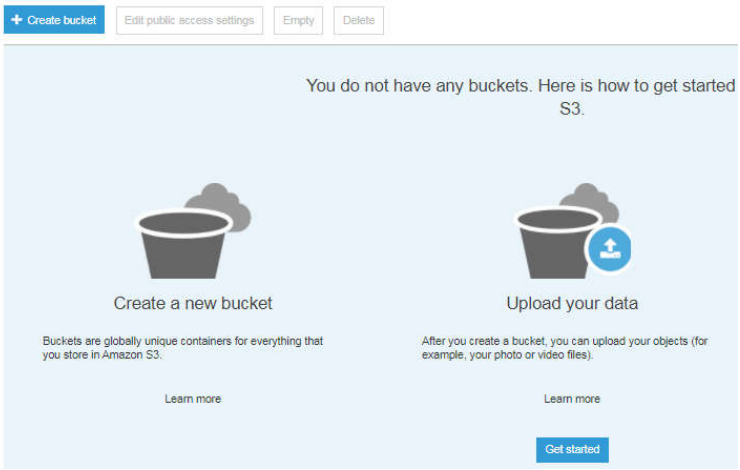
# S3

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

### Step 1: Select the S3 service – Create Bucket

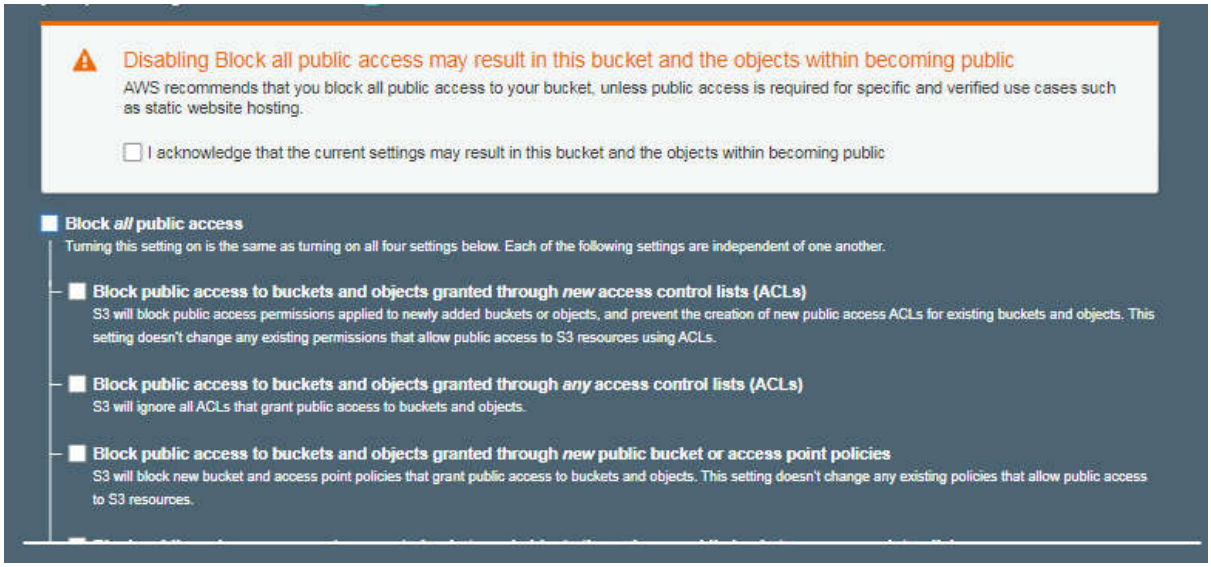



### Step 2: Specify Bucket name which is a unique name globally

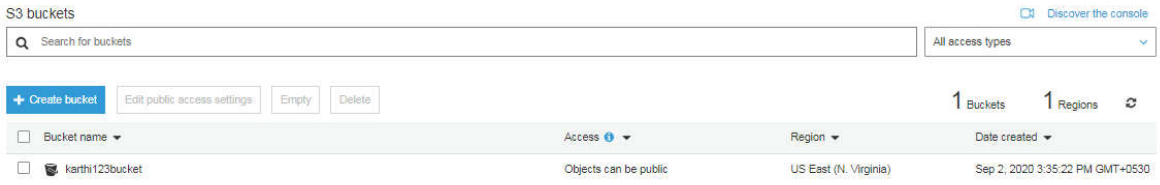

