John Do
CSC 452-001
Dr. Droz
November 1, 2019

Lab 4: Assignment

THE BLACK TEXTBOX ARE THE OUTPUTS FROM THE PROGRAMS. IT WAS TOO LONG TO SCREENSHOT SO I COPY AND PASTE IT ON HERE.

1. Make a new environment on AWS Cloud 9. Write and execute **Python Boto3** programs that do the following using **Amazon Elastic Cloud (EC2).**

a.

i. Get basic information about your Amazon EC2 Instances

ii.
```
{'Reservations': [{'Groups': [], 'Instances': [{'AmiLaunchIndex':
0, 'ImageId': 'ami-045e373af522ed8ef', 'InstanceId': 'i-
0ae020bd6b53616a3', 'InstanceType': 't2.micro', 'LaunchTime': date
time.datetime(2019, 10, 24, 19, 15, 22, tzinfo=tzlocal()), 'Monito
ring': {'State': 'disabled'}, 'Placement': {'AvailabilityZone': 'u
s-east-
2c', 'GroupName': '', 'Tenancy': 'default'}, 'PrivateDnsName': 'ip
-172-31-41-202.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.41.202', 'Product
Codes': [], 'PublicDnsName': 'ec2-18-222-170-19.us-east-
2.compute.amazonaws.com', 'PublicIpAddress': '18.222.170.19', 'Sta
te': {'Code': 16, 'Name': 'running'}, 'StateTransitionReason': '',
 'SubnetId': 'subnet-7b942d37', 'VpcId': 'vpc-
e671818d', 'Architecture': 'x86_64', 'BlockDeviceMappings': [{'Dev
iceName': '/dev/xvda', 'Ebs': {'AttachTime': datetime.datetime(201
9, 10, 24, 19, 15, 23, tzinfo=tzlocal()), 'DeleteOnTermination': T
rue, 'Status': 'attached', 'VolumeId': 'vol-
0f385f6a3ca5480bc'}}], 'ClientToken': 'aws-c-Insta-
TLC11LAEICI1', 'EbsOptimized': False, 'EnaSupport': True, 'Hypervi
sor': 'xen', 'NetworkInterfaces': [{'Association': {'IpOwnerId': '
amazon', 'PublicDnsName': 'ec2-18-222-170-19.us-east-
2.compute.amazonaws.com', 'PublicIp': '18.222.170.19'}, 'Attachmen
t': {'AttachTime': datetime.datetime(2019, 10, 24, 19, 15, 22, tzi
nfo=tzlocal()), 'AttachmentId': 'eni-attach-
09f8f35700603144d', 'DeleteOnTermination': True, 'DeviceIndex': 0,
 'Status': 'attached'}, 'Description': '', 'Groups': [{'GroupName'
: 'aws-cloud9-PythonAWSLab4-96d7c723888c4216803f9843a61de546-
InstanceSecurityGroup-EAM80YJFI9PV', 'GroupId': 'sg-
0fff90f241d382d47'}], 'Ipv6Addresses': [], 'MacAddress': '0a:69:de
:59:4c:90', 'NetworkInterfaceId': 'eni-
05e652bc91fdbfb47', 'OwnerId': '801211634689', 'PrivateDnsName': '
ip-172-31-41-202.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.41.202', 'Private
IpAddresses': [{'Association': {'IpOwnerId': 'amazon', 'PublicDnsN
ame': 'ec2-18-222-170-19.us-east-
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019


```
2.compute.amazonaws.com', 'PublicIp': '18.222.170.19'}, 'Primary':
 True, 'PrivateDnsName': 'ip-172-31-41-202.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.41.202'}], 'Sourc
eDestCheck': True, 'Status': 'in-use', 'SubnetId': 'subnet-
7b942d37', 'VpcId': 'vpc-
e671818d', 'InterfaceType': 'interface'}], 'RootDeviceName': '/dev
/xvda', 'RootDeviceType': 'ebs', 'SecurityGroups': [{'GroupName':
'aws-cloud9-PythonAWSLab4-96d7c723888c4216803f9843a61de546-
InstanceSecurityGroup-EAM80YJFI9PV', 'GroupId': 'sg-
0fff90f241d382d47'}], 'SourceDestCheck': True, 'Tags': [{'Key': 'a
ws:cloudformation:stack-id', 'Value': 'arn:aws:cloudformation:us-
east-2:801211634689:stack/aws-cloud9-PythonAWSLab4-
96d7c723888c4216803f9843a61de546/9a7619c0-f692-11e9-ab11-
0ade6091f9d8'}, {'Key': 'Name', 'Value': 'aws-cloud9-
PythonAWSLab4-
96d7c723888c4216803f9843a61de546'}, {'Key': 'aws:cloud9:environmen
t', 'Value': '96d7c723888c4216803f9843a61de546'}, {'Key': 'aws:clo
ud9:owner', 'Value': '801211634689'}, {'Key': 'aws:cloudformation:
logical-
id', 'Value': 'Instance'}, {'Key': 'aws:cloudformation:stack-
name', 'Value': 'aws-cloud9-PythonAWSLab4-
96d7c723888c4216803f9843a61de546'}], 'VirtualizationType': 'hvm',
'CpuOptions': {'CoreCount': 1, 'ThreadsPerCore': 1}, 'CapacityRese
rvationSpecification': {'CapacityReservationPreference': 'open'},
'HibernationOptions': {'Configured': False}}], 'OwnerId': '8012116
34689', 'RequesterId': '043320173835', 'ReservationId': 'r-
05803fe9f22f6ad1b'}, {'Groups': [], 'Instances': [{'AmiLaunchIndex
': 0, 'ImageId': 'ami-09ee46c8d9abcb966', 'InstanceId': 'i-
0f407d811567d595e', 'InstanceType': 't2.micro', 'KeyName': 'John_L
ab3key', 'LaunchTime': datetime.datetime(2019, 10, 23, 22, 45, 49,
 tzinfo=tzlocal()), 'Monitoring': {'State': 'disabled'}, 'Placemen
t': {'AvailabilityZone': 'us-east-
2c', 'GroupName': '', 'Tenancy': 'default'}, 'Platform': 'windows'
, 'PrivateDnsName': 'ip-172-31-38-45.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.38.45', 'ProductC
odes': [], 'PublicDnsName': '', 'State': {'Code': 80, 'Name': 'sto
pped'}, 'StateTransitionReason': 'User initiated (2019-10-
24 02:57:41 GMT)', 'SubnetId': 'subnet-7b942d37', 'VpcId': 'vpc-
e671818d', 'Architecture': 'x86_64', 'BlockDeviceMappings': [{'Dev
iceName': '/dev/sda1', 'Ebs': {'AttachTime': datetime.datetime(201
9, 10, 17, 19, 56, 43, tzinfo=tzlocal()), 'DeleteOnTermination': T
rue, 'Status': 'attached', 'VolumeId': 'vol-
0bb1943a27830f761'}}], 'ClientToken': '', 'EbsOptimized': False, '
EnaSupport': True, 'Hypervisor': 'xen', 'NetworkInterfaces': [{'At
tachment': {'AttachTime': datetime.datetime(2019, 10, 17, 19, 56,
42, tzinfo=tzlocal()), 'AttachmentId': 'eni-attach-
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019


```
07dd2756b17b70d09', 'DeleteOnTermination': True, 'DeviceIndex': 0,
 'Status': 'attached'}, 'Description': '', 'Groups': [{'GroupName'
: 'launch-wizard-1', 'GroupId': 'sg-
072cf6562ea4b3c77'}], 'Ipv6Addresses': [], 'MacAddress': '0a:5d:e2
:0a:35:94', 'NetworkInterfaceId': 'eni-
0506a7da1425ace93', 'OwnerId': '801211634689', 'PrivateDnsName': '
ip-172-31-38-45.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.38.45', 'PrivateI
pAddresses': [{'Primary': True, 'PrivateDnsName': 'ip-172-31-38-
45.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.38.45'}], 'Source
DestCheck': True, 'Status': 'in-use', 'SubnetId': 'subnet-
7b942d37', 'VpcId': 'vpc-
e671818d', 'InterfaceType': 'interface'}], 'RootDeviceName': '/dev
/sda1', 'RootDeviceType': 'ebs', 'SecurityGroups': [{'GroupName':
'launch-wizard-1', 'GroupId': 'sg-
072cf6562ea4b3c77'}], 'SourceDestCheck': True, 'StateReason': {'Co
de': 'Client.UserInitiatedShutdown', 'Message': 'Client.UserInitia
tedShutdown: User initiated shutdown'}, 'VirtualizationType': 'hvm
', 'CpuOptions': {'CoreCount': 1, 'ThreadsPerCore': 1}, 'CapacityR
eservationSpecification': {'CapacityReservationPreference': 'open'
}, 'HibernationOptions': {'Configured': False}}], 'OwnerId': '8012
11634689', 'ReservationId': 'r-
06098e80830a11a7d'}, {'Groups': [], 'Instances': [{'AmiLaunchIndex
': 1, 'ImageId': 'ami-0d5d9d301c853a04a', 'InstanceId': 'i-
00d4bfd0ee3814815', 'InstanceType': 't2.micro', 'KeyName': 'John_U
nix2', 'LaunchTime': datetime.datetime(2019, 10, 23, 22, 45, 49, t
zinfo=tzlocal()), 'Monitoring': {'State': 'disabled'}, 'Placement'
: {'AvailabilityZone': 'us-east-
2c', 'GroupName': '', 'Tenancy': 'default'}, 'PrivateDnsName': 'ip
-172-31-39-155.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.39.155', 'Product
Codes': [], 'PublicDnsName': '', 'State': {'Code': 80, 'Name': 'st
opped'}, 'StateTransitionReason': 'User initiated (2019-10-
24 02:57:41 GMT)', 'SubnetId': 'subnet-7b942d37', 'VpcId': 'vpc-
e671818d', 'Architecture': 'x86_64', 'BlockDeviceMappings': [{'Dev
iceName': '/dev/sda1', 'Ebs': {'AttachTime': datetime.datetime(201
9, 10, 22, 19, 28, 5, tzinfo=tzlocal()), 'DeleteOnTermination': Tr
ue, 'Status': 'attached', 'VolumeId': 'vol-
0e48f1f93c8deba35'}}], 'ClientToken': '', 'EbsOptimized': False, '
EnaSupport': True, 'Hypervisor': 'xen', 'NetworkInterfaces': [{'At
tachment': {'AttachTime': datetime.datetime(2019, 10, 22, 19, 28,
4, tzinfo=tzlocal()), 'AttachmentId': 'eni-attach-
0654dfb838deefa0a', 'DeleteOnTermination': True, 'DeviceIndex': 0,
 'Status': 'attached'}, 'Description': '', 'Groups': [{'GroupName'
: 'launch-wizard-3', 'GroupId': 'sg-
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019

```
085a7e281dc20437d'}], 'Ipv6Addresses': [], 'MacAddress': '0a:14:7c
:64:5a:c2', 'NetworkInterfaceId': 'eni-
003854a15b290e5f0', 'OwnerId': '801211634689', 'PrivateDnsName': '
ip-172-31-39-155.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.39.155', 'Private
IpAddresses': [{'Primary': True, 'PrivateDnsName': 'ip-172-31-39-
155.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.39.155'}], 'Sourc
eDestCheck': True, 'Status': 'in-use', 'SubnetId': 'subnet-
7b942d37', 'VpcId': 'vpc-
e671818d', 'InterfaceType': 'interface'}], 'RootDeviceName': '/dev
/sda1', 'RootDeviceType': 'ebs', 'SecurityGroups': [{'GroupName':
'launch-wizard-3', 'GroupId': 'sg-
085a7e281dc20437d'}], 'SourceDestCheck': True, 'StateReason': {'Co
de': 'Client.UserInitiatedShutdown', 'Message': 'Client.UserInitia
tedShutdown: User initiated shutdown'}, 'VirtualizationType': 'hvm
', 'CpuOptions': {'CoreCount': 1, 'ThreadsPerCore': 1}, 'CapacityR
eservationSpecification': {'CapacityReservationPreference': 'open'
}, 'HibernationOptions': {'Configured': False}}, {'AmiLaunchIndex'
: 0, 'ImageId': 'ami-0d5d9d301c853a04a', 'InstanceId': 'i-
0f38542e6a1741748', 'InstanceType': 't2.micro', 'KeyName': 'John_U
nix2', 'LaunchTime': datetime.datetime(2019, 10, 23, 22, 45, 49, t
zinfo=tzlocal()), 'Monitoring': {'State': 'disabled'}, 'Placement'
: {'AvailabilityZone': 'us-east-
2c', 'GroupName': '', 'Tenancy': 'default'}, 'PrivateDnsName': 'ip
-172-31-45-90.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.45.90', 'ProductC
odes': [], 'PublicDnsName': '', 'State': {'Code': 80, 'Name': 'sto
pped'}, 'StateTransitionReason': 'User initiated (2019-10-
24 02:57:41 GMT)', 'SubnetId': 'subnet-7b942d37', 'VpcId': 'vpc-
e671818d', 'Architecture': 'x86_64', 'BlockDeviceMappings': [{'Dev
iceName': '/dev/sda1', 'Ebs': {'AttachTime': datetime.datetime(201
9, 10, 22, 19, 28, 5, tzinfo=tzlocal()), 'DeleteOnTermination': Tr
ue, 'Status': 'attached', 'VolumeId': 'vol-
0b05d9fcdfa7616e0'}}], 'ClientToken': '', 'EbsOptimized': False, '
EnaSupport': True, 'Hypervisor': 'xen', 'NetworkInterfaces': [{'At
tachment': {'AttachTime': datetime.datetime(2019, 10, 22, 19, 28,
4, tzinfo=tzlocal()), 'AttachmentId': 'eni-attach-
02610b799914e70bb', 'DeleteOnTermination': True, 'DeviceIndex': 0,
 'Status': 'attached'}, 'Description': '', 'Groups': [{'GroupName'
: 'launch-wizard-3', 'GroupId': 'sg-
085a7e281dc20437d'}], 'Ipv6Addresses': [], 'MacAddress': '0a:f1:30
:86:75:e4', 'NetworkInterfaceId': 'eni-
07358425607d2cb95', 'OwnerId': '801211634689', 'PrivateDnsName': '
ip-172-31-45-90.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.45.90', 'PrivateI
```

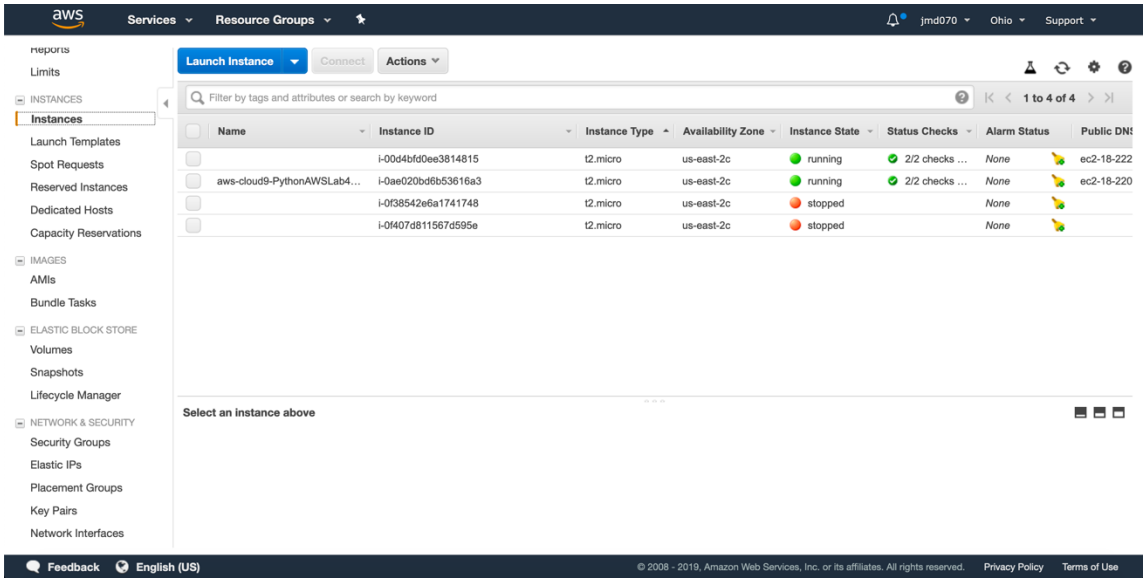
John Do
CSC 452-001
Dr. Droz
November 1, 2019


```
pAddresses': [{'Primary': True, 'PrivateDnsName': 'ip-172-31-45-
90.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.45.90'}], 'Source
DestCheck': True, 'Status': 'in-use', ': [{'GroupName': 'launch-
wizard-3', 'GroupId': 'sg-
085a7e281dc20437d'}], 'SourceDestCheck': True, 'StateReason': {'Co
de': 'Client.UserInitiatedShutdown', 'Message': 'Client.UserInitia
tedShutdown: User initiated shutdown'}, 'VirtualizationType': 'hvm
', 'CpuOptions': {'CoreCount': 1, 'ThreadsPerCore': 1}, 'CapacityR
eservationSpecification': {'CapacityReservationPreference': 'open'
}, 'HibernationOptions': {'Configured': False}}], 'OwnerId': '8012
11634689', 'ReservationId': 'r-
0c1564ee8f7c59929'}], 'ResponseMetadata': {'RequestId': 'cf997cc9-
3ef3-4b2c-b50d-
81445ad2312b', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'transfer-
encoding': 'chunked', 'vary': 'accept-
encoding', 'date': 'Thu, 24 Oct 2019 19:37:35 GMT', 'server': 'Ama
zonEC2'}, 'RetryAttempts': 0}}
```

```
Process exited with code: 0
```

```
Pane is dead
```


iii.



b.

i. Start and Stop Detailed Monitoring of an EC2 instance

ii.


```
{'InstanceMonitorings': [{'InstanceId': 'i-
0ae020bd6b53616a3', 'Monitoring': {'State': 'enabled'}}], 'Respons
eMetadata': {'RequestId': 'e38bd77a-e437-40dc-af9b-
726fec2ad400', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
```

```
type': 'text/xml;charset=UTF-8', 'content-
length': '424', 'date': 'Thu, 24 Oct 2019 19:47:11 GMT', 'server':
 'AmazonEC2'}, 'RetryAttempts': 0}}
```

c. OFF:

i. ii.
```
{'StoppingInstances': [{'CurrentState': {'Code': 64, 'Name': 'stop
ping'}, 'InstanceId': 'i-
00d4bfd0ee3814815', 'PreviousState': {'Code': 16, 'Name': 'running
'}}], 'ResponseMetadata': {'RequestId': 'cf9c91d5-9d50-44b8-b253-
d461e692e3c8', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '579', 'date': 'Thu, 24 Oct 2019 20:01:15 GMT', 'server':
 'AmazonEC2'}, 'RetryAttempts': 0}}
```

```
ON:
       {'StartingInstances': [{'CurrentState': {'Code': 0, 'Name': 'pend
ing'}, 'InstanceId': 'i-
00d4bfd0ee3814815', 'PreviousState': {'Code': 80, 'Name': 'stopped'}}], '
ResponseMetadata': {'RequestId': '3c49ca16-8d39-47cf-a5f1-
0ba6affa624c', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '579', 'date': 'Fri, 25 Oct 2019 21:19:22 GMT', 'server': 'Amazo
nEC2'}, 'RetryAttempts': 0}}
```

d.

i. This is rebooting the instance

```
success {'ResponseMetadata': {'RequestId': '6fc30fc1-56e5-466e-9558-
535150dc415b', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '231', 'date': 'Wed, 30 Oct 2019 15:29:47 GMT', 'server': '
AmazonEC2'}, 'RetryAttempts': 0}}
```

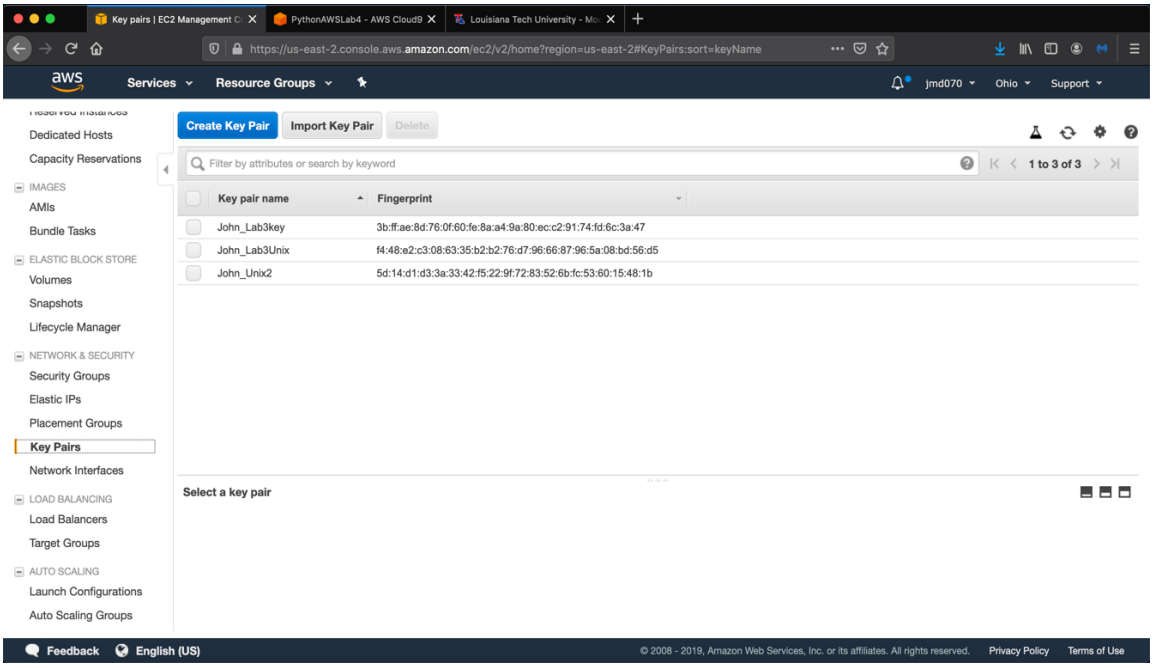
e.

i. Get info about the key pairs

ii.
```
success {'KeyPairs': [{'KeyFingerprint': '3b:ff:ae:8d:76:0f:60:fe:
8a:a4:9a:80:ec:c2:91:74:fd:6c:3a:47', 'KeyName': 'John_Lab3key'},
{'KeyFingerprint': 'f4:48:e2:c3:08:63:35:b2:b2:76:d7:96:66:87:96:5
a:08:bd:56:d5', 'KeyName': 'John_Lab3Unix'}, {'KeyFingerprint': '5
d:14:d1:d3:3a:33:42:f5:22:9f:72:83:52:6b:fc:53:60:15:48:1b', 'KeyN
ame': 'John_Unix2'}], 'ResponseMetadata': {'RequestId': 'a4fb448e-
00a1-4898-9b52-
c86bbbfec5e9', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '773', 'date': 'Wed, 30 Oct 2019 15:34:00 GMT', 'server':
 'AmazonEC2'}, 'RetryAttempts': 0}}
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019

iii.



f.

