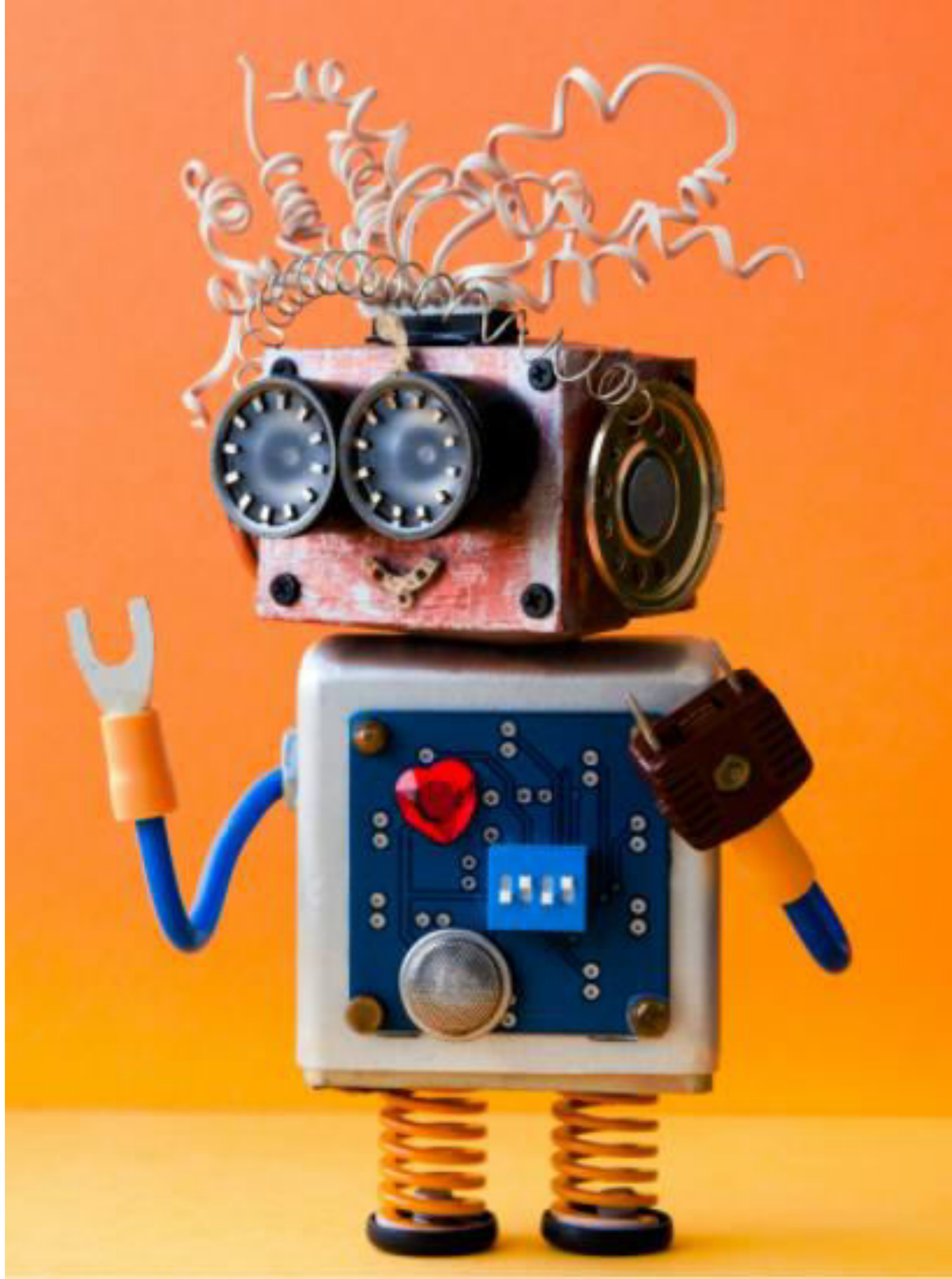
INTRODUCTION

EASY ▲ ADVANCED



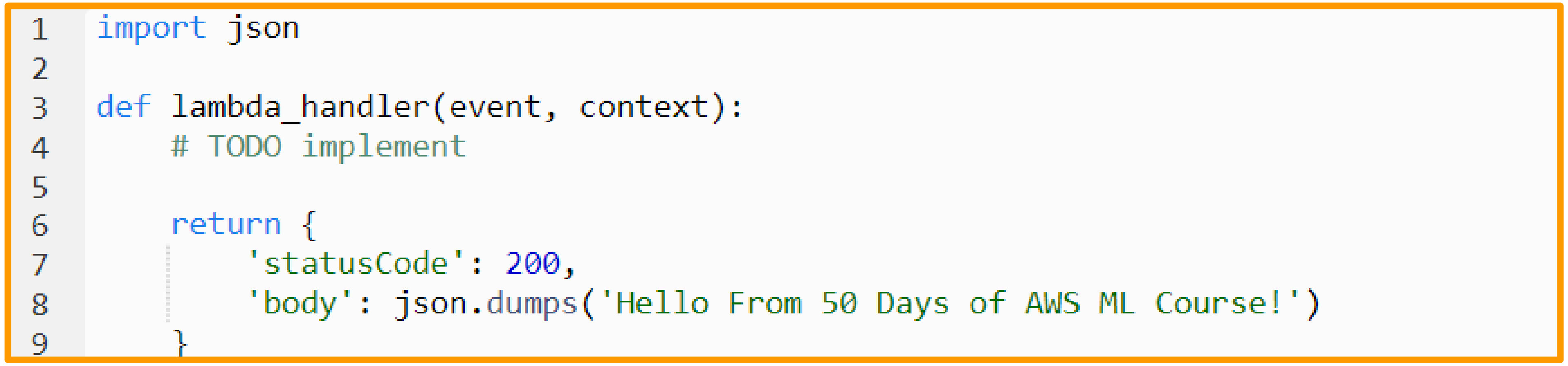
INTRODUCTION

- In this project, we will learn how to define and invoke lambda functions using AWS Boto3 SDK.
- Lambda is the most popular and used service in AWS.
- AWS lambda free developers from the worry of provisioning resources, specifying operating systems, managing Hardware, and performing maintenance.
- Simply write your code and run it on Lambda!

  1. Define a Lambda function using Boto3 SDK.
  2. Test the lambda function using Eventbridge (cloudwatch events).
  3. Understand the difference between synchronous and asynchronous invocations.
  4. Invoke a Lambda function using Boto3 SDK.

AWS LAMBDA FUNCTION ANATOMY (RECAP)

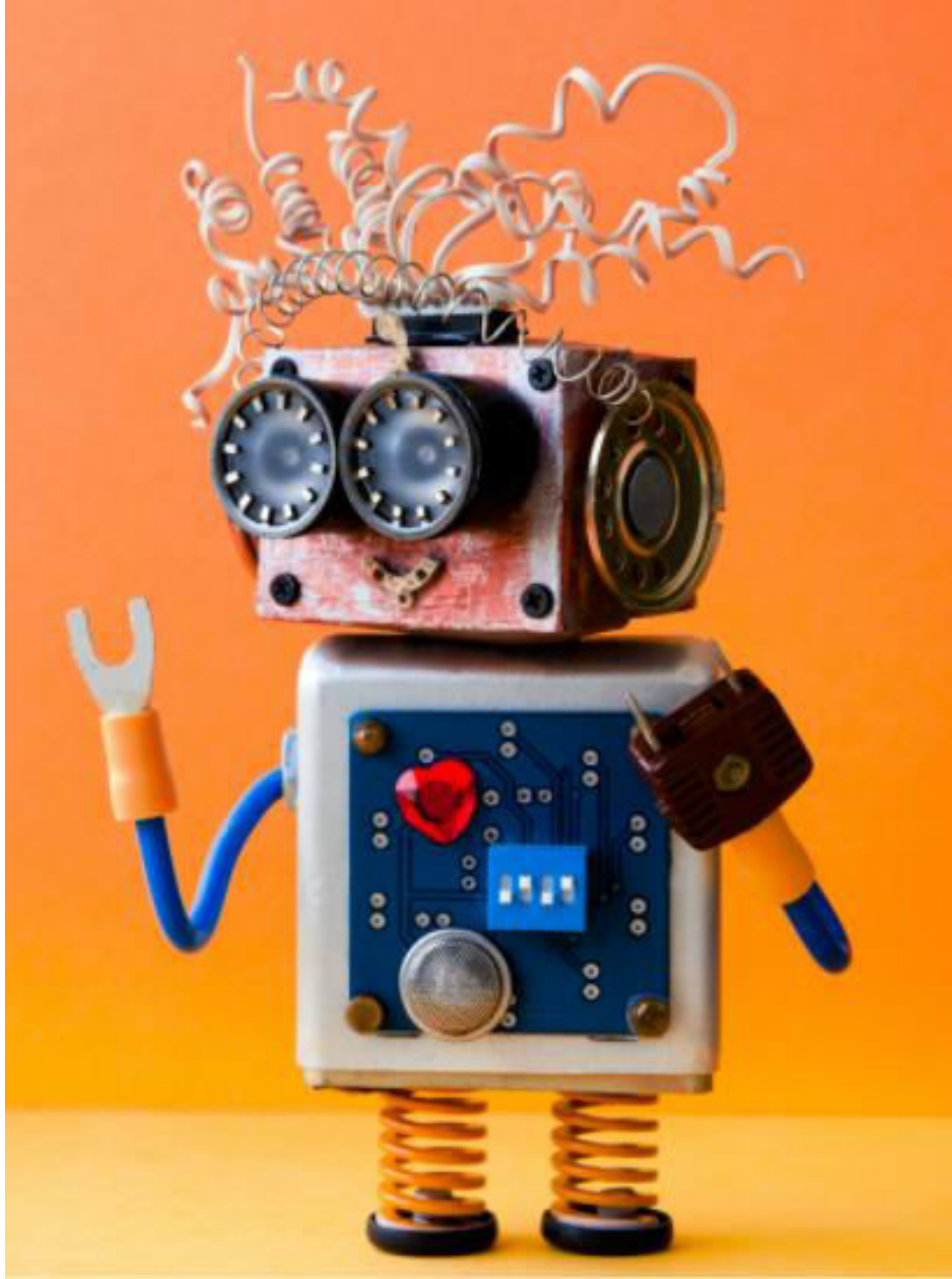
- **Handler() Function:** Function to be executed upon invocation and it requires two arguments “event” and “context”.
- **Event Object:** data sent during lambda function invocation, for example if a request is made from S3, the event object will contain the bucket key and what kind of action has been performed on the bucket.
- **Context object:** this is generated by the platform and contains information about the underlying infrastructure and execution environment such as allowed runtime and memory.



```python
import json

def lambda_handler(event, context):
    # TODO implement

    return {
        'statusCode': 200,
        'body': json.dumps('Hello From 50 Days of AWS ML Course!')
    }
```

SYNCHRONOUS Vs. ASYNCHRONOUS INCOVATIONS

EASY ADVANCED



SYNCHRONOUS Vs. ASYNCHRONOUS

- Lambda functions could be invoked (called) using the Console, AWS SDK, Lambda API and AWS Command Line Interface (AWS CLI).
- There are generally two ways of invoking a lambda function: **Synchronously** and **Asynchronously**.
- Lambda functions invocation documentation: https://docs.aws.amazon.com/lambda/latest/dg/lambda-invocation.html

Synchronous Invocation

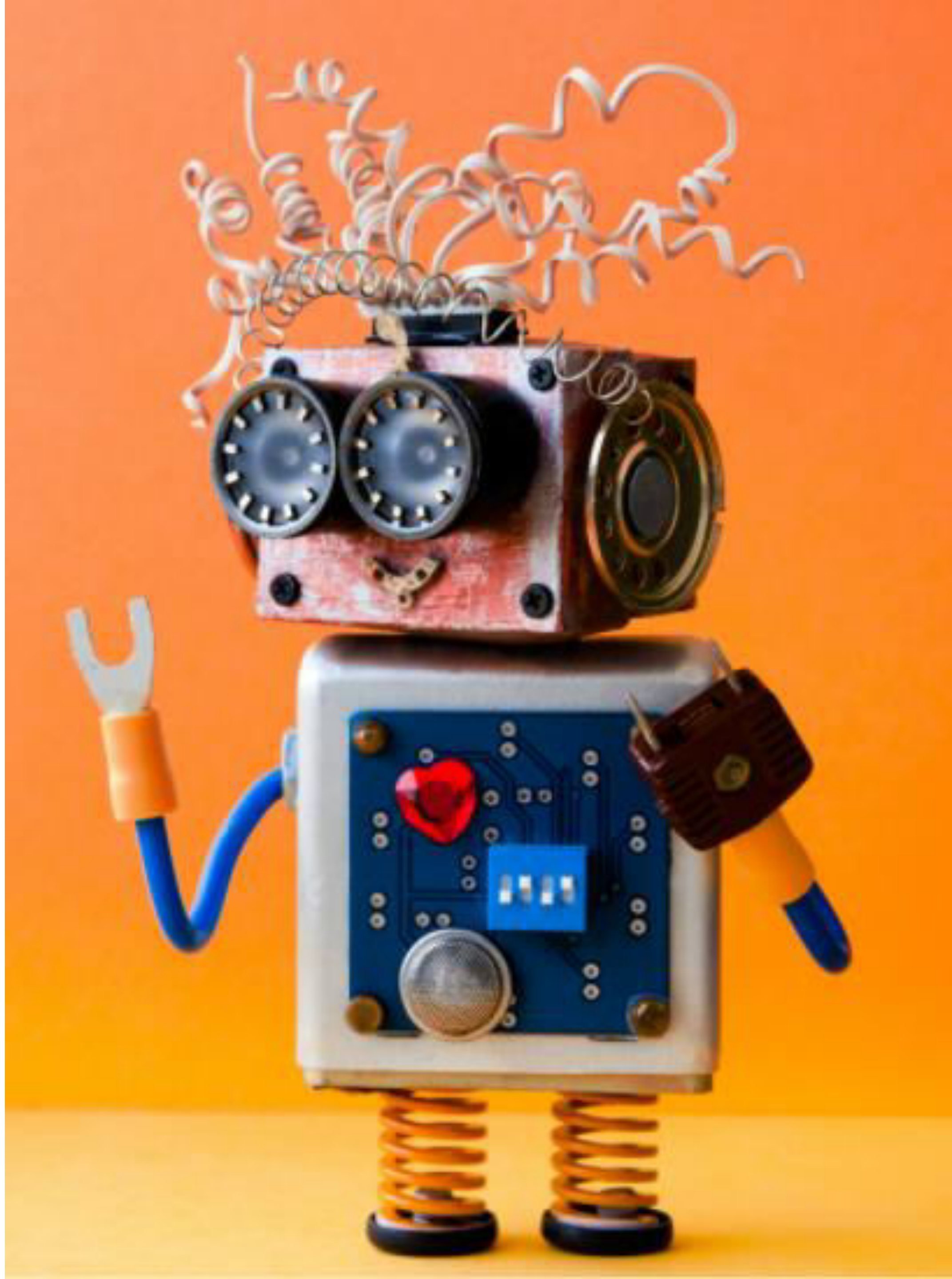
- With synchronous invocation, you wait for the function to process the event and return a response.
- Synchronous invocations are best suited for Machine Learning workflows.

Asynchronous Invocation

- With asynchronous invocation, Lambda queues the event for processing, so you don't have to wait for a response from Lambda.
- For asynchronous invocation, Lambda handles retries and can send invocation records to a destination.

