REPORT Lab 3- Cloud Computing using AWS

Hamed RAHIMI, Aleksei PASHININ, Jonathan MALLET

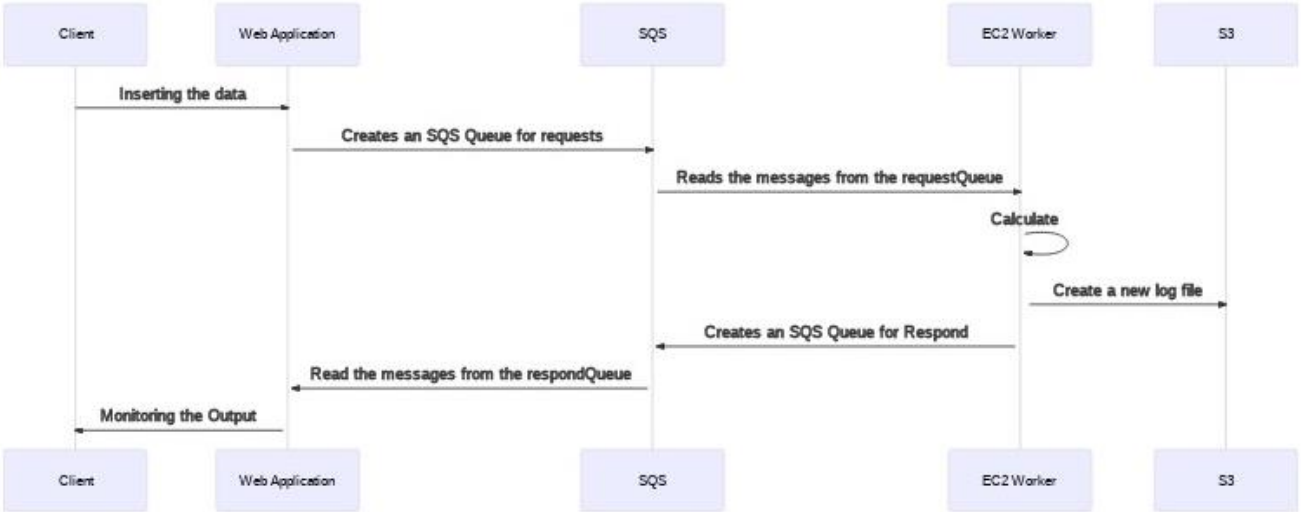
ABSTRACT:

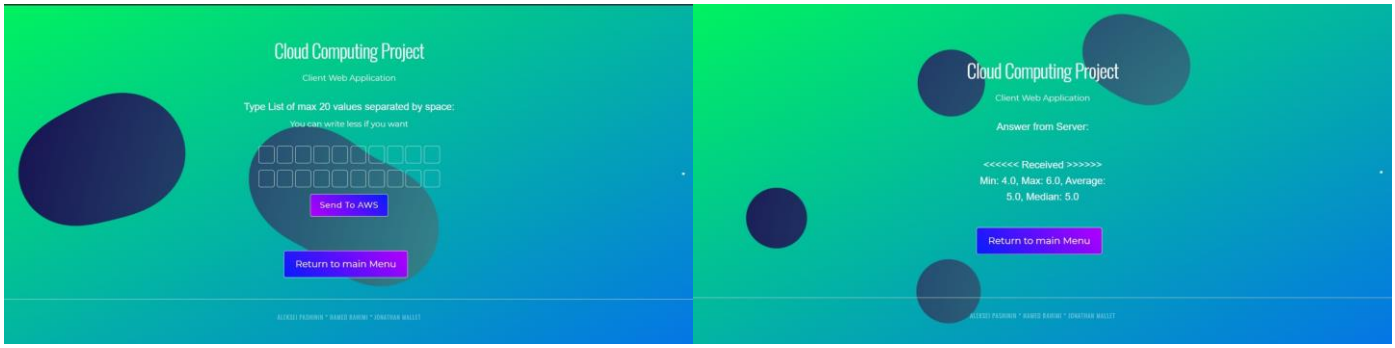
Amazon Web Services is a subsidiary of Amazon providing on-demand cloud computing platforms and APIs to individuals, companies, and governments, on a metered pay-as-you-go basis. Amazon Web Services offers reliable, scalable, and inexpensive cloud computing services. In this project, we aim to use AWS to develop two computing systems within a web application:

1- CloudCal and 2- CloudPic

- **PART 1. CloudCal**

CloudCal is an AWS application in which a client submits a request, which is composed of a list of numerical values, to a worker on EC2. The EC2 worker is a Python application that receives the request and calculates the min, max, average, and median, and then send them back to the client. Moreover, EC2 Worker stores the calculate values as a log file on S3 Amazon Storage Cloud. The Sequence Diagram is as follows.





