CITS5503 Lab2

Wenxiao Zhang 22792191

Create an EC2 instance using awscli

[1] Create a security group

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503$ aws ec2 create-security-group --
group-name 22792191-sg --description "security group for development envir
onment"
{
    "GroupId": "sg-0625d7c77737ad40c"
}
```

[2] Authorise inbound traffic for ssh

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503$ aws ec2 authorize-security-group
-ingress --group-name 22792191-sg --protocol tcp --port 22 --cidr 0.0.0.0/
0
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-082d3833d3ffd8ff6",
            "GroupId": "sg-0625d7c77737ad40c",
            "GroupOwnerId": "523265914192",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv4": "0.0.0.0/0"
        }
    ]
}
```

[3] Create a key pair that will allow you to ssh to the EC2 instance

we make a directory named `.ssh`, then copy the key pair to the directory.

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503$ mkdir ~/.ssh
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503$ chmod 400 22792191-key.pem
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503$ cp 22792191-key.pem ~/.ssh
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503$ chmod 400 ~/.ssh/22792191-key.pem
```

[4] Create the instance and note the instance id

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503$ aws ec2 run-instances --image-id ami-d38a4ab1 --secu
rity-group-ids 22792191-sg --count 1 --instance-type t2.micro --key-name 22792191-key --query
'Instances[0].InstanceId'
"i-034e9ff70b3e45d0b"
```

add a tag to the instance:

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503$ aws ec2 create-tags --resources i-034e9ff70b3e45d0b
--tags Key=Name,Value=22792191
```

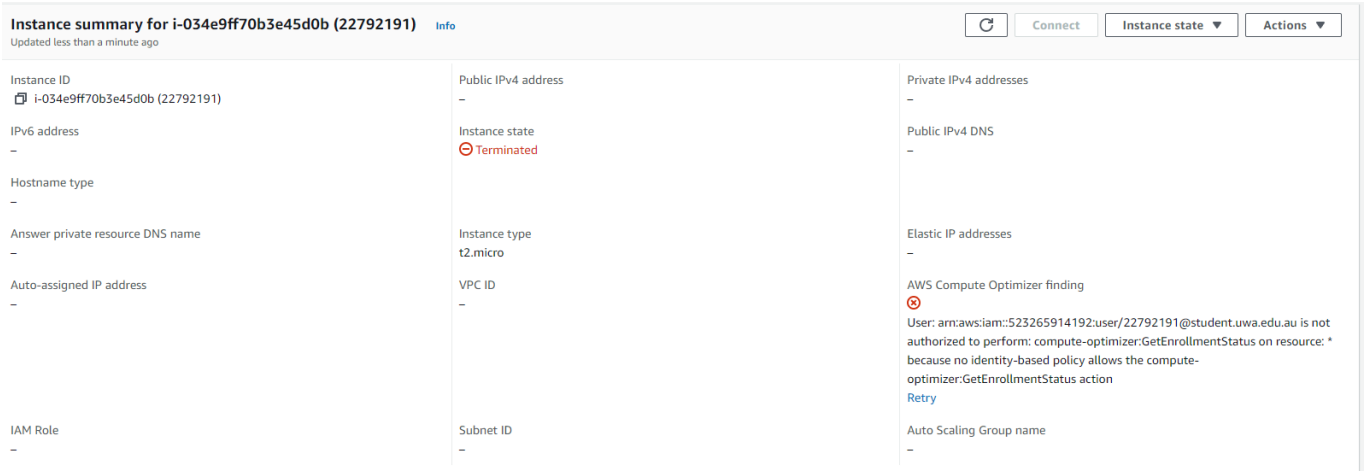
[5] Get the public IP address

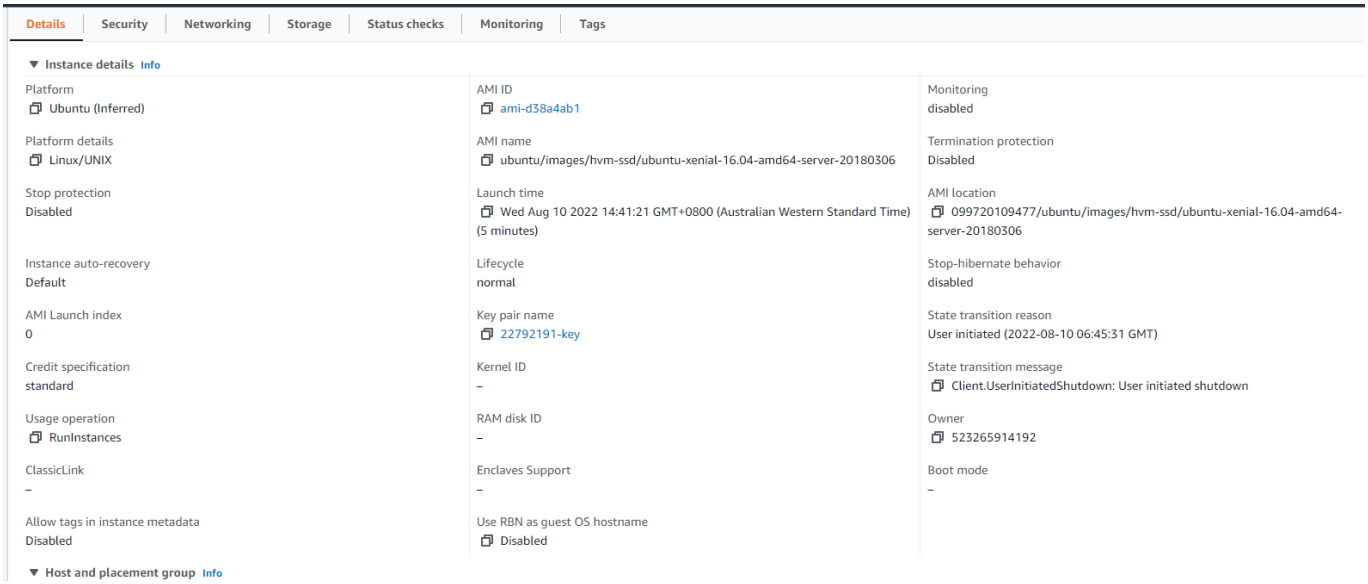
```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503$ aws ec2 describe-instances --instance-ids i-034e9ff7
0b3e45d0b --query 'Reservations[0].Instances[0].PublicIpAddress'
"54.253.178.81"
```

[6] Connect to the instance

