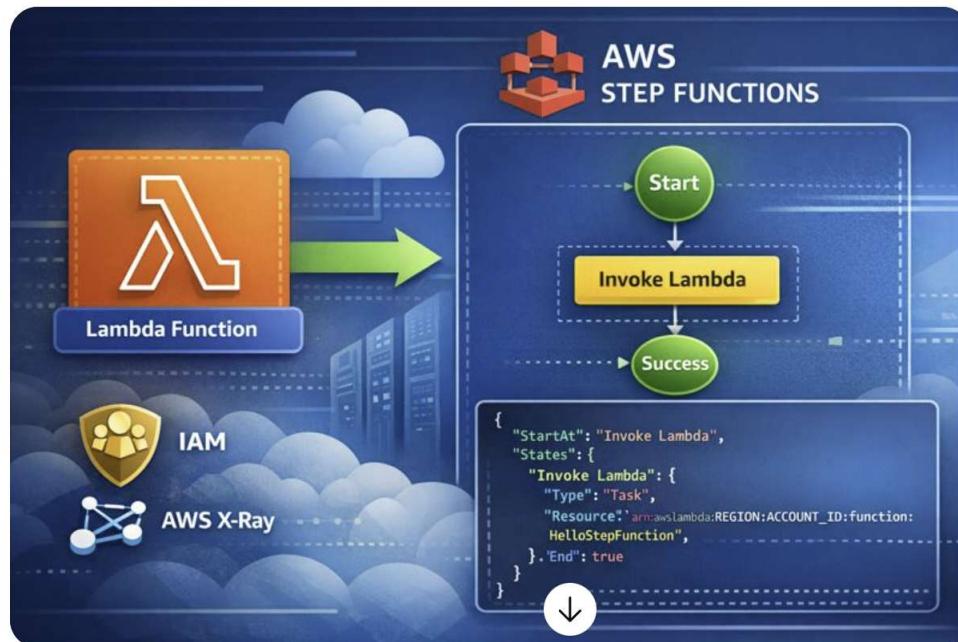


AWS Hands-on Lab | Lambda (Python) & Step Functions



Réalisation d'un lab très simple pour comprendre le fonctionnement des **AWS Step Functions** et leur intégration avec **AWS Lambda**.

Objectif :

- Créer une fonction Lambda minimale, codée en Python
- Orchestrer son exécution avec une **State Machine (Standard)**
- Comprendre le workflow visuel, le code JSON généré automatiquement et les rôles IAM associés
- Tester l'exécution et analyser le résultat

Services utilisés :

- AWS Lambda (Python 3.11)
- AWS Step Functions (Standard)
- IAM
- AWS X-Ray (permissions)

Sommaire :

1 : Créer une fonction Lambda simple	2
2 : Créer un “state machine” (Step function)	2

3 Créer la Step Function.....	4
4 : Tester la Step Function.....	6

1 : Crée une fonction Lambda simple

Dans la console AWS, Aller sur AWS Lambda → Create Function → Author from scratch.

Nom : HelloStepFunction.

Runtime : Python 3.11 (ou Node.js si tu préfères).

Dans le code Lambda, mettons quelque chose de minimal :

```
def lambda_handler(event, context):
    return {
        "statusCode": 200,
        "message": "Hello from Step Function!"
    }
```

Déployer la fonction.

The screenshot shows the AWS Lambda console interface. In the top navigation bar, 'Lambda' is selected under 'Functions'. Below the navigation, the text 'Functions (2)' is displayed, followed by a search bar with placeholder text 'Search by attributes or search by keyword'. A table lists two functions:

Function name	Description	Package type	Runtime
Hellostepfunction	-	Zip	Python 3.14

At the top right of the table, it says 'Last fetched 1/16/2026, 8:1'.

2 : Crée un “state machine” (Step function)

Dans la console AWS, Aller sur Step Functions → Create state machine → Standard et “create from blank”

