



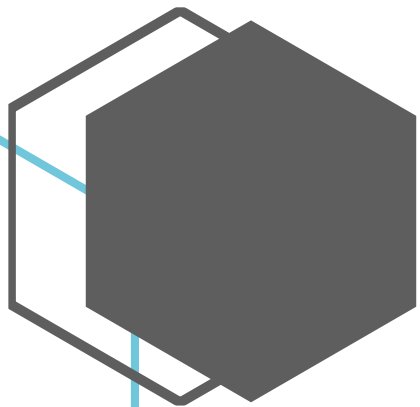
CSCI 5410

Assignment 4 – Part C

Name: Benny Daniel Tharigopala

Banner ID: B00899629

GitLab URL: <https://git.cs.dal.ca/benny/csci5410> B00899629 Benny Tharigopala



Event-driven Serverless application with AWS Lambda & Comprehend

S3Buckets Operations:

Amazon S3

Buckets

Access Points

Object Lambda Access Points

Multi-Region Access Points

Batch Operations

Access analyzer for S3

Block Public Access settings for this account

Storage Lens

Dashboards

AWS Organizations settings

Feature spotlight

AWS Marketplace for S3

Amazon S3 > Buckets

Account snapshot

Last updated: Jul 6, 2022 by Storage Lens. Metrics are generated every 24 hours. [Learn more](#)

[View Storage Lens dashboard](#)

Total storage: 43.0 KB Object count: 24 Avg. object size: 1.8 KB

You can enable advanced metrics in the "default-account-dashboard" configuration.

Buckets (2) [Info](#)

Buckets are containers for data stored in S3. [Learn more](#)

< 1 > ⚙

| | Name | AWS Region | Access | Creation date |
|-----------------------|----------------|----------------------------|-----------------------|-------------------------------------|
| <input type="radio"/> | source00899629 | US West (Oregon) us-west-2 | Objects can be public | June 23, 2022, 13:58:01 (UTC-03:00) |
| <input type="radio"/> | tag00899629 | US West (Oregon) us-west-2 | Objects can be public | June 23, 2022, 13:58:02 (UTC-03:00) |

Amazon S3

Buckets

Access Points

Object Lambda Access Points

Multi-Region Access Points

Batch Operations

Access analyzer for S3

Block Public Access settings for this account

Storage Lens

Dashboards

AWS Organizations settings

Feature spotlight

AWS Marketplace for S3

Amazon S3 > Buckets

Account snapshot

Last updated: Jul 6, 2022 by Storage Lens. Metrics are generated every 24 hours. [Learn more](#)

[View Storage Lens dashboard](#)

Total storage: 43.0 KB Object count: 24 Avg. object size: 1.8 KB

You can enable advanced metrics in the "default-account-dashboard" configuration.

Buckets (4) [Info](#)

Buckets are containers for data stored in S3. [Learn more](#)

< 1 > ⚙

| | Name | AWS Region | Access | Creation date |
|-----------------------|----------------------------|----------------------------|-----------------------|-------------------------------------|
| <input type="radio"/> | destinationbuckets00899629 | US West (Oregon) us-west-2 | Objects can be public | July 7, 2022, 11:16:29 (UTC-03:00) |
| <input type="radio"/> | source00899629 | US West (Oregon) us-west-2 | Objects can be public | June 23, 2022, 13:58:01 (UTC-03:00) |
| <input type="radio"/> | tag00899629 | US West (Oregon) us-west-2 | Objects can be public | June 23, 2022, 13:58:02 (UTC-03:00) |
| <input type="radio"/> | twitterdata00899629 | US West (Oregon) us-west-2 | Objects can be public | July 7, 2022, 11:16:28 (UTC-03:00) |

Lambda Functions:

Function – “extractFeatures”:

[Alt+S]

Lambda > Functions

Functions (0) Last fetched 20 seconds ago

< 1 > ⚙

| | Function name | Description | Package type | Runtime | Last modified |
|------------------------------|---------------|-------------|--------------|---------|---------------|
| There is no data to display. | | | | | |

Lambda > Functions > Create function

Create function info

Choose one of the following options to create your function.

☒ Author from scratch
Start with a simple Hello World example.

☐ Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.

☐ Container image
Select a container image to deploy for your function.

☐ Browse serverless app repository
Deploy a sample Lambda application from the AWS Serverless Application Repository.

