

Assignment 4

Part C

Meet Patel (B00899516)

Dalhousie University

Subject

**CSCI 5410 (Serverless Data
Processing)**

Professor

Dr. Saurabh Dey

Project **Git Repository**

Gitlab Repository Link: https://git.cs.dal.ca/patel13/csci5410_b00899516_meet_patel.git

Build an event-driven serverless application using AWS Comprehend. In this part of the assignment, you need to use S3 bucket, Lambda Functions, and AWS Comprehend. [B00xxxxxx = your B00 number] used in bucket naming.

Screenshots of all the steps performed

- A. Create your 1st S3 bucket TwitterDataB00xxxxxx and upload the given tweets file. You need to write a script or use the SDK to upload the file on the bucket.**

Figure 1 shows the AWS management console

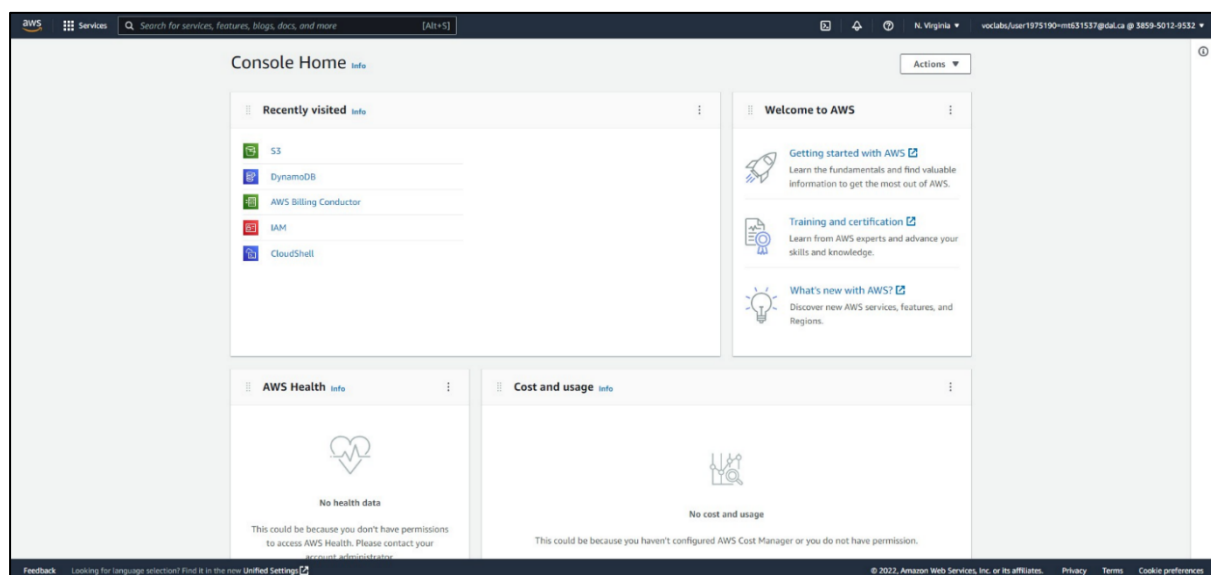


Figure 1: AWS Management Console

Figure 2 shows the Amazon S3 empty dashboard.

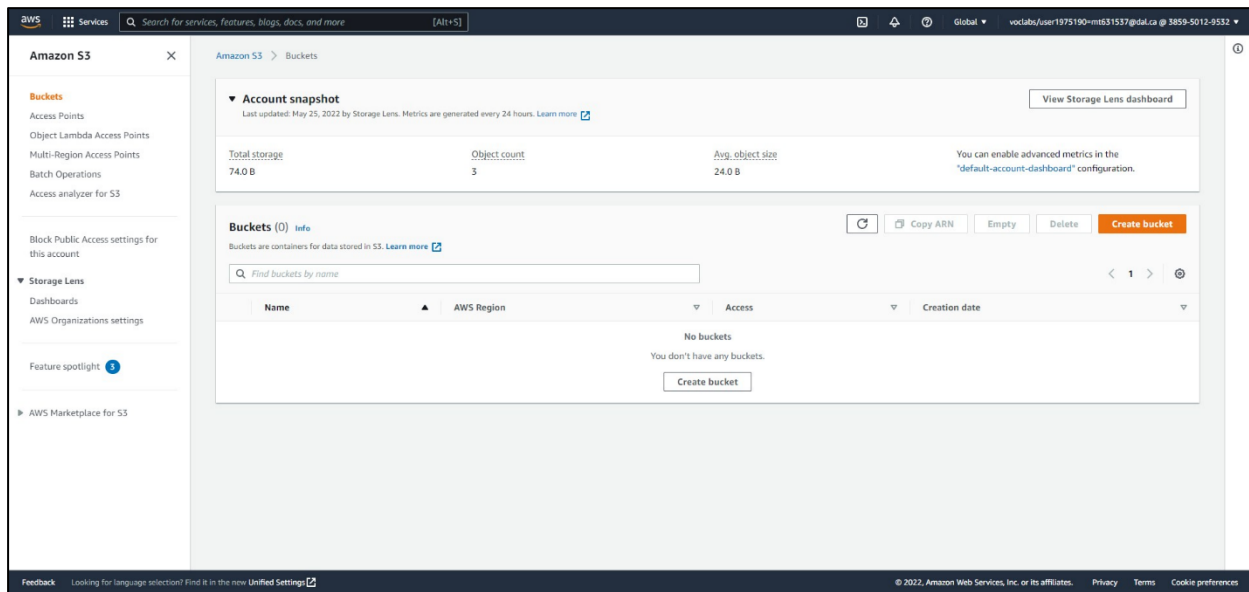


Figure 2: ScreenShot before creating Amazon S3 bucket

Here we are creating two Amazon S3 bucket

- **twitterdatab00899516** – This bucket is responsible for storing the **file_mongo_tweets.txt** (File provided by the professor) which contains raw tweets in txt file.
- **sentimentanalysisb00899516** - This bucket is responsible for storing **analyzedtweets.json** file which contains the sentiment analyzed tweets. Sentiment analysis is done on each tweet. For each tweet positive score, negative score, neutral score, and overall sentiment result is analyzed and stored in json format.

Figure 3 shows the successful creation of **TwitterDataB00899516** and **ProcessedTwitterDataB00899516** using java and AWS java SDK

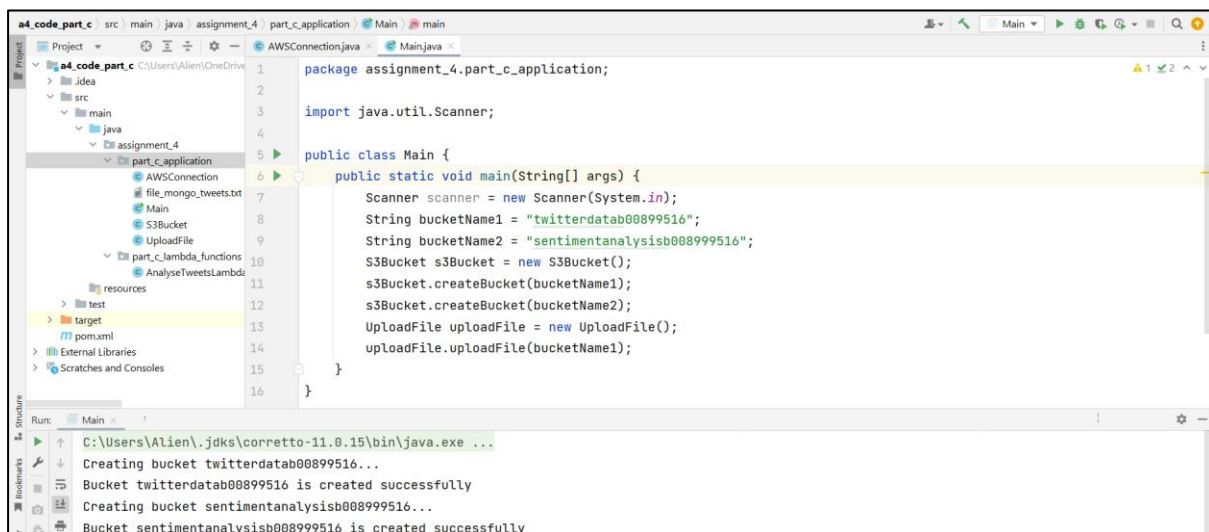


Figure 3: Creation of two Amazon S3 Bucket namely "twitterdatab00899516" and "sentimentanalysisb00899516" using java and AWS java

Figure 4 is responsible for displaying the two Amazon S3 bucket namely “**twitterdatab00899516**” and “**sentimentanalysisb00899516**” which are created successfully created on Amazon S3 console.