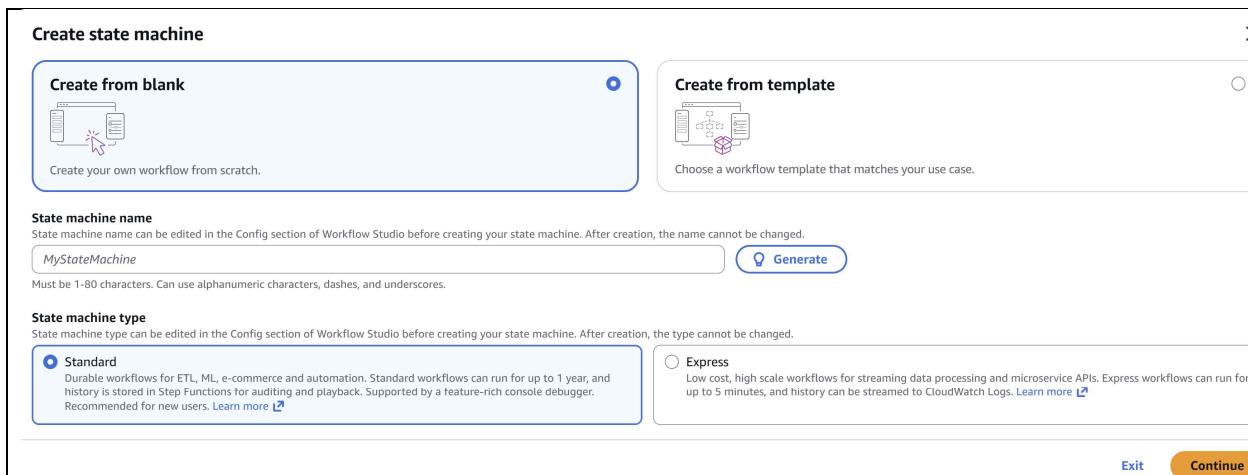
Nom : SimpleLab.

Choisir : Design your workflow visually.

The screenshot shows the AWS Step Functions console interface. In the top navigation bar, 'Step Functions' is selected under 'State machines'. Below the navigation, the text 'State machines (2)' is displayed, followed by a search bar with placeholder text 'Search for state machines'. A table lists two state machines:

Name	Type	Creation date	Status
SimpleLab	Standard	Jan 16, 2026, 20:51:48 (UTC-05:00)	Active
MyState	Standard	Jan 16, 2026, 20:43:08 (UTC-05:00)	Active

At the top right of the table, there are buttons for 'View execution counts', 'View details', 'Edit', 'Copy to new', 'Delete', and 'Create state machine'.

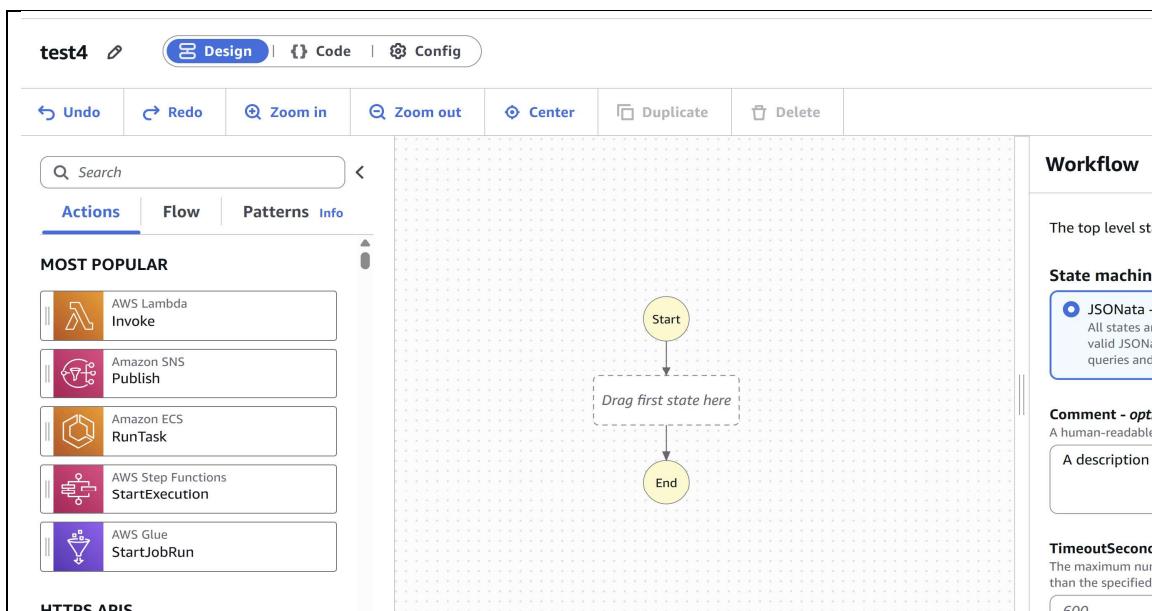


Cliquer sur "continuer"

