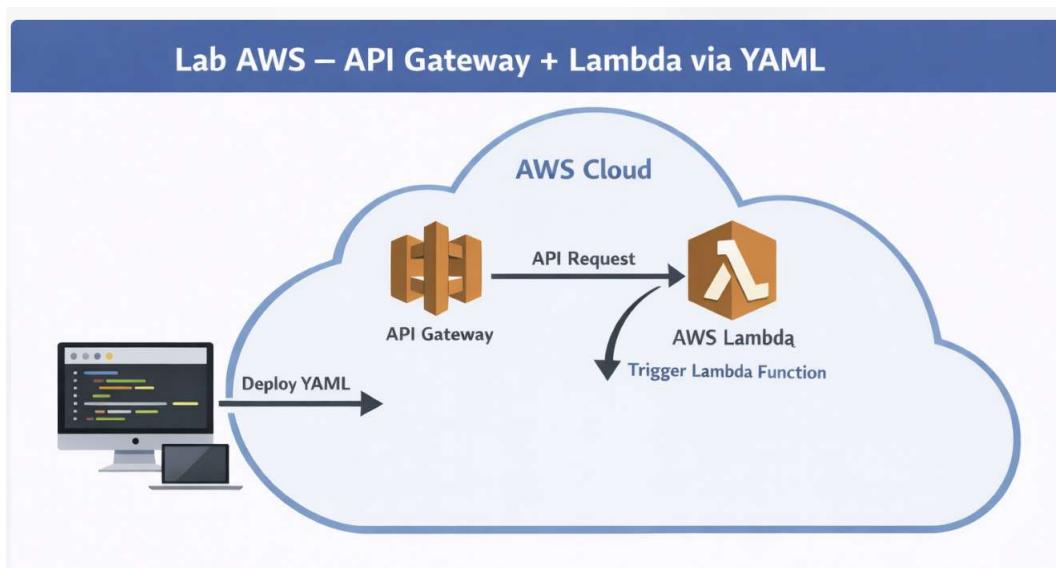


🚀 Lab AWS – API Gateway + Lambda via YAML

Objectif : faire fonctionner un Endpoint `/dev/hello` importé en **YAML OpenAPI**, exactement comme une création manuelle dans la console AWS.

Le Endpoint est de type **HTTP API**

- ◆ Lambda simple (Hello World)
- ◆ Import OpenAPI dans API Gateway
- ◆ Ajout de la permission Lambda → API Gateway
- ◆ Déploiement du stage dev
- ◆ Test du endpoint → réponse OK



Sommaire

Étape 1 : Préparer la Lambda.....	2
Étape 2 : Préparer le YAML correct.....	3
Étape 3 : Importer le YAML dans API Gateway	3
Étape 4 : Ajouter la permission Lambda pour API Gateway.....	4
Étape 5 : Déployer le stage dev	4
Étape 6 : Vérifier l'intégration avec Lambda	5

Étape 7 : Tester le endpoint	5
8 <input checked="" type="checkbox"/> Résumé	6
9 Divers écrans	6
10 Annexe 1	7
11 Références.....	8