```
"54.253.178.81"
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503$ ssh -i 22792191-key.pem ubuntu@54.253.178.81
The authenticity of host '54.253.178.81 (54.253.178.81)' can't be established.
ECDSA key fingerprint is SHA256:NkW0uSLkAE1sTLV1DcWmnB5y88CiVSYW4CETHjfw0og.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '54.253.178.81' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.4.0-1052-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

0 packages can be updated.
0 updates are security updates.



The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-40-221:~$
```

[7] Look at the instance using the AWS console





[8] Terminate the instance

```
ubuntu@ip-172-31-40-221:~$ exit
logout
Connection to 54.253.178.81 closed.
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503$ aws ec2 terminate-instances --instance-ids i-034e9ff
70b3e45d0b
{
    "TerminatingInstances": [
        {
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "InstanceId": "i-034e9ff70b3e45d0b",
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

Create an EC2 instance with Python Boto script

Repeat the steps above using the equivalent Boto commands in a python script. The script should output the IP address to connect to.

Step 1 – Create a security group

python code:

create_SG.py | ssh.py | key_pair.py | create_instance.py | get_ip.py

2022s2 > cits5503 > labs > lab2 > create_SG.py > ...

```python
import boto3

AWS_REGION = "ap-southeast-2"
EC2_RESOURCE = boto3.resource('ec2', region_name=AWS_REGION)

security_group = EC2_RESOURCE.create_security_group(
    Description='Allow inbound SSH traffic',
    GroupName='22792191',
    TagSpecifications=[
        {
            'ResourceType': 'security-group',
            'Tags': [
                {
                    'Key': 'Name',
                    'Value': '22792191'
                },
            ]
        },
    ],
)

print(f'Security Group Created {security_group.id} has been created in vpc {security_group.vpc_id}')
```

output:

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ python3 create_SG.py
Security Group Created sg-02e03d15c337cfec2 has been created in vpc vpc-0b754f714cd1af245
```

Step 2 - Authorise inbound traffic for ssh, from port/to port 22 indicates ssh, and CidrIp 0.0.0.0/0 indicates directions

python code:

create_SG.py | ssh.py | key_pair.py | create_instance.py | get_ip.py

2022s2 > cits5503 > labs > lab2 > ssh.py > ...

