



# Salesforce AWS Integration using API gateway and Lambda

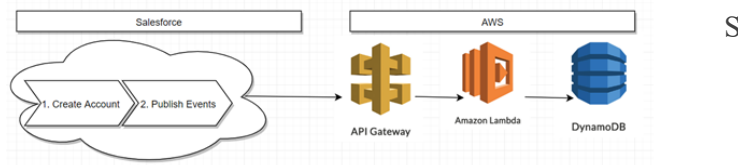
Published on May 27, 2019

**Rajesh K.**

Senior Salesforce Development Engineer at Akamai | Ex-Morgan Stanley

5 articles

[+ Follow](#)



## *Prerequisite*

1. AWS Free Tier Account ,
2. Salesforce Developer Account
3. Basic to advance knowledge of Salesforce Architecture and Coding Pattern.
4. Basic to Intermediate knowledge AWS lambda, Dynamo DB and server less processing.

## *Consideration*

1. All AWS service creation has been kept simple .. i.e most part like AWS lambda has been configured to have admin access role .
2. Connection b/w Salesforce and AWS api gateway has been made through open authentication.
3. I will have separate article for security consideration b/w salesforce and AWS connection

## *Use Case Flow*

1. Account is created in Salesforce



Messaging



3. Amazon Gateway receives the data and transform and pass to an

4. Amazon Lambda has logic to pass this data to dynamo DB and in



## Salesforce Changes

a) Create Platform Event name “**SubscribetoAWSLambda**” (you can name it as you want)

b) Create two custom field in platform event “**AccountId**” & **Name** in (Refer below image for more detail)

Platform Event  
SubscribetoAWSLambda

Standard Fields (2) | Custom Fields & Relationships (2)

[Edit](#) [Delete](#)

Platform Event Definition Detail		Description	
Singular Label	SubscribetoAWSLambda	Deployment Status	Deployed
Plural Label	SubscribetoAWSLambda		
Object Name	SubscribetoAWSLambda		
API Name	SubscribetoAWSLambda_e		
Event Type	High Volume		
Created By	User User: 5/26/2019 12:11 PM	Modified By	User User: 5/26/2019 12:11 PM

Standard Fields				
Action	Field Label	Field Name	Data Type	Indexed
	Created By	CreatedBy	Lookup(User)	
	Created Date	CreatedDate	Date/Time	
	Replay ID	ReplayId	External Lookup	

Custom Fields & Relationships					
Action	Field Label	API Name	Data Type	Indexed	Modified By
<a href="#">Edit</a>   <a href="#">Del</a>	AccountId	Accountid__c	Text(255)		User User: 5/26/2019 12:11 PM
<a href="#">Edit</a>   <a href="#">Del</a>	Name	Name__c	Text(255)		User User: 5/26/2019 12:12 PM

c) Create process builder to publish data in above vents when account is created ,

d) Create below class (Please note this class can be optimised for bulk operations and AWS can be read from custom Metadata/Custom Setting) . Please note that endpoint will be replaced when we create API gateway in later section.

```

public class AWSCallout
{
    @future(callout = true)
    public static void callAwsgateway(String recordId,String Name)
  
```





```

HttpRequest request = new HttpRequest();

//Please set the endpoint of API gateway which we are going to create in

request.setEndpoint('awsapigateway');

request.setMethod('POST');

request.setHeader('Content-Type', 'application/json');

// Set the body as a JSON object

request.setBody('{ "id": "' + recordId + '", "Name": "' + Name + '" }');

HttpResponse response = http.send(request);

System.debug('response===' + response.getBody());

// Parse the JSON response

if (response.getStatusCode() != 201) {

    System.debug('The status code returned was not expected: ' +

        response.getStatusCode() + ' ' + response.getStatus());

} else {

    System.debug(response.getBody());

}

}

}

}

```

e) Create Subscriber on platform Event (basically platform event Trigger)

```

trigger callAWSAPIgateway on SubscribetoAWSLambda__e (after Insert) {

    for (SubscribetoAWSLambda__e saws : Trigger.new)

    {

        AWSCallout.callAwsgateway(saws.AccountId__c,saws.Name__c);

    }

}

```

f) Do not forget to add the actual end point mentioned in Step c) to remote site setting.

