06

Consolidando o seu conhecimento

Chegou a hora de você seguir todos os passos realizados por mim durante esta aula. Caso já tenha feito, excelente. Se ainda não, seguem os principais passos para integrar o S3 e DynamoDB através do Lambda:

No serviço AWS Lambda crie uma nova função:

- Nome: `PhotoCollection_AtualizaDynamoDB`
- Runtime: Python 3.7
- Selecione o serviço S3 como *trigger*
  - Coloque o nome do seu bucket
  - Escolhe `PUT` como *event type*
  - Sufixo `.jpg`
  - Use a implementação no *final desse exercício*
  - Salve tudo

No serviço Amazon S3 escolhe o seu bucket e vá nas propriedades:

- Escolhe o card `Events`
- Adicione uma nova notificação
  - Selecione `Permanently deleted` e adicione sufixo `.jpg`
  - Escolhe *Lambda function* no `Send to` e o nome da função Lambda criada anteriormente
  - Salve tudo

No serviço IAM escolhe o link `Policies`:

- Crie uma nova policy
- No service selecione o serviço DynamoDB
- No actions selcione as permissões `List`, `Read`, `Write`
- No resources coloquei o ARN da tabela no DynamoDB

No serviço IAM escolhe o link `Roles`:

- Abra o role já criada pelo Lambda
- Adicione uma descrição
- Associe a policy criado no passo anterior

Para testar, faça um upload de uma foto JPG para o seu bucket. Fique de olho no logs dentro do CloudWatch.

Segue o código Python da função Lambda:

```
import boto3

resource_dynamodb = boto3.resource('dynamodb')
```

```
def lambda_handler(event, context):
    arquivo = event['Records'][0]['s3']['object']['key']
    evento = event['Records'][0]['eventName'].split(':')
    evento = evento[1]
    main(arquivo,evento)


def extrai_arquivo(arquivo):
    atributos = arquivo[:-4]
    atributos = tuple(atributos.split('-'))
    return (atributos)


def atualiza_database(atributos,evento):
    table = resource_dynamodb.Table('PhotoCollection')
    if evento == 'Put':
        response = table.put_item(
            Item={
                'id': int(atributos[0]),
                'assunto': atributos[1],
                'colecao': atributos[2],
                'descricao': atributos[3]
            }
        )
    if evento == 'Delete':
        response = table.delete_item(
            Key={
                'id': int(atributos[0])
            }
        )

    if response['ResponseMetadata']['HTTPStatusCode'] == 200:
        print(f'DB atualizado: {atributos}')
    else:
        print('ERRO de atualizacao')

def main(arquivo,evento):
    atributos = extrai_arquivo(arquivo)
    atualiza_database(atributos,evento)
```