# AIM: To import CSV data into DynamoDB using Lambda and S3 Event Triggers:

# As part of my learning curve on DynamoDB and its interaction with various AWS services, Here S3 event triggers an action on a Lambda function to import CSV data from S3 Bucket and do some transformation and saved into a DynamoDB table using AWS Management Console.

# S3 Bucket

**Amazon Simple Storage (Amazon S3)** is an object storage service. You can store and protect any amount of data for a range of use cases, like backup and restore, data lakes, archive, mobile application, and websites. Infact this object storage service offering industry-leading scalability, data availability, security, and performance.

When someone creates an object or modifies (removes or updates) an object stored in your S3 bucket, S3 will trigger an event. You can use Lambda function to process such event notifications from S3.

So in this project we are adding a trigger to our Amazon S3 bucket to call our Lambda function whenever new data arrives. Amazon Lambda needs permission from the function's resource-based policy to invoke Lambda function so that we must give the S3 access permission to lambda.

# An IAM Role

# An IAM Role manages Amazon Web Services (AWS) users and their access to AWS accounts and services. It controls the level of access a user can have over an AWS account & set user, grant permission, and allows a user to use different features of an AWS account. Identity and access management is mainly used to manage users, groups, roles, and Access policies The account we created to sign in to Amazon web services is known as the root account and it holds all the administrative rights and has access to all parts of the account. The new user created an AWS account, by default they have no access to any services in the account & it is done with the help of IAM that the root account holder can implement access policies and grant permission to the user to access certain services.

# ****AWS Lambda****

**AWS Lambda** AWS Lambda is an Amazon serverless computing system that runs code and automatically manages the underlying computing resources like EC2. It is an event-driven computing service. It lets a person automatically run code in response to many types of events, such as HTTP requests from the Amazon API gateway, table updates in Amazon DynamoDB, and state transitions. It also enables the person to extend to other AWS services with custom logic and even creates its own back-end services. For example, just write the code and then upload it as a .zip file or any container image. The service works by running code on high-availability computer infrastructure.

# DynamoDB

# ****DynamoDB**** is a fully managed NoSQL database service that allows to create database tables that can store and retrieve any amount of data. It automatically manages the data traffic of tables over multiple servers and maintains performance. It also relieves the customers from the burden of operating and scaling a distributed database. Hence, hardware provisioning, setup, configuration, replication, software patching, cluster scaling, etc. is managed by Amazon.

# DynamoDB is fully integrated with AWS Backups. You can use the DynamoDB console, API, and AWS Command Line Interface (AWS CLI) to enable automatic backups and restore for your DynamoDB tables.

# Objectives:

* Create an S3 bucket.
* Upload a CSV file.
* Creating Lambda Function with a timeout of more than 1 minute, which contains the code to import the CSV data into DynamoDB.
* Create a Amazon DynamoDB table.
* All associated IAM roles needed for the solution, configured according to the principle of least privilege.
* Test the CSV Data Import in Lambda
* Adding Event Triggers to the S3 Bucket to call our Lambda function whenever new data arrives.
* Test the setup - Testing S3 Event Trigger to Import New Data into DynamoDB
* Cleanup

# Pre-requisites:

* AWS user account with admin access, not a root account.
* Create an IAM role.

## **1. Create an S3 bucket**

* On Amazon S3 Console / Create bucket / Under Create bucket and do general configuration.

**Bucket name:** bucket456789

**Create bucket.**

A screenshot of a computer

Description automatically generated

## **2. Upload a CSV file**

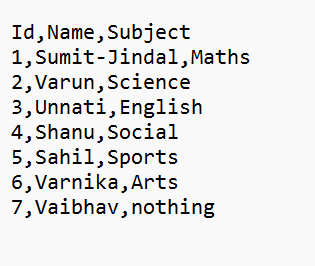
* Click on the S3 bucket – bucket456789
* Upload a csv file here

**File name:** friends.csv

* Under bucket456789, Objects, Upload / Under Upload, For Files and folders / Add files

**select** - friends.csv

**Upload**

****

## **3. Creating IAM Roles**

## All associated IAM roles needed for the solution, configured according to the principle of least privilege

* On the IAM dashboard / Roles / IAM dashboard / Create Role / Under Select trusted entity, Trusted entity type / Select AWS service / Under Use case / Select Lambda /
* Next
* Under Name, review, and create, Role details /

**Role name:** role5678  
**Create role**

A screenshot of a computer

Description automatically generated

* Click the role you just created **role5678**  
  drop-down Add permissions / select Attach policies
* Under Attach policy to csv-lambda-role / Search for **AmazonDynamoDBFullAccess** / Check the box / Attach poilicies
* Attach **AmazonS3FullAccess** in the same way.