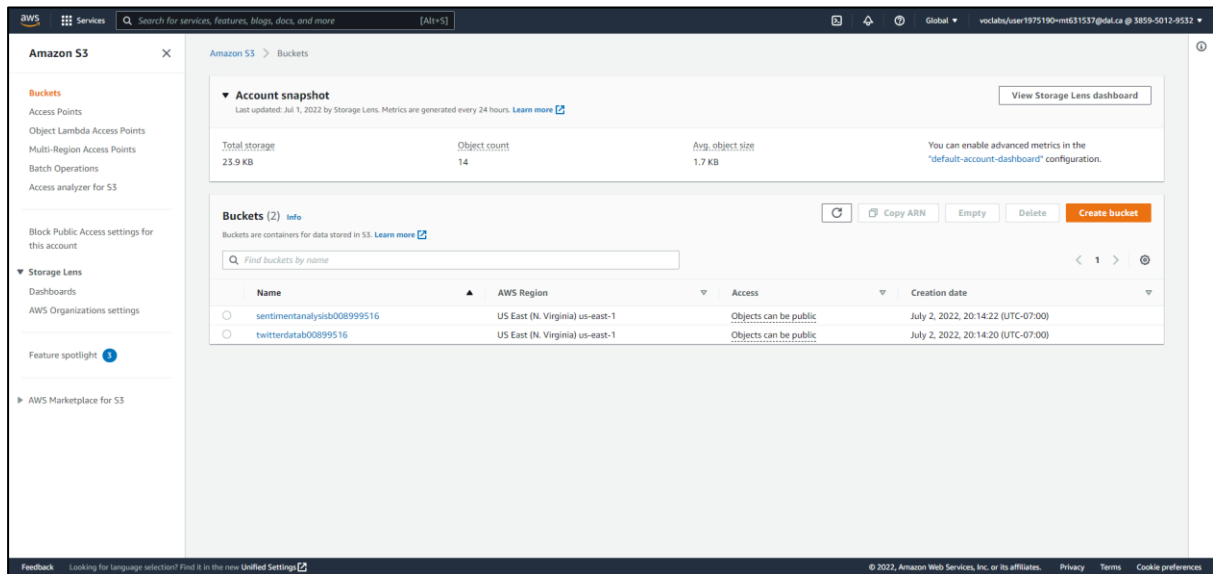


Figure 4: Successful creation of two bucket namely " **twitterdatab00899516**" and " **sentimentanalysisb00899516**"
Amazon S3 console

Figure 5 shows the content of the **file_mongo_tweets.txt**

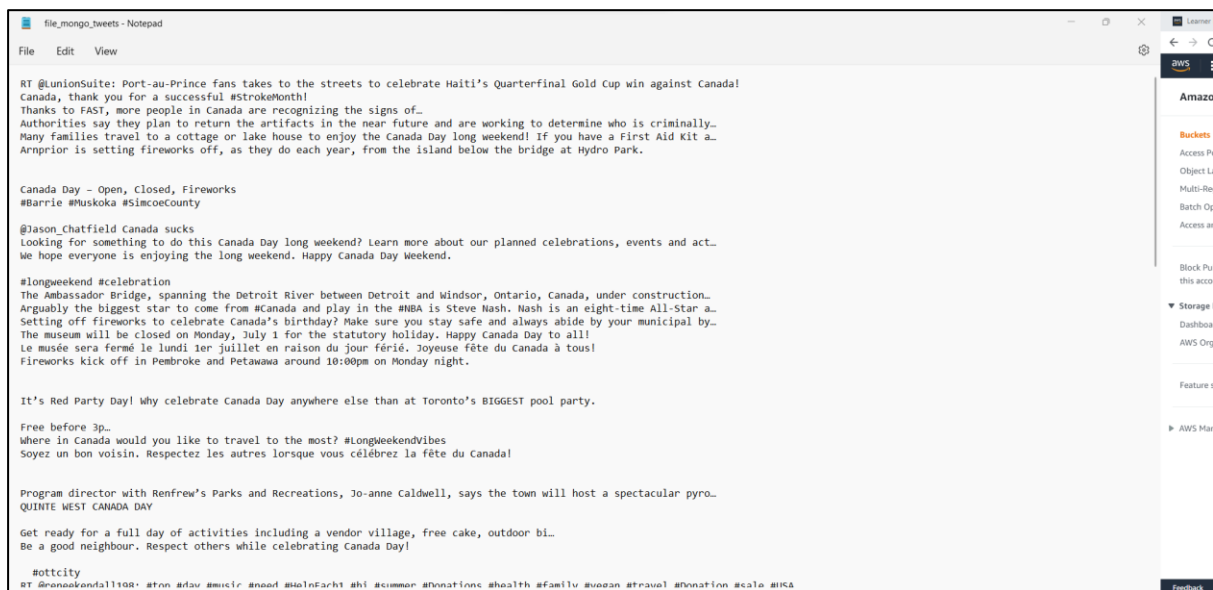


Figure 5: Content of the **file_mongo_tweets.txt**

Figure 6 shows the empty “twitterdatab00899516” Amazon S3 bucket

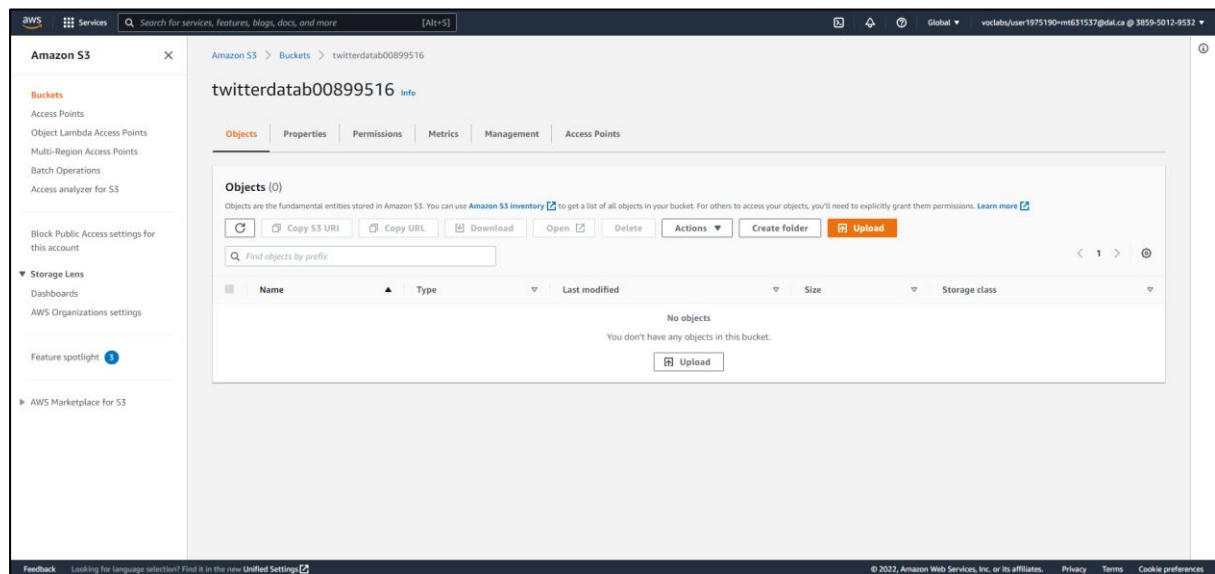


Figure 6: Empty “twitterdatab00899516” Amazon S3 bucket

Figure 7 shows the successful upload of file_mongo_tweets.txt file to twitterdataB00899516 amazon S3 bucket using java and AWS java SDK

