Python script with AWS and Azure

* Install Python 3.6.x from python.org
* $ pip install python3
* $ python --version
* Install pip (get-pip.py file is to be downloaded from github and run python get-pip.py)
* $ pip –V to check the version of pip
* Install boto3 and awscli modules for python
* $ pip install boto3
* $ pip install awscli ( AWS command line interface)
* Install pyCharm IDE

AWS Management

* Visit aws.amazon.com
* Create new account to get AccessId and secret key
* Create new user account with “Enable programmatic access” feature by going to IAM (Identity Access Manager)
* Access Key ID : AKIAIRU2GJZ4WNLKCWNQ
* Secret access key: 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
* You can also download credentials csv
* C:\> aws configure
  + AWS access ley : xxxxxx
  + Secret key : xxxxx
  + Default region name : press enter (default)
  + Default output format : Press enter
* C:\>python
* >>> import boto3
* >>> print(boto3)
* >>> s3= boto3.resource(‘s3’)
* >>> for bucket in s3.buckets.all():
* … print(bucket.name)

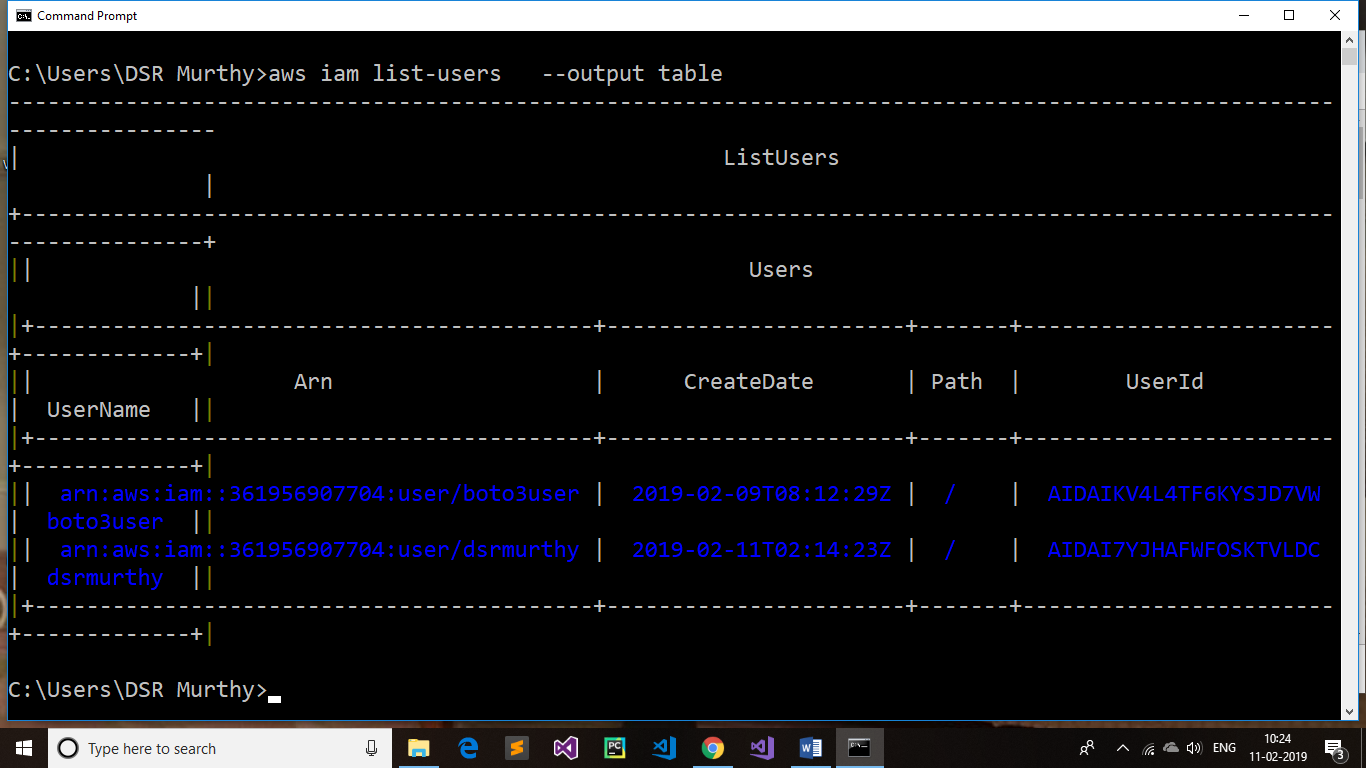
Outut:

