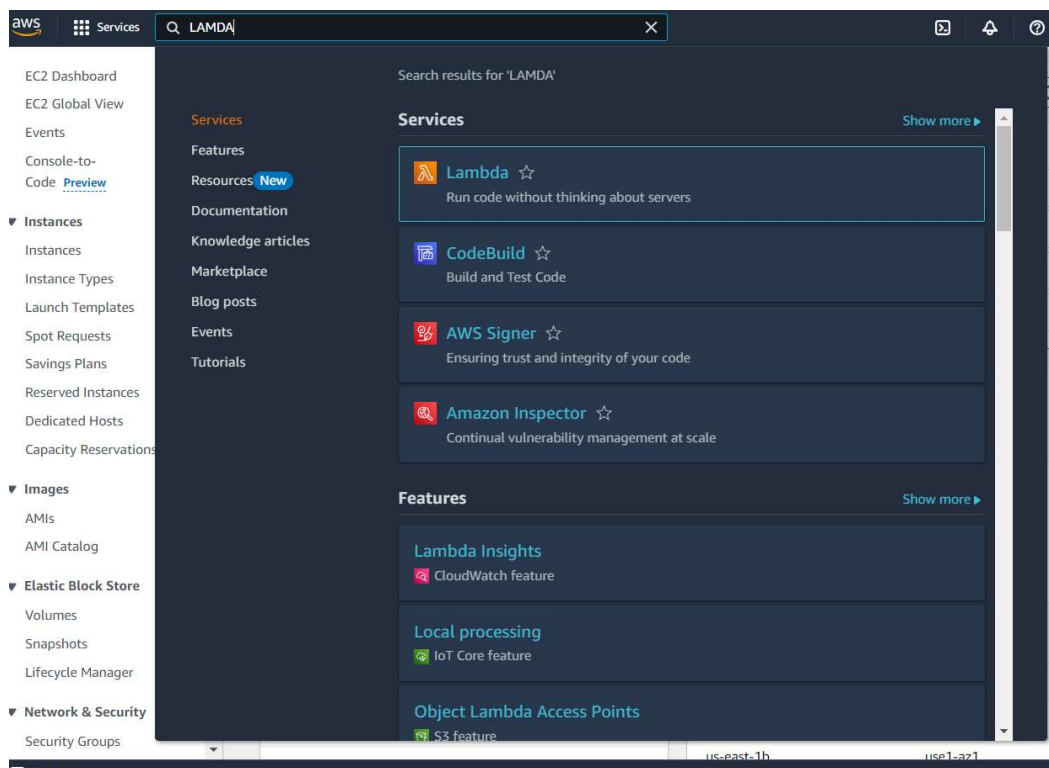


Aim: To understand AWS Lambda, its workflow, various functions and create your first Lambda functions using Python / Java / Nodejs.

