AWS Serverless Solution with DynamoDB

Contents

Overview

In this section we will be covering on how to implement a serverless solution communicating with DynamoDB. Since this is going to be serverless, we will be utilising AWS Lambda Functions.

Implementing this solution would mean the application can communicate to our database with little to no downtime, so this is a great solution to offer to customers and, as engineers, this would mean less maintenance work for us.

When discussing implementing this kind of solution with a client, the important point to mention is that this solution is cost-effective; you only pay per request made to the AWS Lambda function. The alternative would be deploying the application onto a server, which would mean paying for the server uptime, and investing money and time into maintaining the server as well as the code.

Prerequisite

Topics

- IAM Roles
- IAM User Programmatic Access
- AWS CLI
- DynamoDB

IAM Roles

You will need to create an IAM role for AWS Lambda function to have DynamoDB full access.

This is for learning purposes only; please look at IAM permissions on how to provide permissions for AWS Lambda Read, Write, Update and Deleting on DynamoDB.

IAM User Programmatic Access

Ensure you have created a User with programmatic access and provide this user DynamoDB full access permission, as we require this to create a DynamoDB table.

AWS CLI

You will need the AWS CLI, as this would be the quickest method of creating a DynamoDB table. The commands for installing the AWS CLI can be found [here][https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-install.html]

Quickly installing on Ubuntu 18.04 LTS:

```
sudo apt update -y

sudo apt install awscli -y
```

DynamoDB

Create a DynamoDB table called `Movies`, with `Title` as the primary key and `Year` as its sort key.

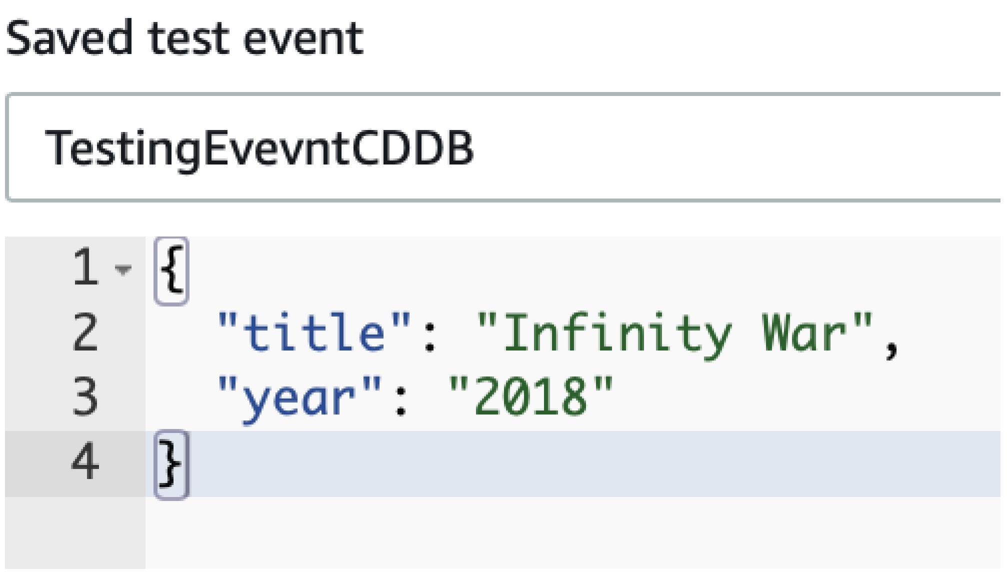
```
aws dynamodb create-table \
--table-name Movies \
--attribute-definitions AttributeName=Title,AttributeType=S
AttributeName=Year,AttributeType=S \
--key-schema AttributeName=Title,KeyType=HASH AttributeName=Year,KeyType=RANGE \
--provisioned-throughput ReadCapacityUnits=1,WriteCapacityUnits=1
```

Resource

GitLab link to the code source:
https://gitlab.com/qacdevops/dynamodb_project.git

Tutorial

1. Navigate to the AWS Console and sign in here
2. Search for AWS Lambda and ensure you are in the eu-west-1, which is Ireland.
3. Create a new function; this function is going to add items to our DynamoDB table. Call this function "dynamodb_add_item".
4. Provide this new function the IAM role created earlier. This role is supposed to have DynamoDB Full Access.
5. Change the Runtime to **Python3.8**. Proceed to creating your function.
6. Paste the create.py code provided from the GitHub link onto the Lambda code, replacing the existing default code, and click save.
7. This code requires input parameters to be passed in. Click on the **Test**, next to the **Save** button. You will be prompted to create a test case. This is essentially configuring when you want to test this Lambda function and what parameters should be passed in. Whether it uses it or not, you will need to configure this. For this example, we will be configuring the test case as follows:



TestingEvevntCDDB

```
{
  "title": "Infinity War",
  "year": "2018"
}
```

8. Click the **Test** button. This should end with a success message from AWS Lambda. You can double check it has worked by navigating to DynamoDB, going into the `Movies` table that was created and looking at the items inside this table.

9. You will need to repeat this process for Get, Update and Delete. The code for all of these is provided in the GitHub link.

Exercises

There are no exercises for this module.