

**Qualcomm Car-to-Cloud Platform**

**API AUTHENTICATION WITH LAMBDA AUTHORIZER**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Prepared By / Last Updated By** | **Reviewed By** | **Approved By** |
| **Name** | Chinchu P Shaji | Sudheer | Sudheer |
| **Role** | Associate | Sr Manager | Sr Manager |
| **Signature** |  |  |  |
| **Date** | June 09, 2021 | June 10, 2021 | June 10, 2021 |

June 9, 2021

COGNIZANT

Contents

[1. Cognito configurations - User Pool and User Creation 2](#_Toc74258734)

[1.1. User Pool Creation 2](#_Toc74258735)

[1.2. Add App Client 2](#_Toc74258736)

[1.3. Add User 3](#_Toc74258737)

[**2.** **Lambda Function Creation** 4](#_Toc74258738)

[**3.** **API Gateway Configuration** 4](#_Toc74258739)

[3.1. Create the REST API Configurations 4](#_Toc74258740)

[3.2. Create Lambda Authorizer 5](#_Toc74258741)

[3.3. Deploy the API to the API Gateway 6](#_Toc74258742)

[3.4. Swagger API Import Configuration and Deployment 8](#_Toc74258743)

This document specifies the stepwise details to authenticate an API for a Cognito user using API Gateway through Lambda Authorizer.

# Cognito configurations - User Pool and User Creation

* 1. User Pool Creation

First we need to create a **User Pool** using **Cognito**

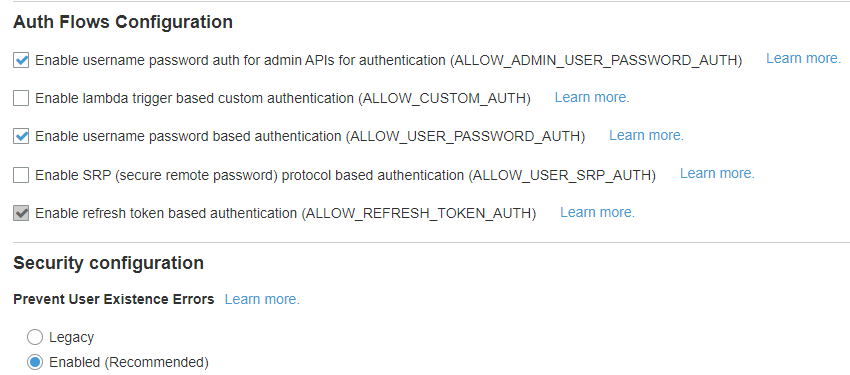
1. Login to the AWS console
2. Click on **Services -> Cognito**
3. Click on **Manage User Pools**
4. Click on **Create a User Pool**
5. Enter the **Pool name (e.g. :** DeviceRegPool)and click on **Review defaults**
6. Click on **create pool** button
7. Note the **pool id**

A user pool with the specified name will be created.

* 1. Add App Client

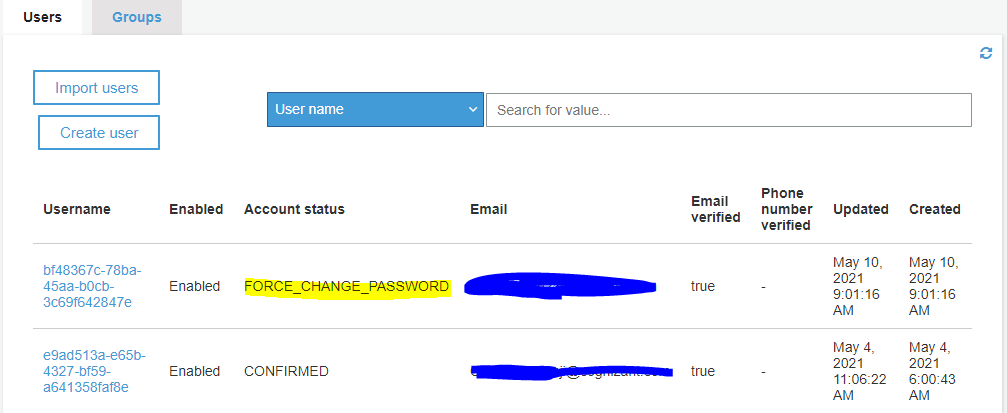
Once the user pool is created, we can see it after clicking on the Manage user pools button. Follow below steps to add an app client to the user pool.