* + dsrbucket
  + elasticbeanstalk-ap-south-1928343
* To see the buckets in AWS , click Services 🡪 Storage 🡪 S3🡪 see your buckets.
* You can create Bucket by adding bucket name, Select region and add settings
  + Note: Bucket name should be globally unique (So use UUID)
  + You can upload new images, videos , projects etc and get url
    - s3://dsrtestbucket/passport photo.jpg

Session2 : Working with awscli

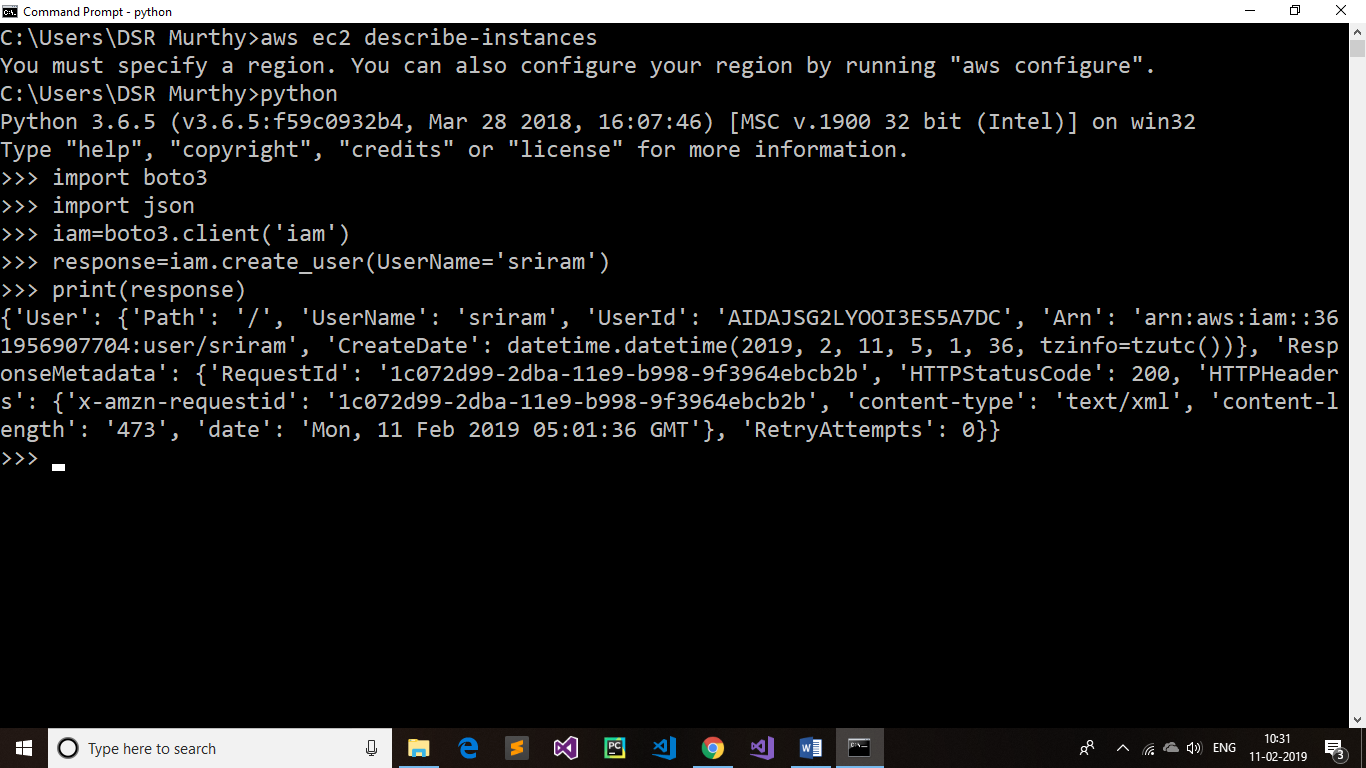
* c:\> pip install - -upgrade awscli (to upgrade)
* c:\> aws - -version (Check the aws cli version)