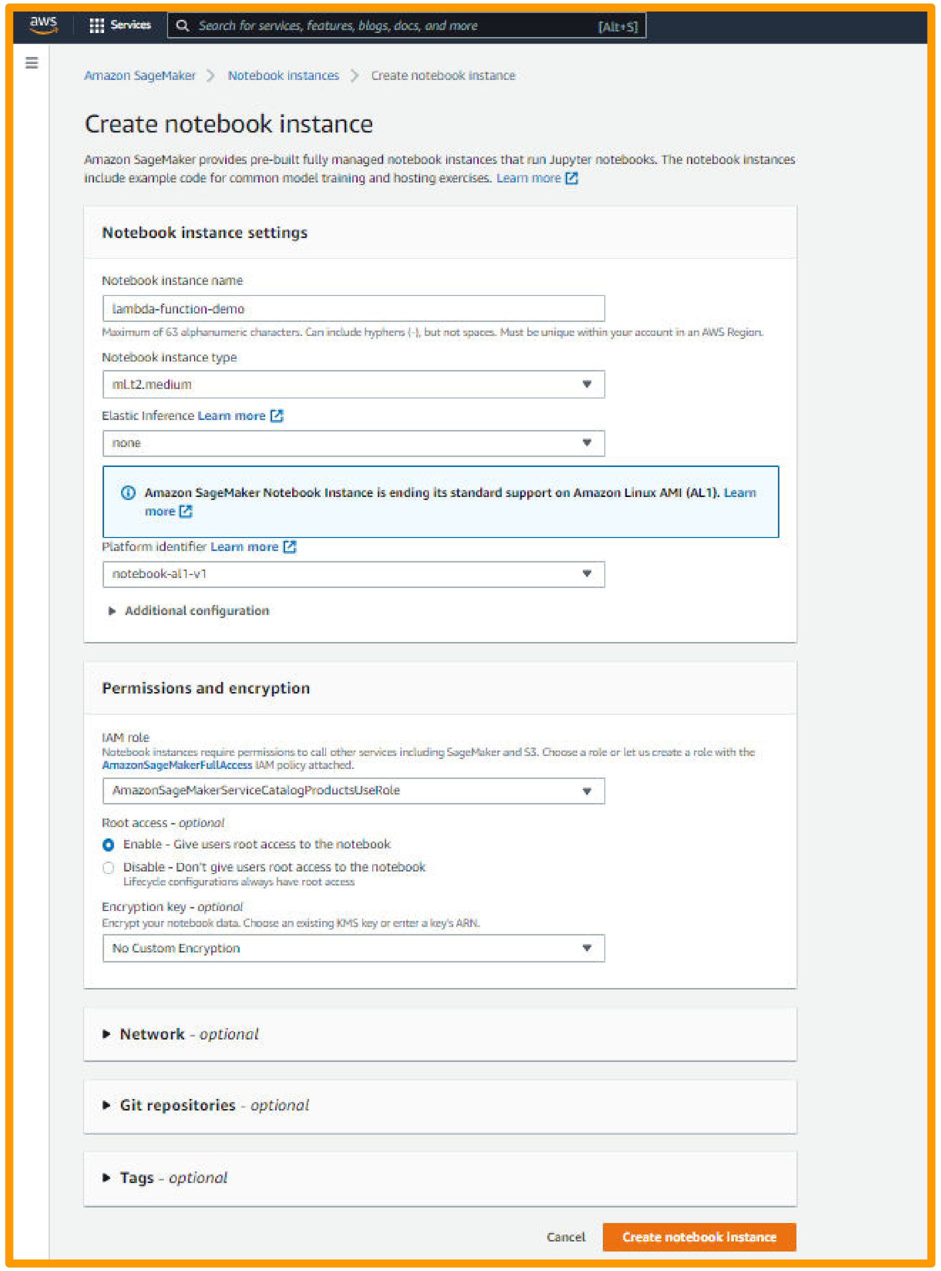
DEMO: DEFINE AN AWS LAMBDA FUNCTION USING BOTO3 SDK

EASY ADVANCED



DEMO: CREATE AN AWS LAMBDA FUNCTION USING BOTO3

CREATE A NOTEBOOK INSTANCE



Amazon SageMaker > Notebook instances > Create notebook instance

Create notebook instance

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. Learn more

Notebook instance settings

Notebook instance name: lambda-function-demo

Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Notebook instance type: ml.t2.medium

Elastic Inference Learn more: none

Amazon SageMaker Notebook Instance is ending its standard support on Amazon Linux AMI (AL1). Learn more

Platform identifier Learn more: notebook-al1-v1

▸ Additional configuration

Permissions and encryption

IAM role
Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the AmazonSageMakerFullAccess IAM policy attached.

AmazonSageMakerServiceCatalogProductsUseRole

Root access - *optional*

[✓ Enable - Give users root access to the notebook]
Disable - Don't give users root access to the notebook
Lifecycle configurations always have root access

Encryption key - *optional*
Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.

No Custom Encryption

▸ **Network** - *optional*

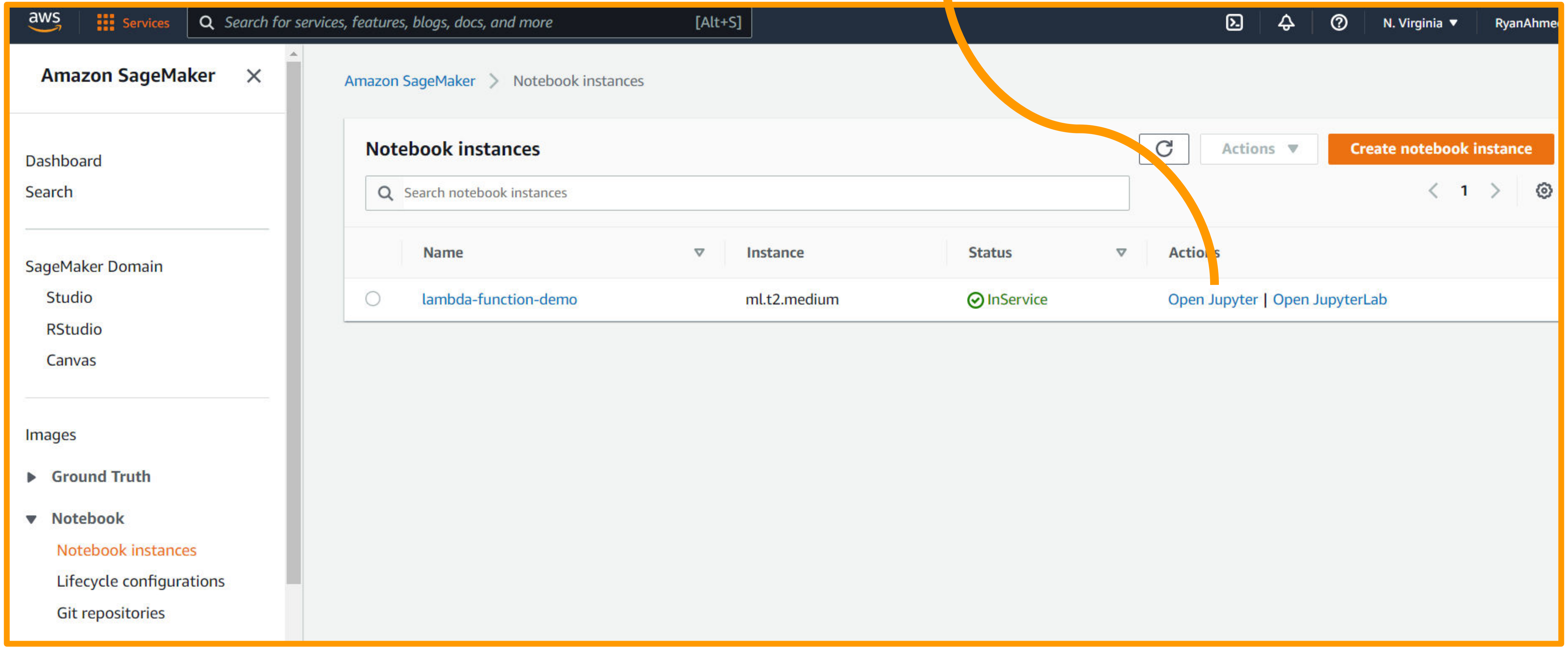
▸ **Git repositories** - *optional*

▸ **Tags** - *optional*

Cancel | Create notebook instance

DEMO: CREATE AN AWS LAMBDA FUNCTION USING BOTO3

CLICK ON OPEN JUPYTER

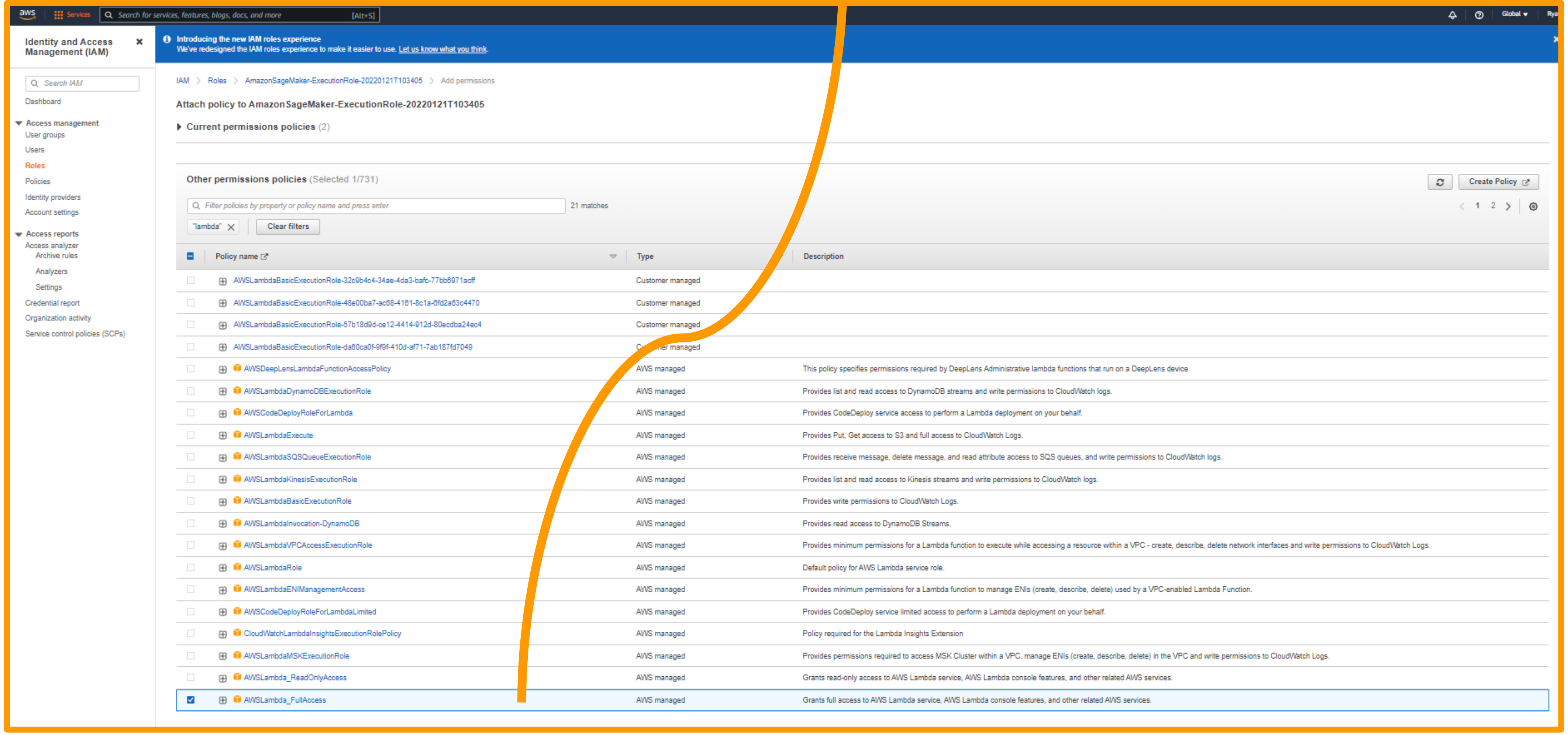


Notebook instances

Name	Instance	Status	Actions
lambda-function-demo	ml.t2.medium	InService	Open Jupyter | Open JupyterLab

DEMO: CREATE AN AWS LAMBDA FUNCTION USING BOTO3

GO TO THE SAGEMAKER EXECUTION ROLE AND ATTACH "AWSLAMBDA_FULLACCESS" POLICY



Introducing the new IAM roles experience
We've redesigned the IAM roles experience to make it easier to use. Let us know what you think.

IAM > Roles > AmazonSageMaker-ExecutionRole-20220121T103405 > Add permissions

Attach policy to AmazonSageMaker-ExecutionRole-20220121T103405

Current permissions policies (2)

Other permissions policies (Selected 1/731)

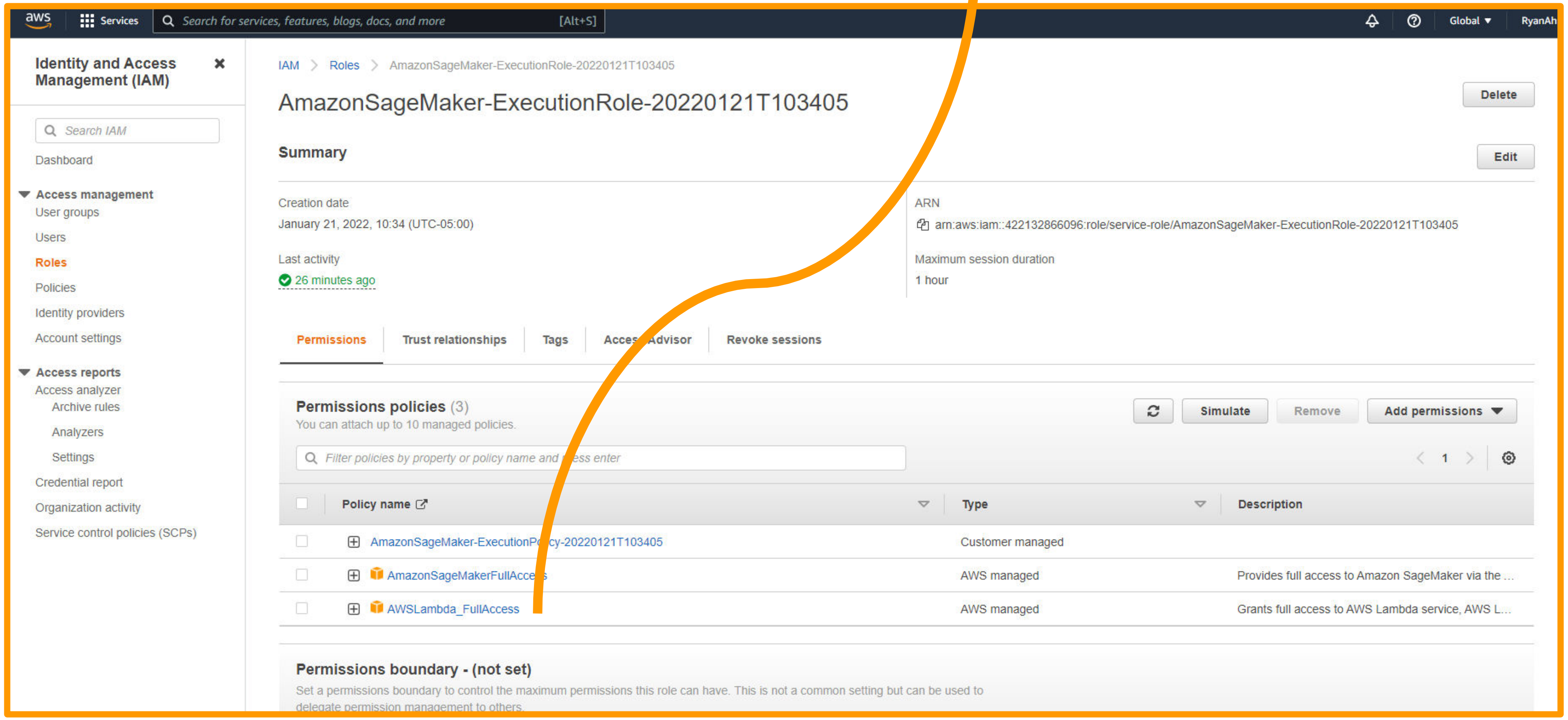
Filter policies by property or policy name and press enter — 21 matches

"lambda" — Clear filters

	Policy name	Type	Description
	AWSLambdaBasicExecutionRole-32c9b4c4-34ae-4da3-bafc-77bb6971acff	Customer managed	
	AWSLambdaBasicExecutionRole-48e00ba7-ac68-4161-8c1a-6fd2a63c4470	Customer managed	
	AWSLambdaBasicExecutionRole-57b18d9d-ce12-4414-912d-80ecdba24ec4	Customer managed	
	AWSLambdaBasicExecutionRole-da60ca0f-9f9f-410d-af71-7ab187fd7049	Customer managed	
	AWSDeepLensLambdaFunctionAccessPolicy	AWS managed	This policy specifies permissions required by DeepLens Administrative lambda functions that run on a DeepLens device
	AWSLambdaDynamoDBExecutionRole	AWS managed	Provides list and read access to DynamoDB streams and write permissions to CloudWatch logs.
	AWSCodeDeployRoleForLambda	AWS managed	Provides CodeDeploy service access to perform a Lambda deployment on your behalf.
	AWSLambdaExecute	AWS managed	Provides Put, Get access to S3 and full access to CloudWatch Logs.
	AWSLambdaSQSQueueExecutionRole	AWS managed	Provides receive message, delete message, and read attribute access to SQS queues, and write permissions to CloudWatch logs.
	AWSLambdaKinesisExecutionRole	AWS managed	Provides list and read access to Kinesis streams and write permissions to CloudWatch logs.
	AWSLambdaBasicExecutionRole	AWS managed	Provides write permissions to CloudWatch Logs.
	AWSLambdaInvocation-DynamoDB	AWS managed	Provides read access to DynamoDB Streams.
	AWSLambdaVPCAccessExecutionRole	AWS managed	Provides minimum permissions for a Lambda function to execute while accessing a resource within a VPC - create, describe, delete network interfaces and write permissions to CloudWatch Logs.
	AWSLambdaRole	AWS managed	Default policy for AWS Lambda service role.
	AWSLambdaENIManagementAccess	AWS managed	Provides minimum permissions for a Lambda function to manage ENIs (create, describe, delete) used by a VPC-enabled Lambda Function.
	AWSCodeDeployRoleForLambdaLimited	AWS managed	Provides CodeDeploy service limited access to perform a Lambda deployment on your behalf.
	CloudWatchLambdaInsightsExecutionRolePolicy	AWS managed	Policy required for the Lambda Insights Extension
	AWSLambdaMSKExecutionRole	AWS managed	Provides permissions required to access MSK Cluster within a VPC, manage ENIs (create, describe, delete) in the VPC and write permissions to CloudWatch Logs.
	AWSLambda_ReadOnlyAccess	AWS managed	Grants read-only access to AWS Lambda service, AWS Lambda console features, and other related AWS services.
✓	AWSLambda_FullAccess	AWS managed	Grants full access to AWS Lambda service, AWS Lambda console features, and other related AWS services.

