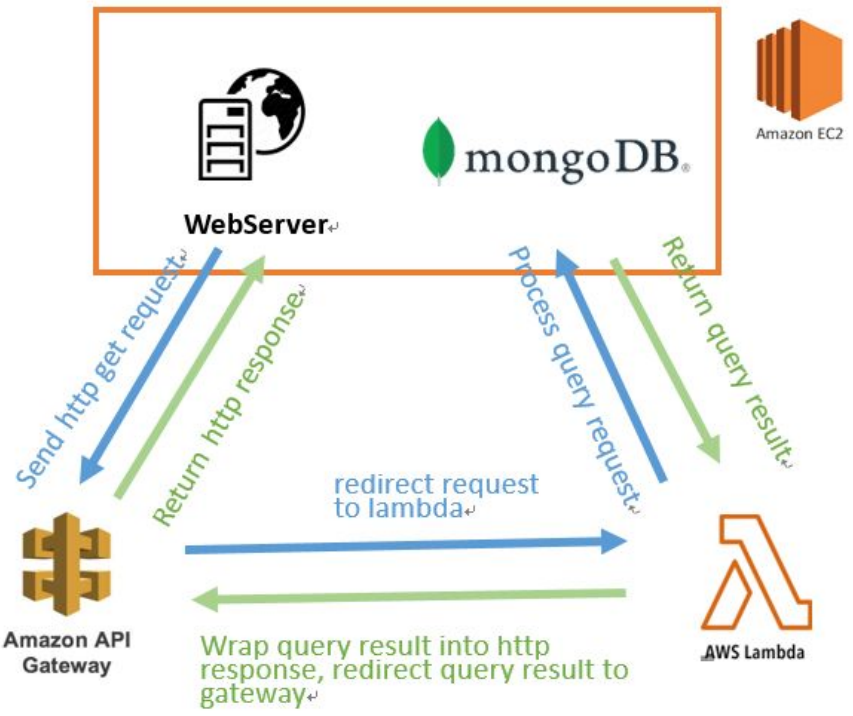
HW2 Report

r09922102 資工所碩一 韓秉勳

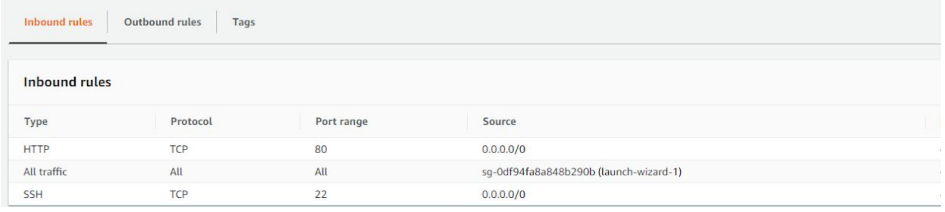
1. Design flow



2. Environment
   Mongo & webserver - aws ec2 t2.micro ubuntu 20.04
   amazon lambda & API gateway

3. github link - https://github.com/nba556677go/cloud_computing2020.git
   (service is not running due to AWS payment, please contact me if following installation not working)

4. installation guide
   a. launch a new ec2 instance, security group config as follow



Inbound rules | Outbound rules | Tags

Inbound rules

Type	Protocol	Port range	Source
HTTP	TCP	80	0.0.0.0/0
All traffic	All	All	sg-0df94fa8a848b290b (launch-wizard-1)
SSH	TCP	22	0.0.0.0/0

Inbound rules | Outbound rules | Tags

Outbound rules

Type	Protocol	Port range	Destination
HTTP	TCP	80	0.0.0.0/0
Custom TCP	TCP	8080	0.0.0.0/0
All traffic	All	All	0.0.0.0/0
All traffic	All	All	sg-0df94fa8a848b290b (launch-wizard-1)

b. git clone https://github.com/nba556677go/cloud_computing2020.git
c. cd hw2
d. bash init.sh (if some pip3 or docker installment failed, please check init.sh and rerun the installation command)
e. upload function.zip to amazon lambda (set under same security group as ec2)
f. set environment variable DB_HOST to the previous ec2 private IP (WARNING - do not use public ip in DB_HOST, lambda function cannot resolve public IP!)

Environment variables (1) Edit

The environment variables below are encrypted at rest with the default Lambda service key.

Key	Value
DB_HOST	172.31.19.238

g. create API gateway for this lambda function. set to RESTAPI.