### Step 3: Set Configure options parameters to default.

### Step 4: Uncheck block all public access – click -- I acknowledge

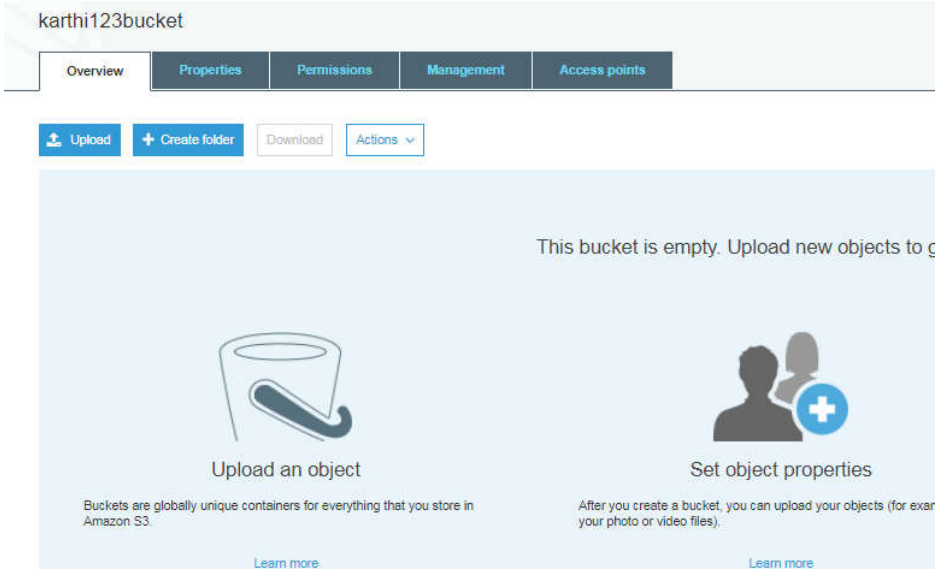
This allows access of files stored in bucket by other applications.