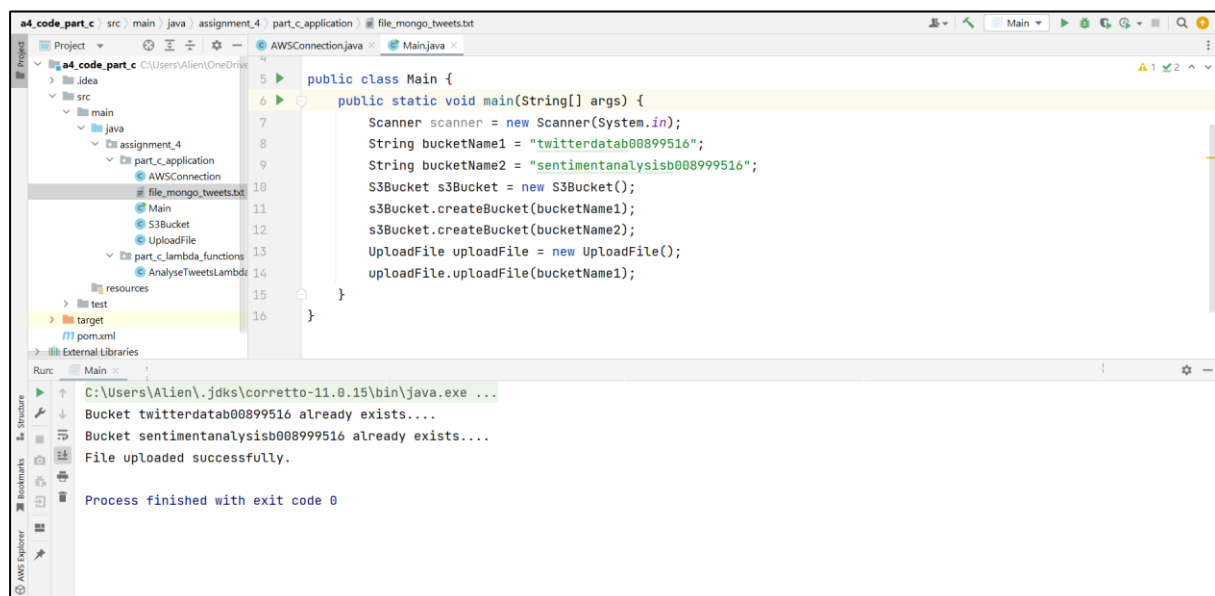


Figure 7: Successful upload of file_mongo_tweets.txt file to twitterdataB00899516 amazon S3 bucket using java and AWS java SDK

Figure 8 shows the “twitterdataB00899516” Amazon S3 bucket console where `file_mongo_tweets.txt` file have been uploaded successfully.

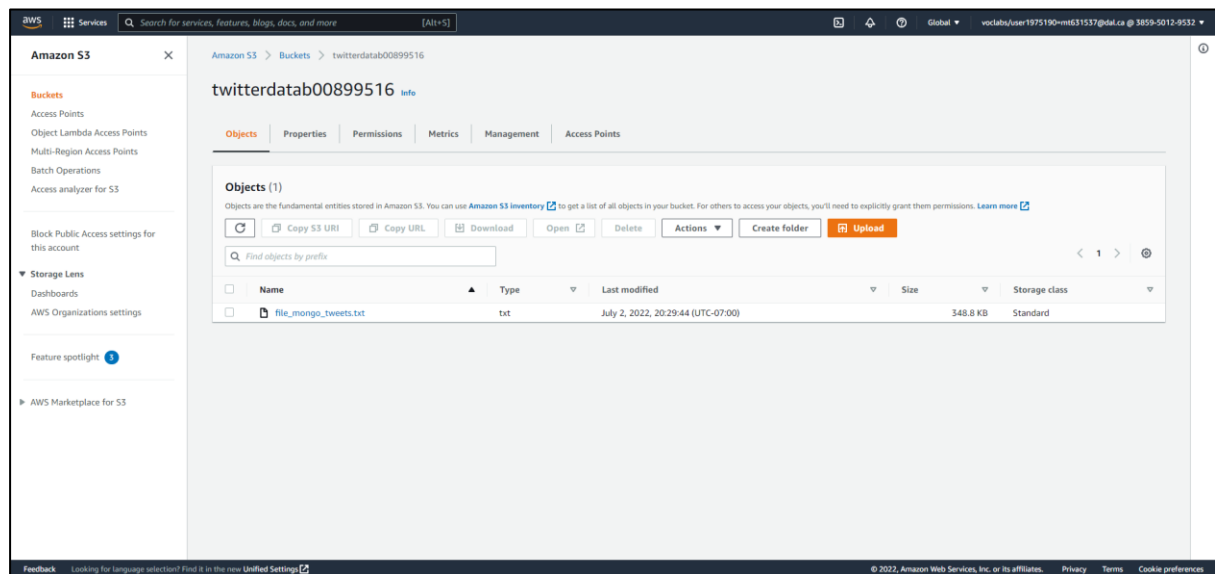


Figure 8: “twitterdataB00899516” Amazon S3 bucket console where `file_mongo_tweets.txt` file have been uploaded successfully

B. To perform any pre or post processing of the files, you can write Lambda functions

Creation of Lambda Function

Figure 9 is responsible for the showing the creation “**analysetweets**” lambda function and its configuration

The screenshot shows the AWS Lambda 'Create function' page. The 'Author from scratch' option is selected. The function name is 'analysetweets'. The runtime is set to 'Java 11 (Corretto)'. The architecture is 'x86_64'. The permissions section shows the 'Change default execution role' dropdown set to 'Use an existing role' with 'LabRole' selected. The 'Create function' button is visible at the bottom right.

Figure 9: Creation of " **analysetweets** " lambda Function

Figure 10 shows the successfully created lambda function on Amazon Lambda named as “**analysetweets**”

The screenshot shows the AWS Lambda console for the 'analysetweets' function. The 'Function overview' tab is active, displaying the function name, layers, and description. The 'Code source' tab shows a message that the deployment package is too large for inline editing. The 'Code properties' section shows the package size (19.9 MB), SHA256 hash, and last modified date (July 2, 2022). The 'Runtime settings' section shows the runtime (Java 11 (Corretto)), handler, and architecture (x86_64). The 'Layers' section is empty, indicating no layers are currently attached.

Figure 10: Successfully created lambda Function " **analysetweets** "

Figure 10 shows the creation of trigger for “analysetweets” lambda function, and it will be triggered when object is uploaded to the “twitterdataB00899516” Amazon S3 bucket.