1. Click on to the above created user pool and go to add an **app client** link
2. Add client name (eg : *deviceRegClient)*
3. Provide default values for token expiration date details
4. Click on the **Generate client secret** checkbox
5. Check **Auth Flows Configurations** and **Security configurations** as below



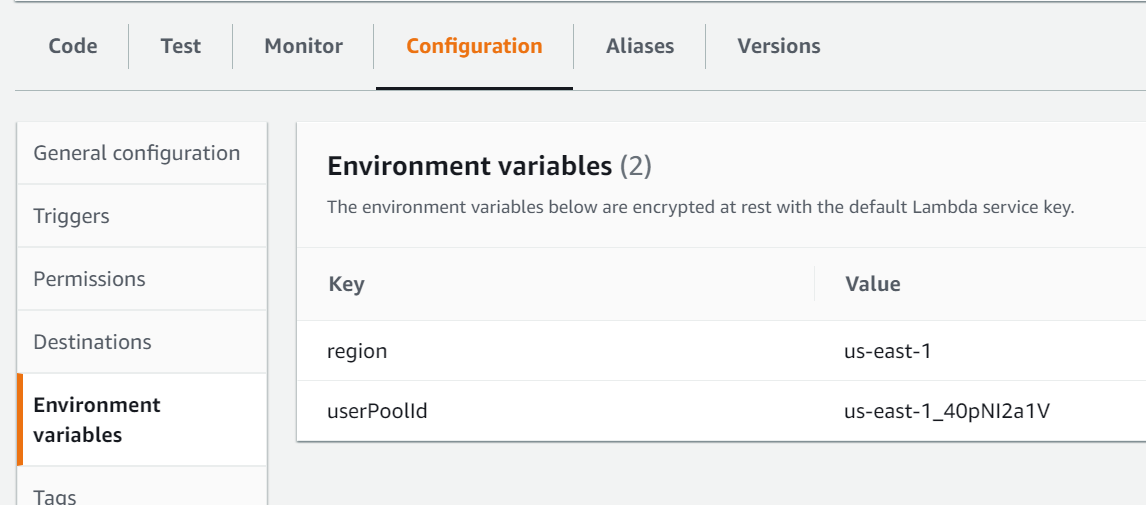
1. Click on **Create app client** button.
2. App client is created. Make a note of **App client id** and **App client secret**
   1. Add User

Follow below steps to create a user on the user pool

1. Click on the user pool and select **Users and groups** under **General settings**
2. Click on **Create user** button
3. Provide the details (username, temporary password, phone number/email)
4. Click on **Create User**
5. New user will be create with Account status as **FORCE\_CHANGE\_PASSWORD**. Once the user changes the temporary password in the first time login, the status will be changed to **CONFIRMED**

# **Lambda Function Creation**

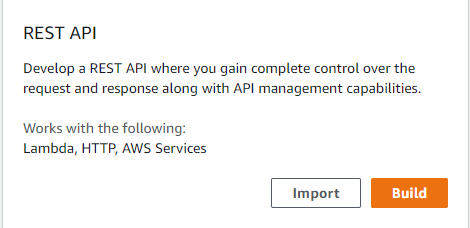
* + 1. Login to the AWS console
    2. Click on **services** -> **lambda**
    3. Click on **create function** and select **Author from scratch**
    4. Enter function name
    5. Select **java 11(corretto)** from **run time** dropdown
    6. Click on **create function**
    7. Go to the function and add com.c2c.cognito.lambda.handler.APIGatewayAuthorizerHandler::handleRequest as Handler info in RunTime settings
    8. Upload the jar in **source-code** section
    9. Add env variables in **configuration -> Environment variables**, add your corresponding **region** and **userPoolId**



