**Instance A - Image Label Rekognition**

**App.java**

package com.njit;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.List;

public class App {

public static void main(String[] args) {

S3Service s3Service = new S3Service();

AnalyzePhotos photos = new AnalyzePhotos();

ReadSQSMessages messages = new ReadSQSMessages();

String bucketName = "njit-cs-643";

List<String> myKeys = s3Service.listBucketObjects(bucketName);

// System.out.println(Arrays.toString(myKeys.toArray()));

List<String> myList = new ArrayList<>();

// loop through each element in the List and tag the assets.

for (String key : myKeys) {

byte[] keyData = s3Service.getObjectBytes(bucketName, key);

// Analyze the photo and return a list where each element is a WorkItem.

String item = photos.detectLabels(keyData, key);

if (item != null) {

myList.add(item);

}

}

myList.add("-1");

System.out.println(messages.sendMessages(myList));

System.out.println("Keys :" + Arrays.toString(myList.toArray()));

}

}

**ReadSQSMessages.java**

package com.njit;

import java.util.List;

import software.amazon.awssdk.auth.credentials.InstanceProfileCredentialsProvider;

import software.amazon.awssdk.regions.Region;

import software.amazon.awssdk.services.sqs.SqsClient;

import software.amazon.awssdk.services.sqs.model.SendMessageRequest;

public class ReadSQSMessages {

InstanceProfileCredentialsProvider awsCreds = InstanceProfileCredentialsProvider.builder().build();

Region region = Region.US\_EAST\_1;

SqsClient sqsClient = SqsClient.builder().credentialsProvider(awsCreds).region(region).build();

String queueUrl = "https://sqs.us-east-1.amazonaws.com/002519620995/photoQueue.fifo";

public String sendMessages(List<String> myList) {

for (String key : myList) {

SendMessageRequest sendRequest = SendMessageRequest.builder().queueUrl(queueUrl)

.messageGroupId("Car-Photos").messageDeduplicationId(String.valueOf(key.hashCode()))

.messageBody(key).build();

sqsClient.sendMessage(sendRequest);

}

return "Messages sent successfully";

}

}

**S3Service.java:**

package com.njit;

import software.amazon.awssdk.auth.credentials.InstanceProfileCredentialsProvider;

import software.amazon.awssdk.core.ResponseBytes;

import software.amazon.awssdk.regions.Region;

import software.amazon.awssdk.services.s3.S3Client;

import software.amazon.awssdk.services.s3.model.GetObjectRequest;

import software.amazon.awssdk.services.s3.model.GetObjectResponse;

import software.amazon.awssdk.services.s3.model.S3Exception;

import software.amazon.awssdk.services.s3.model.ListObjectsResponse;

import software.amazon.awssdk.services.s3.model.S3Object;

import software.amazon.awssdk.services.s3.model.ListObjectsRequest;

import java.util.ArrayList;

import java.util.List;

public class S3Service {

private S3Client getClient() {

InstanceProfileCredentialsProvider awsCreds = InstanceProfileCredentialsProvider.builder().build();

Region region = Region.US\_EAST\_1;

return S3Client.builder().credentialsProvider(awsCreds).region(region).build();

}

public byte[] getObjectBytes(String bucketName, String keyName) {

S3Client s3 = getClient();

try {

GetObjectRequest objectRequest = GetObjectRequest.builder().key(keyName).bucket(bucketName).build();

// Return the byte[] from this object.

ResponseBytes<GetObjectResponse> objectBytes = s3.getObjectAsBytes(objectRequest);

return objectBytes.asByteArray();

} catch (S3Exception e) {

System.err.println(e.awsErrorDetails().errorMessage());

System.exit(1);

}

return null;

}

// Returns the names of all images in the given bucket.

public List<String> listBucketObjects(String bucketName) {

S3Client s3 = getClient();

String keyName;

List<String> keys = new ArrayList<>();

try {

ListObjectsRequest listObjects = ListObjectsRequest.builder().bucket(bucketName).build();

ListObjectsResponse res = s3.listObjects(listObjects);

List<S3Object> objects = res.contents();

for (S3Object myValue : objects) {

keyName = myValue.key();

keys.add(keyName);

}

return keys;

} catch (S3Exception e) {

System.err.println(e.awsErrorDetails().errorMessage());

System.exit(1);

}

return null;

