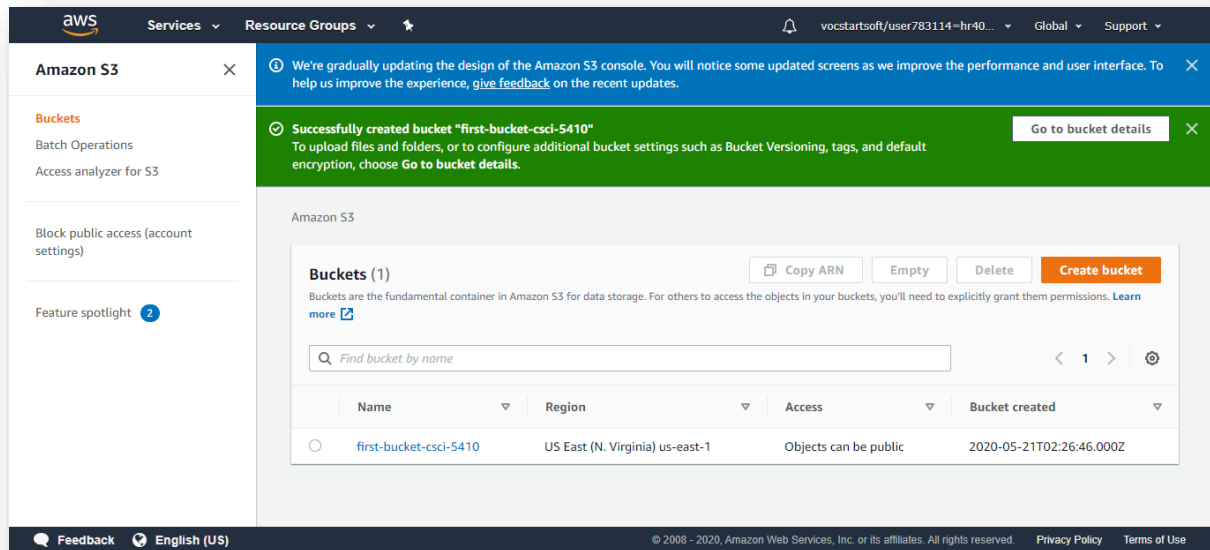


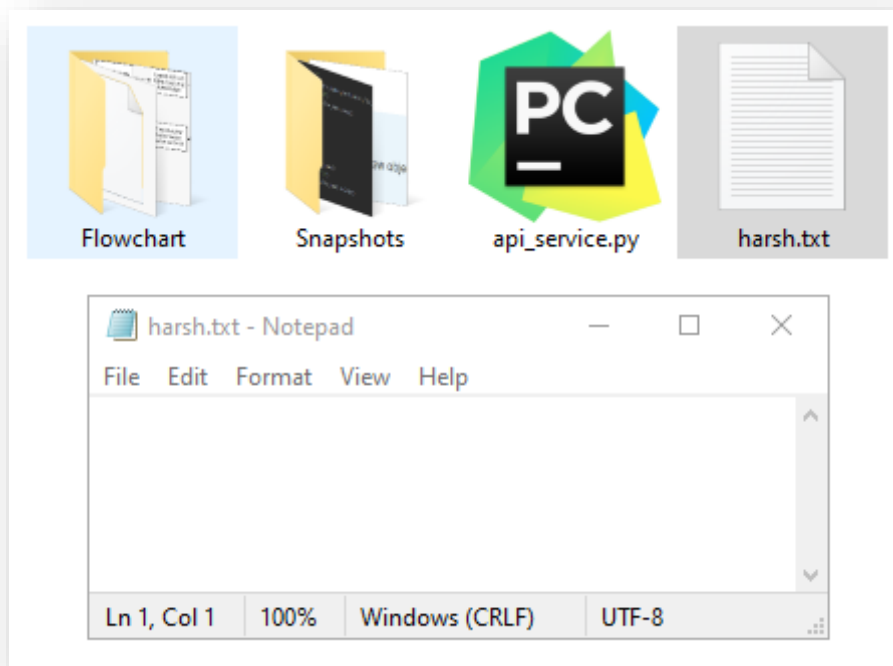
## Part B: AWS S3 storage experiment

a.

