**APIGATEWAY POST**

Accept:application/json

Content-Type: application/json

{

"email": "card@gmail.com",

"name": "sant",

"lastName": "card",

"age": "29",

"city": "Envigado",

"country": "Colombia",

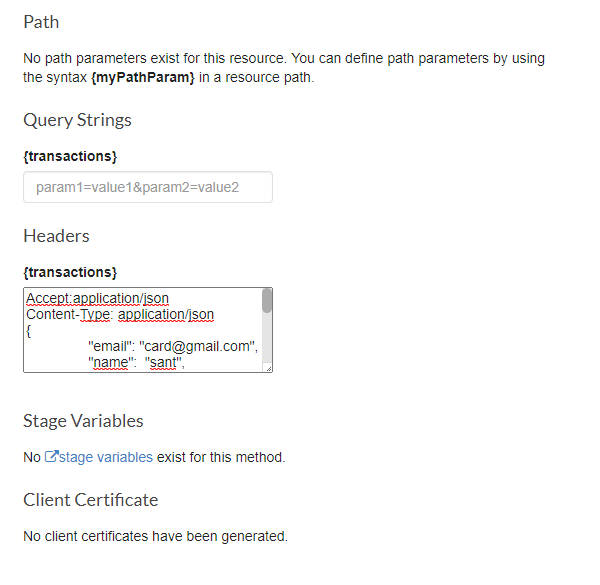
"EPS": "Sura",

"weight": "75",

"height": "185",

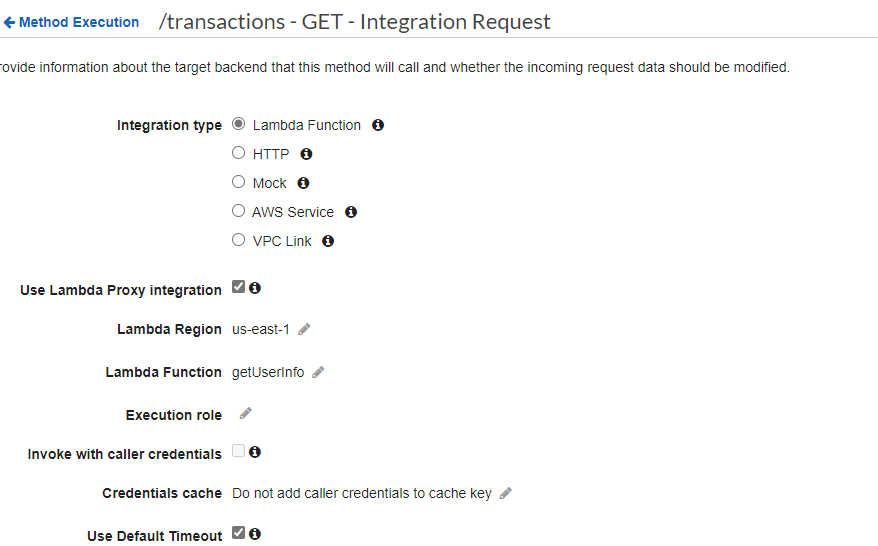
"profession": "engineer"

