BU.330.740 Large Scale Computing on the Cloud

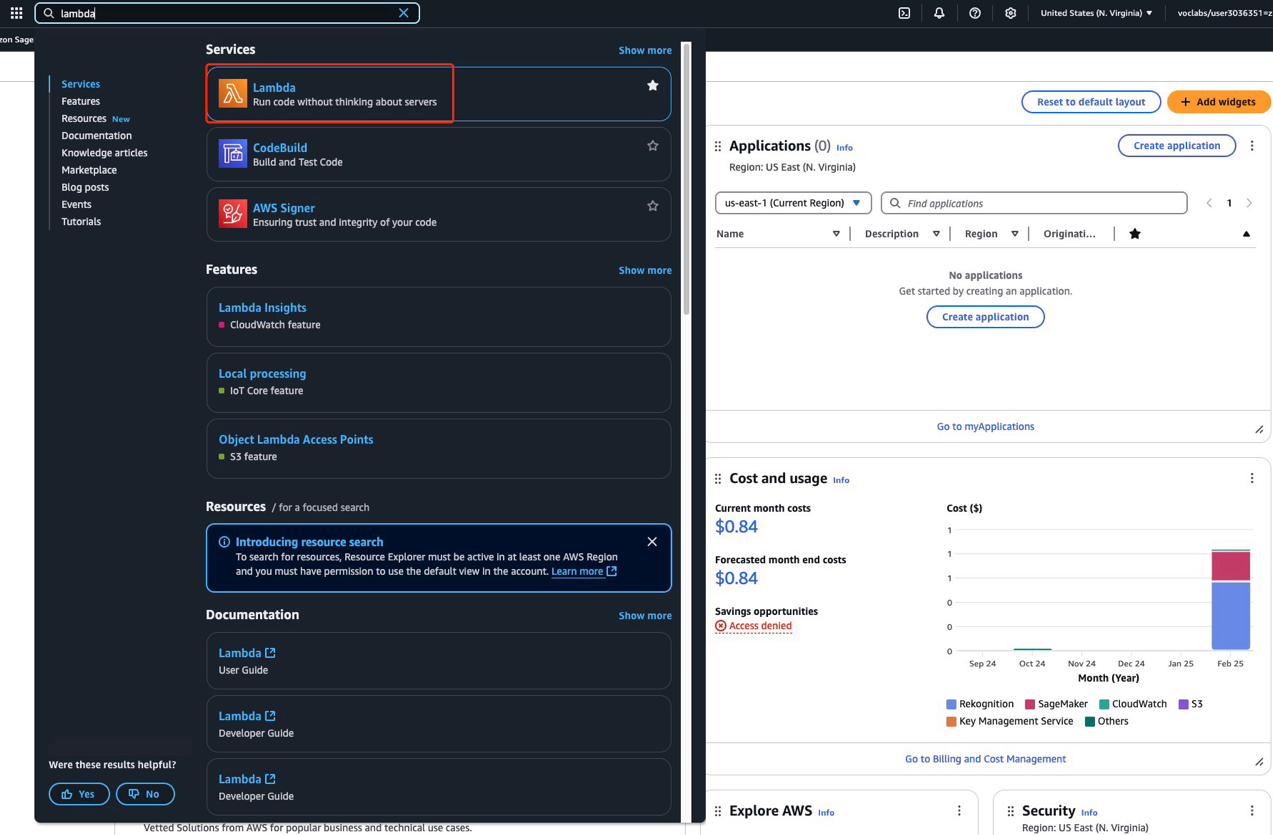
**Lab 5. AWS Lambda and Q Developer**

Learning Goal: In Lab 5, we practice using Q Developer to generate and modify AWS Lambda functions. The first task involves creating a Lambda function to validate email addresses using regular expressions and testing it with JSON input from Amazon Q. The second task focuses on writing a Lambda function to upload a text file to an S3 bucket, deploying it, and verifying the upload. This lab provides hands-on experience with Lambda, Q Developer, S3 integration, and AWS automation.

Login into your AWS Academy account at <https://awsacademy.instructure.com/login/canvas>. Then please follow “*AWS Academy Learner Lab – Student Guide.pdf”* provided by AWS to launch **AWS Management Console**.

**Task 1: Using Q Developer to Generate an AWS Lambda Function for Email Validation**

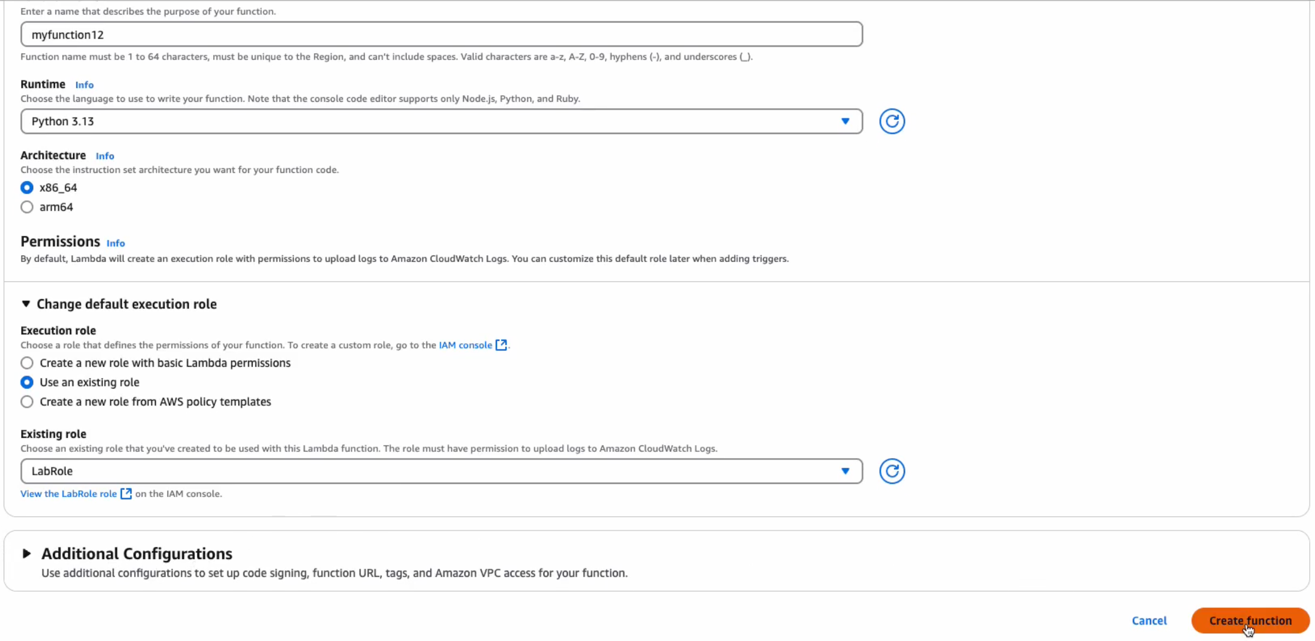
1. Create a Lambda function.
   1. Search for **Lambda** in **Search Bar**. Once you are on the Lambda page, click **Create Function**.