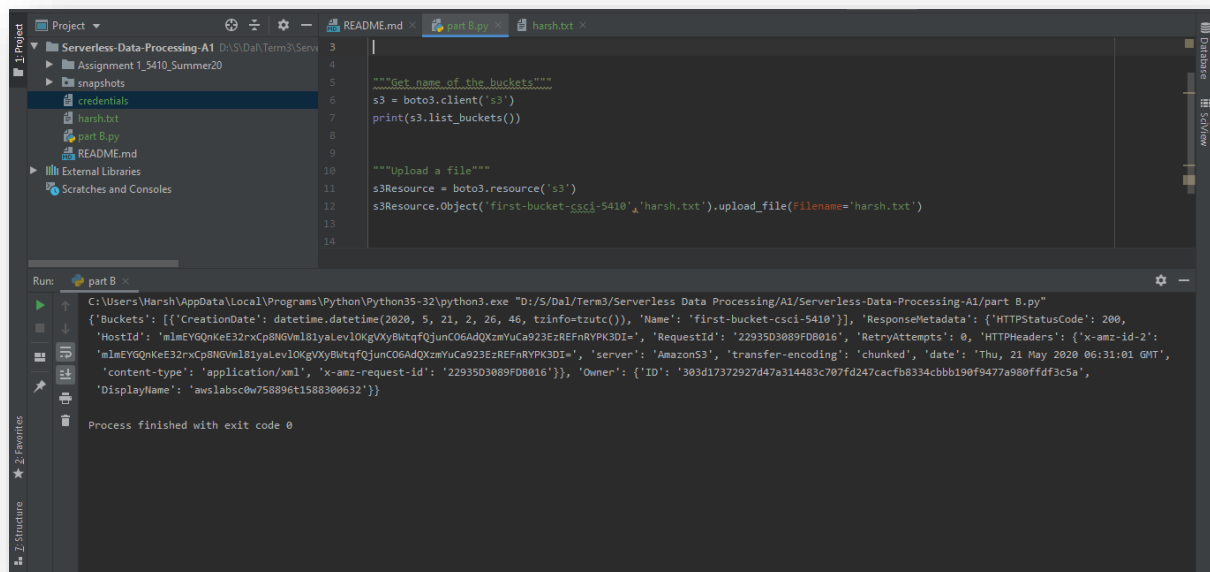
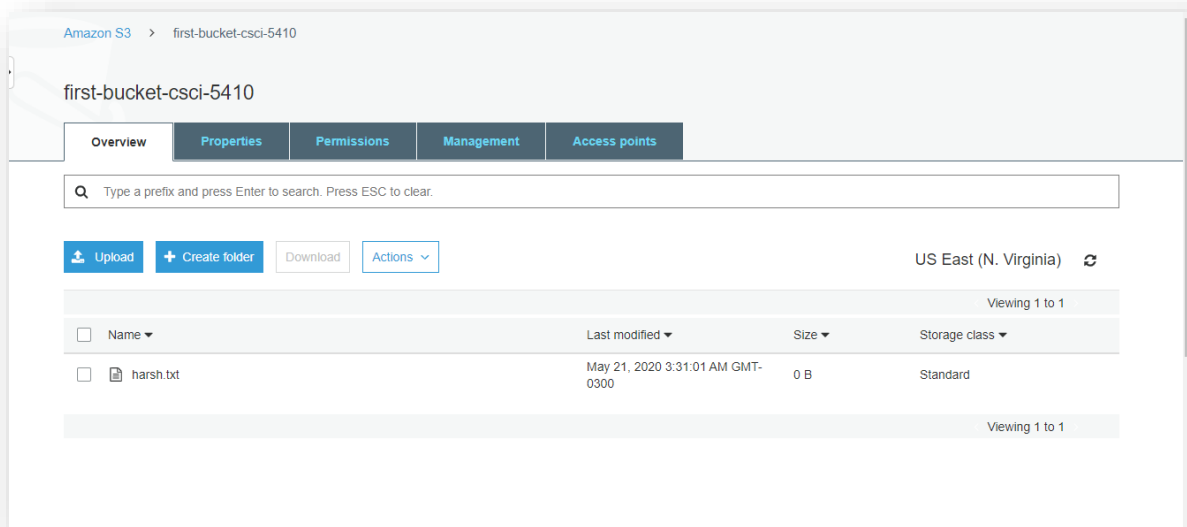
1. Create an empty bucket named “first-bucket-csci-5410”



2. Create an empty text file ‘harsh.txt’



b. Upload the file on the bucket “first-bucket-csci-5410”



C.

