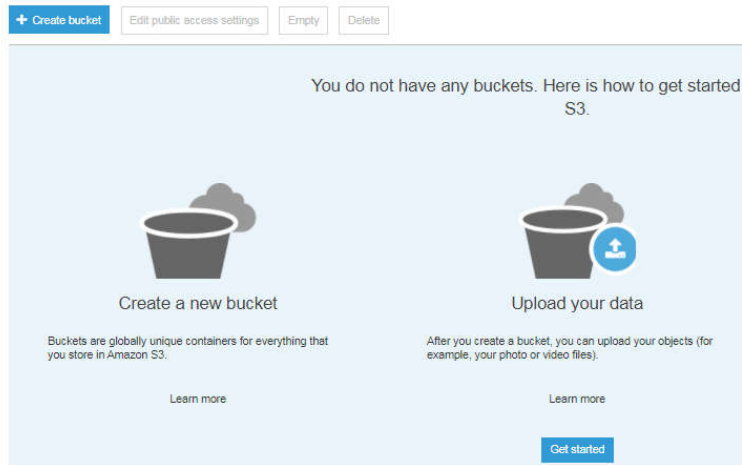


S3

Task 1: Create buckets and store files in S3 using the console

Step 1: Select the S3 service – Create Bucket



Step 2: Specify Bucket name which is a unique name globally

This screenshot shows the 'Name and region' section of the Amazon S3 console. It includes a 'Bucket name' field with a help icon and the text 'karthi123bucket'. Below this is a 'Region' dropdown menu currently set to 'US East (N. Virginia)'. At the bottom, there is a section titled 'Copy settings from an existing bucket' with a dropdown menu showing 'You have no buckets0 Buckets'.

Step 3: Set Configure options parameters to default.

Step 4: Uncheck block all public access – click -- I acknowledge This allows access of files stored in bucket by other applications.

⚠ Disabling Block all public access may result in this bucket and the objects within becoming public
 AWS recommends that you block all public access to your bucket, unless public access is required for specific and verified use cases such as static website hosting.

☐ I acknowledge that the current settings may result in this bucket and the objects within becoming public

Block all public access
 Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

- Block public access to buckets and objects granted through *new* access control lists (ACLs)**
 S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
- Block public access to buckets and objects granted through *any* access control lists (ACLs)**
 S3 will ignore all ACLs that grant public access to buckets and objects.
- Block public access to buckets and objects granted through *new* public bucket or access point policies**
 S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

Step 5: Review and create the bucket

S3 buckets				Discover the console
<input type="text" value="Search for buckets"/>				All access types
+ Create bucket Edit public access settings Empty Delete				1 Buckets 1 Regions Refresh
<input type="checkbox"/>	Bucket name	Access	Region	Date created
<input type="checkbox"/>	karthi123bucket	Objects can be public	US East (N. Virginia)	Sep 2, 2020 3:35:22 PM GMT+0530

Step 6: Upload Files in S3 bucket

karthi123bucket

Overview

Properties


Permissions

Management

Access points

[Upload](#)[+ Create folder](#)[Download](#)[Actions](#)


This bucket is empty. Upload new objects to c



Upload an object

Buckets are globally unique containers for everything that you store in Amazon S3.

[Learn more](#)



Set object properties

After you create a bucket, you can upload your objects (for exam your photo or video files).

[Learn more](#)

Step 7: In permission Tab set access to public mode

1 Files Size: 759.6 KB Target path: karthi123bucket

Manage users

User ID	Objects	Object permissions
awslabsc0w301478t1557859542(Owner)	<input checked="" type="checkbox"/> Read	<input checked="" type="checkbox"/> Read <input checked="" type="checkbox"/> Write

Access for other AWS account [+ Add account](#)

Account	Objects	Object permissions

Manage public permissions

Grant public read access to this object(s)

Warning: This object(s) has public read access. Everyone in the world will have read access to this object(s).