The Architecture of the project is consisting of two main elements: 1-Client (on the web-application) 2-EC2Worker (on AWS application).

- Client

Client is a python code hosted on local machines and is responsible to create an SQS Queue for requests (called the requestQueue) on order to submit a request containing a list of 20 integers to cloud and receives the response from the EC2 worker. Implementation of the code is as follows.

```
def startDialog():
    StringFromUser = input("Type List of 20 values separated by space: ")
    listOfValues = [int(s) for s in StringFromUser.split()]
    print("Sending message >>> ")
    for i in listOfValues:
        print(i, end=" ")
    #print(StringFromUser)
    return (StringFromUser)


def receive():
    messages = queue.receive_messages()
    print('Wait answer. Answers for this moment: ', len(messages))
    #print('Received >>>', messages)
    while True:
            for message in messages:
                data = stringToList(message.body)
                message.delete()
            return(data)

if __name__ == "__main__":
    sqs = boto3.resource('sqs')
    queue = sqs.get_queue_by_name(QueueName='requestQueue')
    sendList = startDialog()
    response = queue.send_message(MessageBody=sendList)
    time.sleep(3)
    queue = sqs.create_queue(QueueName='responseQueue')
    data = receive()
    print(data)
```

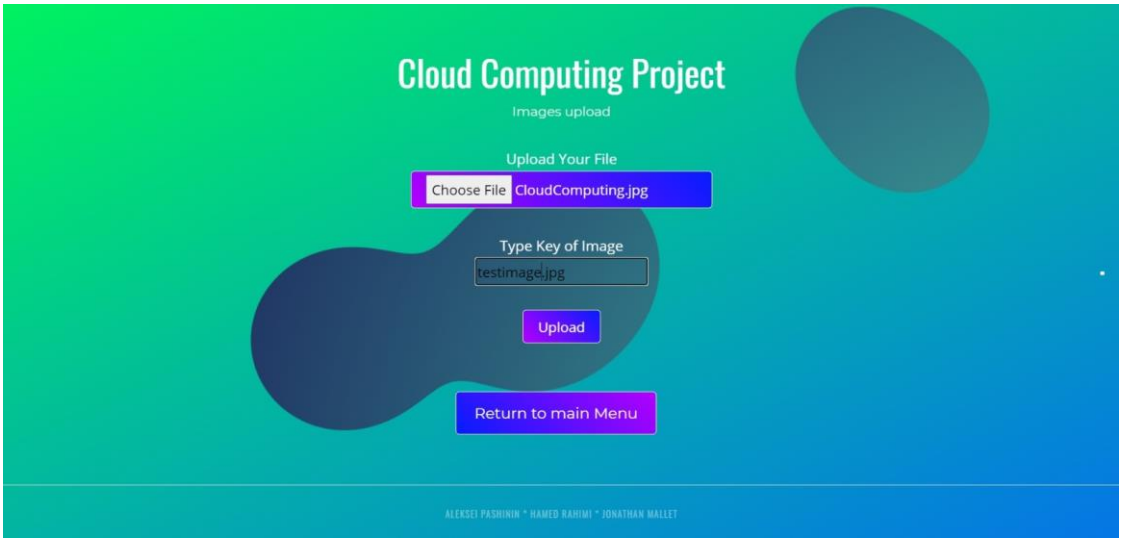
- EC2 Worker

