Cloud Computing

Project report : lab work 3

Réalisé par :
Joe AL HASROUNI
Mohammed GHAZLANE
Youssef MOUSSAOUI
Ghizlane ZEHNINE

Encadré par :
Mme Charlotte Laclau

Année universitaire : 2020-2021

Project Architecture

The project has a classic Web-Queue-Worker architecture.The core component of this architecture are a web front-end that serves client requests, and a worker that performs tasks. The client communicates with the worker through a message queue.

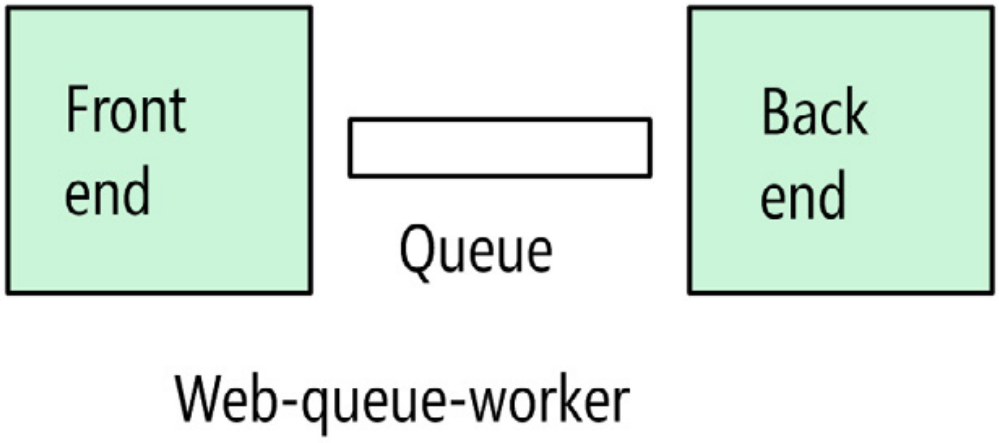

Figure 1 – Web-queue-worker architecture

Chapitre 1

Instance and security group creation

The server is an instance created in aws-educate. The instance is an EC2-worker and the characteristics chosen to be free for a student. Mainly, it is a 64-bit linux machine t2.micro that have us-east-1 as a region. We can access it by SSH using puTTy for example thanks to the security group created that can allow SSH connections to b established. We also generate a key for the machine and this key also allows puTTy to reach our instance correctly.

We also need to change every 3 hours the config file on the machine because the session of aws-educate expires after this duration and we have to renew it.

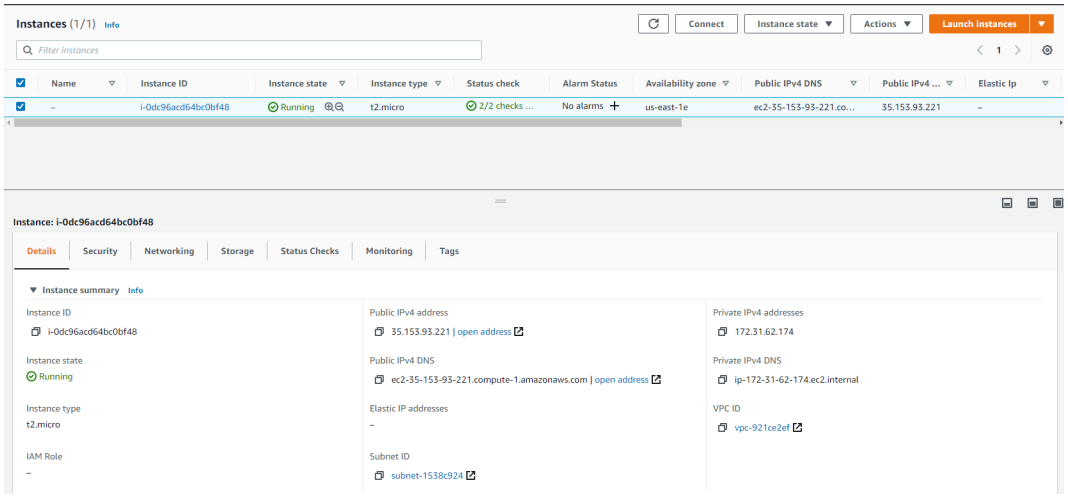

FIGURE 1.1 – Client code

Chapitre 2

Client / EC2 worker

For the first part of this project we have an application composed of a client that communicates with a server. First, the client creates a SQS queue for requests and use it to submit a list of 20 integers to the server who reads the integers from the request queue and calculates the median, minimum, maximum, average and sum of the integers. Then, the server creates another SQS queue for the response and submits the results in it. Finally, the client reads the response of the server fro the response queue.

```
import boto3
import time


# Get the service resource
sqs = boto3.resource('sqs')

# Create the queue. This returns an SQS.Queue instance
queue = sqs.create_queue(QueueName='requestQueue', Attributes={
                'DelaySeconds': '0'
                })

print(queue.url)



val = input("Enter your value: ")
response = queue.send_message(MessageBody=val)


time.sleep(4)

# Get the queue
queueResponse = sqs.get_queue_by_name(QueueName='responseQueue')
# Process messages by printing out body and optional author name
for message in queueResponse.receive_messages():
        print(format(message.body))
        message.delete(QueueUrl='https://sqs.us-east-1.amazonaws.com/580530755374/responseQueue', ReceiptHandle=message.receipt_handle)
```