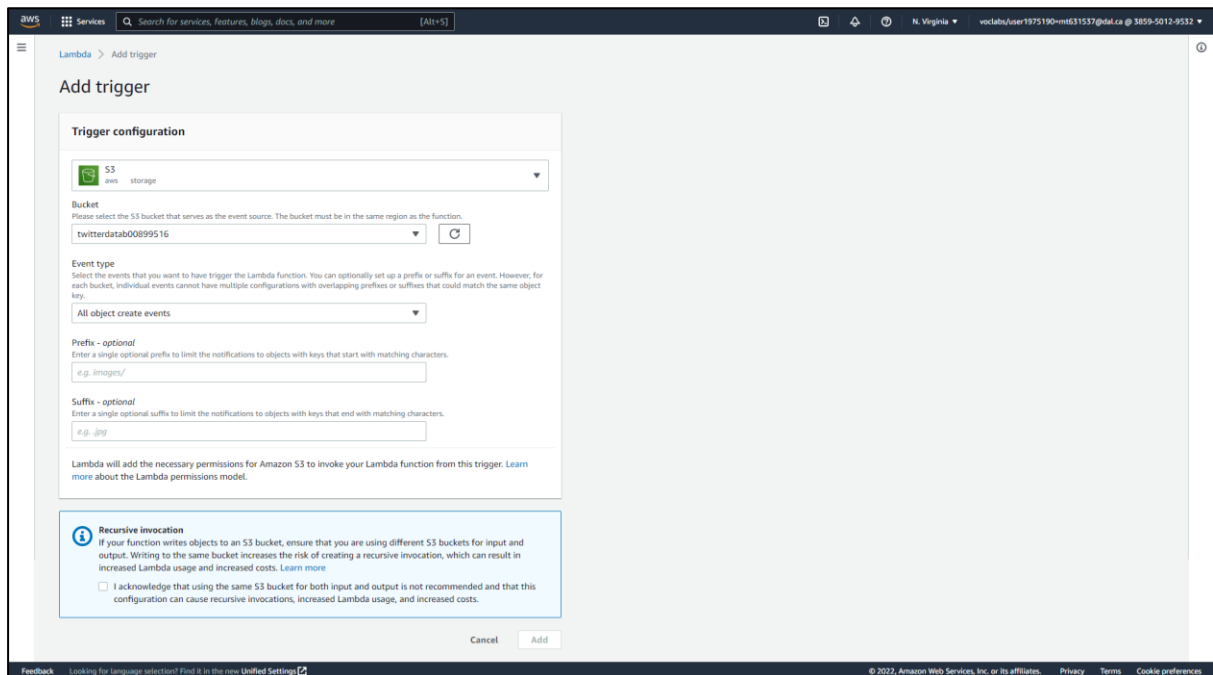


Figure 11: Creation of trigger for “analysetweets” lambda function

Figure 12 shows the successful creation of trigger for “analysetweets” lambda function

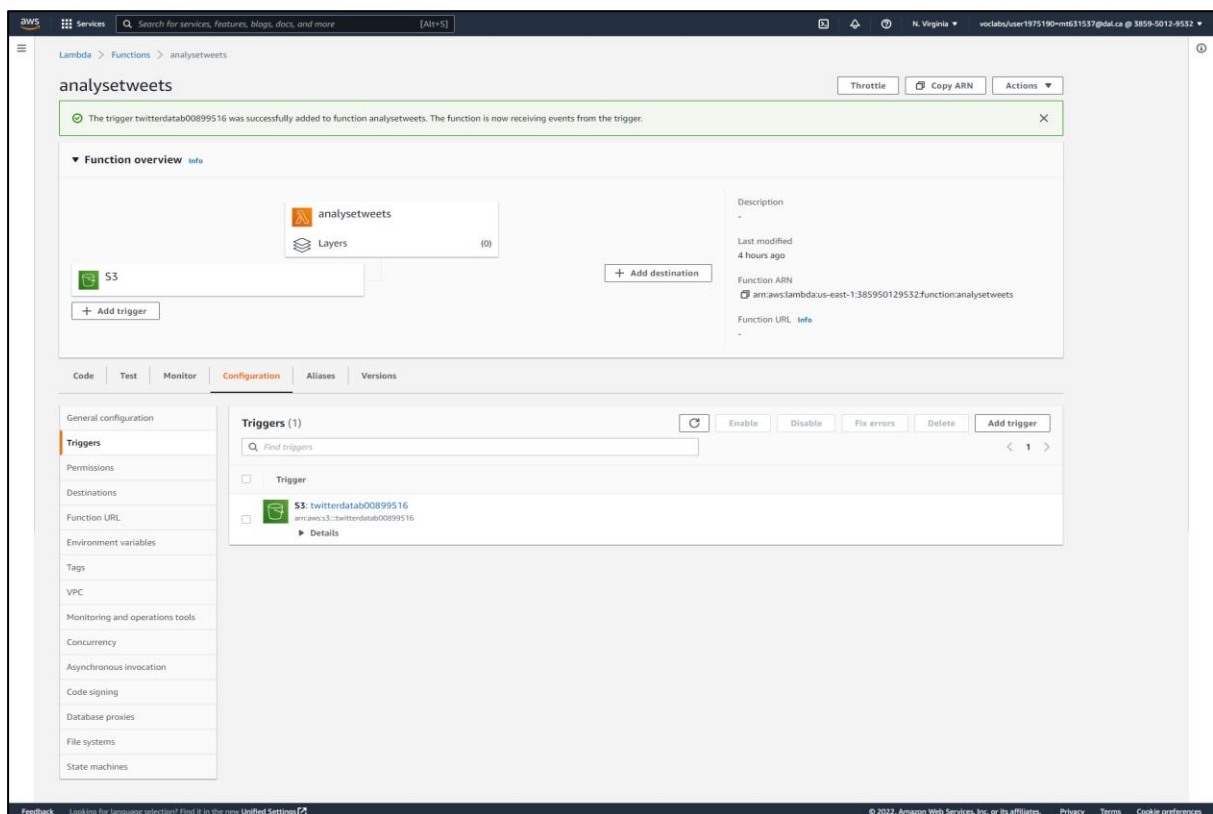


Figure 12: Successful creation of trigger for “analysetweets” lambda function

C. Once the file containing all the tweets is uploaded on the bucket, AWS Comprehend is used to perform sentiment analysis of tweets

Figure 13, 14 and 15 shows the cloudwatch logs. This shows that sentiment analysis has been done on the tweets which are stored in `file_mongo_tweets.txt` file.

Before the sentiment analysis the tweets have been cleaned. The “#”, “.”, “\n” have been removed.

