CITS5503 Lab3

Wenxiao Zhang 22792191

[Step 1] Preparation

Download the python code cloudstorage.py from https://github.com/dglance/cits5503/blob/master/Labs/src/cloudstorage.py

Create a directory rootdir

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3$ wget https://raw.githubusercontent.com/uwacsp/cits5503/master/Labs/src/cloudstorage.py
--2022-08-14 20:30:07--  https://raw.githubusercontent.com/uwacsp/cits5503/master/Labs/src/cloudstorage.py
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.109.133, 185.199.110.133, 185.199.111.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.109.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 905 [text/plain]
Saving to: ‘cloudstorage.py’

cloudstorage.py                100%[=====================================================>]
 905  --.-KB/s    in 0s

2022-08-14 20:30:08 (42.6 MB/s) - ‘cloudstorage.py’ saved [905/905]
```

Create a file in rootdir called rootfile.txt and put some content in it “1\n2\n3\n4\n5\n”

Create a second directory in rootdir called subdir and create another file subfile.txt with the same content as rootfile.txt

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3$ mkdir rootdir
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3$ cd rootdir
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3/rootdir$ touch rootfile.txt
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3/rootdir$ sudo nano rootfile.txt
[sudo] password for moebuta:
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3/rootdir$ cat rootfile.txt
1\n2\n3\n4\n5\n
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3/rootdir$ mkdir subdir
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3/rootdir$ cd subdir
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3/rootdir/subdir$ touch subfile.txt
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3/rootdir/subdir$ sudo nano subfile.txt
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3/rootdir/subdir$ cat subfile.txt
1\n2\n3\n4\n5\n
```

[Step 2] Save to S3

Edit cloudstorage.py to take one argument: -i, --initialise=True – this will use boto to create a bucket on S3 that is identified by 22792191-cloudstorage

Insert boto commands to save each file that is found as the program traverses the directory starting at the root directory rootdir.

python code:

cloudstorage.py

2022s2 > cits5503 > labs > lab3 > cloudstorage.py > main

```python
from ast import parse
import os
from py_compile import main
import boto3
import base64
import argparse
import logging
from botocore.exceptions import ClientError

ROOT_DIR = '.'
ROOT_S3_DIR = '22792191-cloudstorage'

s3 = boto3.client("s3")
bucket_config = {'LocationConstraint': 'ap-southeast-2'}

def upload_file(folder_name, file, file_name):
    print("Uploading %s" % file)
    try:
        s3.upload_file(file,ROOT_S3_DIR, folder_name+file_name)
    except ClientError as e:
        logging.error(e)

def parse_args():
    parser = argparse.ArgumentParser(description="arg parser")
    parser.add_argument("-i", "--initialise", default=True, type=bool)
    return parser.parse_args()

# Main program
# Insert code to create bucket if not there
def main():
    args = parse_args()
    if args.initialise:
        try:
            response = s3.create_bucket(Bucket=ROOT_S3_DIR, CreateBucketConfiguration=bucket_config)
            print(response)
        except Exception as error:
            pass

    # parse directory and upload files
    for dir_name, subdir_list, file_list in os.walk(ROOT_DIR, topdown=True):
        if dir_name != ROOT_DIR:
            for fname in file_list:
                upload_file("%s/" % dir_name[2:], "%s/%s" % (dir_name, fname), fname)
    print("done")

if __name__ == "__main__":
    main()
```