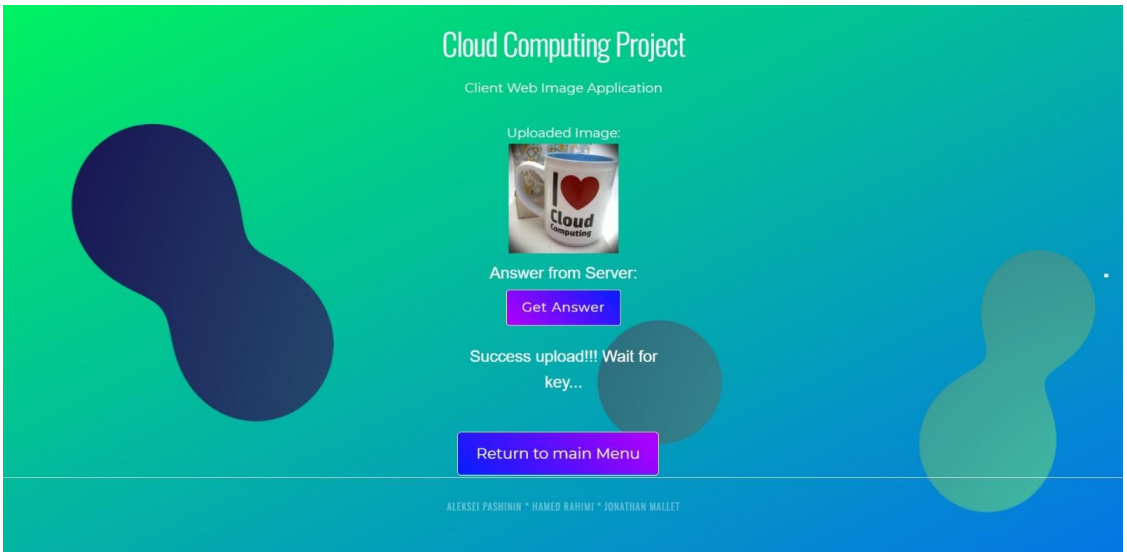
EC2 Worker is a python code hosted on cloud and runs on an EC2 instance in order to reads the messages from the requestQueue, Calculates the outputs, creates a response queue (called responseQueue) and submits the result, and finally, Create a new log file (a text file) and store it on Amazon S3.

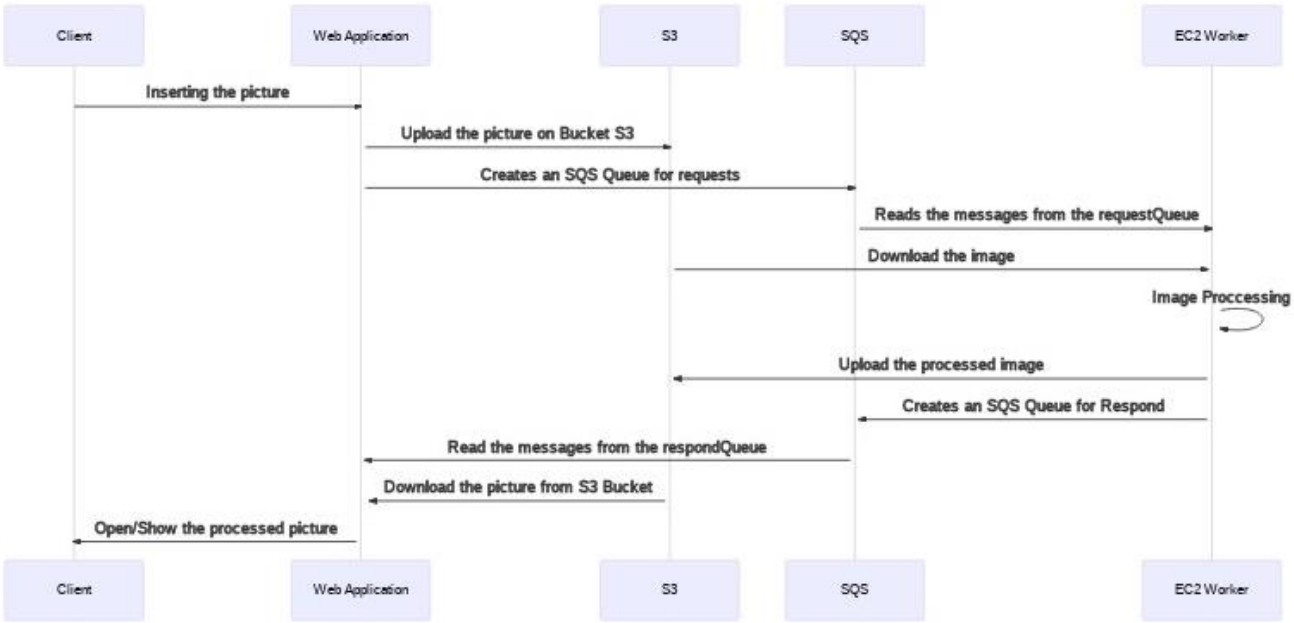
```
# CREATE THE QUEUE, GET MESSAGE, CALCULATION AND SEND MESSAGE
sqs = boto3.resource('sqs')
s3 = boto3.resource('s3')
bucket = s3.Bucket('mybucketnumber1')
key = 'CloudLogs.txt'
print('Received: ')
queue = sqs.create_queue(QueueName='CloudComputingProject7')
# resetMessages(queue)
while True:
    messages = queue.receive_messages()
    print('Number of messages received: ', len(messages))
    for message in messages:
        print('Received: ', message.body)
        time.sleep(5)
        resultList = stringToList(message.body)
        message.delete()
        print('Result of Calculation: ', mathCalculate(resultList))
        sendList = listToString(mathCalculate(resultList))
        print('Send Message: ', sendList)
        queue = sqs.get_queue_by_name(QueueName='CloudComputingProject8')
        # Create a new message
        response = queue.send_message(MessageBody=sendList)
        SS3.writefile(sendList, bucket, key)
        sys.exit("Stop code")

# GENERAL MAIN
if __name__ == "__main__":
    print("Hello World!")
```