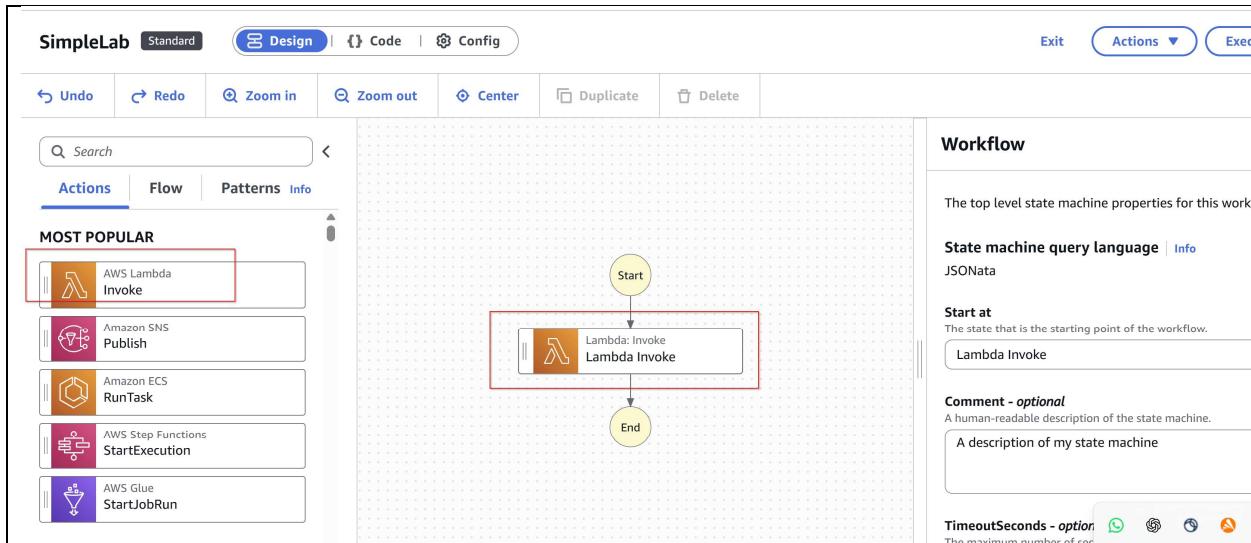
Ajouter un état Task :

Resource : votre fonction Lambda HelloStepFunction.

Pour cela faire glisser la fonction du pannel de gauche, intitulée : "AWS Lambda invoke" vers le graphique dans le champs : "drag first state here". Le "first state" correspond au premier "état" qui est : "Lambda Invoke".



Cela remplis automatiquement le panel, intitulé "workflow sur la droite" avec les données de la fonction lambda. Voir le graphique ci-dessous.



Exemple de définition JSON très simple :

```
{
  "Comment": "A simple Step Function",
  "StartAt": "HelloTask",
  "States": {
    "HelloTask": {
      "Type": "Task",
      "Resource": "arn:aws:lambda:REGION:ACCOUNT_ID:function:HelloStepFunction",
      "End": true
    }
  }
}
```

On n'utilisera pas ce code Json , mais on utilisera a la place le code Json , décrit au paragraphe suivant.

3 Créer la Step Function.

L'outil AWS Step function workflow , génère du code Json automatiquement, que l'on peut voir dans la fenêtre “code”. Le code Json appelle (invoke) la fonction lambda créée précédemment.

SimpleLab

Design | Code | Config

Code View:

```

1 {
2   "Comment": "A description of my state machine",
3   "StartAt": "Lambda Invoke",
4   "States": {
5     "Lambda Invoke": {
6       "Type": "Task",
7       "Resource": "arn:aws:states:::lambda:invoke",
8       "Output": "{$states.result.Payload}",
9       "Arguments": {
10         "FunctionName": "arn:aws:lambda:ca-central-1:ACCOUNT_ID:function:Hellostepfunction:$LATEST",
11         "Payload": "{$states.input}"
12       },
13       "Retry": [
14         {
15           "ErrorEquals": [
16             "Lambda.ServiceException",
17             "Lambda.AWSLambdaException",
18             "Lambda.SdkClientException",
19             "Lambda.TooManyRequestsException"
20           ],
21           "IntervalSeconds": 1,
22           "MaxAttempts": 3,
23           "BackoffRate": 2,
24           "JitterStrategy": "FULL"
25         }
26       ],
27       "End": true
28     }
29   }
30 }
```