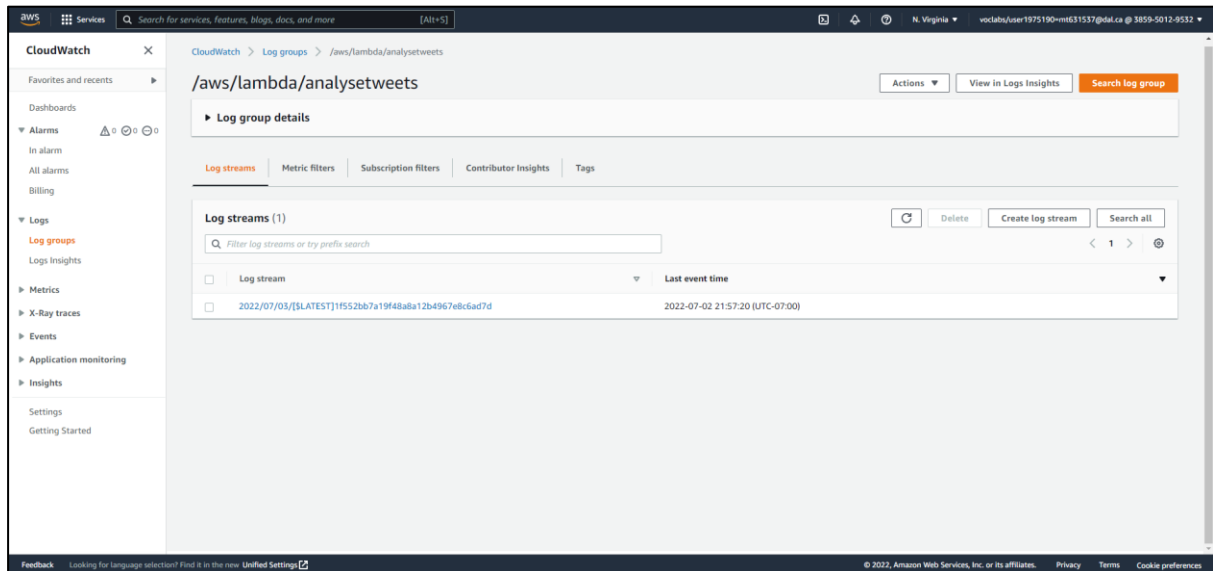


Figure 13: Cloudwatch log

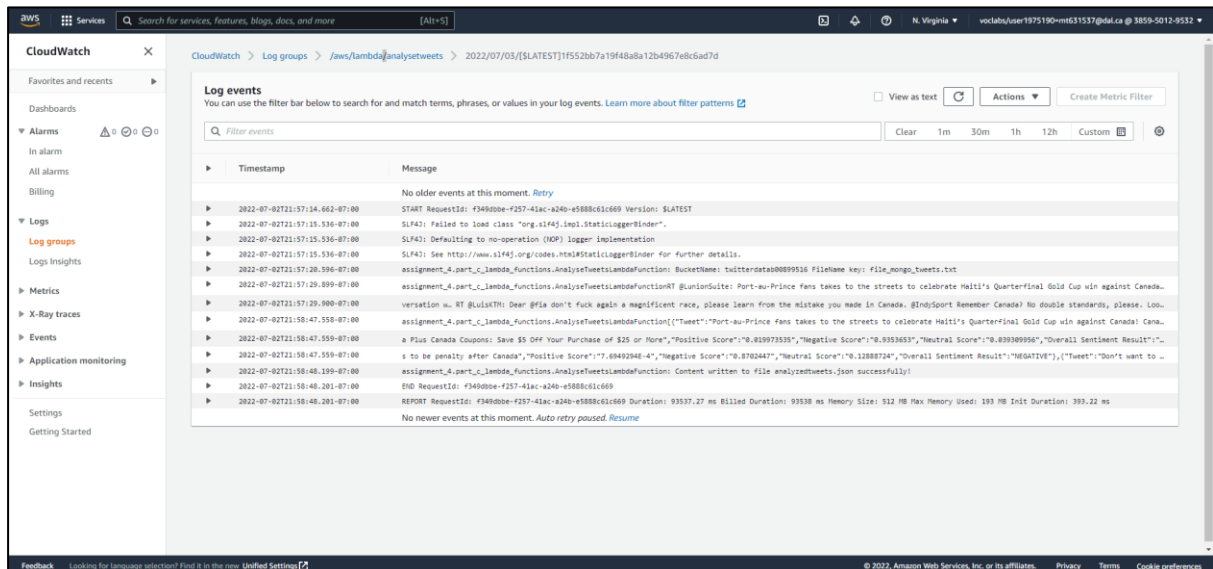


Figure 14: Cloudwatch log

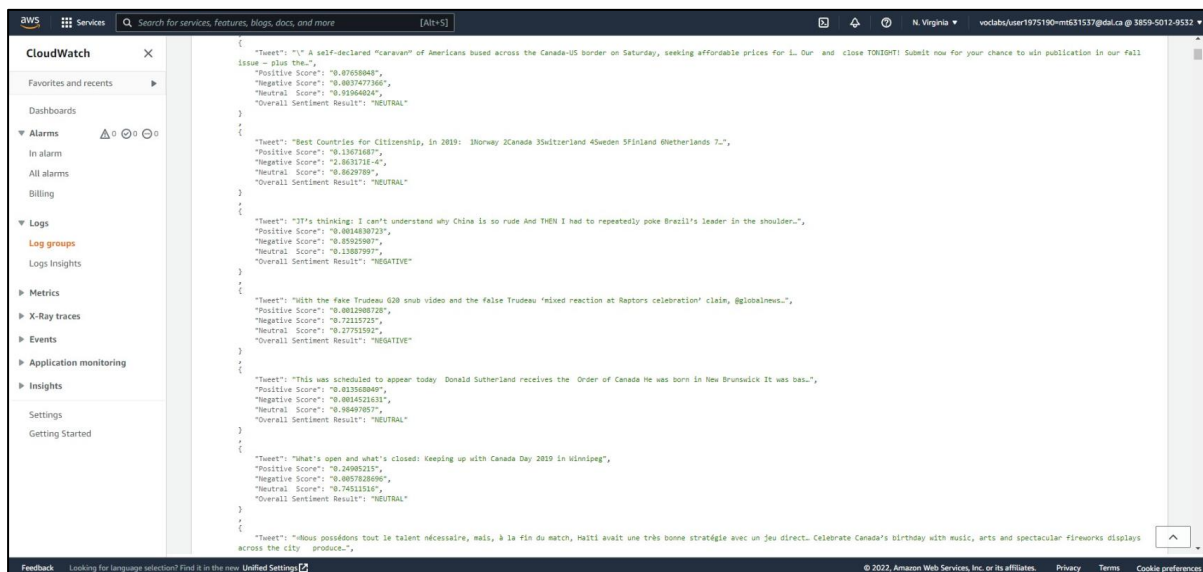


Figure 15: CloudWatch log showing sentiment analysis of tweets (Includes Positive Score, Negative Score, Neutral Score and Overall Sentiment Result) – 1

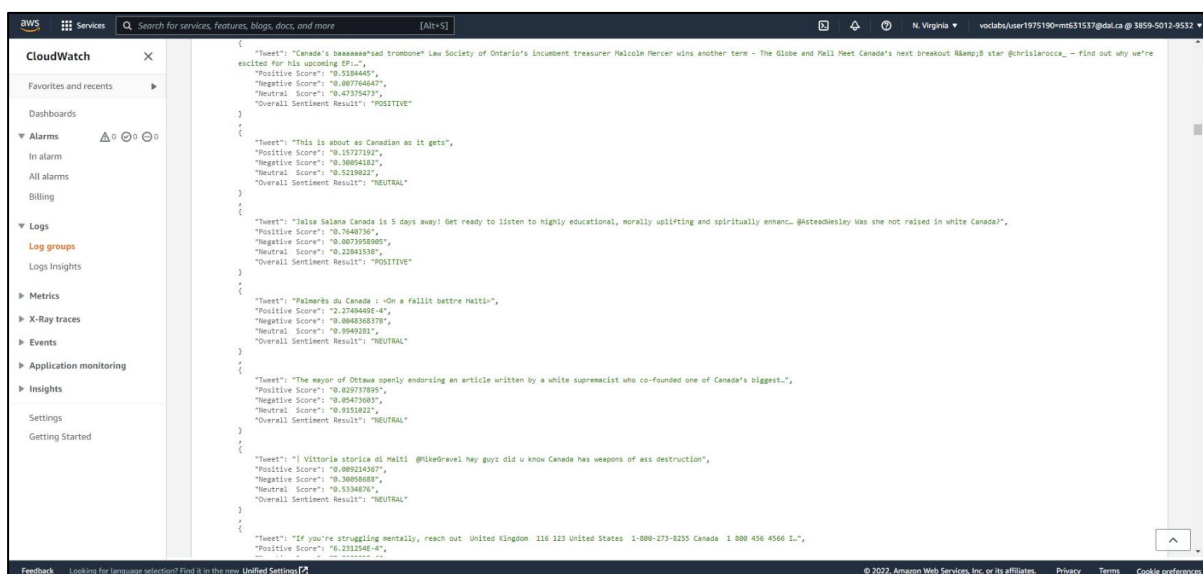


Figure 16: CloudWatch log showing sentiment analysis of tweets (Includes Positive Score, Negative Score, Neutral Score and Overall Sentiment Result) – 2