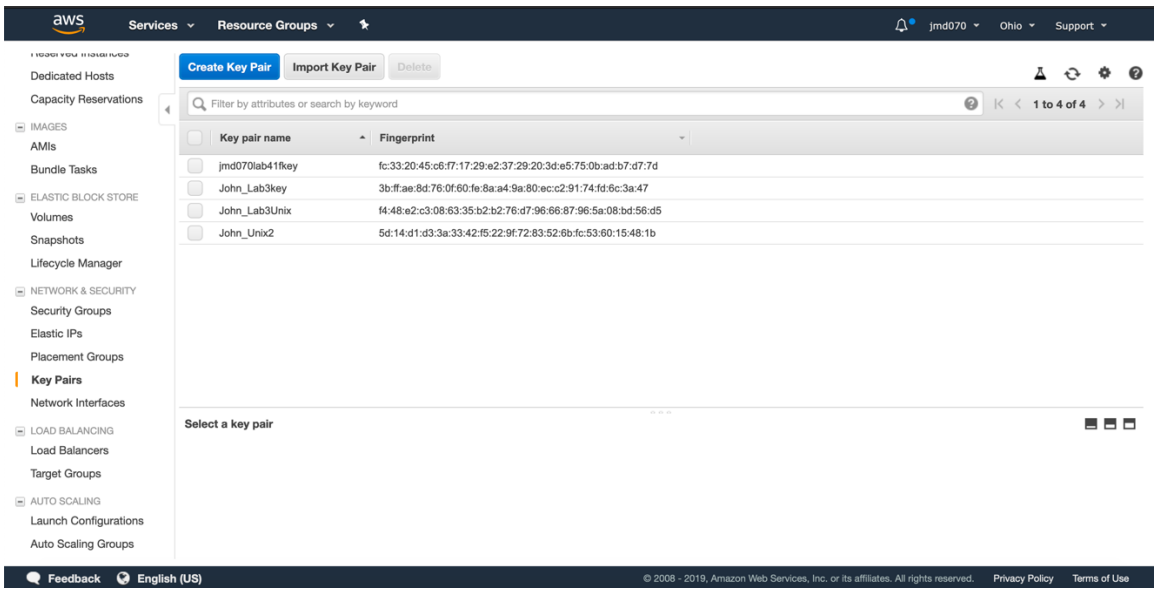
i. Create a new key pair

ii.

```
success {'KeyFingerprint': 'fc:33:20:45:c6:f7:17:29:e2:37:29:20:3d
:e5:75:0b:ad:b7:d7:7d', 'KeyMaterial': '-----
BEGIN RSA PRIVATE KEY-----
\nMIIEogIBAAKCAQEAs1Ry7qJ8RG6witc7MRajslN7TBC/DUphP29vIBLA7rvlgz7Z
1l3ycPwqNJRR\nOSWNgmoyISG0N6tAgWI+bJkZM2ejIF/d0UKRh5cxa53t1fqkHdMj
20pIhlyGrPhjjkuOzh+Qj44V\nXoLZsUu3riNZLiW7XWrJvX7xXagz0kYWAjf+pPwW
23v12YajCc+nzkwLjiTCAcDHicQ2AXI6rrDi\nIFtWGWCBYZbpLD+8/WAUGbG6hoc9
6sE1AYmpgfEgepuQYNHH5ChSEkdt/afC8o2XLcDy7Y8afxwS\n0Li5MsvWtKvyjBtW
beFx665Ve8yiRKVfJ935ihubcoAN+Ajevfu55QIDAQABAoIBAGt8NXsk9tjJ\ng5Vk
Vf0hNbKf6On5q5mauFkeQHv34Z32DyWwWT7shnEEI4PqVhirknrVjgBfxDYNthcWvJ
W42er+\ntj6kVz6i6LldfOik+zXjCdU6hWZdysSF1s77/if+z7RqJOp2XgaM5VAoMw
rz8vL++KOQayTiXmcl\ndxIUogJnbazwi2Dp4bZWFEl76yhxI4JIesioHggdZD/0LZ
3khhW0SKmEAz7ZsbbEastZZ5AK52sB\nO9TnpOMesHYYyb5l3apW+ZNX98iAStUQAk
iMqi+bXTeAPiAAPxYgMP8gln+nPM2oWXMcGrfGqYiZ\nMNQTvbzYqrqv/0bYILXFvA
uOTM0CgYEA6cjfRISqdNZaPmC6TkKYlW4lKTGLeTae/YStpR80d0QB\nQpYfmJYPy/
Why1tm9/d4d5rb9RB+1AcEqEjoNm2a0BrvhpS5zRYI69Dt1B73fJ6Q4UYGX0GEJaMT
\nX+0xGNyYDbfbxK28N5P5aS2zLaoRAKihWi+d8qKbMloExUQ2tCsCgYEAxF7kBcun
8/yIpwID2Nlq\ngkHV5K2zWX5gm+kjhAkRp1hK443pRLG2uYYqFb09UaSfSMlfIf8X
UkcG7u45DrVh6IPYO3gGAnui\nY1pSP9gSCsKPEGQXeO1TbJe0QKySUT2i2C+/gA2z
zfuY8WStN0hB/l2Ds4PlZJ/NI2x+75Tg8i8C\ngYA0lZcKnWCN2S0BPNaLBdW2Pskj
```

```
A97gS9XlJjO8oVE2xlgWNxgyRQaWlNBook191u8E0BNLgwNU\nA6QbihKZM3wP9Q8N
UWSdaahmCl+ER9bY8/2Yfhyuzli/8ndIK1zQv8V/yGwpR/6+7Yy7CKl1yvv6\nDvQE
ChPX4JVR9Qd/q4E2cwKBgBZbz46xX5AyQsTxP8rjBnOKV3ZQKMmZ0Bowc3DHL5FkRj
Kep97A\ndVOuaQBMqbywB8rb0s8mIlul5q4xxaFdGabFzEsJIXYK6bRyRkcqrPM4r9
1n6BxGuWb15vouLALA\nB9ihpyZHaR+c9A8YLXjL62ew8hvwep/4af0Mj29r/ijVAo
GAdFbpJbqILA321fbHNHvNx3gbBN6X\niOS+0/mP4uRWCP4qpHzPgH8syXgxjoFGJW
JvdA//Caekshri8OQZy96dGeAnD3buBB6LMYCRYID1\nRX7unX5ZP1L001R3wenTFn
bDuNxRBeyAQcZFN5IwHkT5wSiIiRf5WjmPtB777XReuiM=\n-----
END RSA PRIVATE KEY-----
', 'KeyName': 'jmd070lab41fkey', 'ResponseMetadata': {'RequestId':
 '6c891874-9014-4d00-8e55-
dea80c757d54', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '2039', 'date': 'Wed, 30 Oct 2019 15:40:03 GMT', 'server'
: 'AmazonEC2'}, 'RetryAttempts': 0}}
```


iii.



g.

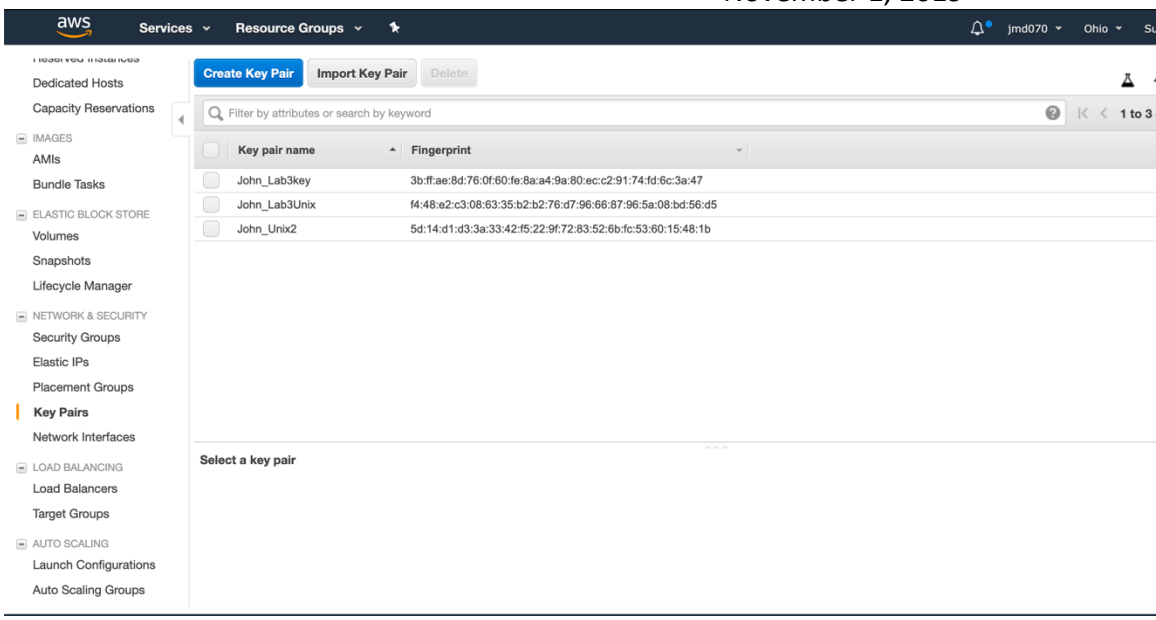
i. Delete a key pair

ii.


```
success {'ResponseMetadata': {'RequestId': '43a7eb29-6b79-4f50-
9ed6-
99f5812655c7', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '227', 'date': 'Wed, 30 Oct 2019 15:44:00 GMT', 'server':
 'AmazonEC2'}, 'RetryAttempts': 0}}
```


iii.

John Do
CSC 452-001
Dr. Droz
November 1, 2019



h.

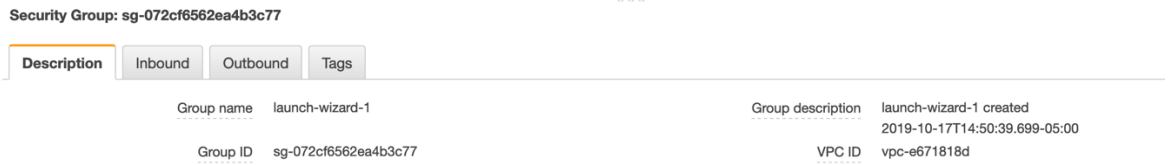
i. Get info about the security group

ii.

```
success {'SecurityGroups': [{'Description': 'launch-wizard-
1 created 2019-10-17T14:50:39.699-05:00', 'GroupName': 'launch-
wizard-
1', 'IpPermissions': [{'FromPort': 80, 'IpProtocol': 'tcp', 'IpRan
ges': [{'CidrIp': '0.0.0.0/0'}], 'Ipv6Ranges': [{'CidrIpv6': '::/0
'}], 'PrefixListIds': [], 'ToPort': 80, 'UserIdGroupPairs': []}, {
'FromPort': 22, 'IpProtocol': 'tcp', 'IpRanges': [{'CidrIp': '0.0.
0.0/0'}], 'Ipv6Ranges': [], 'PrefixListIds': [], 'ToPort': 22, 'Us
erIdGroupPairs': []}, {'FromPort': 3389, 'IpProtocol': 'tcp', 'IpR
anges': [{'CidrIp': '0.0.0.0/0'}], 'Ipv6Ranges': [], 'PrefixListId
s': [], 'ToPort': 3389, 'UserIdGroupPairs': []}], 'OwnerId': '8012
11634689', 'GroupId': 'sg-
072cf6562ea4b3c77', 'IpPermissionsEgress': [{'IpProtocol': '-
1', 'IpRanges': [{'CidrIp': '0.0.0.0/0'}], 'Ipv6Ranges': [], 'Pref
ixListIds': [], 'UserIdGroupPairs': []}], 'VpcId': 'vpc-
e671818d'}], 'ResponseMetadata': {'RequestId': 'e20060e1-1bfe-
4f5c-8d30-
eac9d7cc2616', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '2631', 'vary': 'accept-
encoding', 'date': 'Wed, 30 Oct 2019 15:46:38 GMT', 'server': 'Ama
zonEC2'}, 'RetryAttempts': 0}}
```

iii.

John Do
CSC 452-001
Dr. Droz
November 1, 2019

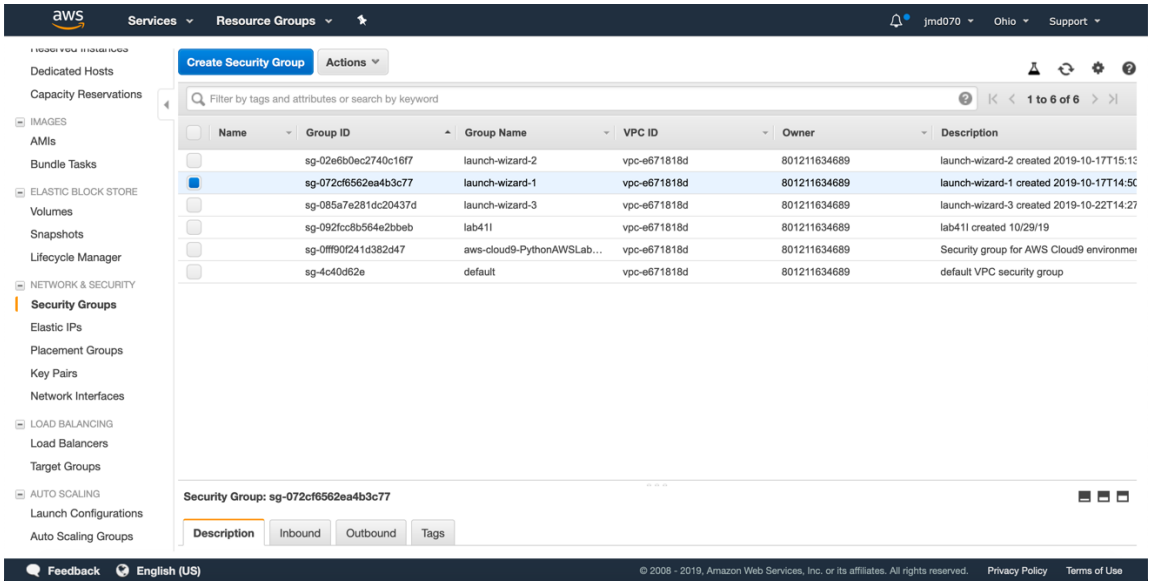


i.

i. Create a security group to access an Amazon EC2 instance

ii.
```
success {'GroupId': 'sg-
092fcc8b564e2bbeb', 'ResponseMetadata': {'RequestId': '327b03cf-
bc3a-4393-b7e8-
b7b9efe94770', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '283', 'date': 'Wed, 30 Oct 2019 15:51:44 GMT', 'server':
 'AmazonEC2'}, 'RetryAttempts': 0}}
```

iii.



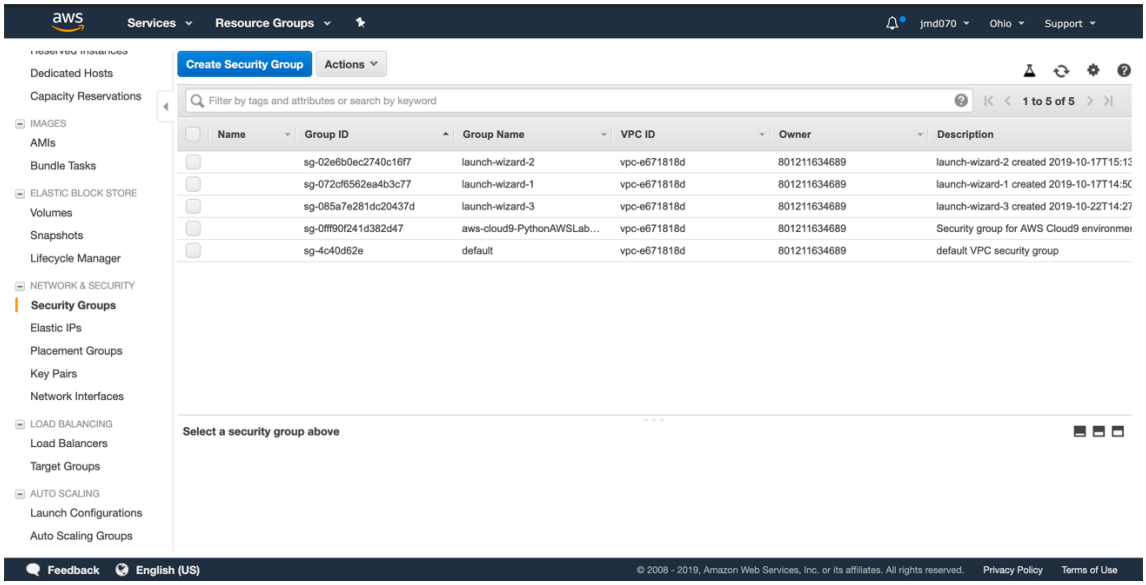
j.

i. Delete a security Group

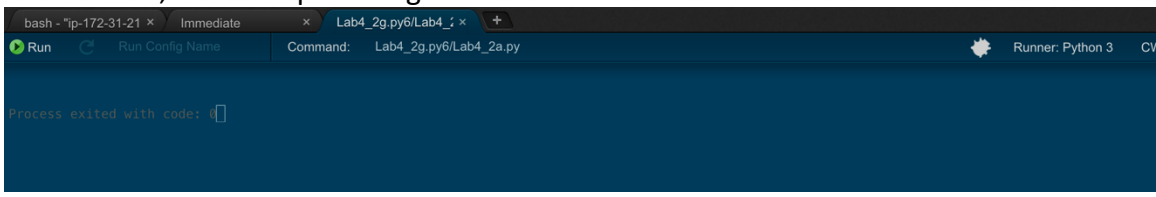
ii.
```
success {'ResponseMetadata': {'RequestId': '9a9bbfc6-5232-44c3-
af8d-
bac74089ccc5', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '239', 'date': 'Wed, 30 Oct 2019 15:56:03 GMT', 'server':
 'AmazonEC2'}, 'RetryAttempts': 0}}
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019

iii.

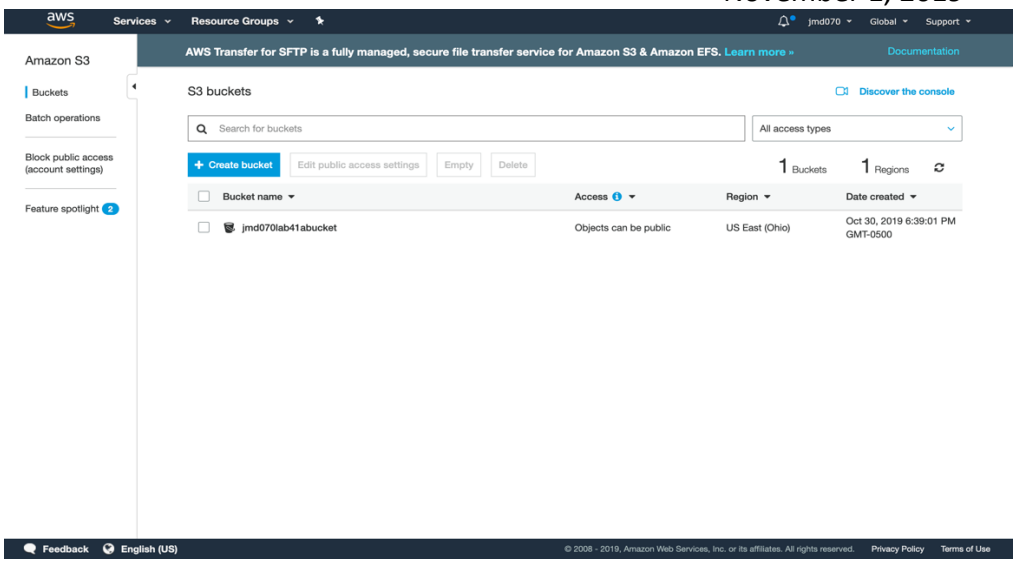


2. Write and Execute Python Boto3 programs that do the following using Amazon S3:
   a.
      i. Create an Amazon S3 Bucket
      ii. Return True, so no output was generated



      iii.

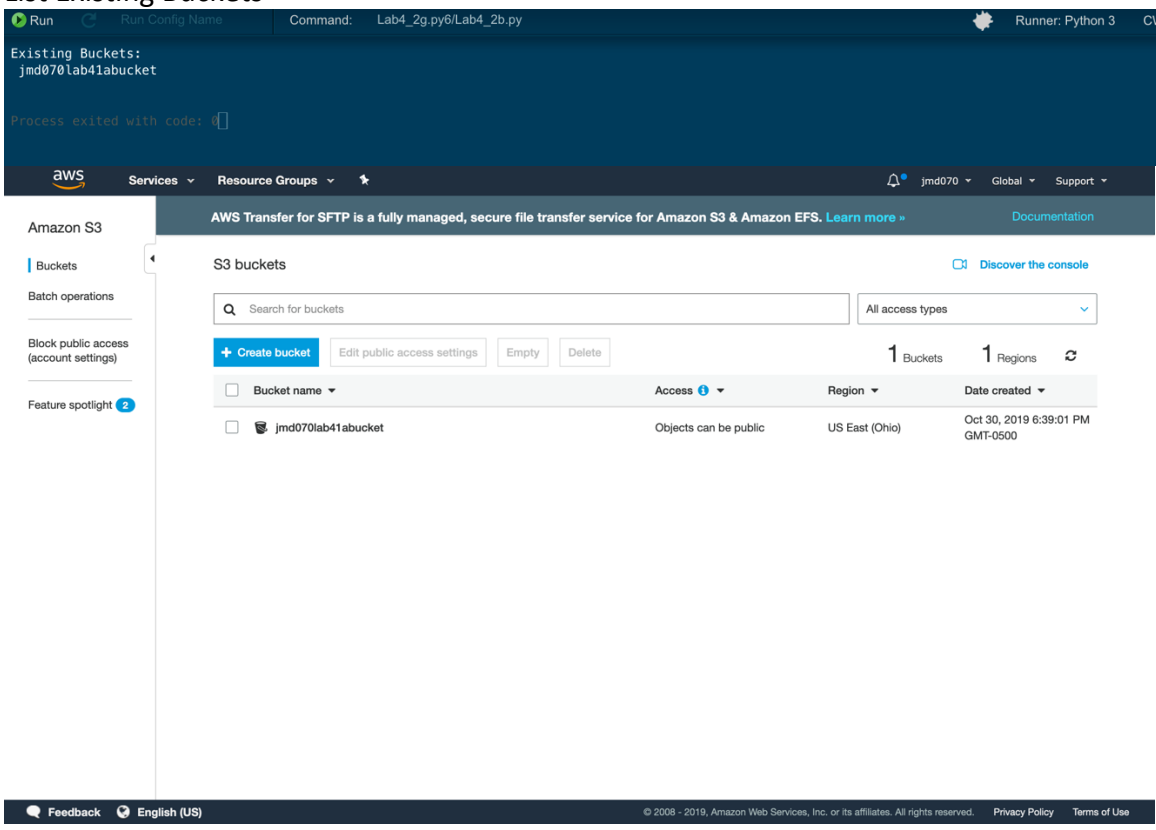
John Do
CSC 452-001
Dr. Droz
November 1, 2019



b.

i. List Existing Buckets

ii.



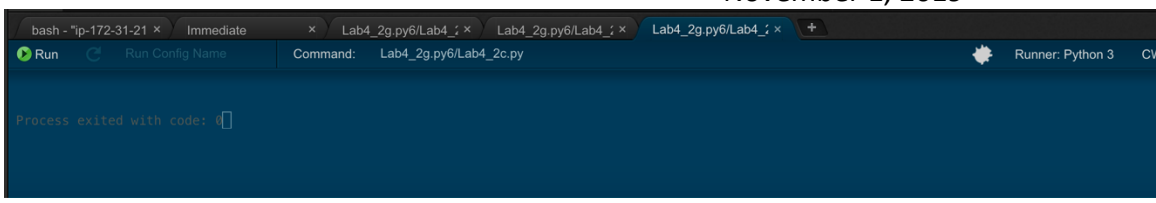
iii.

c.

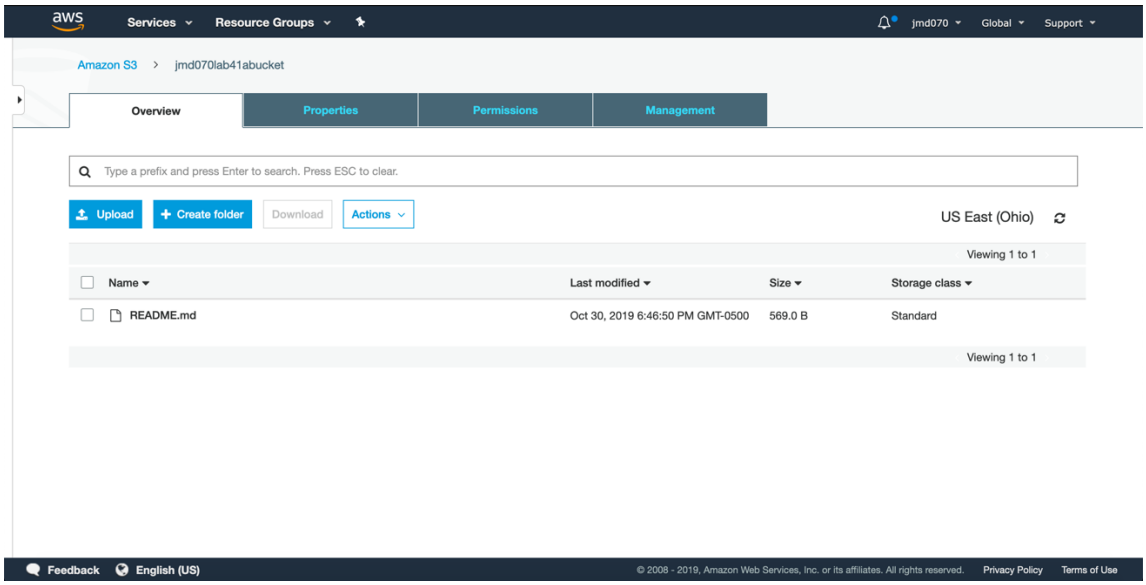
i. Upload Files to an S3 Bucket

John Do
CSC 452-001
Dr. Droz
November 1, 2019

ii.



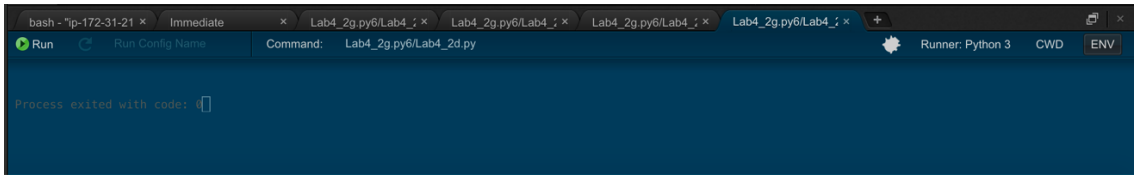
iii.



d.

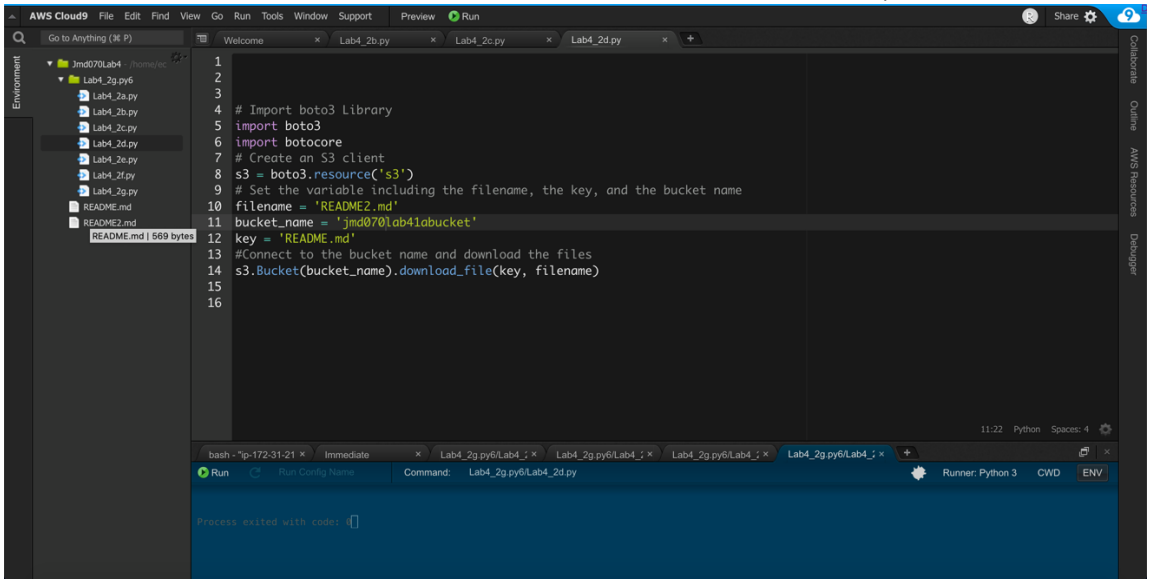
i. Download Files from an S3 Bucket

ii.



John Do
CSC 452-001
Dr. Droz
November 1, 2019

iii.



e.

i. Grant temporary access using a presigned URL

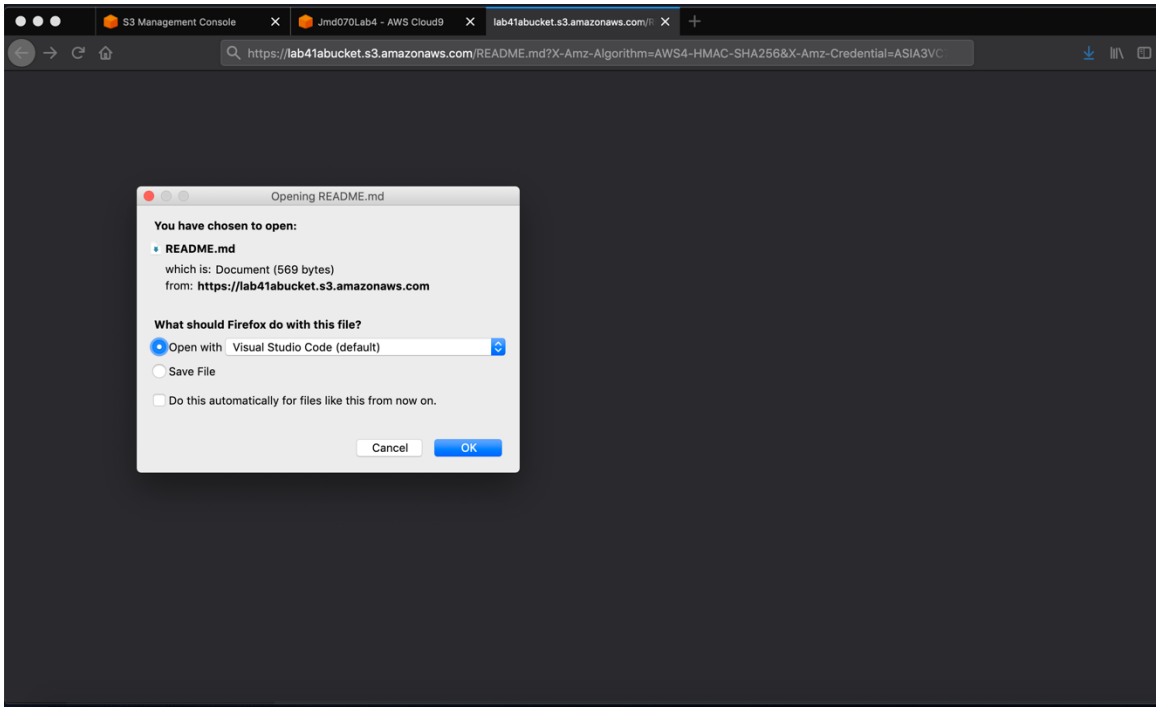
ii. This is the presigned URL:

https://lab41abucket.s3.amazonaws.com/README.md?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=ASIA3VC7PKQA5GW65XZT%2F20191030%2Fus-east-2%2Fs3%2Faws4_request&X-Amz-Date=20191030T235850Z&X-Amz-Expires=3600&X-Amz-SignedHeaders=host&X-Amz-Security-Token=IQoJb3JpZ2luX2VjEMD%2F%2F%2F%2F%2F%2F%2F%2F%2F%2FwEaCXVzLWVhc3QtMiJHMEUCIQCBXionpp8zmoP6drLNZgCUbeKPWRp6FKMJmBrs2%2FZSgQIgDrRoqlPJOgv0wg158vM9FptMBdGm8hqiFdPRrWASwvcqwgIlyf%2F%2F%2F%2F%2F%2F%2F%2F%2F%2F%2FARAAGgw4MDEyMTE2MzQ2ODkiDNchTracsBYj7UXTvCqWAoz2kWGZJjiWPHBZLo7D7D60zaL4pZN1XDJ0JzI7T6Lv%2F9X%2Fo5VULFXsqb%2FjqCJAXShoR9DL6Sl02i2GK%2BfCjvPWHTQ%2BB%2Fqr8lYa7fMXMjV56APMT%2BE%2F9PyGWcbmz21P9J42%2BuEyKy2DYI3lTdjggZsV7rjMP%2B57ijOPyS%2FzL8c1d9aS1ooHm2xJqCqZUfrlNDDJyERvUBHerB5mRgPJuVYgnu6%2FNlORK0qiUS3k8wUduQyXcTEY4f%2Fx24KVI%2BeeybuleXqlrxQuQ6A39P9iSAYuvPYiRc%2BPDcr8vBPD2Og90w9WHoz5s8EAJLj9IoNr%2BoUEtyhYSiBiXkjDMmJfFO%2FGIwVi%2BMKr%2BO0wXfdJ%2BL5Auxwpes0cMNzV5u0FOusCgYJvsTVAX%2FgsJgshMJohUXEAk76Pzzvv2HJTfRqQrba7rA%2BKROEUDqbrleVdKprgLoAONr%2F7larrR1dmfOwGrfALOQSb8xczj0FPVqanYHEw9i7w7KpoAMNJUPXcb1znT6l3l1WXsDM1ZzuH%2FseAWgycDBZh6qwIgx55Gv3FDWTpnrbJHdfeVZhfyFpqzs5OhUDHbWBbh2cnlvmKIprS1%2BTPxgyWUYnGZg461YCXRQkNFKg6slLmLdl1wS6BXvXLtoWeZPlQSByS5aiTEA8RXRDZ7%2FIBbuY0Rs%2B7pZ%2BWcZ3nVhsihOKExifJN3OvO%2BoJI4QuDr3Ej9V2RRm4EIOkUa%2B1CB

John Do
CSC 452-001
Dr. Droz
November 1, 2019

uDVinmpGndeOsfVhd9gcHEBE1OkHprFsPRSCgEZ%2BYLmBP5WzITWe3U56bCAhXia7hE2vgQM0nqKhiXVw%2B4V4D0cdt3DjbHpgERJW3B09PIc7dTd9pdW1%2F36GBh4qfk8D3GsxVu9M86&X-Amz-Signature=b2407d1de47d63af99f67fbfa78f317c7757e8970b632543fdcc88d4364b8bbc

iii.



f.

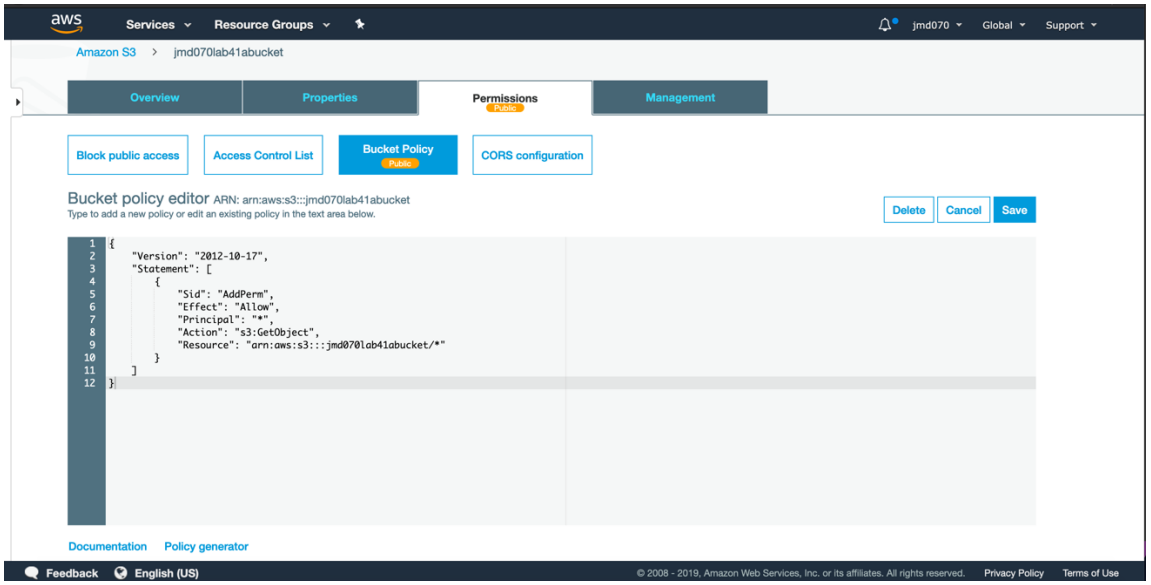
i. Retrieve a bucket policy

ii.