- **PART II. CloudPic**

CloudPic is an AWS application that enhances the quality of a picture up to 30% in which a user uploads an image to the cloud through a bucket in S3 [3], places the key to this uploaded image in the inbox, and queues to EC2 Worker and waits for this message to be processed. EC2 worker wait for messages to appear in the inbox queue, and then it retrieves the referenced image file from S3, enhance the quality of the image, place the resulting image in S3, place the key to the result image file in the outbox queue, and delete the message from the inbox queue. The Sequence Diagram is as follows.


Cloud Computing Project
Images upload
Upload Your File
Choose File
CloudComputing.jpg
Type Key of Image
testimage.jpg
Upload
Return to main Menu
ALEKSEI PASHININ * HAMED RAHIMI * JONATHAN MALLET


Cloud Computing Project
Client Web Image Application
Uploaded Image:
Answer from Server:
Get Answer
Success upload!!! Wait for key...
Return to main Menu
ALEKSEI PASHININ * HAMED RAHIMI * JONATHAN MALLET


Client
Web Application
S3
SQS
EC2 Worker
Inserting the picture
Upload the picture on Bucket S3
Creates an SQS Queue for requests
Reads the messages from the requestQueue
Download the image
Image Proccessing
Upload the processed image
Creates an SQS Queue for Respond
Read the messages from the respondQueue
Download the picture from S3 Bucket
Open/Show the processed picture
Client
Web Application
S3
SQS
EC2 Worker

The Architecture of the project is consisting of two main elements: 1-Client (on the web-application) 2-EC2Worker (on AWS application).

- Client

Client is a python code hosted on local machines and is responsible to upload an image into the cloud through a bucket in S3, to create an SQS Queue for requests (called the requestQueue), and to receive the response from the EC2 worker.

- EC2 Worker

EC2 Worker is a python code hosted on cloud and runs on an EC2 instance in order to reads the messages from the requestQueue, download the image from S3 bucket. The code of this part is as follows.

```
# GENERAL CODE WITH COMMENTS
sqs = boto3.resource('sqs')
s3 = boto3.resource('s3')
queue = sqs.create_queue(QueueName='outbox1')
while True:
    messages = queue.receive_messages()
    for message in messages:
        # WATING FOR RECIEVING A REQUEST FROM WEB APP FOR IMAGE PROCCESING
        print("New request for image processing is recieved!")
        tempos = message.body
        print(tempos)
        resultList = list(tempos.split(" "))
        message.delete()
        key = resultList[0]
        bucketName = resultList[1]
        print(key)
        print(bucketName)
        myBucket = s3.Bucket(bucketName)
        myBucket.download_file(key, 'image.jpg')
        time.sleep(5)

        # IMAGE PROCCESSING AND SAVING NEW IMAGE
        print("Image is downloaded and now it is under processing")
        im = Image.open('image.jpg')
        enh = ImageEnhance.Contrast(im)
        enh.enhance(1.8).save('new_image.jpg')
        time.sleep(5)
```

Then, as shows as follows it enhances the pricture, upload the processed image in the bucket S3, creates a response queue (called responseQueue) to the Client.

```
        # UPLOADING THE NEW IMAGE IN A NEW BUCKET WITH A NEW KEY
        print("Image processed and now it is being prepared to be stored in s3 bucket")
        new_key = "newImage1.jpg"
        myBucket.upload_file('new_image.jpg', new_key)
        time.sleep(1)

        # SENDING QUEUE TO WEB APP FOR INFORMING
        queue = sqs.get_queue_by_name(QueueName='inbox1')
        response = queue.send_message(MessageBody=new_key)
        time.sleep(1)
        print("The processed image is stored and the web app is informed")
        sys.exit("Stop code")

# GENERAL MAIN
if __name__ == "__main__":
    print("Hello World!")
```