D. Your output should be captured in a csv or json format.

Figure 17 shows the empty **sentimentanalysisb008999516** Amazon S3 bucket

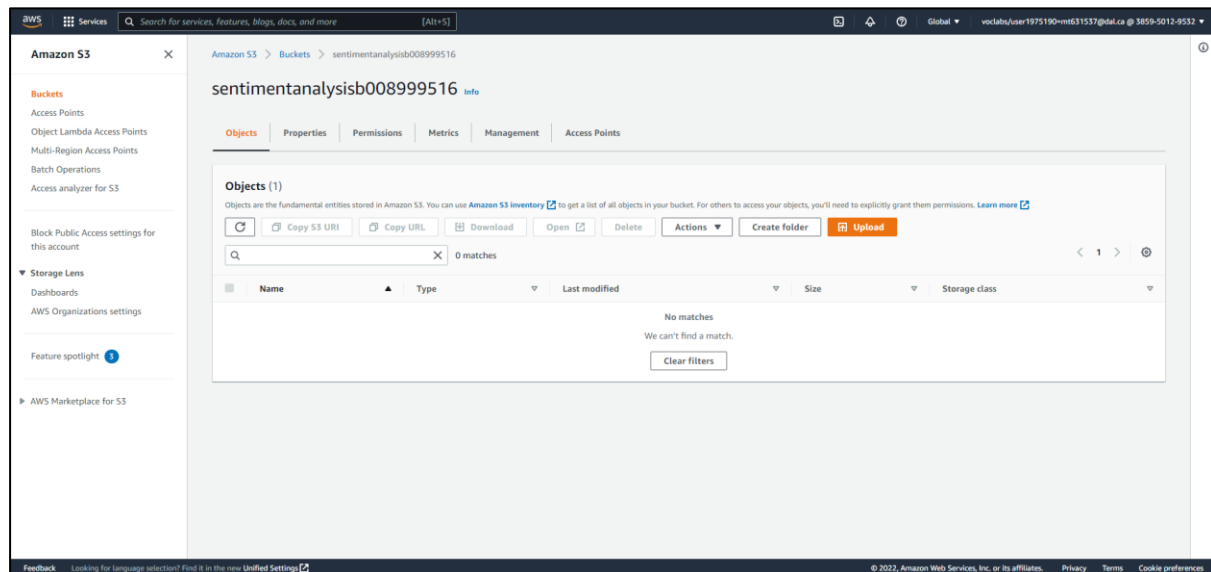


Figure 17: Empty **sentimentanalysisb008999516** Amazon S3 bucket

Figure 18 shows the successful creation of **analyzedtweets.json** file on **sentimentanalysisb008999516** Amazon S3 bucket

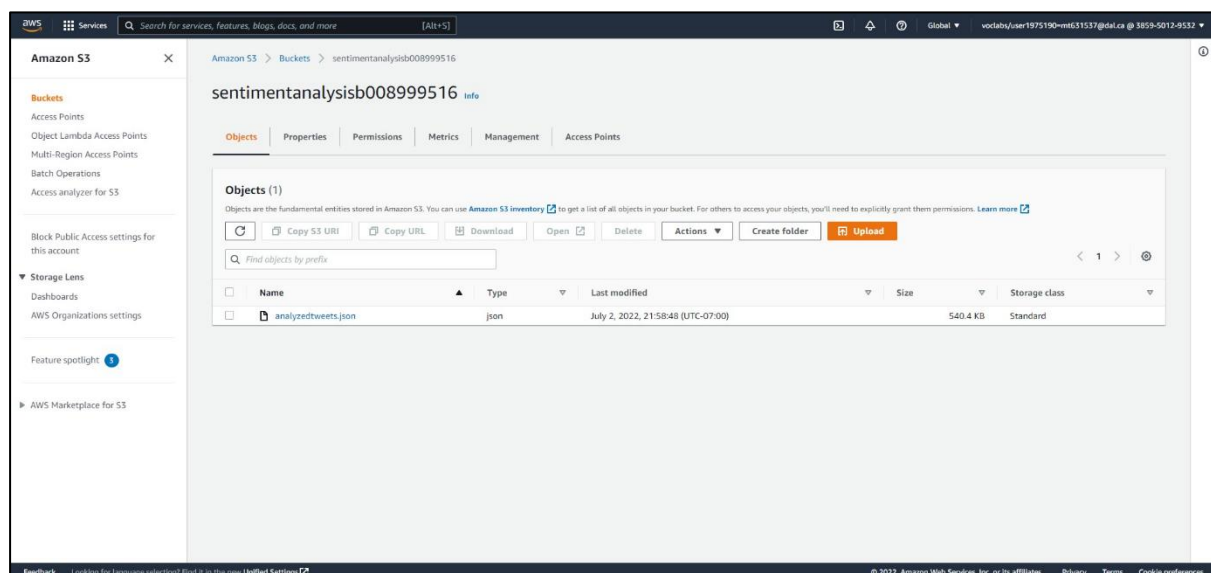
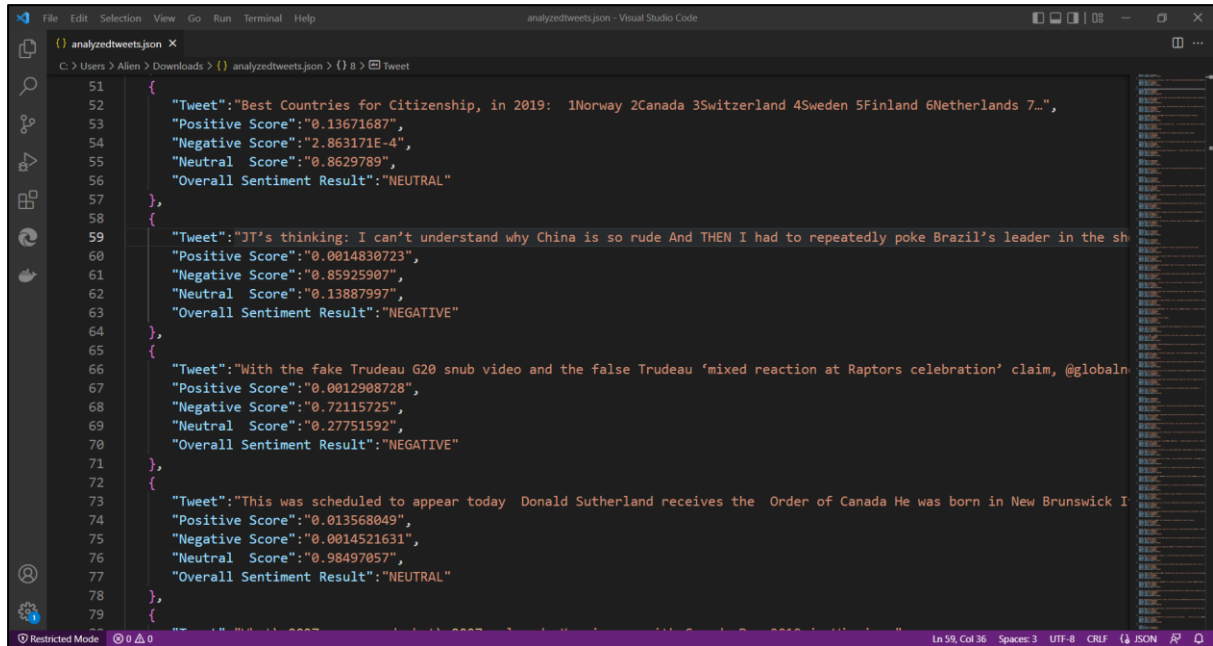


Figure 18: Creation of **analyzedtweets.json** file on **sentimentanalysisb008999516** Amazon S3 bucket

Figure 19 and 20 shows the downloaded **analyzedtweets.json** file and its content.

