AWS

1. Write an AWS Lambda function that adds two numbers and returns the result.


```python
def lambda_handler(event, context):
    try:
        # Retrieve inputs
        num1 = event.get('num1')
        num2 = event.get('num2')

        # Validate inputs
        if num1 is None or num2 is None:
            return {
                'statusCode': 400,
                'body': "Error: Both 'num1' and 'num2' are required in the input."
            }

        # Ensure inputs are numbers
        if not isinstance(num1, (int, float)) or not isinstance(num2, (int, float)):
            return {
                'statusCode': 400,
                'body': "Error: 'num1' and 'num2' must be numbers."
            }
```

```python
21          # Perform addition
22          result = num1 + num2
23
24          # Return success response
25          return {
26              'statusCode': 200,
27              'body': f"The sum of {num1} and {num2} is {result}."
28          }
29      except Exception as e:
30          # Catch unexpected errors
31          return {
32              'statusCode': 500,
33              'body': f"An unexpected error occurred: {str(e)}"
34          }
35
```

DEPLOY

Deploy (Ctrl+Shift+U)

Test (Ctrl+Shift+I)

TEST EVENTS [SELECTED: TEST_ADD_TWO_NUM_]

+ Create new test event

Shareable saved events

test_add_two_num

test_add

test_add_two_num_

ENVIRONMENT VARIABLES

PROBLEMS   OUTPUT   CODE REFERENCE LOG   TERMINAL

Execution Results

Status: Succeeded
Test Event Name: test_add_two_num_

Lambda > Functions > Addtwonumber

Successfully updated the function **Addtwonumber**.

Event name

test_add_two_num

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

```json
1  {
2      "num1": 5,
3      "num2": 7
4  }
5
```

Code source Info

Upload from ▼

← →   🔍 Addtwonumber

```
lambda_function.py ×

📄 lambda_function.py
  1    def lambda_handler(event, context):
```

| PROBLEMS | OUTPUT | CODE REFERENCE LOG | TERMINAL |

Execution Results ▼

```
Status: Succeeded
Test Event Name: test_add_two_num_

Response:
{
  "statusCode": 200,
  "body": "The sum of 5 and 7 is 12."
}

Function Logs:
START RequestId: 5fc432ce-3707-4bad-86d9-ca273b2ad0da Version: $LATEST
END RequestId: 5fc432ce-3707-4bad-86d9-ca273b2ad0da
REPORT RequestId: 5fc432ce-3707-4bad-86d9-ca273b2ad0da  Duration: 2.77 ms   Billed Duration: 3 ms   Memory Size: 128 MB Max Memory Used: 31
MB  Init Duration: 79.45 ms

Request ID: 5fc432ce-3707-4bad-86d9-ca273b2ad0da
```

EXPLORER

ADDTWONUMBER
- add_two_number.py
- lambda_function.py

DEPLOY

Deploy (Ctrl+Shift+U)

Test (Ctrl+Shift+I)

TEST EVENTS [SELECTED: TEST_ADD_TWO_NUM_]
- ➕ Create new test event
- Shareable saved events

2. Code an AWS Lambda function to store a document or PDF file in an S3 bucket

Lambda > Functions > UploadToS3Function

Successfully updated the function **UploadToS3Function**.

lambda_function.py

```python
import json
import base64
import boto3

def lambda_handler(event, context):
    try:
        # Get the file name and content from the event body
        file_name = event['file_name']
        file_content = event['file_content']

        # Decode the base64 file content
        file_data = base64.b64decode(file_content)

        # Create an S3 client
        s3 = boto3.client('s3')

        # Upload the file to S3
        s3.put_object(Bucket='storedocumentins3', Key=file_name, Body=file_data)
```

PROBLEMS   OUTPUT   CODE REFERENCE LOG   TERMINAL   Execution Results

Status: Succeeded
Test Event Name: testpdf

Lambda > Functions > UploadToS3Function

Successfully updated the function **UploadToS3Function**.

lambda_function.py

```python
def lambda_handler(event, context):
    try:

        # Upload the file to S3
        s3.put_object(Bucket='storedocumentins3', Key=file_name, Body=file_data)

        # Return success response
        return {
            'statusCode': 200,
            'body': json.dumps('File uploaded successfully')
        }

    except Exception as e:
        return {
            'statusCode': 500,
            'body': json.dumps(f"Error: {str(e)}")
        }
```

PROBLEMS   OUTPUT   CODE REFERENCE LOG   TERMINAL   Execution Results

Status: Succeeded
Test Event Name: testpdf

us-east-1.console.aws.amazon.com/lambda/home?region=us-east-1#/functions/UploadToS3Function?tab=code

aws

Search [Alt+S]

United States (N. Virginia) ▼   Prafull%20Sutar ▼

Lambda > Functions > UploadToS3Function

✓ Successfully updated the function **UploadToS3Function**.   ✕

EXPLORER   lambda_function.py ✕

∨ UPLOADTOS3FUNCTION
    lambda_function.py

```
5    def lambda_handler(event, context):
6        try:
```

PROBLEMS   OUTPUT   CODE REFERENCE LOG   TERMINAL   Execution Results ▼

∨ DEPLOY

Deploy (Ctrl+Shift+U)

Test (Ctrl+Shift+I)

Status: Succeeded
Test Event Name: testpdf

Response:
```
{
  "statusCode": 200,
  "body": "\"File uploaded successfully\""
}
```

Function Logs:
START RequestId: 2983e303-2760-49d5-9e72-c9f3c7c3f9d5 Version: $LATEST
END RequestId: 2983e303-2760-49d5-9e72-c9f3c7c3f9d5
REPORT RequestId: 2983e303-2760-49d5-9e72-c9f3c7c3f9d5 Duration: 2863.88 ms Billed Duration: 2864 ms Memory Size: 128 MB Max Memory

∨ TEST EVENTS [SELECTED: TESTPDF]
    + Create new test event
    ∨ [☐] Shareable saved events
        testpdf

Used: 81 MB Init Duration: 289.47 ms

Request ID: 2983e303-2760-49d5-9e72-c9f3c7c3f9d5

∨ ENVIRONMENT VARIABLES
    ▷ Amazon Q

Ln 18, Col 9   Spaces: 4   UTF-8   LF   Python   Lambda   Layout: U