A screenshot of a computer

Description automatically generated

## **4. Creating Lambda Function**

## We have to create a Lambda functionwith a timeout of more than 1 minute, which already contains the code to import the CSV data from S3 bucket and transform that file and then saved into DynamoDB.

* On the Lambda Console / Functions / Create function / Select Author from scratch / Under Basic information

**Function name:** function5678  
**Runtime:** From the drop-down choose **Python 3.12**

* Click on Change default execution role / select Use an existing Role
* Select the role you created just now - **csv-lambda-role**

**Create function**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

* Once the function is created, it will open the main page of the Lambda function.
* The python code do the following:
  + Imports the CSV file from S3 bucket.
  + Splits the CSV data into multiple strings.
  + Uploads data into the DynamoDB table.
* Remove the existing code in the function code environment window.
* Write the code into the lambda\_function.py

A screenshot of a computer

Description automatically generated

* After updating the code, Click on **Deploy** button to save the code.
* Change the function timeout as follows:
* Navigate to the Configuration
* click on General configuration / click on Edit
* In the Edit Basic setting / change the Timeout value to 1 min
* Click on Save button

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

# 5. Creating table in DynamoDB

* On DynamoDB Dashboard / Tables / Under Create table , Table Details

**Table Name:** table6789  
**Partition key:** Id  
**Type:** String

**Create Table**

**Status** should be Active

A computer screen shot of a computer

Description automatically generated

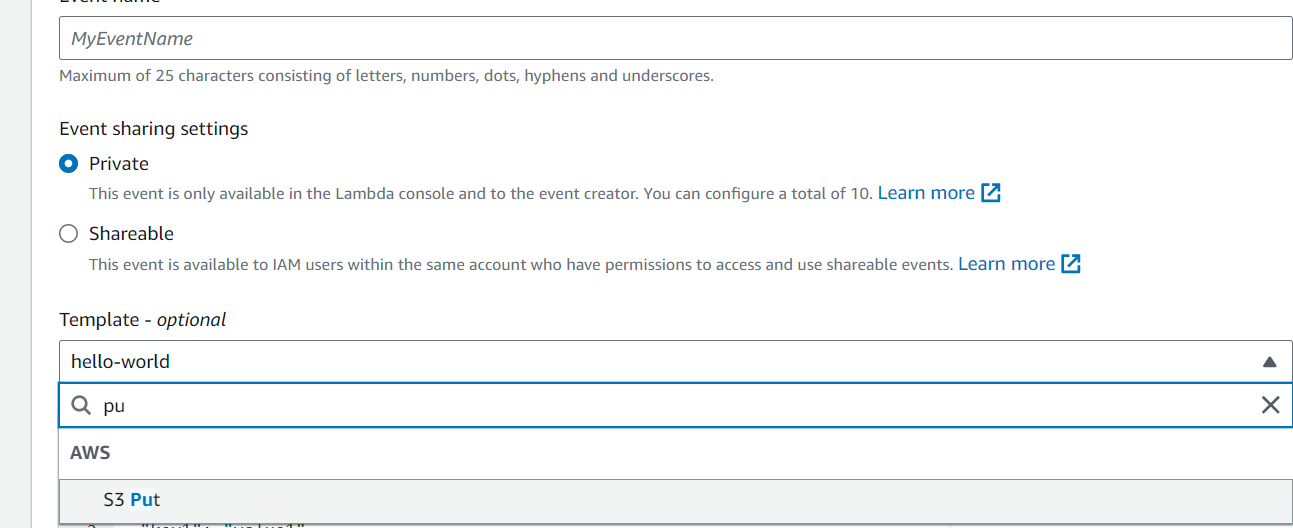
## **6. Test the CSV Data Import in Lambda:**

* In the csv-s3-lambda lambda function page, click on the Test tab.
* Configure to test event data as follows:

**Test event action:** Create a new event

**Name:** Event name - csv

**Template:** Select Amazon S3 Put, Upon selection, it will be displayed as s3-put



* Below in the JSON code:

**Under S3 → bucket → name →** Enter Students\_Details

**arn": "arn:aws:s3:::table6789"**

**Under S3 → object → Key →** Enter friends.csv

* Click on Create and then Save to save the changes.

