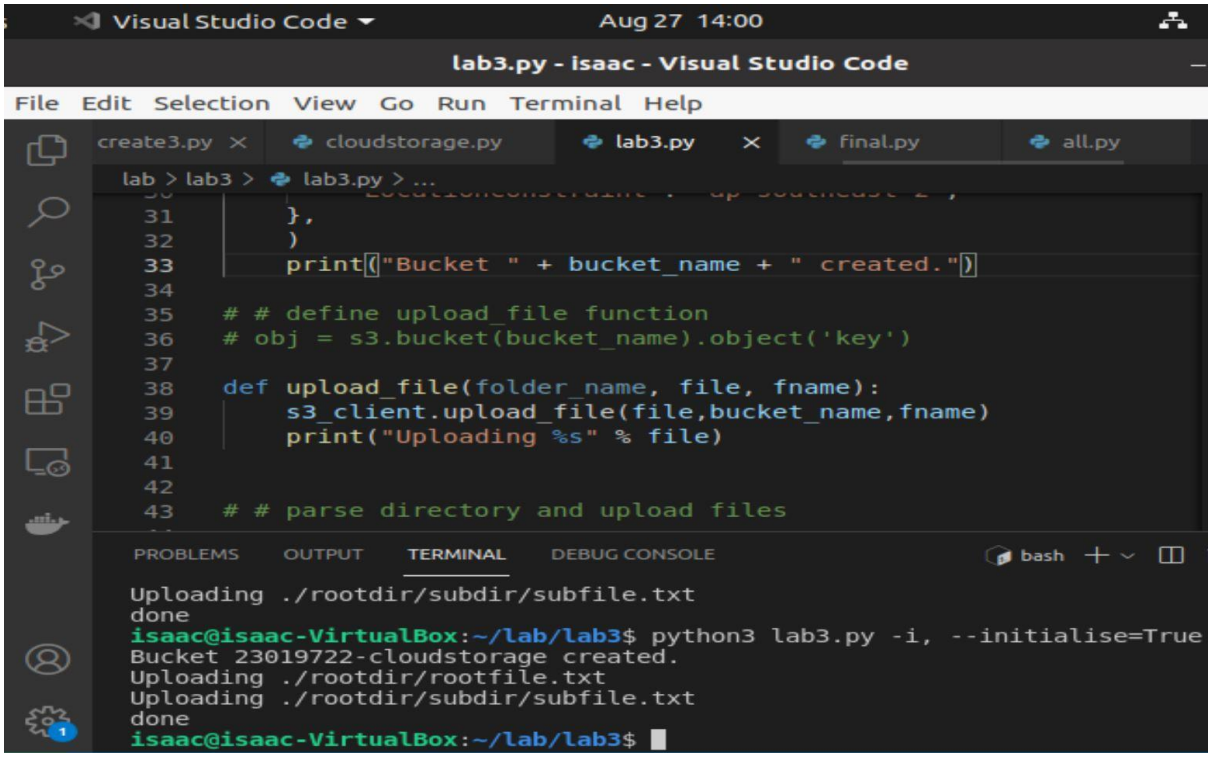
1. Upload Files to S3

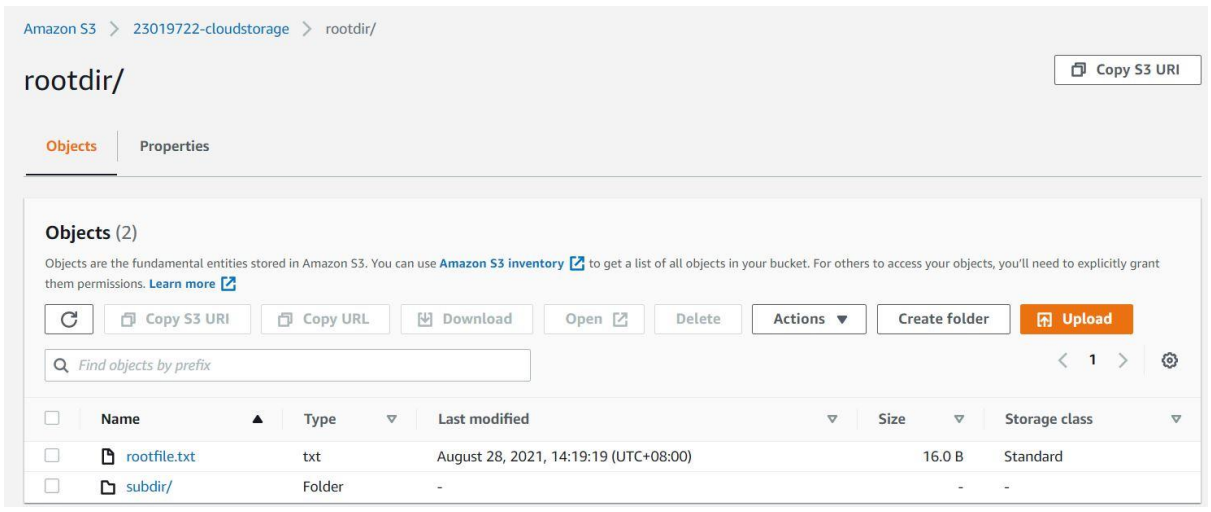
1.1 Script

```
import os
import boto3, botocore
from botocore.exceptions import ClientError

ROOT_DIR = '.'
s3 = boto3.resource('s3')
s3_client = boto3.client('s3')
bucket_name = '23019722-cloudstorage'
# key = 'AKIAZB5ESDWMTO34OCOP'

def check_bucket(bucket):
    try:
        s3.meta.client.head_bucket(Bucket=bucket_name)
        print("Bucket Exists!")
        return True
    except botocore.exceptions.ClientError as e:
        error_code = int(e.response['Error']['Code'])
        if error_code == 403:
            print("Private Bucket. Forbidden Access!")
            return True
        elif error_code == 404:
            return False

if check_bucket(bucket_name) is False:
    s3.create_bucket(
    Bucket=bucket_name,
    CreateBucketConfiguration={
        'LocationConstraint': 'ap-southeast-2',
    },
    )
    print("Bucket " + bucket_name + " created.")

# # define upload_file function
# obj = s3.bucket(bucket_name).object('key')

def upload_file(file):
    s3_client.upload_file(file,bucket_name,file[2:])
    print("Uploading %s" % file)
```

```
# # parse directory and upload files

for dir_name, subdir_list, file_list in os.walk(ROOT_DIR, topdown=Tr
ue):
    print(dir_name, subdir_list, file_list)
    if dir_name != ROOT_DIR:
        for fname in file_list:
            upload_file("%s/%s" % (dir_name, fname))
print("done")
```