# **API Gateway Configuration**

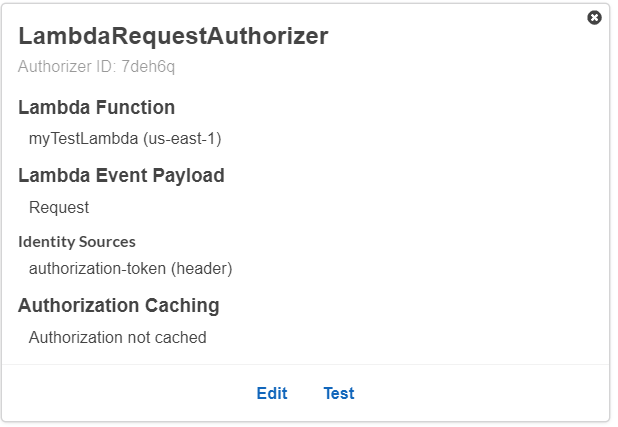
* 1. Create the REST API Configurations

1. Login to the AWS console
2. Click on **Services -> API Gateway**
3. Click on **Create API**
4. Choose the API type as **REST API** and click on **Build**

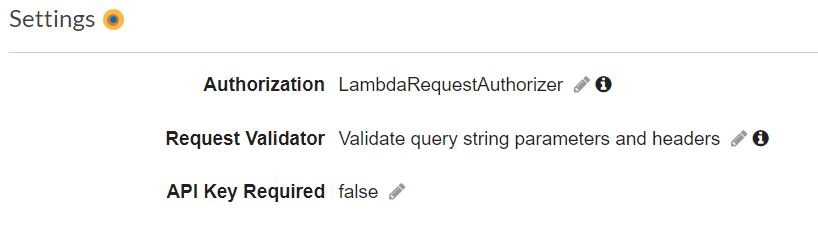


1. Choose the protocol as **REST**
2. Select **New API** from **Create New API** section and provide **API Name**, **Description** and **Endpoint Type** as **Regional**
3. Click on **Create API** and a new API will be created
   1. Create Lambda Authorizer

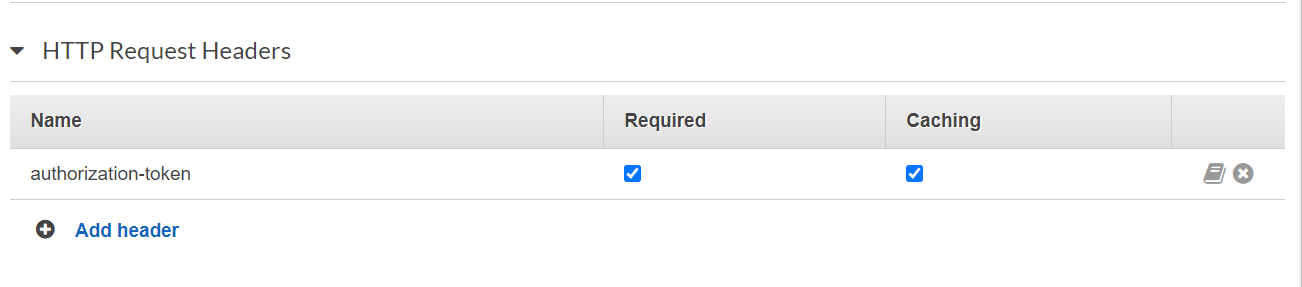
Lambda authorizer enable you to control access to your APIs using Amazon Cognito User Pools. We can define the user pool and header name details over here