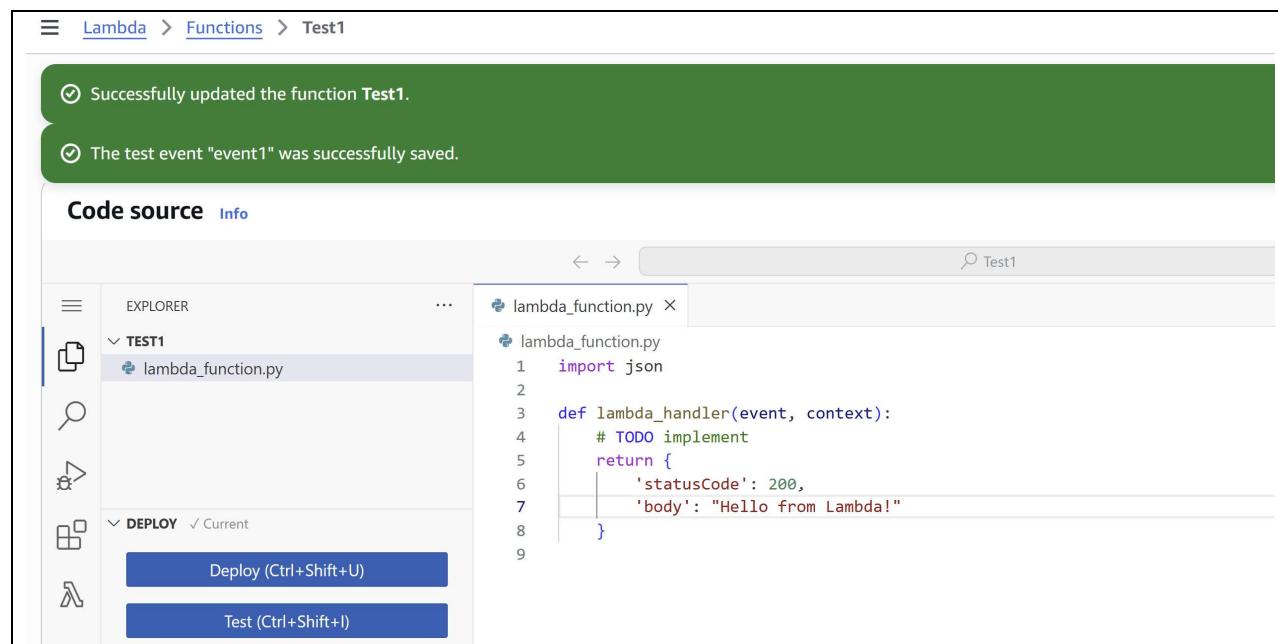
Étape 1 : Préparer la Lambda

1. Aller dans AWS Lambda → Test1
2. Vérifier le code :

```
def lambda_handler(event, context):
    return {
        "statusCode": 200,
        "body": "Hello from Lambda!"
    }
```

3. Déployer la Lambda.

Cette Lambda fonctionne toute seule si on la teste.



The screenshot shows the AWS Lambda console interface. At the top, there's a breadcrumb navigation: Lambda > Functions > Test1. Below this, a green success message box contains two items: "Successfully updated the function Test1." and "The test event "event1" was successfully saved." The main area is titled "Code source" with an "Info" link. On the left is an "EXPLORER" sidebar showing a folder structure for "TEST1" containing "lambda_function.py". The "TEST1" folder is expanded. On the right is a code editor window titled "lambda_function.py" with the following Python code:

```
import json

def lambda_handler(event, context):
    # TODO implement
    return {
        'statusCode': 200,
        'body': "Hello from Lambda!"
    }
```

At the bottom of the interface, there are two buttons: "Deploy (Ctrl+Shift+U)" and "Test (Ctrl+Shift+I)".

Étape 2 : Préparer le YAML correct

Voici un YAML minimal **clé en main**, prêt à être importé :

```
openapi: 3.0.1
info:
  title: Simple HTTP API Lab
  version: "1.0"

paths:
  /hello:
    get:
      x-amazon-apigateway-integration:
        type: aws_proxy
        httpMethod: POST
        payloadFormatVersion: "2.0"
        uri: arn:aws:apigateway:ca-central-1:lambda:path/2015-03-
31/functions/arn:aws:lambda:ca-central-
1:123456789:function:Test1/invocations
      responses:
        "200":
          description: OK

x-amazon-apigateway-importexport-version: "1.0"
```

⚠ Vérifie que l'ARN de la Lambda correspond exactement.

Remplacer :

ca-central-1 si besoin, par votre région de déploiement.
remplacer 123456789 par l'ID de votre compte AWS

Étape 3 : Importer le YAML dans API Gateway

1. Aller dans **API Gateway → HTTP APIs**
 2. Cliquer sur **Create API → Import from OpenAPI**
 3. Choisir ton fichier YAML et importer (ou copier-coller le code source .yaml)
 4. Create API
-

Étape 4 : Ajouter la permission Lambda pour API Gateway

Après l'import, la Lambda **n'a pas la permission** d'être invoquée.

Dans **CMD Windows** : il faut donc ajouter la permission.

```
aws lambda add-permission --function-name Test1 --statement-id apigateway-test-hello --action lambda:InvokeFunction --principal apigateway.amazonaws.com --source-arn arn:aws:execute-api:ca-central-1:123456789:1tvc33fz66/*/GET/hello
```

- Test1 → nom exact de ta Lambda
- 1tvc33fz66 → ID de votre API Gateway 1tvc33fz66
- Remplacer : **123456789** par l'ID de votre compte AWS
- * → toutes les étapes (ou remplacer par dev si on veux restreindre)

```
C:\Users\henri>aws lambda add-permission --function-name Test1 --statement-id apigateway-test-hello --action lambda:InvokeFunction --principal apigateway.amazonaws.com --source-arn arn:aws:execute-api:ca-central-1:██████████:1tvc33fz66/*/GET/hello
{
    "Statement": "{\"Sid\":\"apigateway-test-hello\",\"Effect\":\"Allow\",\"Principal\":{\"Service\":\"apigateway.amazonaws.com\"},\"Action\":\"lambda:InvokeFunction\",\"Resource\":\"arn:aws:lambda:ca-central-1:██████████:function:Test1\",
    \"Condition\":{\"ArnLike\":{\"AWS:SourceArn\":\"arn:aws:execute-api:ca-central-1:760606734887:1tvc33fz66/*/GET/hello\"}}
  }"
}

C:\Users\henri>
```

On peut aussi faire cette étape **dans la console Lambda → Permissions → Add Permission**.