DEMO: CREATE AN AWS LAMBDA FUNCTION USING BOTO3

NOW WE HAVE FULL ACCESS TO LAMBDA!



AmazonSageMaker-ExecutionRole-20220121T103405

Summary

Creation date
January 21, 2022, 10:34 (UTC-05:00)

Last activity
26 minutes ago

ARN
arn:aws:iam::422132866096:role/service-role/AmazonSageMaker-ExecutionRole-20220121T103405

Maximum session duration
1 hour

Permissions policies (3)
You can attach up to 10 managed policies.

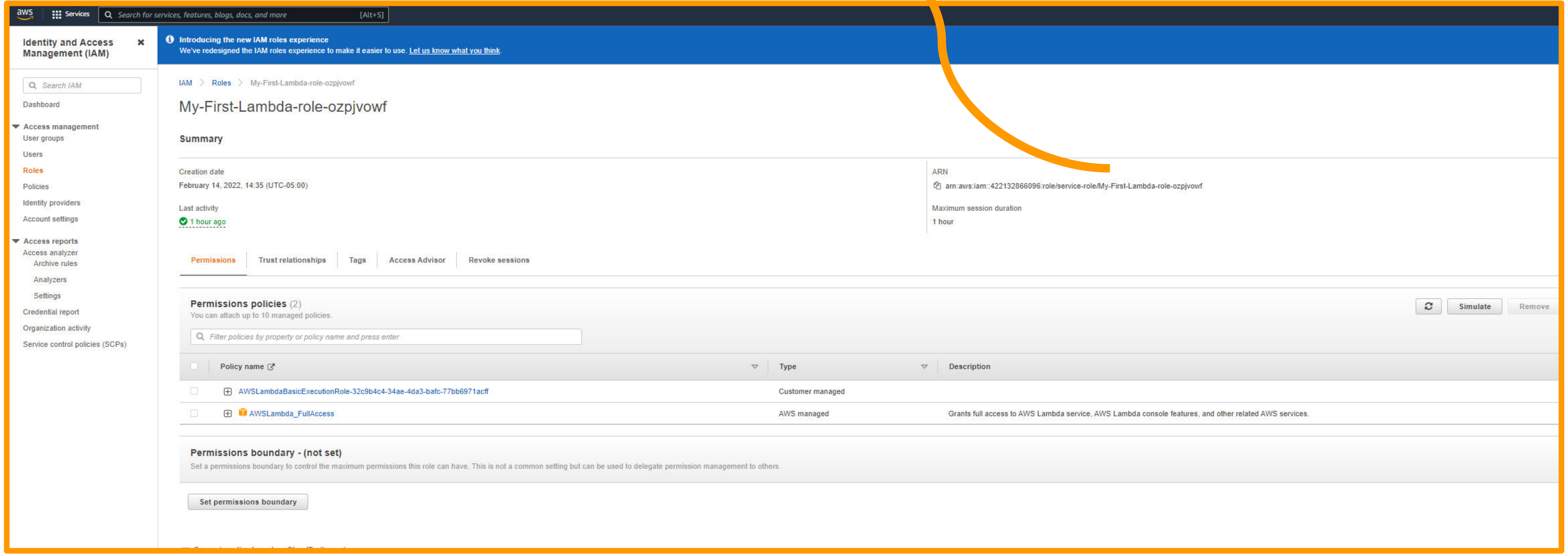
Filter policies by property or policy name and press enter

Policy name	Type	Description
AmazonSageMaker-ExecutionPolicy-20220121T103405	Customer managed	
AmazonSageMakerFullAccess	AWS managed	Provides full access to Amazon SageMaker via the ...
AWSLambda_FullAccess	AWS managed	Grants full access to AWS Lambda service, AWS L...

Permissions boundary - (not set)
Set a permissions boundary to control the maximum permissions this role can have. This is not a common setting but can be used to delegate permission management to others.

DEMO: CREATE AN AWS LAMBDA FUNCTION USING BOTO3

THIS IS THE ROLE ARN YOU WILL NEED IN CODE



Introducing the new IAM roles experience
We've redesigned the IAM roles experience to make it easier to use. Let us know what you think.

IAM > Roles > My-First-Lambda-role-ozpjvowf

My-First-Lambda-role-ozpjvowf

Summary

Creation date
February 14, 2022, 14:35 (UTC-05:00)

Last activity
1 hour ago

ARN
arn:aws:iam::422132866096:role/service-role/My-First-Lambda-role-ozpjvowf

Maximum session duration
1 hour

Permissions policies (2)
You can attach up to 10 managed policies.

Filter policies by property or policy name and press enter

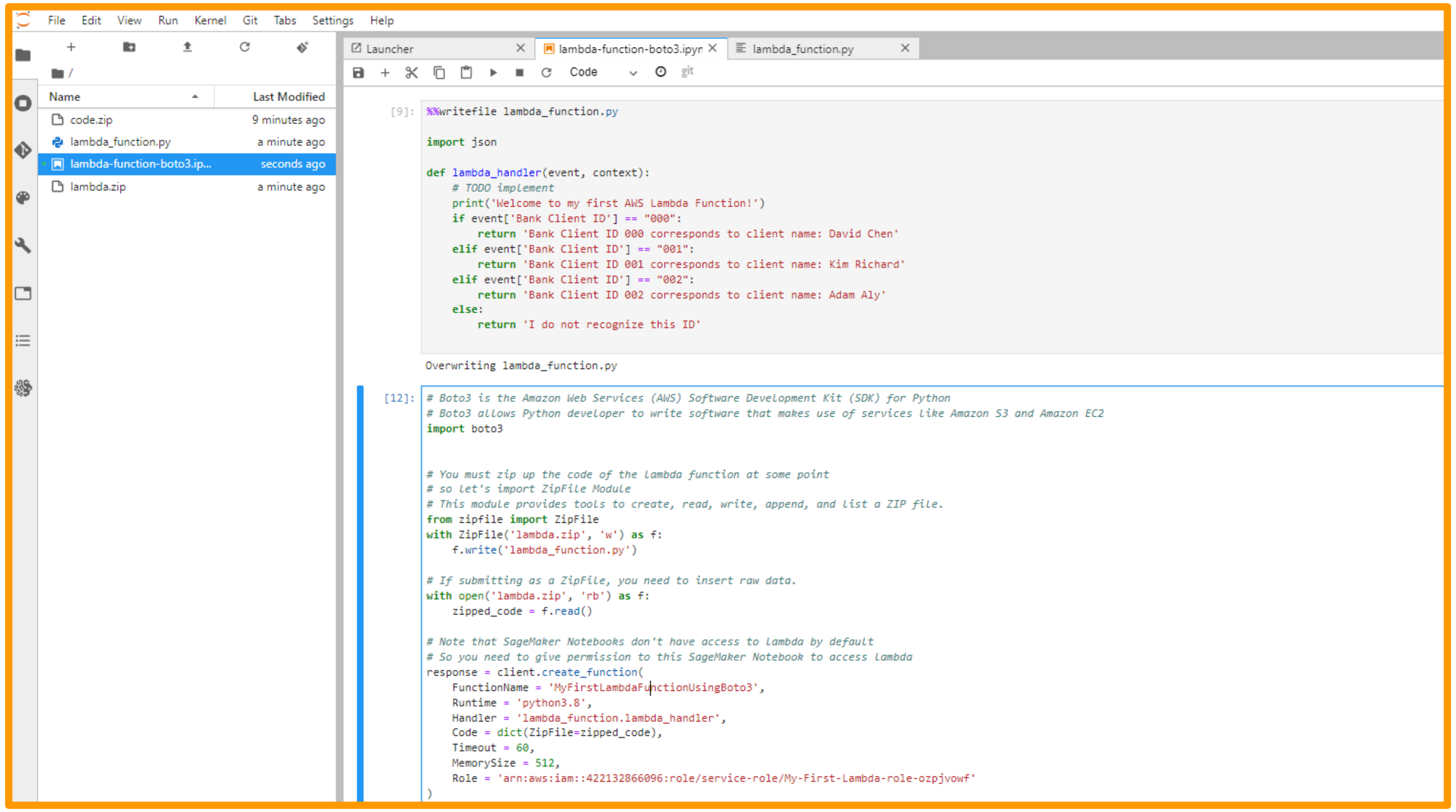
Policy name	Type	Description
AWSLambdaBasicExecutionRole-32c9b4c4-34ae-4da3-bafc-77bb6971acff	Customer managed	
AWSLambda_FullAccess	AWS managed	Grants full access to AWS Lambda service, AWS Lambda console features, and other related AWS services.

Permissions boundary - (not set)
Set a permissions boundary to control the maximum permissions this role can have. This is not a common setting but can be used to delegate permission management to others.

DEMO: CREATE AN AWS LAMBDA FUNCTION USING BOTO3

RUN THIS CODE, YOU SHOULD SEE A NEWLY CREATED LAMBDA FUNCTION ENTITLED “MyFirstLambdaFunctionUsingBoto3” AVAILABLE.

Note: %%writefile lets you output code developed in a Notebook to a Python module



Name	Last Modified
code.zip	9 minutes ago
lambda_function.py	a minute ago
lambda-function-boto3.ip...	seconds ago
lambda.zip	a minute ago

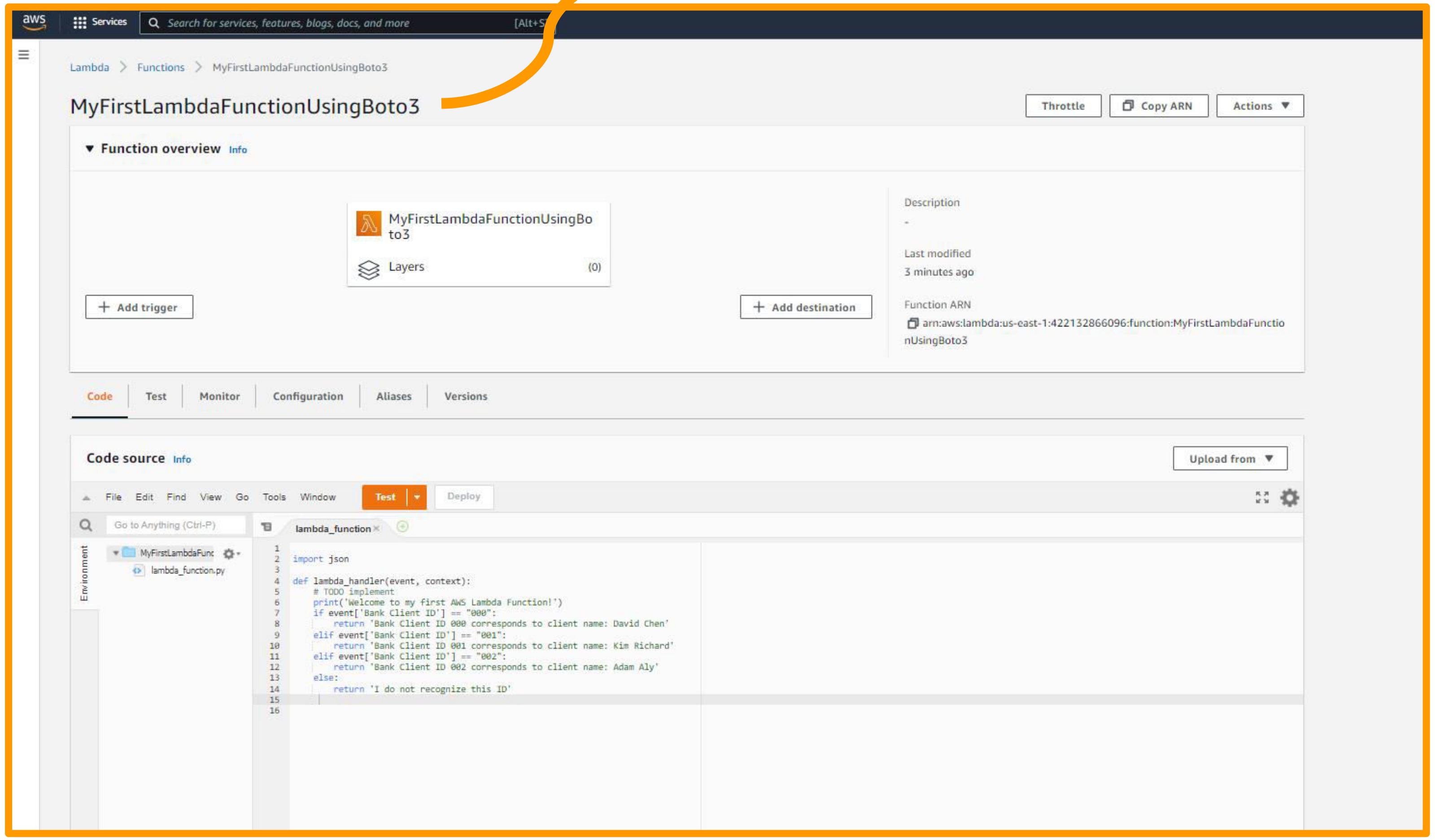
```
[9]: %%writefile lambda_function.py

import json

def lambda_handler(event, context):
    # TODO implement
    print('Welcome to my first AWS Lambda Function!')
    if event['Bank Client ID'] == "000":
        return 'Bank Client ID 000 corresponds to client name: David Chen'
    elif event['Bank Client ID'] == "001":
        return 'Bank Client ID 001 corresponds to client name: Kim Richard'
    elif event['Bank Client ID'] == "002":
        return 'Bank Client ID 002 corresponds to client name: Adam Aly'
    else:
        return 'I do not recognize this ID'
```

```
Overwriting lambda_function.py
```

```
[12]: # Boto3 is the Amazon Web Services (AWS) Software Development Kit (SDK) for Python
# Boto3 allows Python developer to write software that makes use of services like Amazon S3 and Amazon EC2
import boto3


# You must zip up the code of the lambda function at some point
# so let's import ZipFile Module
# This module provides tools to create, read, write, append, and list a ZIP file.
from zipfile import ZipFile
with ZipFile('lambda.zip', 'w') as f:
    f.write('lambda_function.py')

# If submitting as a ZipFile, you need to insert raw data.
with open('lambda.zip', 'rb') as f:
    zipped_code = f.read()

# Note that SageMaker Notebooks don't have access to lambda by default
# So you need to give permission to this SageMaker Notebook to access lambda
response = client.create_function(
    FunctionName = 'MyFirstLambdaFunctionUsingBoto3',
    Runtime = 'python3.8',
    Handler = 'lambda_function.lambda_handler',
    Code = dict(ZipFile=zipped_code),
    Timeout = 60,
    MemorySize = 512,
    Role = 'arn:aws:iam::422132866096:role/service-role/My-First-Lambda-role-ozpjvowf'
)
```

DEMO: CREATE AN AWS LAMBDA FUNCTION USING BOTO3

A NEW LAMBDA FUNCTION ENTITLED "MyFirstLambdaFunctionUsingBoto3" HAS BEEN CREATED USING BOTO3 SDK