Diagram View:

```

graph TD
    Start((Start)) --> LambdaInvoke[Lambda: Invoke Lambda Invoke]
    LambdaInvoke --> End((End))

```

Dans l'URL :

"arn:aws:lambda:ca-central-1: ACCOUNT_ID:function:Hellostepfunction:\$LATEST"

A la place de **ACCOUNT_ID** , on aura l'ID de votre compte AWS

Cliquer sur le bouton “create”

Lorsqu'on clique sur “create” , une autre fenêtre s’Affiche qui demande confirmation pour ajouter des rôles et permissions pour les rôles, afin que AWS Step Function puisse exécuter la fonction Lambda et aussi avoir le droit d’utiliser le service AWS X-Ray pur le debugging.

Confirm role creation

An execution role will be created with full permissions.
A new execution role named StepFunctions-SimpleLab-role-ch8utigwg will be created. All required permissions for the actions specified in your state machine will be auto-generated.

Role permissions

Service	Action(s)	Status	Documentation links
AWS Lambda	lambda:Invoke	Policy will be generated to perform the action for specified Lambda resources only	Call Lambda with Step Functions Lambda policies for Step Functions
AWS X-Ray	xray:PutTraceSegments xray:PutTelemetryRecords xray:GetSamplingRules xray:GetSamplingTargets	Policies will be generated for X-Ray tracing	X-Ray policies for Step Functions

Cancel [View role configuration](#) Confirm

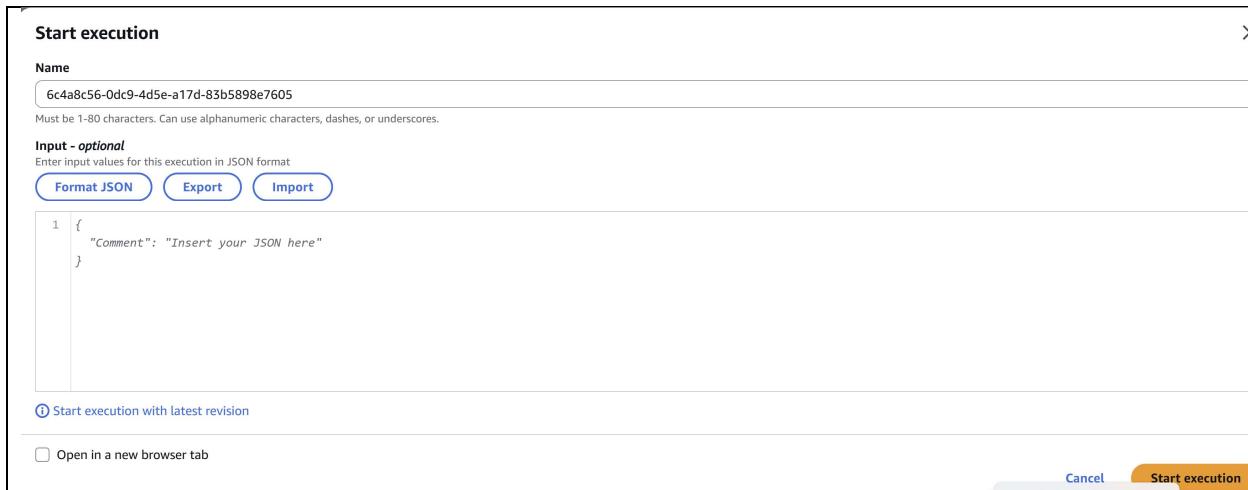
Cliquer sur “confirmation ”

On peut aussi cliquer sur le bouton “view rôle configuration” afin de voir le détail des rôles.

4 : Tester la Step Function

Cliquer sur Start execution.

La fenêtre suivante s’Affiche :



Pour ce premier test , laisser entrée : {} (vide).

Résultat attendu :

```
{  
  "statusCode": 200,  
  "message": "Hello from Step Function!"}
```

The screenshot shows the 'Execution details' page for execution ID '6c4a8c56-0dc9-4d5e-a17d-83b5898e7605'. The 'Execution input and output' tab is selected. On the left, under 'State input', there is a text box containing '{}'. On the right, under 'State output', there is a text box containing the following JSON:

```
1 | {  
2 |     "statusCode": 200,  
3 |     "message": "Hello from Step Function!"  
4 | }
```

At the bottom of the page are 'Graph view' and 'Table view' buttons.

Vérifier l’état des boutons sur le graphique :

Le status “vert” indique le succès de l’opération