1.2 Result





2. Retore All Files Back from S3 & Write Files Info into DynamoDB

2.1 Script

```
import os
import errno
import boto3

client = boto3.client('s3')
resource = boto3.resource('s3')
bucket_name = '23019722-cloudstorage'
my_bucket = resource.Bucket(bucket_name)

dynamodb = boto3.client('dynamodb', endpoint_url='http://localhost:8
000')
dynamodb_res = boto3.resource('dynamodb', endpoint_url='http://local
host:8000')

tableName = 'CloudFile2'
table = dynamodb_res.Table(tableName)
existing_tables = dynamodb.list_tables()['TableNames']

# create table, AttributeDefinitions, KeySchema, TableName
if tableName not in existing_tables:
    print('Table ' + tableName + 'does not exist.')
    response = dynamodb.create_table(
        AttributeDefinitions = [
            {
                'AttributeName': 'userId',
                'AttributeType': 'S'
            },
            {
                'AttributeName': 'fileName',
                'AttributeType': 'S'
            },
        ],
        KeySchema = [
            {
                'AttributeName': 'userId',
                'KeyType': 'HASH'
            },
            {
                'AttributeName': 'fileName',
```

```
                'KeyType': 'RANGE'
            }
        ],
        ProvisionedThroughput = {
            'ReadCapacityUnits': 5,
            'WriteCapacityUnits': 5
        },
        TableName = tableName
    )
    print(response)

def assert_dir_exists(path):
    try:
        os.makedirs(path)
    except OSError as e:
        if e.errno != errno.EEXIST:
            raise

def download_dir(bucket, path, target):
    # Handle missing / at end of prefix
    if not path.endswith('/'):
        path += '/'
    paginator = client.get_paginator('list_objects_v2')
    for result in paginator.paginate(Bucket=bucket, Prefix=path):
        for key in result['Contents']:
            file_name = str(key['Key']).rsplit('/')[-1]
            last_time = str(key['LastModified']).rsplit('+')[0]
            objects = client.get_object_acl(Bucket=bucket_name, Key
= key['Key'])
            perm = str(objects['Grants'][0]['Permission'])
            dynamodb.put_item(
                TableName = tableName,
                Item = {'userId':{'S':objects['ResponseMetadata']['R
equestId']},
                        'fileName':{'S':file_name},
                        'path':{'S':key['Key']},
                        'lastUpdated':{'S':last_time},
                        'owner':{'S':objects['Owner']['DisplayName']
},
                        'permissions':{'S':perm}})
            # Calculate relative path
            rel_path = key['Key'][len(path):]
```

```
            # Skip paths ending in /
            if not key['Key'].endswith('/'):
                local_file_path = os.path.join(target, rel_path)
                # Make sure directories exist
                local_file_dir = os.path.dirname(local_file_path)
                assert_dir_exists(local_file_dir)
                client.download_file(bucket, key['Key'], local_file_
path)

download_dir(bucket_name, 'rootdir/', '/home/isaac/lab/lab3/rootdir/
')
print(table.scan())
```

2.2 Result 1

Note 1: use "get_paginator" if there are more than 1000 files in S3.

Note 2: use "os.makedirs()" to recursively create all directories if any local directories don't exist.