Lambda > Functions > MyFirstLambdaFunctionUsingBoto3

MyFirstLambdaFunctionUsingBoto3

Function overview Info

MyFirstLambdaFunctionUsingBoto3

Description
-

Last modified
3 minutes ago

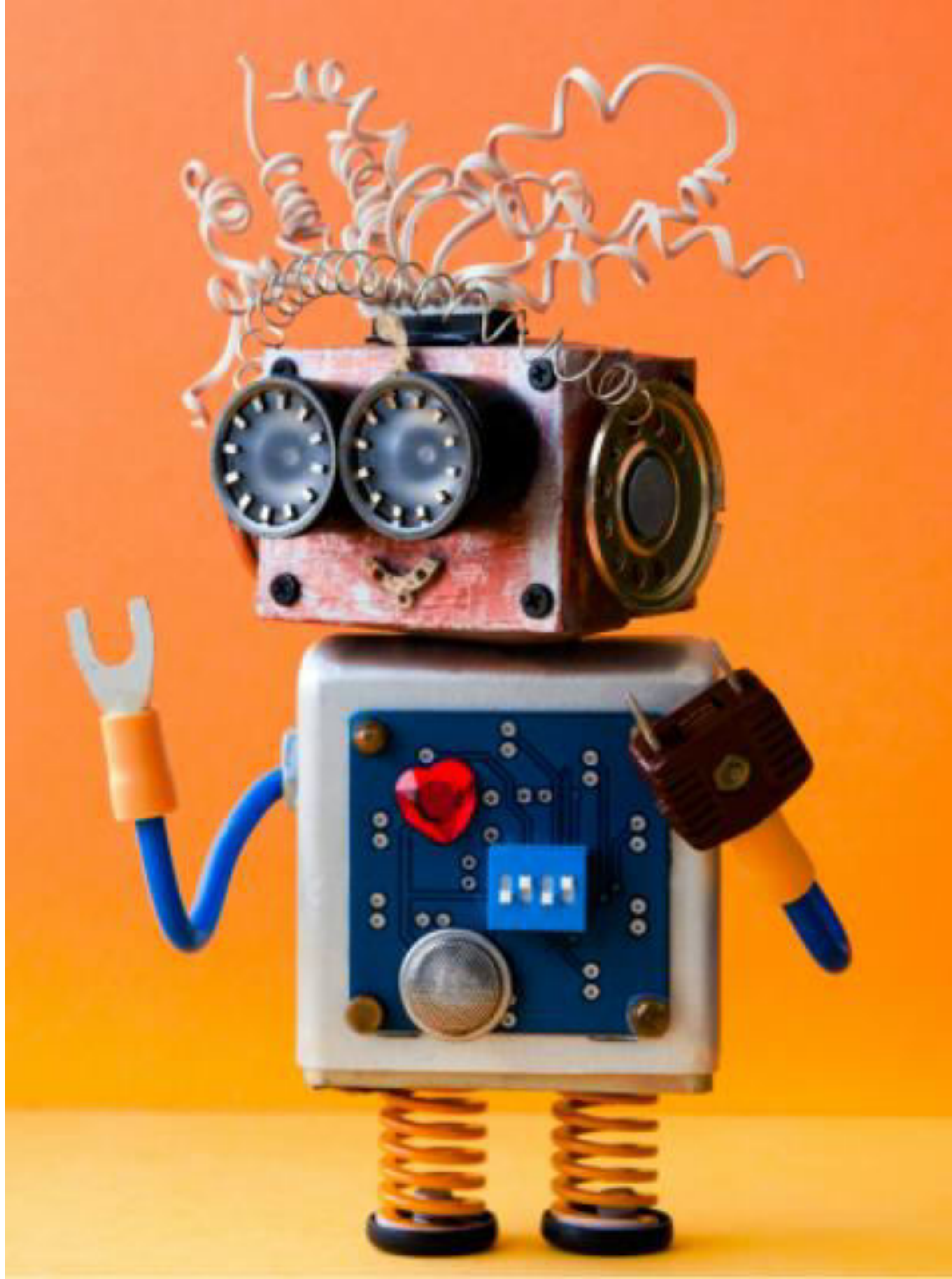
Function ARN
arn:aws:lambda:us-east-1:422132866096:function:MyFirstLambdaFunctionUsingBoto3

Code source Info

```python

import json

def lambda_handler(event, context):
    # TODO implement
    print('Welcome to my first AWS Lambda Function!')
    if event['Bank Client ID'] == "000":
        return 'Bank Client ID 000 corresponds to client name: David Chen'
    elif event['Bank Client ID'] == "001":
        return 'Bank Client ID 001 corresponds to client name: Kim Richard'
    elif event['Bank Client ID'] == "002":
        return 'Bank Client ID 002 corresponds to client name: Adam Aly'
    else:
        return 'I do not recognize this ID'

```

DEMO: LAMBDA INVOCATION WITH BOTO3 SDK

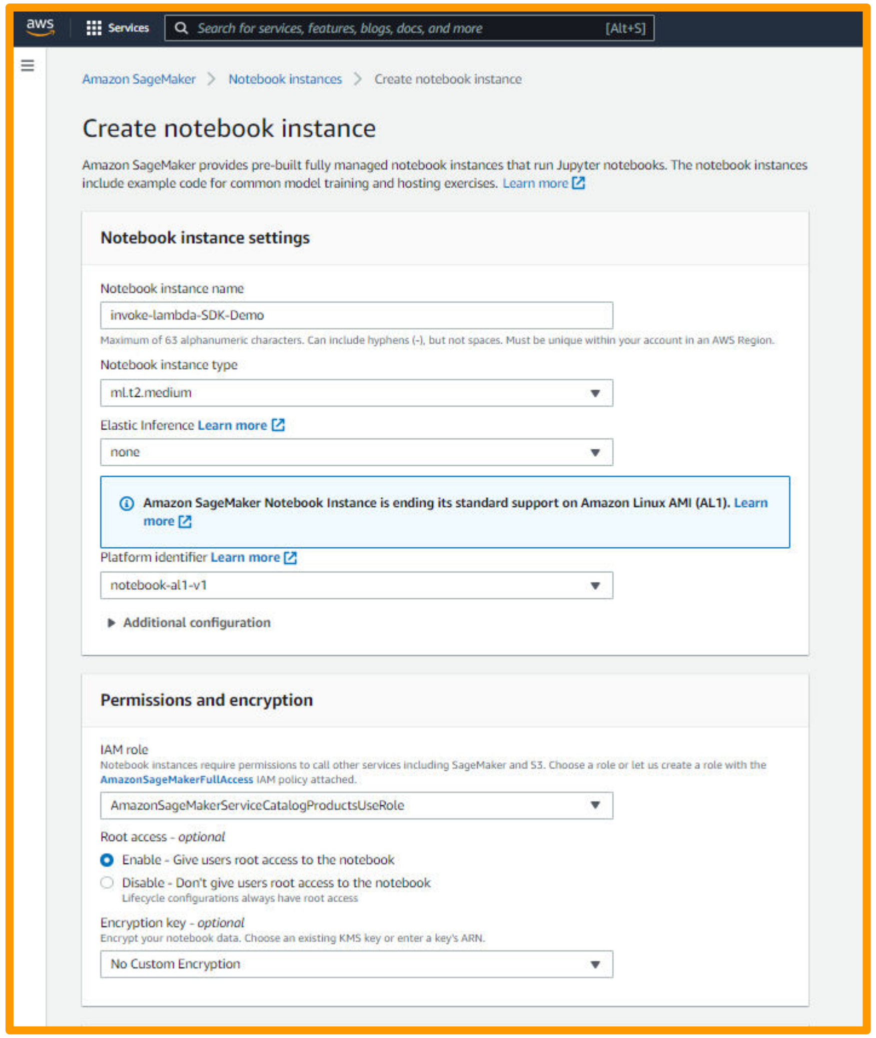
EASY ▲ ADVANCED



DEMO: LAMBDA INVOCATION WITH SDK

- Great Boto3 Documentation: https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/lambda.html#Lambda.Client.invoke

GO TO SAGEMAKER AND START A NEW NOTEBOOK



Amazon SageMaker > Notebook instances > Create notebook instance

Create notebook instance

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. Learn more

Notebook instance settings

Notebook instance name
invoke-lambda-SDK-Demo
Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Notebook instance type
ml.t2.medium

Elastic Inference Learn more
none

Amazon SageMaker Notebook Instance is ending its standard support on Amazon Linux AMI (AL1). Learn more

Platform identifier Learn more
notebook-al1-v1

Additional configuration

Permissions and encryption

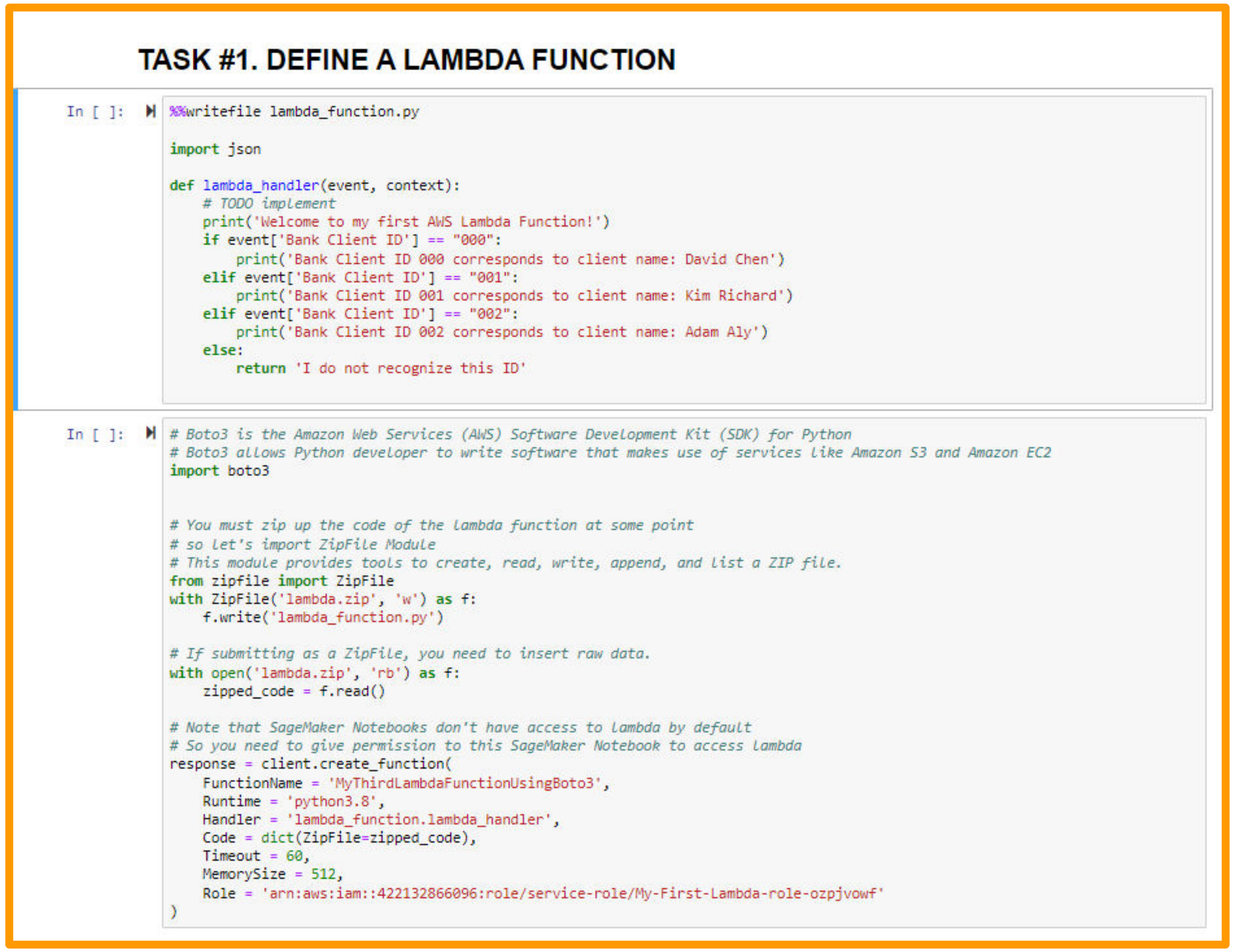
IAM role
Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the AmazonSageMakerFullAccess IAM policy attached.
AmazonSageMakerServiceCatalogProductsUseRole

Root access - *optional*
[✓ Enable - Give users root access to the notebook]
Disable - Don't give users root access to the notebook
Lifecycle configurations always have root access

Encryption key - *optional*
Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.
No Custom Encryption

DEMO: LAMBDA INVOCATION WITH SDK

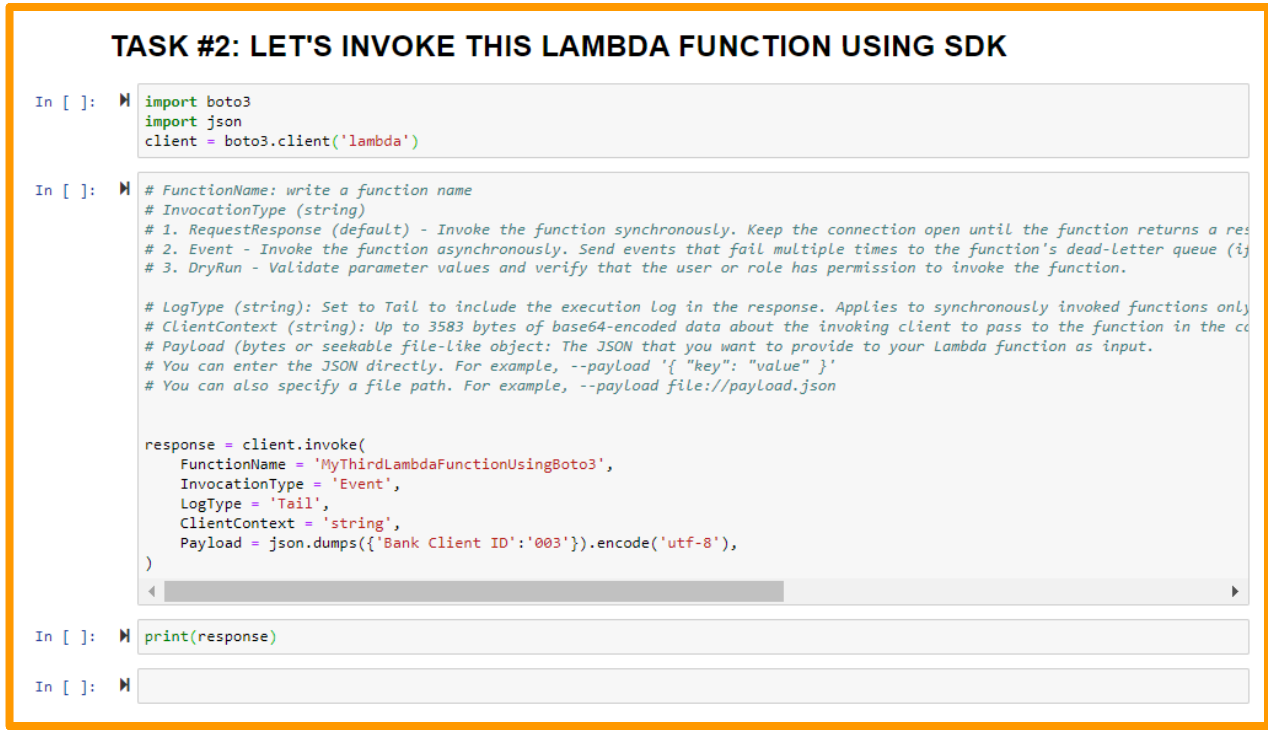
LET's DEFINE A NEW FUNCTION AND THEN INVOKE IT IN THE SDK



TASK #1. DEFINE A LAMBDA FUNCTION

```
In [ ]:  %%writefile lambda_function.py

         import json

         def lambda_handler(event, context):
             # TODO implement
             print('Welcome to my first AWS Lambda Function!')
             if event['Bank Client ID'] == "000":
                 print('Bank Client ID 000 corresponds to client name: David Chen')
             elif event['Bank Client ID'] == "001":
                 print('Bank Client ID 001 corresponds to client name: Kim Richard')
             elif event['Bank Client ID'] == "002":
                 print('Bank Client ID 002 corresponds to client name: Adam Aly')
             else:
                 return 'I do not recognize this ID'
```

```
In [ ]:  # Boto3 is the Amazon Web Services (AWS) Software Development Kit (SDK) for Python
         # Boto3 allows Python developer to write software that makes use of services like Amazon S3 and Amazon EC2
         import boto3


         # You must zip up the code of the lambda function at some point
         # so let's import ZipFile Module
         # This module provides tools to create, read, write, append, and list a ZIP file.
         from zipfile import ZipFile
         with ZipFile('lambda.zip', 'w') as f:
             f.write('lambda_function.py')

         # If submitting as a ZipFile, you need to insert raw data.
         with open('lambda.zip', 'rb') as f:
             zipped_code = f.read()

         # Note that SageMaker Notebooks don't have access to lambda by default
         # So you need to give permission to this SageMaker Notebook to access lambda
         response = client.create_function(
             FunctionName = 'MyThirdLambdaFunctionUsingBoto3',
             Runtime = 'python3.8',
             Handler = 'lambda_function.lambda_handler',
             Code = dict(ZipFile=zipped_code),
             Timeout = 60,
             MemorySize = 512,
             Role = 'arn:aws:iam::422132866096:role/service-role/My-First-Lambda-role-ozpjvowf'
         )
```



TASK #2: LET'S INVOKE THIS LAMBDA FUNCTION USING SDK

```
In [ ]:  import boto3
         import json
         client = boto3.client('lambda')
```

```
In [ ]:  # FunctionName: write a function name
         # InvocationType (string)
         # 1. RequestResponse (default) - Invoke the function synchronously. Keep the connection open until the function returns a res
         # 2. Event - Invoke the function asynchronously. Send events that fail multiple times to the function's dead-letter queue (i
         # 3. DryRun - Validate parameter values and verify that the user or role has permission to invoke the function.

         # LogType (string): Set to Tail to include the execution log in the response. Applies to synchronously invoked functions onl
         # ClientContext (string): Up to 3583 bytes of base64-encoded data about the invoking client to pass to the function in the c
         # Payload (bytes or seekable file-like object: The JSON that you want to provide to your Lambda function as input.
         # You can enter the JSON directly. For example, --payload '{ "key": "value" }'
         # You can also specify a file path. For example, --payload file://payload.json


         response = client.invoke(
             FunctionName = 'MyThirdLambdaFunctionUsingBoto3',
             InvocationType = 'Event',
             LogType = 'Tail',
             ClientContext = 'string',
             Payload = json.dumps({'Bank Client ID':'003'}).encode('utf-8'),
         )
```

```
In [ ]:  print(response)
```

```
In [ ]:
```