[Upload](#) [Previous](#) [Next](#)

Step 8: User can select the storage class for the object – default is Standard Class

Storage class
Choose a storage class based on your use case and access requirements. [Learn more](#) or see [Amazon S3 pricing](#)

Storage class	Designed for	Availability Zones	Min storage duration	Min billable object size	Monitoring and automation fees	Retrieval fees
<input checked="" type="radio"/> Standard	Frequently accessed data	≥ 3	-	-	-	-
<input type="radio"/> Intelligent-Tiering	Long-lived data with changing or unknown access patterns	≥ 3	30 days	-	Per-object fees apply	-
<input type="radio"/> Standard-IA	Long-lived, infrequently accessed data	≥ 3	30 days	128KB	-	Per-GB fees apply
<input type="radio"/> One Zone-IA	Long-lived, infrequently accessed, non-critical data	≥ 1	30 days	128KB	-	Per-GB fees apply
<input type="radio"/> Glacier	Archive data with retrieval times ranging from minutes to hours	≥ 3	90 days	40KB	-	Per-GB fees apply
<input type="radio"/> Glacier Deep Archive	Archive data that rarely, if ever, needs to be accessed with retrieval times in hours	≥ 3	180 days	40KB	-	Per-GB fees apply
<input type="radio"/> Reduced Redundancy (Not recommended)	Frequently accessed, non-critical data	≥ 3	-	-	-	-

[Upload](#) [Previous](#) [Next](#)

Step 9: Review and upload the file. Object URL is used to view the file.

Penguins.jpg Latest version ▾

Overview

Properties

Permissions

Select from

Open

Download

Download as

Make public

Copy path

Owner

awslabsc0w301478t1557859542

Last modified

Sep 2, 2020 3:40:30 PM GMT+0530

Etag

9d377b10ce778c4938b3c7e2c63a229a

Storage class

Standard

Server-side encryption

None

Size

759.6 KB

Key

Penguins.jpg

Object URL

<https://karthi123bucket.s3.amazonaws.com/Penguins.jpg>

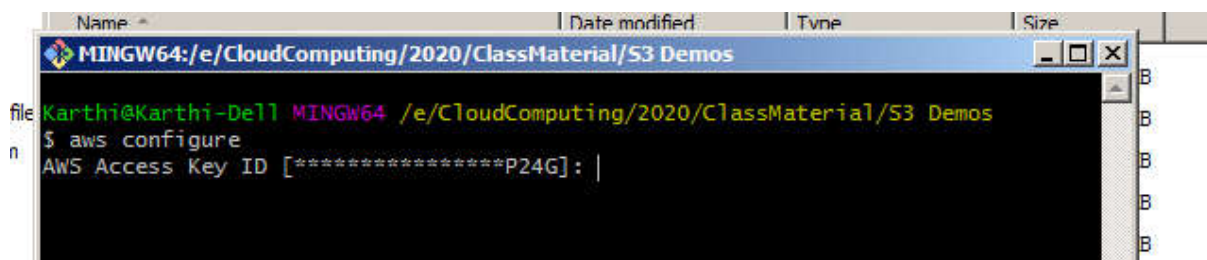
S3 - commands with the AWS CLI

The AWS Command Line Interface (CLI) is a unified tool to manage your AWS services.

Basic S3 commands in AWS

Step 1: Go to folder where you have files for upload and open Gitbash from the current directory