}

}

**AnalyzePhotos.java:**

package com.njit;

import software.amazon.awssdk.auth.credentials.InstanceProfileCredentialsProvider;

import software.amazon.awssdk.core.SdkBytes;

import software.amazon.awssdk.regions.Region;

import software.amazon.awssdk.services.rekognition.RekognitionClient;

import software.amazon.awssdk.services.rekognition.model.Image;

import software.amazon.awssdk.services.rekognition.model.DetectLabelsRequest;

import software.amazon.awssdk.services.rekognition.model.DetectLabelsResponse;

import software.amazon.awssdk.services.rekognition.model.Label;

import software.amazon.awssdk.services.rekognition.model.RekognitionException;

import java.util.List;

public class AnalyzePhotos {

// Returns a list of WorkItem objects that contains labels.

public String detectLabels(byte[] bytes, String key) {

InstanceProfileCredentialsProvider awsCreds = InstanceProfileCredentialsProvider.builder().build();

Region region = Region.US\_EAST\_1;

RekognitionClient rekClient = RekognitionClient.builder().credentialsProvider(awsCreds).region(region).build();

try {

SdkBytes sourceBytes = SdkBytes.fromByteArray(bytes);

// Create an Image object for the source image.

Image souImage = Image.builder().bytes(sourceBytes).build();

DetectLabelsRequest detectLabelsRequest = DetectLabelsRequest.builder().image(souImage).maxLabels(10)

.minConfidence(90F).build();

DetectLabelsResponse labelsResponse = rekClient.detectLabels(detectLabelsRequest);

// Write the results to a WorkItem instance.

List<Label> labels = labelsResponse.labels();

for (Label label : labels) {

String labelName = label.name();

if (labelName.equals("Car")) {

return key;

}

}

} catch (RekognitionException e) {

System.out.println(e.getMessage());

System.exit(1);

}

return null;

}

}

**WorkItem.java:**

package com.njit;

public class WorkItem {

private String key;

private String name;

private String confidence;

public void setKey(String key) {

this.key = key;

}

public String getKey() {

return this.key;

}

public void setName(String name) {

this.name = name;

}

public String getName() {

return this.name;

}

public void setConfidence(String confidence) {

this.confidence = confidence;

}

public String getConfidence() {

return this.confidence;

}

}

**Instance B - Image Text Rekognition**

**App.java:**

package com.njit;

import java.io.BufferedWriter;

import java.io.FileWriter;

import java.io.IOException;

import java.util.ArrayList;

import java.util.List;

import software.amazon.awssdk.services.sqs.model.Message;

public class App {

public static void main(String[] args) throws IOException {

S3Service s3Service = new S3Service();

AnalyzePhotos photos = new AnalyzePhotos();

ReadSQSMessages messages = new ReadSQSMessages();

FileWriter writer = new FileWriter("./output.txt");

BufferedWriter bw = new BufferedWriter(writer);

String bucketName = "njit-cs-643";

Boolean flag = true;

List<Message> myMessages = new ArrayList<>();

String text = "";

while (flag) {

myMessages = messages.receiveMessages();

for (Message m : myMessages) {

String msg = m.body();

if (msg.equals("-1")) {

flag = false;

} else {

if (msg != null) {

byte[] keyData = s3Service.getObjectBytes(bucketName, msg);

text = photos.detectTextLabels(keyData, msg);

if (text != null && text != "") {

System.out.println("text :" + text);

bw.append("==========================================");

bw.newLine();

bw.append("Image Index : " + msg + ", Text : " + text);

bw.newLine();

bw.append("==========================================");

bw.newLine();

}

}

messages.deleteMessages(m);

}

}

}

bw.close();

System.out.println("Task Completed");

}

}

**AnalyzePhotos.java:**

package com.njit;

import software.amazon.awssdk.auth.credentials.InstanceProfileCredentialsProvider;

import software.amazon.awssdk.core.SdkBytes;

import software.amazon.awssdk.regions.Region;

import software.amazon.awssdk.services.rekognition.RekognitionClient;

import software.amazon.awssdk.services.rekognition.model.Image;

import software.amazon.awssdk.services.rekognition.model.DetectTextRequest;

import software.amazon.awssdk.services.rekognition.model.DetectTextResponse;

import software.amazon.awssdk.services.rekognition.model.RekognitionException;

import software.amazon.awssdk.services.rekognition.model.TextDetection;

import java.io.IOException;

import java.util.List;

public class AnalyzePhotos {

public String detectTextLabels(byte[] bytes, String key) throws IOException {

InstanceProfileCredentialsProvider awsCreds = InstanceProfileCredentialsProvider.builder().build();

Region region = Region.US\_EAST\_1;

RekognitionClient rekClient = RekognitionClient.builder().credentialsProvider(awsCreds).region(region).build();

try {

SdkBytes sourceBytes = SdkBytes.fromByteArray(bytes);

// Create an Image object for the source image.