1. Move on to the Authorizers Tab and click on **Create New Authorizer** Button
2. Enter Authorizer **name** and click on the **Lambda** radio button
3. Select the **lambda** function and select **lambda event payload** as **Request**
4. Add **identity sources** as in the above figure
5. Click on **create** and the lambda Authorizer will be created
   1. Deploy the API to the API Gateway
      1. **Create Resource**
         1. Move on to the **Resources** tab under the above created API section
         2. Select **create resource** from the **Actions** dropdown
         3. Enter the **Resource name** and click on **create resource** button
      2. **Create Method**
         1. Select **create method** from the **Actions** dropdown
         2. Select the appropriate **Request type** from the dropdown and click on the **tick** mark
         3. Choose integration type as **HTTP**
         4. Provide the **Endpoint** **URL** and Content handling as **Passthrough**
         5. Click on **Save**
      3. **Method Request**
         1. Click on the Method Request link
         2. In settings, do the configuration as below



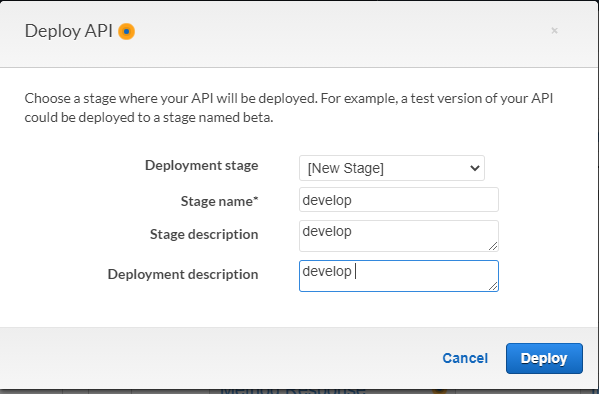
* + - 1. Add headers and query string parameters as below

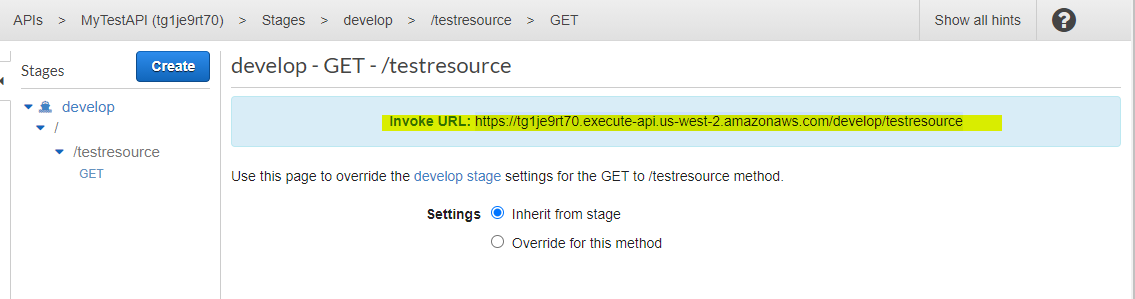


* + 1. **Integration Request**
       1. Click on the Integration Request link
       2. Add HTTP header as below