```
{"Version":"2012-10-17","Statement":[{"Sid":"AddPerm","Effect":"Allow","Principal":"*","Action":"s3:GetObject","Resource":"arn:aws:s3:::jmd070lab41abucket/*"}]}
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019

iii.



g.

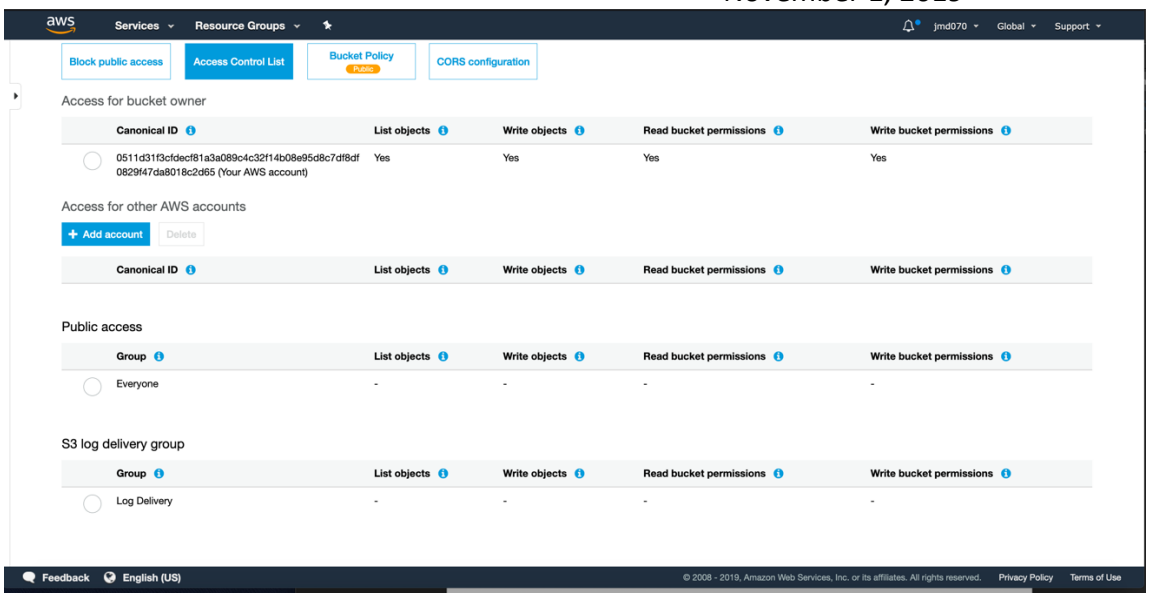
i. Get a Bucket Access Control List

ii.

```
{'ResponseMetadata': {'RequestId': '28565A3384081CD0', 'HostId': '
/Vd6/Y+OmHEBOQRFk3cVI6eODg0m+H5utMFIAMqH03A+Z1LOFN+o+5GajYW/qpU8/9
mx8u9dyg0=', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amz-id-
2': '/Vd6/Y+OmHEBOQRFk3cVI6eODg0m+H5utMFIAMqH03A+Z1LOFN+o+5GajYW/q
pU8/9mx8u9dyg0=', 'x-amz-request-
id': '28565A3384081CD0', 'date': 'Thu, 31 Oct 2019 00:09:14 GMT',
'content-type': 'application/xml', 'transfer-
encoding': 'chunked', 'server': 'AmazonS3'}, 'RetryAttempts': 0},
'Owner': {'ID': '0511d31f3cfdecf81a3a089c4c32f14b08e95d8c7df8df082
9f47da8018c2d65'}, 'Grants': [{'Grantee': {'ID': '0511d31f3cfdecf8
1a3a089c4c32f14b08e95d8c7df8df0829f47da8018c2d65', 'Type': 'Canoni
calUser'}, 'Permission': 'FULL_CONTROL'}]}
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019

iii.

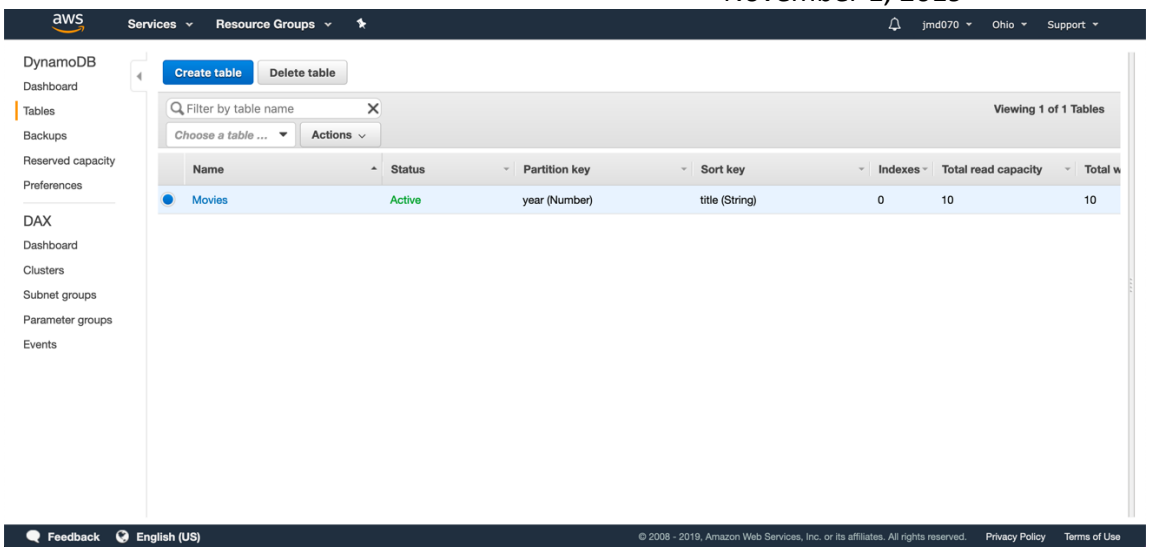


3. (**Extra Credit**, max +2 points on your final grade) Write and execute **Python Boto3** programs that do the following using **Amazon DynamoDB**:
   a.
      i. Create a new table

ii.
```
{'TableDescription': {'AttributeDefinitions': [{'AttributeName': '
title', 'AttributeType': 'S'}, {'AttributeName': 'year', 'Attribut
eType': 'N'}], 'TableName': 'Movies', 'KeySchema': [{'AttributeNam
e': 'year', 'KeyType': 'HASH'}, {'AttributeName': 'title', 'KeyTyp
e': 'RANGE'}], 'TableStatus': 'CREATING', 'CreationDateTime': date
time.datetime(2019, 11, 1, 13, 16, 26, 7000, tzinfo=tzlocal()), 'P
rovisionedThroughput': {'NumberOfDecreasesToday': 0, 'ReadCapacity
Units': 10, 'WriteCapacityUnits': 10}, 'TableSizeBytes': 0, 'ItemC
ount': 0, 'TableArn': 'arn:aws:dynamodb:us-east-
2:801211634689:table/Movies', 'TableId': '47a3a42a-593f-4388-aa96-
8c437777d24f'}, 'ResponseMetadata': {'RequestId': 'CBC5U8VP9MR31VK
LA6SP3JNPR3VV4KQNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode': 200, 'HT
TPHeaders': {'server': 'Server', 'date': 'Fri, 01 Nov 2019 13:16:2
5 GMT', 'content-type': 'application/x-amz-json-1.0', 'content-
length': '568', 'connection': 'keep-alive', 'x-amzn-
requestid': 'CBC5U8VP9MR31VKLA6SP3JNPR3VV4KQNSO5AEMVJF66Q9ASUAAJG'
, 'x-amz-crc32': '1456136828'}, 'RetryAttempts': 0}}
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019

iii.



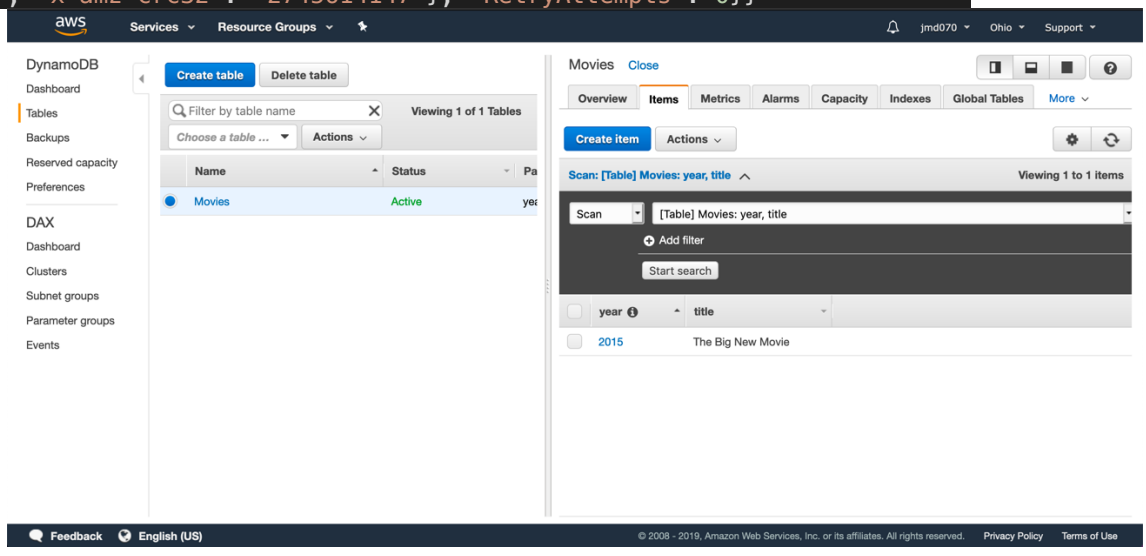
b.

i. Create a new item

ii.

```
{'ResponseMetadata': {'RequestId': '9P5CLCMPE1DSF1K896U90R7H1VVV4K
QNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode': 200, 'HTTPHeaders': {'s
erver': 'Server', 'date': 'Fri, 01 Nov 2019 13:18:22 GMT', 'conten
t-type': 'application/x-amz-json-1.0', 'content-
length': '2', 'connection': 'keep-alive', 'x-amzn-
requestid': '9P5CLCMPE1DSF1K896U90R7H1VVV4KQNSO5AEMVJF66Q9ASUAAJG'
, 'x-amz-crc32': '2745614147'}, 'RetryAttempts': 0}}
```

iii.



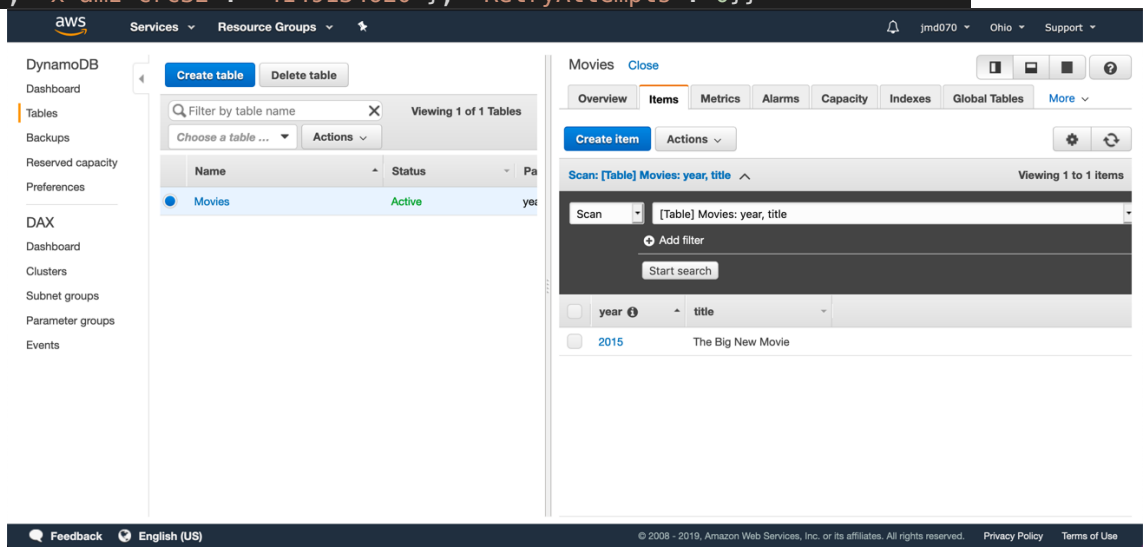
c.

i. Get (retrieve) an item

John Do
CSC 452-001
Dr. Droz
November 1, 2019

ii.
```
{'Item': {'title': {'S': 'The Big New Movie'}}, 'ResponseMetadata'
: {'RequestId': '64NPO5TDGNROBV92R3H2JQOS4RVV4KQNSO5AEMVJF66Q9ASUA
AJG', 'HTTPStatusCode': 200, 'HTTPHeaders': {'server': 'Server', '
date': 'Fri, 01 Nov 2019 13:20:04 GMT', 'content-
type': 'application/x-amz-json-1.0', 'content-
length': '44', 'connection': 'keep-alive', 'x-amzn-
requestid': '64NPO5TDGNROBV92R3H2JQOS4RVV4KQNSO5AEMVJF66Q9ASUAAJG'
, 'x-amz-crc32': '4149134620'}, 'RetryAttempts': 0}}
```

iii.



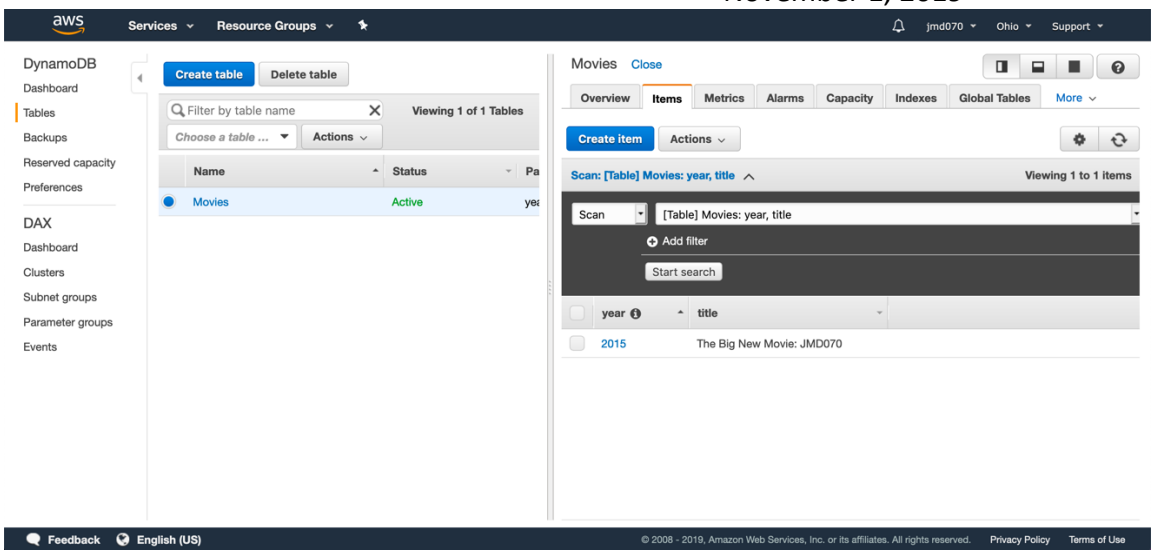
d.

i. Update attributes of the item

ii.
```
{'ResponseMetadata': {'RequestId': 'ODN666S8K1ND071BM9V2CELQVRVV4K
QNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode': 200, 'HTTPHeaders': {'s
erver': 'Server', 'date': 'Fri, 01 Nov 2019 13:20:54 GMT', 'conten
t-type': 'application/x-amz-json-1.0', 'content-
length': '2', 'connection': 'keep-alive', 'x-amzn-
requestid': 'ODN666S8K1ND071BM9V2CELQVRVV4KQNSO5AEMVJF66Q9ASUAAJG'
, 'x-amz-crc32': '2745614147'}, 'RetryAttempts': 0}}
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019

iii.



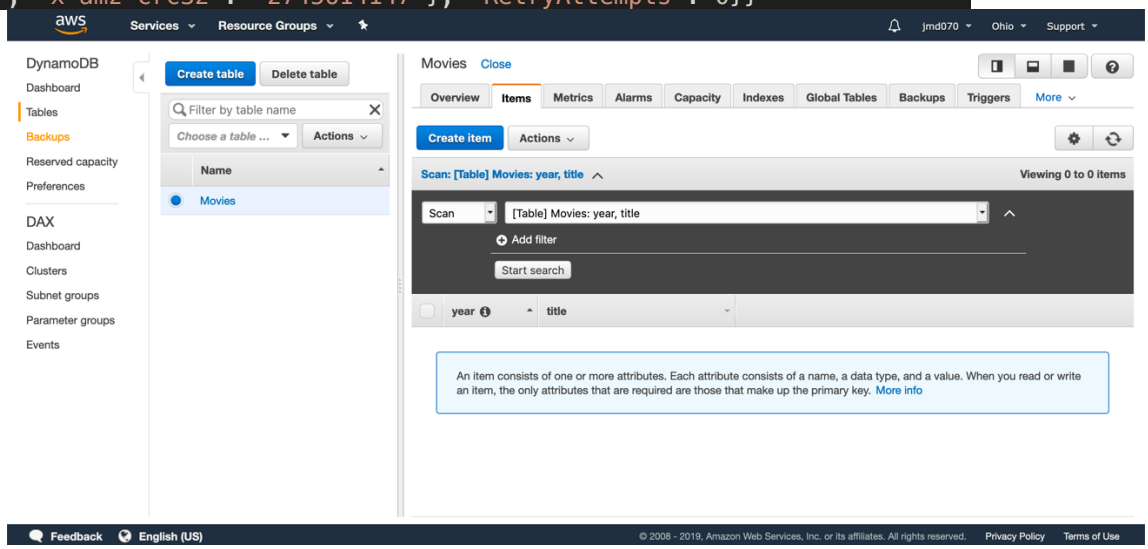
e.

i. Delete an item

ii.