* 1. Enter a name for the function and refer to the screenshot below for the configuration settings. Select **Python 3.13** as the **Runtime** and **x86\_64** as the **Architecture**.

Please note that under **Execution Role**, you should select **Use an existing role** and choose **LabRole**.

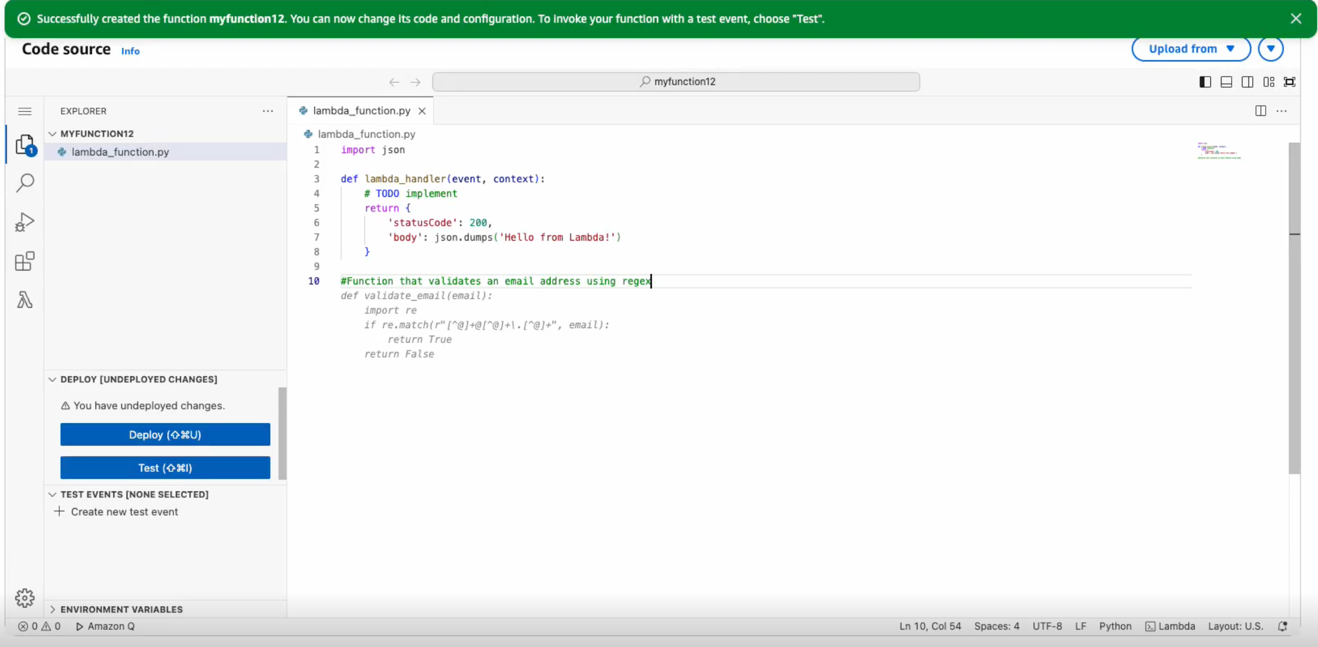
Once completed, click **Create Function**.



1. Edit Lambda Function using Q Developer. Scroll down to Code Source, and below the pre-generated lambda function, write:

**# Function that validates an email address using regex**

Once you wrote this input, Q Developer will automatically generate a function that aligns with the above input. Press the **tab** key on your computer to accept this output.



Remove line 5 – 8, and in the place of # TODO implement, write:

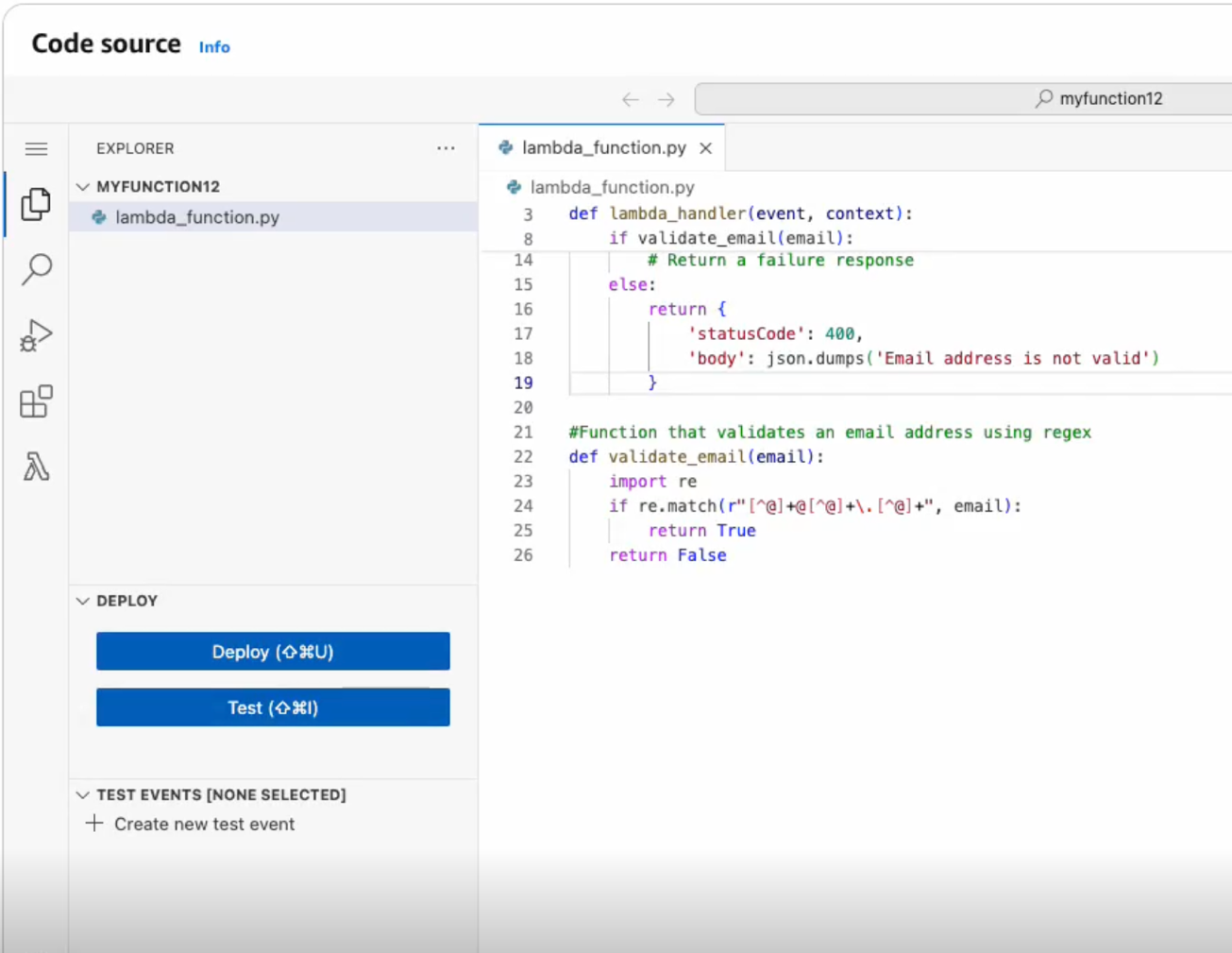
**# Get the email address from the event**