1. Create a second bucket named “second-bucket-csci-5410” using python.

**Buckets (2)** Copy ARN Empty Delete Create bucket

Buckets are the fundamental container in Amazon S3 for data storage. For others to access the objects in your buckets, you'll need to explicitly grant them permissions. [Learn more](#)

Find bucket by name

	Name	Region	Access	Bucket created
<input type="radio"/>	first-bucket-csci-5410	US East (N. Virginia) us-east-1	Objects can be public	2020-05-21T02:26:46.000Z
<input type="radio"/>	second-bucket-csci-5410	US East (N. Virginia) us-east-1	Objects can be public	2020-05-21T06:56:55.000Z

```
""" Creating another bucket """  
def createBucket(self, bucket_name):  
    s3Resource = boto3.resource('s3')  
    s3Resource.create_bucket(Bucket=bucket_name, )  
    print('Bucket created')
```

2. Change the access permission to “disable public access”

Block all public access Edit

On

- Block public access to buckets and objects granted through new access control lists (ACLs)  
On
- Block public access to buckets and objects granted through any access control lists (ACLs)  
On
- Block public access to buckets and objects granted through new public bucket or access point policies  
On
- Block public and cross-account access to buckets and objects through any public bucket or access point policies  
On

```

"""No Public Access"""
s3Client = boto3.client('s3',region_name='us-east-1')
s3Client.put_public_access_block(
    Bucket="second-bucket-cscli-5410",
    PublicAccessBlockConfiguration={
        'BlockPublicAcls': True,
        'IgnorePublicAcls': True,
        'BlockPublicPolicy': True,
        'RestrictPublicBuckets': True
    },
)

```

### 3. Change the ACL write option to “no” for bucket owner.

Block public access
Access Control List
Bucket Policy
CORS configuration

Access for bucket owner

Canonical ID	List objects	Write objects	Read bucket permissions	Write bucket permissions
<input type="radio"/> 303d17372927d47a314483c707fd247cacfb8334cbbb190f9477a980ffdf3c5a (Your AWS account)	-	-	Yes	-

Access for other AWS accounts

+ Add account
Delete

Canonical ID	List objects	Write objects	Read bucket permissions	Write bucket permissions
--------------	--------------	---------------	-------------------------	--------------------------

Project
Serverless-Data-Processing-A1
Assignment 1\_5410\_Summer20
snapshots
harsh.txt
part B.py
README.md
sample.py
External Libraries
Scratches and Consoles

part B.py
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
"""get ACL permissions"""
s3Client = boto3.client('s3')
json\_file=s3Client.get\_bucket\_acl(bucket='second-bucket-cscli-5410')
json\_file.pop('ResponseMetadata')
json\_file['Grants'][0]['Permission'] = 'READ'
s3Client.put\_bucket\_acl(Bucket='second-bucket-cscli-5410', AccessControlPolicy=json\_file)
print(s3Client.get\_bucket\_acl(bucket='second-bucket-cscli-5410'))

Run: part B
C:\Users\Harsh\AppData\Local\Programs\Python\Python35-32\python.exe "D:/S/Dal/Term3/Serverless Data Processing/A1/Serverless-Data-Processing-A1/part B.py"
{"Owner": {"DisplayName": "awslabsc0w758896t1588300632", "ID": "303d17372927d47a314483c707fd247cacfb8334cbbb190f9477a980ffdf3c5a"}, "ResponseMetadata": {"HostId": "oA0ua47+YxpH9z96OH9brh8UAAnkpp8eU48FZcau/Sj17jgPj+TCzhQ29nQ92fwDQ1F99Hg4RxxI=", "RequestId": "29F286A5CDC74A50", "HTTPStatusCode": 200, "HTTPHeaders": {"x-amz-id-2": "oA0ua47+YxpH9z96OH9brh8UAAnkpp8eU48FZcau/Sj17jgPj+TCzhQ29nQ92fwDQ1F99Hg4RxxI=", "x-amz-request-id": "29F286A5CDC74A50", "date": "Fri, 22 May 2020 01:29:13 GMT", "content-type": "application/xml", "server": "AmazonS3", "transfer-encoding": "chunked", "RetryAttempts": 0}, "Grants": [{"Grantee": {"Type": "CanonicalUser", "DisplayName": "awslabsc0w758896t1588300632", "ID": "303d17372927d47a314483c707fd247cacfb8334cbbb190f9477a980ffdf3c5a"}, "Permission": "READ"}]}
Process finished with exit code 0