```
{'ResponseMetadata': {'RequestId': 'N21R5ODA2605J06PDTRA5JN8GJVV4K
QNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode': 200, 'HTTPHeaders': {'s
erver': 'Server', 'date': 'Fri, 01 Nov 2019 13:24:50 GMT', 'conten
t-type': 'application/x-amz-json-1.0', 'content-
length': '2', 'connection': 'keep-alive', 'x-amzn-
requestid': 'N21R5ODA2605J06PDTRA5JN8GJVV4KQNSO5AEMVJF66Q9ASUAAJG'
, 'x-amz-crc32': '2745614147'}, 'RetryAttempts': 0}}
```

iii.



f.

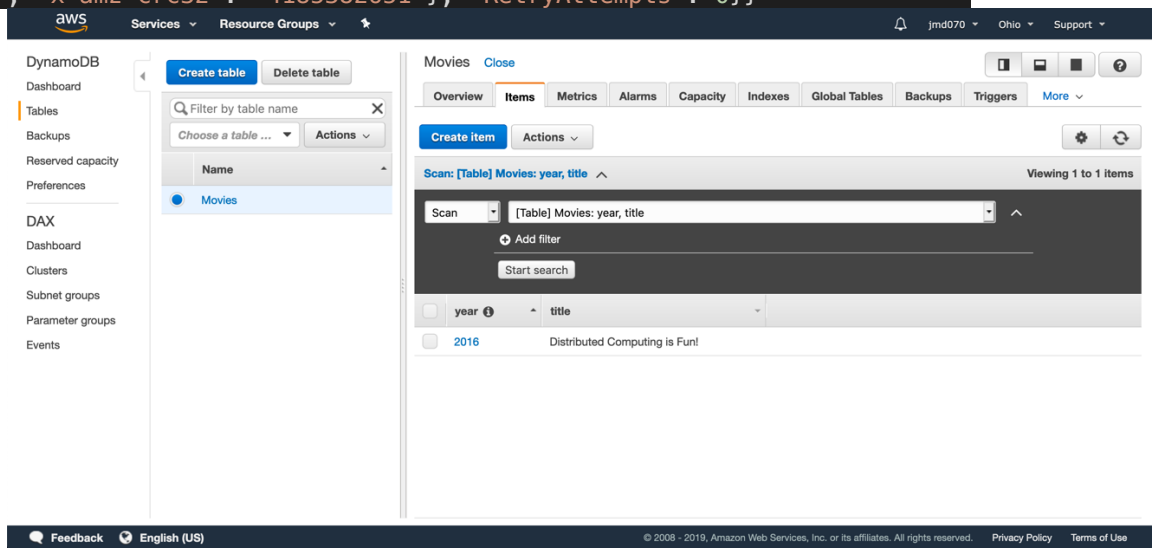
i. Batch writes to a table

John Do
CSC 452-001
Dr. Droz
November 1, 2019

ii.

```
{'UnprocessedItems': {}, 'ResponseMetadata': {'RequestId': 'U4GTFF
VDS5O1A0935MSF9I8ARJVV4KQNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode':
 200, 'HTTPHeaders': {'server': 'Server', 'date': 'Fri, 01 Nov 201
9 13:26:17 GMT', 'content-type': 'application/x-amz-json-
1.0', 'content-length': '23', 'connection': 'keep-alive', 'x-amzn-
requestid': 'U4GTFFVDS5O1A0935MSF9I8ARJVV4KQNSO5AEMVJF66Q9ASUAAJG'
, 'x-amz-crc32': '4185382651'}, 'RetryAttempts': 0}}
```

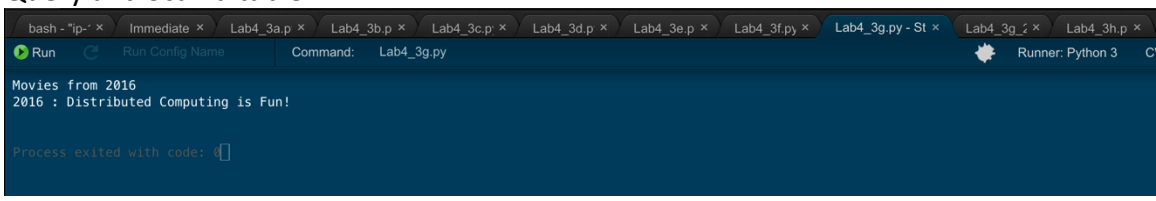
iii.



g.

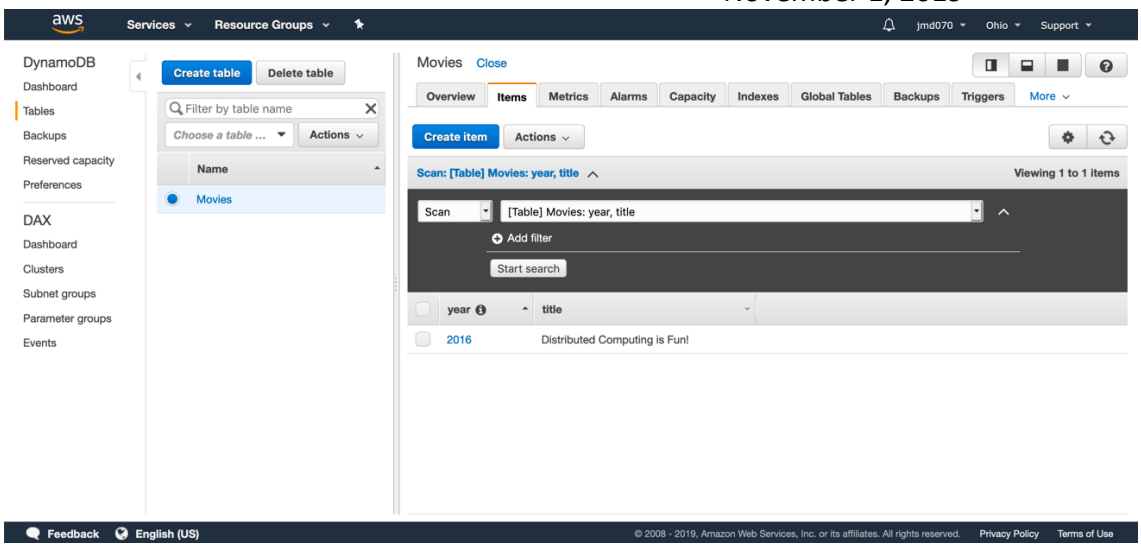
i. Query and Scan a table

ii.

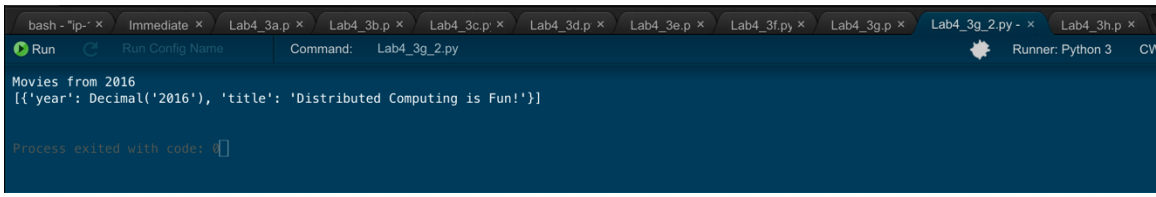


John Do
CSC 452-001
Dr. Droz
November 1, 2019

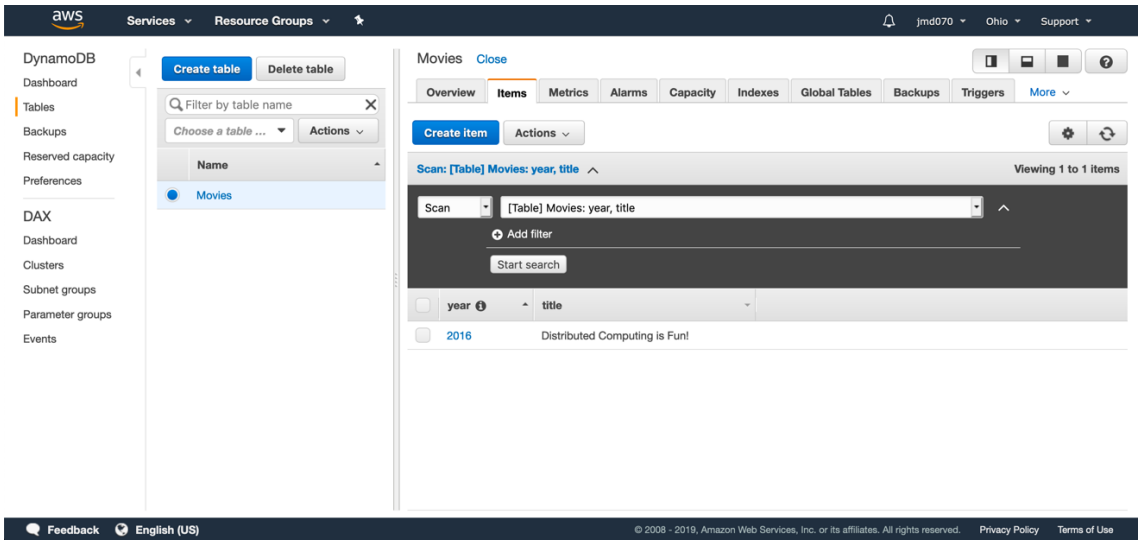
iii.



iv.



v.



h.

i. Delete a table

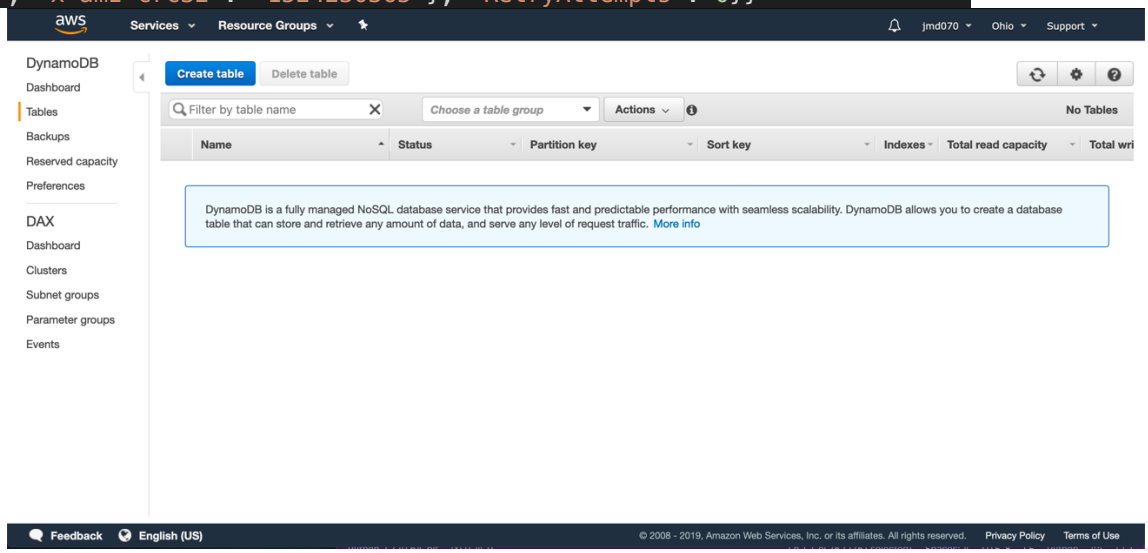
ii.

```
{'TableDescription': {'TableName': 'Movies', 'TableStatus': 'DELET
ING', 'ProvisionedThroughput': {'NumberOfDecreasesToday': 0, 'Read
CapacityUnits': 10, 'WriteCapacityUnits': 10}, 'TableSizeBytes': 0
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019

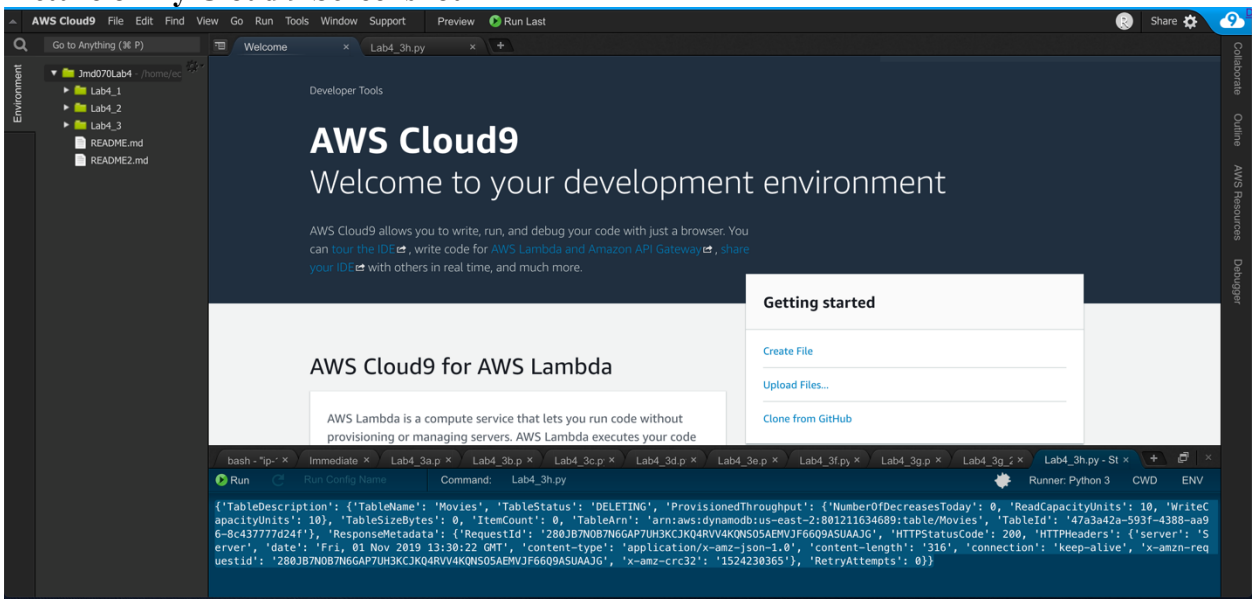

```
, 'ItemCount': 0, 'TableArn': 'arn:aws:dynamodb:us-east-
2:801211634689:table/Movies', 'TableId': '47a3a42a-593f-4388-aa96-
8c437777d24f'}, 'ResponseMetadata': {'RequestId': '280JB7NOB7N6GAP
7UH3KCJKQ4RVV4KQNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode': 200, 'HT
TPHeaders': {'server': 'Server', 'date': 'Fri, 01 Nov 2019 13:30:2
2 GMT', 'content-type': 'application/x-amz-json-1.0', 'content-
length': '316', 'connection': 'keep-alive', 'x-amzn-
requestid': '280JB7NOB7N6GAP7UH3KCJKQ4RVV4KQNSO5AEMVJF66Q9ASUAAJG'
, 'x-amz-crc32': '1524230365'}, 'RetryAttempts': 0}}
```


iii.



4. Picture of my Cloud 9 Screenshot

a.



John Do
CSC 452-001
Dr. Droz
November 1, 2019

Lab Report 4:

I. Introduction:

Distributed Systems has been the norm in computer science, whereas past decades of sequential programming have been the standard, developing parallel programs for distributed systems. The purpose of this is lab is to get the basic fundamentals of using AWS EC2 server and using basic functions from python such as boto3. As a class, we are creating a new environment on AWS Cloud 9. AWS Cloud 9 allows you to write, run, and debug your code with just a browser. With Cloud 9, you have immediate access to a rich code editor, integrated debugger, and built-in terminal with preconfigured AWS CLI (Command Line Interface). Within the Cloud 9 editor, we are creating a new environment and writing and executing python boto3 programs with the use of EC2 service and S3 storage. With the EC2 service, we are messing with the instances such as getting basic information to detailed information, starting, stopping, and rebooting. An addition, we are doing the same with key pairs and security groups. After we run and execute the codes, the class will provide screenshot of our results along with pasting the code down to ensure that the lab is done. For the extra credit, we also wrote and execute python programs that interacts with Amazon DynamoDB. The following program creates a table and a new item. It gets the

information from the table when we specify it. Whenever we want to change a certain information, we can update the attribute. The program deletes an item when calling upon, and we can batch write puts one or multiple information in one or more tables. Some programs can query the information from the table when using the right query. We can also scan for the specified information that the user wants. When the table is unneeded, we can delete it using the program as well. The structure of report to create a new environment on the cloud 9 service to execute rich python code that has the built-in function called boto3 to interact with the EC2 service and S3 storage. The problem of EC2 is that there are limitations of what we can do since we don't have the proper authorization since the class is using the free version of the AWS. For example, storage can be a problem because we are giving 5 GB of storage and if we want to increase more than we have to pay money. This relates to the definition of pay-as-you-grow method. AWS is developed by amazon to host as platform-as-a-service because it allows developers to deploy their applications on their platform without having to handle the complexity of the hardware and OS. Another problem is sometimes the code cannot interact with the instances because of some permissions or region issues. The precision of the writing these codes to understand the python boto3 documentation to fully comprehend the process and how the results are being generated. After implementing different methods and algorithms, experiments will be conducted and reported on different values being inputted and get an understanding on how the outcome is generated. Results will be stated the report after conducting experiments and demonstrating how the outcome was gotten. At the end of the report, there will be a conclusions and future work, which will summarize the whole report and determining how this lab will be used for future conducting experiments.

II. Methods:

The metrics of importance to our distributed or cloud systems are time, availability, and interacting with services with different methods on the service. During the lab, we are creating, deleting, rebooting, gathering information different instances, security groups, and keypairs within the EC2 service. Also, we are uploading, accessing, listing, and retrieving S3 buckets from the AWS S3 storage. AWS is a subsidiary of amazon that provides on-demand cloud computing platforms to individuals, companies, and governments. It is on a metered pay-as-you-grow basis. We are creating python codes that can-do different methods of interaction within the EC2 service or S3 service. For example, we created a new environment within the cloud 9 service, which is a browser idle to write and execute codes. After we created the environment, we are implementing different methods of interaction using the boto3 function within the python library. The methods consist of getting basic information, start and stop detailed monitoring, start and stop, reboot Amazon EC2 instances. Along with getting information, creating, and deleting key pairs and security groups. Then we are writing and executing python boto3 programs that do follow using Amazon S3 such as create, list, upload files, download files, grant temporary access using a resigned URL, retrieve a bucket policy, and get a bucket access control list. Some of these programs require specific ID's and names to be accurate with the function inputs such as Group ID, Group Name, Instance ID, etc. The following program creates a table and a new item. It gets the information from the table when we specify it. Whenever we want to change a certain information, we can update the attribute. The program deletes an item when calling upon, and we can batch write puts one or multiple information in one or more tables. Some programs can

query the information from the table when using the right query. We can also scan for the specified information that the user wants. When the table is unneeded, we can delete it using the program as well. These are important when you are trying to specify a certain instance or bucket to interact with since we are only demonstrating using one of them. Another input that might be considered is the region of where the service is being hosted on. In this case, I was using us-east-1 for some of my programs because the functions need to locate where you instance, and buckets are being hosted on.

Write and execute **Python Boto3** Programs that do the following using **Amazon Elastic Computer Cloud (EC2)**

A. Get Basic information about your amazon EC2 instance:
    a. Line 5: Import boto3 Library
    b. Line 7: Set the variable to connect to the client, which is ec2
    c. Line 9: Set the response to describe the instance in ec2
    d. Line 11: print the response variable; outputting the information of the instance
B. Start and stop detailed monitoring of an Amazon EC2 instance
    a. Line 5-6: Import boto3 libraries
    b. Line 8: Set the ec2 variable to connect to the ec2 client
    c. Line 11- 14: If the user set the index argument to "ON", it will monitor the specific instance that it was set to. Else, it would set the response to unmonitored the specific instance
    d. Line 16: Once we got the response variable, it will print out the result
C. Start and stop an Amazon EC2 instance
    a. Line 5-7: Import the libraries
    b. Line 9: Set the variable to connect to the ec2 server
    c. Line 11: Whatever the user input is for the argument, it will be uppercase
    d. Line 13: If the action variable is "ON"
    e. Line 14-19: Do a dry run first to verify permissions
    f. Line 22-26: Dry run succeeded, run start instances again without dry run
    g. Line 29-33: Do a dry run first to verify permissions
    h. Line 36-40: Dry run succeeded, run stop instances again without dry run
D. Reboot an Amazon EC2 instance
    a. Line 5-7: import the libraries
    b. Line 9: Set the variable to connect to the ec2 server
    c. Line 13-18: Do a dry run first to verify permissions to reboot the instance and if it fails then it prints out an error message
    d. Line 22-26: Dry run succeeded, reboot instances again without dry run printing out the success response. Else prints out the error message
E. Get information about your key pairs
    a. Line 5-7: import the libraries
    b. Line 9: set the variable to connect to the EC2 server
    c. Line 11: Describe the EC2 Key Pair
    d. Line 13: Print success with the response
F. Create a key pair to access an Amazon EC2 instance

a. Line 5: import the library
b. Line 9: Set the variable to connect to the EC2 server
c. Line 11: This variable creates a new key pair with the desired name
d. Line 13: print out the success and the response

G. Delete an existing key pair
    a. Line 5: Import the library
    b. Line 9: Set the variable to connect to the EC2 server
    c. Line 11: Set the variable to delete the specific key pair
    d. Line 13: print out the success message, along with the response
H. Get information about your security groups
    a. Line 5-6: Import the libraries
    b. Line 9: Set the variable to connect to the ec2 server
    c. Line 12-16: Try to set the response variable to get information about the security group. Print success and the response variable. Else, it fails and prints out a error message.
I. Create a security group to access an Amazon EC2 instance
    a. Line 5-6: Import The libraries
    b. Line 9: Set the variable to connect to the ec2 server
    c. Line 13-18: Try to create a new security group with the desired name and a description. print success when it is done along with the responses. If it fails, then it prints out the error.
J. Delete a security Group:
    a. Line 5-6: Import the libraries
    b. Line 8: Set the variable to connect to the ec2 server
    c. Line 12-16: Try to delete the security group with the specific Group ID. Print out the success and the response when its finished. Else if it fails, it prints out the error message.