}



**APIGATEWAY GET**

Se debe habilitar use lambda proxy integration en Integration Request



para probar

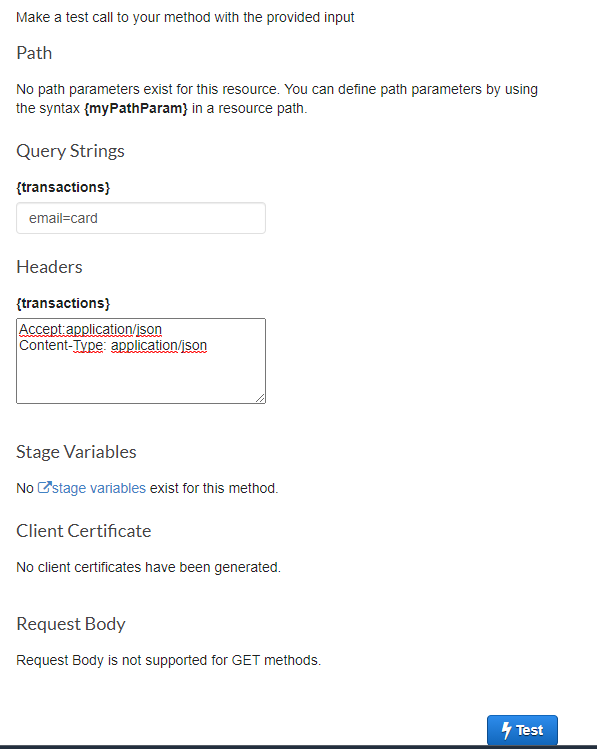
QUERY STRINGS

email=card

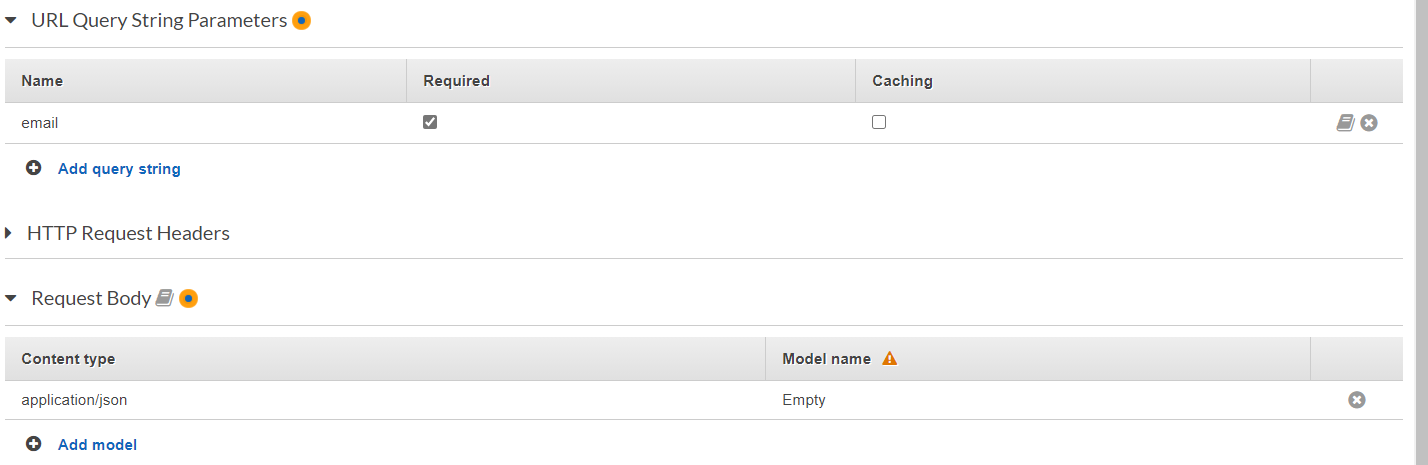
HEADERS

Accept:application/json

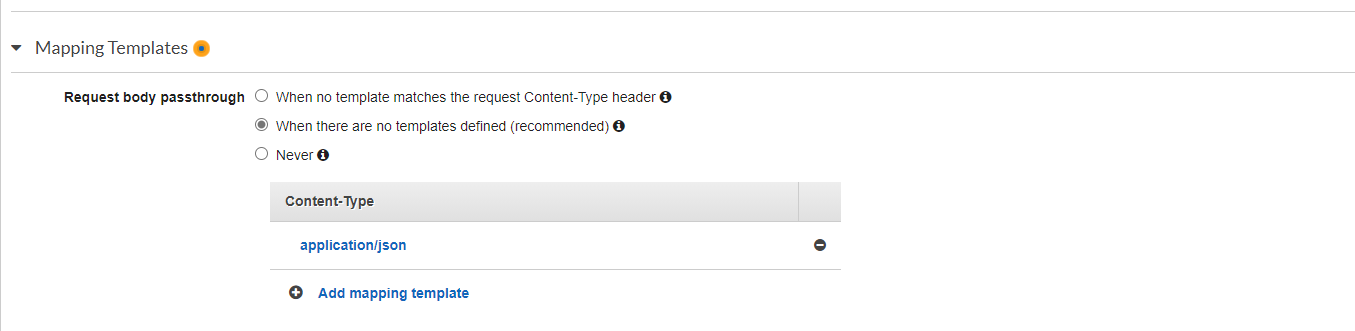
Content-Type: application/json

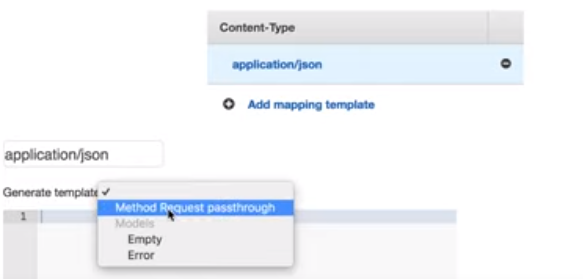


Para poder recibir los parámetros que envía la aplicación o postman se deben espicificar en URL Query String Parameters



Además en el Integration Request se debe agregar un Mapping Templates. Se puede agregar el que trae por defecto (Method Request passthrough)





[Paso 3: Crear, leer, actualizar y eliminar elementos - Amazon DynamoDB](https://docs.aws.amazon.com/es_es/amazondynamodb/latest/developerguide/GettingStarted.NodeJs.03.html)

**Lambda Operations**

**Get User information**

// This is just a sample script. Paste your real code (javascript or HTML) here.