Image souImage = Image.builder().bytes(sourceBytes).build();

DetectTextRequest textRequest = DetectTextRequest.builder().image(souImage).build();

DetectTextResponse textResponse = rekClient.detectText(textRequest);

List<TextDetection> textCollection = textResponse.textDetections();

String message = "";

for (TextDetection text : textCollection) {

if (text.type().toString().equals("LINE")) {

message = text.detectedText();

}

}

return message;

} catch (RekognitionException e) {

System.out.println(e.getMessage());

System.exit(1);

}

return null;

}

}

**ReadSQSMessages.java:**

package com.njit;

import java.util.List;

import software.amazon.awssdk.auth.credentials.InstanceProfileCredentialsProvider;

import software.amazon.awssdk.regions.Region;

import software.amazon.awssdk.services.sqs.SqsClient;

import software.amazon.awssdk.services.sqs.model.DeleteMessageRequest;

import software.amazon.awssdk.services.sqs.model.Message;

import software.amazon.awssdk.services.sqs.model.ReceiveMessageRequest;

import software.amazon.awssdk.services.sqs.model.SqsException;

public class ReadSQSMessages {

InstanceProfileCredentialsProvider awsCreds = InstanceProfileCredentialsProvider.builder().build();

Region region = Region.US\_EAST\_1;

SqsClient sqsClient = SqsClient.builder().credentialsProvider(awsCreds).region(region).build();

String queueUrl = "https://sqs.us-east-1.amazonaws.com/002519620995/photoQueue.fifo";

public List<Message> receiveMessages() {

try {

// snippet-start:[sqs.java2.sqs\_example.retrieve\_messages]

ReceiveMessageRequest receiveMessageRequest = ReceiveMessageRequest.builder().queueUrl(queueUrl)

.maxNumberOfMessages(10).build();

List<Message> messages = sqsClient.receiveMessage(receiveMessageRequest).messages();

return messages;

} catch (SqsException e) {

System.err.println(e.awsErrorDetails().errorMessage());

System.exit(1);

}

return null;

}

public void deleteMessages(Message message) {

try {

DeleteMessageRequest deleteMessageRequest = DeleteMessageRequest.builder().queueUrl(queueUrl)

.receiptHandle(message.receiptHandle()).build();

sqsClient.deleteMessage(deleteMessageRequest);

System.out.println("Deleted :" + message.body());

} catch (SqsException e) {

System.err.println(e.awsErrorDetails().errorMessage());

System.exit(1);

}

}

}

**S3Service.java:**

package com.njit;

import software.amazon.awssdk.auth.credentials.InstanceProfileCredentialsProvider;

import software.amazon.awssdk.core.ResponseBytes;

import software.amazon.awssdk.regions.Region;

import software.amazon.awssdk.services.s3.S3Client;

import software.amazon.awssdk.services.s3.model.GetObjectRequest;

import software.amazon.awssdk.services.s3.model.GetObjectResponse;

import software.amazon.awssdk.services.s3.model.S3Exception;

public class S3Service {

private S3Client getClient() {

InstanceProfileCredentialsProvider awsCreds = InstanceProfileCredentialsProvider.builder().build();

Region region = Region.US\_EAST\_1;

return S3Client.builder().credentialsProvider(awsCreds).region(region).build();

}

public byte[] getObjectBytes(String bucketName, String keyName) {

S3Client s3 = getClient();

try {

GetObjectRequest objectRequest = GetObjectRequest.builder().key(keyName).bucket(bucketName).build();

// Return the byte[] from this object.

ResponseBytes<GetObjectResponse> objectBytes = s3.getObjectAsBytes(objectRequest);

return objectBytes.asByteArray();

} catch (S3Exception e) {

System.err.println(e.awsErrorDetails().errorMessage());

System.exit(1);

}

return null;

}

}

**WorkItem.java:**

package com.njit;

public class WorkItem {

private String key;

private String name;

private String confidence;

public void setKey(String key) {

this.key = key;

}

public String getKey() {

return this.key;

}

public void setName(String name) {

this.name = name;

}

public String getName() {

return this.name;

}

public void setConfidence(String confidence) {

this.confidence = confidence;

}

public String getConfidence() {

return this.confidence;

}

}