Write and execute **Python Boto3** programs that do the following using **Amazon S3**:

A. Create an Amazon S3 Bucket
    a. Line 5-7: Import the libraries
    b. Line 10: This is a function with two input
    c. Line 12-15: if the region is none, connect to the s3 server and create a bucket with the bucket name
    d. Line 19-23: s3 client connects to s3 with the specified region. Prints out the location constraint with the region. Create a bucket with the desired bucket name with the configuration of the location
    e. Line 25-27: If that fails, it prints out an error message
    f. Line 29: if it doesn't print out a message, it returns true.
B. List Existing Buckets
    a. Line 5: Import boto3 Library
    b. Line 7: Connect to s3 client
    c. Line 9: set the response variable to list the buckets
    d. Line 11-13: output the bucket names

John Do
CSC 452-001
Dr. Droz
November 1, 2019

C. Upload Files to an S3 Bucket
    a. Line 5: Import boto3 Library
    b. Line 7: set the s3 client variable to connect to s3 using boto3
    c. Line 9-10: set the variable containing the file name and the bucket name
    d. Line 12: This will upload the file to s3 storage with the filename using the bucket name
D. Download files from an S3 Bucket
    a. Line 5-6: Import boto3 Library
    b. Line 8: Set the s3 variable to connect the s3 resources
    c. Line 10-12: Set the variable including the filename, the key, and the bucket name
    d. Line 14: Connect to the bucket name and download the files
E. Grant temporary access using a preassigned URL
    a. Line 5-7: Import boto3 Library
    b. Line 10: This is the function with 3 arguments including the bucket name, object name, and the expiration.
    c. Line 12: Set the s3_client variable to connect to the s3 storage client
    d. Line 14-15: Try to set the response to the generated preassigned URL
    e. Line 17-19: if it fails, then it will print an error
    f. Line 22: The response contains the preassigned URL
    g. Line 25: This will call the function with the desired arguments
F. Retrieve a bucket policy
    a. Line 5-6: Import boto3 Library
    b. Line 9: Set the s3 variable to connect to the s3 storage
    c. Line 10: Set the bucket name with the specified bucket
    d. Line 15-24: This will set the bucket policy
    e. Line 26: set the bucket policy variable, this will convert the bucket policy to JSON format since it is python
    f. Line 28: This will replace the policy fora specific Amazon S3 bucket. A bucket policy can be set.
    g. Line 31: The response variable is set to retrieve a bucket's policy by calling the AWS SDK
    h. Line 33: This will print the response and the policy
G. Get the Bucket Access Control List
    a. Line 5-7: Import boto3 Library
    b. Line 9-11: Set the bucket name variable to a specified name, filename, and key
    c. Line 13: Connect to the s3 storage
    d. Line 15: Set the response variable to get the bucket access control list using the bucket name
    e. Line 17: This will print the response

(**Extra Credit**, max +2 points on your final grade) Write and execute **Python Boto3** programs that do the following using **Amazon DynamoDB**:

A. Create a new table
    a. Line 2-3: Import libraries

   b. Line 6: Connect to the dynamodb database
   c. Line 10-37: Set the response variable to create a table name "Movies" with different attributes such as year and title. The year and title are in a dictionary format to apply key values to it
   d. Line 40: print the response

B. Create a new item:
   a. Line 2-3: Import libraries
   b. Line 7: Connect to the dynamodb database
   c. Line 10-11: Set the variables
   d. Line 13-24: set the key with values in the table
   e. Line 27: print the response

C. Get (retrieve) an item
   a. Line 2-3: Import libraries
   b. Line 7: Connect to the dynamodb database
   c. Line 10-11: Set the variables
   d. Line 14-28: Get the specified attribute from the table
   e. Line 31: Print the response

D. Update attributes of the item
   a. Line 2-3: Import libraries
   b. Line 7: Connect to the dynamodb database
   c. Line 10-11: Set the variables
   d. Line 14-25: Update the item in the table
   e. Line 28: Print the response

E. Delete an item
   a. Line 2-3: Import libraries
   b. Line 7: Connect to the dynamodb database
   c. Line 10-11: Set the variables
   d. Line 13-24: Delete an item from the table
   e. Line 27: Print the response

F. Batch writes to a table
   a. Line 2-3: Import libraries
   b. Line 7: Connect to the dynamodb database
   c. Line 10-11: Set the variables
   d. Line 14-30: The batch write operation puts and deletes multiple items in one or more tables
   e. Line 34: Print the response

G. Query and Scan a table
   a. Query
      i. Line 2-3: Import libraries
      ii. Line 7: Connect to the dynamodb database
      iii. Line 9: Set the variables
      iv. Line 11: print out a statement
      v. Line 13-15: query to retrieve information about the specified input
      vi. Line 18-19: Print the response
   b. Scan

i. Line 2-3: Import libraries
ii. Line 7: Connect to the dynamodb database
iii. Line 9: Set the variables
iv. Line 11: print out a statement
v. Line 13-15: Scan the table to look for a certain input
vi. Line 18-19: Print the response

H. Delete a table:
   a. Line 2-3: Import libraries
   b. Line 7: Connect to the dynamodb database
   c. Line 9: Set the variable to the movie table
   d. Line 12: delete the table
   e. Line 15: print the response

III. Experiments Results:

Test Matrix

Labs	Description	Tools
Lab4_1a	Get basic information about your Amazon EC2 instances	HW: AWS SW: Amazon Webserver, Cloud 9 editor, Amazon EC2 instances. Programming Language: Python
Lab4_1b	Start and stop detailed monitoring of an Amazon EC2 instance	HW: AWS SW: Amazon Webserver, Cloud 9 editor, Amazon EC2 instances. Programming Language: Python
Lab4_1c	Start and stop an Amazon EC2 instance	HW: AWS SW: Amazon Webserver, Cloud 9 editor, Amazon EC2 instances. Programming Language: Python
Lab4_1d	Reboot an Amazon EC2 instance	HW: AWS SW: Amazon Webserver, Cloud 9 editor, Amazon EC2 instances. Programming Language: Python

Lab4_1e	Get information about your key pairs	HW: AWS SW: Amazon Webserver, Cloud 9 editor, Amazon EC2 instances. Programming Language: Python
Lab4_1f	Create a key pair to access an Amazon EC2 instance	HW: AWS SW: Amazon Webserver, Cloud 9 editor, Amazon EC2 instances. Programming Language: Python
Lab4_1g	Delete an existing key pair	HW: AWS SW: Amazon Webserver, Cloud 9 editor, Amazon EC2 instances. Programming Language: Python
Lab4_1h	Get information about your security groups	HW: AWS SW: Amazon Webserver, Cloud 9 editor, Amazon EC2 instances. Programming Language: Python
Lab4_1i	Create a security group to access an Amazon EC2 instance	HW: AWS SW: Amazon Webserver, Cloud 9 editor, Amazon EC2 instances. Programming Language: Python
Lab4_1j	Delete an existing security group	HW: AWS SW: Amazon Webserver, Cloud 9 editor, Amazon EC2 instances. Programming Language: Python
Lab4_2a	Create an Amazon S3 Bucket	HW: AWS SW: Amazon Webserver, Cloud 9 editor, S3 Storage Programming Language: Python
Lab4_2b	List Existing Buckets	HW: AWS SW: Amazon Webserver, Cloud 9 editor, S3 Storage

		Programming Language: Python
Lab4_2c	Upload Files to an S3 Bucket	HW: AWS SW: Amazon Webserver, Cloud 9 editor, S3 Storage Programming Language: Python
Lab4_2d	Download Files from an S3 Bucket	HW: AWS SW: Amazon Webserver, Cloud 9 editor, S3 Storage Programming Language: Python
Lab4_2e	Grant temporary access using a presigned URL	HW: AWS SW: Amazon Webserver, Cloud 9 editor, S3 Storage Programming Language: Python
Lab4_2f	Retrieve a bucket policy	HW: AWS SW: Amazon Webserver, Cloud 9 editor, S3 Storage Programming Language: Python
Lab4_2g	Get a Bucket Access Control List	HW: AWS SW: Amazon Webserver, Cloud 9 editor, S3 Storage Programming Language: Python

Extra Credit (Lab4_3):

Lab4_3a	Create a new table	HW: AWS SW: Amazon Webserver, Cloud 9 editor, DynamoDB Programming Language: Python
Lab4_3b	Create a new item	HW: AWS SW: Amazon Webserver, Cloud 9 editor, DynamoDB Programming Language: Python
Lab4_3c	Get(retrieve) an item	HW: AWS SW: Amazon Webserver, Cloud 9 editor, DynamoDB

		Programming Language: Python
Lab4_3d	Update attributes of the item	HW: AWS SW: Amazon Webserver, Cloud 9 editor, DynamoDB Programming Language: Python
Lab4_3e	Delete an item	HW: AWS SW: Amazon Webserver, Cloud 9 editor, DynamoDB Programming Language: Python
Lab4_3f	Batch write to a table	HW: AWS SW: Amazon Webserver, Cloud 9 editor, DynamoDB Programming Language: Python
Lab4_3g	Query and scan a table	HW: AWS SW: Amazon Webserver, Cloud 9 editor, DynamoDB Programming Language: Python
Lab4_3h	Delete a table	HW: AWS SW: Amazon Webserver, Cloud 9 editor, DynamoDB Programming Language: Python

5. The utility that was provided is called a “AWS”. AWS is a subsidiary of amazon that provides on-demand cloud computing platforms to individuals, companies, and governments. It is on a metered pay-as-you-grow basis. This cluster is located at an Amazon server. We acquire access to it by making an account on AWS. After we created the account, we create, start, delete, reboot, and describe instances on the EC2 by implementing programs using the cloud 9 environment. The programs are written in python using the boto3 library. This is also done on the S3 storage within the AWS webserver. We are creating python codes that can-do different methods of interaction

within the EC2 service or S3 service. For example, we created a new environment within the cloud 9 service, which is a browser idle to write and execute codes. After we created the environment, we are implementing different methods of interaction using the boto3 function within the python library. The methods consist of getting basic information, start and stop detailed monitoring, start and stop, reboot Amazon EC2 instances. Along with getting information, creating, and deleting key pairs and security groups. Then we are writing and executing python boto3 programs that do follow using Amazon S3 such as create, list, upload files, download files, grant temporary access using a resigned URL, retrieve a bucket policy, and get a bucket access control list.

THE BLACK TEXTBOX ARE THE OUTPUTS FROM THE PROGRAMS. IT WAS TOO LONG TO SCREENSHOT SO I COPY AND PASTE IT ON HERE.

6. Make a new environment on AWS Cloud 9. Write and execute **Python Boto3** programs that do the following using **Amazon Elastic Cloud (EC2).**

a.

i. Get basic information about your Amazon EC2 Instances

ii.

```
{'Reservations': [{'Groups': [], 'Instances': [{'AmiLaunchIndex':
0, 'ImageId': 'ami-045e373af522ed8ef', 'InstanceId': 'i-
0ae020bd6b53616a3', 'InstanceType': 't2.micro', 'LaunchTime': date
time.datetime(2019, 10, 24, 19, 15, 22, tzinfo=tzlocal()), 'Monito
ring': {'State': 'disabled'}, 'Placement': {'AvailabilityZone': 'u
s-east-
2c', 'GroupName': '', 'Tenancy': 'default'}, 'PrivateDnsName': 'ip
-172-31-41-202.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.41.202', 'Product
Codes': [], 'PublicDnsName': 'ec2-18-222-170-19.us-east-
2.compute.amazonaws.com', 'PublicIpAddress': '18.222.170.19', 'Sta
te': {'Code': 16, 'Name': 'running'}, 'StateTransitionReason': '',
 'SubnetId': 'subnet-7b942d37', 'VpcId': 'vpc-
e671818d', 'Architecture': 'x86_64', 'BlockDeviceMappings': [{'Dev
iceName': '/dev/xvda', 'Ebs': {'AttachTime': datetime.datetime(201
9, 10, 24, 19, 15, 23, tzinfo=tzlocal()), 'DeleteOnTermination': T
rue, 'Status': 'attached', 'VolumeId': 'vol-
0f385f6a3ca5480bc'}}], 'ClientToken': 'aws-c-Insta-
TLC11LAEICI1', 'EbsOptimized': False, 'EnaSupport': True, 'Hypervi
sor': 'xen', 'NetworkInterfaces': [{'Association': {'IpOwnerId': '
amazon', 'PublicDnsName': 'ec2-18-222-170-19.us-east-
2.compute.amazonaws.com', 'PublicIp': '18.222.170.19'}, 'Attachmen
t': {'AttachTime': datetime.datetime(2019, 10, 24, 19, 15, 22, tzi
nfo=tzlocal()), 'AttachmentId': 'eni-attach-
09f8f35700603144d', 'DeleteOnTermination': True, 'DeviceIndex': 0,
 'Status': 'attached'}, 'Description': '', 'Groups': [{'GroupName'
: 'aws-cloud9-PythonAWSLab4-96d7c723888c4216803f9843a61de546-
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019


```
InstanceSecurityGroup-EAM80YJFI9PV', 'GroupId': 'sg-
0fff90f241d382d47'}], 'Ipv6Addresses': [], 'MacAddress': '0a:69:de
:59:4c:90', 'NetworkInterfaceId': 'eni-
05e652bc91fdbfb47', 'OwnerId': '801211634689', 'PrivateDnsName': '
ip-172-31-41-202.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.41.202', 'Private
IpAddresses': [{'Association': {'IpOwnerId': 'amazon', 'PublicDnsN
ame': 'ec2-18-222-170-19.us-east-
2.compute.amazonaws.com', 'PublicIp': '18.222.170.19'}, 'Primary':
 True, 'PrivateDnsName': 'ip-172-31-41-202.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.41.202'}], 'Sourc
eDestCheck': True, 'Status': 'in-use', 'SubnetId': 'subnet-
7b942d37', 'VpcId': 'vpc-
e671818d', 'InterfaceType': 'interface'}], 'RootDeviceName': '/dev
/xvda', 'RootDeviceType': 'ebs', 'SecurityGroups': [{'GroupName':
'aws-cloud9-PythonAWSLab4-96d7c723888c4216803f9843a61de546-
InstanceSecurityGroup-EAM80YJFI9PV', 'GroupId': 'sg-
0fff90f241d382d47'}], 'SourceDestCheck': True, 'Tags': [{'Key': 'a
ws:cloudformation:stack-id', 'Value': 'arn:aws:cloudformation:us-
east-2:801211634689:stack/aws-cloud9-PythonAWSLab4-
96d7c723888c4216803f9843a61de546/9a7619c0-f692-11e9-ab11-
0ade6091f9d8'}, {'Key': 'Name', 'Value': 'aws-cloud9-
PythonAWSLab4-
96d7c723888c4216803f9843a61de546'}, {'Key': 'aws:cloud9:environmen
t', 'Value': '96d7c723888c4216803f9843a61de546'}, {'Key': 'aws:clo
ud9:owner', 'Value': '801211634689'}, {'Key': 'aws:cloudformation:
logical-
id', 'Value': 'Instance'}, {'Key': 'aws:cloudformation:stack-
name', 'Value': 'aws-cloud9-PythonAWSLab4-
96d7c723888c4216803f9843a61de546'}], 'VirtualizationType': 'hvm',
'CpuOptions': {'CoreCount': 1, 'ThreadsPerCore': 1}, 'CapacityRese
rvationSpecification': {'CapacityReservationPreference': 'open'},
'HibernationOptions': {'Configured': False}}], 'OwnerId': '8012116
34689', 'RequesterId': '043320173835', 'ReservationId': 'r-
05803fe9f22f6ad1b'}, {'Groups': [], 'Instances': [{'AmiLaunchIndex
': 0, 'ImageId': 'ami-09ee46c8d9abcb966', 'InstanceId': 'i-
0f407d811567d595e', 'InstanceType': 't2.micro', 'KeyName': 'John_L
ab3key', 'LaunchTime': datetime.datetime(2019, 10, 23, 22, 45, 49,
 tzinfo=tzlocal()), 'Monitoring': {'State': 'disabled'}, 'Placemen
t': {'AvailabilityZone': 'us-east-
2c', 'GroupName': '', 'Tenancy': 'default'}, 'Platform': 'windows'
, 'PrivateDnsName': 'ip-172-31-38-45.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.38.45', 'ProductC
odes': [], 'PublicDnsName': '', 'State': {'Code': 80, 'Name': 'sto
pped'}, 'StateTransitionReason': 'User initiated (2019-10-
24 02:57:41 GMT)', 'SubnetId': 'subnet-7b942d37', 'VpcId': 'vpc-
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019

```
e671818d', 'Architecture': 'x86_64', 'BlockDeviceMappings': [{'Dev
iceName': '/dev/sda1', 'Ebs': {'AttachTime': datetime.datetime(201
9, 10, 17, 19, 56, 43, tzinfo=tzlocal()), 'DeleteOnTermination': T
rue, 'Status': 'attached', 'VolumeId': 'vol-
0bb1943a27830f761'}}], 'ClientToken': '', 'EbsOptimized': False, '
EnaSupport': True, 'Hypervisor': 'xen', 'NetworkInterfaces': [{'At
tachment': {'AttachTime': datetime.datetime(2019, 10, 17, 19, 56,
42, tzinfo=tzlocal()), 'AttachmentId': 'eni-attach-
07dd2756b17b70d09', 'DeleteOnTermination': True, 'DeviceIndex': 0,
 'Status': 'attached'}, 'Description': '', 'Groups': [{'GroupName'
: 'launch-wizard-1', 'GroupId': 'sg-
072cf6562ea4b3c77'}], 'Ipv6Addresses': [], 'MacAddress': '0a:5d:e2
:0a:35:94', 'NetworkInterfaceId': 'eni-
0506a7da1425ace93', 'OwnerId': '801211634689', 'PrivateDnsName': '
ip-172-31-38-45.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.38.45', 'PrivateI
pAddresses': [{'Primary': True, 'PrivateDnsName': 'ip-172-31-38-
45.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.38.45'}], 'Source
DestCheck': True, 'Status': 'in-use', 'SubnetId': 'subnet-
7b942d37', 'VpcId': 'vpc-
e671818d', 'InterfaceType': 'interface'}], 'RootDeviceName': '/dev
/sda1', 'RootDeviceType': 'ebs', 'SecurityGroups': [{'GroupName':
'launch-wizard-1', 'GroupId': 'sg-
072cf6562ea4b3c77'}], 'SourceDestCheck': True, 'StateReason': {'Co
de': 'Client.UserInitiatedShutdown', 'Message': 'Client.UserInitia
tedShutdown: User initiated shutdown'}, 'VirtualizationType': 'hvm
', 'CpuOptions': {'CoreCount': 1, 'ThreadsPerCore': 1}, 'CapacityR
eservationSpecification': {'CapacityReservationPreference': 'open'
}, 'HibernationOptions': {'Configured': False}}], 'OwnerId': '8012
11634689', 'ReservationId': 'r-
06098e80830a11a7d'}, {'Groups': [], 'Instances': [{'AmiLaunchIndex
': 1, 'ImageId': 'ami-0d5d9d301c853a04a', 'InstanceId': 'i-
00d4bfd0ee3814815', 'InstanceType': 't2.micro', 'KeyName': 'John_U
nix2', 'LaunchTime': datetime.datetime(2019, 10, 23, 22, 45, 49, t
zinfo=tzlocal()), 'Monitoring': {'State': 'disabled'}, 'Placement'
: {'AvailabilityZone': 'us-east-
2c', 'GroupName': '', 'Tenancy': 'default'}, 'PrivateDnsName': 'ip
-172-31-39-155.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.39.155', 'Product
Codes': [], 'PublicDnsName': '', 'State': {'Code': 80, 'Name': 'st
opped'}, 'StateTransitionReason': 'User initiated (2019-10-
24 02:57:41 GMT)', 'SubnetId': 'subnet-7b942d37', 'VpcId': 'vpc-
e671818d', 'Architecture': 'x86_64', 'BlockDeviceMappings': [{'Dev
iceName': '/dev/sda1', 'Ebs': {'AttachTime': datetime.datetime(201
9, 10, 22, 19, 28, 5, tzinfo=tzlocal()), 'DeleteOnTermination': Tr
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019

```
ue, 'Status': 'attached', 'VolumeId': 'vol-
0e48f1f93c8deba35'}}], 'ClientToken': '', 'EbsOptimized': False, '
EnaSupport': True, 'Hypervisor': 'xen', 'NetworkInterfaces': [{'At
tachment': {'AttachTime': datetime.datetime(2019, 10, 22, 19, 28,
4, tzinfo=tzlocal()), 'AttachmentId': 'eni-attach-
0654dfb838deefa0a', 'DeleteOnTermination': True, 'DeviceIndex': 0,
 'Status': 'attached'}, 'Description': '', 'Groups': [{'GroupName'
: 'launch-wizard-3', 'GroupId': 'sg-
085a7e281dc20437d'}], 'Ipv6Addresses': [], 'MacAddress': '0a:14:7c
:64:5a:c2', 'NetworkInterfaceId': 'eni-
003854a15b290e5f0', 'OwnerId': '801211634689', 'PrivateDnsName': '
ip-172-31-39-155.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.39.155', 'Private
IpAddresses': [{'Primary': True, 'PrivateDnsName': 'ip-172-31-39-
155.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.39.155'}], 'Sourc
eDestCheck': True, 'Status': 'in-use', 'SubnetId': 'subnet-
7b942d37', 'VpcId': 'vpc-
e671818d', 'InterfaceType': 'interface'}], 'RootDeviceName': '/dev
/sda1', 'RootDeviceType': 'ebs', 'SecurityGroups': [{'GroupName':
'launch-wizard-3', 'GroupId': 'sg-
085a7e281dc20437d'}], 'SourceDestCheck': True, 'StateReason': {'Co
de': 'Client.UserInitiatedShutdown', 'Message': 'Client.UserInitia
tedShutdown: User initiated shutdown'}, 'VirtualizationType': 'hvm
', 'CpuOptions': {'CoreCount': 1, 'ThreadsPerCore': 1}, 'CapacityR
eservationSpecification': {'CapacityReservationPreference': 'open'
}, 'HibernationOptions': {'Configured': False}}, {'AmiLaunchIndex'
: 0, 'ImageId': 'ami-0d5d9d301c853a04a', 'InstanceId': 'i-
0f38542e6a1741748', 'InstanceType': 't2.micro', 'KeyName': 'John_U
nix2', 'LaunchTime': datetime.datetime(2019, 10, 23, 22, 45, 49, t
zinfo=tzlocal()), 'Monitoring': {'State': 'disabled'}, 'Placement'
: {'AvailabilityZone': 'us-east-
2c', 'GroupName': '', 'Tenancy': 'default'}, 'PrivateDnsName': 'ip
-172-31-45-90.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.45.90', 'ProductC
odes': [], 'PublicDnsName': '', 'State': {'Code': 80, 'Name': 'sto
pped'}, 'StateTransitionReason': 'User initiated (2019-10-
24 02:57:41 GMT)', 'SubnetId': 'subnet-7b942d37', 'VpcId': 'vpc-
e671818d', 'Architecture': 'x86_64', 'BlockDeviceMappings': [{'Dev
iceName': '/dev/sda1', 'Ebs': {'AttachTime': datetime.datetime(201
9, 10, 22, 19, 28, 5, tzinfo=tzlocal()), 'DeleteOnTermination': Tr
ue, 'Status': 'attached', 'VolumeId': 'vol-
0b05d9fcdfa7616e0'}}], 'ClientToken': '', 'EbsOptimized': False, '
EnaSupport': True, 'Hypervisor': 'xen', 'NetworkInterfaces': [{'At
tachment': {'AttachTime': datetime.datetime(2019, 10, 22, 19, 28,
4, tzinfo=tzlocal()), 'AttachmentId': 'eni-attach-
```

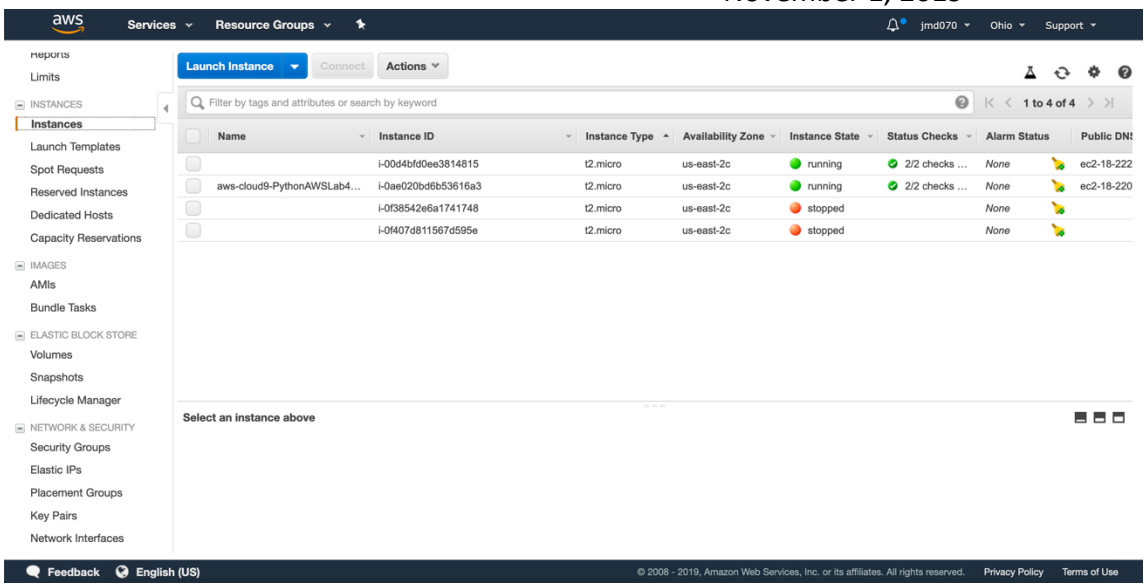
John Do
CSC 452-001
Dr. Droz
November 1, 2019

```
02610b799914e70bb', 'DeleteOnTermination': True, 'DeviceIndex': 0,
 'Status': 'attached'}, 'Description': '', 'Groups': [{'GroupName'
: 'launch-wizard-3', 'GroupId': 'sg-
085a7e281dc20437d'}], 'Ipv6Addresses': [], 'MacAddress': '0a:f1:30
:86:75:e4', 'NetworkInterfaceId': 'eni-
07358425607d2cb95', 'OwnerId': '801211634689', 'PrivateDnsName': '
ip-172-31-45-90.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.45.90', 'PrivateI
pAddresses': [{'Primary': True, 'PrivateDnsName': 'ip-172-31-45-
90.us-east-
2.compute.internal', 'PrivateIpAddress': '172.31.45.90'}], 'Source
DestCheck': True, 'Status': 'in-use', ': [{'GroupName': 'launch-
wizard-3', 'GroupId': 'sg-
085a7e281dc20437d'}], 'SourceDestCheck': True, 'StateReason': {'Co
de': 'Client.UserInitiatedShutdown', 'Message': 'Client.UserInitia
tedShutdown: User initiated shutdown'}, 'VirtualizationType': 'hvm
', 'CpuOptions': {'CoreCount': 1, 'ThreadsPerCore': 1}, 'CapacityR
eservationSpecification': {'CapacityReservationPreference': 'open'
}, 'HibernationOptions': {'Configured': False}}], 'OwnerId': '8012
11634689', 'ReservationId': 'r-
0c1564ee8f7c59929'}], 'ResponseMetadata': {'RequestId': 'cf997cc9-
3ef3-4b2c-b50d-
81445ad2312b', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'transfer-
encoding': 'chunked', 'vary': 'accept-
encoding', 'date': 'Thu, 24 Oct 2019 19:37:35 GMT', 'server': 'Ama
zonEC2'}, 'RetryAttempts': 0}}

    Process exited with code: 0
Pane is dead
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019

iii.



b.

i. Start and Stop Detailed Monitoring of an EC2 instance

ii.
```
{'InstanceMonitorings': [{'InstanceId': 'i-
0ae020bd6b53616a3', 'Monitoring': {'State': 'enabled'}}], 'Respons
eMetadata': {'RequestId': 'e38bd77a-e437-40dc-af9b-
726fec2ad400', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '424', 'date': 'Thu, 24 Oct 2019 19:47:11 GMT', 'server':
 'AmazonEC2'}, 'RetryAttempts': 0}}
```

c. OFF:

i.