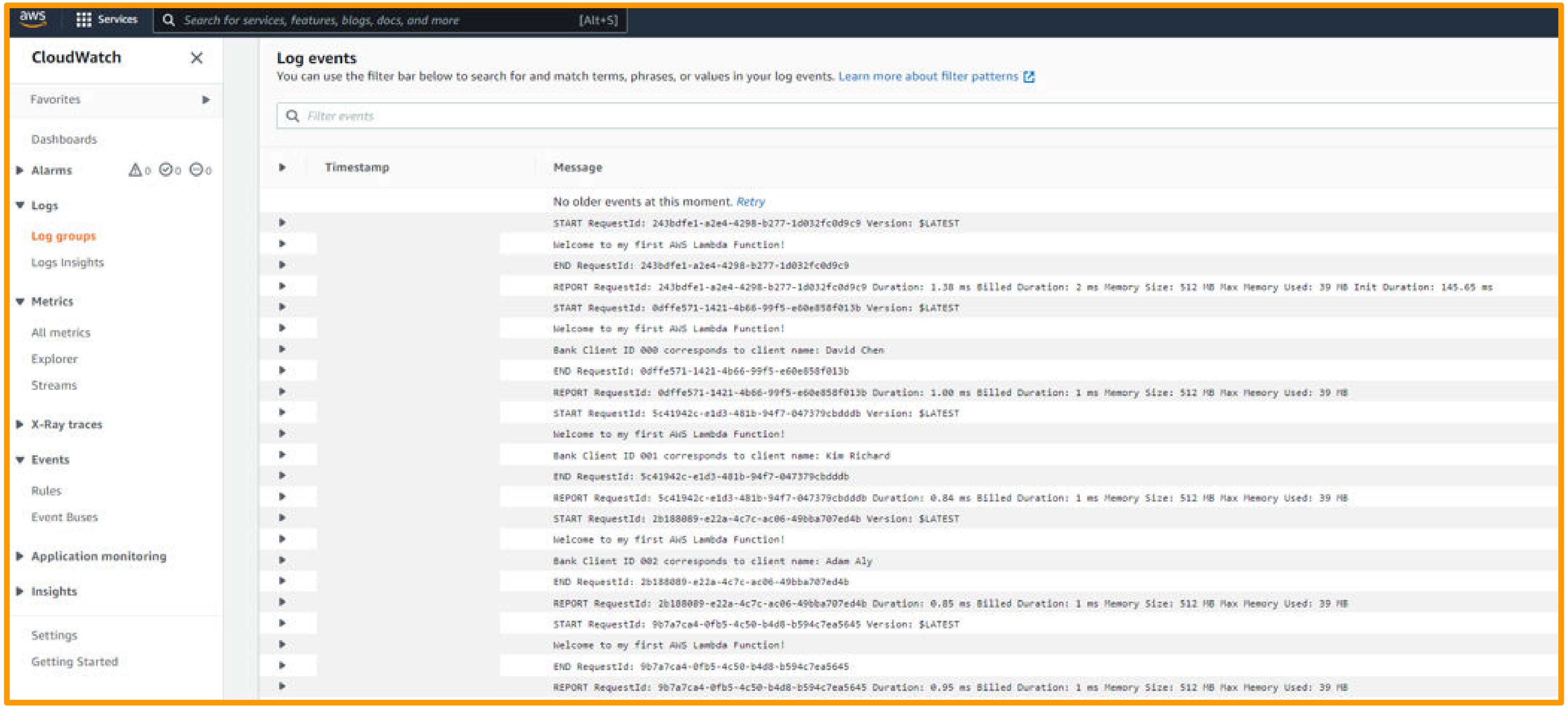
DEMO: LAMBDA INVOCATION WITH SDK

- **FunctionName:** write a function name
- **InvocationType (string)**
- 1. RequestResponse (default): Invoke the function synchronously.
- 2. Event: Invoke the function asynchronously.
- 3. DryRun: Validate parameter values and verify that the user or role has permission to invoke the function.

- **LogType (string):** Set to Tail to include the execution log in the response. Applies to synchronously invoked functions only.
- **ClientContext (string):** Up to 3583 bytes of base64-encoded data about the invoking client to pass to the function in the context object.
- **Payload:** The JSON that you want to provide to your Lambda function as input.
- You can enter the JSON directly. For example, --payload '{ "key": "value" }'
- You can also specify a file path. For example, --payload file://payload.json

DEMO: LAMBDA INVOCATION WITH SDK

GO TO CLOUDWATCH AND EXPLORE THE LOGS



Log events

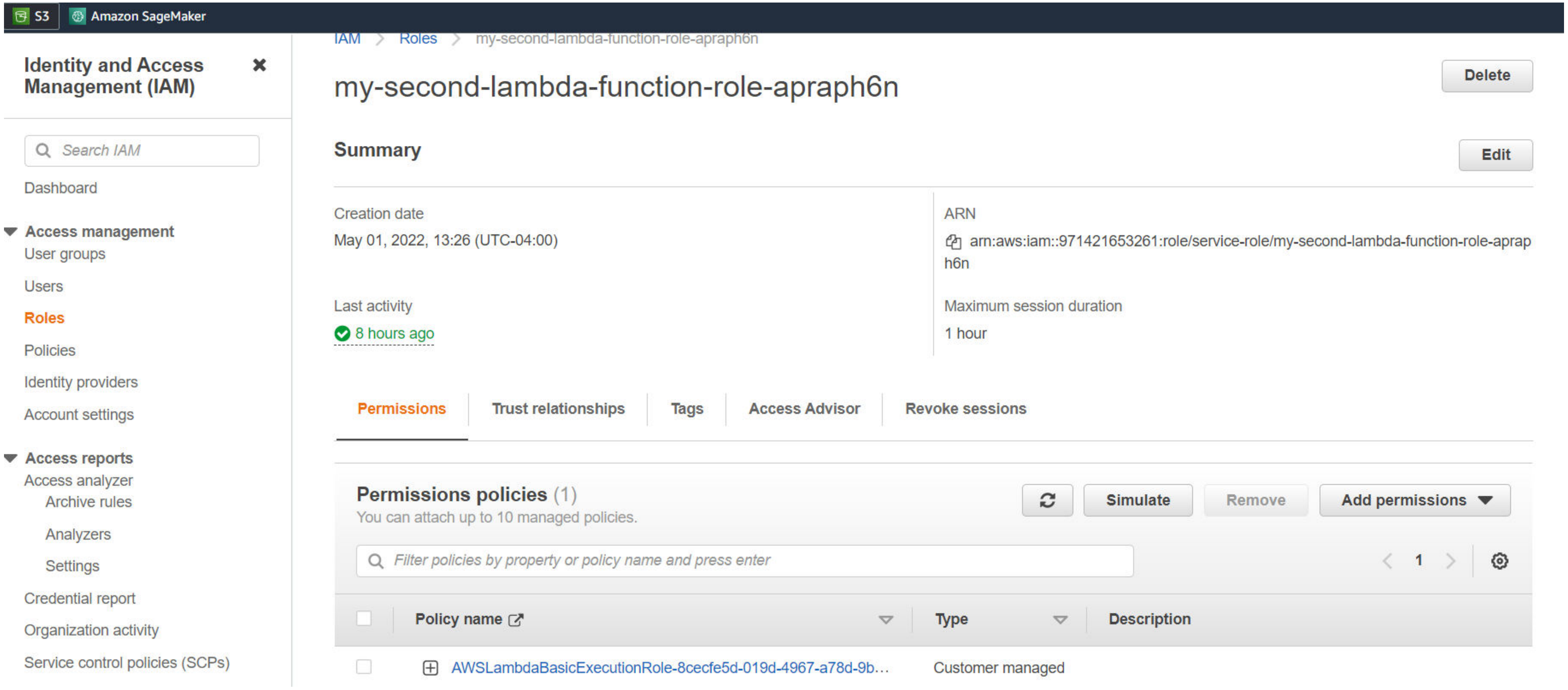
You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

No older events at this moment. *Retry*

```
START RequestId: 243bdfe1-a2e4-4298-b277-1d032fc0d9c9 Version: $LATEST
Welcome to my first AWS Lambda Function!
END RequestId: 243bdfe1-a2e4-4298-b277-1d032fc0d9c9
REPORT RequestId: 243bdfe1-a2e4-4298-b277-1d032fc0d9c9 Duration: 1.38 ms Billed Duration: 2 ms Memory Size: 512 MB Max Memory Used: 39 MB Init Duration: 145.65 ms
START RequestId: 0dffe571-1421-4b66-99f5-e60e858f013b Version: $LATEST
Welcome to my first AWS Lambda Function!
Bank Client ID 000 corresponds to client name: David Chen
END RequestId: 0dffe571-1421-4b66-99f5-e60e858f013b
REPORT RequestId: 0dffe571-1421-4b66-99f5-e60e858f013b Duration: 1.00 ms Billed Duration: 1 ms Memory Size: 512 MB Max Memory Used: 39 MB
START RequestId: 5c41942c-e1d3-481b-94f7-047379cbdddb Version: $LATEST
Welcome to my first AWS Lambda Function!
Bank Client ID 001 corresponds to client name: Kim Richard
END RequestId: 5c41942c-e1d3-481b-94f7-047379cbdddb
REPORT RequestId: 5c41942c-e1d3-481b-94f7-047379cbdddb Duration: 0.84 ms Billed Duration: 1 ms Memory Size: 512 MB Max Memory Used: 39 MB
START RequestId: 2b188089-e22a-4c7c-ac06-49bba707ed4b Version: $LATEST
Welcome to my first AWS Lambda Function!
Bank Client ID 002 corresponds to client name: Adam Aly
END RequestId: 2b188089-e22a-4c7c-ac06-49bba707ed4b
REPORT RequestId: 2b188089-e22a-4c7c-ac06-49bba707ed4b Duration: 0.85 ms Billed Duration: 1 ms Memory Size: 512 MB Max Memory Used: 39 MB
START RequestId: 9b7a7ca4-0fb5-4c50-b4d8-b594c7ea5645 Version: $LATEST
Welcome to my first AWS Lambda Function!
END RequestId: 9b7a7ca4-0fb5-4c50-b4d8-b594c7ea5645
REPORT RequestId: 9b7a7ca4-0fb5-4c50-b4d8-b594c7ea5645 Duration: 0.95 ms Billed Duration: 1 ms Memory Size: 512 MB Max Memory Used: 39 MB
```

DEMO: LAMBDA INVOCATION WITH SDK

NOTE THAT YOU MIGHT NEED TO ATTACH CLOUDWATCH FULL ACCESS POLICY TO YOUR LAMBDA FUNCTION ROLE TO SEE THE LOGS



my-second-lambda-function-role-apraph6n

Summary

Creation date
May 01, 2022, 13:26 (UTC-04:00)

ARN
arn:aws:iam::971421653261:role/service-role/my-second-lambda-function-role-apraph6n

Last activity
8 hours ago

Maximum session duration
1 hour

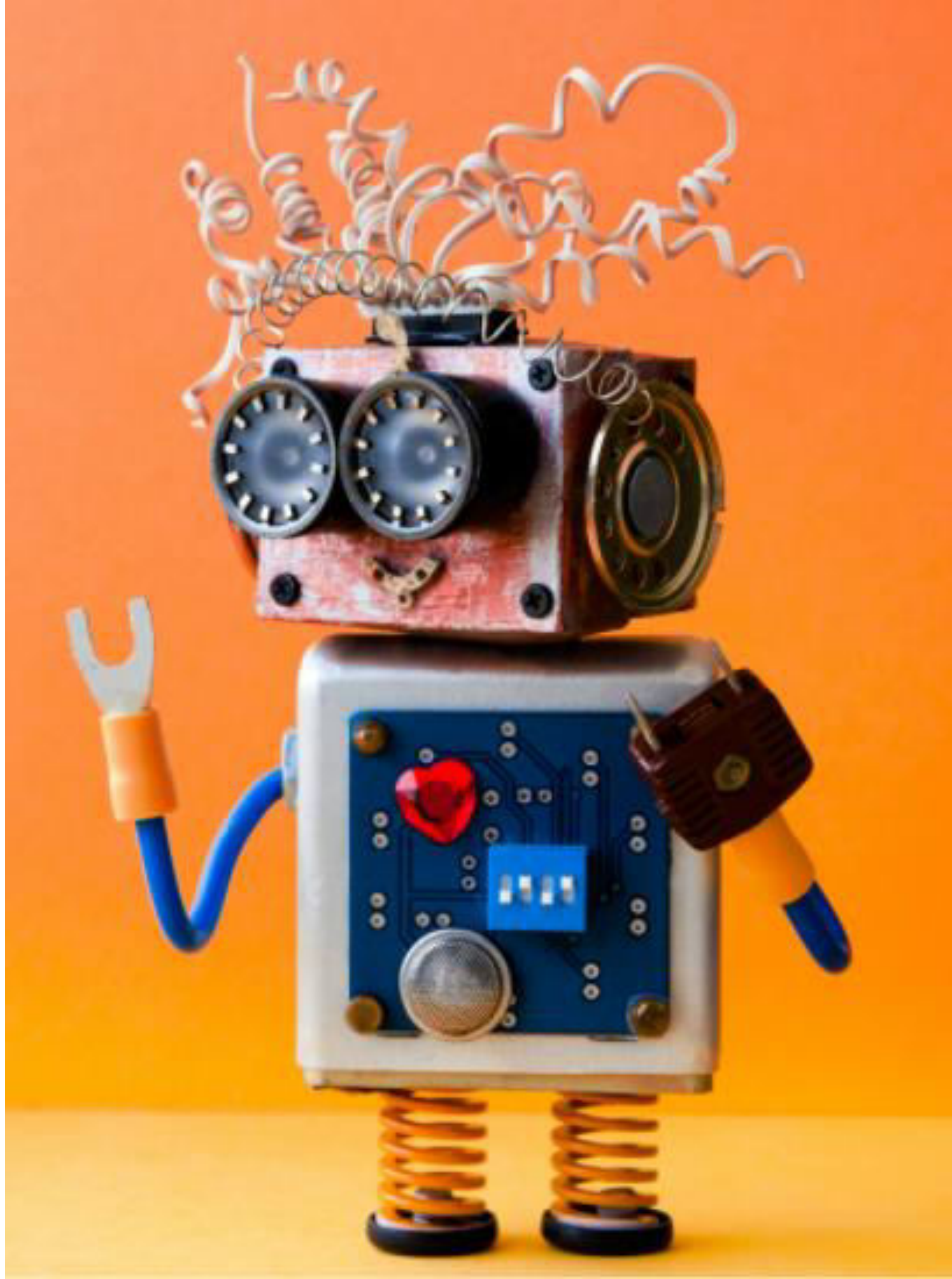
Permissions policies (1)
You can attach up to 10 managed policies.