FIGURE 2.1 – Client code

```python
import boto3
import statistics
import time


# Get the service resource
sqs1 = boto3.resource('sqs')


#Get the queue
queueRequest = sqs1.get_queue_by_name(QueueName="requestQueue")

queue = sqs1.create_queue(QueueName='responseQueue', Attributes={
                'DelaySeconds': '0',

                })

while(1):


    try:
        for message in queueRequest.receive_messages():

            t=[int(n) for n in (format(message.body)).split()]
            print(t)
            minimum = min(t)
            maximum = max(t)
            sum1 = sum(t)
            length = len(t)
            average = sum1/length
            median = statistics.median(t)

            message1 = "minimum is %d " %(minimum) + " maximum is %d " %(maximum) +  " average is %d  " %(average) + " median is %d " %(median)
            print(message1)
            response = queue.send_message(MessageBody=message1)

            message.delete(QueueUrl='https://sqs.us-east-1.amazonaws.com/580530755374/requestQueue', ReceiptHandle=message.receipt_handle)
            print("request queue deleted")



    except:
        print("Exception")
```

FIGURE 2.2 – Server code

Chapitre 3

Image Processing Application

In the second part of the project, we still have the same client-server architecture. The difference is that we use the S3 service provided by AWS in order to upload and download images. So at first, the client upload an image on an S3 bucket and sends via the inbox queue the key to this image to the server(ec2-instace). Then, the instance download this image from the S3 bucket using the key provided in the inbox queue. The server code do some basic image processing built in function provided by the skiimage library and retruns the adjusted image to the same S3 bucket as before. Now the client can have the adjusted image and can access it using the key provided by the outbox queue created by the server. The client can now download the new image to a local address on its local machine.

```
import boto3
import time


s3=boto3.client('s3')
s3.upload_file('lab3-image.jpg','group13-lab3.part2','s3_image.jpg')

sqs1= boto3.resource('sqs')
# Create the queue. This returns an SQS.Queue instance
inboxQueue = sqs1.create_queue(QueueName='inboxQueue', Attributes={
                'DelaySeconds': '0'
                })



response = inboxQueue.send_message(MessageBody='s3_image.jpg')

time.sleep(6)


outboxQueue = sqs1.get_queue_by_name(QueueName="outboxQueue")

for message in outboxQueue.receive_messages():
    key=format(message.body)
    print(key)

    s3 = boto3.client('s3')
    s3.download_file('group13-lab3.part2', key ,'C:\\Users\\USER\\Desktop\\aws\\'+key)

    message.delete(QueueUrl='https://sqs.us-east-1.amazonaws.com/580530755374/outboxQueue', ReceiptHandle=message.receipt_handle)
    print("request queue deleted")
```

FIGURE 3.1 – Client code

```python
import boto3
from skimage import io
from skimage import exposure

sqs1 = boto3.resource('sqs')

#Get the inbox queue
inboxQueue = sqs1.get_queue_by_name(QueueName="inboxQueue")

# Create the queue. This returns an SQS.Queue instance
outboxQueue = sqs1.create_queue(QueueName='outboxQueue', Attributes={
                'DelaySeconds': '0'
                })


print(outboxQueue.url)

while(1):

        try:
                for message in inboxQueue.receive_messages():
                        key=format(message.body)
                        print(key)

                        s3 = boto3.client('s3')
                        s3.download_file('group13-lab3.part2', key ,'./'+key)


                        loadedPic = io.imread('./'+key)
                        image_equalized = exposure.equalize_hist(loadedPic)
                        io.imsave('editedPic.png',image_equalized)

                        s3=boto3.client('s3')
                        s3.upload_file('editedPic.png','group13-lab3.part2','s3_Newimage.jpg')
                        response = outboxQueue.send_message(MessageBody='s3_Newimage.jpg')

                        message.delete(QueueUrl='https://sqs.us-east-1.amazonaws.com/580530755374/inboxQueue', ReceiptHandle=message.receipt_handle)
    except:
                print("Exception")
```

FIGURE 3.2 – Server code