ii.
```
{'StoppingInstances': [{'CurrentState': {'Code': 64, 'Name': 'stop
ping'}, 'InstanceId': 'i-
00d4bfd0ee3814815', 'PreviousState': {'Code': 16, 'Name': 'running
'}}], 'ResponseMetadata': {'RequestId': 'cf9c91d5-9d50-44b8-b253-
d461e692e3c8', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '579', 'date': 'Thu, 24 Oct 2019 20:01:15 GMT', 'server':
 'AmazonEC2'}, 'RetryAttempts': 0}}
```

ON:

```
        {'StartingInstances': [{'CurrentState': {'Code': 0, 'Name': 'pend
ing'}, 'InstanceId': 'i-
00d4bfd0ee3814815', 'PreviousState': {'Code': 80, 'Name': 'stopped'}}], '
ResponseMetadata': {'RequestId': '3c49ca16-8d39-47cf-a5f1-
0ba6affa624c', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
```

```
length': '579', 'date': 'Fri, 25 Oct 2019 21:19:22 GMT', 'server': 'Amazo
nEC2'}, 'RetryAttempts': 0}}
```


d.

i. This is rebooting the instance


```
success {'ResponseMetadata': {'RequestId': '6fc30fc1-56e5-466e-9558-
535150dc415b', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '231', 'date': 'Wed, 30 Oct 2019 15:29:47 GMT', 'server': '
AmazonEC2'}, 'RetryAttempts': 0}}
```

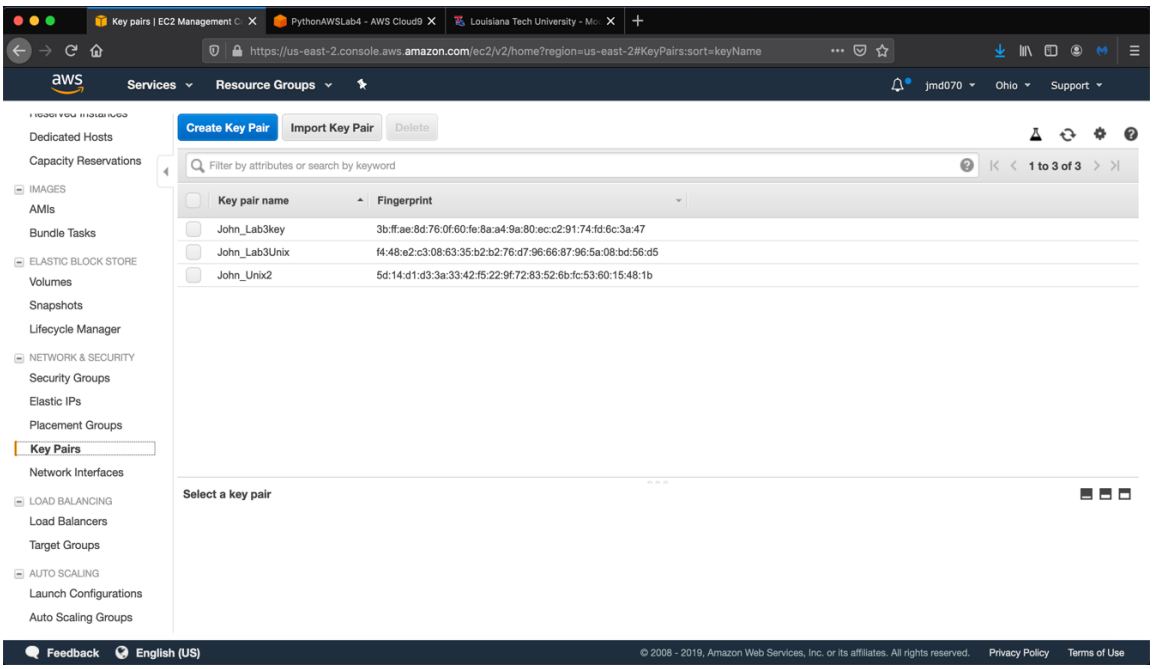

e.

i. Get info about the key pairs

ii.


```
success {'KeyPairs': [{'KeyFingerprint': '3b:ff:ae:8d:76:0f:60:fe:
8a:a4:9a:80:ec:c2:91:74:fd:6c:3a:47', 'KeyName': 'John_Lab3key'},
{'KeyFingerprint': 'f4:48:e2:c3:08:63:35:b2:b2:76:d7:96:66:87:96:5
a:08:bd:56:d5', 'KeyName': 'John_Lab3Unix'}, {'KeyFingerprint': '5
d:14:d1:d3:3a:33:42:f5:22:9f:72:83:52:6b:fc:53:60:15:48:1b', 'KeyN
ame': 'John_Unix2'}], 'ResponseMetadata': {'RequestId': 'a4fb448e-
00a1-4898-9b52-
c86bbbfec5e9', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '773', 'date': 'Wed, 30 Oct 2019 15:34:00 GMT', 'server':
'AmazonEC2'}, 'RetryAttempts': 0}}
```


iii.



John Do
CSC 452-001
Dr. Droz
November 1, 2019

f.

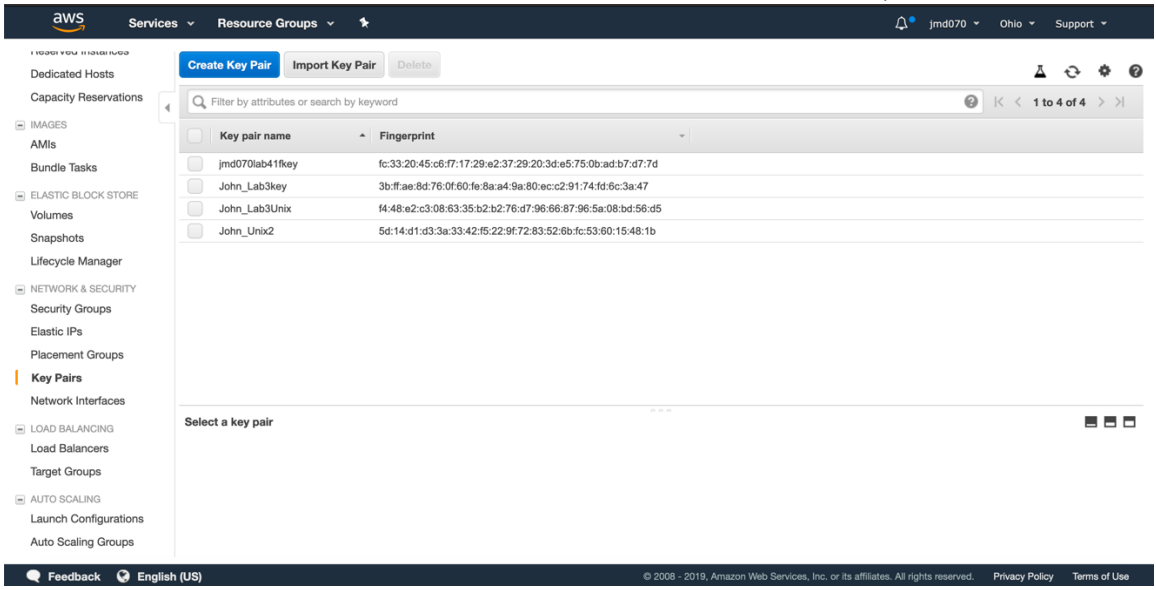
i. Create a new key pair

ii.


```
success {'KeyFingerprint': 'fc:33:20:45:c6:f7:17:29:e2:37:29:20:3d
:e5:75:0b:ad:b7:d7:7d', 'KeyMaterial': '-----
BEGIN RSA PRIVATE KEY-----
\nMIIEogIBAAKCAQEAs1Ry7qJ8RG6witc7MRajslN7TBC/DUphP29vIBLA7rvlgz7Z
1l3ycPwqNJRR\nOSWNgmoyISG0N6tAgWI+bJkZM2ejIF/d0UKRh5cxa53t1fqkHdMj
20pIhlyGrPhjjkuOzh+Qj44V\nXoLZsUu3riNZLiW7XWrJvX7xXagz0kYWAjf+pPwW
23v12YajCc+nzkwLjiTCAcDHicQ2AXI6rrDi\nIFtWGWCBYZbpLD+8/WAUGbG6hoc9
6sE1AYmpgfEgepuQYNHH5ChSEkdt/afC8o2XLcDy7Y8afxwS\n0Li5MsvWtKvyjBtW
beFx665Ve8yiRKVfJ935ihubcoAN+Ajevfu55QIDAQABAoIBAGt8NXsk9tjJ\ng5Vk
Vf0hNbKf6On5q5mauFkeQHv34Z32DyWwWT7shnEEI4PqVhirknrVjgBfxDYNthcWvJ
W42er+\ntj6kVz6i6LldfOik+zXjCdU6hWZdysSF1s77/if+z7RqJOp2XgaM5VAoMw
rz8vL++KOQayTiXmcl\ndxIUogJnbazwi2Dp4bZWFEl76yhxI4JIesioHggdZD/0LZ
3khhW0SKmEAz7ZsbbEastZZ5AK52sB\nO9TnpOMesHYYyb5l3apW+ZNX98iAStUQAk
iMqi+bXTeAPiAAPxYgMP8gln+nPM2oWXMcGrfGqYiZ\nMNQTvbzYqrqv/0bYILXFvA
uOTM0CgYEA6cjfRISqdNZaPmC6TkKYlW4lKTGLeTae/YStpR80d0QB\nQpYfmJYPy/
Why1tm9/d4d5rb9RB+1AcEqEjoNm2a0BrvhpS5zRYI69Dt1B73fJ6Q4UYGX0GEJaMT
\nX+0xGNyYDbfbxK28N5P5aS2zLaoRAKihWi+d8qKbMloExUQ2tCsCgYEAxF7kBcun
8/yIpwID2Nlq\ngkHV5K2zWX5gm+kjhAkRp1hK443pRLG2uYYqFb09UaSfSMlfIf8X
UkcG7u45DrVh6IPYO3gGAnui\nY1pSP9gSCsKPEGQXeO1TbJe0QKySUT2i2C+/gA2z
zfuY8WStN0hB/l2Ds4PlZJ/NI2x+75Tg8i8C\ngYA0lZcKnWCN2S0BPNaLBdW2Pskj
A97gS9XlJjO8oVE2xlgWNxgyRQaWlNBook191u8E0BNLgwNU\nA6QbihKZM3wP9Q8N
UWSdaahmCl+ER9bY8/2Yfhyuzli/8ndIK1zQv8V/yGwpR/6+7Yy7CKl1yvv6\nDvQE
ChPX4JVR9Qd/q4E2cwKBgBZbz46xX5AyQsTxP8rjBnOKV3ZQKMmZ0Bowc3DHL5FkRj
Kep97A\ndVOuaQBMqbywB8rb0s8mIlul5q4xxaFdGabFzEsJIXYK6bRyRkcqrPM4r9
1n6BxGuWb15vouLALA\nB9ihpyZHaR+c9A8YLXjL62ew8hvwep/4af0Mj29r/ijVAo
GAdFbpJbqILA321fbHNHvNx3gbBN6X\niOS+0/mP4uRWCP4qpHzPgH8syXgxjoFGJW
JvdA//Caekshri8OQZy96dGeAnD3buBB6LMYCRYID1\nRX7unX5ZP1L001R3wenTFn
bDuNxRBeyAQcZFN5IwHkT5wSiIiRf5WjmPtB777XReuiM=\n-----
END RSA PRIVATE KEY-----
', 'KeyName': 'jmd070lab41fkey', 'ResponseMetadata': {'RequestId':
 '6c891874-9014-4d00-8e55-
dea80c757d54', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '2039', 'date': 'Wed, 30 Oct 2019 15:40:03 GMT', 'server'
: 'AmazonEC2'}, 'RetryAttempts': 0}}
```


iii.

John Do
CSC 452-001
Dr. Droz
November 1, 2019

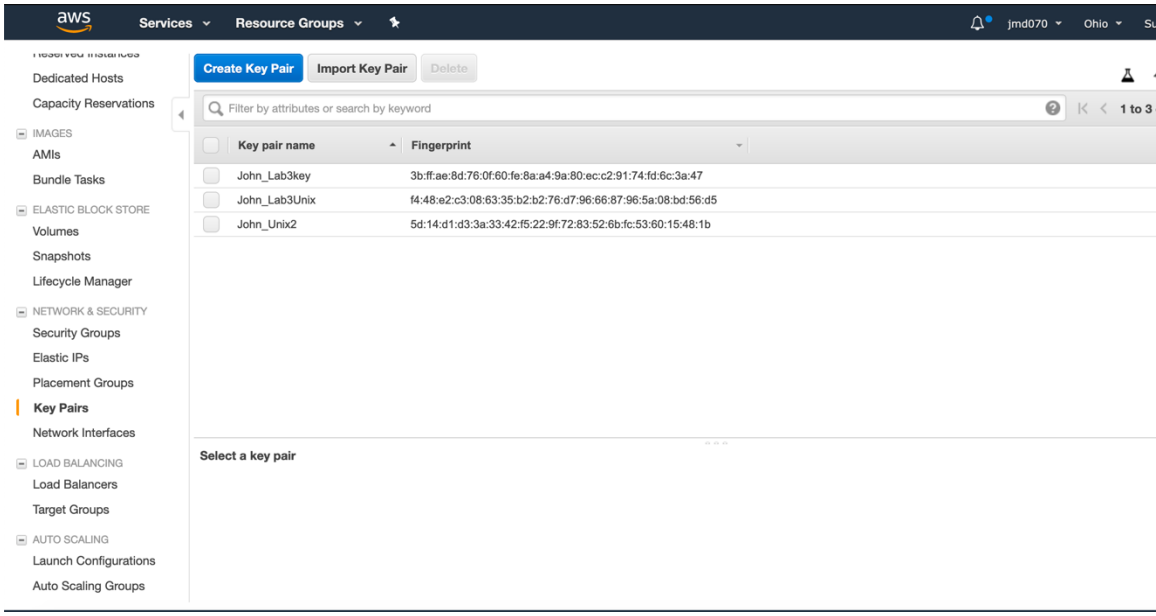


g.

i. Delete a key pair

ii.
```
success {'ResponseMetadata': {'RequestId': '43a7eb29-6b79-4f50-
9ed6-
99f5812655c7', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '227', 'date': 'Wed, 30 Oct 2019 15:44:00 GMT', 'server':
 'AmazonEC2'}, 'RetryAttempts': 0}}
```

iii.



h.

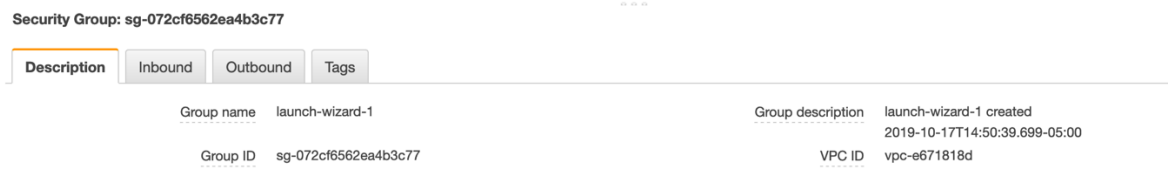
John Do
CSC 452-001
Dr. Droz
November 1, 2019

i. Get info about the security group

ii.

```
success {'SecurityGroups': [{'Description': 'launch-wizard-
1 created 2019-10-17T14:50:39.699-05:00', 'GroupName': 'launch-
wizard-
1', 'IpPermissions': [{'FromPort': 80, 'IpProtocol': 'tcp', 'IpRan
ges': [{'CidrIp': '0.0.0.0/0'}], 'Ipv6Ranges': [{'CidrIpv6': '::/0
'}], 'PrefixListIds': [], 'ToPort': 80, 'UserIdGroupPairs': []}, {
'FromPort': 22, 'IpProtocol': 'tcp', 'IpRanges': [{'CidrIp': '0.0.
0.0/0'}], 'Ipv6Ranges': [], 'PrefixListIds': [], 'ToPort': 22, 'Us
erIdGroupPairs': []}, {'FromPort': 3389, 'IpProtocol': 'tcp', 'IpR
anges': [{'CidrIp': '0.0.0.0/0'}], 'Ipv6Ranges': [], 'PrefixListId
s': [], 'ToPort': 3389, 'UserIdGroupPairs': []}], 'OwnerId': '8012
11634689', 'GroupId': 'sg-
072cf6562ea4b3c77', 'IpPermissionsEgress': [{'IpProtocol': '-
1', 'IpRanges': [{'CidrIp': '0.0.0.0/0'}], 'Ipv6Ranges': [], 'Pref
ixListIds': [], 'UserIdGroupPairs': []}], 'VpcId': 'vpc-
e671818d'}], 'ResponseMetadata': {'RequestId': 'e20060e1-1bfe-
4f5c-8d30-
eac9d7cc2616', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '2631', 'vary': 'accept-
encoding', 'date': 'Wed, 30 Oct 2019 15:46:38 GMT', 'server': 'Ama
zonEC2'}, 'RetryAttempts': 0}}
```


iii.



i.

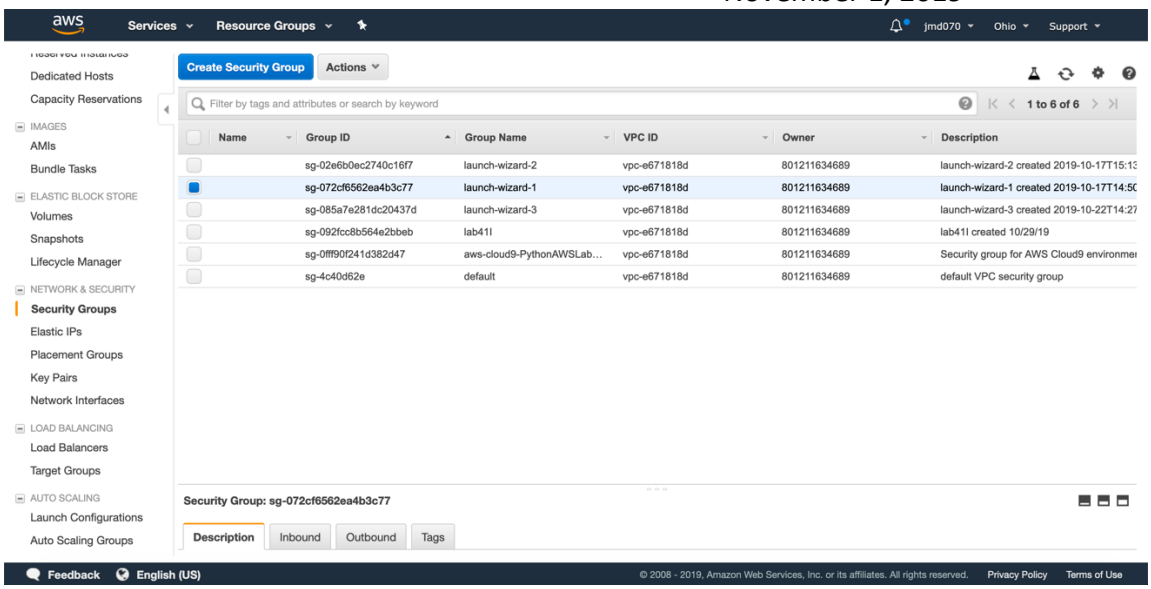
i. Create a security group to access an Amazon EC2 instance

ii.

```
success {'GroupId': 'sg-
092fcc8b564e2bbeb', 'ResponseMetadata': {'RequestId': '327b03cf-
bc3a-4393-b7e8-
b7b9efe94770', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '283', 'date': 'Wed, 30 Oct 2019 15:51:44 GMT', 'server':
 'AmazonEC2'}, 'RetryAttempts': 0}}
```


iii.

John Do
CSC 452-001
Dr. Droz
November 1, 2019

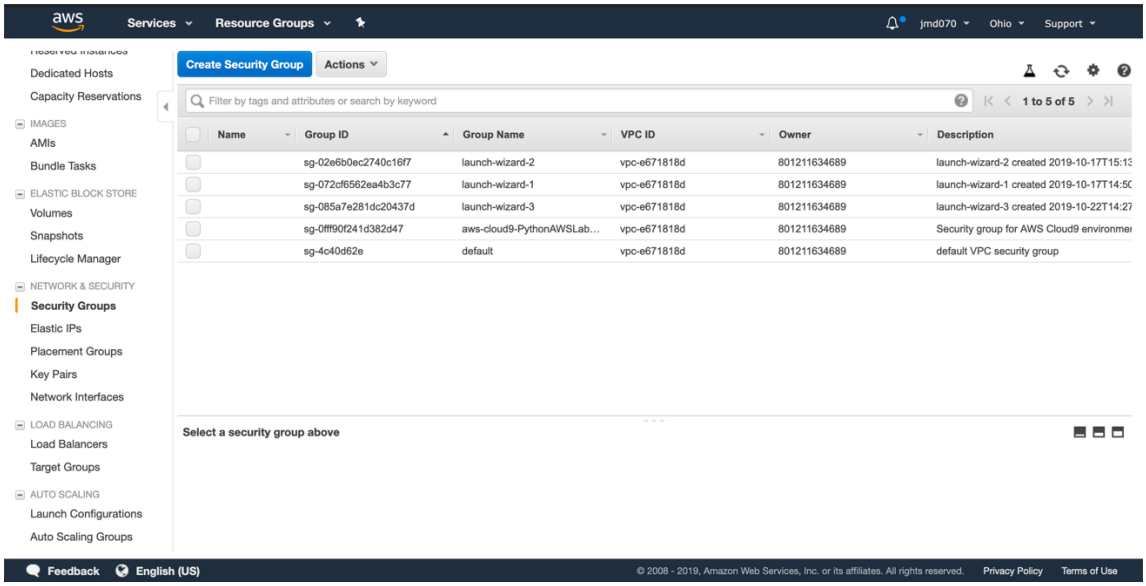


j.

i. Delete a security Group

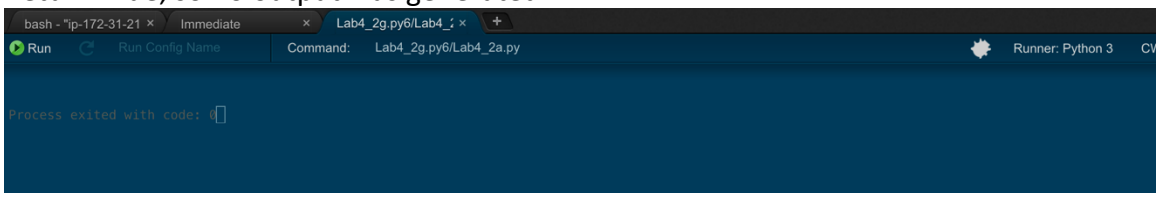
ii.
```
success {'ResponseMetadata': {'RequestId': '9a9bbfc6-5232-44c3-
af8d-
bac74089ccc5', 'HTTPStatusCode': 200, 'HTTPHeaders': {'content-
type': 'text/xml;charset=UTF-8', 'content-
length': '239', 'date': 'Wed, 30 Oct 2019 15:56:03 GMT', 'server':
 'AmazonEC2'}, 'RetryAttempts': 0}}
```

iii.

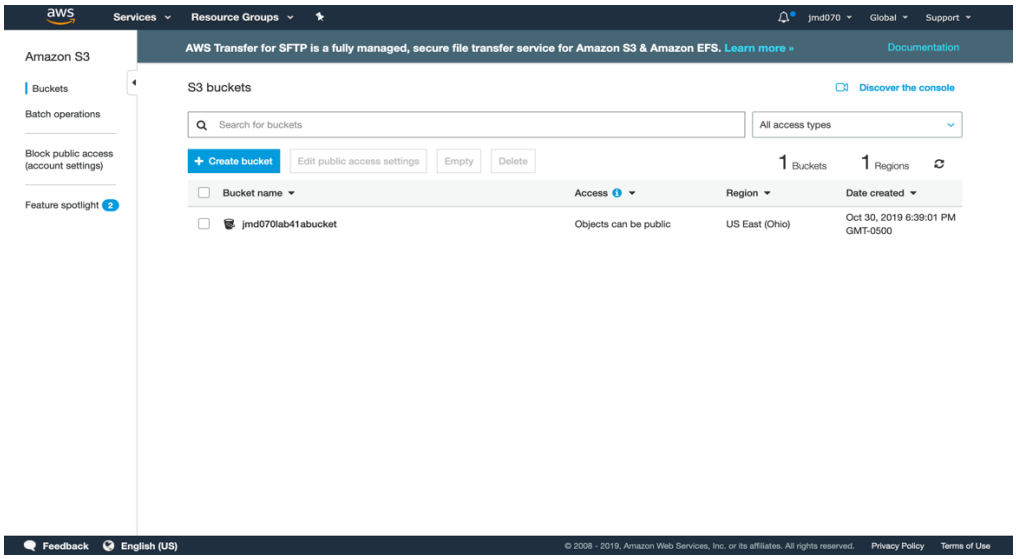


John Do
CSC 452-001
Dr. Droz
November 1, 2019

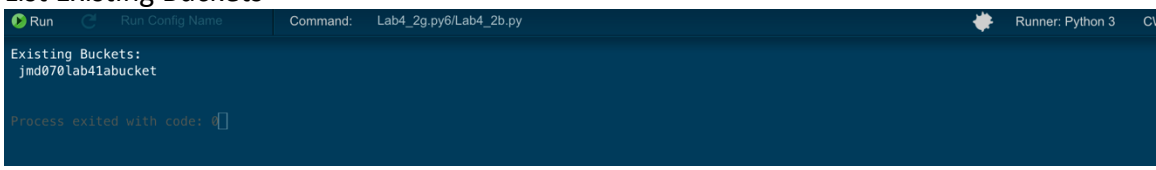
7. Write and Execute Python Boto3 programs that do the following using Amazon S3:
   a.
      i. Create an Amazon S3 Bucket
      ii. Return True, so no output was generated



      iii.