Voir Annexe 1 , les détails de la police qui a été créée et associée à Lambda. Dans le cadre de cet exercice , la police a été créé via la commande CLI ci-dessus et elle a généré une police, pour la fonction Lambda que l'on retrouve au niveau de la console :

AWS – Lambda-Functions-Test1-Permissions- Resource-based policy statements

Étape 5 : Déployer le stage **dev**

1. Aller dans **API Gateway → Deployments**
2. Crée un **nouveau déploiement** → stage **dev**
3. Déployer

Sans cette étape, l'API n'utilisera pas le YAML importé.

Étape 6 : Vérifier l'intégration avec Lambda

1. La route qui fait un GET doit pouvoir appeler la fonction Lambda
2. Vérifier que l'intégration de la fonction Lambda est effectivement attachée à la route GET.

The screenshot shows the AWS API Gateway console. In the left sidebar under 'Develop', 'Integrations' is selected. The main area is titled 'Integrations' with tabs for 'Attach integrations to routes' and 'Manage integrations'. A green success message at the top says 'Successfully deleted API "Simple HTTP API Lab".' Below it, there's a table for 'Routes for Simple HTTP API Lab'. One row shows a route for 'GET /hello' with an 'AWS Lambda' integration. The 'Integration details for route' section on the right shows 'Lambda function Test1 (ca-central-1)'. A button for 'Manage integration' is visible. The URL in the browser bar is 'https://1tvc33fz66.execute-api.ca-central-1.amazonaws.com/dev/hello'.

Étape 7 : Tester le endpoint

<https://1tvc33fz66.execute-api.ca-central-1.amazonaws.com/dev/hello>

- Si tout est correct, on verra :

```
{  
  "statusCode": 200,  
  "body": "Hello from Lambda!"  
}
```

The screenshot shows a web browser window with the URL 'https://1tvc33fz66.execute-api.ca-central-1.amazonaws.com/dev/hello'. The page content is a single line of text: 'Hello from Lambda!', which is highlighted with a red box. The browser's address bar and various tabs are visible at the top.

8 Résumé

- **Lambda** : code correct et testé
- **YAML** : importé correctement avec type `aws_proxy`
- **Permission Lambda** : ajoutée pour API Gateway
- **Stage** : déployé
- **Test** : `/dev/hello` fonctionne

9 Divers écrans :

The screenshot shows the AWS API Gateway console with the following details:

API Gateway > APIs > Routes - Simple HTTP API Lab (1tvc33fz66)

Routes section:

- Routes for Simple HTTP API Lab**: Shows a single route `/hello` with a `GET` method.
- Route details**: Shows ARN `arn:aws:apigateway:ca-central-1::apis/1tvc33fz66/routes/dz8lq4g`, Authorization status (No authorizer attached), and a `Deploy` button.

Stages section:

- Stages for Simple HTTP API Lab**: Shows a single stage `dev`.
- Stage details**: Shows Name `dev`, Created `January 20, 2026 8:47 PM`, Last updated `January 20, 2026 8:48 PM`, and Invoke URL `https://1tvc33fz66.execute-api.ca-central-1.amazonaws.com/dev`.

Both sections include a green success message: `Successfully created deployment for Simple HTTP API Lab. This deployment is active for dev.`

Successfully created deployment for Simple HTTP API Lab. This deployment is active for dev.

Name	Description	ID	Protocol	API endpoint type	Created	Security policy	API status
MySimpleLambda		8dbox6jzr6	HTTP	Regional	2026-01-20	-	-
Simple HTTP API Lab		1tvc33fz66	HTTP	Regional	2026-01-21	-	-

10 Annexe 1 :

Lambda »Function »Test1 :

Resource-based policy statements (1) [Info](#)

Statement ID	Principal	PrincipalOrgID	Conditions	Action
apigateway-test-hello	apigateway.amazonaws.com	-	ArnLike	lambda:InvokeFunction

Quand on clique sur : [apigateway-test-info](#) , l'écran ci-dessous s'affiche.

Il contient la police en format .Json

Dans lequel : 123456789 devrait correspondre à l'ID de votre compte AWS

Qui permet dans le contexte de votre compte que Api Gateway puisse invoquer la fonction Lambda qui a pour ID : 1tvc33fz66

```
{
  "ArnLike": {
```

```
        "AWS:SourceArn": "arn:aws:execute-api:ca-central-1:123456789:1tvc33fz66/*/GET/hello"
    }
}
```



11 Références

<https://docs.aws.amazon.com/apigateway/latest/developerguide/http-api.html>