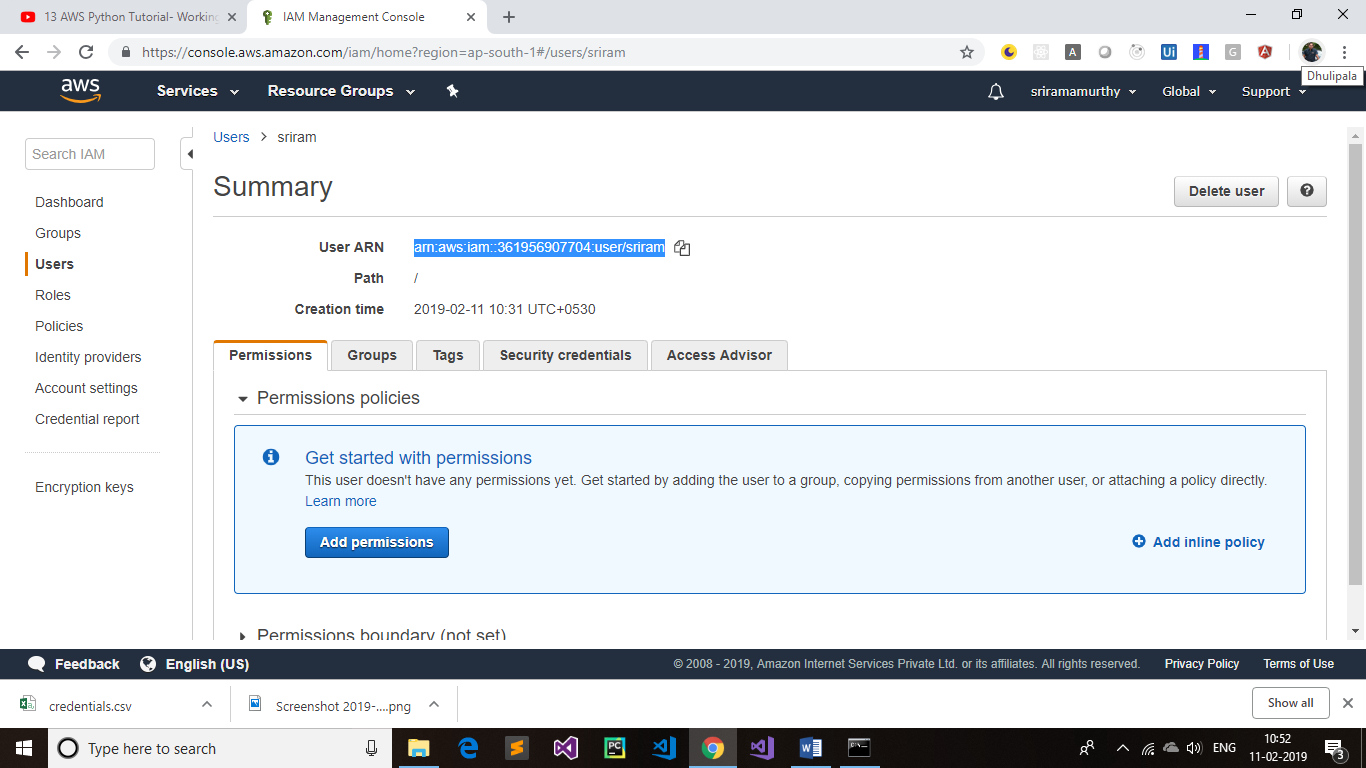
aws configure command:

* used to set access
* c:\> aws s3 ls (For listing all buckets in s3 of aws)
* c:\> aws iam list-users --output table ( list all uses in table format)
* 
* C:\>aws ec2 describe-instances
* Check EC2 instances in aws else start any instance

Session 3 : Creating new user from boto3 and attaching policy (permissions)

* Start python instance
* >>> import boto3
* >>> import json
* >>> iam= boto3.client(‘iam’)
* >>> response=iam.create\_user(UserName=”sriram”)
* >>> print (response)



* Now attach policy (grant permission) to access service
* Find source-ARN in Home🡪 users🡪 dbl. click on sriram 🡪 copy source-urn
* 
* >>>RESOURCE\_ARN=”arn:aws:iam::361956907704:user/sriram”
* >>>policy={ “Version”:”2019-2-12”,”Statement”: [

{ “Effect”:”Allow”,”Action”:”logs:CreateLogGroup”,

“Resource”:RESOURCE\_ARN},

{“Effect”:”Allow”,”Action”:[

“Dynamodb:DeleteItem”,”Dynamodb:GetItem”,”Dynamodb:PutItem”],

“Resource”:RESOURCE\_ARN}]}

>>>print (policy)

>>> response=

iam.create\_policy(PolicyName=”mydynamodbpolicy”,PolicyDocument=json.dumps(policy))

>>> policy.attach() or policy,detach() to attach/detach policy)

Session 4: Managing uses with iam

>>> import boto3

>>> iam=boto3.client(‘iam’)

>>>response= iam.create\_user(UserName=’laltha’)

>>> print(response)

Open AWS management console🡪 Users 🡪 refresh to see new user created

To List the users

>>>user=iam.get\_paginator(‘list\_users’)

>>> print(iam.paginate())

>>> for usr in user.paginate():

… print(usr)

>>>iam.update\_user(UserName=’lalitha’,NewUserName=’lalithasri’)

>>> iam.delete\_user(UserName=’lalitha’)

Visit users console in AWS and observe users by clicking refresh.