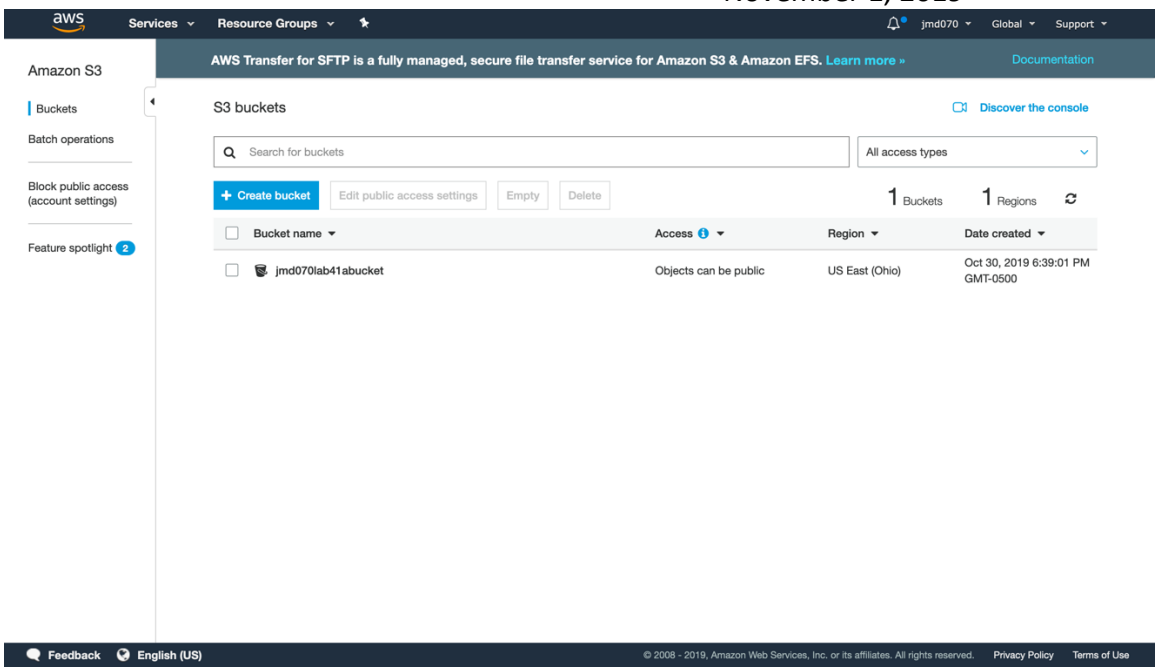
   b.
      i. List Existing Buckets



      ii.

John Do
CSC 452-001
Dr. Droz
November 1, 2019

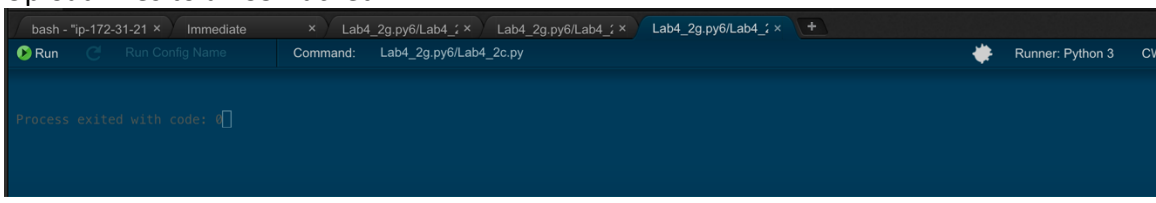
iii.



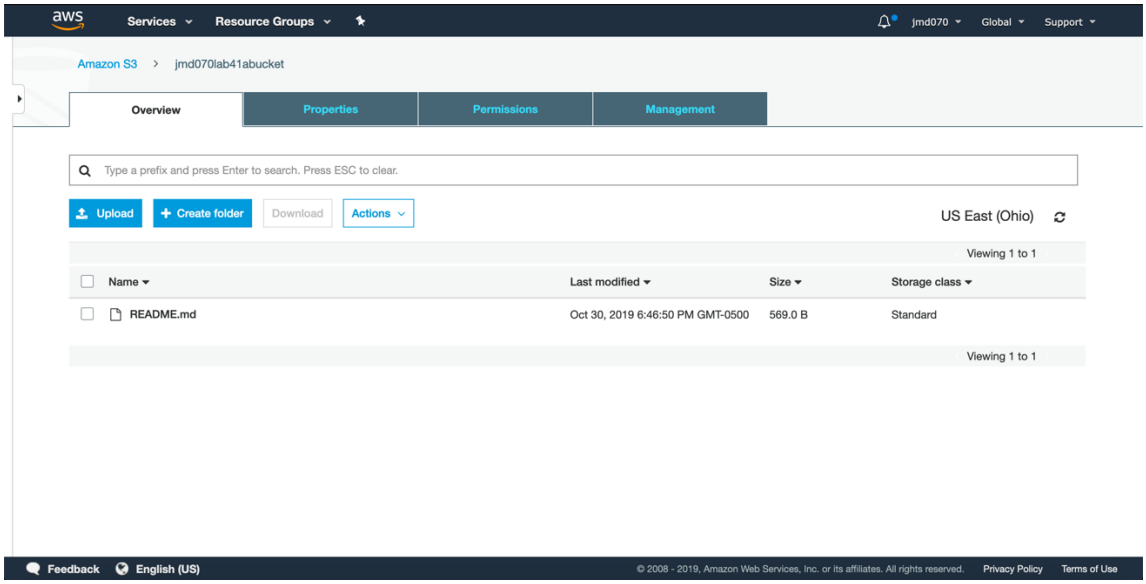
c.

i. Upload Files to an S3 Bucket

ii.



iii.

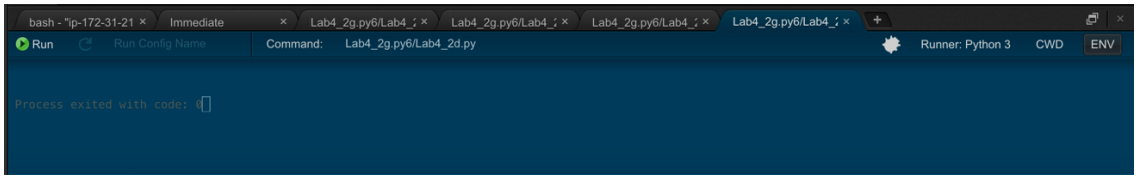


d.

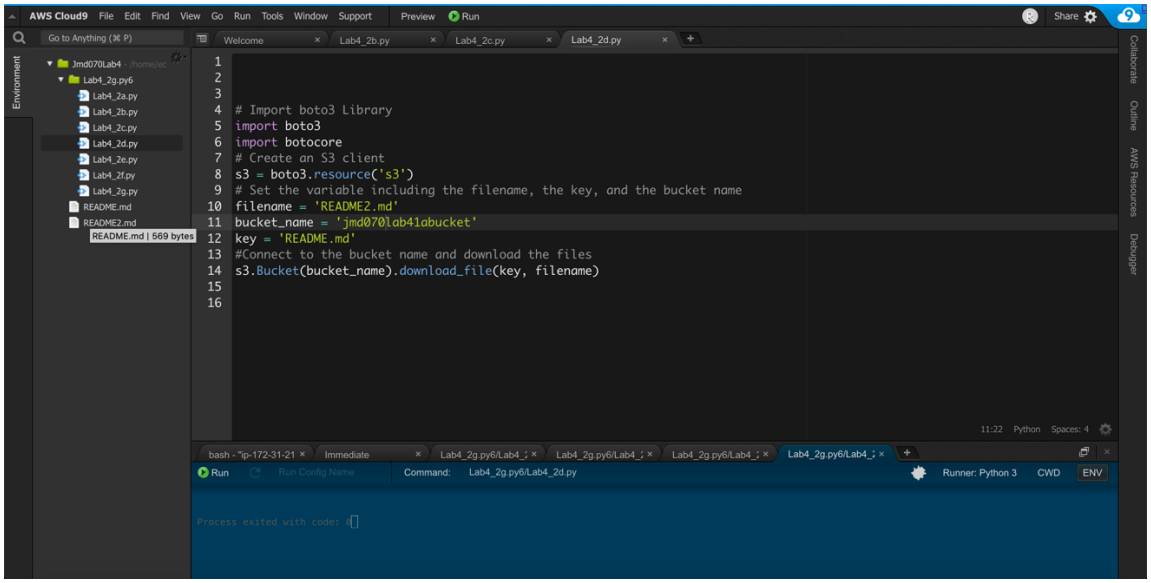
John Do
CSC 452-001
Dr. Droz
November 1, 2019

i. Download Files from an S3 Bucket

ii.



iii.



e.

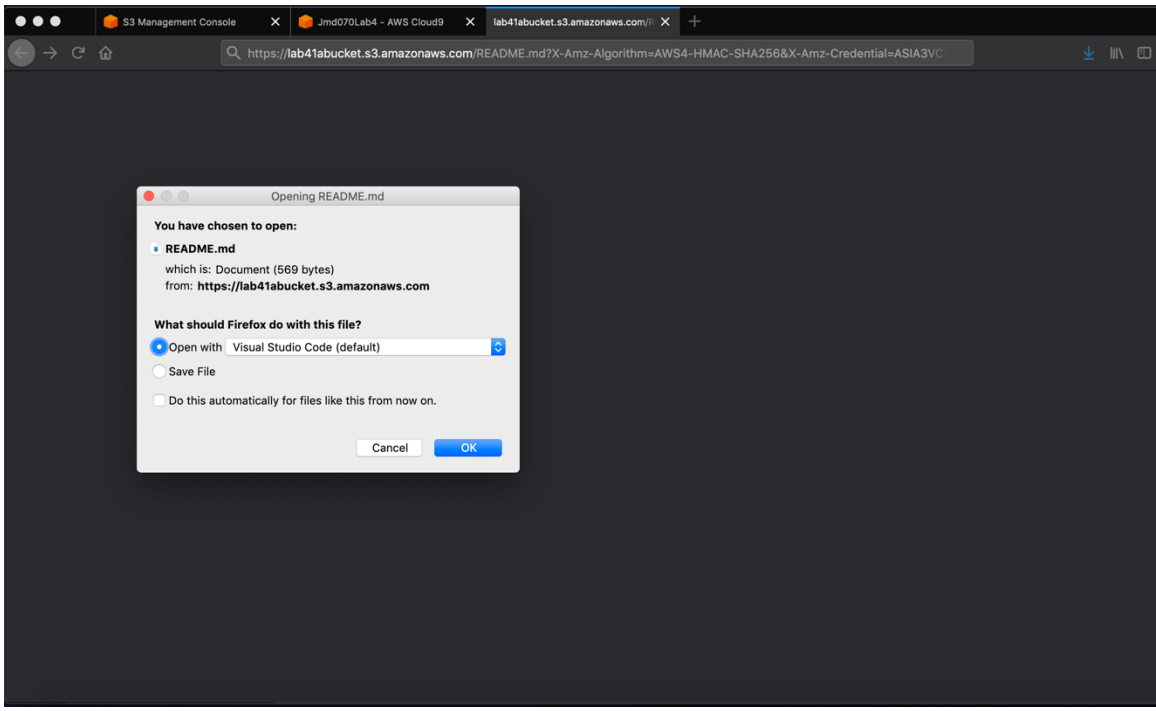
i. Grant temporary access using a presigned URL

ii. **This is the presigned URL:**
https://lab41abucket.s3.amazonaws.com/README.md?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=ASIA3VC7PKQA5GW65XZT%2F20191030%2Fus-east-2%2Fs3%2Faws4_request&X-Amz-Date=20191030T235850Z&X-Amz-Expires=3600&X-Amz-SignedHeaders=host&X-Amz-Security-Token=IQoJb3JpZ2luX2VjEMD%2F%2F%2F%2F%2F%2F%2F%2F%2F%2FwEaCXVzLWVhc3QtMiJHMEUCIQCBXionpp8zmoP6drLNZgCUbeKPWRp6FKMJmBrs2%2FZSgQIgDrRoqlPJOgv0wg158vM9FptMBdGm8hqiFdPRrWASwvcqwgIIyf%2F%2F%2F%2F%2F%2F%2F%2F%2F%2F%2FARAAGgw4MDEyMTE2MzQ2ODkiDNchTracsBYj7UXTvCqWAoz2kWGZJjiWPHBZLo7D7D60zaL4pZN1XDJ0JzI7T6Lv%2F9X%2Fo5VULFXsqb%2FjqCJAXShoR9DL6Sl02i2GK%2BfCjvPWHTQ%2BB%2Fqr8lYa7fMXMjV56APMT%2BE%2F9PyGWcbmz21P9J42%2BuEyKy2DYI3lTdjggZsV7rjMP%2B57ijOPyS%2FzL8c1d9aS1ooHm2xJqCqZUfrlNDDJyERvUBHerB5mRgPJuVYgnu6%2FNlORK0qiUS3k8wUduQyXcTEY4f%2Fx24KVI%2BeeybuleXqlrxQuQ6A39P9iSAYuvPYiRc%2BPDcr8vBPD2Og90w9WHoz5s8EAJLj9IoNr%2BoUEtyhYSiBiXkjDMmJfFO%2FGIwVi%2BMKr%2BO0wXfdJ%2BL5Auxwpes0cMNzV5u0FOusCgYJvsTVAX%2FgsJgs

John Do
CSC 452-001
Dr. Droz
November 1, 2019

hMJohUXEAk76Pzzvv2HJTfRqQrba7rA%2BKROEUDqbrleVdKprgLoAONr%2F7larrR1dmfOwGrfALOQSb8xczj0FPVqanYHEw9i7w7KpoAMNJUPXcb1znT6l3l1WXsDM1ZzuH%2FseAWgycDBZh6qwIgx55Gv3FDWTpnrbJHdfeVZhfyFpqzs5OhUDHbWBbh2cnlvmKIprS1%2BTPxgyWUYnGZg461YCXRQkNFKg6slLmLdl1wS6BXvXLtoWeZPlQSByS5aiTEA8RXRDZ7%2FIBbuY0Rs%2B7pZ%2BWcZ3nVhsihOKExifJN3OvO%2BoJI4QuDr3Ej9V2RRm4EIOkUa%2B1CBuDVinmpGndeOsfVhd9gcHEBE1OkHprFsPRSCgEZ%2BYLmBP5WzITWe3U56bCAhXia7hE2vgQM0nqKhiXVw%2B4V4D0cdt3DjbHpgERJW3B09PIc7dTd9pdW1%2F36GBh4qfk8D3GsxVu9M86&X-Amz-Signature=b2407d1de47d63af99f67fbfa78f317c7757e8970b632543fdcc88d4364b8bbc

iii.



f.

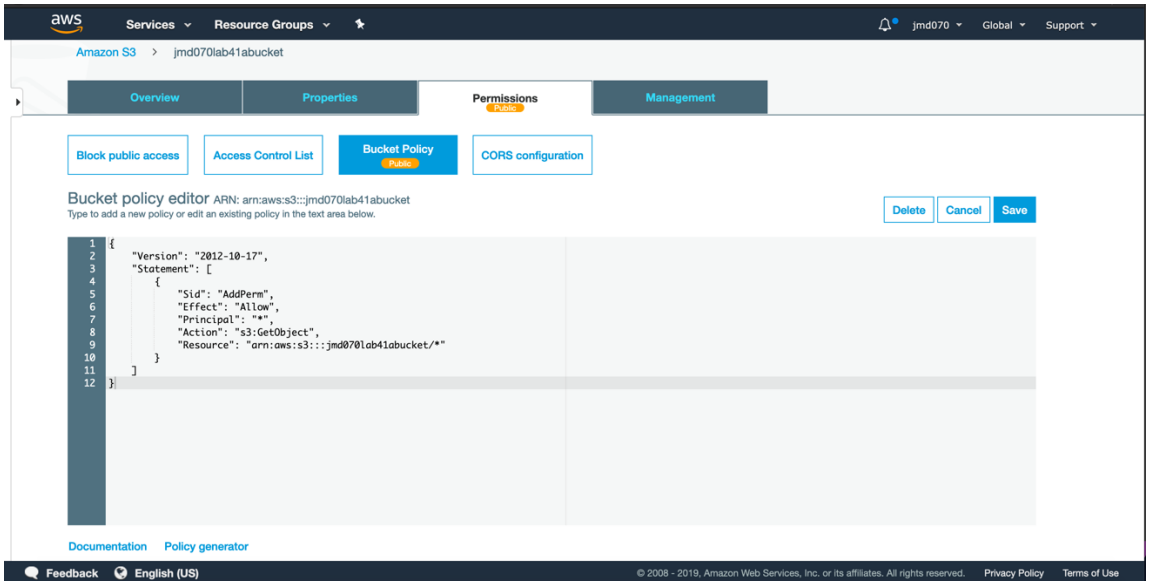
i. Retrieve a bucket policy

ii.

```
{"Version":"2012-10-17","Statement":[{"Sid":"AddPerm","Effect":"Allow","Principal":"*","Action":"s3:GetObject","Resource":"arn:aws:s3:::jmd070lab41abucket/*"}]}
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019

iii.



g.

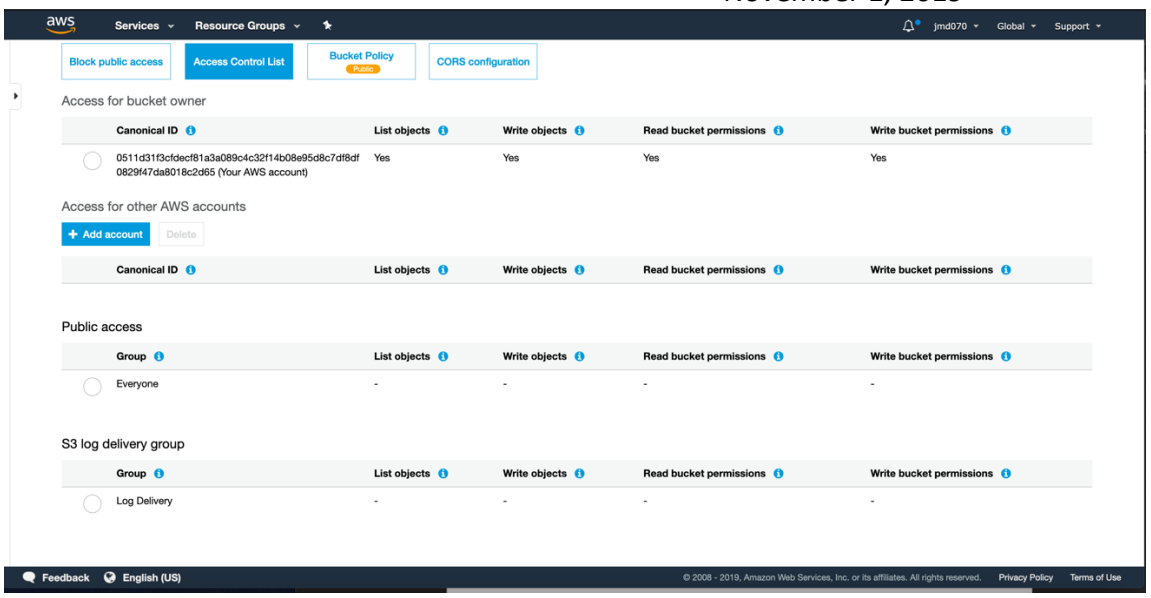
i. Get a Bucket Access Control List

ii.

```
{'ResponseMetadata': {'RequestId': '28565A3384081CD0', 'HostId': '
/Vd6/Y+OmHEBOQRFk3cVI6eODg0m+H5utMFIAMqH03A+Z1LOFN+o+5GajYW/qpU8/9
mx8u9dyg0=', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amz-id-
2': '/Vd6/Y+OmHEBOQRFk3cVI6eODg0m+H5utMFIAMqH03A+Z1LOFN+o+5GajYW/q
pU8/9mx8u9dyg0=', 'x-amz-request-
id': '28565A3384081CD0', 'date': 'Thu, 31 Oct 2019 00:09:14 GMT',
'content-type': 'application/xml', 'transfer-
encoding': 'chunked', 'server': 'AmazonS3'}, 'RetryAttempts': 0},
'Owner': {'ID': '0511d31f3cfdecf81a3a089c4c32f14b08e95d8c7df8df082
9f47da8018c2d65'}, 'Grants': [{'Grantee': {'ID': '0511d31f3cfdecf8
1a3a089c4c32f14b08e95d8c7df8df0829f47da8018c2d65', 'Type': 'Canoni
calUser'}, 'Permission': 'FULL_CONTROL'}]}
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019

iii.

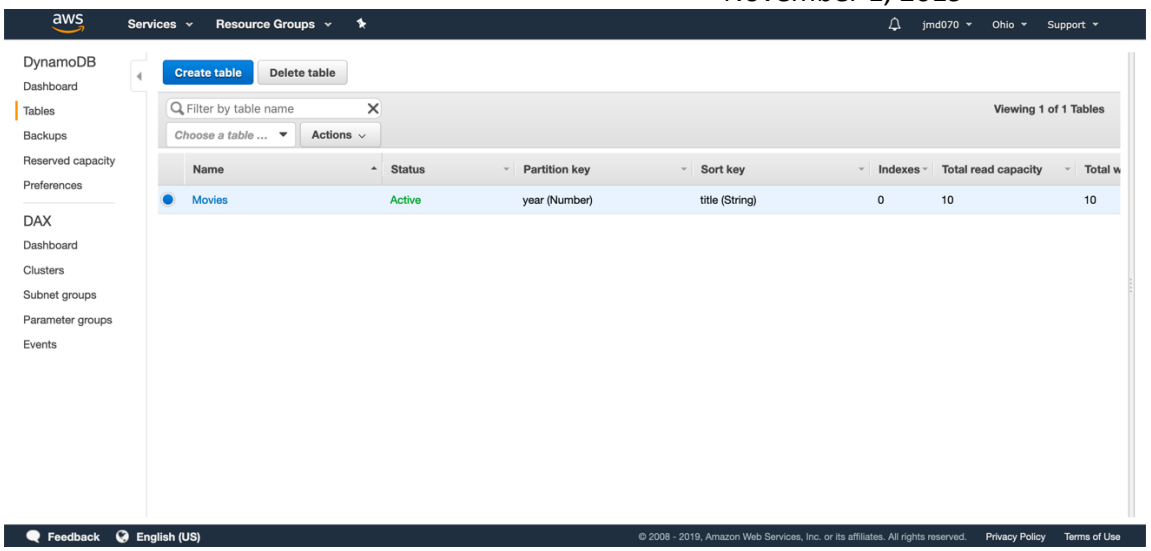


8. (**Extra Credit**, max +2 points on your final grade) Write and execute **Python Boto3** programs that do the following using **Amazon DynamoDB**:
   a.
      i. Create a new table

ii.
```
{'TableDescription': {'AttributeDefinitions': [{'AttributeName': '
title', 'AttributeType': 'S'}, {'AttributeName': 'year', 'Attribut
eType': 'N'}], 'TableName': 'Movies', 'KeySchema': [{'AttributeNam
e': 'year', 'KeyType': 'HASH'}, {'AttributeName': 'title', 'KeyTyp
e': 'RANGE'}], 'TableStatus': 'CREATING', 'CreationDateTime': date
time.datetime(2019, 11, 1, 13, 16, 26, 7000, tzinfo=tzlocal()), 'P
rovisionedThroughput': {'NumberOfDecreasesToday': 0, 'ReadCapacity
Units': 10, 'WriteCapacityUnits': 10}, 'TableSizeBytes': 0, 'ItemC
ount': 0, 'TableArn': 'arn:aws:dynamodb:us-east-
2:801211634689:table/Movies', 'TableId': '47a3a42a-593f-4388-aa96-
8c437777d24f'}, 'ResponseMetadata': {'RequestId': 'CBC5U8VP9MR31VK
LA6SP3JNPR3VV4KQNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode': 200, 'HT
TPHeaders': {'server': 'Server', 'date': 'Fri, 01 Nov 2019 13:16:2
5 GMT', 'content-type': 'application/x-amz-json-1.0', 'content-
length': '568', 'connection': 'keep-alive', 'x-amzn-
requestid': 'CBC5U8VP9MR31VKLA6SP3JNPR3VV4KQNSO5AEMVJF66Q9ASUAAJG'
, 'x-amz-crc32': '1456136828'}, 'RetryAttempts': 0}}
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019

iii.



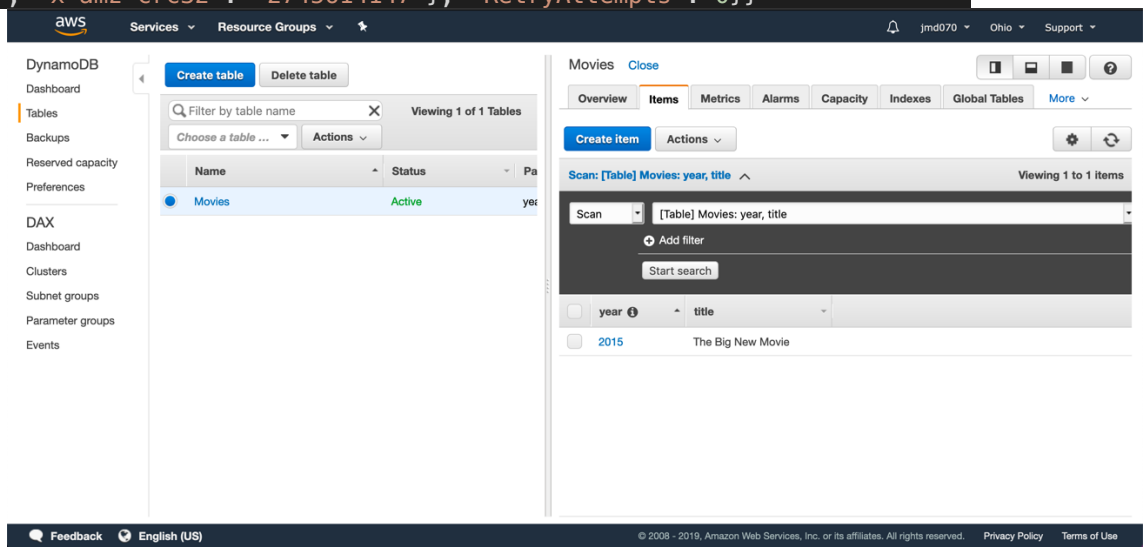
b.

i. Create a new item

ii.