Run
TODO
Version Control
Terminal
Python Console
Event Log
32-26 CRLF UTF-8 4 spaces Git: master Python 3.5.2 (C:\Users\H...\Python35-32\python3.exe)

d. Move the file from “first-bucket-csci-5410” to “second-bucket-csci-5410”.

second-bucket-csci-5410

Overview Properties Permissions Management Access points

🔍 Type a prefix and press Enter to search. Press ESC to clear.

Upload Create folder Download Actions

US East (N. Virginia) 🔁

Viewing 1 to 1

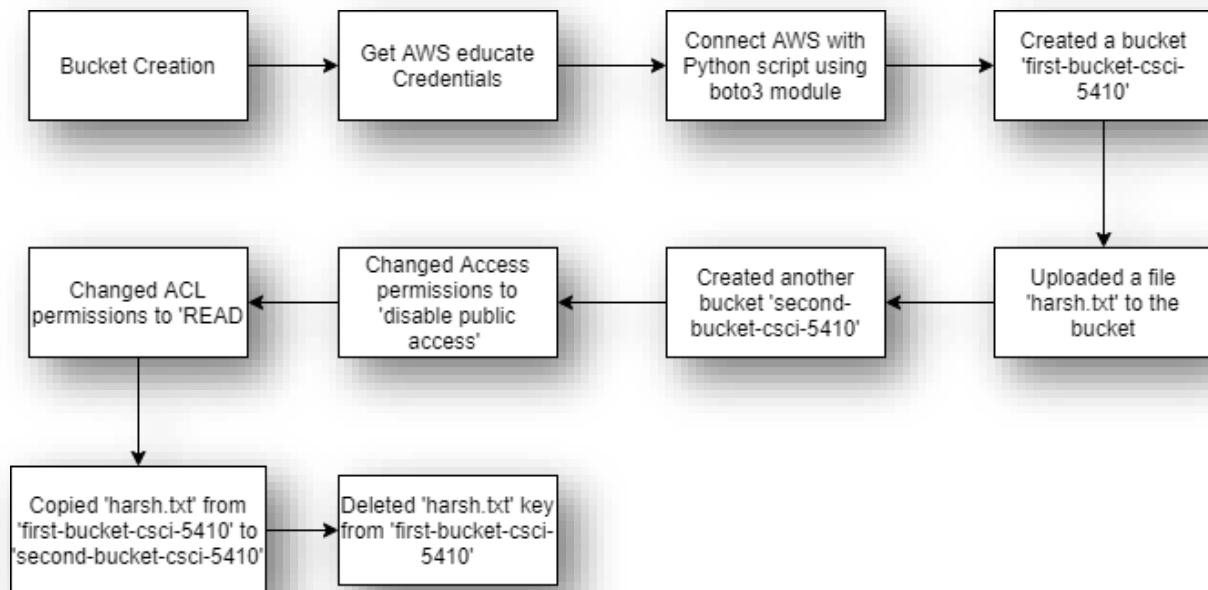
<input type="checkbox"/>	Name ▾	Last modified ▾	Size ▾	Storage class ▾
<input type="checkbox"/>	📄 harsh.txt	May 22, 2020 11:14:07 PM GMT-0300	0 B	Standard

Viewing 1 to 1

```
""" Copy keys between buckets """
def CopyKeysBetweenBuckets(self, source_bucket, dest_bucket, file_name):
    s3_resource = boto3.resource('s3')
    s3_resource.Object(dest_bucket, file_name).copy({
        'Bucket': source_bucket,
        'Key': file_name
    })
    print('key Copied')

""" Delete keys from a bucket """
def deleteKeys(self, bucketName, file_name):
    s3_resource = boto3.resource('s3')
    s3_resource.Object(bucketName, file_name).delete()
    print('Key deleted')
```