* + 1. **Deploy API**



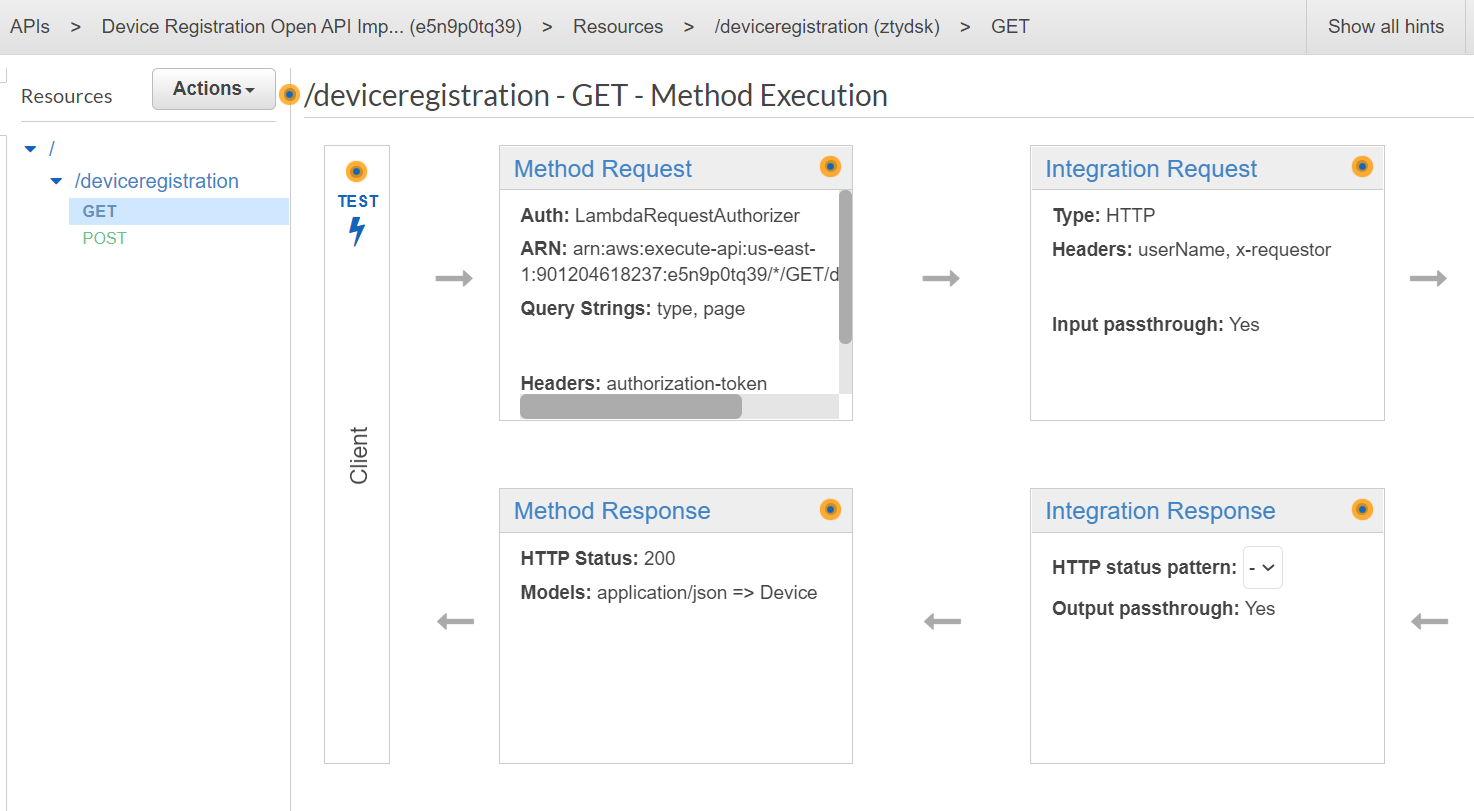
1. Select **Deploy API** from **Actions** dropdown
2. Deployment stage = New Stage
3. Provide stage name as **develop**
4. Click Deploy
5. This will generate an Invoke URL which can be used for Authentication
   1. Swagger API Import Configuration and Deployment
      1. Login to the AWS console
      2. Click on **Services -> API Gateway**
      3. Click on **Create API**
      4. Choose the API type as **REST API** and click on **Import**
      5. Choose the protocol as **REST**
      6. Select **Import from swagger or Open API 3** from **Create New API** section
      7. Paste the Open API configuration for the corresponding swagger file definitions

Attaching a sample Open API configuration for a **GET** (Deviceregistration ) and **POST**(Mock method).



This will automatically configure the resources and methods from your swagger definition and also will map the methods with a valid authorizer

* + 1. Select the **endpoint type** as **regional** and click on the **import** button
    2. /deviceregistration APIs with a **GET** and **POST** method will be created



* + 1. Deploy the API as mentioned in **2.3.3**
    2. Enable Cloud watch logs from stages
    3. This will generate an Invoke URL which can be used for Authentication
    4. After execution, can see the logs in CloudWatch