Basic information

Function name info
Enter a name that describes the purpose of your function.

Runtime info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Architecture info
Choose the instruction set architecture you want for your function code.
☒ x86_64
☐ arm64

Permissions info
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

▼ Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.


- ☐ Create a new role with basic Lambda permissions
- ☒ Use an existing role
- ☐ Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

View the LabRole role on the IAM console.

Add trigger

Trigger configuration

 **S3**
aws storage

Bucket
Please select the S3 bucket that serves as the event source. The bucket must be in the same region as the function.

Event type
Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.

Prefix - optional
Enter a single optional prefix to limit the notifications to objects with keys that start with matching characters.

Suffix - optional
Enter a single optional suffix to limit the notifications to objects with keys that end with matching characters.

Recursive invocation
If your function writes objects to an S3 bucket, ensure that you are using different S3 buckets for input and output. Writing to the same bucket increases the risk of creating a recursive invocation, which can result in increased Lambda usage and increased costs. [Learn more](#)

☒ I acknowledge that using the same S3 bucket for both input and output is not recommended and that this configuration can cause recursive invocations, increased Lambda usage, and increased costs.

Lambda will add the necessary permissions for AWS S3 to invoke your Lambda function from this trigger. [Learn more](#) about the Lambda permissions model.

Cancel

Add

processTweetFile Throttle Copy ARN Actions

✓ The trigger twitterdatab00899629 was successfully added to function processTweetFile. The function is now receiving events from the trigger.

▼ Function overview [Info](#)

processTweetFile
Layers (0)

Description

-

Last modified

34 minutes ago

Function ARN

arn:aws:lambda:us-west-2:789760207353:function:processTweetFile

Function URL [Info](#)

-

+ Add destination

S3

+ Add trigger

Code Test Monitor **Configuration** Aliases Versions

General configuration

Triggers

Permissions

Destinations

Function URL

Environment variables

Tags

var

Triggers (1)

Find triggers

1

Trigger

S3: twitterdatab00899629

arn:aws:s3::twitterdatab00899629

▼ Details

Bucket arn: arn:aws:s3::twitterdatab00899629

Event type: s3:ObjectCreated:*

Notification name: 6fa7ff96-8f64-478c-9ece-79994041d300

Fix errors Edit Delete Add trigger

Snips of Operations in Chronological Order:

1. Upload Tweet File to the S3 Bucket

```

public class UploadTweetFile {
    public static void main(String[] args) throws InterruptedException {
        CreateS3Buckets obj = new CreateS3Buckets();
        String id = Credentials.aws_access_key_id;
        String key = Credentials.aws_secret_access_key;
        String token = Credentials.aws_session_token;
        BasicSessionCredentials sessionCredentials = new BasicSessionCredentials(id, key, token);
        AmazonS3 s3cli = AmazonS3ClientBuilder.standard().withCredentials(new AWSStaticCredentialsProvider(sessionCredentials)).withRegion(Region.US_WEST_2);
        String s3Bucket1 = "twitterdatab00899629";
        File folderPath = new File( "pathname: ./src/tweets");
        File[] allFiles = folderPath.listFiles();
        for (File f : allFiles) {
            s3cli.putObject(s3Bucket1, f.getName(), f);
            System.out.println("The Tweet File - " + f.getName() + " has been uploaded to the S3 Bucket - " + s3Bucket1);
        }
    }
}

```

Run: Assignment-3 (1) x

"C:\Program Files\jdk-17-windows-x64-bin\jdk-17.0.3.1\bin\java.exe" ...

Jul. 07, 2022 11:21:35 A.M. com.amazonaws.util.Base64 warn

WARNING: JAXB is unavailable. Will fallback to SDK implementation which may be less performant. If you are using Java 9+, you will need to include javax.xml.bind:jaxb-api as a dep

The Tweet File - file_mongo_tweets.txt has been uploaded to the S3 Bucket - twitterdatab00899629

Process finished with exit code 0

2. Tweet File in the Source S3 Bucket

Amazon S3 > Buckets > twitterdatab00899629

twitterdatab00899629 [Info](#)

[Objects](#) | [Properties](#) | [Permissions](#) | [Metrics](#) | [Management](#) | [Access Points](#)