```python
import boto3

AWS_REGION = "ap-southeast-2"
EC2_RESOURCE = boto3.resource('ec2', region_name=AWS_REGION)
SECURITY_GROUP_ID = 'sg-02e03d15c337cfec2'

security_group = EC2_RESOURCE.SecurityGroup(SECURITY_GROUP_ID)


response = security_group.authorize_ingress(
    CidrIp='0.0.0.0/0',
    FromPort=22,
    ToPort=22,
    IpProtocol='tcp',
)
print(f'Ingress successfully set {response}')
```

output:

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ python3 ssh.py
Ingress successfully set {'Return': True, 'SecurityGroupRules': [{'SecurityGroupRuleId': 'sgr-
0f4e9aaf610f20faf', 'GroupId': 'sg-02e03d15c337cfec2', 'GroupOwnerId': '523265914192', 'IsEgre
ss': False, 'IpProtocol': 'tcp', 'FromPort': 22, 'ToPort': 22, 'CidrIpv4': '0.0.0.0/0'}], 'Res
ponseMetadata': {'RequestId': '584cbfc2-aaeb-40e4-8958-28c8029f4d98', 'HTTPStatusCode': 200, '
HTTPHeaders': {'x-amzn-requestid': '584cbfc2-aaeb-40e4-8958-28c8029f4d98', 'cache-control': 'n
o-cache, no-store', 'strict-transport-security': 'max-age=31536000; includeSubDomains', 'conte
nt-type': 'text/xml;charset=UTF-8', 'content-length': '719', 'date': 'Thu, 11 Aug 2022 03:00:2
0 GMT', 'server': 'AmazonEC2'}, 'RetryAttempts': 0}}
```

Step 3 – Create the key pair via create_key_pair function to allow ssh into EC2 instance

python code:

create_SG.py | ssh.py | key_pair.py | create_instance.py | get_ip.py

2022s2 > cits5503 > labs > lab2 > key_pair.py > ...

```python
import boto3

AWS_REGION = "ap-southeast-2"
EC2_RESOURCE = boto3.resource('ec2', region_name=AWS_REGION)
SECURITY_GROUP_ID = 'sg-02e03d15c337cfec2'

key_pair = EC2_RESOURCE.create_key_pair(
    KeyName='Z22792191',
    TagSpecifications=[
        {
            'ResourceType': 'key-pair',
            'Tags': [
                {
                    'Key': 'Name',
                    'Value': 'Z22792191'
                },
            ]
        },
    ]
)
print( key_pair.key_fingerprint)
print(key_pair.key_material)
```

output:

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ python3 key_pair.py
b9:5d:67:eb:f1:bc:40:49:c9:6a:45:13:3e:ff:34:3f:d9:ea:f9:ee
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAq2NlxKB4SP4g+n0RgjUI001NHuuc7GcantQCdhrOSz6xUks4
6f2sbk7TDqP2vaQ8pIwtkuhqXj2RGfftBFUEshBEtWT9uBVftWvBAbnaQqC/o9gh
zVqWO2e+tcr5oEuCbVzeogNQqWsRTKuwyhlptFGnJCs7p047BL4a7mSglSzlpBQD
FB8j+hNOCE5h7gDadizVQHpY9hYfsqe2hhTSe1M73KablUgVIRo6uNMfKR+ixJgp
n3/87QheCgPhv0PrqubXKnYY2rZfwHh2PTwGDV0fw6Q3/mixbiJ9kq1mqwt/cXww
kbzn6oHJ8qh4Hn1qq6teSLbqIsaJ1f/yQhfmJwIDAQABAoIBAQCckCR7RowbQesm
Z/ICnCf/vvHgeSPU4Sr0RIArG7mwCwk5P/Yx6sYFzoQHL8lK3Cl5t4Q6W44voNZY
fp9KAdBUSYA6pJMSApR3yoN57ClyDB6Esf/O9B2hDtisgUsgH1p6RQgrwBODcIdR
HtZVCHoDe0PQSp4n69MGNhOsCsiW+/VuEf0tTV6D4gycz3WPbukfsaJygIdCW1q6
nsFHYRpyYTdISJXSGHbi9ty8xZaZ1oIRuEX8fj0NeAFm2nWB1v7DCyZIOtE3ikLB
82iyKyVTunL1G5DLVVkpm7iXc7u/WGf/ZRvLQG9ZEkHaNHhQHaSVWqy2S9QKqB1R
Xw2EcE4BAoGBAPQQwWPQ5Ek/s4iLYDxHToT0ekoAt6SBSePoHxWQP5ym8+xC1iRI
oqoE8gtyMblJhfa6Nmi3G0k8PwCYiC9u/YPS5oQiHDvU+DHeBJ6Xy1jMsx9sjo5O
5W1oAxj8NT54m0GWU0qa/ryAFpDxxoCvJiwx/xfPiU+zqSXZafUgM3LnAoGBALPE