### AWS Changes

> Create below dynamodb table with "id" as Primary partition key





Search

Choose a table ...

Actions

Name
<input checked="" type="radio"/> Account

OverviewItemsMetrics

Create itemActions

Scan: [Table] Account: id

Scan[Table] Account: id

+ Add filter

Start search

	id
<input type="checkbox"/>	000000111
<input type="checkbox"/>	0016D00000ANePGQA1
<input type="checkbox"/>	0016D00000ANePLQA1
<input type="checkbox"/>	22222222222222222222222222222222
<input type="checkbox"/>	id001
<input type="checkbox"/>	id002
<input type="checkbox"/>	id003
<input type="checkbox"/>	id009

> Login to AWS and go to IAM and create custom role with Administrator access policy .I have used this role for AWS lambda , please note in real time situation we should not give admin access security should be limited here I have used just for demo purpose to avoid any access related issue .

Roles > customAdmin

Summary

Role ARN

Role description

Instance Profile ARNs

Path

Creation time

Maximum CLI/API session duration

Permissions

Trust relationships

Tags

Access Advisor

Revoke sessions

Permissions policies (1 policy applied)

Attach policies

Policy name	Policy type
AdministratorAccess	AWS managed policy

> In services search for lambda function and create below lambda function and in execution role provide role created in above steps .



Messaging



Search



Try Premium Free  
for 1 Month

```
const dynamodb = new AWS.DynamoDB({apiVersion: '2012-08-10'});

exports.handler = (event, context, callback) => {
  dynamodb.putItem({
    TableName: process.env.tableName,
    Item: {
      "id": {
        S: event.id
      },
      "Name": {
        S: event.Name
      }
    }
  }, function(err, data) {
    if (err) {
      console.log(err, err.stack);
      callback(null, {
        statusCode: '500',
        body: err
      });
    } else {
      callback(null, {
        statusCode: '200',
        body: 'Hello ' + event.id + event.Name + '!'
      });
    }
  })
};
```

Below image shows the setup for lambda function



Messaging



Search



Try Premium Free  
for 1 Month

### Environment variables

You can define environment variables as key-value pairs that are accessible from your function code. These variables are stored in the AWS Lambda console.

► Encryption configuration

### Tags

You can use tags to group and filter your functions. A tag consists of a case-sensitive key-value pair. [Learn more](#)

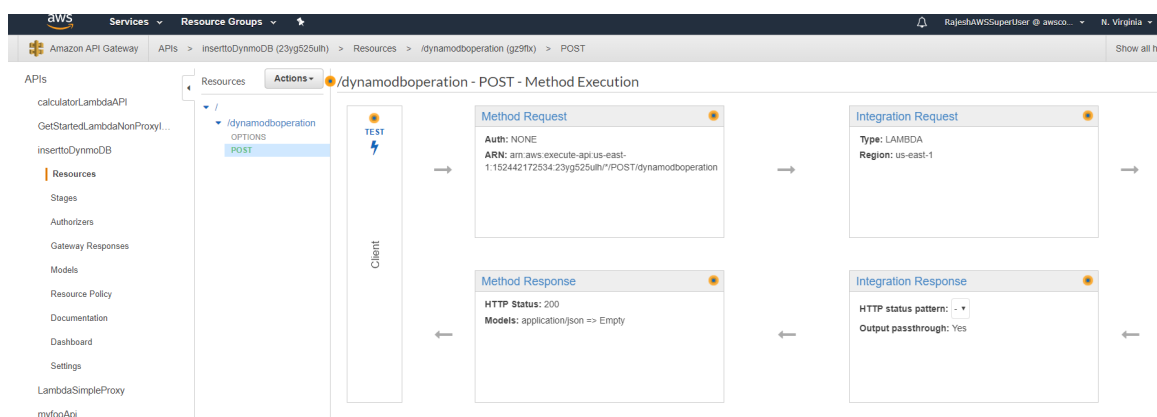
### Execution role

Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

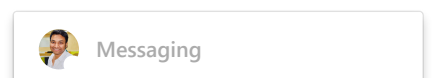
Existing role  
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

[View the customAdmin role](#) on the IAM console.