upload file

create bucket

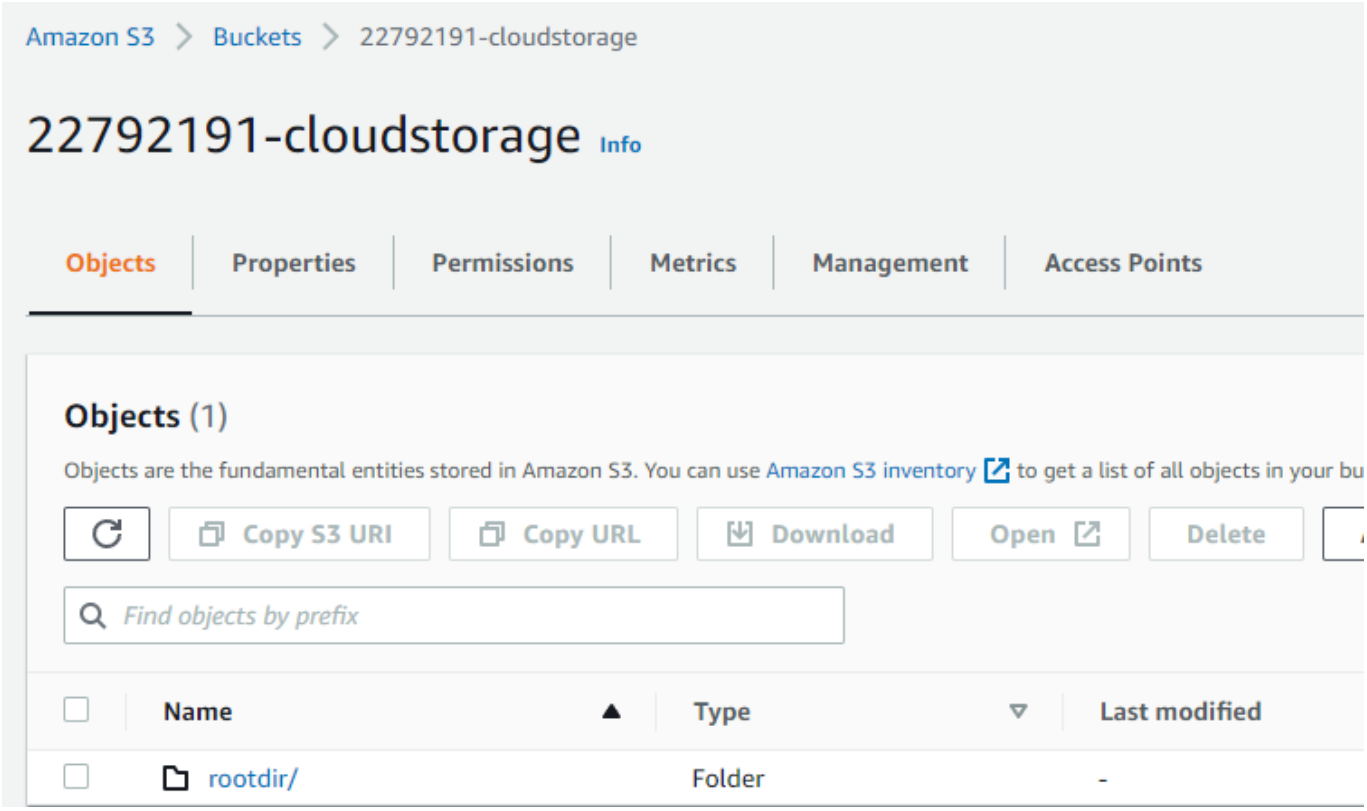
output for creating a bucket with student number:

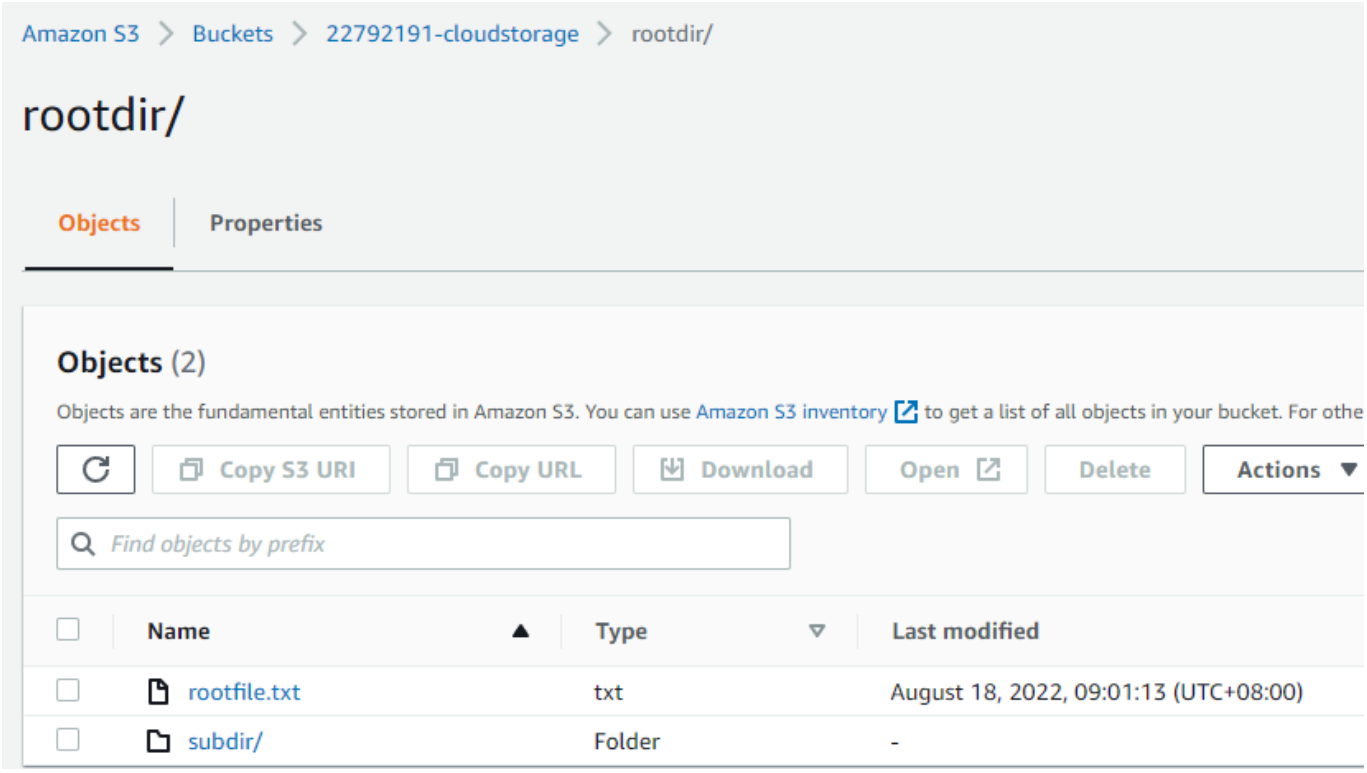
```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3$ python3 cloudstorage.py -i True
{'ResponseMetadata': {'RequestId': '2PJ70WRK4XR78Y8C', 'HostId': '2yj2TrX7eU1uRRNM6/4M1Aim3doFD6TB6+pXX9677MI5lwWn73o
OXMiiSVAhGi8req87fJzuBvI=', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amz-id-2': '2yj2TrX7eU1uRRNM6/4M1Aim3doFD6TB6+p
XX9677MI5lwWn73oOXMiiSVAhGi8req87fJzuBvI=', 'x-amz-request-id': '2PJ70WRK4XR78Y8C', 'date': 'Thu, 18 Aug 2022 07:10:2
1 GMT', 'location': 'http://22792191-cloudstorage.s3.amazonaws.com/', 'server': 'AmazonS3', 'content-length': '0'}, '
RetryAttempts': 0}, 'Location': 'http://22792191-cloudstorage.s3.amazonaws.com/'}
```