My directory is e:/CloudComputing/2020/Classmaterials/s3Demos

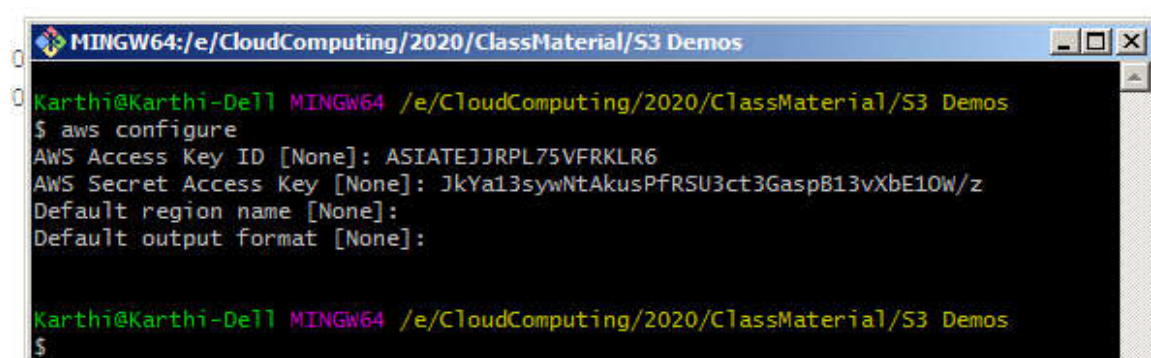


```
MINGW64: e:/CloudComputing/2020/ClassMaterial/S3 Demos
Karthi@Karthi-De11 MINGW64 /e/CloudComputing/2020/ClassMaterial/S3 Demos
$ aws configure
AWS Access Key ID [*****P24G]: |
```

Step 2: Configure AWS CLI to connect to AWS



Enter **Access Key Id** and **AWS Secret Access Key** from the Account details



```
MINGW64: e:/CloudComputing/2020/ClassMaterial/S3 Demos
Karthi@Karthi-De11 MINGW64 /e/CloudComputing/2020/ClassMaterial/S3 Demos
$ aws configure
AWS Access Key ID [None]: ASIATEJJRPL75VFRKLR6
AWS Secret Access Key [None]: JkYal3sywNtAkusPFRSU3ct3GaspB13vXbE10W/z
Default region name [None]:
Default output format [None]:

Karthi@Karthi-De11 MINGW64 /e/CloudComputing/2020/ClassMaterial/S3 Demos
$
```

Type a sample Command: **aws s3 ls**

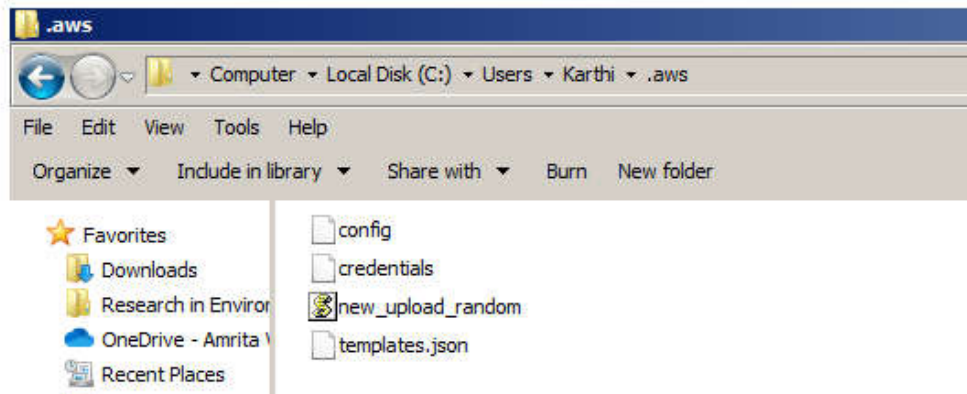
List all the buckets in S3

```
Karthi@Karthi-De11 MINGW64 /e/CloudComputing/2020/ClassMaterial/S3 Demos
$ aws s3 ls

An error occurred (InvalidAccessKeyId) when calling the ListBuckets operation: The AWS Access Key Id you provided does not exist in our records.

Karthi@Karthi-De11 MINGW64 /e/CloudComputing/2020/ClassMaterial/S3 Demos
$
```

This gives an error so update the credentials file of aws - **c:\Users\Karthi\.aws\credentials**