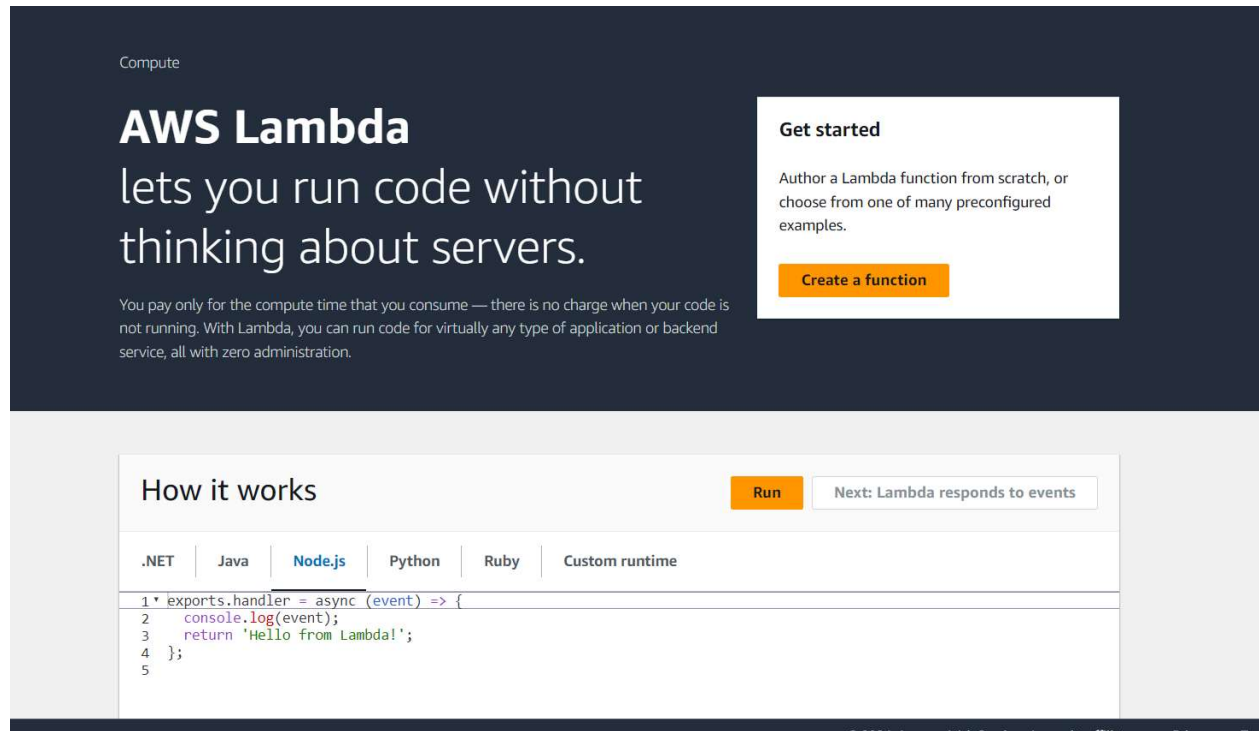
Step 1: Accessing AWS

Log in to your AWS Personal/Academy account. Navigate to the Lambda service by searching for "Lambda" in the AWS Management Console.



Step 2: Creating a New Lambda Function

Click on the "Create function" button. Provide a name for your Lambda function and select the language you wish to use, such as Python 3.12. For architecture, choose x86, and for execution role, opt to create a new role with basic Lambda g permissions.



Step 3: Configuring Basic Settings

To modify the basic settings, navigate to the "Configuration" tab and click on "Edit" under General Settings. Here, you can add a description and adjust the memory and timeout settings. For this experiment, I set the timeout to 1 second, which is sufficient for testing.

Basic information

Function name
Enter a name that describes the purpose of your function.

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime [Info](#)
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Python 3.12

▼

↺

Architecture [Info](#)
Choose the instruction set architecture you want for your function code.