Continue to press the **tab** key multiple times until you accept all the code shown in the screenshot below.



1. Deploy the function.

Click **Deploy** on the left side of your code to deploy the updated the function.



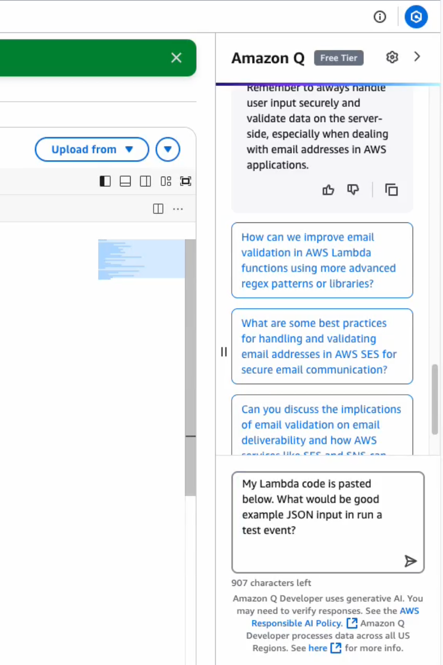
1. Generate a JSON input using Amazon Q.

On the upper right-hand corner of the page, click the icon shown in blue in the screenshot below to start a conversation with Amazon Q.

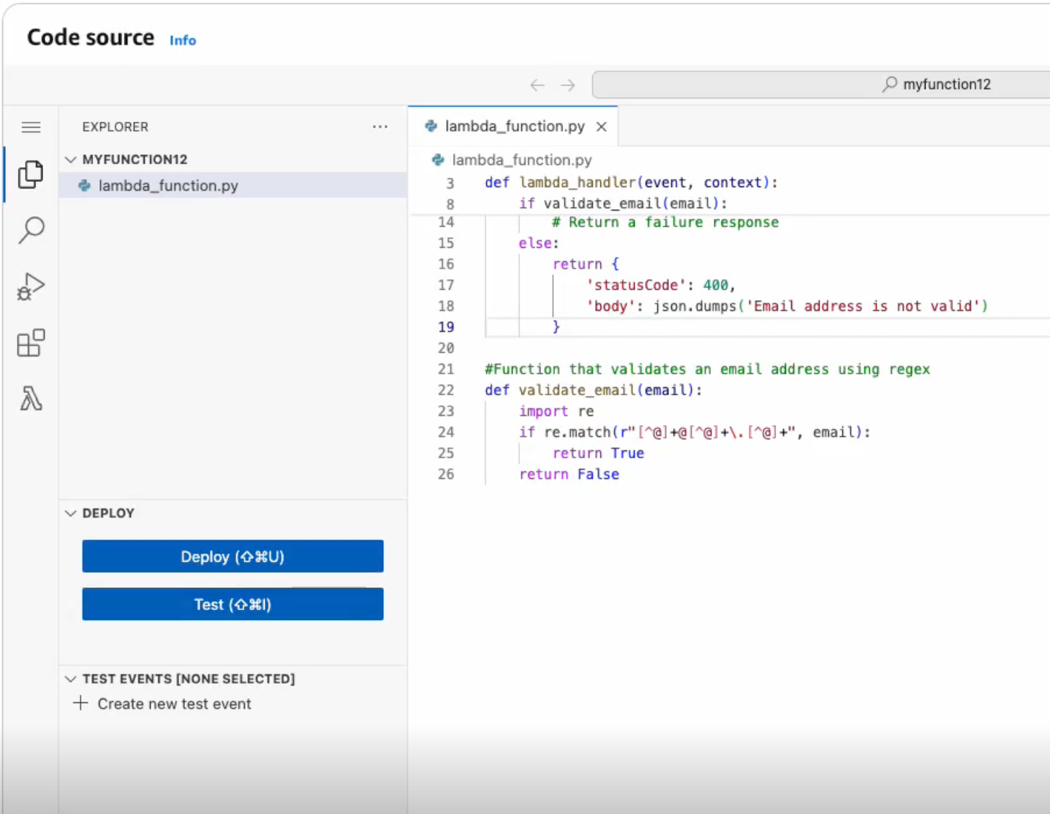
Write the following question and **paste the entire code** below this sentence:

**My Lambda code is pasted below. What would be good example JSON input to run a test event?**

Once a response is generated by Amazon Q, **copy** the JSON input that Amazon Q provides to you.