> Create below API gateway



> Create below model inside the gateway





Search



Try Premium Free  
for 1 Month

APIs

- calculatorLambdaAPI
- GetStartedLambdaNonProxyl...
- inserttoDynamoDB
- Resources
- Stages
- Authorizers
- Gateway Responses
- Models**
- Resource Policy
- Documentation
- Dashboard
- Settings
- LambdaSimpleProxy

Models

- Empty
- Error
- inputMap

Create

### Update Model

Make changes to your model in the f

Model name

Content type

Model description

Model schema\*

```
1 {
2   "type": "object",
3   "properties": {
4     "id": { "type": "string",
5     "Name": { "type": "stri
6   },
7   "title": "inputMap"
8 }
```

> Configure post method as below .

Resource Groups

inserttoDynamoDB (23yg525ulh) > Resources > /dynamodboperation (gz9flx) > POST

Resources

Actions

Method Execution /dynamodboperation - POST - Method Request

Provide information about this method's authorization settings and the parameters it can receive.

Settings

Authorization NONE

Request Validator Validate body

API Key Required false

URL Query String Parameters

HTTP Request Headers

Request Body

Content type	Model name
application/json	inputMap

Add model

SDK Settings

> Deploy API gateway to stage (you can stage name whatever you want) I have named test.

> Go to "test" Stage and click on POST method and note down the Invoke URL and replace into your code in Salesforce Changes Step c) .

> Test the functionality by creating account in Salesforce which will flow to dynamo db table created earlier .



Messaging



Search



Try Premium Free  
for 1 Month

purpose for production code need to include our security rules

### Published by

**Rajesh K.**

Senior Salesforce Development Engineer at Akamai | Ex-Morgan Stanley  
Published • 1y

Salesforce-AWS Dynamo DB Integration with AWS lambda and API gateway .

Like Comment Share

### Reactions



### 7 Comments

Most Relevant ▾



Add a comment...



**Anirban Goswami** • 3rd+  
Senior Data Engineer - Apple

1y ...

truly fine stuff

| · 1 Reply

**Rajesh K.** • 2nd  
Senior Salesforce Development Engineer at Akamai | Ex-Morgan Stanley  
Thanks **Anirban** !!!

1y ...

|



**Siddharth Raj** • 2nd  
Senior Data Scientist at Accenture

1y ...

great job Rajesh..... really helpful

| · 1 Reply

**Rajesh K.** • 2nd  
Senior Salesforce Development Engineer at Akamai | Ex-Morgan Stanley  
Thank you Bhaiya !!

1y ...



Messaging





Search



Try Premium Free  
for 1 Month

## Rajesh K.

Senior Salesforce Development Engineer at Akamai | Ex-Morgan Stanley

+ Follow

in Rajesh K.

Salesforce using JAVA

on LinkedIn

'Javascript Promises' in Lightning  
Web Component.

Rajesh K. on LinkedIn

Sharing Javascript code in  
Web Component(LWC)

Rajesh K. on LinkedIn

Articles

LinkedIn

About

Community Guidelines

Privacy & Terms

Sales Solutions

Safety Center

Accessibility

Careers

Ad Choices

Mobile

Talent Solutions

Marketing Solutions

Advertising

Small Business



Questions?

Visit our Help Center.



Manage your account and  
privacy

Go to your Settings.

Select Language

English (English)

LinkedIn Corporation © 2020



Messaging