Filter policies by property or policy name and press enter

Policy name	Type	Description
AWSLambdaBasicExecutionRole-8cecfe5d-019d-4967-a78d-9b…	Customer managed	

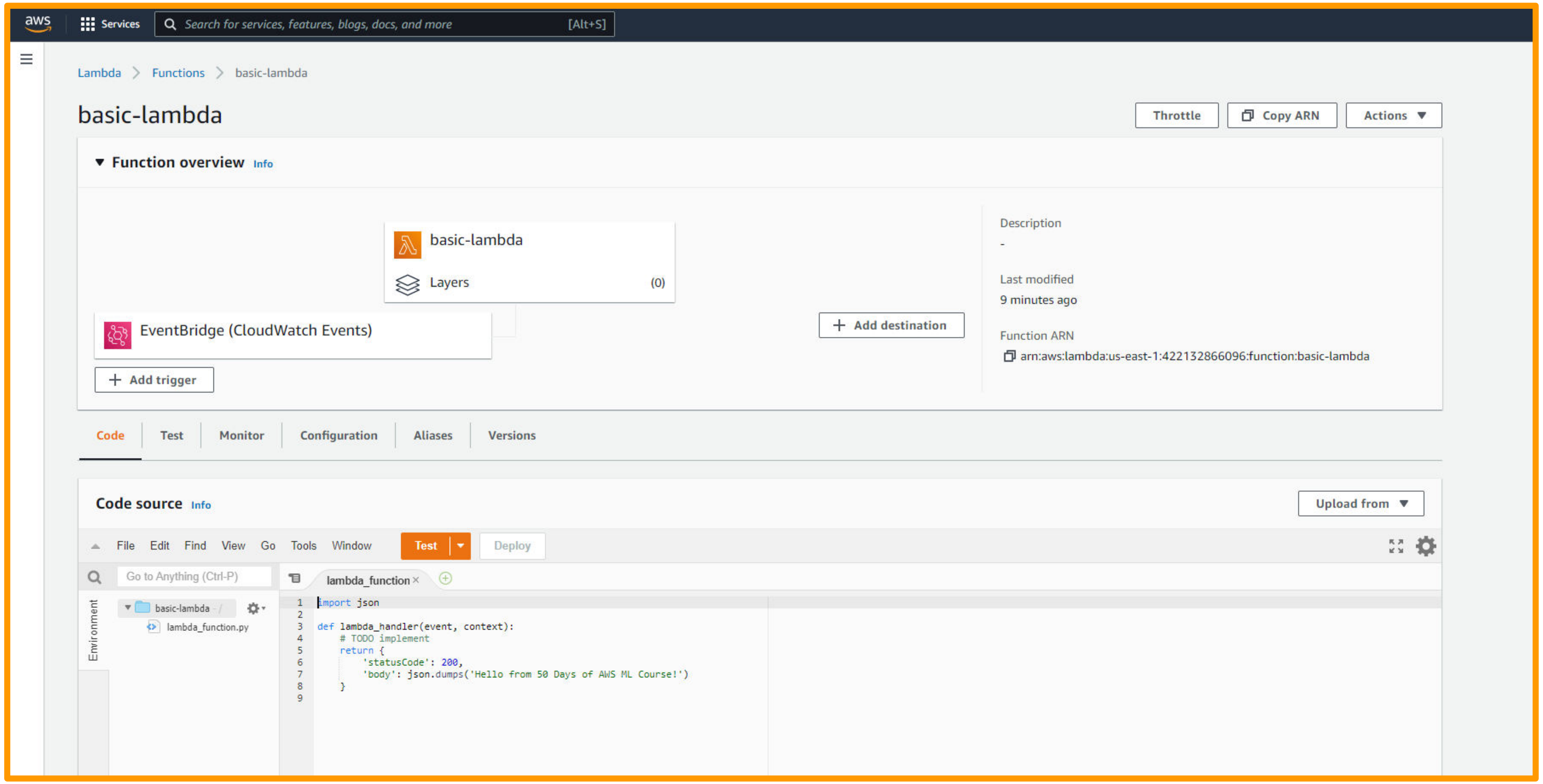
DEMO: LAMBDA INVOCATION WITH EVENTBRIDGE

EASY ADVANCED



DEMO: LAMBDA INVOCATION WITH EVENTBRIDGE

CREATE A BASIC LAMBDA FUNCTION NAMED “basic-lambda”



Description
-

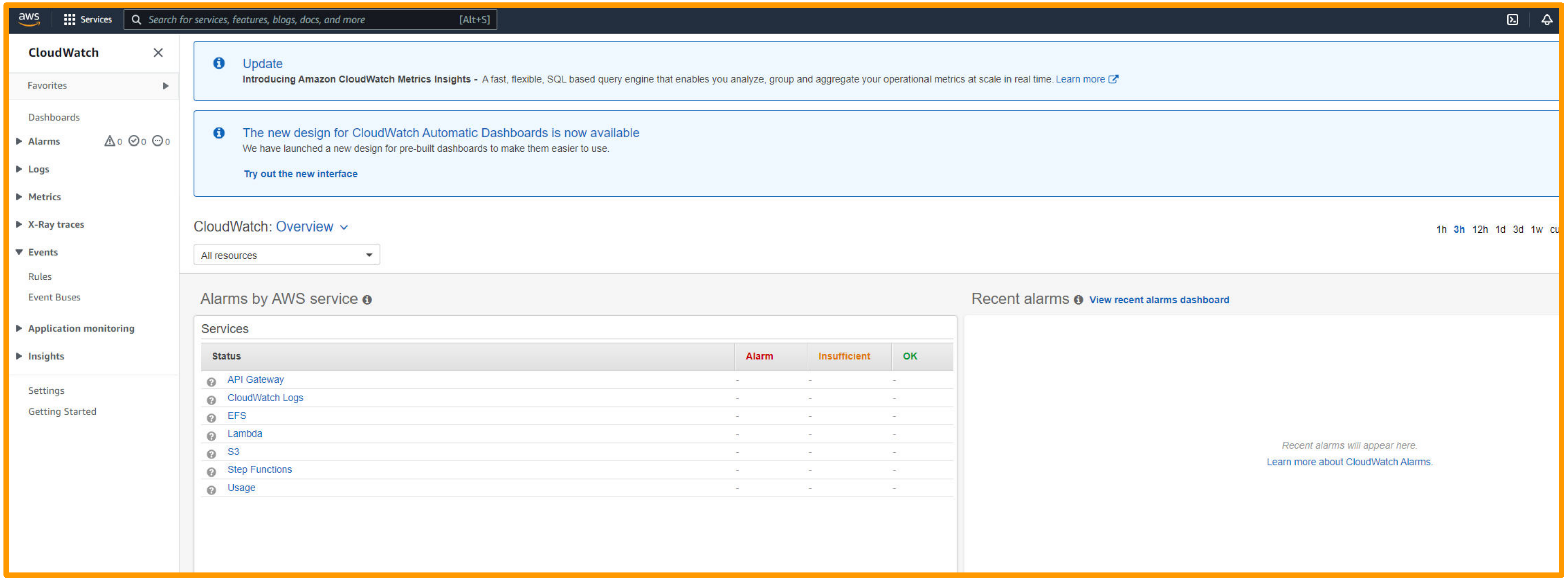
Last modified
9 minutes ago

Function ARN
arn:aws:lambda:us-east-1:422132866096:function:basic-lambda

```python
import json

def lambda_handler(event, context):
    # TODO implement
    return {
        'statusCode': 200,
        'body': json.dumps('Hello from 50 Days of AWS ML Course!')
    }

```

DEMO: LAMBDA INVOCATION WITH EVENTBRIDGE

GO TO CLOUDWATCH AND CLICK RULES



Update

Introducing Amazon CloudWatch Metrics Insights - A fast, flexible, SQL based query engine that enables you analyze, group and aggregate your operational metrics at scale in real time. Learn more

The new design for CloudWatch Automatic Dashboards is now available

We have launched a new design for pre-built dashboards to make them easier to use.

Try out the new interface

CloudWatch: Overview

All resources

1h 3h 12h 1d 3d 1w cu

Alarms by AWS service

Services

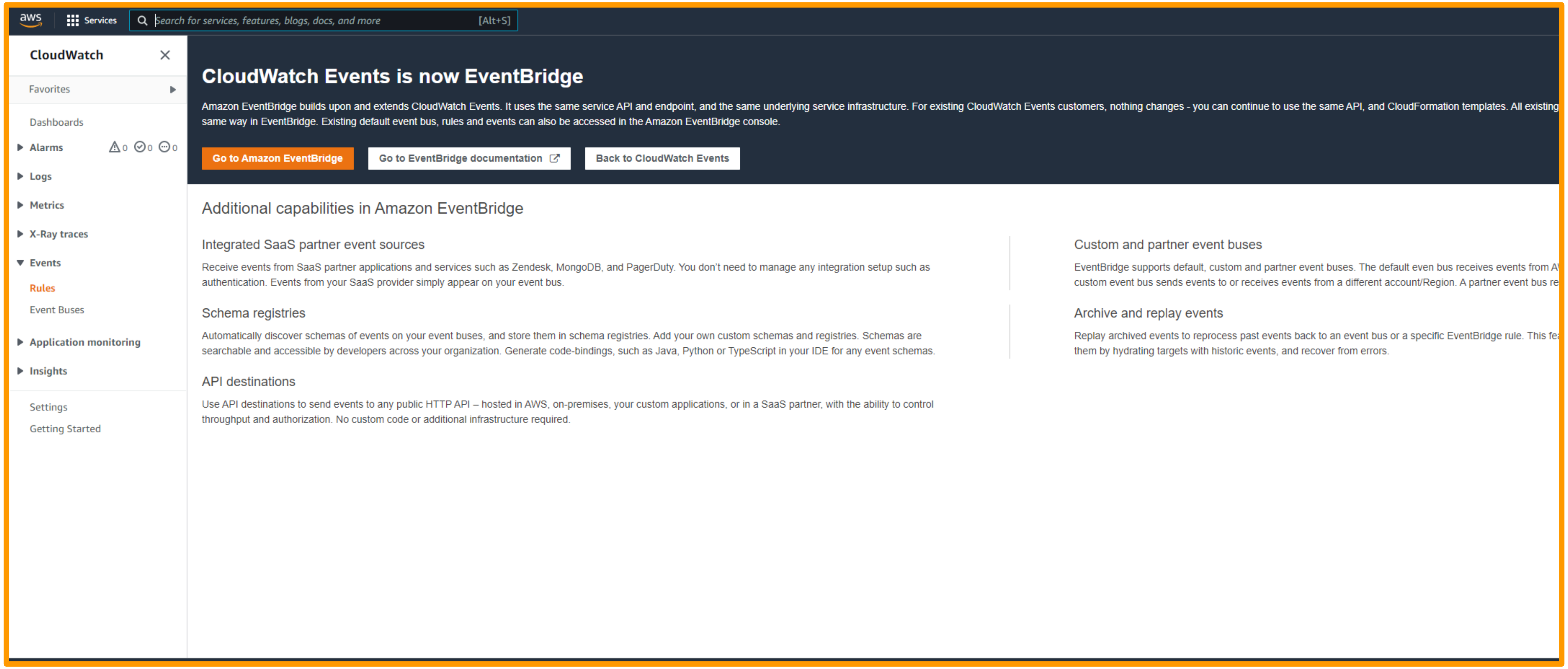
Status	Alarm	Insufficient	OK
API Gateway	-	-	-
CloudWatch Logs	-	-	-
EFS	-	-	-
Lambda	-	-	-
S3	-	-	-
Step Functions	-	-	-
Usage	-	-	-

Recent alarms View recent alarms dashboard

Recent alarms will appear here.
Learn more about CloudWatch Alarms.

DEMO: LAMBDA INVOCATION WITH EVENTBRIDGE

CLOUDWATCH EVENT IS NOW CALLED EVENTBRIDGE,
CLICK ON GO TO AMAZON EVENTBRIDGE



CloudWatch Events is now EventBridge

Amazon EventBridge builds upon and extends CloudWatch Events. It uses the same service API and endpoint, and the same underlying service infrastructure. For existing CloudWatch Events customers, nothing changes - you can continue to use the same API, and CloudFormation templates. All existing
same way in EventBridge. Existing default event bus, rules and events can also be accessed in the Amazon EventBridge console.

Additional capabilities in Amazon EventBridge

Integrated SaaS partner event sources

Receive events from SaaS partner applications and services such as Zendesk, MongoDB, and PagerDuty. You don't need to manage any integration setup such as authentication. Events from your SaaS provider simply appear on your event bus.

Schema registries

Automatically discover schemas of events on your event buses, and store them in schema registries. Add your own custom schemas and registries. Schemas are searchable and accessible by developers across your organization. Generate code-bindings, such as Java, Python or TypeScript in your IDE for any event schemas.

API destinations

Use API destinations to send events to any public HTTP API – hosted in AWS, on-premises, your custom applications, or in a SaaS partner, with the ability to control throughput and authorization. No custom code or additional infrastructure required.

Custom and partner event buses

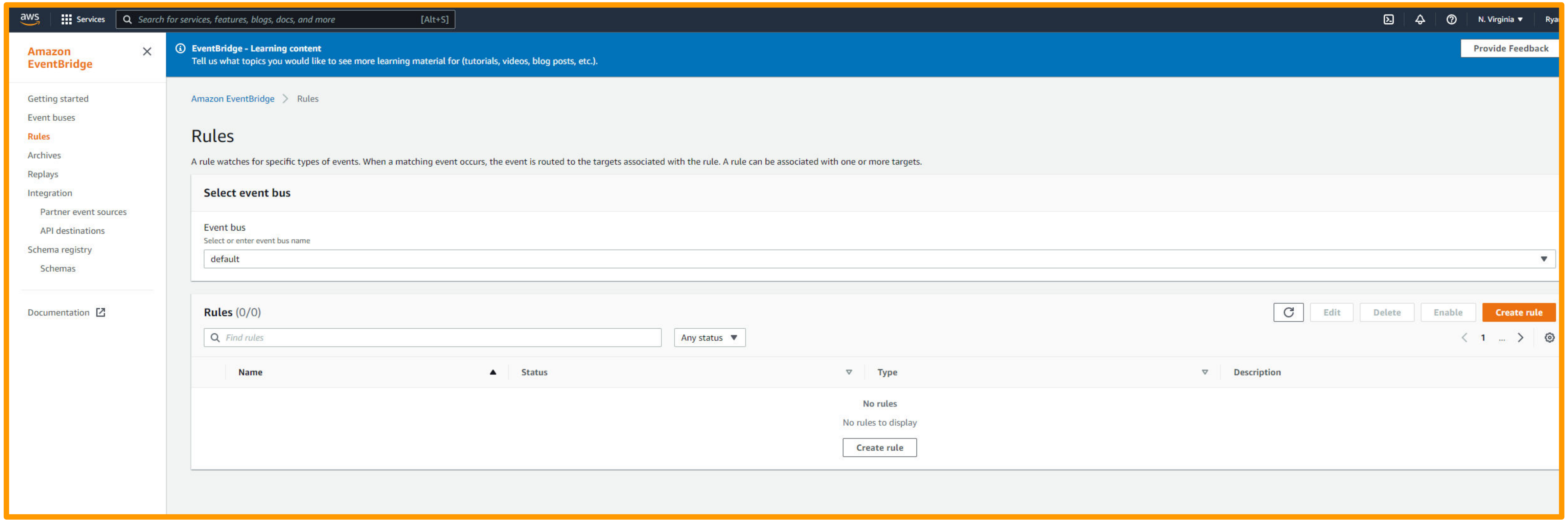
EventBridge supports default, custom and partner event buses. The default even bus receives events from A
custom event bus sends events to or receives events from a different account/Region. A partner event bus re

Archive and replay events

Replay archived events to reprocess past events back to an event bus or a specific EventBridge rule. This fe
them by hydrating targets with historic events, and recover from errors.

DEMO: LAMBDA INVOCATION WITH EVENTBRIDGE

CLICK ON CREATE RULE



EventBridge - Learning content
Tell us what topics you would like to see more learning material for (tutorials, videos, blog posts, etc.).

Rules

A rule watches for specific types of events. When a matching event occurs, the event is routed to the targets associated with the rule. A rule can be associated with one or more targets.

Select event bus

Event bus
Select or enter event bus name
default

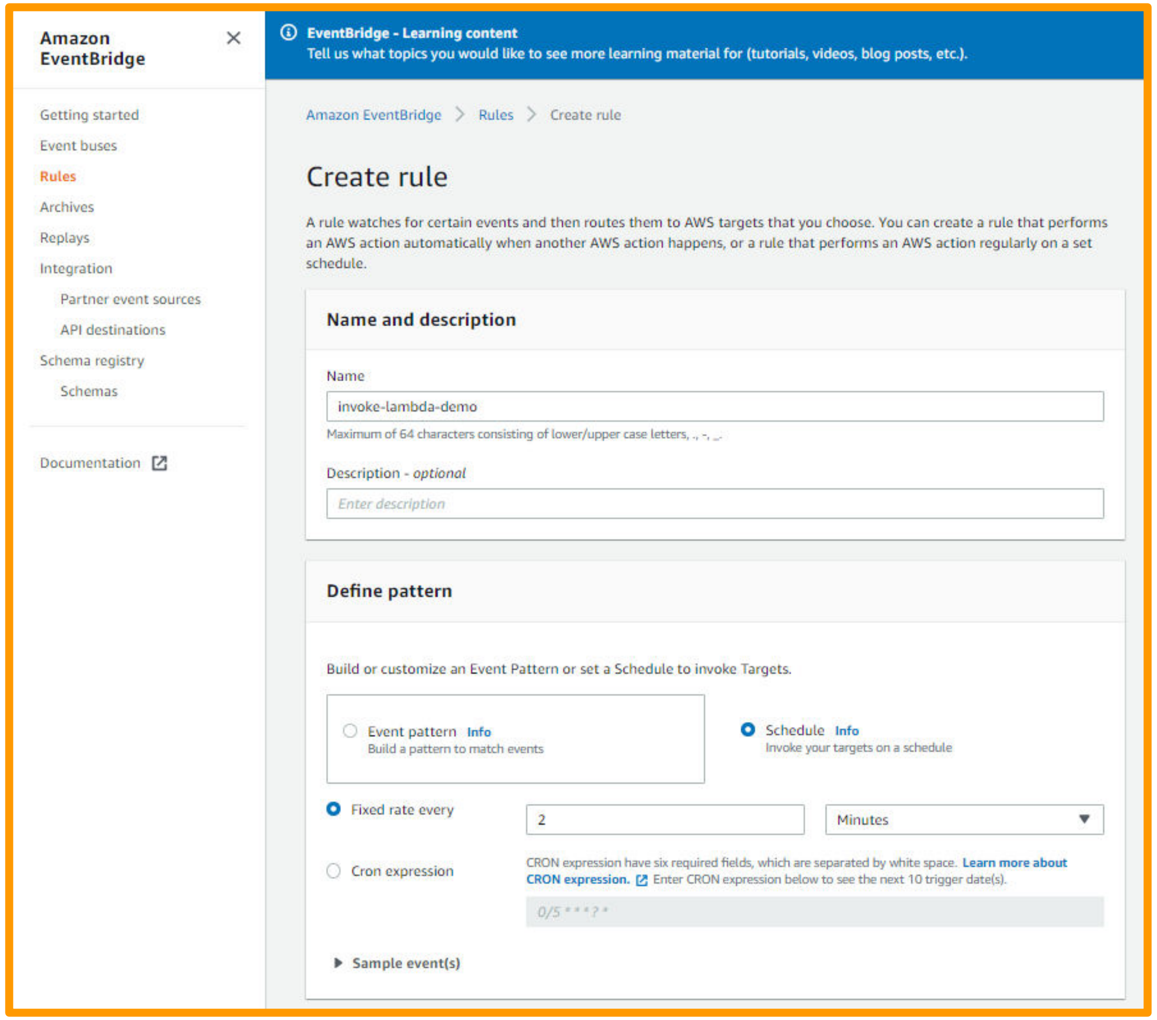
Rules (0/0)

Name	Status	Type	Description

No rules
No rules to display

DEMO: LAMBDA INVOCATION WITH EVENTBRIDGE

LET'S INVOKE THE LAMBDA FUNCTION ONCE EVERY 2 MINUTES



EventBridge - Learning content
Tell us what topics you would like to see more learning material for (tutorials, videos, blog posts, etc.).

