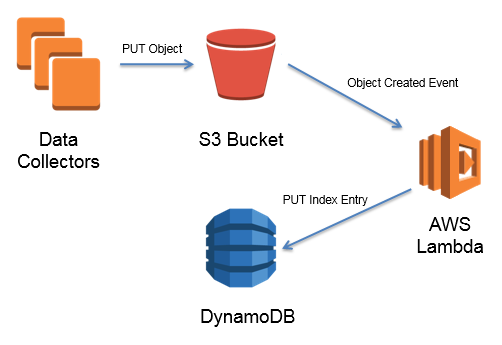
**Lab 4**



**Lab Scenario**

**Introduction:**

AWS Lambda is a serverless compute service that lets you run code without provisioning or managing servers, creating workload-aware cluster scaling logic, maintaining event integrations, or managing runtimes. With Lambda, you can run code for virtually any type of application or backend service - all with zero administration. Just upload your code as a ZIP file or container image, and Lambda automatically and precisely allocates compute execution power and runs your code based on the incoming request or event, for any scale of traffic. **You can set up your code to automatically trigger from 140 AWS services or call it directly from any web or mobile app**. You can write Lambda functions in your favorite language (Node.js, Python, Go, Java, and more) and use both serverless and container tools, such as AWS SAM or Docker CLI, to build, test, and deploy your functions.

**Objective:** Create the **Lambda function** to trigger with S3 bucket objects and store the object details in DynamoDB Table.

**Activities:**

Task 0: Create the Role of Full Dynamo DB service using IAM service.

Task 1: Create Lambda function and configure it with S3 bucket and DynamoDB database

Task 2: Add trigger for uploading objects in S3 Bucket

Task 3: Create a table in DynamoDB and check the items details

Task 4: Check the proper working of all the three Web servers with proper working.

Task 5: Take the snapshots of all performed tasks and create a doc/pdf of your enrolment number\_lab04 (Ex: E18CSE022\_Lab04) and upload the file on LMS.

Sample Code:

/\*

import boto3

from uuid import uuid4

def lambda\_handler(event, context):

s3 = boto3.client("s3")

dynamodb = boto3.resource('dynamodb')

for record in event['Records']:

bucket\_name = record['s3']['bucket']['name']

object\_key = record['s3']['object']['key']

size = record['s3']['object'].get('size', -1)

event\_name = record ['eventName']

event\_time = record['eventTime']

dynamoTable = dynamodb.Table('Lab04')

dynamoTable.put\_item(

Item={'unique': str(uuid4()), 'Bucket': bucket\_name, 'Object': object\_key,'Size': size, 'Event': event\_name, 'EventTime': event\_time})

\*/