: Hello
127.0.0.1 - - [28/Mar/2025 14:40:54] "POST /logging HTTP/1.1" 201 -
Message: Its me
127.0.0.1 - - [28/Mar/2025 14:41:22] "POST /logging HTTP/1.1" 201 -
Message: Den4ik
127.0.0.1 - - [28/Mar/2025 14:41:34] "POST /logging HTTP/1.1" 201 -
127.0.0.1 - - [28/Mar/2025 14:56:21] "GET /logging HTTP/1.1" 200 -
C:\Users\denis\OneDrive\Рабочий стол\ПРС>
```

```
C:\WINDOWS\system32\cmd.exe

C:\Users\denis>curl http://localhost:8880
["logging-service: [\\"Hello\\",\\"Its me\\",\\"Den4ik\\"]\n","message-service: Message-service not implemented yet"]

C:\Users\denis>curl -X POST -d "txt=test retry" http://localhost:8880
{"error": "RetryError[<Future at 0x1ff1435ea40 state=finished raised ConnectionError>]"}

C:\Users\denis>
```

Перевірка duplicate:

```
C:\WINDOWS\system32\cmd.exe - python facade.py
C:\Users\denis\OneDrive\Рабочий стол\ПР>python facade.py
* Serving Flask app 'facade'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:8880
Press CTRL+C to quit
127.0.0.1 - - [28/Mar/2025 14:56:21] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [28/Mar/2025 15:00:24] "POST / HTTP/1.1" 500 -

C:\WINDOWS\system32\cmd.exe - python loggingser.py
C:\Users\denis\OneDrive\Рабочий стол\ПР>python loggingser.py
* Serving Flask app 'loggingser'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:8881
Press CTRL+C to quit
Message: Duplicate test
127.0.0.1 - - [28/Mar/2025 15:04:21] "POST /logging HTTP/1.1" 201 -
Duplicate message detected: 1111-2222-3333
127.0.0.1 - - [28/Mar/2025 15:04:21] "POST /logging HTTP/1.1" 200 -
Duplicate message detected: 1111-2222-3333
127.0.0.1 - - [28/Mar/2025 15:04:22] "POST /logging HTTP/1.1" 200 -

C:\WINDOWS\system32\cmd.exe
C:\Users\denis>curl http://localhost:8880
["logging-service: [\Hello\", \"Its me\", \"Den4ik\"]\n", "message-service: Message-service not implemented yet"]

C:\Users\denis>curl -X POST -d "txt=test retry" http://localhost:8880
{"error": "RetryError[<Future at 0x1ff1435ea40 state=finished raised ConnectionError>]"}

C:\Users\denis>curl -X POST -d "id=1111-2222-3333&txt=Duplicate test" http://localhost:8881/logging
"Message logged"

C:\Users\denis>curl -X POST -d "id=1111-2222-3333&txt=Duplicate test" http://localhost:8881/logging
"Duplicate detected"

C:\Users\denis>curl -X POST -d "id=1111-2222-3333&txt=Duplicate test" http://localhost:8881/logging
"Duplicate detected"

C:\Users\denis>
```

Чез грпс:

```
C:\WINDOWS\system32\cmd.exe - python facade_grpc.py
C:\Users\denis\OneDrive\Рабочий стол\ПР>python facade_grpc.py
* Serving Flask app 'facade_grpc'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:8880
Press CTRL+C to quit
127.0.0.1 - - [28/Mar/2025 15:25:38] "POST / HTTP/1.1" 200 -
127.0.0.1 - - [28/Mar/2025 15:25:42] "POST / HTTP/1.1" 200 -
127.0.0.1 - - [28/Mar/2025 15:25:45] "POST / HTTP/1.1" 200 -
[2025-03-28 15:25:54.158] ERROR in app: Exception on / [GET]

C:\WINDOWS\system32\cmd.exe
C:\Users\denis>curl http://localhost:8880
["logging-service: [\Hello\", \"Its me\", \"Den4ik\"]\n", "message-service: Message-service not implemented yet"]

C:\Users\denis>curl -X POST -d "txt=test retry" http://localhost:8880
{"error": "RetryError[<Future at 0x1ff1435ea40 state=finished raised ConnectionError>]"}

C:\Users\denis>curl -X POST -d "id=1111-2222-3333&txt=Duplicate test" http://localhost:8881/logging
"Message logged"

C:\Users\denis>curl -X POST -d "id=1111-2222-3333&txt=Duplicate test" http://localhost:8881/logging
"Duplicate detected"

C:\Users\denis>curl -X POST -d "id=1111-2222-3333&txt=Duplicate test" http://localhost:8881/logging
"Duplicate detected"

C:\Users\denis>curl -X POST -d "txt=Hello" http://localhost:8880
{"id": "31270763-12df-44b3-8310-07d57aa27328", "status": "Logged"}

C:\Users\denis>curl -X POST -d "txt=Its me" http://localhost:8880
{"id": "9dcffabb-c9e8-4823-9ff4-be8923e24ef8", "status": "Logged"}

C:\Users\denis>curl -X POST -d "txt=Den4ik" http://localhost:8880
{"id": "eca2d37e-5375-4fb4-aeae-88829afff16f", "status": "Logged"}

C:\WINDOWS\system32\cmd.exe - python logging_grpc.py
C:\Users\denis\OneDrive\Рабочий стол\ПР>python logging_grpc.py
Message: Hello
Message: Its me
Message: Den4ik

C:\WINDOWS\system32\cmd.exe
C:\Users\denis>curl -X POST -d "id=1111-2222-3333&txt=Duplicate test" http://localhost:8881/logging
"Duplicate detected"

C:\Users\denis>curl -X POST -d "txt=Hello" http://localhost:8880
{"id": "31270763-12df-44b3-8310-07d57aa27328", "status": "Logged"}

C:\Users\denis>curl -X POST -d "txt=Its me" http://localhost:8880
{"id": "9dcffabb-c9e8-4823-9ff4-be8923e24ef8", "status": "Logged"}

C:\Users\denis>curl -X POST -d "txt=Den4ik" http://localhost:8880
{"id": "eca2d37e-5375-4fb4-aeae-88829afff16f", "status": "Logged"}

C:\Users\denis>curl http://localhost:8880
<idctype html>
<html lang=en>
<title>500 Internal Server Error</title>
<h1>Internal Server Error</h1>
<p>The server encountered an internal error and was unable to complete your request. Either the server is overloaded or there is an error in the application.</p>

C:\Users\denis>curl http://localhost:8880
["Hello", "Its me", "Den4ik"]

C:\Users\denis>
```