Add **aws-session-token** details in the file

```
credentials - Notepad
File Edit Format View Help

[default]
aws_access_key_id = ASIATEJJRPL75VFRKLR6
aws_secret_access_key = JkYa13sywNtAkusPFRSU3ct3GaspB13vXbE10w/z
aws_session_token=FwOGZXIVYXdzEKT////////wEaDF9GALZieZ3hh+pouILFAZeAuSwiy5jPVTQ8r06aDzh49nfrdpr4HsMDP1YwPWucJ5t4ueEm3BwbINT/qwEOHXX
+P4IQxyDdhTMy7aMgs/wJdhY0gvGeafFnpNne5QvootL84P8wuok0d+34s
+TwgYWAjTFYwXn1UORE7WQQha2j51jFwd880Zx/9ez8irzht5Xvv/zpoTaa3Ec1N50MZKARYu2gf2bxOyPu8IE139MR49z7ZYY4Do
+vyozp06eCXuHzhkFQvArJne40NZTuwtFMQZQKJHwvf0FM1KOSJb2BLHU68F35Qc1xavcZUNaP7aIv1Gu3FPJPr4j7sb5iqKUYs7Ab6DVwA=
```

AWS is configured in CLI mode

AWS CLI Command for S3

- a) List all the buckets in S3 - `aws s3 ls`

```
Karthi@Karthi-Dell MINGW64 /e/CloudComputing/2020/ClassMaterial/S3 Demos
$ aws s3 ls
2020-09-02 15:35:22 karthi123bucket
2020-09-02 16:01:01 karthidemosample
```

- b) Copy a file from local system to a bucket - `aws s3 cp sample.txt s3://karthi123bucket`

```
Karthi@Karthi-Dell MINGW64 /e/CloudComputing/2020/ClassMaterial/S3 Demos
$ aws s3 cp sample.txt s3://karthi123bucket
upload: .\sample.txt to s3://karthi123bucket/sample.txt
```

- c) Make a bucket in s3 - `aws s3 mb s3://newbucketkarthi`

```
Karthi@Karthi-Dell MINGW64 /e/CloudComputing/2020/ClassMaterial/S3 Demos
$ aws s3 mb s3://newbucketkarthi
make_bucket: newbucketkarthi
```

- d) Synchronize Two buckets - `aws s3 sync s3://karthi123bucket s3://newbucketkarthi`

```
Karthi@Karthi-Dell MINGW64 /e/CloudComputing/2020/ClassMaterial/S3 Demos
$ aws s3 sync s3://karthi123bucket s3://newbucketkarthi
copy: s3://karthi123bucket/sample.txt to s3://newbucketkarthi/sample.txt
copy: s3://karthi123bucket/Penguins.jpg to s3://newbucketkarthi/Penguins.jpg
```

- e) Remove a bucket - `aws s3 rb s3://newbucketkarthi --force`

```
Karthi@Karthi-Dell MINGW64 /e/CloudComputing/2020/ClassMaterial/S3 Demos
$ aws s3 rb s3://newbucketkarthi --force
delete: s3://newbucketkarthi/Penguins.jpg
delete: s3://newbucketkarthi/sample.txt
remove_bucket: newbucketkarthi
```

--force is used when bucket has elements.

- f) Remove a file from bucket - `aws s3 rm s3://karthi123bucket/data.txt`

```
Karthi@Karthi-Dell MINGW64 /e/CloudComputing/2020/ClassMaterial/S3 Demos
$ aws s3 rm s3://karthi123bucket/data.txt
delete: s3://karthi123bucket/data.txt
```

- g) Move files from / to bucket - `aws s3 mv readCSV.js s3://karthimybucket123`

```
Karthi@Karthi-Dell MINGW64 /e/CloudComputing/2020/ClassMaterial/S3 Demos/s3samples
$ aws s3 mv readCSV.js s3://karthimybucket123
move: .\readCSV.js to s3://karthimybucket123/readCSV.js
```


Node.js with S3

Program 1: Create a Bucket in S3 from Nodejs

Step 1: Make a directory and initialize a project in your local system

Use Gitbash and enter the command

```
mkdir nodeS3
```

```
npm init -y // initialize for node
```

```
Karthi@Karthi-Dell MINGW64 /e/CloudComputing/2020/ClassMaterial/S3 Demos/nodeS3 You are in nodeS3 Directory
$ npm init -y
Wrote to E:\CloudComputing\2020\ClassMaterial\S3 Demos\nodeS3\package.json:

{
  "name": "nodeS3",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
```