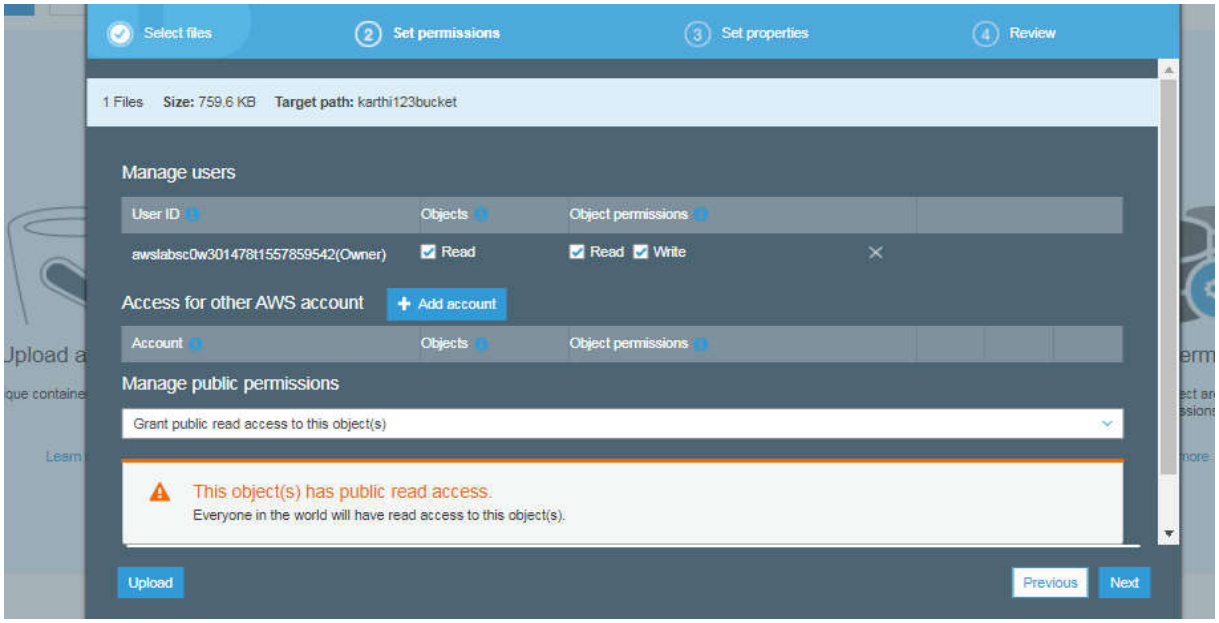

## Step 5: Review and create the bucket



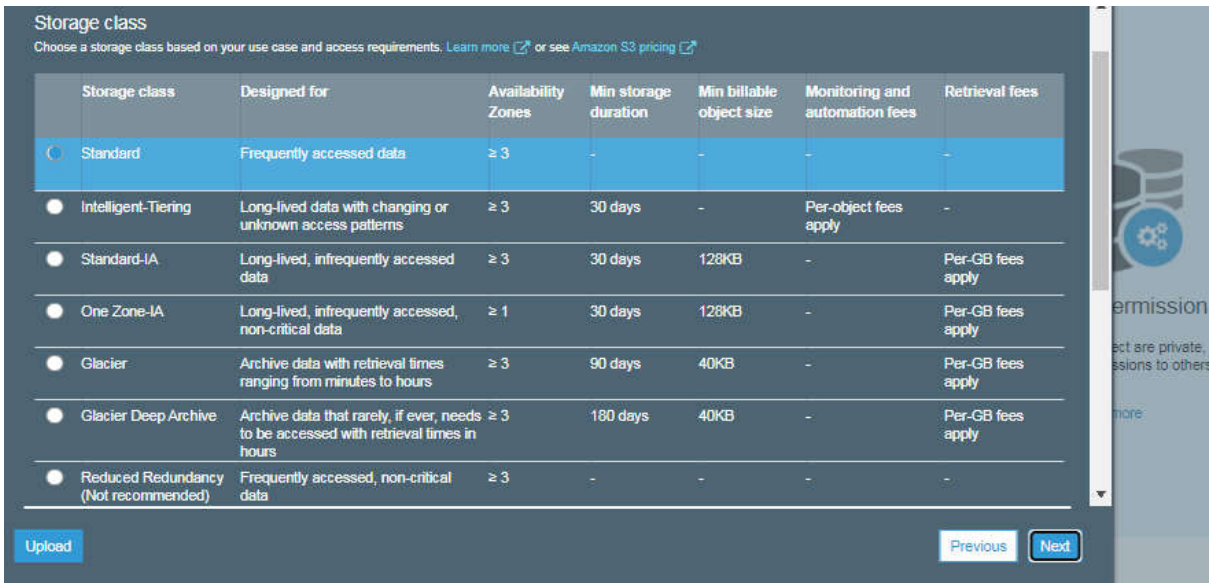

## Step 6: Upload Files in S3 bucket




## Step 7: In permission Tab set access to public mode




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




| Storage class | Designed for | Availability Zones | Min storage duration | Min billable object size | Monitoring and automation fees | Retrieval fees |
|---|---|---|---|---|---|---|
| Standard | Frequently accessed data | ≥ 3 | - | - | - | - |
| Intelligent-Tiering | Long-lived data with changing or unknown access patterns | ≥ 3 | 30 days | - | Per-object fees apply | - |
| Standard-IA | Long-lived, infrequently accessed data | ≥ 3 | 30 days | 128KB | - | Per-GB fees apply |
| One Zone-IA | Long-lived, infrequently accessed, non-critical data | ≥ 1 | 30 days | 128KB | - | Per-GB fees apply |
| Glacier | Archive data with retrieval times ranging from minutes to hours | ≥ 3 | 90 days | 40KB | - | Per-GB fees apply |
| 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 |
| Reduced Redundancy (Not recommended) | Frequently accessed, non-critical data | ≥ 3 | - | - | - | - |