Objects (1)
Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

[Refresh](#) [Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) [Actions](#) [Create folder](#) [Upload](#)

| <input type="checkbox"/> | Name | Type | Last modified | Size | Storage class |
|--------------------------|---------------------------------------|------|------------------------------------|----------|---------------|
| <input type="checkbox"/> | file_mongo_tweets.txt | txt | July 7, 2022, 11:21:37 (UTC-03:00) | 350.7 KB | Standard |

3. CloudWatch Log for the Tweet File Upload

CloudWatch

Log groups > /aws/lambda/processTweetFile > 2022/07/07/[!\$LATEST]5f5ddc4b19524db686afbe63a3fa0c23

Log events
You can use the filter bar below to search for and match terms, phrases, or values in your log events. [Learn more about filter patterns](#)

☐ View as text [Refresh](#) [Actions](#) [Create Metric Filter](#)

[Clear](#) [1m](#) [30m](#) [1h](#) [12h](#) [Custom](#) [Filter icon](#)

| Timestamp | Message |
|--|--|
| No older events at this moment. Retry | |
| 2022-07-07T13:03:30.090-03:00 | START RequestId: 45f85b00-b341-49a3-a2f5-3c2b8b56f0ef Version: \$LATEST |
| 2022-07-07T13:03:33.099-03:00 | END RequestId: 45f85b00-b341-49a3-a2f5-3c2b8b56f0ef |
| 2022-07-07T13:03:33.099-03:00 | REPORT RequestId: 45f85b00-b341-49a3-a2f5-3c2b8b56f0ef Duration: 3000.34 ms Billed Duration: 3000 ms Memory Size: 128 MB Max Memory Used: 78 MB Init Du... |
| 2022-07-07T13:03:33.099-03:00 | 2022-07-07T16:03:33.0992 45f85b00-b341-49a3-a2f5-3c2b8b56f0ef Task timed out after 3.01 seconds |
| 2022-07-07T13:03:36.889-03:00 | START RequestId: e3a5fe06-f5bc-4d33-999e-c259602f9683 Version: \$LATEST |
| 2022-07-07T13:03:39.897-03:00 | END RequestId: e3a5fe06-f5bc-4d33-999e-c259602f9683 |
| 2022-07-07T13:03:39.897-03:00 | REPORT RequestId: e3a5fe06-f5bc-4d33-999e-c259602f9683 Duration: 3005.93 ms Billed Duration: 3000 ms Memory Size: 128 MB Max Memory Used: 35 MB |
| 2022-07-07T16:03:39.8962 e3a5fe06-f5bc-4d33-999e-c259602f9683 | Task timed out after 3.01 seconds |
| No newer events at this moment. Auto retry paused . Resume | |

4. Processed Tweets & Polarities file in the destination bucket

destinationbucketb00899629 [Info](#)

[Objects](#) | [Properties](#) | [Permissions](#) | [Metrics](#) | [Management](#) | [Access Points](#)

Objects (1)
Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

[Refresh](#) [Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) [Actions](#) [Create folder](#) [Upload](#)

| <input type="checkbox"/> | Name | Type | Last modified | Size | Storage class |
|--------------------------|--|------|------------------------------------|--------|---------------|
| <input type="checkbox"/> | file_mongo_tweets.json | json | July 7, 2022, 15:06:25 (UTC-03:00) | 1.0 KB | Standard |

5. Tweets and Polarities File – Contents

- Buckets
 - Access Points
 - Object Lambda Access Points
 - Multi-Region Access Points
 - Batch Operations
 - Access analyzer for S3
-
- Block Public Access settings for this account
-
- Storage Lens**
- Dashboards
 - AWS Organizations settings
-
- Feature spotlight **3**
-
- AWS Marketplace for S3