Step 2: Install packages for aws in node

```
npm install aws-sdk
```

```
Karthi@Karthi-Dell MINGW64 /e/CloudComputing/2020/ClassMaterial/S3 Demos/nodeS3
$ npm install aws-sdk
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN nodeS3@1.0.0 No description
npm WARN nodeS3@1.0.0 No repository field.

+ aws-sdk@2.745.0
added 14 packages from 66 contributors and audited 14 packages in 14.993s
found 0 vulnerabilities
```

Step 4: Generate the keys to connect to aws

Go to workbench – Click Account details

Copy and paste the following into ~/.aws/credentials

[default]

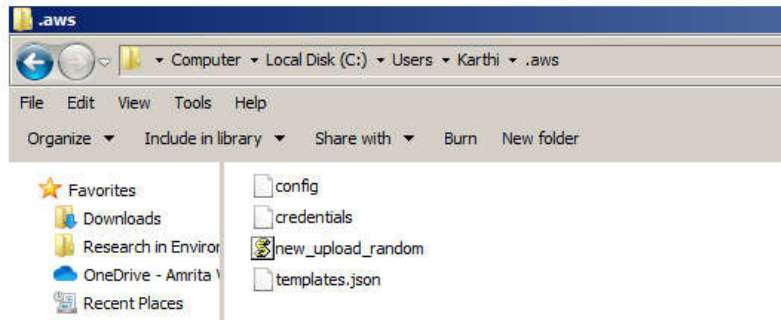
aws_access_key_id=ASIATEJJRPL7SYPEWSEH

aws_secret_access_key=tJ/7MC3NaIUQ0MYD5bDNAIYWCPiJVrYKZ3tj7ksM

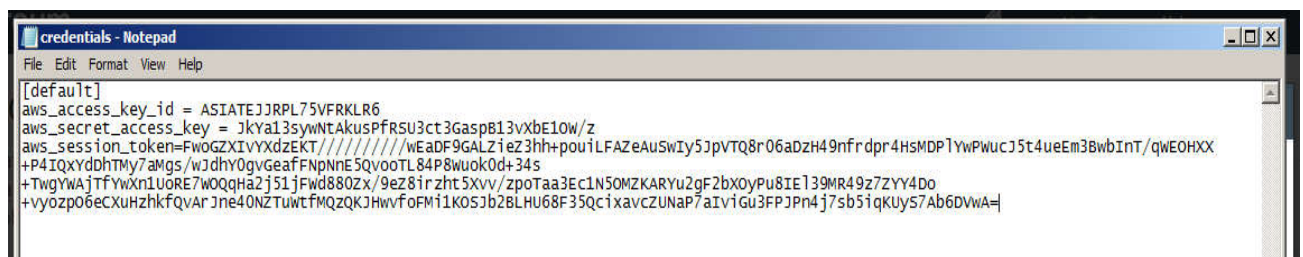
aws_session_token=FwoGZXIvYXdzEL7////////wEaDINNElb4ZGblYBpjeCLFAT1YrPfL3TRdYnysf9OyC3v6GNC1+OkzPpCImwnVe/mn4bNbD3gzOWLaaMSYdCWFSGSqm eYZddG3Mkd7uhGXBXF+o6LpEEZD0qwkLFu1 WyRx4pSfT3bShDbjvfMfGzawirXR4/

kCXIns9NC6c5TFBhdO52GH3DDRTyHIQB8gbEQH4SP5n8oxgNX6B5nXHj7wNexPIf
jPFHBsMTGPhQXU5P2qUI7iBR7abPAFM9xEYfnnNhBLmQQm4GORgJhIwNUqma2
nd9yPKI7Mw/oFMi32loXcoFa4mdfONAkWZjGT6PnseZm3ZsRHnNCQosduv/Sug0a
HIcOz25Fug=

c:\Users\Karthi\.aws\credentials



Add credential information to file



Step 5:

Step 5.1 : create the bucket in S3 using Nodejs

create_bucket_new.js

```
// CREATE BUCKET

var AWS = require('aws-sdk');

var s3bucket = new AWS.S3(); // service interface object

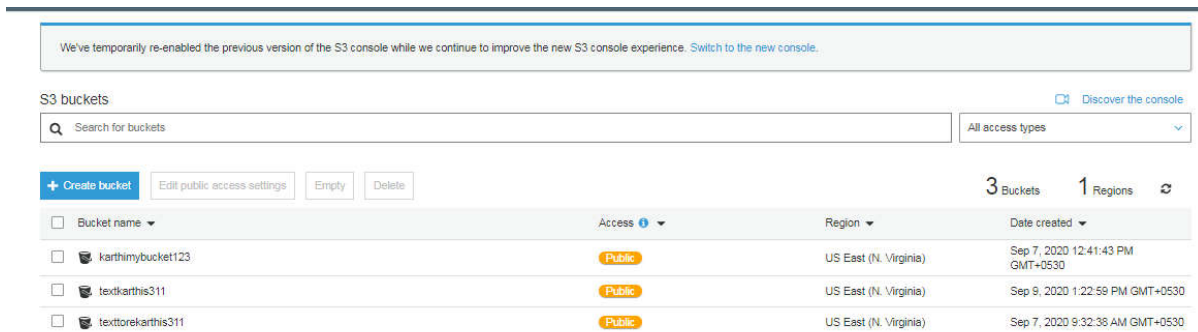
var params = {Bucket: 'textkarthis311', ACL: 'public-read'};

s3bucket.createBucket(params, function(err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.Location);
  }
});
```

Run the program in Visual Studio Code:

```
E:\CloudComputing\2020\ClassMaterial\S3 Demos\s3samples>node create_bucket_new.js
Success /textkarthis311
```

Bucket is created



Program 2: Create file in Bucket

// Insert File in bucket

Insert_bucket_file.js

```
var AWS = require("aws-sdk");

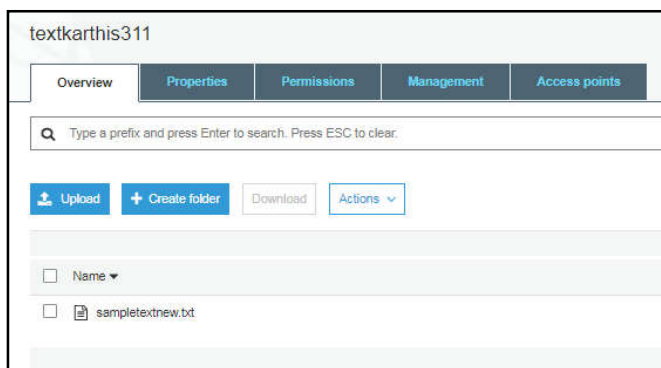
var s3bucket = new AWS.S3();

var params = {Bucket: 'textkarthis311', Key:'sampletextnew.txt', Body: 'Hello this data!'};

s3bucket.putObject(params, function(err, data) {
    if (err) {
        console.log("Error uploading data: ", err);
    } else {
        console.log("Successfully uploaded data to bucket");
    }
});
```

Run the program in Visual Studio Code:

```
E:\CloudComputing\2020\ClassMaterial\S3 Demos\s3samples>node insert_bucket_file.js
Successfully uploaded data to bucket
```



Program 3: Download a File from S3 Bucket

// Insert File in bucket
download_bucket_file.js

```
var AWS = require('aws-sdk');

var s3bucket = new AWS.S3();

var params = {Bucket: 'textkarthis311', Key: 'sampletextnew.txt'};

s3bucket.getObject(params, function(err, data) {
  if (err) {
    console.log("Error uploading data: ", err);
  } else {
    console.log("Successfully downloaded data from bucket");
    const body = Buffer.from(data.Body).toString('utf8');
    console.log(body);
  }
});
```

Output

```
E:\CloudComputing\2020\ClassMaterial\S3 Demos\s3samples>node download_bucket_file.js
Successfully downloaded data from bucket
Hello this data!
```