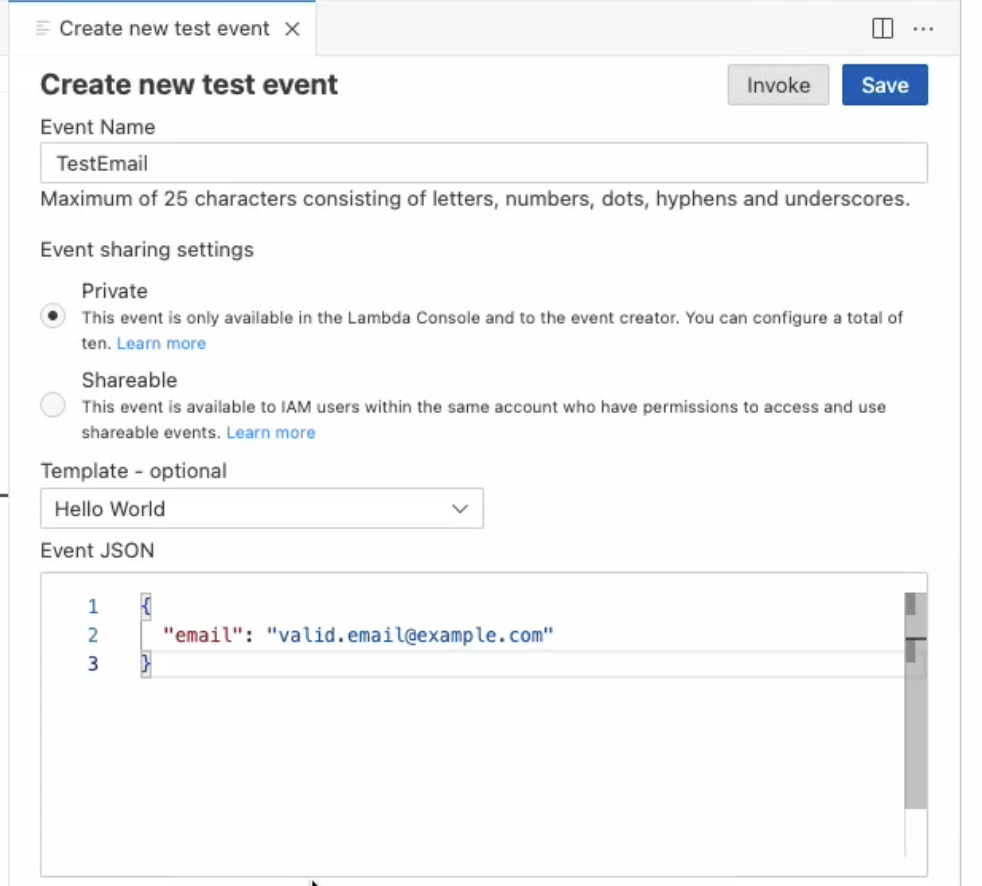
1. Test the function.

Click **Test** button below the Deploy button you clicked earlier. 

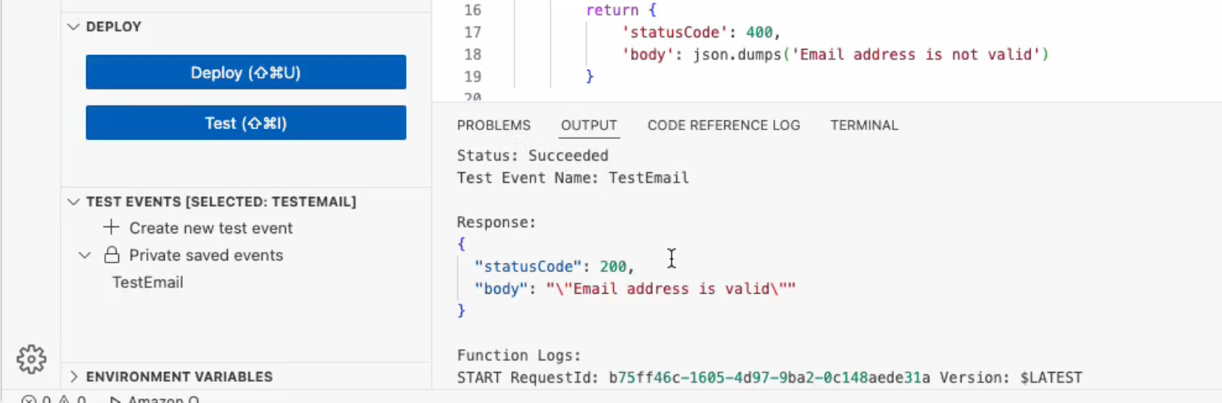
Click **Create new test event**.

Enter a name for your test event, and **paste** the **JSON code** you copied from Amazon Q.

Click **Save** to save the test.