Bytes returned: 1038 B

```
{
  "_1": [
    {
      "Tweet Text": " RT Port au Prince fans takes to the streets to celebrate Haiti s Quarterfinal Gold Cup win against Canada ",
      "Polarity": "NEUTRAL"
    },
    {
      "Tweet Text": "Canada thank you for a successful StrokeMonth ",
      "Polarity": "POSITIVE"
    },
    {
      "Tweet Text": "Thanks to FAST more people in Canada are recognizing the signs of ",
      "Polarity": "POSITIVE"
    },
    {
      "Tweet Text": "Authorities say they plan to return the artifacts in the near future and are working to determine who is criminally ",
      "Polarity": "NEUTRAL"
    },
    {
      "Tweet Text": "Many families travel to a cottage or lake house to enjoy the Canada Day long weekend If you have a First Aid Kit a ",
      "Polarity": "NEUTRAL"
    },
    {
      "Tweet Text": "Arnprior is setting fireworks off as they do each year from the island below the bridge at Hydro Park ",
      "Polarity": "NEUTRAL"
    },
    {
      "Tweet Text": "Canada Day Open Closed Fireworks",
      "Polarity": "NEUTRAL"
    },
    {
      "Tweet Text": " Barrie Muskoka SimcoeCounty",
      "Polarity": "NEUTRAL"
    },
    {
      "Tweet Text": " ",
      "Polarity": "NEUTRAL"
    },
    {
      "Tweet Text": " Chatfield Canada sucks",
      "Polarity": "NEGATIVE"
    }
  ]
}
```

Code Blocks

CreateS3Bucket.java [\[1-4\]](#)

```
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicSessionCredentials;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;

public class CreateS3Buckets
{
    public static void createBucket(AmazonS3 s3cli,String bucketName)
    {
        var it = s3cli.createBucket(bucketName);
    }

    public static void main(String[] args)
    {
        String id = Credentials.aws_access_key_id;
        String key = Credentials.aws_secret_access_key;
        String token = Credentials.aws_session_token;
```

```

        BasicSessionCredentials sessionCredentials = new
BasicSessionCredentials(id,key,token);
        AmazonS3 s3cli = AmazonS3ClientBuilder.standard().withCredentials(new
AWSStaticCredentialsProvider(sessionCredentials)).withRegion(Regions.DEFAULT_REGION)
.build();
        createBucket( s3cli," twitterdatab00899629");
        createBucket( s3cli," destinationbucketb00899629");

    }
}

```

UploadTweetFile.java

```

import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicSessionCredentials;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import java.io.File;
import java.util.concurrent.TimeUnit;

public class UploadTweetFile {
    public static void main(String[] args) throws InterruptedException
    {
        CreateS3Buckets obj = new CreateS3Buckets();
        String id = Credentials.aws_access_key_id;
        String key = Credentials.aws_secret_access_key;
        String token = Credentials.aws_session_token;
        BasicSessionCredentials sessionCredentials = new
BasicSessionCredentials(id,key,token);
        AmazonS3 s3cli = AmazonS3ClientBuilder.standard().withCredentials(new
AWSStaticCredentialsProvider(sessionCredentials)).withRegion(Regions.US_WEST_2).build();

        String s3Bucket1 = "twitterdatab00899629";
        File folderPath = new File("./src/tweets");
        File[] allFiles = folderPath.listFiles();
        for (File f : allFiles)
        {
            s3cli.putObject(s3Bucket1, f.getName(), f);
            System.out.println("The Tweet File - " + f.getName() + " has been
uploaded to the S3 Bucket - " + s3Bucket1);
        }
    }
}

```

ProcessTweetFile.py (Lambda Function to Process Tweet Files and Invoke AWS Comprehend on the Tweet Texts)[\[5-10\]](#)

```

import urllib.parse
import re

```

```

import json
import urllib.parse
import re
import json
import boto3
import logging

def lambda_handler(event, context):
    s3Client = boto3.client("s3")
    s3Upload = boto3.resource('s3')
    comprehendClient = boto3.client('comprehend')

    data = []

    destinationBucket = "destinationbucketb00899629"
    sourceBucket = event['Records'][0]['s3']['bucket']['name']
    key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'],
encoding='utf-8')

    json_file_name = key.split('.')
    tweets = s3Client.get_object(Bucket=sourceBucket,
Key=key)['Body'].read().decode("utf-8")

    tweets = re.sub(r'[A-Za-z0-9]*@[A-Za-z]*\.[A-Za-z0-9]*', "", tweets)

    tweetsWithoutEmojis = re.compile("[\u0001F600-\u0001F64F"]+", flags=re.UNICODE)

    tweets = tweetsWithoutEmojis.sub(r'', tweets)

    tweets = re.sub(r"^[a-zA-Z0-9\n]+", ' ', tweets)

    count = 0

    for tweet in tweets.splitlines():
        if len(tweet)>0:
            sentiment=comprehendClient.detect_sentiment(Text=tweet,LanguageCode='en')['
Sentiment']
            data.append({'Tweet Text': tweet,'Polarity':sentiment})
            count = count + 1

        if count == 10:
            json_str = json.dumps(data)
            s3Upload.Object(destinationBucket, json_file_name[0]+ ".json").put(Body =
(json_str))