2/kLzRRuvIsDZ2gJpJjNB7Jzl0ljW3o2oGMVvUGmnqh8Rpis7v8zkLXWQyxNhlyX
El+vBkv0Su9Uqb1g93d8R6TCdUrfjR1CH+rY/kBNlCot3pzyli5Ssd1dX7OIVedg
q9Frh65pO1XSfGP1jMKpXOcLWWFYrF2Q8mzXvMrBAoGBAPFmJTGJ6lG8zOFQIaZd
TP9aJB5P6VOvgRsHer9ERna19usHDHEk2qbZb2H818uROgxYD/qwqS9dhF8dfw2T
dPJOxiuI5W4Ubkw8cUs/jiRNixmYsMN2q5vz1aR/yFX6ym0rt0RCW5SsCCXizVh4
ZIQwC0RMxF5tr+68xP1rlRefAoGAAz4DsaQy2LF+mQtArDGwVEkuYFuRWPN/zN57
Wg4kUbDNn8BODabg0J1/eKgaogvr5nObEzq25FvNli9yuEWtcOw+5ovTTbxmcSam
KLgtLb+4cmJNZ4tAhilbp+BtP3GwCvJFZwuHcO/Y8B1RvQPKvCAN2uTrgcP1p+Jb
XPcvIoECgYBxmz0GtVKmIBJQ4Sg7AVTvqNAFrTdVBTCHY70kWxT8kgNBAvDG+TUE
VKet796XSFz1KJgr4a9zdKsD99NytxYHAjyZ+otnX5yL1PPNY9bxwtG7xMVsFtBp
rpIYSYRK1aLSII2mzAT1TMMCU3fFtWOAqFwvK3EKvNt2UBaF09P5EA==
-----END RSA PRIVATE KEY-----
```

Step 4 – Create the instance via run_instance function and return the instance id

python code:

create_SG.py | ssh.py | key_pair.py | create_instance.py | get_ip.py

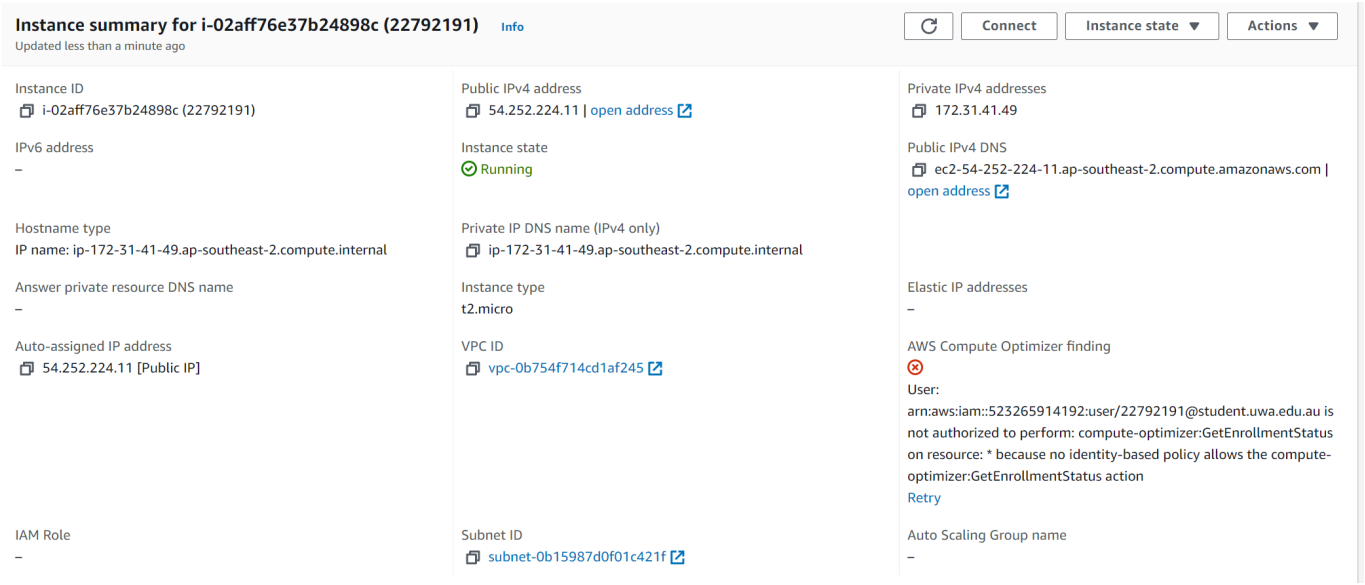
2022s2 > cits5503 > labs > lab2 > create_instance.py > ...

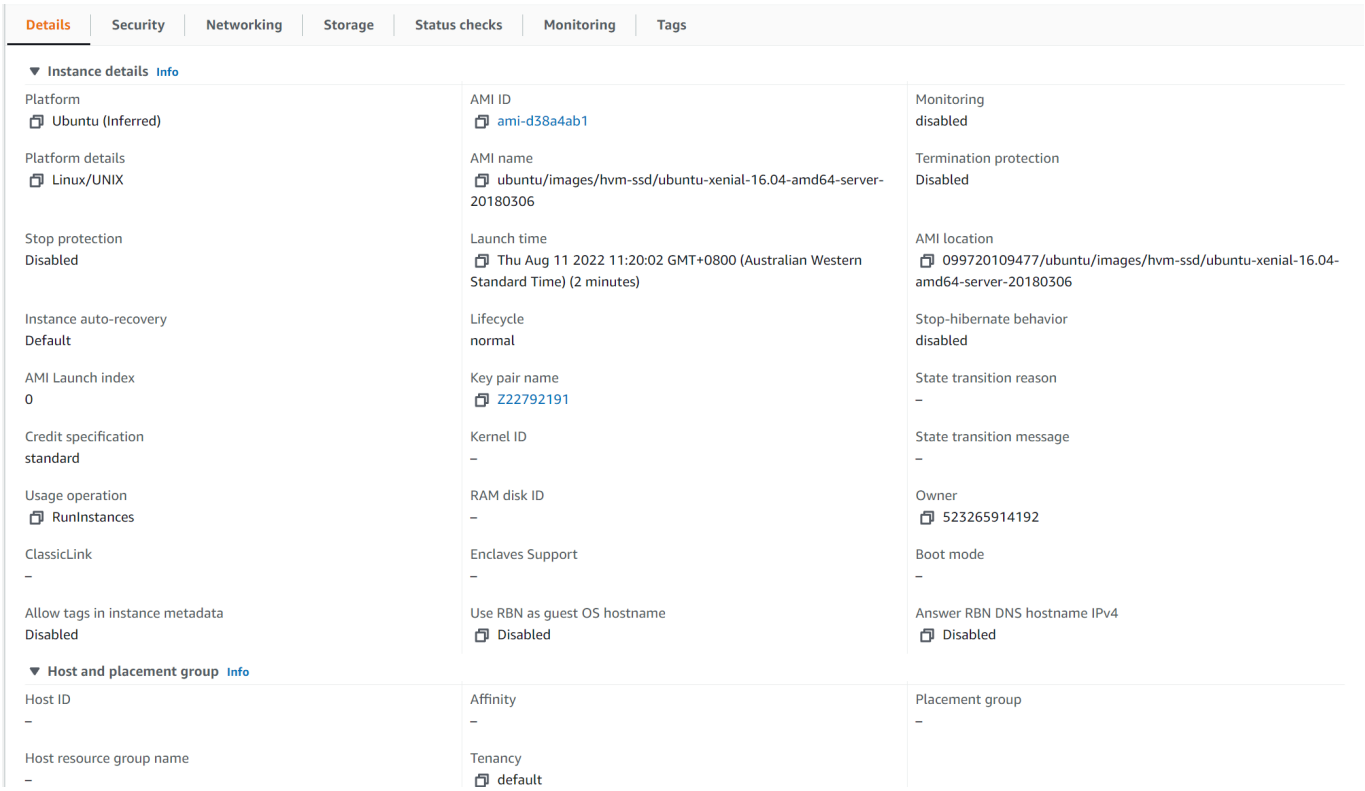
```python
import boto3