Program 4: Upload a text file from local drive to S3 using Nodejs

Upload.js

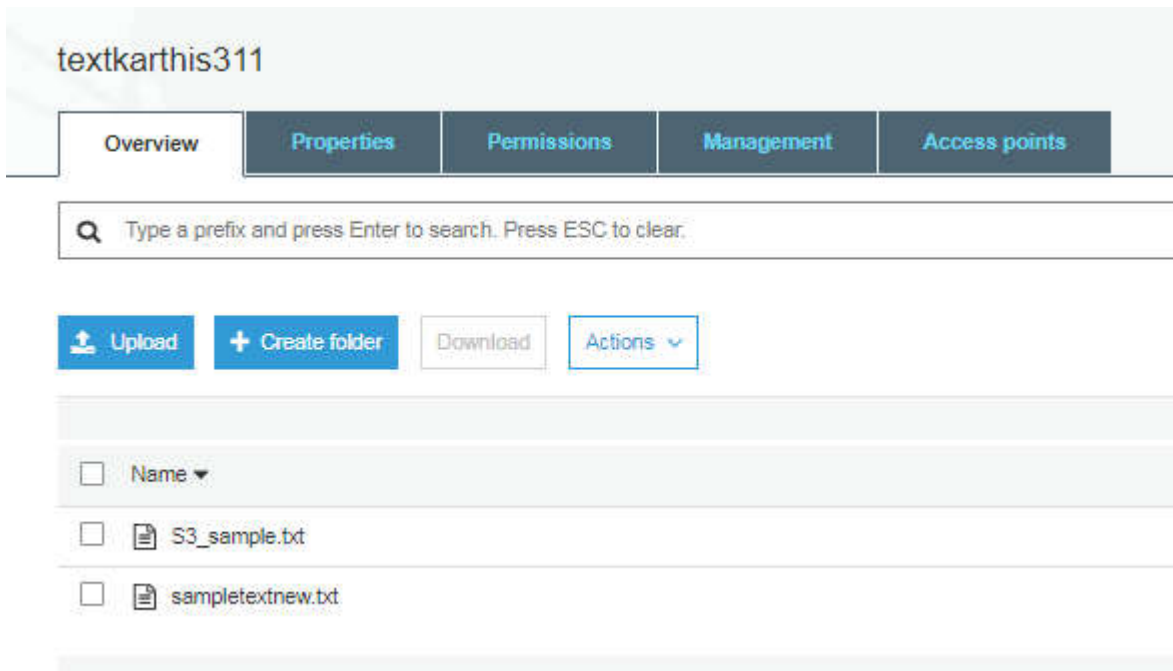
```
const fs = require('fs');
const AWS = require('aws-sdk');

const s3 = new AWS.S3();

const fileName = 'sample.txt';
const uploadFile = () => {
  fs.readFile(fileName, (err, data) => {
    if (err) throw err;
    const params = {
      Bucket: 'textkarthis311', // pass your bucket name
      Key: 'S3_sample.txt', // file will be saved
      Body: data
    };
    // console.log(data)
    s3.upload(params, function(s3Err, data) {
      if (s3Err) throw s3Err
      console.log(`File uploaded successfully at ${data.Location}`)
    });
  });
};

uploadFile();
```

```
E:\CloudComputing\2020\ClassMaterial\S3 Demos\s3samples>node upload.js
File uploaded successfully at https://textkarthis311.s3.amazonaws.com/S3_sample.txt
```



Program 5: Read a JSON File from S3

```
Read_json.js
const fs = require('fs');
const AWS = require('aws-sdk');

var s3 = new AWS.S3({apiVersion: '2006-03-01'});

var params = {Bucket: 'karthimybuck123', Key: 'users.json', ResponseContentType: 'application/json' };
const f = s3.getObject(params, function(err, data) {
  if (err) {
    console.log("Error uploading data: ", err);
  } else {
    console.log("Successfully uploaded data to bucket");

    x = data.Body.toString('utf-8');
    console.log(x);
    dat = JSON.parse(x);

    console.log(dat.length);

    for (i = 0; i < dat.length; i++) {
      console.log(dat[i].name, dat[i].age);
    }
  }
});
```

