Összetettebb példa

Egy minőségbiztosító rendszer mérőgépének 3 állapotát küldjük egy 'qualityQueue' nevű üzenetsorra. Készítsen egy több komponensből álló alkalmazást, amely 2 kliensen keresztül kommunikál az üzenetsorral az alábbi módon:

- Az első kliens, ami a mérőgépre helyezett érzékelőre kapcsolódik a 'qualityQueue' üzenetsorra pont-pont csatlakozással véletlenszerűen GOOD, EXCELLENT és WRONG üzeneteket küld másodpercenként.
- **Készítsen egy komponenst** amely a 'GOOD', 'EXCELLENT' és a 'WRONG' üzeneteket leolvassa a qualityQueue sorról és gyűjti. Minden 10 megkapott azonos üzenet után a 'qualityStatistics' sorra küld egy üzenetet, amiben azt jelzi, hogy 10 (adott minőségű) üzenetet feldolgozott.
- **Készítsen egy második klienst**, ami a 'qualityStatistics' sorrol olvassa a statisztikát és a konzolba kiírja hogy pl. '10 'WRONG' messages has been processed'

A fenti feladatot a http://docker.iit.uni-miskolc.hu-n keretrendszerben oldjuk meg.

RabbitMQ indítása docker-ben

A feladat megoldásához több instance-t (konzolt) érdemes indítani. Az első konzol fogja a rabbitMQ szervert indítani. Adjunk hozzá egy konzolt (node 1) és futtassuk a következő parancsot:

```
docker run -it --rm --name rabbitmq -p 5672:5672 -p 15672:15672
rabbitmq:3.11-management
</code>

A futtatás után a rabitMQ management konzol elérhető az 15672-es porton, a
guest/guest megadásával. A bal oldali listában láthatjuk a node1 10.x.y.z
belső IP címét, amit használhatunk a kliensekben.

Hozzunk létre egy másik konzolt és indítsuk el az alábbi parancsot:

<code>
pip install pika
```

Ezzel telepítettük a pika modult, ami a rabbitMQ-hoz való csatlakozást biztosítja.

Hozzuk létre a quality_message_sender.py-t:

```python
import pika
import random
import time

qualities = ['EXCELLENT', 'GOOD', 'WRONG']

# RabbitMQ settings
```

```
connection = pika.BlockingConnection(pika.ConnectionParameters('10.x.y.z'))
channel = connection.channel()

channel.queue_declare(queue='qualityQueue')

while True:
    quality = random.choice(qualities)
    message = f'{quality}'
    channel.basic_publish(exchange='', routing_key='qualityQueue',
body=message)
    print(f'Sent message: {message}')
    time.sleep(1)

connection.close()
```

Állítsuk be a megfelelő IP címet a *pika.ConnectionParameters()* függvénynél.

Indítsunk egy másik terminált, futtassuk a '*pip install pika*' parancsot. Majd hozzuk létre a *quality_consumer.py* állományt az alábbi kóddal.

A *pika.ConnectionParameters()*-t értelem szerűen állítsuk be.

```
import pika

# RabbitMQ settings
connection = pika.BlockingConnection(pika.ConnectionParameters('10.x.y.z'))
channel = connection.channel()

channel.queue_declare(queue='qualityStatistics')

def callback(ch, method, properties, body):
    quality = body.decode()
    print(f'{quality} messages has been processed')
    ch.basic_ack(delivery_tag=method.delivery_tag)

channel.basic_consume(queue='qualityStatistics',
on_message_callback=callback)

print('Waiting for quality statistics...')
channel.start_consuming()
```