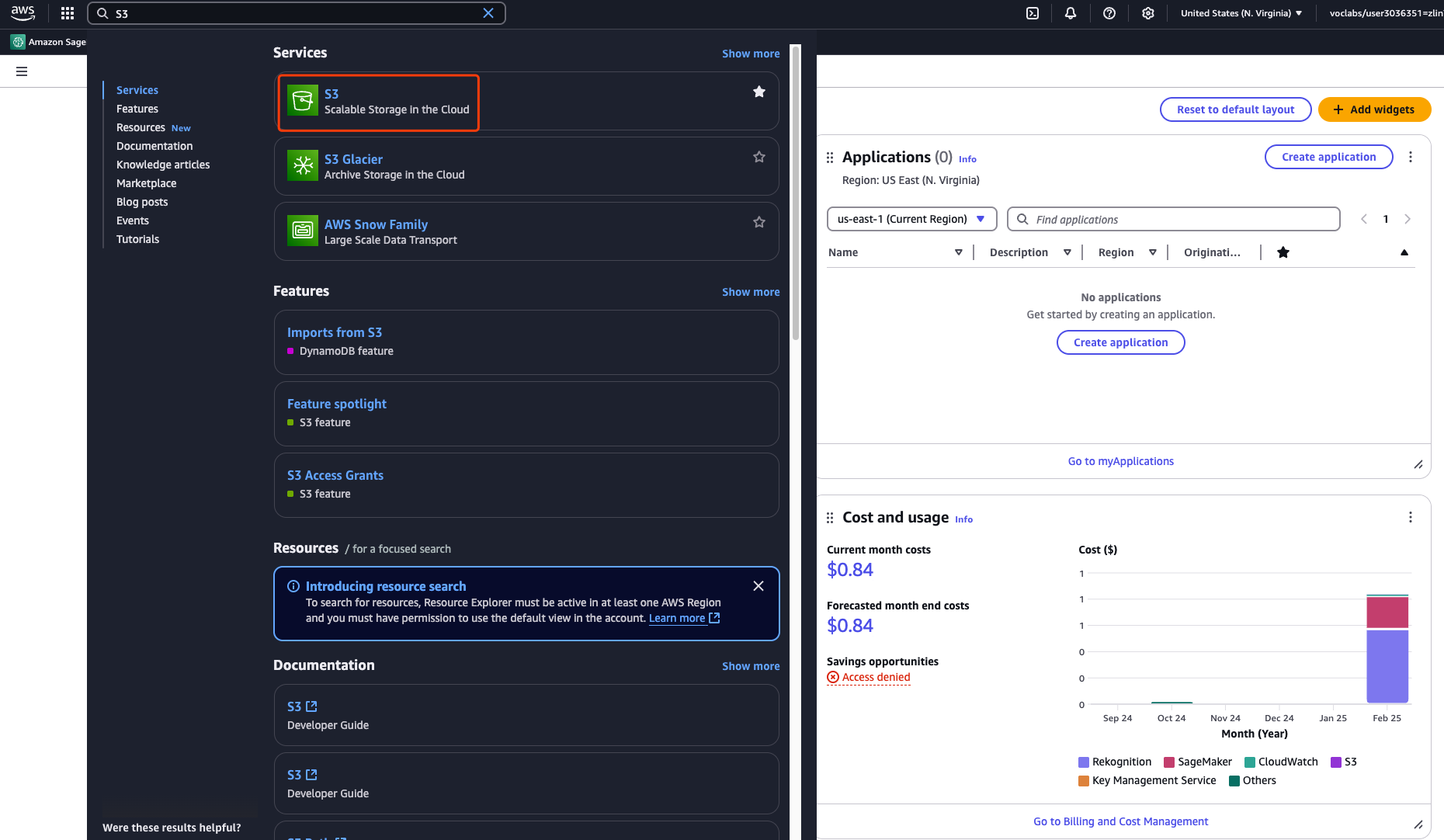


1. Run the test. Click **Invoke** to test the test event.

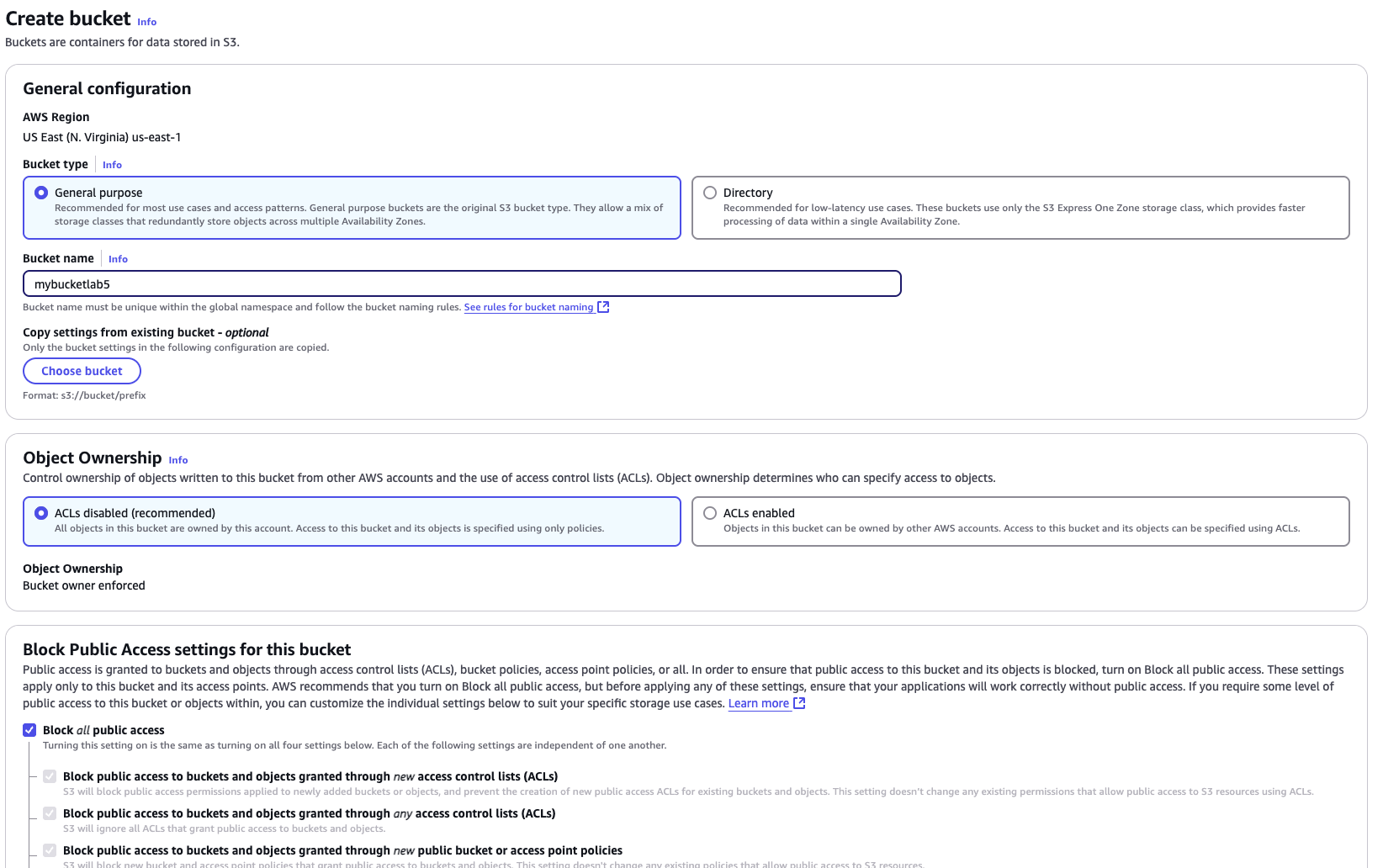
An output is generated based on the test, indicating whether the email is valid.