```
{'ResponseMetadata': {'RequestId': '9P5CLCMPE1DSF1K896U90R7H1VVV4K
QNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode': 200, 'HTTPHeaders': {'s
erver': 'Server', 'date': 'Fri, 01 Nov 2019 13:18:22 GMT', 'conten
t-type': 'application/x-amz-json-1.0', 'content-
length': '2', 'connection': 'keep-alive', 'x-amzn-
requestid': '9P5CLCMPE1DSF1K896U90R7H1VVV4KQNSO5AEMVJF66Q9ASUAAJG'
, 'x-amz-crc32': '2745614147'}, 'RetryAttempts': 0}}
```

iii.



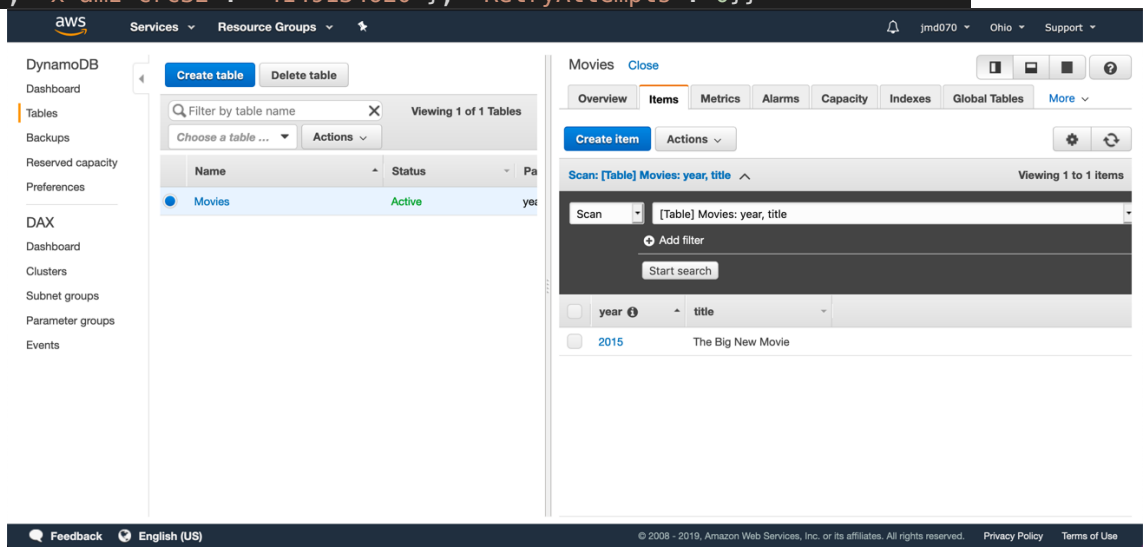
c.

i. Get (retrieve) an item

John Do
CSC 452-001
Dr. Droz
November 1, 2019

ii.
```
{'Item': {'title': {'S': 'The Big New Movie'}}, 'ResponseMetadata'
: {'RequestId': '64NPO5TDGNROBV92R3H2JQOS4RVV4KQNSO5AEMVJF66Q9ASUA
AJG', 'HTTPStatusCode': 200, 'HTTPHeaders': {'server': 'Server', '
date': 'Fri, 01 Nov 2019 13:20:04 GMT', 'content-
type': 'application/x-amz-json-1.0', 'content-
length': '44', 'connection': 'keep-alive', 'x-amzn-
requestid': '64NPO5TDGNROBV92R3H2JQOS4RVV4KQNSO5AEMVJF66Q9ASUAAJG'
, 'x-amz-crc32': '4149134620'}, 'RetryAttempts': 0}}
```

iii.



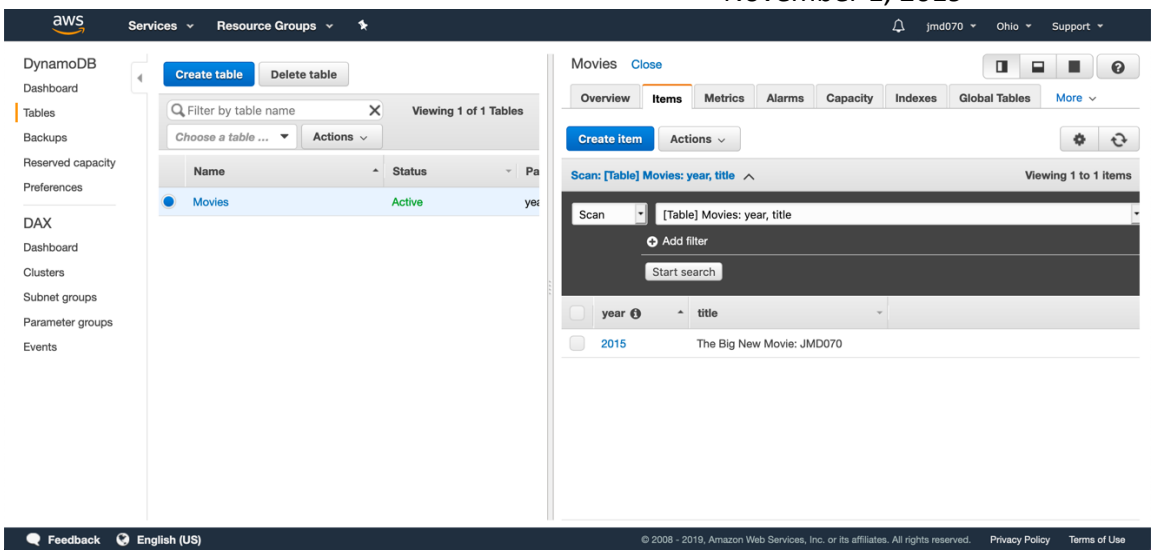
d.

i. Update attributes of the item

ii.
```
{'ResponseMetadata': {'RequestId': 'ODN666S8K1ND071BM9V2CELQVRVV4K
QNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode': 200, 'HTTPHeaders': {'s
erver': 'Server', 'date': 'Fri, 01 Nov 2019 13:20:54 GMT', 'conten
t-type': 'application/x-amz-json-1.0', 'content-
length': '2', 'connection': 'keep-alive', 'x-amzn-
requestid': 'ODN666S8K1ND071BM9V2CELQVRVV4KQNSO5AEMVJF66Q9ASUAAJG'
, 'x-amz-crc32': '2745614147'}, 'RetryAttempts': 0}}
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019

iii.



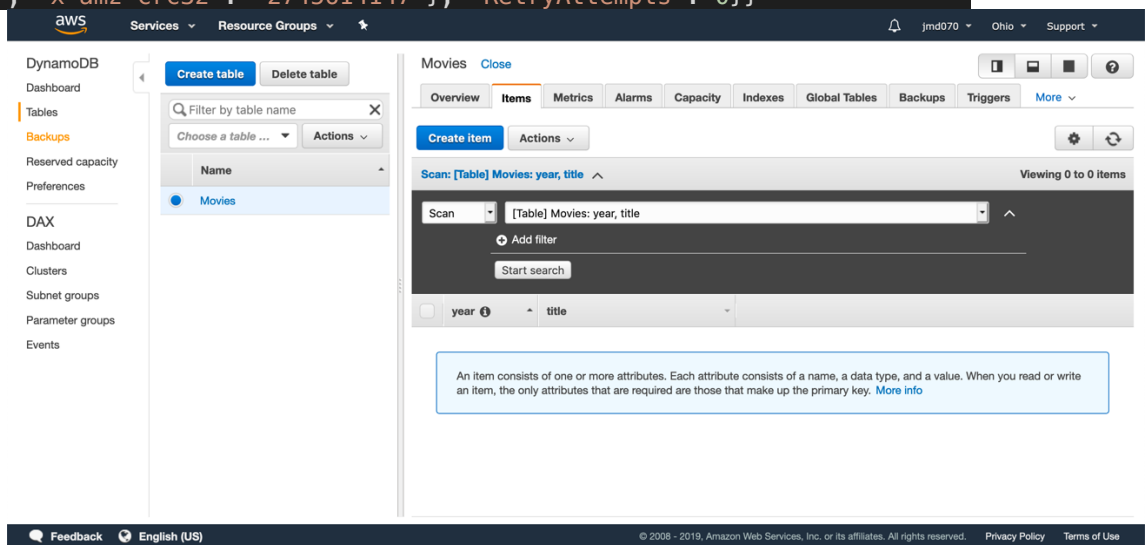
e.

i. Delete an item

ii.

```
{'ResponseMetadata': {'RequestId': 'N21R5ODA2605J06PDTRA5JN8GJVV4K
QNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode': 200, 'HTTPHeaders': {'s
erver': 'Server', 'date': 'Fri, 01 Nov 2019 13:24:50 GMT', 'conten
t-type': 'application/x-amz-json-1.0', 'content-
length': '2', 'connection': 'keep-alive', 'x-amzn-
requestid': 'N21R5ODA2605J06PDTRA5JN8GJVV4KQNSO5AEMVJF66Q9ASUAAJG'
, 'x-amz-crc32': '2745614147'}, 'RetryAttempts': 0}}
```

iii.



f.

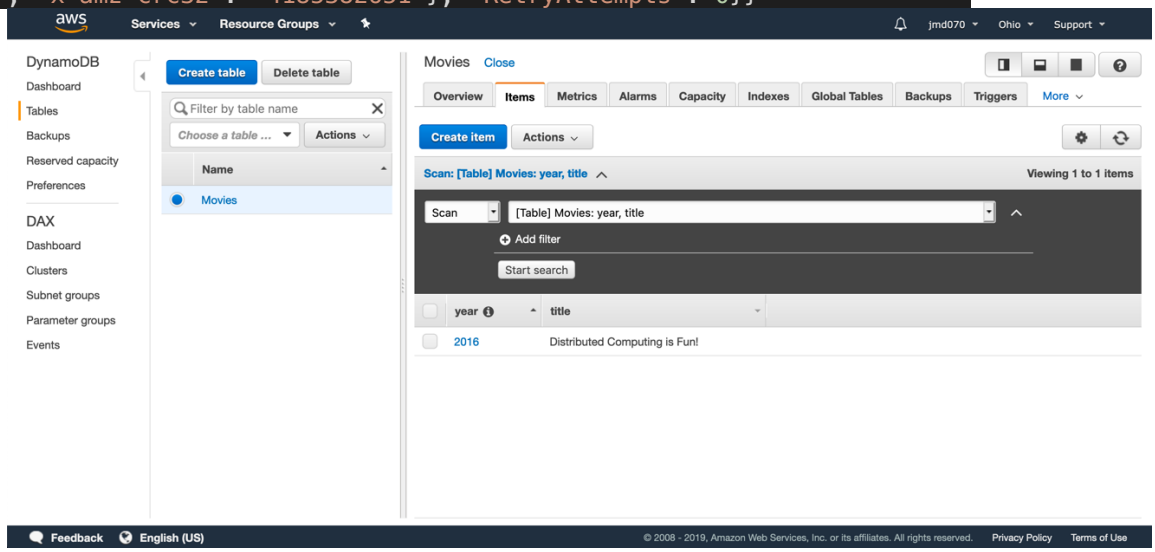
i. Batch writes to a table

John Do
CSC 452-001
Dr. Droz
November 1, 2019

ii.

```
{'UnprocessedItems': {}, 'ResponseMetadata': {'RequestId': 'U4GTFF
VDS5O1A0935MSF9I8ARJVV4KQNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode':
 200, 'HTTPHeaders': {'server': 'Server', 'date': 'Fri, 01 Nov 201
9 13:26:17 GMT', 'content-type': 'application/x-amz-json-
1.0', 'content-length': '23', 'connection': 'keep-alive', 'x-amzn-
requestid': 'U4GTFFVDS5O1A0935MSF9I8ARJVV4KQNSO5AEMVJF66Q9ASUAAJG'
, 'x-amz-crc32': '4185382651'}, 'RetryAttempts': 0}}
```

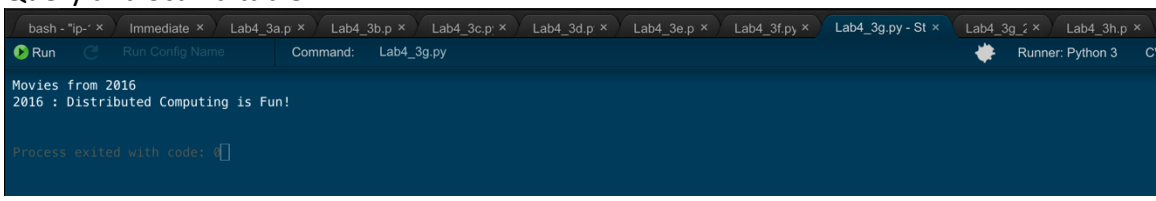
iii.



g.

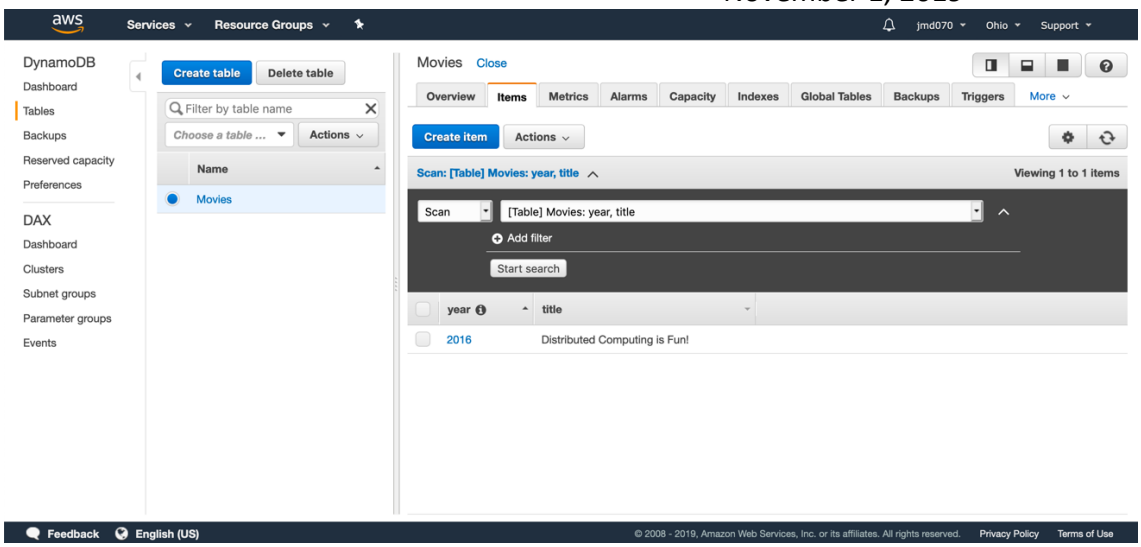
i. Query and Scan a table

ii.

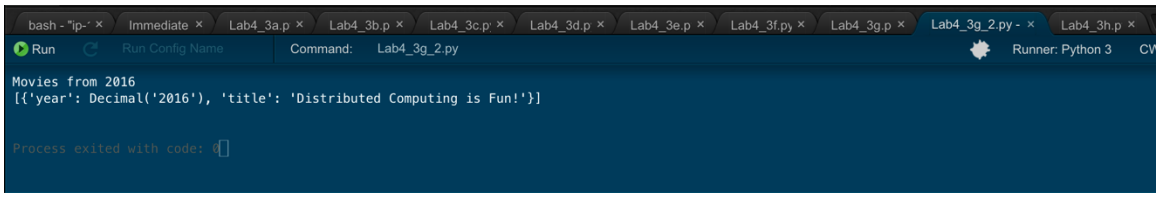


John Do
CSC 452-001
Dr. Droz
November 1, 2019

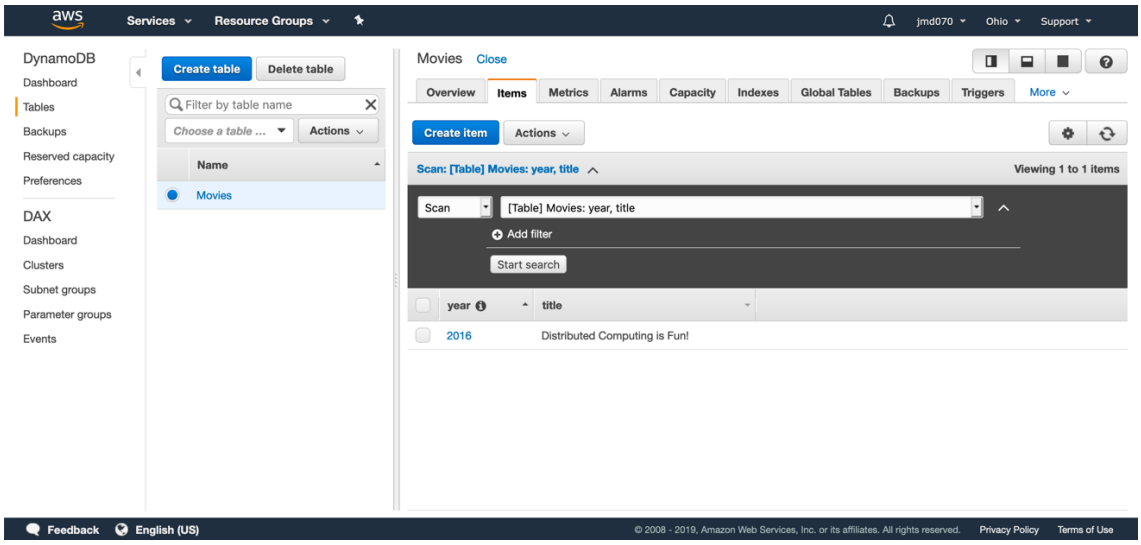
iii.



iv.



v.



h.

i. Delete a table

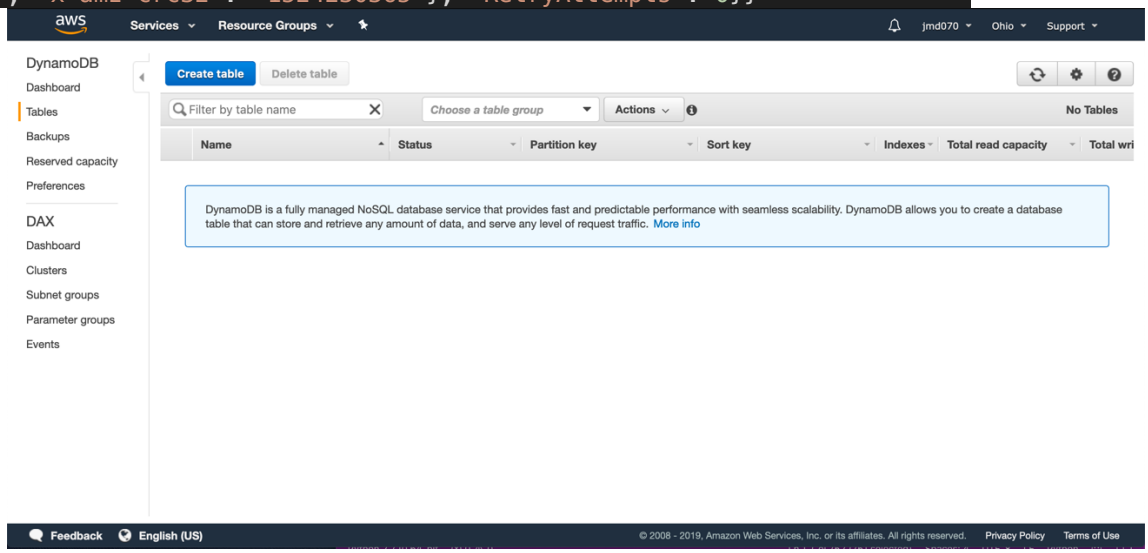
ii.

```
{'TableDescription': {'TableName': 'Movies', 'TableStatus': 'DELET
ING', 'ProvisionedThroughput': {'NumberOfDecreasesToday': 0, 'Read
CapacityUnits': 10, 'WriteCapacityUnits': 10}, 'TableSizeBytes': 0
```

John Do
CSC 452-001
Dr. Droz
November 1, 2019

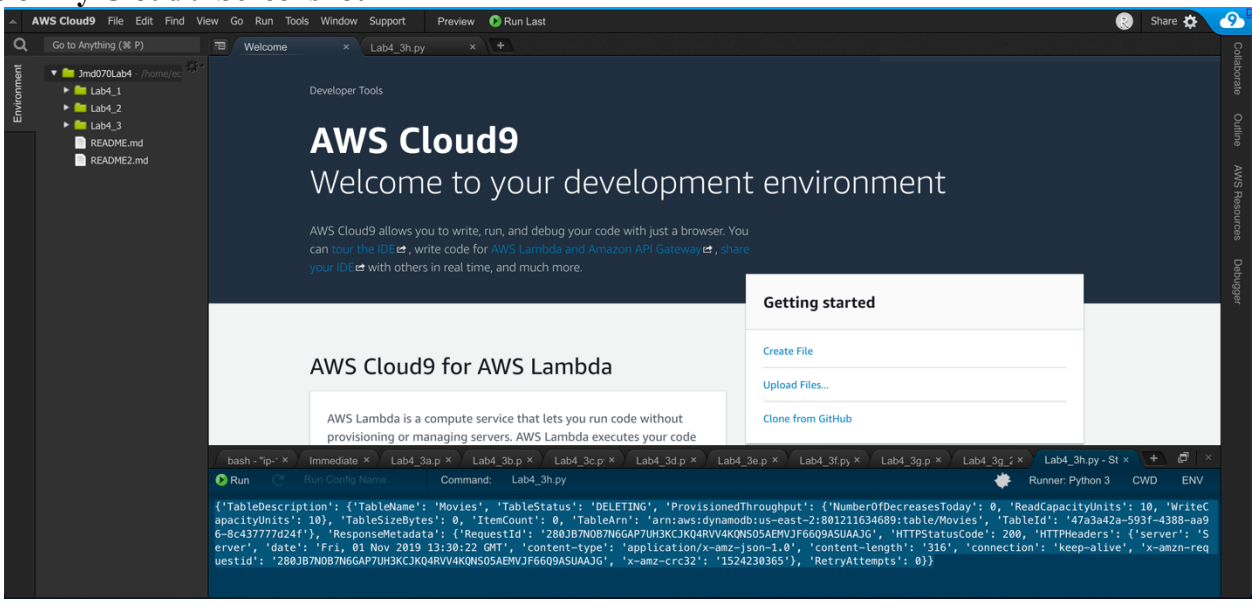

```
, 'ItemCount': 0, 'TableArn': 'arn:aws:dynamodb:us-east-
2:801211634689:table/Movies', 'TableId': '47a3a42a-593f-4388-aa96-
8c437777d24f'}, 'ResponseMetadata': {'RequestId': '280JB7NOB7N6GAP
7UH3KCJKQ4RVV4KQNSO5AEMVJF66Q9ASUAAJG', 'HTTPStatusCode': 200, 'HT
TPHeaders': {'server': 'Server', 'date': 'Fri, 01 Nov 2019 13:30:2
2 GMT', 'content-type': 'application/x-amz-json-1.0', 'content-
length': '316', 'connection': 'keep-alive', 'x-amzn-
requestid': '280JB7NOB7N6GAP7UH3KCJKQ4RVV4KQNSO5AEMVJF66Q9ASUAAJG'
, 'x-amz-crc32': '1524230365'}, 'RetryAttempts': 0}}
```


iii.



9. Picture of my Cloud 9 Screenshot

a.



IV. Conclusions:

John Do
CSC 452-001
Dr. Droz
November 1, 2019

From the results being shown, each experiment was creating a new environment within the cloud 9 service. AWS Cloud 9 allows you to write, run, and debug your code with just a browser. With Cloud 9, you have immediate access to a rich code editor, integrated debugger, and built-in terminal with preconfigured AWS CLI (Command Line Interface). Within the Cloud 9 editor, we are creating a new environment and writing and executing python boto3 programs with the use of EC2 service and S3 storage. As a class, we wrote programs that can help interact with the EC3 and S3 service with a few functions from python. For example, with the EC2 service, we are messing with the instances such as getting basic information to detailed information, starting, stopping, and rebooting. An addition, we are doing the same with key pairs and security groups. We wrote another python program that interacts with buckets within Amazon S3 such as creating, listing, uploading files, downloading files, grant temporary access, retrieving a bucket policy, and getting the bucket access control list. Each of these labs explains the use of executable programs to be involve with cloud services such as AWS. AWS is an infrastructure-as-a-service, which is pretty much giving us the foundation and resources to allow us to deploy applications, instances, etc. The following program creates a table and a new item. It gets the information from the table when we specify it. Whenever we want to change a certain information, we can update the attribute. The program deletes an item when calling upon, and we can batch write puts one or multiple information in one or more tables. Some programs can query the information from the table when using the right query. We can also scan for the specified information that the user wants. When the table is unneeded, we can delete it using the program as well. This will be beneficial on working with executable programs in general because it helps us understand how the interaction works between the service and the program. The different libraries and environments that can be configured to help set up the connection between them. The programs do not have to be python such as this experiment, it could be any other programming languages that could write efficient code with varieties of functions from the libraries. This will help us during future tasks of using the EC2 server and S3 when communicating with the service using executable programs written from a programming language.

GITHUB LINK:
https://github.com/jonathanmdo/CSC_452_Labs/tree/Lab4