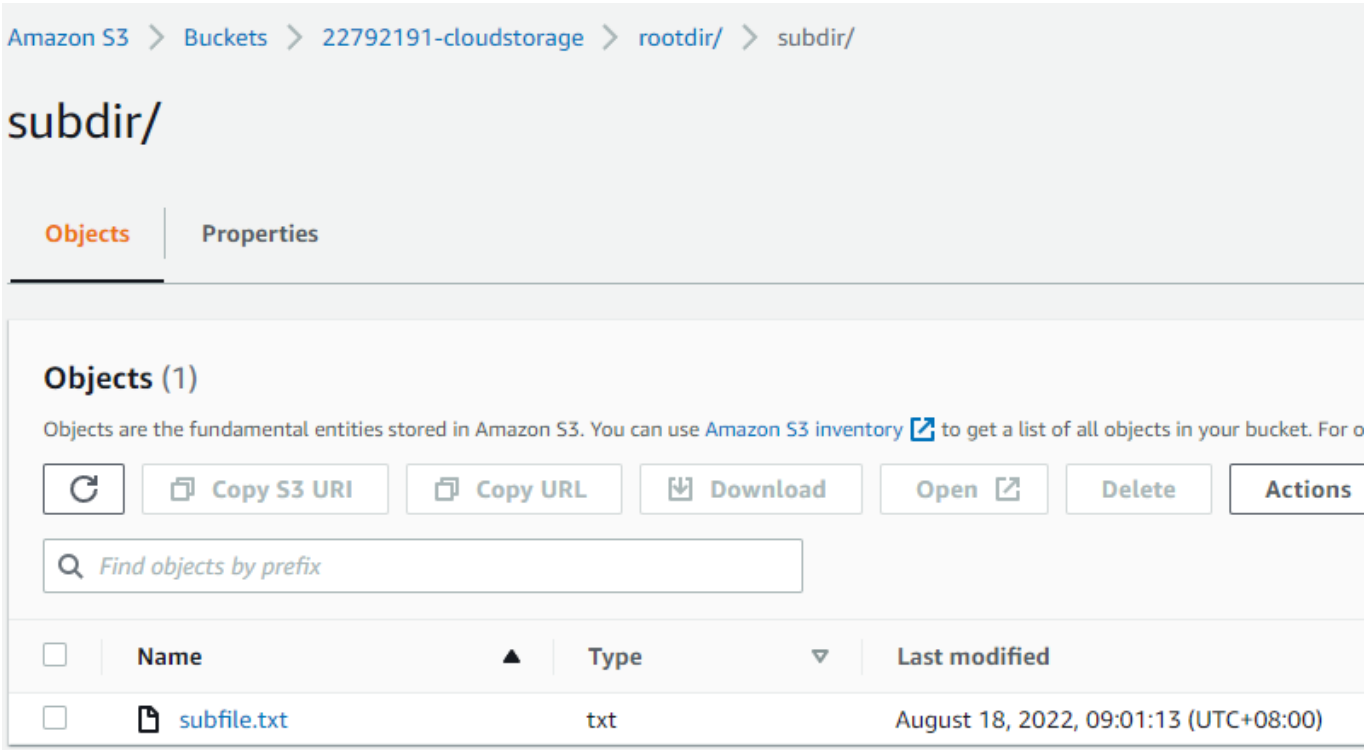
output for save each file:

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3$ python3 cloudstorage.py
Uploading ./rootdir/rootfile.txt
Uploading ./rootdir/subdir/subfile.txt
done
```

Open AWS console to confirm:







[Step 3] Restore from S3

Create a new program called restorefromcloud.py that reads the S3 bucket and writes the contents of the bucket within the appropriate directories. You should have a copy of the files and the directories you started with.

python code:

restorefromcloud.py

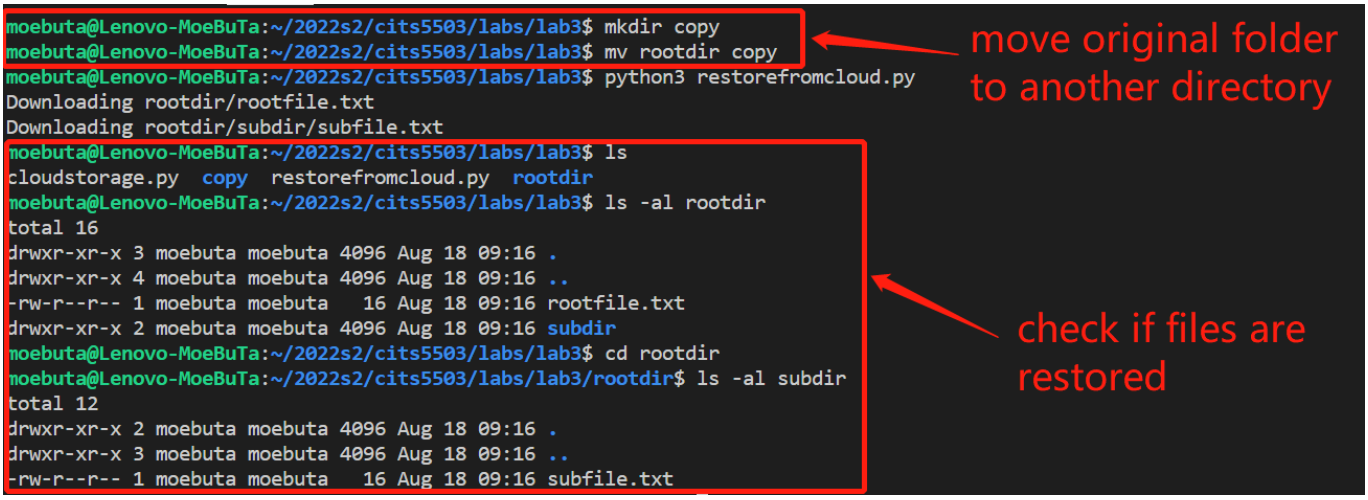
2022s2 > cits5503 > labs > lab3 > restorefromcloud.py > ...