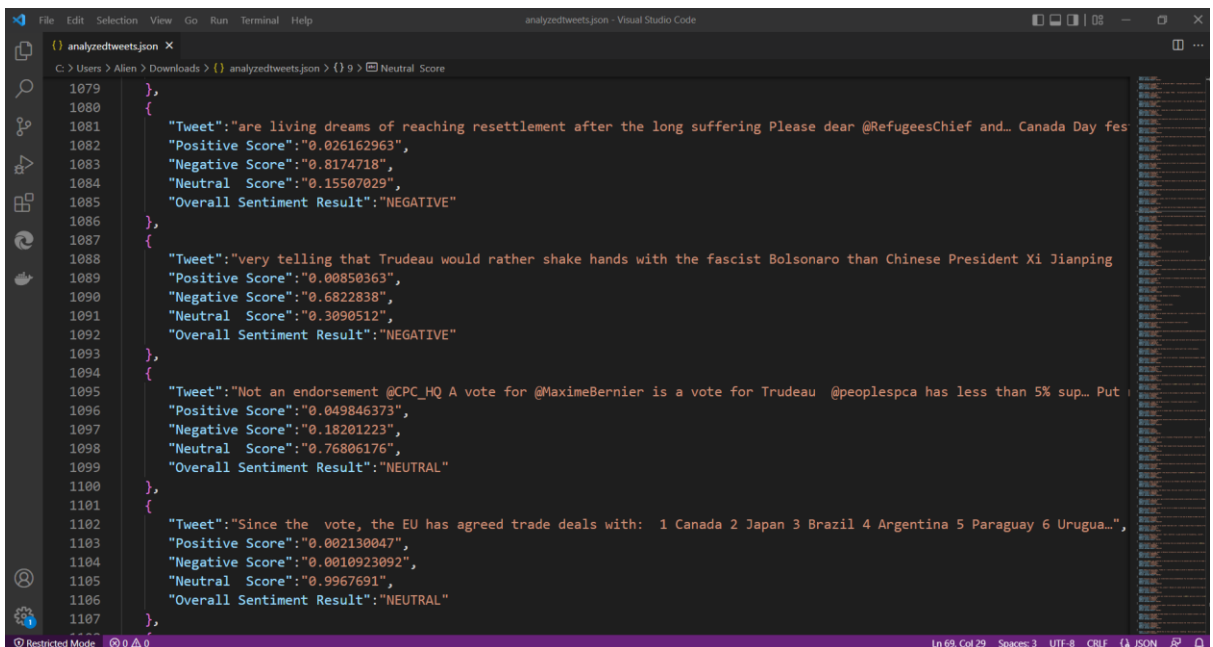


```

51  {
52    "Tweet": "Best Countries for Citizenship, in 2019: 1Norway 2Canada 3Switzerland 4Sweden 5Finland 6Netherlands 7...",
53    "Positive Score": "0.13671687",
54    "Negative Score": "2.863171E-4",
55    "Neutral Score": "0.8629789",
56    "Overall Sentiment Result": "NEUTRAL"
57  },
58  {
59    "Tweet": "JT's thinking: I can't understand why China is so rude And THEN I had to repeatedly poke Brazil's leader in the sh
60    "Positive Score": "0.0014830723",
61    "Negative Score": "0.85925907",
62    "Neutral Score": "0.13887997",
63    "Overall Sentiment Result": "NEGATIVE"
64  },
65  {
66    "Tweet": "With the fake Trudeau G20 snub video and the false Trudeau 'mixed reaction at Raptors celebration' claim, @globalIn
67    "Positive Score": "0.0012908728",
68    "Negative Score": "0.72115725",
69    "Neutral Score": "0.27751592",
70    "Overall Sentiment Result": "NEGATIVE"
71  },
72  {
73    "Tweet": "This was scheduled to appear today Donald Sutherland receives the Order of Canada He was born in New Brunswick I
74    "Positive Score": "0.013568049",
75    "Negative Score": "0.0014521631",
76    "Neutral Score": "0.98497057",
77    "Overall Sentiment Result": "NEUTRAL"
78  },
79  },

```

Figure 19: Downloaded **analyzedtweets.json** file



```

1079  },
1080  {
1081    "Tweet": "are living dreams of reaching resettlement after the long suffering Please dear @RefugeesChief and... Canada Day fes
1082    "Positive Score": "0.026162963",
1083    "Negative Score": "0.8174718",
1084    "Neutral Score": "0.15507029",
1085    "Overall Sentiment Result": "NEGATIVE"
1086  },
1087  {
1088    "Tweet": "very telling that Trudeau would rather shake hands with the fascist Bolsonaro than Chinese President Xi Jinping
1089    "Positive Score": "0.00850363",
1090    "Negative Score": "0.6822838",
1091    "Neutral Score": "0.3090512",
1092    "Overall Sentiment Result": "NEGATIVE"
1093  },
1094  {
1095    "Tweet": "Not an endorsement @CPC_HQ A vote for @MaximeBernier is a vote for Trudeau @peoplespc has less than 5% sup... Put
1096    "Positive Score": "0.049846373",
1097    "Negative Score": "0.18201223",
1098    "Neutral Score": "0.76806176",
1099    "Overall Sentiment Result": "NEUTRAL"
1100  },
1101  {
1102    "Tweet": "Since the vote, the EU has agreed trade deals with: 1 Canada 2 Japan 3 Brazil 4 Argentina 5 Paraguay 6 Uruguay...",
1103    "Positive Score": "0.002130047",
1104    "Negative Score": "0.0010923092",
1105    "Neutral Score": "0.9967691",
1106    "Overall Sentiment Result": "NEUTRAL"
1107  },

```

Figure 20: Downloaded **analyzedtweets.json** file

Program Script

Application Files

AWSConnection.java

This java class is used to establish connection with the **Amazon S3 service**

```
package assignment_4.part_c_application;
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 AWSConnection {
    private static final String AWS_ACCESS_KEY_ID = "ASIAVTXDLTV6ALDVPWYP";
    private static final String AWS_SECRET_ACCESS_KEY = "VRutoq2x0c/AN48o0w74jtgilxPB1OypDTQiqrXI";
    private static final String AWS_SESSION_TOKEN = "FwoGZXIvYXZzEFQaDj8a2dsDCc7TA81gGiLAAUUm" +
        "fKbyZIoWmXzVmpyQHRbsCrAD/HQLqbx7rnDbLDJEheH09voYBHRX42XSbd36BTYHKQMiowsQe1Ode+PdHd" +
        "0cvgOP6RtArLis4WrMFltFXLcUmVbFZgm6YggKIjRCNCYpnd6rnPC4R8LuK6vrj3AWwE8pvH8nsJRB2Eh3" +
        "nhdyephMDPlrLIDrfjh7h37oYwyn9alyBAFYglHKctUyEyBV9ro27o2Q1010vZWUQ3fGh43MWftP5UEgn" +
        "XWCwx35gyjcyjISWBjIttvFQfqa9D4x2hSgZwaZkSko3n7rVuJQPtmDXEfxewthmfkx1o543j1CThAUK";
    public AmazonS3 createAmazonS3ClientBuilder() {
        BasicSessionCredentials basicSessionCredentials = new BasicSessionCredentials(AWS_ACCESS_KEY_ID,
AWS_SECRET_ACCESS_KEY, AWS_SESSION_TOKEN);
        AmazonS3 amazonS3Object = AmazonS3ClientBuilder.standard()
            .withCredentials(new AWSStaticCredentialsProvider(basicSessionCredentials))
            .withRegion(Regions.US_EAST_1)
            .build();
        return amazonS3Object;
    }
}
```