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

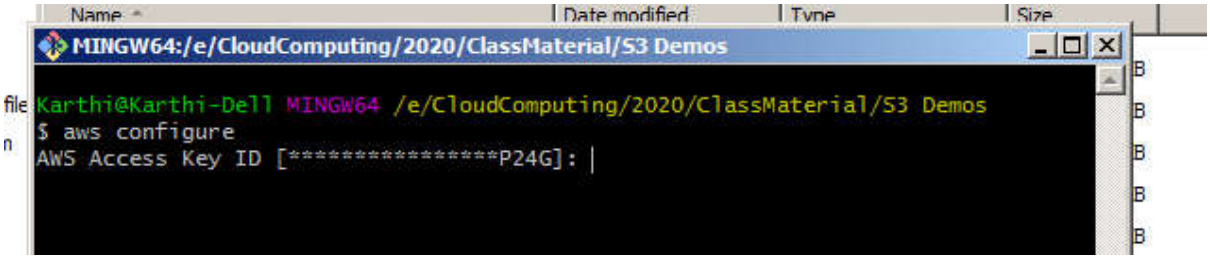
**Object URL**
https://karthi123bucket.s3.amazonaws.com/Penguins.jpg

# S3 - commands with the AWS CLI

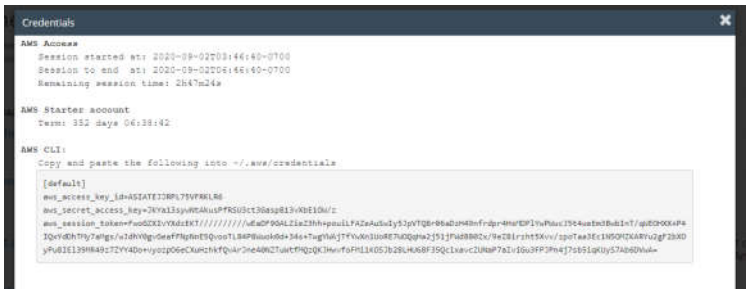
The AWS Command Line Interface (CLI) is a unified tool to manage your AWS services.

**Basic S3 commands in AWS**

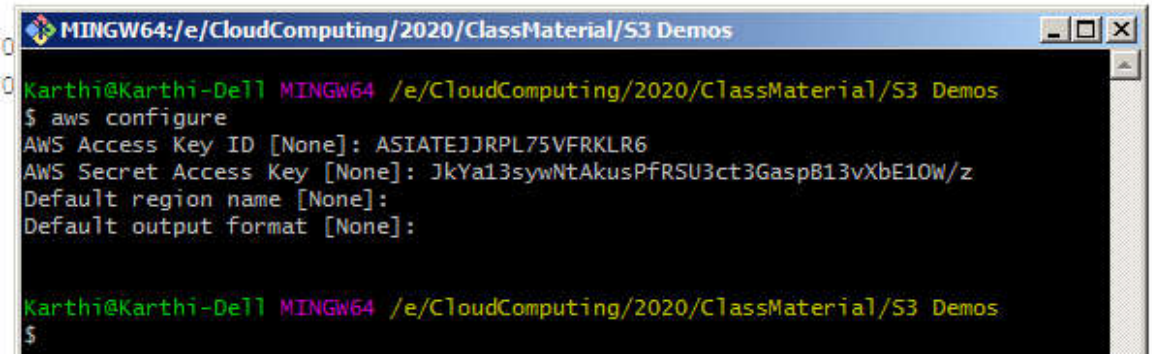
Step 1: Go to folder where you have files for upload and open Gitbash from the current directory
My directory is e:/CloudComputing/2020/Classmaterials/s3Demos