```python
import os
import boto3
ROOT_DIR = '.'
ROOT_S3_DIR = '22792191-cloudstorage'

s3 = boto3.client("s3")
bucket_config = {'LocationConstraint': 'ap-southeast-2'}

for key in s3.list_objects(Bucket = ROOT_S3_DIR)['Contents']:
    print("Downloading %s" % key['Key'])
    if not os.path.exists(os.path.dirname(key['Key'])):
        os.makedirs(os.path.dirname(key['Key']))
    s3.download_file(ROOT_S3_DIR, key['Key'],  key['Key'])
```

output:



[Step 4] Write information about files to DynamoDB

Install DynamoDB on your VM.

mkdir dynamodb;

cd dynamodb

Install jre if not done

sudo apt-get install default-jre

wget https://s3-ap-northeast-1.amazonaws.com/dynamodb-local-tokyo/dynamodb_local_latest.tar.gz

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3$ mkdir dynamodb
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3$ cd dynamodb
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3/dynamodb$ sudo apt-get install default-jre
[sudo] password for moebuta:
Reading package lists... Done
Building dependency tree
Reading state information... Done
default-jre is already the newest version (2:1.11-72).
0 upgraded, 0 newly installed, 0 to remove and 105 not upgraded.
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3/dynamodb$ wget https://s3-ap-northeast-1.amazonaws.com/
dynamodb-local-tokyo/dynamodb_local_latest.tar.gz
--2022-08-17 10:27:12--  https://s3-ap-northeast-1.amazonaws.com/dynamodb-local-tokyo/dynamodb_local_lates
t.tar.gz
Resolving s3-ap-northeast-1.amazonaws.com (s3-ap-northeast-1.amazonaws.com)... 52.219.16.210
Connecting to s3-ap-northeast-1.amazonaws.com (s3-ap-northeast-1.amazonaws.com)|52.219.16.210|:443... conn
ected.
HTTP request sent, awaiting response... 200 OK
Length: 44189228 (42M) [application/x-tar]
Saving to: ‘dynamodb_local_latest.tar.gz’

dynamodb_local_latest.tar. 100%[===================================>]  42.14M  9.04MB/s    in 5.4s

2022-08-17 10:27:19 (7.77 MB/s) - ‘dynamodb_local_latest.tar.gz’ saved [44189228/44189228]
```

java -Djava.library.path=./DynamoDBLocal_lib -jar DynamoDBLocal.jar –sharedDb

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3/dynamodb$ java -Djava.library.path=./DynamoDBLocal_lib
-jar DynamoDBLocal.jar -sharedDb
Initializing DynamoDB Local with the following configuration:
Port:   8000
InMemory:       false
DbPath: null
SharedDb:       false
shouldDelayTransientStatuses:   false
CorsParams:     *
```

Create a table on your local DynamoDB with the key userId The attributes for the table will be:

```
CloudFiles = {
    'userId',
    'fileName',
    'path',
    'lastUpdated',
'owner',
    'permissions'
    }
)
```

For every file that is stored in S3, get the information to put in the DynamoDB item and write it to the table. You will have to find functions in Python to get details like time lastUpdated, owner and permissions. All of this information can be stored as strings.

Creating table: We set `CloudFiles` to be the table name, `userId` to be the partition key, and `fileName` to be the sort key.