☒ x86_64

☐ arm64


Permissions [Info](#)
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

[► Change default execution role](#)

By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

Execution role

- Create a new role with basic Lambda permissions

-  Role creation might take a few minutes. Please do not delete the role or edit the trust or permissions policies in this role.

► Additional Configurations

Cancel

Create function

aws

Services

Search

[Alt+S]

Successfully created the function **lamda_demo**. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

InfoTutorials

Lambda > Functions > lamda_demo

lamda_demo

ThrottleCopy ARNActions

Function overviewInfo

DiagramTemplate

lamda_demo

Layers (0)

+ Add trigger

+ Add destination

Description

-

Last modified

26 seconds ago

Function ARN

arn:aws:lambda:eu-north-1:010928207735:function:lamda_demo

Function URLInfo

-

Code

Test

Monitor

Configuration

Aliases

Versions

Create a simple web app

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

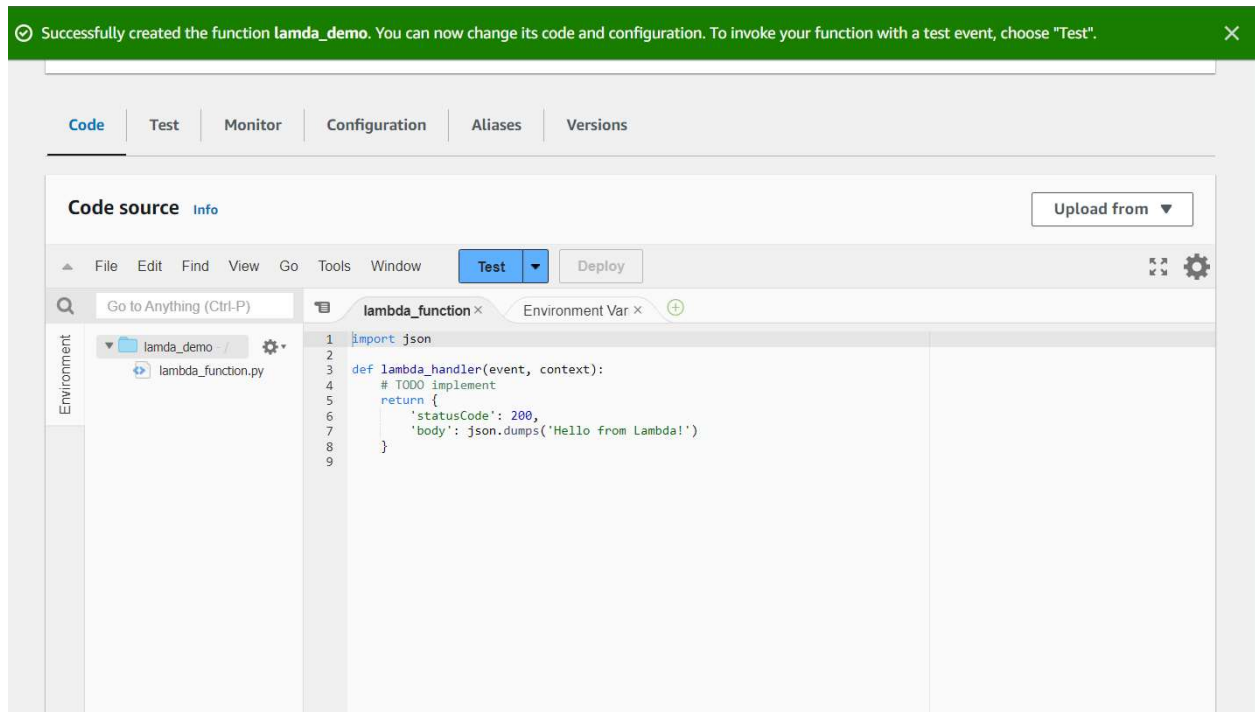
[Learn more](#)