Sesson 5: Creating access key with iam

>>> import boto3

>>> iam=boto3.client(‘iam’)

>>> response=iam.create\_access\_key(UserName=’sriram’)

>>> print (response[‘AccessKey’])

Observe Access key and secret key on console output. Now we can use this in aws to programmatically access resources on AWS.

Session 6: Elastic public IP address Management (creating and releasing)

>>> import boto3

>>> ec2=boto3.client(‘ec2’)

>>> print(ec2)

>>> filters= [ { ‘Name’:’domain’,’Values’:[‘vpc’]}]

>>> response=ec2.describe\_addresses(Filters=filters)

>>> print (response)

>>> allocation=ec2.allocate\_address(Domain=’vpc’)

>>>print (allocation)

Now goto aws🡪 ec2 🡪 instances 🡪 click running instances🡪 copy instanceID of any instance and also observe by clicking Elastic IP address (Copy allocation ID) eg (eipalloc-c343ac3,i-03fsdk3fsfsk3)

>>.response=ec2.associate\_address(AllocationId=‘copy allocatin id here’,InstanceId=’copy instance

id here’)

>>>print(response)

>>> response=ec2.release\_address(AllocationId=’eipalloc-c338sfs2’)

>>>print (response)

Session 6: working with region and group

>>>import boto3

>>> ec2=boto3.client(‘ec2’)

>>>ec2.describe\_regions()

>>> resp=ec2.describe\_availablity\_zones()

>>>print (‘Availability zones:’,resp[‘AvailabilityZones’])

>>> exit()

Security group :

>>>import boto3

>>> ec2=boto3.client(‘ec2’)

>>>result=ec2.describe\_security\_groups()

>>>print(result)

Session 8: Start /Stop Ec2 instance and monitor ec2 instance

>>>import boto3

>>>ec2=boto3.client(‘ec2’) //create client connection

>>>resp=ec2.describe\_instances() //gives ouput of running instance

>>>print (resp)

>>> result=ec2.monitor\_instances(InstanceIds=[‘get instance id and paste’]

Note : get instance id from instance management page in aws and paste instanceid

To stop monitoring , use ec2.unmonitor\_instances(InstanceIds=[‘i-acscs’]

>>>print (result)

>>> ec2.stop\_instances(InstanceIds=[‘i-aceasffs’])

>>>ec2.start\_instances(InstanceIds=[‘i-fsdfsd’]

>>>ec2.reboot\_instances(InstanceIds=[‘i-fksdjfkl’]

Session 9:

>>> import boto3

>>>s3=boto3.client(‘s3’)

>>> web={‘ErrorDocument’:{‘key’:’error.html’},’IndexDocument’:{'Suffix’:’index.html’}}

>>> s3.put\_bucket\_website(Bucket=’mynamebuckets’,WebsiteConfiguration=web\_

>>>result=s3.get\_bucket\_website(Bucket=’mynamebuckets’)

>>> print(result)

Now , observe in aws S3 , buckets, a new bucket is created

Check “Static website Hosting” box by clicking and observe Use this bucket to host web site with index.html and error.html settings.

Create index.html and error.html (sample) in a dummy folder

To upload this to static bucket

>>> s3.upload\_file(‘index.html’,’mynamebuckets’,’index.html’)

Refresh aws bucket and observe, index.html is uploaded successfully.

>>> resp=s3.get\_bucket\_policy(Bucket=’mynamebuckets’)

>>> print(resp)

No policy set (error). Now let set policy

>>> bucket\_policy={‘Version’:’2019-02-22’,’Statement’:[{‘Sid’:’AddPerm’,’Effect’:’Allow’,’Principal’:’\*’,’Actoin’:[‘s3:GetObject’],’Resources’:”arn:aws:s3:::%s/\*” % ‘mynamebuckets’}]}

>>>import json

>>>bucket\_policy=json.dumps(bucket\_policy)

>>> s3.put\_bucket\_policy(Bucket=’myamebuckets’,Policy=bucket\_policy)

>>> s3.delete\_bucket\_policy(Bucket=’mynamebuckets’)

Sesssion 10: creating new ubucket and uploading zip file to bucket

>>>import boto3

>>>s3=boto3.client(‘s3’)

>>> response=s3.list\_buckets()

>>> print(response)

>>> buckets=[ bucket[‘Name’] for nucket in resonse(‘Buckets’]]

