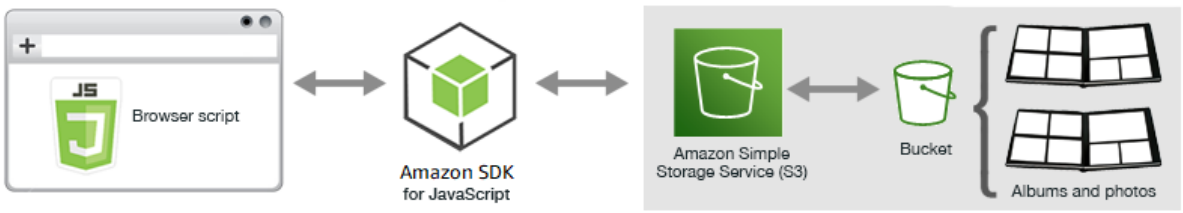
S3 Photo Application

Lab Overview

LAB OVERVIEW:

Today we are going to use a range of AWS services to build a JavaScript + HTML Browser Application used to Manage Photos (Upload/Download) in the Cloud.

In this example, a simple HTML page provides a browser-based application for creating photo albums in an Amazon S3 bucket into which a user can upload photos. The application also lets you delete photos and albums that you add.



The browser script uses the SDK for JavaScript to interact with an Amazon S3 bucket. It uses the following methods of the Amazon S3 client class from the SDK to enable respective functionality in the photo album application:

- PutObjectCommand
- DeleteObjectCommand
- ListObjectsCommand

Requirements

- ***AWS Free Tier Account***
- ***Basic Exposure to the AWS Console and completion of prior days learning.***

Resources

Please download the lab resources zip file here.

Exercise Overview

Exercise 1 – Create the S3 Bucket to host the website code
Exercise 2 – Create the Amazon Cognito Identity Pool
Exercise 3 – Compile the Code using CloudShell
Exercise 4 – Create the Static Website

Exercise 1 - Create the S3 Bucket

Task 1 – Create the S3 Bucket to host the website code.

Create the S3 bucket and populate it with the HTML / CSS code.

1. Head over to the Amazon S3 Console and click 'Create Bucket'.
2. First, we will create the S3 Bucket. Call it '***myphotostore-123456***, with the numbers appended being a random string of numbers individual to yourself.
3. Select 'Create Bucket'.
4. Go to the bucket's Permissions tab to edit Block public access and uncheck the checkbox named 'Block all public access'. And save changes.
5. We need to add a bucket policy. Copy the code from the lab resources called 'S3-Bucket-permissions.json' (in the permissions-code directory) and ensure you edit the ARN under the "Resource".