```
moebuta@Lenovo-MoeBuTa:~$ aws dynamodb create-table \
> --table-name CloudFiles \
> --attribute-definitions \
> AttributeName=userId,AttributeType=S \
> AttributeName=fileName,AttributeType=S \
> --key-schema \
> AttributeName=userId,KeyType=HASH \
> AttributeName=fileName,KeyType=RANGE \
> --provisioned-throughput ReadCapacityUnits=10,WriteCapacityUnits=10 \
> --endpoint-url=http://localhost:8000
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "userId",
                "AttributeType": "S"
            },
            {
                "AttributeName": "fileName",
                "AttributeType": "S"
            }
        ],
        "TableName": "CloudFiles",
        "KeySchema": [
            {
                "AttributeName": "userId",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "fileName",
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "ACTIVE",
        "CreationDateTime": 1660889189.301,
        "ProvisionedThroughput": {
            "LastIncreaseDateTime": 0.0,
            "LastDecreaseDateTime": 0.0,
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 10,
            "WriteCapacityUnits": 10
        },
        "TableSizeBytes": 0,
        "ItemCount": 0,
        "TableArn": "arn:aws:dynamodb:ddblocal:000000000000:table/CloudFiles"
    }
}
```

Python code to extract user information from bucket and put them into the local dynamoDB:

storeuserinfo.py

2022s2 > cits5503 > labs > lab3 > storeuserinfo.py > ...

```python
import boto3

ROOT_S3_DIR = '22792191-cloudstorage'
TABLE = 'CloudFiles'

dynamodb = boto3.resource('dynamodb', region_name='ap-southeast-2', endpoint_url='http://localhost:8000')
table = dynamodb.Table(TABLE)
s3 = boto3.client('s3')
response = s3.list_objects(Bucket=ROOT_S3_DIR)

userId = str(response['Contents'][0]['Owner']['ID'])
owner = response['Contents'][0]['Owner']['DisplayName']
permission = s3.get_bucket_acl(Bucket=ROOT_S3_DIR)['Grants'][0]['Permission']

i = 1
for content in response['Contents']:
    item = {
        'id': i,
        'userId':userId,
        'fileName': content['Key'].split('/')[-1],
        'path': content['Key'],
        'lastUpdated': str(content['LastModified']),
        'owner': owner,
        'permissions': permission
    }
    print('puting the following item into CloudFiles table:\n', item, '\n')
    i+=1
    table.put_item(Item = item)
print('done!')
```

Output:

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3$ python3 storeuserinfo.py
puting the following item into CloudFiles table:
 {'id': 1, 'userId': 'e899a06030e20f8e9945922db62a14112f50d6a6d19721ca733875beb9e50f3c', 'fileName': 'rootfile.txt',
'path': 'rootdir/rootfile.txt', 'lastUpdated': '2022-08-19 06:12:18+00:00', 'owner': 'mdanwarulkaium.patwary', 'permi
ssions': 'FULL_CONTROL'}

puting the following item into CloudFiles table:
 {'id': 2, 'userId': 'e899a06030e20f8e9945922db62a14112f50d6a6d19721ca733875beb9e50f3c', 'fileName': 'subfile.txt', '
path': 'rootdir/subdir/subfile.txt', 'lastUpdated': '2022-08-19 06:12:18+00:00', 'owner': 'mdanwarulkaium.patwary', '
permissions': 'FULL_CONTROL'}

done!
```

Scan the content of the local table:

```
moebuta@Lenovo-MoeBuTa:~/2022s2/cits5503/labs/lab3$ aws dynamodb scan --table-name CloudFiles --endpoint-url http://l
ocalhost:8000
{
    "Items": [
        {
            "owner": {
                "S": "mdanwarulkaium.patwary"
            },
            "path": {
                "S": "rootdir/rootfile.txt"
            },
            "lastUpdated": {
                "S": "2022-08-19 06:12:18+00:00"
            },
            "fileName": {
                "S": "rootfile.txt"
            },
            "permissions": {
                "S": "FULL_CONTROL"
            },
            "id": {
                "N": "1"
            },
            "userId": {
                "S": "e899a06030e20f8e9945922db62a14112f50d6a6d19721ca733875beb9e50f3c"
            }
        },
        {
            "owner": {
                "S": "mdanwarulkaium.patwary"
            },
            "path": {
                "S": "rootdir/subdir/subfile.txt"
            },
            "lastUpdated": {
                "S": "2022-08-19 06:12:18+00:00"
            },
            "fileName": {
                "S": "subfile.txt"
            },
            "permissions": {
                "S": "FULL_CONTROL"
            },
            "id": {
                "N": "2"
            },
            "userId": {
                "S": "e899a06030e20f8e9945922db62a14112f50d6a6d19721ca733875beb9e50f3c"
            }
        }
    ],
    "Count": 2,
    "ScannedCount": 2,
    "ConsumedCapacity": null
}
```