Start tutorial

CloudShell

Feedback

© 2024, Amazon Web Services, Inc. or its affiliates. PrivacyTermsCookie preferences



Step 4: Testing the Function

Click on the "Test" tab and select "Create a new event." Name your event, set the event sharing to private, and choose the "hello-world" template.

Test event Info

Save Test

To invoke your function without saving an event, configure the JSON event, then choose Test.

Test event action

☒ Create new event ☐ Edit saved event

Event name

MyEventName

Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.

Event sharing settings

☒ Private
This event is only available in the Lambda console and to the event creator. You can configure a total of 10. [Learn more](#)

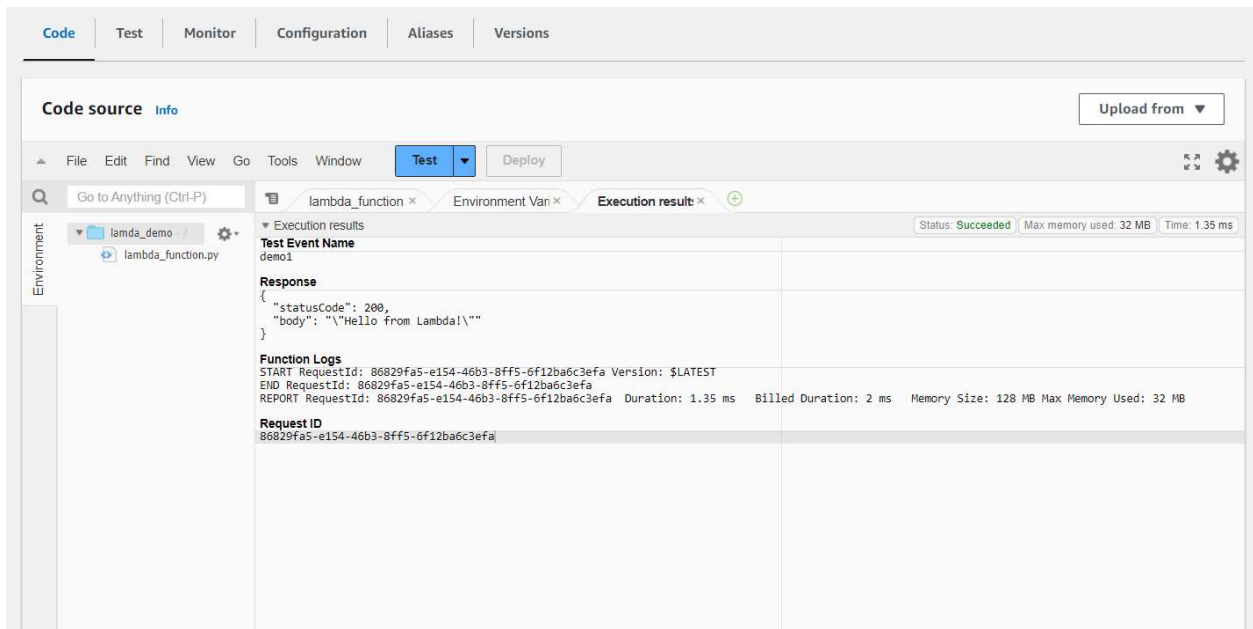
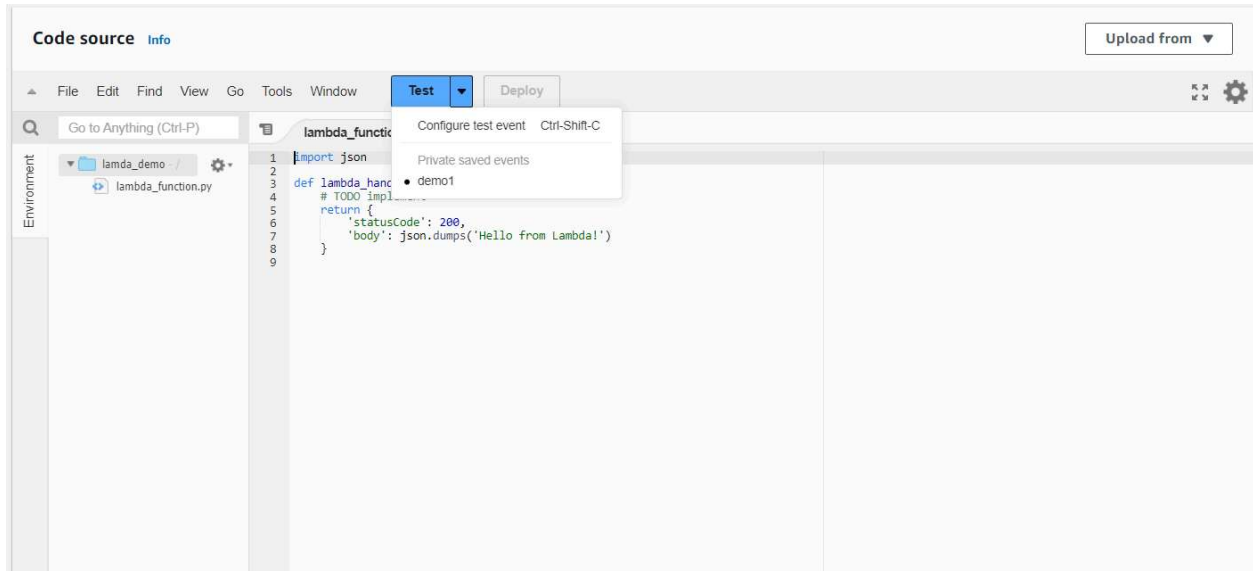
☐ Shareable
This event is available to IAM users within the same account who have permissions to access and use shareable events. [Learn more](#)