var aws = require('aws-sdk');

var ddb = new aws.DynamoDB.DocumentClient();

exports.handler = (event, context, callback) => {

// TODO implement

// if(event.httpMethod == 'GET'){

var email = event.params.querystring.email

var params = {

TableName: 'User',

Key: {

"email": email

}

};

ddb.get(params, function(err, data) {

if (err) {

//callback(err, null)

callback(null, {

statusCode: 500,

body: 'Error! -->' + err,

});

} else {

// callback(null,data)

callback(null, {

statusCode: 200,

body:JSON.stringify(data["Item"])

});

}

});

};

**Save user**

var aws = require('aws-sdk');

var ddb = new aws.DynamoDB.DocumentClient();

exports.handler = (event, context, callback) => {

// If the required parameters are present, proceed

// if (event.request.userAttributes.sub) {

var params = JSON.parse(event.body)

// -- Write data to DDB

let ddbParams = {

TableName: 'User',

Item: {

"email": params.email,

"name": params.name,

"lastName": params.lastName,

"age": params.age,

"city": params.city,

"country": params.country,

"EPS": params.EPS,

"weight": params.weight,

"height": params.height,

"profession": params.profession,

"isDoctor": params.isDoctor,

},

};

let response = {};

// Call DynamoDB

ddb.put(ddbParams, function(err, data) {

if (err) {

//callback(err, null)

callback(null, {

statusCode: 500,

body: 'Error! -->' + err,

});

} else {

callback(null, {

statusCode: 200,

body: 'Ok!',

});

}

});

};

**Operaciones con GET, UPDATE Y POST**

// This is just a sample script. Paste your real code (javascript or HTML) here.

var aws = require('aws-sdk');

var ddb = new aws.DynamoDB.DocumentClient();

exports.handler = async (event, context, callback) => {

var publicationToSave = JSON.parse(event.body)

var method = event.httpMethod;

let items;

const itemsAll = [];

/\* var publication = {

"id": "",

"title": "",

"description": "",

"comments": [{

"email": "",

"email": ""

}]

}\*/

var publication;

var params = {

TableName: 'publications'

};

let ddbParams;

switch (method) {

case 'GET':

do {

items = await ddb.scan(params).promise();

items.Items.forEach((item) => itemsAll.push(item));

params.ExclusiveStartKey = items.LastEvaluatedKey;

} while (typeof items.LastEvaluatedKey != "undefined");

callback(null, {

statusCode: 200,

body: JSON.stringify(itemsAll)

});

break;

case 'POST':

savePublications(publicationToSave);

break;

case 'PUT':

/\*

var paramsToFind = {

TableName: 'publications',

Key: {

"id": publicationToSave.id

}

};

var pu;

await ddb.get(paramsToFind, function(err, data) {

pu = JSON.stringify(data);

if (err) {

callback(null, {

statusCode: 500,

body:'Error'+err

});

} else {

}

})\*/

var response = updatePublication(publicationToSave.id, publicationToSave);

if(response){

callback(null, {

statusCode: 200,

body:'Ok'+publication

});

}

else{

callback(null, {

statusCode: 500,

body: 'Error!'+response[0]

});

}

break;

}

function savePublications(publicationToSave) {

if (publicationToSave.comments == undefined) {

ddbParams = {

TableName: 'publications',

Item: {

"id": publicationToSave.id,

"title": publicationToSave.title,

"description": publicationToSave.description,

"userCreator": publicationToSave.userCreator

}

}

} else {

const commentArray=[];

for(var i = 0; i <= publicationToSave.comments.length; i++){

commentArray.push(publicationToSave.comments[i]);

}

ddbParams = {

TableName: 'publications',

Item: {

"id": publicationToSave.id,

"title": publicationToSave.title,

"description": publicationToSave.description,

"userCreator": publicationToSave.userCreator,

"comments": [commentArray]

}

}

}

callback(null, {

statusCode: 200,

body: 'Ok! ' + ddbParams.Item,

});

// Call DynamoDB

ddb.put(ddbParams, function(err, data) {

if (err) {

callback(null, {

statusCode: 500,

body: 'Error! ' + err,

});

} else {

callback(null, {

statusCode: 200,

body: 'Ok! '

});

}

});

}

function updatePublication(id, publicationToSave) {

var response=[];

var params = {

TableName: 'publications',

Key: {

"id": id

}

};

const commentArray=[];

for(var i = 0; i <= publicationToSave.comments.length; i++){

commentArray.push(publicationToSave.comments[i]);

}

ddbParams = {

TableName: 'publications',

Item: {

"id": publicationToSave.id,

"title": publicationToSave.title,

"description": publicationToSave.description,

"comments": commentArray

}

}

ddb.put(ddbParams, function(err, data) {

if (err) {

response[0].push(1);

response[1].push(err);

} else {

response[0].push(1);

}

});

return response;

}

};