Step 2: Configure AWS CLI to connect to AWS




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

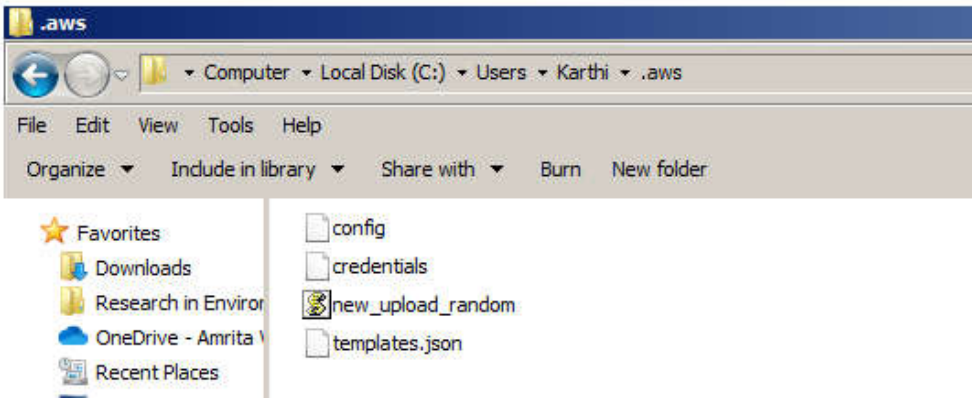



Type a sample Command: aws s3 ls

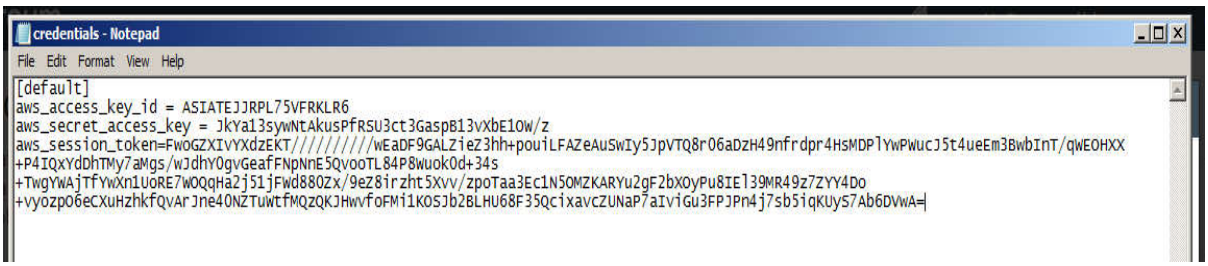
List all the buckets in S3

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

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

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

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




Add aws-session-token details in the file




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/s3sampl
es
$ 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
$ 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"
}
```

**You are in nodeS3 Directory**

### 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

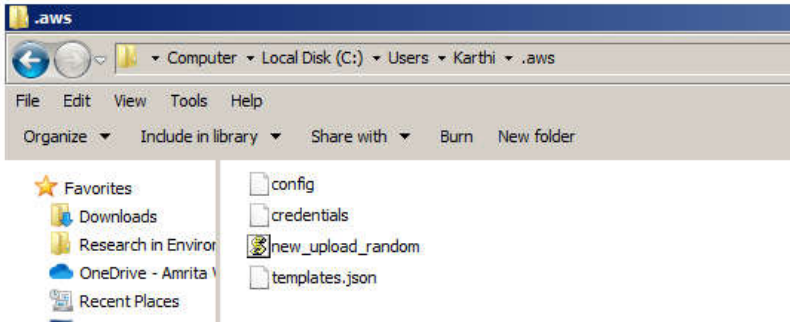
Go to workbench – Click Account details

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

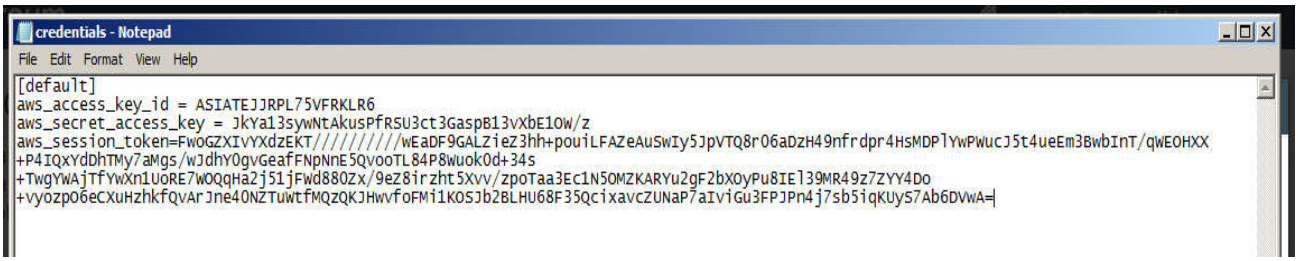
```
[default]
aws_access_key_id=ASIATEJJRPL7SYPEWSEH