Amazon EventBridge > Rules > Create rule

Create rule

A rule watches for certain events and then routes them to AWS targets that you choose. You can create a rule that performs an AWS action automatically when another AWS action happens, or a rule that performs an AWS action regularly on a set schedule.

Name and description

Name: invoke-lambda-demo
Maximum of 64 characters consisting of lower/upper case letters, ., -, _.

Description - *optional*: *Enter description*

Define pattern

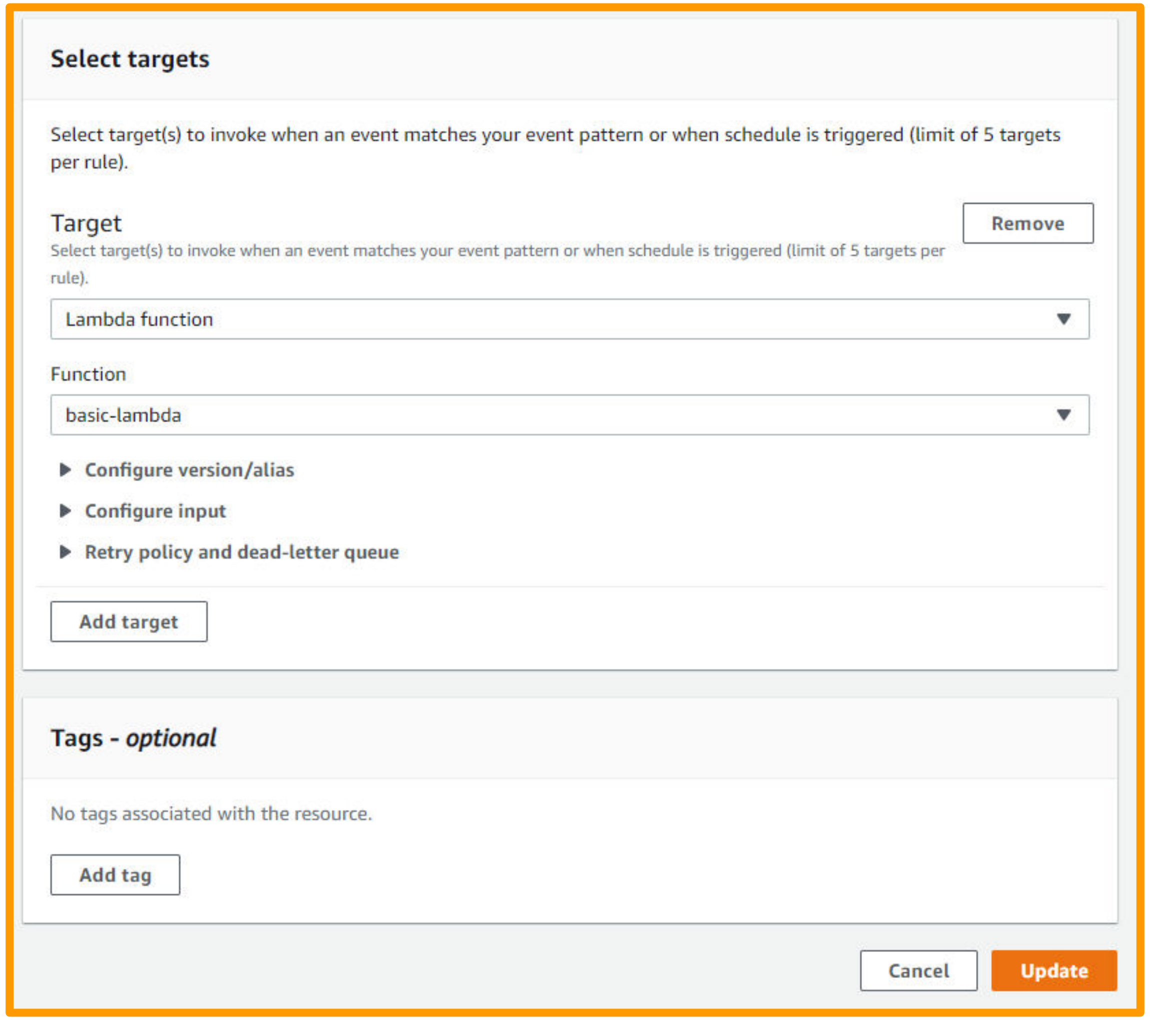
Build or customize an Event Pattern or set a Schedule to invoke Targets.

Event pattern Info — Build a pattern to match events [✓ Schedule Info — Invoke your targets on a schedule]

[✓ Fixed rate every] 2 Minutes

Cron expression — CRON expression have six required fields, which are separated by white space. Learn more about CRON expression. Enter CRON expression below to see the next 10 trigger date(s).
*0/5 * * * ? **

▶ Sample event(s)



Select targets

Select target(s) to invoke when an event matches your event pattern or when schedule is triggered (limit of 5 targets per rule).

Target
Select target(s) to invoke when an event matches your event pattern or when schedule is triggered (limit of 5 targets per rule).

Lambda function

Function: basic-lambda

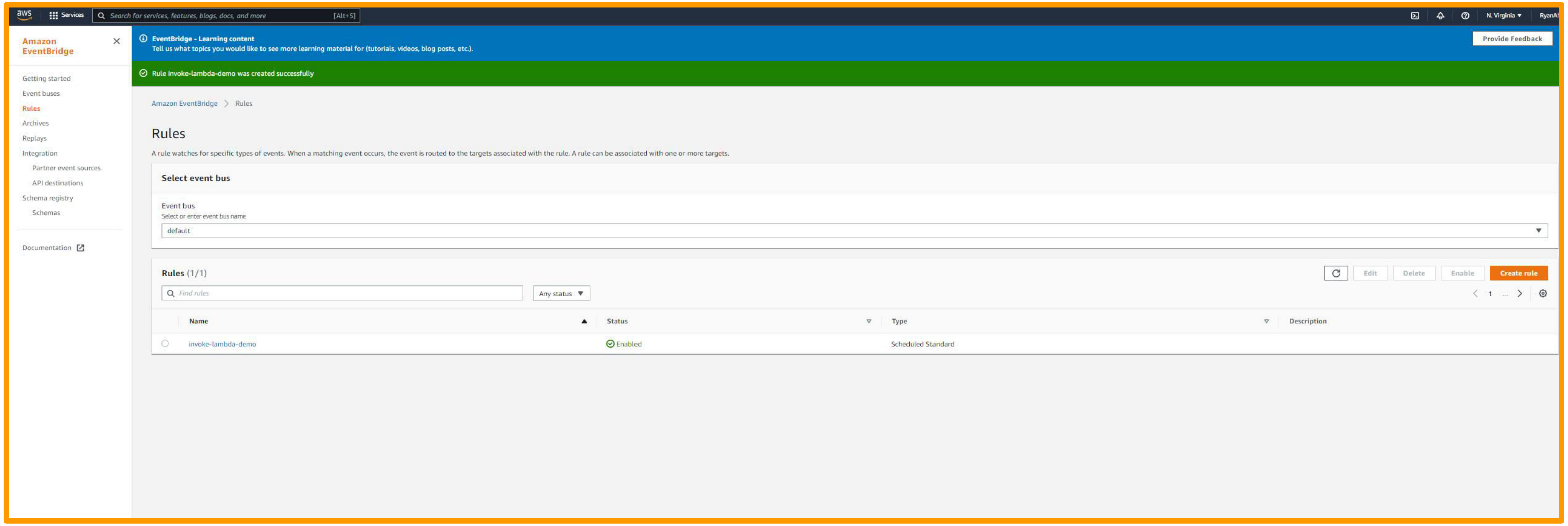
▶ Configure version/alias
▶ Configure input
▶ Retry policy and dead-letter queue

Tags - *optional*

No tags associated with the resource.

DEMO: LAMBDA INVOCATION WITH EVENTBRIDGE

NOW THE RULE HAS BEEN ACTIVATED, LET'S SEE IF IT WORKS!



EventBridge - Learning content
Tell us what topics you would like to see more learning material for (tutorials, videos, blog posts, etc.).

Rule invoke-lambda-demo was created successfully

Rules

A rule watches for specific types of events. When a matching event occurs, the event is routed to the targets associated with the rule. A rule can be associated with one or more targets.

Select event bus

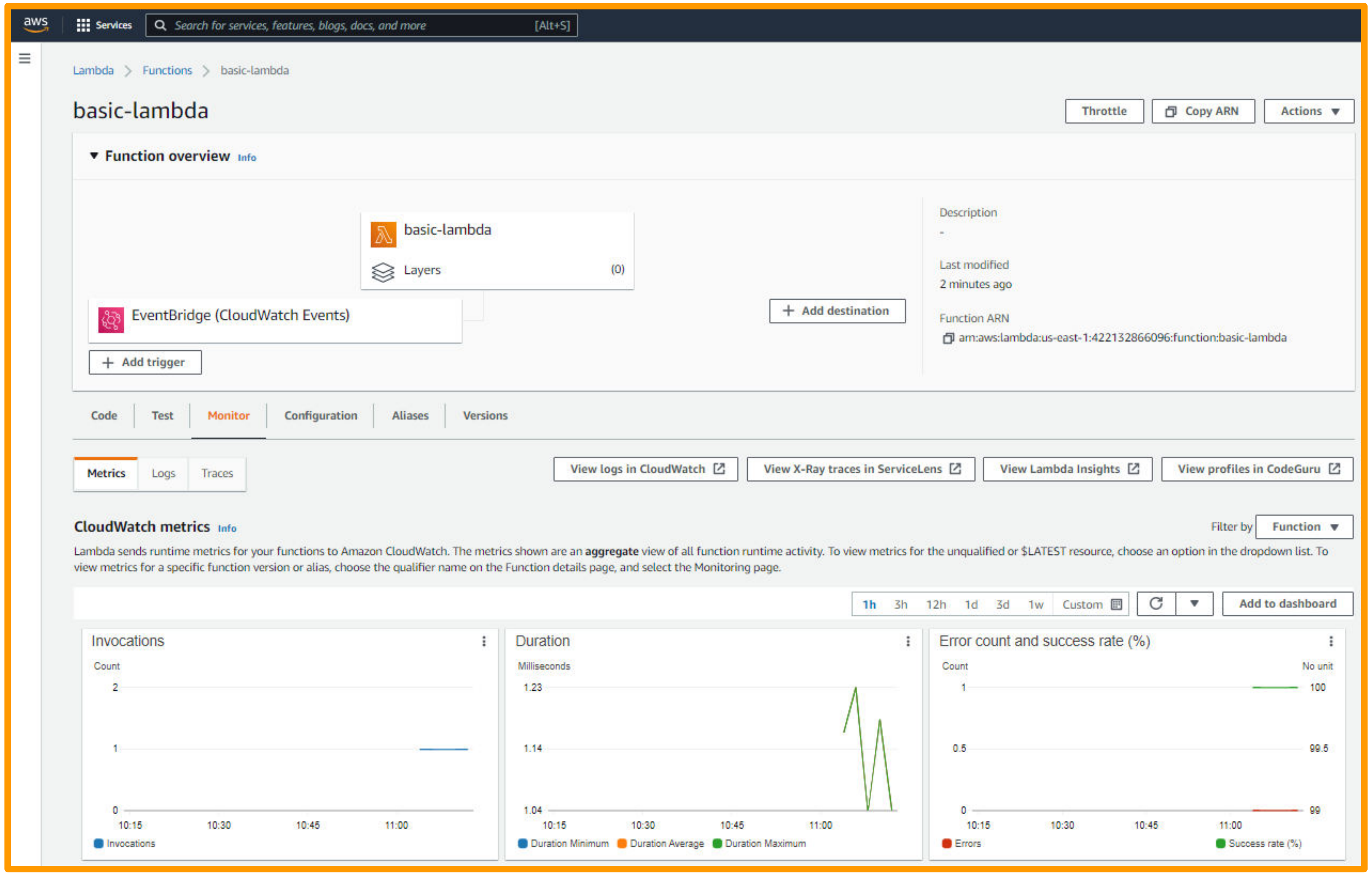
Event bus
Select or enter event bus name
default

Rules (1/1)

Name	Status	Type	Description
invoke-lambda-demo	Enabled	Scheduled Standard	

DEMO: LAMBDA INVOCATION WITH EVENTBRIDGE

GO BACK TO THE LAMBDA FUNCTION AND CLICK ON MONITOR


aws
Services
Search for services, features, blogs, docs, and more [Alt+S]
Lambda > Functions > basic-lambda

basic-lambda

Function overview Info

Description
-

Last modified
2 minutes ago

Function ARN
arn:aws:lambda:us-east-1:422132866096:function:basic-lambda

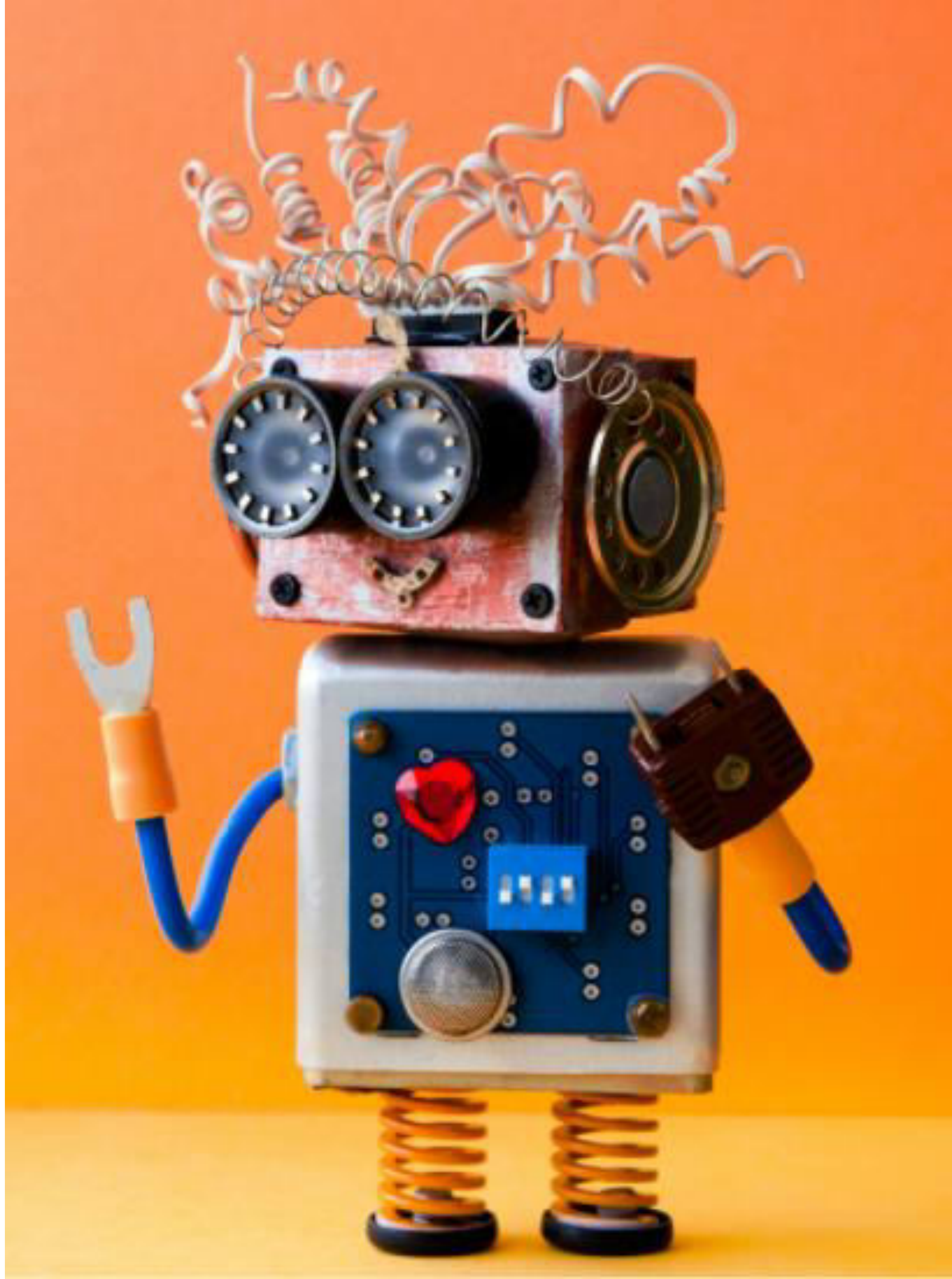
CloudWatch metrics Info

Filter by Function

Lambda sends runtime metrics for your functions to Amazon CloudWatch. The metrics shown are an **aggregate** view of all function runtime activity. To view metrics for the unqualified or $LATEST resource, choose an option in the dropdown list. To view metrics for a specific function version or alias, choose the qualifier name on the Function details page, and select the Monitoring page.