API Gateway: getDBdata-API

arn:aws:execute-api:us-east-1:145461671858:cyu5jgvlak/*/*/getDBdata

▼ Details

API endpoint: **https://cyu5jgvlak.execute-api.us-east-1.amazonaws.com/default/getDBdata**

API type: **REST**

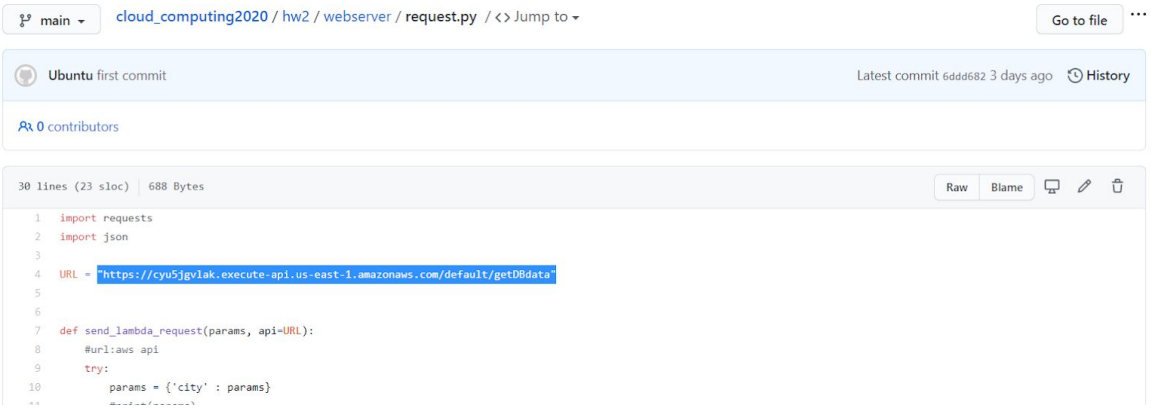
Authorization: **NONE**

Method: **ANY**

Resource path: **/getDBdata**

Stage: **default**

h. change webserver/request.py api URL



i. open web page and type in ec2 ip. you should see the following web page

j. type in existing city in mongo db, such as “hampden”, it will return location and population for you

if type in a city not in DB, it returns wrong input.

5. design explain
   a. Webserver
   using python flask as simple webserver. it acts as a simple frontend and send http request to API to receive city information. I used **volume**

to mirror my code into container, so I can modify and persistent my changes without extra modification.

b. MongoDB
using Mongo is because I don't want to be bound to aws services. However, authenticatication and inserting data could be troublesome, since I spent most of my time debugging in here. I utilized pyMongo, its python api, to do insertion and query. Due to EC2 computation limits, I scale down the origin dataset to merely 200 inputs. Webserver and MongoDB all started with docker-compose, and I volumed mongoDB data for persistence. One only need to insert data once in init.sh, then it won't need to be constructed again. I also **fixed Mongo IP** to easily configure connection issues.

```yaml
version: '3'

services:
    webserver:
        image: webserver
        container_name: webserver
        build:
            context: .
            dockerfile: webserver/Dockerfile
        #expose port -- host:container (not specifing host port -> random assign! )
        ports:
            - "80:5000"
        #volume src path - relevant(must include ./) dest path - need absolute path!!
        volumes:
            - ./webserver:/opt
        entrypoint: python3 /opt/app.py
        networks:
            web_net:
                aliases:
                - webserver
    mongo:
        image: mongo
        container_name: mongo
        restart: always
        environment:
            MONGO_INITDB_ROOT_USERNAME: root
            MONGO_INITDB_ROOT_PASSWORD: root
        ports:
            - "27017:27017"
        volumes:
            - ./mongo/data:/data/db
        networks:
            web_net:
                aliases:
                - mongo
                ipv4_address: 172.18.0.3
```

c. aws lambda & api gateway - lambda receives http request from webserver via gateway api. it perform mongo queries with the input event and return value in mongo DB. Since we need to import pymongo, we need to upload the pip installed package to lambda, which is our function.zip in github. As a reminder, **ec2 can only**

connect with lambda via private IP, and it is feasible since it won't change after rebooting. Using public IP causes connection error!

6. http request

the webserver grab input value from html, and send it as the parameter of http request

```
def send_lambda_request(params, api=URL):
    #url:aws api
    try:
        params = {'city' : params}
        #print(params)
        r = requests.get(url=api, params=params)
        data = r.text
    except Exception as error:
        data = str(error)

    return data
```

in aws lambda, the parameter we send via http will be stored in the field of `multiValueQueryStringParameters` in `event`. We can get our city name via this field, then perform DB query to retrieve information. Webserver will process the response and deal with wrong inputs. upper/lower input case was dealt before sending request.

```
def lambda_handler(event, context):
    resultJson = None
    try:
        print("event: ", event['multiValueQueryStringParameters'])
        request_city = {"city" : event['multiValueQueryStringParameters']['city'][0]}
        print("request_city",  request_city)
        logger.info("logging...")
        #print("dbnames: ", client.list_database_names())
        # TODO implement
        col = client["country"]['country_info']
        print("collection set!")
        #result = col.find_one({"city" : "HAMPDEN"})
        result = col.find_one(request_city)
        print("collection found", result)
        #print(col.find())
        resultJson = {
            'statusCode': 200,
            'body': json.dumps(result)
```