aws_secret_access_key=tJ/7MC3NaIUQ0MYD5bDNAlYWCPiJVrYKZ3tj7ksM
aws_session_token=FwoGZXIvYXdzEL7//////////wEaDINNElb4ZGblYBpjeCLFAT1YrPf
L3TRdYnysf9OyC3v6GNC1+OkzPpClmwnVe/mn4bNbD3gzOWLaaMSYdCWFSGSqm
eYZddG3Mkd7uhGXBXF+o6LpEEZD0qwkLFu1WyRx4pSfT3bShDbjvfMfGzawirXR4/
```

kCXIns9NC6c5TFBhdO52GH3DDRTyHlQB8gbEQH4SP5n8oxgNX6B5nXHj7wNexPIf
jPFHBsMTGPhQXU5P2qUI7iBR7abPAFM9xEYfnnNhBLmQQm4GORgJhIwNUqma2
nd9yPKI7Mw/oFMi32loXcoFa4mdfONAkwWZjGT6PnseZm3ZsRHnNCQosdjuv/Sug0a
HIcOz25Fug=

c:\Users\Karthi\.aws\credentials




Add credential information to file




Step 5:
Step 5.1 : create the bucket in S3 using Nodejs

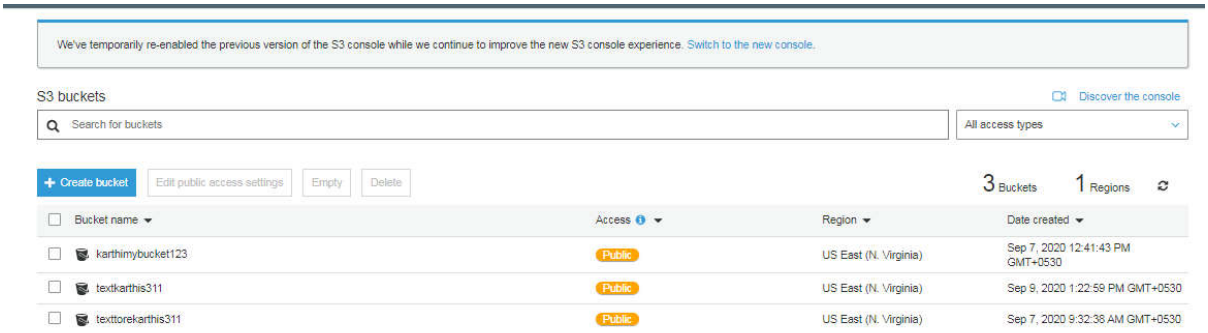
create_bucket_new.js

```
// 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

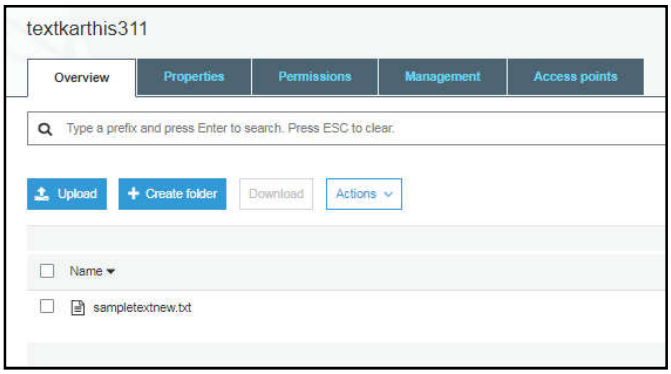



# Program 2: Create file in Bucket

```
// 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 dowloaded data from bucket");
            const body = Buffer.from(data.Body).toString('utf8');
            console.log(body);
        }
    });
```