>>> print(‘Bucket name:%s ’, % buckets)

>>> s3.create\_bucket(Bucket=‘testbucket’,CreateBucketConfiguration={‘LocationConstrint’:’ap-south’})

Check in aws service , new bucket is created (refresh)

Upload a file in above empty bucket

>>> filename=’todofile.zip’

>>> bucket\_name=’testbucket’

>>> s3.upload\_file(filename,bucket\_name,filename)

Goto web console and check , file is uploaded

Downloading file from bucket

>>> import boto3

>>> s3=boto3.client(‘s3’)

>>>> s3.download\_file(‘mynamebuckets’,’todolist.zip’,’client\_todolist.zip’)

Usinng resource method:

>>> import boto3

>>> import botocore (botocore for error handling)

>>> s3=boto3.resource(‘s3’)

>>>try:

… s3.Bucket(‘mynamebuckets’).download\_file(‘todolist.zip’,’resourcetodolist.zip’)

… except botocore.exceptions.ClientErrors as e:

… if e.response[‘Error’][‘Code’]== 404’:

… print(‘The obect does not exist’)

… else:

… raise

….

>>>

Observe file is downloaded very fast than client method

Session : 12: CORS configuration in AWS

>>>import boto3

>>> s3=boto3.client(‘s3’)

>>> result=s3.get\_bucket\_cors(Bucket=’mynamebucket’)

Error: The CORS configuration does not exist

>>> cors\_configuration={

‘CORSRules’:[ {‘AllowedHeaders’:[‘Authorization’],’AllowedMethods’:[‘GET’,’PUT’],’AllowedOrigins’:[‘\*’],’ExposeHeaders’:[‘GET’,’PUT’],’MaxAgeSeconds’:3000}]}

>>>s3.put\_bucket\_cors(Bucket=’mynamebuckets’,CORSConfiguration=cors\_configuration)

>>> result=s3.get\_bucket\_cors(Bucket=’mynamebucket’)

>>>print (result)

>>> result=s2.get\_bucket\_acl(Bucket=’mynamebucket’)

>>> print(result)

CROSS DOMAIN ACCESS In AWS:

Goto AWS, Create 2 buckets

1. Click s3 🡪 CreateBucket 🡪 Bucketname:testbucket111 🡪 OK
2. CreateBucket 🡪 BucketName: testbucket222 🡪 ok
3. Goto testbucket111 🡪 click Static Website Hosting property box 🡪
   1. Check – Use this bucket to host a web site
   2. Index document:index.html
   3. Error document: error.html 🡪 save

Now create index.html in VS Code in a dummy folder and write code:

<html>

<head>

<script src=<http://ajax.googleapis.com/ajax/libs/jquery1.8.1/jquery.min.js>></script>

<body>

<h1>Welcome to AWS web site 111</h1>

<div id=”loaddiv”></div>

</body>

</html>

<script>

$(“#loaddiv”).load(“load.html”);

</script>

Create new file :load.html and type below code:

Hello this is from load.html

Create error.html:

<html>  
<head> <title>error</title></head>

<body>Error - Please contact murthy</body>

</html>

Goto AWS dashboard, upload above files to testbucket111.

Make sure that access to these files are public.

Next🡪 Next🡪 upload.

In Static website property box : click on endpoint link:http://xsfsdfsdfsdfs

To see the output in browser.

Now let us see how load these files from other bucket (testbucket222)

Click properties 🡪 Static website hosting 🡪 check first radio button🡪 index.html

Save

Upload only one file : load.html in this bucket.--> Next 🡪 make it public acccesible

Next🡪 ok

To test, click properties tab 🡪 static website hosting🡪 click endpoint. Link

Observe 403 forbidden. But in the url of browser , type load.html and see the output as we have not created index.html.

Now change ajax url in index.html

….

<script>

$(“#loaddiv”).load(<http://testbucket222.s2-website.ap-south-1.amazonaws.com/load.html>”);

Save and again upload with new changes.

Check again by opening index.html from testbucket111. Open console tab and see error CORS issue.

SO set the Cors to allow the cross orign request.

Goto testbucket222 🡪 permissions 🡪 click CORS configuration tab🡪

Type the template code:

<CORSConfiguration>

<CORSRule>

<AllowedOrigin>\*</AllowedOrigin>

<AllowedMethod>GET</AllowedMethod>

<MaxAgeSeconds>3000</MaxAgeSeconds>

<AllowedHeader>Authorization</AllowedHeader>

<CORule>

</CORSConfiguration>

Save this template

Now open testbucket111 website 🡪 properties 🡪 static web site 🡪 click the endpoint link. Program works now.