e. A flowchart showing the steps performed in the experiment



## Source Code

```
import boto3

class s3Api:
    """ Get name of the buckets """

    def listBuckets(self):
        s3 = boto3.client('s3')
        return s3.list_buckets()

    """ Upload a file """

    def fileUpload(self, bucket_name, source_file_name, file_name):
        s3Resource = boto3.resource('s3')
        s3Resource.Object(bucket_name, source_file_name).upload_file(Filename=file_name)
        print('file uploaded')

    """ Creating another bucket """

    def createBucket(self, bucket_name):
        s3Resource = boto3.resource('s3')
        s3Resource.create_bucket(Bucket=bucket_name, )
        print('Bucket created')

    """ Change bucket access (private, public etc) """

    def manageBucketAccess(self, bucket_name, block_public_acls=True, ignore_public_acls=True,
block_public_policy=True, restrict_public_buckets=True, region='us-east-1'):
        s3_client = boto3.client('s3', region_name=region)
        s3_client.put_public_access_block(
            Bucket=bucket_name,
            PublicAccessBlockConfiguration={
                'BlockPublicAcls': block_public_acls,
                'IgnorePublicAcls': ignore_public_acls,
                'BlockPublicPolicy': block_public_policy,
                'RestrictPublicBuckets': restrict_public_buckets
            },
        )
        print('Access Changed')

    """ Get ACL permissions """

    def getAclPermissions(self, bucket_name):
        s3_client = boto3.client('s3')
        return s3_client.get_bucket_acl(Bucket=bucket_name)
```

```
""" Set ACL permissions """
```

```
def changeAclPermissions(self, bucket_name, permission):  
    s3_client = boto3.client('s3')  
    json_file = s3_client.get_bucket_acl(Bucket=bucket_name)  
    json_file.pop('ResponseMetadata')  
    json_file['Grants'][0]['Permission'] = permission  
    s3_client.put_bucket_acl(Bucket=bucket_name, AccessControlPolicy=json_file)  
    print('ACL permission changed')
```

```
""" Copy keys between buckets """
```

```
def copyKeysBetweenBuckets(self, source_bucket, dest_bucket, file_name):  
    s3_resource = boto3.resource('s3')  
    s3_resource.Object(dest_bucket, file_name).copy({  
        'Bucket': source_bucket,  
        'Key': file_name  
    })  
    print('key Copied')
```

```
""" Delete keys from a bucket """
```

```
def deleteKeys(self, bucketName, file_name):  
    s3_resource = boto3.resource('s3')  
    s3_resource.Object(bucketName, file_name).delete()  
    print('Key deleted')
```

```
""" Downloading a file from the bucket """
```

```
def downloadFile(self, bucket_name, obj_name, dest_file_name):  
    s3Client = boto3.client('s3')  
    s3Client.download_file(bucket_name, obj_name, dest_file_name)  
    print('file downloaded')
```

```
s3 = s3Api()  
s3.listBuckets()  
s3.fileUpload("first-bucket-csci-5410", "harsh.txt", "harsh.txt")  
s3.createBucket("second-bucket-csci-5410")  
s3.manageBucketAccess("second-bucket-csci-5410")  
s3.getAclPermissions("second-bucket-csci-5410")  
s3.changeAclPermissions("second-bucket-csci-5410", "READ")  
s3.copyKeysBetweenBuckets("first-bucket-csci-5410", "second-bucket-csci-5410", "harsh.txt")  
s3.deleteKeys("first-bucket-csci-5410", "harsh.txt")  
s3.downloadFile("first-bucket-csci-5410", "Lookup5410.txt", "Lookup.txt")
```



## References

- [1]"Python S3 Examples — Ceph Documentation", *Docs.ceph.com*, 2020. [Online]. Available: <https://docs.ceph.com/docs/master/radosgw/s3/python/>. [Accessed: 25- May- 2020].
- [2]"Amazon S3 examples — Boto3 Docs 1.13.16 documentation", *Boto3.amazonaws.com*, 2020. [Online]. Available: <https://boto3.amazonaws.com/v1/documentation/api/latest/guide/s3-examples.html>. [Accessed: 25- May- 2020].
- [3]R. Python, "Python, Boto3, and AWS S3: Demystified – Real Python", *Realpython.com*, 2020. [Online]. Available: <https://realpython.com/python-boto3-aws-s3/>. [Accessed: 25- May- 2020].