## Output

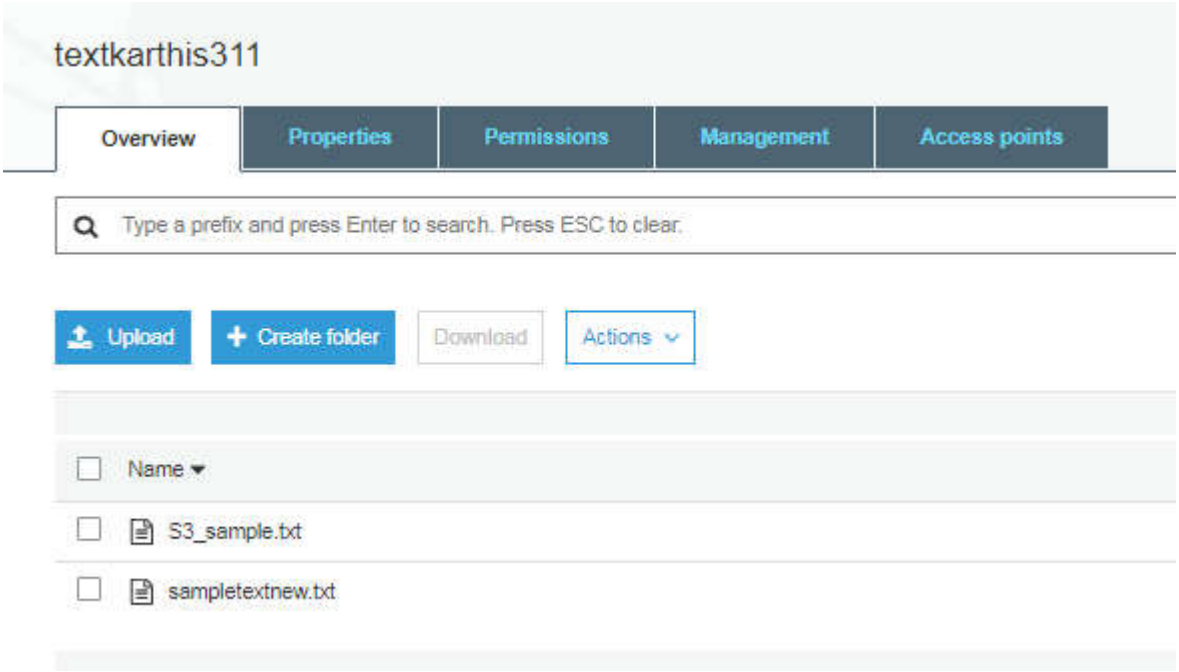
```
E:\CloudComputing\2020\ClassMaterial\S3 Demos\s3samples>node download_bucket_file.js
Successfully dowloaded data from bucket
Hello this data!
```

## 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: 'karthimybucket123', 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 -'karthimybucket123' 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":3
"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: 'karthimybucket123', 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
1UPr9OfqmYJnVkr7rb/WAS2KHKWHgVdl6EZenaxMnIqa14ujr1AYN
jH5fI+KJ71NQzDOOdIJ66tKuwDpRpjVj6zCSyuDTaw8JXjnbuyTGz
EUHl6F3bQbizOpOgopRnSSFYRtdur8E+zj8KJ3h5/oFMi2K+ffWy8
y2SL/QHVatyHxi63dnVMidLudIoh3yv7PBIKAjZos92OlxOdGOW1U
=
```

// 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 karthimybucket123
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:~$