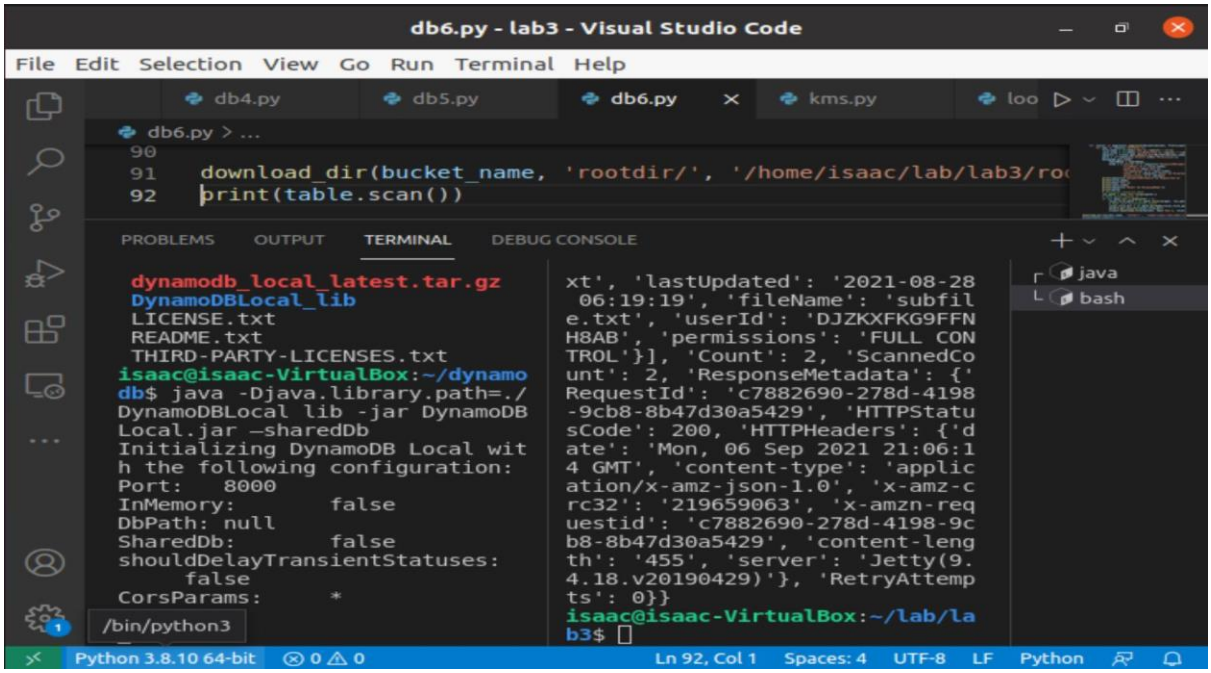
Use these two func above to realise one commend for restoring everything back to hard drive from S3.

```
isaac@isaac-VirtualBox:~/lab/lab3$ ls
lab3.py  restore1.py
isaac@isaac-VirtualBox:~/lab/lab3$ python3 restore1.py
isaac@isaac-VirtualBox:~/lab/lab3$ ls -R ./rootdir/
./rootdir/:
rootfile.txt  subdir

./rootdir/subdir:
subfile.txt
isaac@isaac-VirtualBox:~/lab/lab3$
```

2.3 Result 2 – Local DynamoDB Table

Note to myself for future ref: commend for running DynamoDB:

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

tar -zxvf dynamodb_local_latest.tar.gz

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


db6.py - lab3 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
db4.py
db5.py
db6.py
kms.py
db6.py > ...
90
91 download_dir(bucket_name, 'rootdir/', '/home/isaac/lab/lab3/ro
92 print(table.scan())
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
dynamodb_local_latest.tar.gz
DynamoDBLocal_lib
LICENSE.txt
README.txt
THIRD-PARTY-LICENSES.txt
isaac@isaac-VirtualBox:~/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: *
xt', 'lastUpdated': '2021-08-28 06:19:19', 'fileName': 'subfile.txt', 'userId': 'DJZKXFKG9FFNH8AB', 'permissions': 'FULL CONTROL'}], 'Count': 2, 'ScannedCount': 2, 'ResponseMetadata': {'RequestId': 'c7882690-278d-4198-9cb8-8b47d30a5429', 'HTTPStatusCode': 200, 'HTTPHeaders': {'date': 'Mon, 06 Sep 2021 21:06:14 GMT', 'content-type': 'application/x-amz-json-1.0', 'x-amz-crc32': '219659063', 'x-amzn-requestid': 'c7882690-278d-4198-9cb8-8b47d30a5429', 'content-length': '455', 'server': 'Jetty(9.4.18.v20190429)'}, 'RetryAttempts': 0}}
isaac@isaac-VirtualBox:~/lab/lab3$
java
bash
/bin/python3
Python 3.8.10 64-bit
0 0
Ln 92, Col 1 Spaces: 4 UTF-8 LF Python