**Note:** Make sure the S3 bucket name and file name are correct in the JSON.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

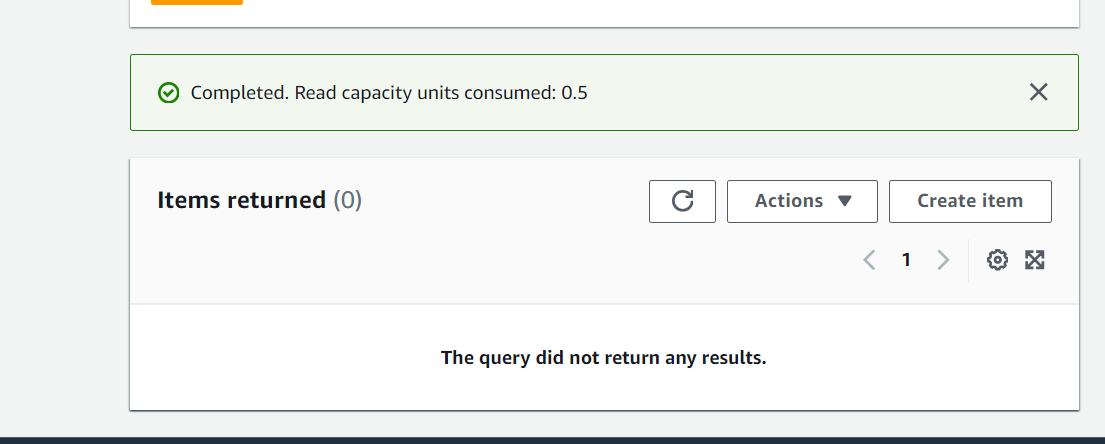
Description automatically generated

* Click on Test in top-right Corner to trigger the lambda function.
* Once the lambda function is successfully executed, you will be able to see a detailed success message with table data.

A screenshot of a computer

Description automatically generated

* Go to the DynamoDB table and then select the table6789 and click on Explore Table Items



## **7. Adding Event Triggers to the S3 Bucket to call our Lambda function whenever new data arrives.**

* On the S3 Console / Click on the s3 bucket named **jindal1234**
* Click on the Properties tab / go down to Event notifications.
* Click on Create event notification button
* Under General configuration /

**Name:**

**Suffix:** .csv

**All Object create events:** check

**Destination:** Select Lambda Function

**Lambda:** Select function5678

**Click on Save changes**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

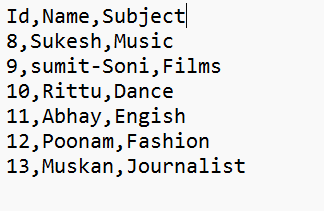
Description automatically generated

* Now every time a CSV file is uploaded to our S3 bucket, it will trigger the lambda to import the CSV data into the DynamoDB table

## **8. Test the setup - Testing S3 Event Trigger to Import New Data into DynamoDB**

**file:** friends1.csv

**Upload the friends1.csv file to the friends-s3 bucket**

****

A screenshot of a computer

Description automatically generated

* This upload event should triggered our Lambda function to import the CSV data into the DynamoDB table **table6789**.
* Go to the DynamoDB table **table6789** to see the changes.
* Click on the refresh button if items have not yet changed.
* You can see that new CSV data has been successfully imported into the DynamoDB table.

A screenshot of a computer

Description automatically generated

## **9. Cleanup**

* delete the Lambda function
* delete the S3 bucket
* delete the DynamoDB table
* delete IAM Role

# What we have done so far

* We have successfully created an Amazon DynamoDB Table.
* We have successfully created a Lambda function and configured it to import CSV data from S3 into DynamoDB.
* We have created an S3 event to trigger our Lambda function.
* We have tested the import of a new CSV file to the DynamoDB table.