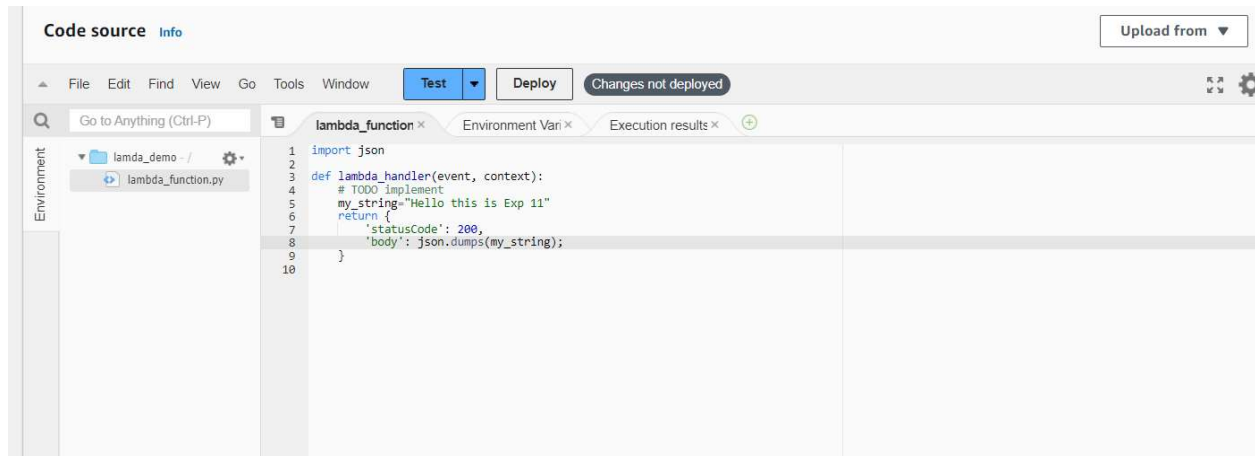
Template - optional

hello-world

Event JSON Format JSON

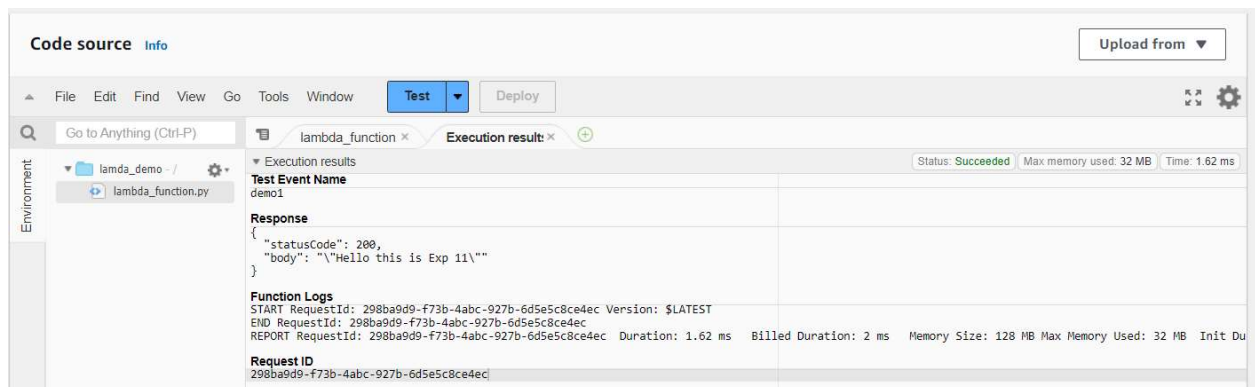
```
1 {
2   "key1": "value1",
3   "key2": "value2",
4   "key3": "value3"
5 }
```





Step 5: Running the Test

In the Code section, select the newly created event from the dropdown menu and click on "Test." You should see the output displayed below.



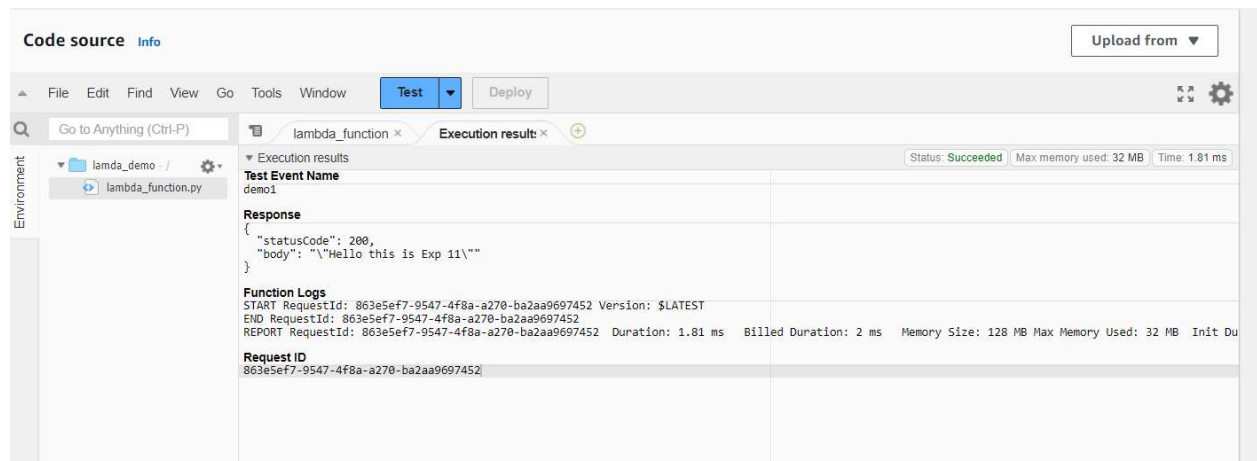
Step 6: Editing and Deploying the Code

You can modify your Lambda function's code as needed. I updated the code to display a new string. After making changes, press `Ctrl + S` to save and then click on "Deploy" to apply the updates.



Step 7: Final Testing

Return to the "Test" tab and execute the test again to observe the output. You should see a status code of 200 along with your string output and function logs confirming a successful deployment.



Conclusion: In this experiment, we successfully created an AWS Lambda function and followed the important steps involved. First, we set up the function using Python and adjusted the timeout setting to 1 second. Then, we created a test event to see how the function works and checked the output to ensure it was correct. We also modified the function's code and redeployed it to see the changes in real-time. So Lambda Function allows you to concentrate on writing code while AWS manages the infrastructure and automatically scales the service as needed. This makes it easier to develop and run applications without worrying about server management.