AWS_REGION = "ap-southeast-2"
AMI_ID ="ami-d38a4ab1"
EC2_RESOURCE = boto3.resource('ec2', region_name=AWS_REGION)
KEY_PAIR_NAME = "Z22792191"
SECURITY_GROUP_ID = 'sg-02e03d15c337cfec2'

EC2_CLIENT = boto3.client('ec2', region_name=AWS_REGION)

instances = EC2_RESOURCE.create_instances(
    MinCount = 1,
    MaxCount = 1,
    ImageId=AMI_ID,
    InstanceType='t2.micro',
    KeyName=KEY_PAIR_NAME,
    SecurityGroupIds=[
        SECURITY_GROUP_ID,
    ],
    TagSpecifications=[
        {
            'ResourceType': 'instance',
            'Tags': [
                {
                    'Key': 'Name',
                    'Value': '22792191'
                },
            ]
        },
    ]
)
for instance in instances:
    print(instance.id)
```

output:

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ python3 create_instance.py
i-02aff76e37b24898c
```

AWS console:





Step 5 – Return the public IP of the instance created from previous steps via describe_instance function

python code:

create_SG.py | ssh.py | key_pair.py | create_instance.py | get_ip.py

2022s2 > cits5503 > labs > lab2 > get_ip.py > ...

```python
EC2_RESOURCE = boto3.resource('ec2', region_name=AWS_REGION)
KEY_PAIR_NAME = "Z22792191"
SECURITY_GROUP_ID = 'sg-02e03d15c337cfec2'
INSTANCE_ID = 'i-02aff76e37b24898c'

EC2_CLIENT = boto3.client('ec2', region_name=AWS_REGION)

response = EC2_CLIENT.describe_instances(
    InstanceIds=[
        INSTANCE_ID
    ],
)
print(response['Reservations'][0]['Instances'][0]['PublicIpAddress'])

```

output:

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ python3 get_ip.py
54.252.224.11
```

Using Docker

[1] Install Docker

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503$ sudo apt install docker.io -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
docker.io is already the newest version (20.10.12-0ubuntu2~20.04.1).
0 upgraded, 0 newly installed, 0 to remove and 105 not upgraded.
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503$ sudo systemctl enable docker
```

[2] Check the version

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503$ docker --version
Docker version 20.10.12, build 20.10.12-0ubuntu2~20.04.1
```

[3] Build and run an httpd container

we make a directory named `html`, then create `index.html` using `touch` command, then edit `index.html` using `nano` command, then we print the content of `index.html` using `cat` command.

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ mkdir html
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ touch index.html
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ sudo nano index.html
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ cat index.html
  <html>
    <head> </head>
    <body>
      <p>Hello World!</p>
    </body>
  </html>
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ mv index.html html
```

[4] Create a file called “Dockerfile” outside the html directory with the following content:

We use `touch` command to create `Dockerfile`, then use `nano` to edit it, then we print the content of `Dockerfile` using `cat` command.