**Task 2: Using AWS Lambda to upload a file in S3 Bucket**

1. Create a S3 bucket. Search for **S3** in **Search Bar.** Once you are on the S3 page, click **Create bucket**.

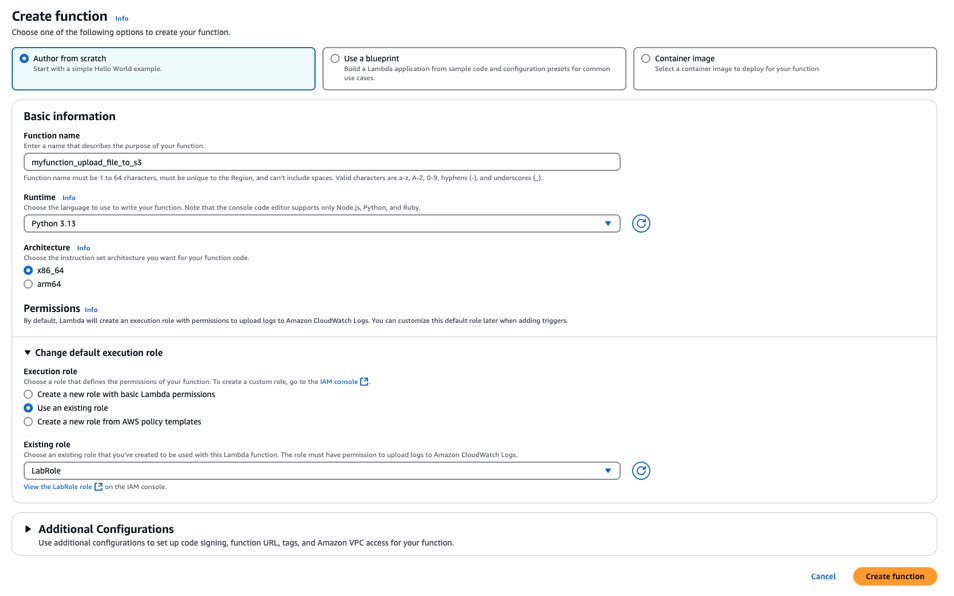


Enter a bucket name, keep all the default settings, and click **Create bucket**. Once the bucket is created, remember the bucket name, as you will use it in the code later.



1. Go to **Lambda** again by searching for **Lambda** in the search bar. Once on the **Lambda** page, click **Create function**. Enter a name, select **Python 3.13** as the **Runtime**, and choose **x86\_64** as the **Architecture**.

Please note that under **Execution Role**, you should select **Use an existing role** and choose **LabRole**.

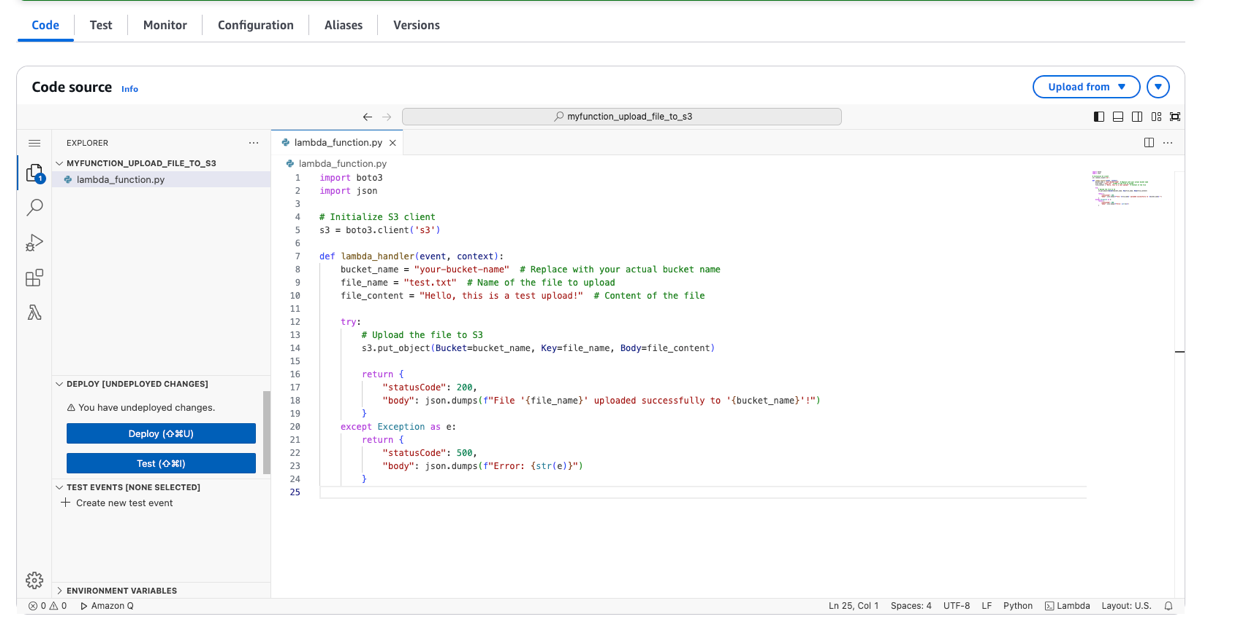
Once completed, click **Create function.**

Go to **Code source**, in the place of # TODO implement, write:

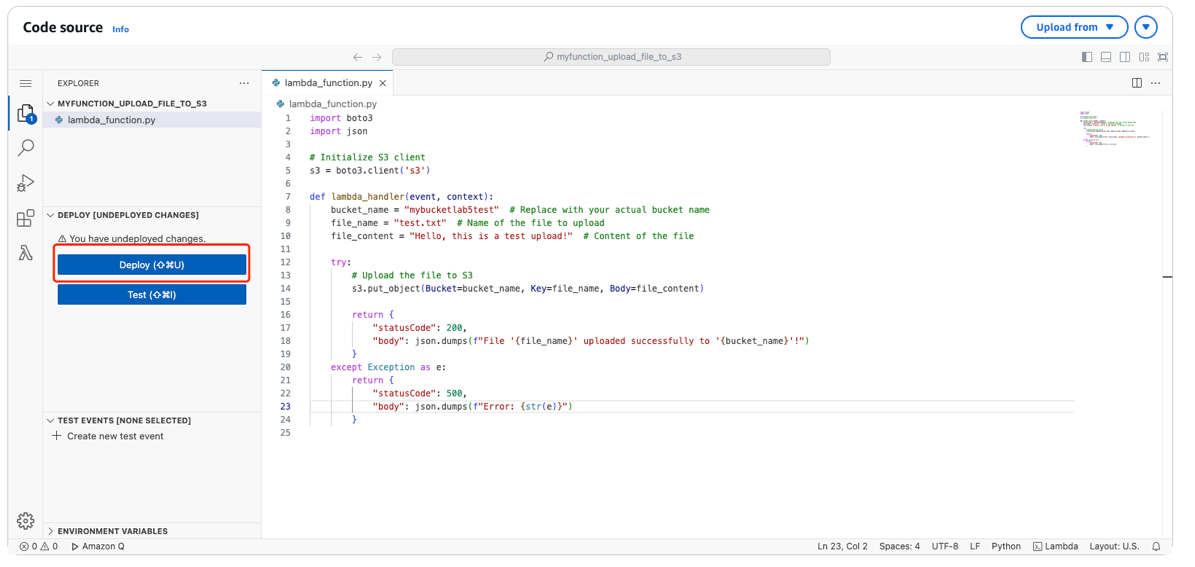
**# upload a file into an s3 bucket**

There are several ways that AWS Q Developer may suggest, such as:

|  |
| --- |
| import boto3  import json    # Initialize S3 client  s3 = boto3.client('s3')    def lambda\_handler(event, context):  bucket\_name = "your-bucket-name" # Replace with your actual bucket name  file\_name = "test.txt" # Name of the file to upload  file\_content = "Hello, this is a test upload!" # Content of the file    try:  # Upload the file to S3  s3.put\_object(Bucket=bucket\_name, Key=file\_name, Body=file\_content)    return {  "statusCode": 200,  "body": json.dumps(f"File '{file\_name}' uploaded successfully to '{bucket\_name}'!")  }  except Exception as e:  return {  "statusCode": 500,  "body": json.dumps(f"Error: {str(e)}")  } |



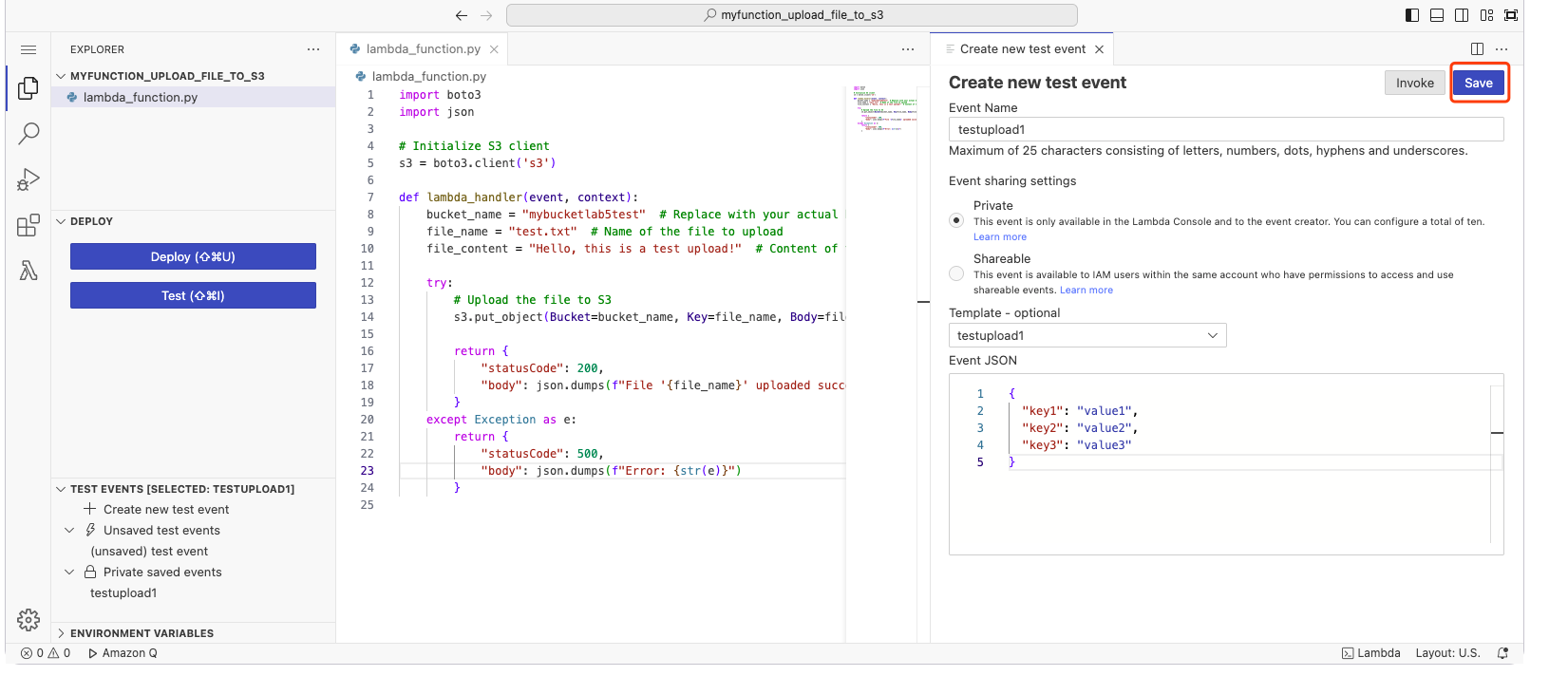
* 1. In the code, replace **"your-bucket-name"** with the name of the bucket you just created. Then click Deploy on the left.