S3Bucket.java

This java class is used to create **Amazon S3 bucket** using **createBucket ()** built method. Before

```
package assignment_4.part_c_application;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.model.AmazonS3Exception;
import com.amazonaws.services.s3.model.Bucket;
import java.util.List;
public class S3Bucket {
    public boolean isBucketExists(String bucketName, AmazonS3 amazonS3ClientBuilder){
        List<Bucket> bucketList = amazonS3ClientBuilder.listBuckets();
        for (Bucket bucket: bucketList){
            if(bucket.getName().equals(bucketName)){
                return true;
            }
        }
        return false;
    }
    public void createBucket(String bucketName){
        AWSConnection awsConnection = new AWSConnection();
        if (isBucketExists(bucketName, awsConnection.createAmazonS3ClientBuilder())){
            System.out.println("Bucket "+bucketName+ " already exists...");
        }else {
            try {
                System.out.println("Creating bucket " + bucketName + "...");
                awsConnection.createAmazonS3ClientBuilder().createBucket(bucketName);
                System.out.println("Bucket "+bucketName+ " is created successfully");
            }catch (AmazonS3Exception e){
                System.err.println(e.getMessage());
            }
        }
    }
}
```

creating bucket this also checks whether **bucket already exists or not**.

UploadFile.java

This java class is used to upload the text file **file_mongo_tweets.txt** to the **Amazon S3 bucket** using **putObject ()** method

```
package assignment_4.part_c_application;

import com.amazonaws.AmazonServiceException;

import java.io.File;
import java.nio.file.Paths;

public class UploadFile {
    AWSConnection awsConnection = new AWSConnection();
    public void uploadFile(String bucketName) {
        final String path = "./src/main/java/assignment_4/part_c_application/file_mongo_tweets.txt";
        final File file = new File(path);
        final String fileName = Paths.get(path).getFileName().toString();
        try {
            awsConnection.createAmazonS3ClientBuilder().putObject(bucketName, fileName, file);
            System.out.println("File uploaded successfully.");
        } catch (final AmazonServiceException e) {
            e.printStackTrace();
        }
    }
}
```

Main.java

This is the main method which pass the bucketnames to the **create bucket** method. Then it will call uploadFile method to upload the file_mongo_tweets.txt file

```
package assignment_4.part_c_application;

import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String bucketName1 = "twitterdatab00899516";
        String bucketName2 = "sentimentanalysisb00899516";
        S3Bucket s3Bucket = new S3Bucket();
        s3Bucket.createBucket(bucketName1);
        s3Bucket.createBucket(bucketName2);
        UploadFile uploadFile = new UploadFile();
        uploadFile.uploadFile(bucketName1);
    }
}
```

Lambda Function

AnalyseTweetsLambdaFunction.js

```
package assignment_4.part_c_lambda_functions;

import com.amazonaws.regions.Regions;
import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.lambda.runtime.RequestHandler;
import com.amazonaws.services.lambda.runtime.events.S3Event;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.google.gson.Gson;
import software.amazon.awssdk.auth.credentials.EnvironmentVariableCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.comprehend.ComprehendClient;
import software.amazon.awssdk.services.comprehend.model.DetectSentimentRequest;
import software.amazon.awssdk.services.comprehend.model.DetectSentimentResponse;
import java.io.UnsupportedEncodingException;
import java.net.URLDecoder;
import java.util.LinkedHashMap;
import java.util.Map;

public class AnalyseTweetsLambdaFunction implements RequestHandler<S3Event, String> {
    final static String LOG = "assignment_4.part_c_lambda_functions.AnalyseTweetsLambdaFunction";
    @Override
    public String handleRequest(S3Event s3Event, Context context) {
        try {
            Map<String, String> entities = new LinkedHashMap<>();
            StringBuilder tweetsInJson = new StringBuilder();
            Region region = Region.US_EAST_1;
            ComprehendClient comClient = ComprehendClient.builder()
                .region(region)
                .credentialsProvider(EnvironmentVariableCredentialsProvider.create())
                .build();

            String bucketName = s3Event.getRecords().get(0).getS3().getBucket().getName();
            String fileNameKey =
                URLDecoder.decode(s3Event.getRecords().get(0).getS3().getObject().getKey().replace('/', ' '), "UTF-8");

            context.getLogger().log(LOG + ": BucketName: " + bucketName + " FileName key: " + fileNameKey);

            String fileContent =
                AmazonS3ClientBuilder.standard().withRegion(Regions.US_EAST_1).build().getObjectAsString(bucketName, fileNameKey);

            context.getLogger().log(LOG + fileContent);

            String[] tweets = fileContent.replaceAll("\n", " ").split("RT @\\w+:");

            Gson gsonObj = new Gson();
            tweetsInJson.append("[");
            for (int i=1; i<tweets.length; i++){
                DetectSentimentRequest detectSentimentRequest = DetectSentimentRequest.builder()
                    .text(tweets[i])
                    .languageCode("en")
                    .build();
                DetectSentimentResponse detectSentimentResult = comClient.detectSentiment(detectSentimentRequest);
                entities.put("Tweet", tweets[i].replaceAll("#\\w+", "").replaceAll("[\\u00A0]+",
                    "").replaceAll("\\.", "").trim());
                entities.put("Positive Score", detectSentimentResult.sentimentScore().positive().toString());
                entities.put("Negative Score", detectSentimentResult.sentimentScore().negative().toString());
                entities.put("Neutral Score", detectSentimentResult.sentimentScore().neutral().toString());
                entities.put("Overall Sentiment Result", detectSentimentResult.sentimentAsString());
                tweetsInJson.append(gsonObj.toJson(entities)).append(",");
            }
            tweetsInJson.replace(tweetsInJson.length() - 1, tweetsInJson.length(), "");
            tweetsInJson.append("]");
            context.getLogger().log(LOG + tweetsInJson);

            comClient.close();

            String newFileName = "analyzedtweets.json";

            AmazonS3ClientBuilder.standard().withRegion(Regions.US_EAST_1).build().putObject("sentimentanalysisb00899516",
                newFileName, tweetsInJson.toString());
            context.getLogger().log(LOG + ": " + "Content written to file " + newFileName + " successfully!");
            return "correct";
        } catch (UnsupportedEncodingException e) {
            context.getLogger().log(LOG + ": " + e.getMessage());
            return "error";
        }
    }
}
```

References

- [1] AWS, "AWS Lambda," Amazon, [Online]. Available: <https://aws.amazon.com/lambda/>. [Accessed: July 3, 2022].
- [2] AWS, "Amazon CloudWatch," Amazon, [Online]. Available: <https://aws.amazon.com/cloudwatch/>. [Accessed: July 3, 2022].
- [3] Amazon, "Amazon Comprehend," AWS, [Online]. Available: <https://aws.amazon.com/comprehend/>. [Accessed: July 3, 2022].
- [4] A. W. Services, "AWS SDK for Java Documentation," Amazon, 2022. [Online]. Available: <https://docs.aws.amazon.com/sdk-for-java/index.html> [Accessed: July 3, 2022].
- [5] AWS, "Amazon S3," Amazon, [Online]. Available: <https://aws.amazon.com/s3/>. [Accessed: July 3, 2022].