FINAL END-OF-DAY CAPSTONE PROJECT

EASY ADVANCED

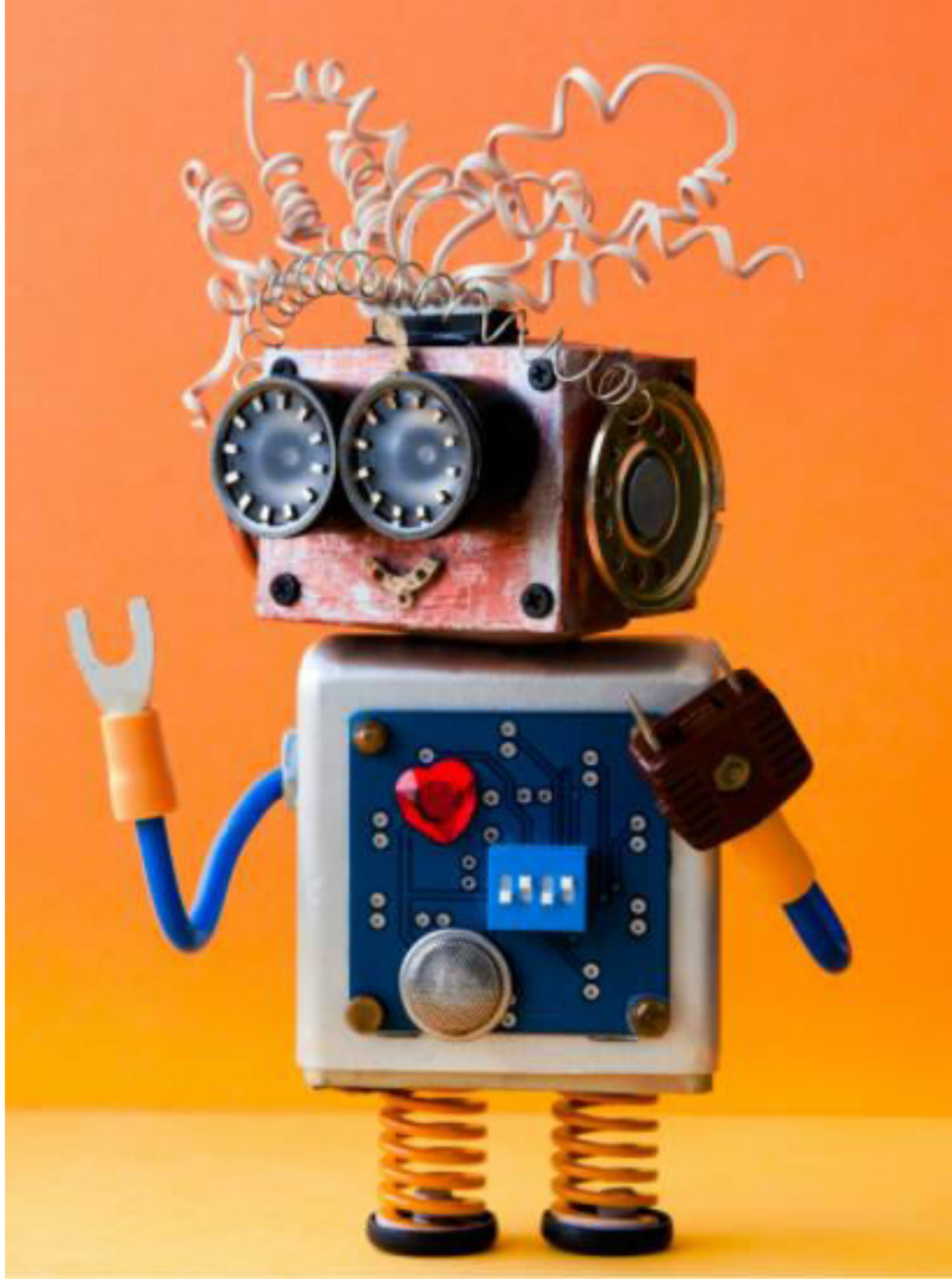


FINAL CAPSTONE PROJECT

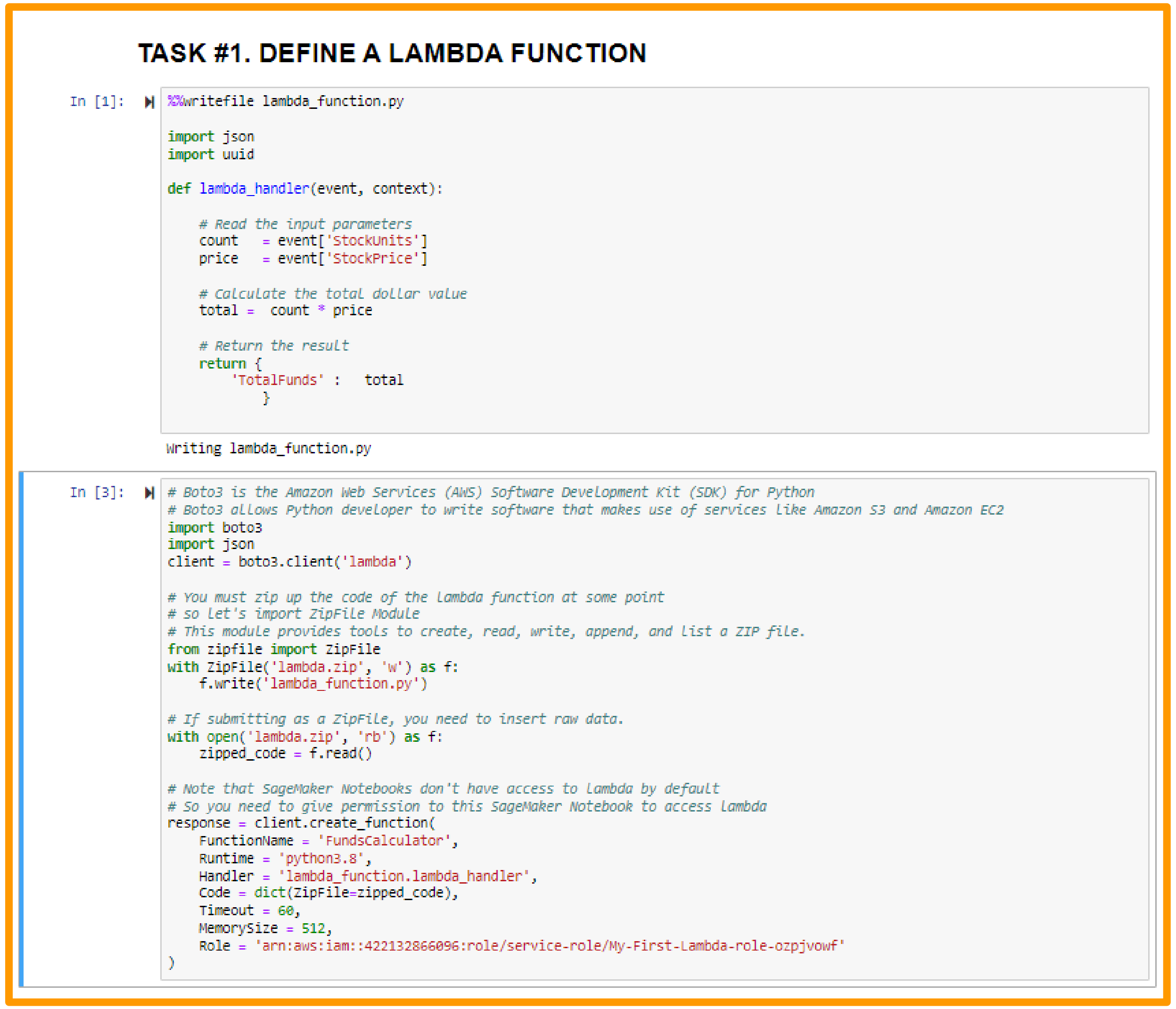
- Using boto3 SDK, define a Lambda Function named “FundsCalculator” that takes in the number of stock units and stock price and calculates the total value of the portfolio.
- Develop a script that consumes the total number of stocks and price of stocks from users.
- Return the total value of the portfolio.
- Configure the following test events using AWS Boto3 SDK:
    - Stock Units = 20, stock value = $1000
    - Stock Units = 5, stock value = $2000
- Monitor the logs in CloudWatch and ensure that the Lambda Function execution was successful.

FINAL END-OF-DAY CAPSTONE PROJECT SOLUTION

EASY
ADVANCED



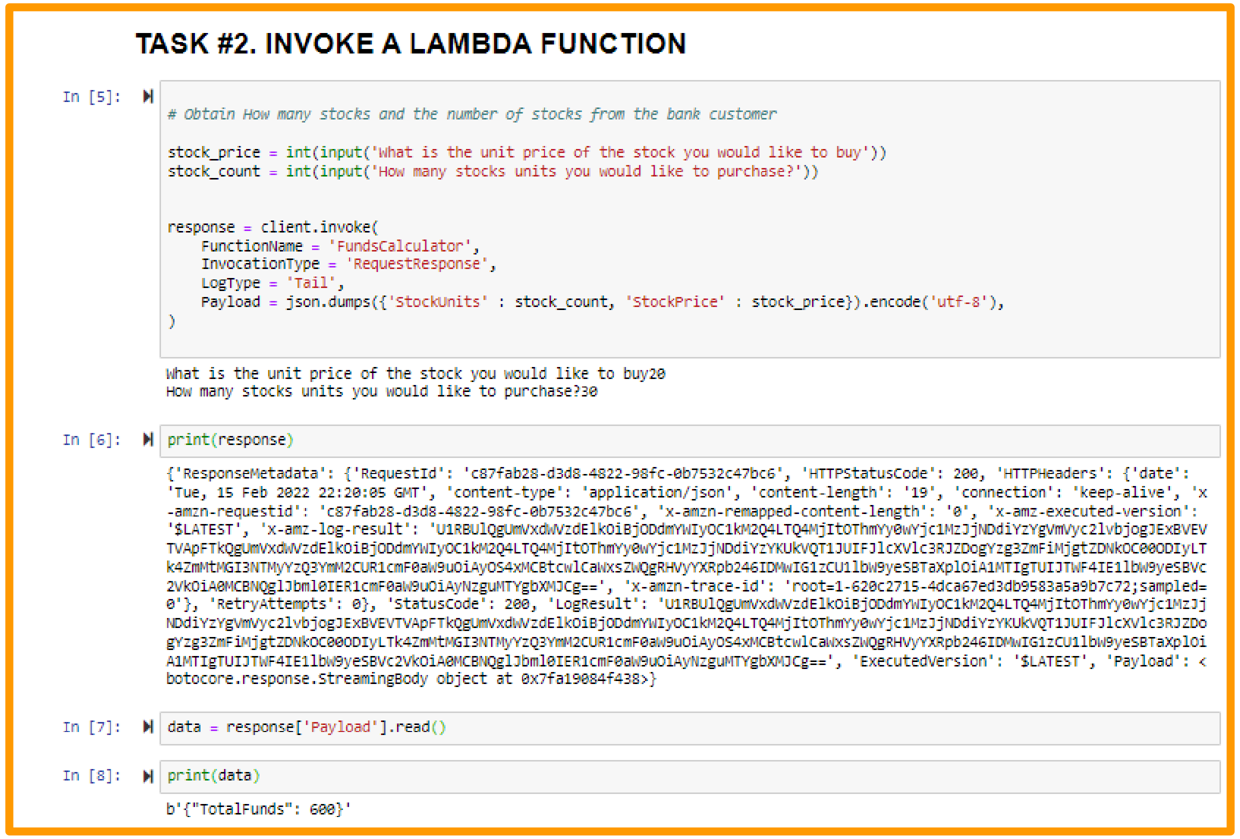
DEMO: LAMBDA INVOCATION WITH SDK



TASK #1. DEFINE A LAMBDA FUNCTION

```
In [1]:  %%writefile lambda_function.py

         import json
         import uuid

         def lambda_handler(event, context):

             # Read the input parameters
             count   = event['StockUnits']
             price   = event['StockPrice']

             # Calculate the total dollar value
             total =   count * price

             # Return the result
             return {
                 'TotalFunds' :   total
                   }

         Writing lambda_function.py
```

```
In [3]:  # Boto3 is the Amazon Web Services (AWS) Software Development Kit (SDK) for Python
         # Boto3 allows Python developer to write software that makes use of services like Amazon S3 and Amazon EC2
         import boto3
         import json
         client = boto3.client('lambda')

         # You must zip up the code of the lambda function at some point
         # so let's import ZipFile Module
         # This module provides tools to create, read, write, append, and list a ZIP file.
         from zipfile import ZipFile
         with ZipFile('lambda.zip', 'w') as f:
             f.write('lambda_function.py')

         # If submitting as a ZipFile, you need to insert raw data.
         with open('lambda.zip', 'rb') as f:
             zipped_code = f.read()

         # Note that SageMaker Notebooks don't have access to lambda by default
         # So you need to give permission to this SageMaker Notebook to access lambda
         response = client.create_function(
             FunctionName = 'FundsCalculator',
             Runtime = 'python3.8',
             Handler = 'lambda_function.lambda_handler',
             Code = dict(ZipFile=zipped_code),
             Timeout = 60,
             MemorySize = 512,
             Role = 'arn:aws:iam::422132866096:role/service-role/My-First-Lambda-role-ozpjvowf'
         )
```



TASK #2. INVOKE A LAMBDA FUNCTION

```
In [5]:  # Obtain How many stocks and the number of stocks from the bank customer

         stock_price = int(input('What is the unit price of the stock you would like to buy'))
         stock_count = int(input('How many stocks units you would like to purchase?'))


         response = client.invoke(
             FunctionName = 'FundsCalculator',
             InvocationType = 'RequestResponse',
             LogType = 'Tail',
             Payload = json.dumps({'StockUnits' : stock_count, 'StockPrice' : stock_price}).encode('utf-8'),
         )

         What is the unit price of the stock you would like to buy20
         How many stocks units you would like to purchase?30
```

```
In [6]:  print(response)

         {'ResponseMetadata': {'RequestId': 'c87fab28-d3d8-4822-98fc-0b7532c47bc6', 'HTTPStatusCode': 200, 'HTTPHeaders': {'date':
         'Tue, 15 Feb 2022 22:20:05 GMT', 'content-type': 'application/json', 'content-length': '19', 'connection': 'keep-alive', 'x
         -amzn-requestid': 'c87fab28-d3d8-4822-98fc-0b7532c47bc6', 'x-amzn-remapped-content-length': '0', 'x-amz-executed-version':
         '$LATEST', 'x-amz-log-result': 'U1RBUlQgUmVxdWVzdElkOiBjODdmYWIyOC1kM2Q4LTQ4MjItOThmYy0wYjc1MzJjNDdiYzYgVmVyc2lvbjogJExBVEV
         TVApFTkQgUmVxdWVzdElkOiBjODdmYWIyOC1kM2Q4LTQ4MjItOThmYy0wYjc1MzJjNDdiYzYKUkVQT1JUIFJlcXVlc3RJZDogYzg3ZmFiMjgtZDNkOC00ODIyLT
         k4ZmMtMGI3NTMyYzQ3YmM2CUR1cmF0aW9uOiAyOS4xMCBtcwlCaWxsZWQgRHVyYXRpb246IDMwIG1zCU1lbW9yeSBTaXplOiA1MTIgTUIJTWF4IE1lbW9yeSBVc
         2VkOiA0MCBNQglJbml0IER1cmF0aW9uOiAyNzguMTYgbXMJCg==', 'x-amzn-trace-id': 'root=1-620c2715-4dca67ed3db9583a5a9b7c72;sampled=
         0'}, 'RetryAttempts': 0}, 'StatusCode': 200, 'LogResult': 'U1RBUlQgUmVxdWVzdElkOiBjODdmYWIyOC1kM2Q4LTQ4MjItOThmYy0wYjc1MzJj
         NDdiYzYgVmVyc2lvbjogJExBVEVTVApFTkQgUmVxdWVzdElkOiBjODdmYWIyOC1kM2Q4LTQ4MjItOThmYy0wYjc1MzJjNDdiYzYKUkVQT1JUIFJlcXVlc3RJZDo
         gYzg3ZmFiMjgtZDNkOC00ODIyLTk4ZmMtMGI3NTMyYzQ3YmM2CUR1cmF0aW9uOiAyOS4xMCBtcwlCaWxsZWQgRHVyYXRpb246IDMwIG1zCU1lbW9yeSBTaXplOi
         A1MTIgTUIJTWF4IE1lbW9yeSBVc2VkOiA0MCBNQglJbml0IER1cmF0aW9uOiAyNzguMTYgbXMJCg==', 'ExecutedVersion': '$LATEST', 'Payload': <
         botocore.response.StreamingBody object at 0x7fa19084f438>}
```

```
In [7]:  data = response['Payload'].read()
```

```
In [8]:  print(data)

         b'{"TotalFunds": 600}'
```