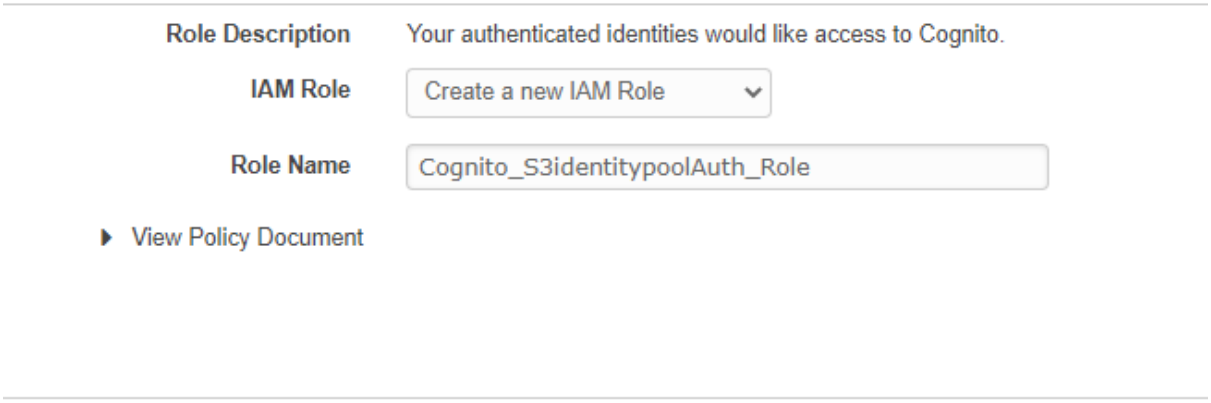
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicReadWriteObjects",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject",
                "s3:PutObject",
                "s3:DeleteObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::MY-BUCKET-HERE/*",
                "arn:aws:s3:::MY-BUCKET-HERE"
            ]
        }
    ]
}
```

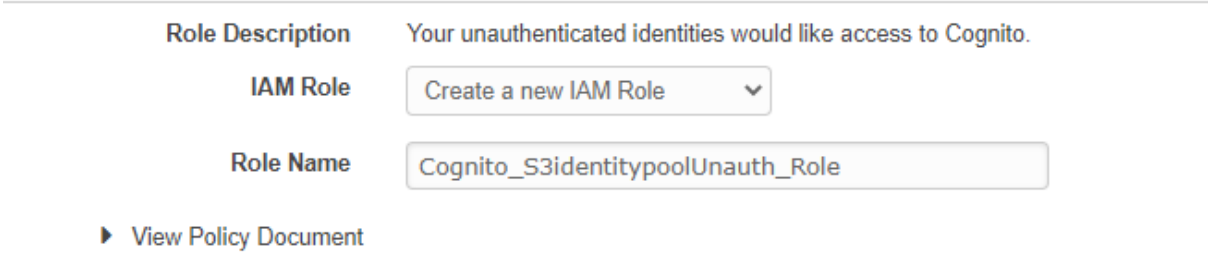
6. Also update CORS settings, copy/pasting the code from the CORS.json file.
7. Click save changes on the S3 bucket – and that completes the S3 bucket setup.

Exercise 2 - Create the Amazon Cognito Identity Pool

Task 1 – Create the Amazon Cognito Identity Pool using Federated Identities with access enabled for unauthenticated users in the same Region as the Amazon S3 bucket.

1. Sign into the Amazon Cognito console, choose the option to grant access to AWS services, or select 'Federated identities' on the left-hand navigation. Create an identity pool.
2. Name the identity pool 'S3-identity-pool'.
3. You will be prompted for access to your AWS resources – tick the box that says 'Enable access to unauthenticated identities'.
4. Click 'Create pool'.
5. Choose Allow to create the two default roles associated with your identity pool—one for unauthenticated users and one for authenticated users. Make a note of the Identity Pool ID.

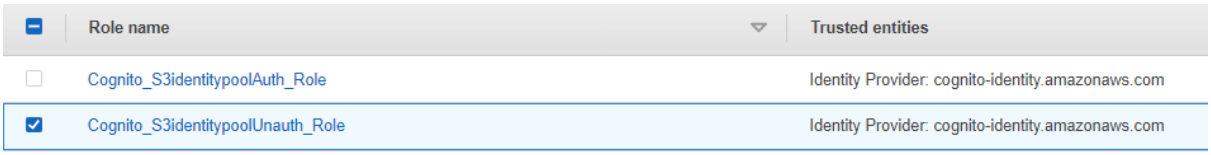




6. Note the identity pool ID in the output presented:

```
// Initialize the Amazon Cognito credentials provider
CognitoCachingCredentialsProvider credentialsProvider = new CognitoCachingCredentialsProvider(
    getApplicationContext(),
    "us-east-1:b09a4742-03ee-452d-9106-4b3a0fcd6fd5", // Identity pool ID
    Regions.US_EAST_1 // Region
);
```

7. Now in the IAM console, we will edit the IAM role created by Amazon Cognito for unauthenticated users.
8. Find the role for unauthenticated users, which should be called 'Cognito_S3identitypoolUnauth_Role'. Select the role.



9. Under ‘Add Permissions’ click ‘Create Inline policy’.
10. Add the JSON from the file ‘unauth_role.json’ file and paste it into the JSON editor. Replace the ‘BUCKETNAME’ with the correct S3 bucket name.
11. Click review policy and call it ‘unauth_role’.

Exercise 3 – Compile the Code using CloudShell

Task 1 – Compile the code using CloudShell, which is a browser based CLI.

1. Navigate to the PhotoAppCode/src directory within the resource download.
2. Open the PhotoApp.ts file with Visual Studio Code.
3. On line 34 make sure the Region is set correctly:

```
33    // Set the AWS Region
34    const REGION = "us-east-1"; //REGION
```

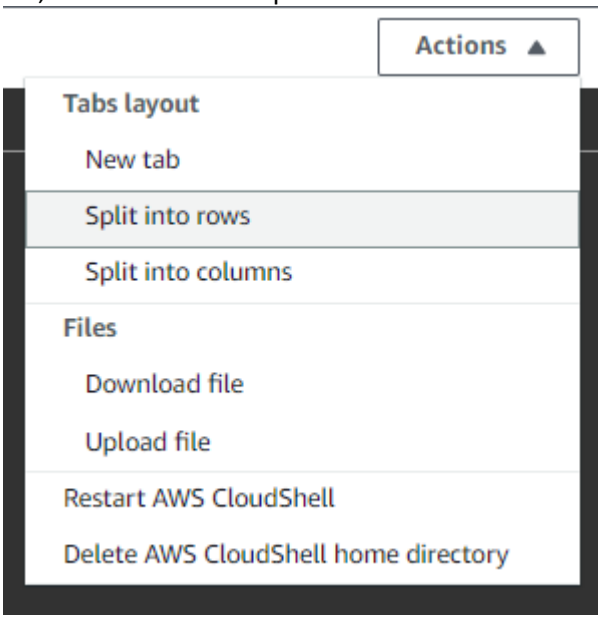
4. On line 41 update the identity pool ID:

```
40        client: new CognitoIdentityClient({ region: REGION }),
41        identityPoolId: "us-east-1:9060a572-6d2a-44f4-9a71-3614ce0c840d", // IDENTITY_POOL_ID
```

5. On line 45 update your bucket name:

```
45    const albumBucketName = "photoapp2-store"; //BUCKET_NAME
46
```

6. Save and close the file.
7. Zip up the PhotoAppCode directory to a file named PhotoAppCode.zip
8. In the AWS console, open AWS CloudShell.
9. Once launched, click actions and Upload File.



10. Upload the zipped PhotoAppCode.zip file.

11. Unzip the folder by running the command

```
unzip PhotoAppCode.zip
```

12. Change into the newly extracted PhotoAppCode folder with the following command

```
cd PhotoAppCode
```

13. We will now install the package manager NPM by running:

```
npm install
```

14. If prompted to do so, run the command to upgrade npm (it will provide the command to run, and you must use 'sudo').

15. Then run this command to compile the code:

```
npm run build
```

16. Your code should now be complied.

17. Copy the files needed for the static website to a new folder:

```
mkdir PhotoAppDownload
cp -r dist PhotoAppDownload/
cp favicon.ico PhotoAppDownload/
cp PhotoApp.html PhotoAppDownload/
cp style.css PhotoAppDownload/
```

18. Change to the PhotoAppDownload directory and run the following code to zip up the code into a folder called PhotoAppDownload.zip

```
zip -r PhotoAppDownload.zip *
```

19. Then go to the actions section in the top right, click download code and enter PhotoAppCode/PhotoAppDownload/PhotoAppDownload.zip, and the file should download to your computer.

Exercise 4 – Create the Static Website on S3

Task 1 – Create the S3 Bucket for Hosting the Code

1. Once you have downloaded and unzipped the code you can test the code by clicking on the PhotoApp.html document within the folder and you should be able to upload an image.
2. Check that the folders / images are created in the bucket created earlier.
3. Next, we will create a new S3 bucket and call it 'myphotoapp-hosting-bucket-123456'.

4. Clear ‘Block all public access’ and accept the changes.
5. Upload the contents of the ‘PhotoAppDownload’ folder to the bucket.
6. Go to the Properties section, scroll down, and enable static website hosting.
7. Enter PhotoApp.html as the index document.
8. Open the ‘website-hosting-policy.json’ file from the resource download and edit the bucket ARN.
9. Add the code as a bucket policy.
10. Go to the static website endpoint to test that it is working correctly.