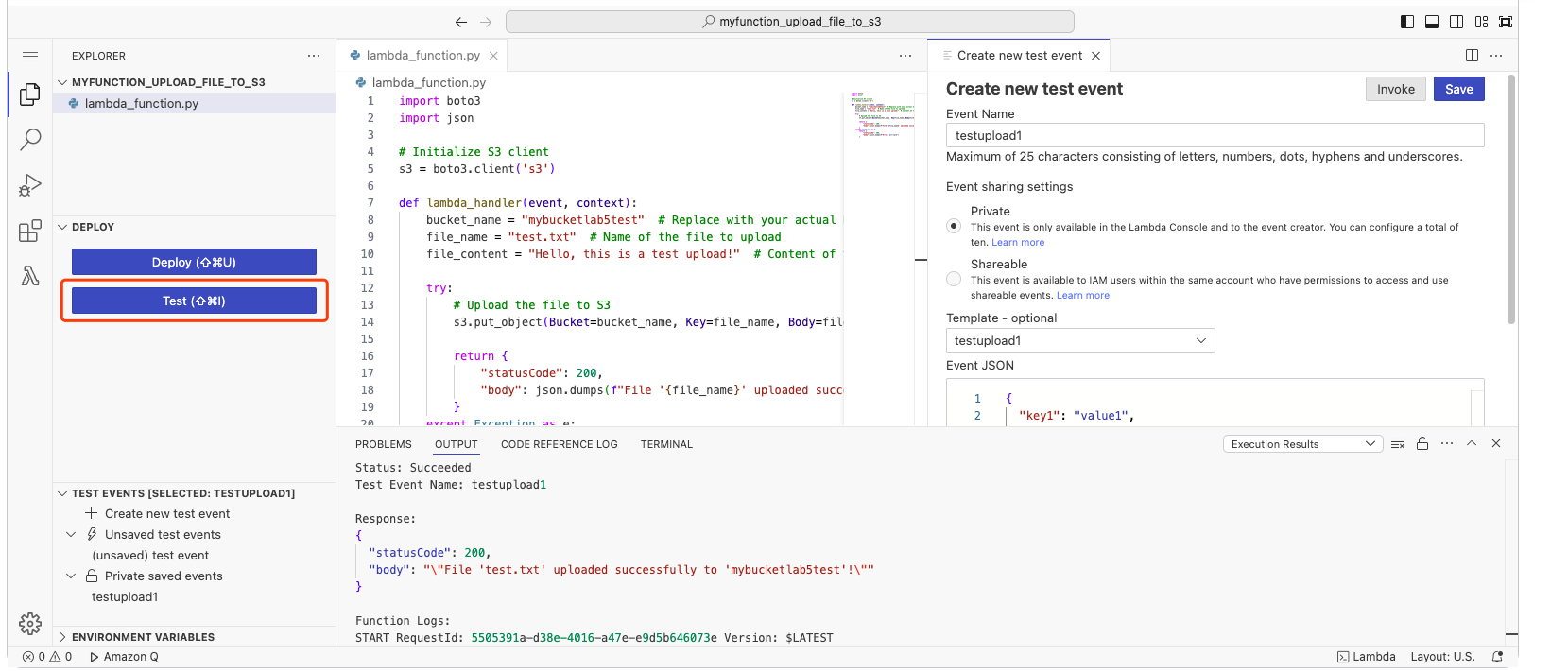
* 1. Once the function deployed, click **Test** on the left and then select **Create New Test**.



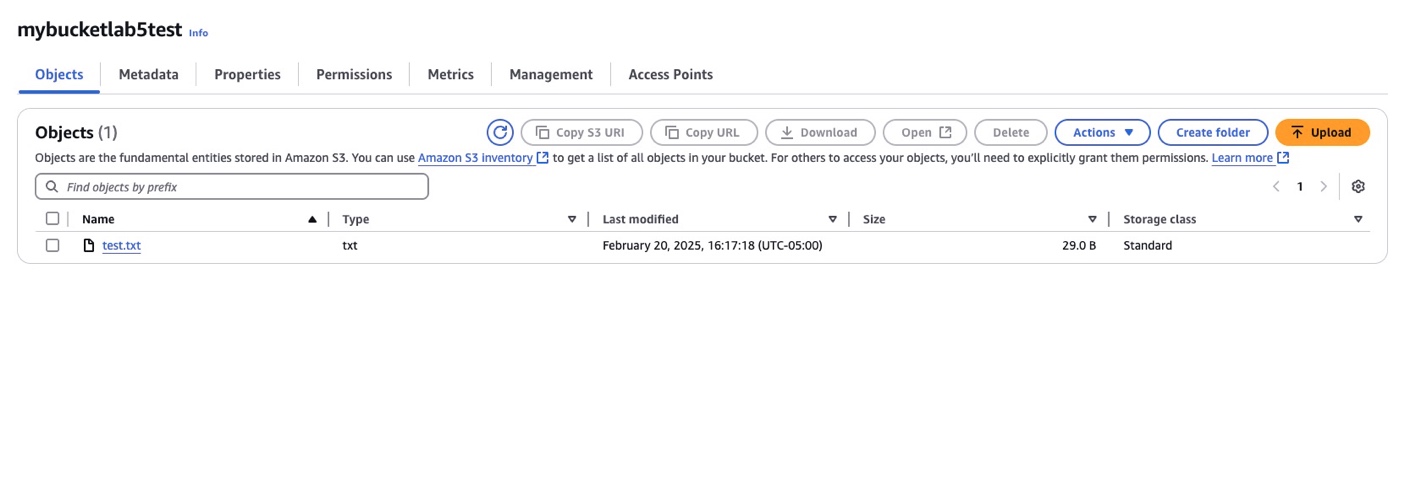
* 1. This time, we don’t change anything in the test. Simply enter an event name and click **Save**.



* 1. Click **Test** on the left again, and you will receive a message confirming that the **.txt** file has been uploaded to the bucket.



* 1. Go to the **S3 bucket** you created, and you will find that the **test.txt** file has been successfully uploaded.



Or another version:

import json

import boto3

def lambda\_handler(event, context):

# upload a file into an s3 bucket

s3 = boto3.client('s3')

s3\_resource = boto3.resource('s3')

bucket\_name = 'your\_bucket\_name'

file\_name = event['file\_name']

file\_content = event['file\_content']

try:

s3\_resource.Object(bucket\_name, file\_name).put(Body=file\_content)

return {

'statusCode': 200,

'body': json.dumps('File uploaded successfully into s3 bucket')

}

# print('File uploaded successfully into s3 bucket')

except FileNotFoundError:

print('The file was not found')

return {

'statusCode': 404,

'body': json.dumps('The file was not found')

}

You can write your own test case accordingly.

**Optional Task 3: Count the number of files in a S3 bucket**

You can test this task on your lab 4 bucket. One of the versions:

import json

import boto3

def lambda\_handler(event, context):

# get a S3 bucket name from event

bucket\_name = event['bucket']

# count the number of files in the bucket

s3 = boto3.client('s3')

response = s3.list\_objects\_v2(Bucket=bucket\_name)

file\_count = len(response['Contents'])

# return the result

return {

'statusCode': 200,

'body': json.dumps(f'The number of files in the bucket "{bucket\_name}" is {file\_count}.')

}

**Reference**: <https://docs.aws.amazon.com/lambda/latest/dg/getting-started.html>