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ touch Dockerfile
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ sudo nano Dockerfile
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ cat Dockerfile
FROM httpd:2.4
COPY ./html/ /usr/local/apache2/htdocs/
```

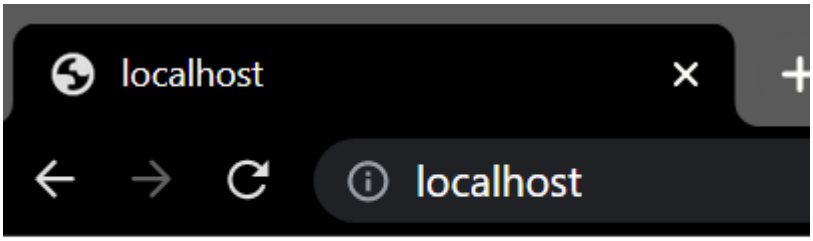
[5] Build the docker image

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ docker build -t my-apache2 .
[+] Building 43.1s (7/7) FINISHED
 => [internal] load build definition from Dockerfile                                0.1s
 => => transferring dockerfile: 98B                                                 0.0s
 => [internal] load .dockerignore                                                   0.0s
 => => transferring context: 2B                                                     0.0s
 => [internal] load metadata for docker.io/library/httpd:2.4                       35.5s
 => [internal] load build context                                                   0.0s
 => => transferring context: 157B                                                   0.0s
 => [1/2] FROM docker.io/library/httpd:2.4@sha256:343452ec820a5d59eb3ab9aaa6201d193f91c3  7.2s
 => => resolve docker.io/library/httpd:2.4@sha256:343452ec820a5d59eb3ab9aaa6201d193f91c3  0.0s
 => => sha256:aed046121ed887f6b4f25442af6223f85c6981d82ff42f394912d55c7800a2 176B / 176B  1.3s
 => => sha256:4340e7be3d7f882f8d9671282629c3590e360ae22f6131bdc165de464e 1.72MB / 1.72MB  0.6s
 => => sha256:343452ec820a5d59eb3ab9aaa6201d193f91c3354f8c4f29705796d935 1.86kB / 1.86kB  0.0s
 => => sha256:98778663b10c3952e9d7dd8a10e1ca2a8ce31f11b5f0ff9d7b3b36ddb8 1.37kB / 1.37kB  0.0s
 => => sha256:f2a976f932ec6fe48978c1cdde2c8217a497b1f080c80e49049e027573 9.04kB / 9.04kB  0.0s
 => => sha256:1efc276f4ff952c055dea726cfc96ec6a4fdb8b62d9eed816bd2b788 31.37MB / 31.37MB  4.3s
 => => sha256:80e368ef21fc51da790af6986dff8b3cce3a477e65ce7a5a6ae7559a 23.97MB / 23.97MB  6.3s
 => => sha256:80cb79a80bbe84544d9f4c2a5018ae5cf2373c44ac2983f1f1fa18b12d2cb9 298B / 298B  1.8s
 => => extracting sha256:1efc276f4ff952c055dea726cfc96ec6a4fdb8b62d9eed816bd2b788f2860ad  1.4s
 => => extracting sha256:aed046121ed887f6b4f25442af6223f85c6981d82ff42f394912d55c7800a2b  0.0s
 => => extracting sha256:4340e7be3d7f882f8d9671282629c3590e360ae22f6131bdc165de464ee8042  0.1s
 => => extracting sha256:80e368ef21fc51da790af6986dff8b3cce3a477e65ce7a5a6ae7559a7684fbc  0.6s
 => => extracting sha256:80cb79a80bbe84544d9f4c2a5018ae5cf2373c44ac2983f1f1fa18b12d2cb99  0.0s
 => [2/2] COPY ./html/ /usr/local/apache2/htdocs/                                   0.2s
 => exporting to image                                                              0.0s
 => => exporting layers                                                             0.0s
 => => writing image sha256:0501e8ed9d1a3b5c2ef18544e2020a90a3b4c0031b0480bd35d7e981456d  0.0s
 => => naming to docker.io/library/my-apache2                                       0.0s
```

[6] Run the image

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ docker run -p 80:80 -dit --name my-app my-a
pache2
363e6fb57dca14ed15e29c2e703ba365c3743503b7381f1fb658f47ce0c68947
```

[7] Open a browser and access address http://localhost Confirm you get Hello World!



Hello World!

[8] Other commands

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ docker ps -a
CONTAINER ID   IMAGE        COMMAND              CREATED              STATUS              PORTS                NAMES
363e6fb57dca   my-apache2   "httpd-foreground"   About a minute ago   Up About a minute   0.0.0.0:80->80/tcp   my-app
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ docker stop my-app
my-app
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab2$ docker rm my-app
my-app
```