File user.json is in S3 - 'karthimybuck123' bucket

```
[{"name": "John", "age": 21, "language": ["JavaScript", "PHP", "Python"]}, {"name": "Smith", "age": 25, "language": ["PHP", "Go", "JavaScript"]}, {"name": "New User", "age": 30, "language": ["PHP", "Go", "JavaScript"]}]
```

Output

```
E:\CloudComputing\2020\ClassMaterial\S3 Demos\s3samples\Json_example>node read_json.js
Successfully uploaded data to bucket
[{"name": "John", "age": 21, "language": ["JavaScript", "PHP", "Python"]}, {"name": "Smith", "age": 25, "language": ["PHP", "Go", "JavaScript"]}, {"name": "New User", "age": 30, "language": ["PHP", "Go", "JavaScript"]}
3
John 21
Smith 25
New User 30
```

Program 6: Accessing Steams in NodeJs

```
Imagereadwrite.js
// Image Read and Write

var AWS = require("aws-sdk");
const fs = require('fs');
var path = require('path');

var s3 = new AWS.S3({apiVersion: '2006-03-01'});

var fileStream = fs.createWriteStream('E:/file1.jpg');

var s3Stream = s3.getObject({Bucket: 'karthimybuck123', Key: 'Chrysanthemum.jpg'}).createReadStream();

// Listen for errors returned by the service
s3Stream.on('error', function(err) {
    // NoSuchKey: The specified key does not exist
    console.error(err);
});

s3Stream.pipe(fileStream).on('error', function(err) {
    // capture any errors that occur when writing data to the file
    console.error('File Stream:', err);
}).on('close', function() {
    console.log('Done.');
```

Output

```
E:\CloudComputing\2020\ClassMaterial\S3 Demos\s3samples>node Imagereadwrite.js
Done.
```

File is downloaded to 'E:/file1.jpg'

Setting session tokens in Ubuntu

```
ubuntu@ip-172-31-44-211:~$ aws configure
AWS Access Key ID [*****cc]: ASIATEJJRPL732USHM6N
AWS Secret Access Key [*****ccc]:
abHEVA2KPUw2CIN7AADffRY/uD4sQX21HJbr8mw1
Default region name [None]:
Default output format [None]:
```

Configure the session token

```
ubuntu@ip-172-31-44-211:~$ aws configure set
aws_session_token
FwoGZXIVYXdzEGMaDDdB3mnBvHXQ6zpwbyLFAdGBJHz/W9lrxkvHU
wpONCHnyAbqZn+gtALWDqjN1vS6glTjbyD3RkUrgMjt1vKD/VagHO
2LUGSEtRIIY8dd6Xp96daOSAdLiTwpfYka6BA/aF3Wmmf3UMqw2Nx
1UPr90fqmYJnVkr7rb/WAS2KHKWHgVd16EZenaxMnIqa14ujr1AYN
jh5fI+KJ71NQzD00dIJ66tKuWpRpjVj6zCSyuDTaw8JXjnbuyTGz
EUHl6F3bQbiZOpogopRnSSFYRtdur8E+zj8KJ3h5/oFMi2K+ffwy8
y2SL/QHvatyHxi63dnVMidLudIoh3yv7PBIKAjZos9201x0dGOW1U
=
```

// check if the data is configured

```
ubuntu@ip-172-31-44-211:~$ aws configure list
```

Name	Value	Type	Location
------	-------	------	----------

-----	-----	-----	-----
profile	<not set>	None	None
access_key	*****HM6N	shared-credentials-file	
secret_key	*****8mw1	shared-credentials-file	
region	<not set>	None	None

check the status using ls command

```
ubuntu@ip-172-31-44-211:~$ aws s3 ls
2020-09-07 07:11:43 karthimybuck123
2020-09-10 05:54:04 karthisample12345
2020-09-09 07:52:59 textkarthis311
2020-09-10 06:10:29 textkarthis311345
2020-09-07 04:02:38 texttorekarthis311
ubuntu@ip-172-31-44-211:~$
```