```



```
print(data)
count = 0

json_str = json.dumps(data)

print(json_str)
```

Credentials.java

```
public class Credentials
{
    public static String aws_access_key_id="ASIA3PYLANH44TJNTPFI";
    public static String
    aws_secret_access_key="QcReBN+F0n/dDqBqXbE7HLmmQkhHd7MInmsNP/Mn";
    public static String
    aws_session_token="FwoGZXIvYXdzEHUaDKSRjb6kVF9lk+/q0iLAAXd9WYWVu0z0W/12qKuUXbkGQMEda
    QzgGP80N4U9ww3GCiQXYLSDDba9monVzbIviCi1UutFnPeAhl40FaSVw27Bdb0tayZ2dQA+K53TKZKYM5DYN
    +eDh51Tz9tYwjoB7wcDB1vlyXu/TKWw5JfsGR154L93pA0tHod0cKQlVRoCPBCMI8+Hd+0S9QSYAf3vpPY1q
    U1Hgmq/8iJC3aCXBCLf9PsUUEoA9w3GbyFQtlkQrXFylLNBfCHwLqSz+OtXAyiMhdOVbjItLbecv2rApPahR
    v/JH6eH0gyOY24+iEowQBudofl7ni/joEv78yMj4+LpFwLz";
}
```

Citations

- [1] “AWS SDK for Java.” *Amazon Web Services, Inc.*, 15 Sept. 2012, aws.amazon.com/sdk-for-java/. Accessed 03 July 2022.
- [2] “Awsdocs/Aws-Doc-Sdk-Examples.” *GitHub*, 5 Sept. 2019, github.com/awsdocs/aws-doc-sdk-examples/blob/main/java/example_code/s3/src/main/java/aws/example/s3/CreateBucket.java. Accessed 04 July 2022.
- [3] baeldung. “AWS S3 with Java | Baeldung.” *Www.baeldung.com*, 24 July 2017, www.baeldung.com/aws-s3-java. Accessed 04 July 2022.
- [4] “Managing Dependencies with AWS SDK for Java – Bill of Materials Module (BOM).” *Amazon Web Services*, 10 Aug. 2015, aws.amazon.com/blogs/developer/managing-dependencies-with-aws-sdk-for-java-bill-of-materials-module-bom. Accessed 04 July 2022.
- [5] anubhava. “Python - Regular Expression to Extract Named Entities from Text Just Based on Capitalization.” *Stack Overflow*, 9 May 2017, stackoverflow.com/questions/43972800/regular-

- expression-to-extract-named-entities-from-text-just-based-on-capitalizat. Accessed 05 July 2022.
- [6] AWS. “How to Read Files from S3 Using Python AWS Lambda.” *Gcptutorials*, 9 Mar. 2013, www.gcptutorials.com/post/how-to-read-files-from-s3-using-python-aws-lambda. Accessed 05 July 2022.
- [7] “Tutorial: Using an Amazon S3 Trigger to Invoke a Lambda Function - AWS Lambda.” *Docs.aws.amazon.com*, 17 June 2013, docs.aws.amazon.com/lambda/latest/dg/with-s3-example.html. Accessed 05 July 2022.
- [8] “c# - `^[A-Za-Z][A-Za-z0-9]*` regular expression?,” *Stack Overflow*, Oct. 31, 2009. <https://stackoverflow.com/questions/1653425/a-za-z-a-za-z0-9-regular-expression> (accessed Jul. 05, 2022).
- [9] “Tweets Preprocessing,” *kaggle.com*, Apr. 18, 2020. <https://www.kaggle.com/code/quentinsarrazin/tweets-preprocessing/notebook> (accessed Jul. 05, 2022).
- [10] C. Munns, “Using AWS Lambda and Amazon Comprehend for sentiment analysis,” *Amazon Web Services*, Apr. 11, 2018. <https://aws.amazon.com/blogs/compute/using-aws-lambda-and-amazon-comprehend-for-sentiment-analysis/> (accessed Jul. 05, 2022).