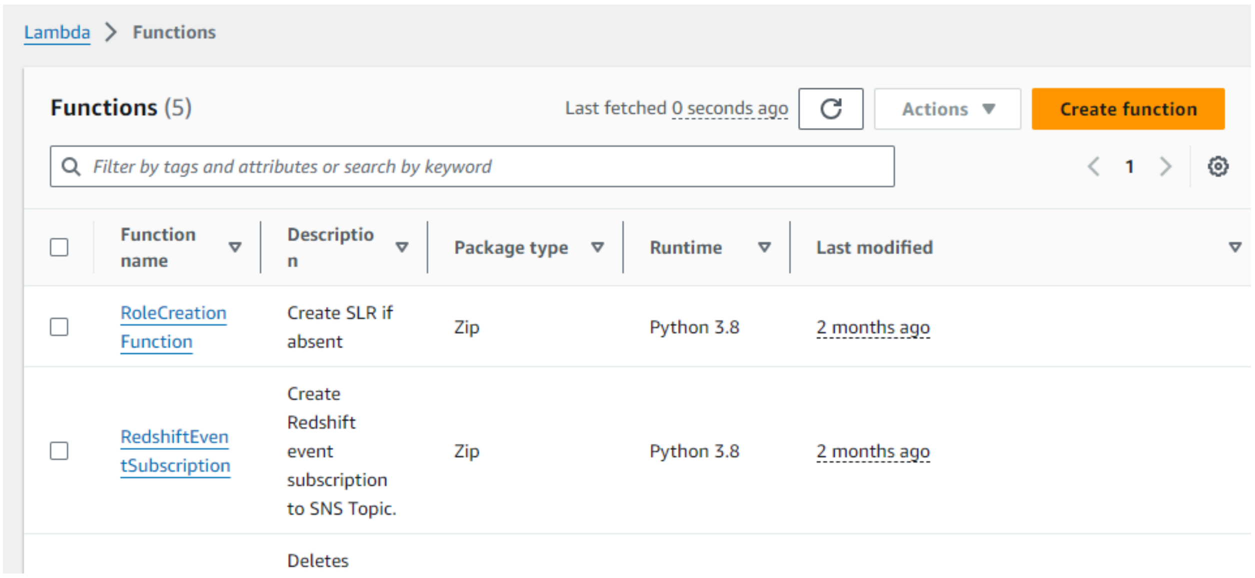
Name: Alok Yadav Div: D15C Roll No. 59

EXPERIMENT NO.11

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

Steps to create Lambda function in AWS :

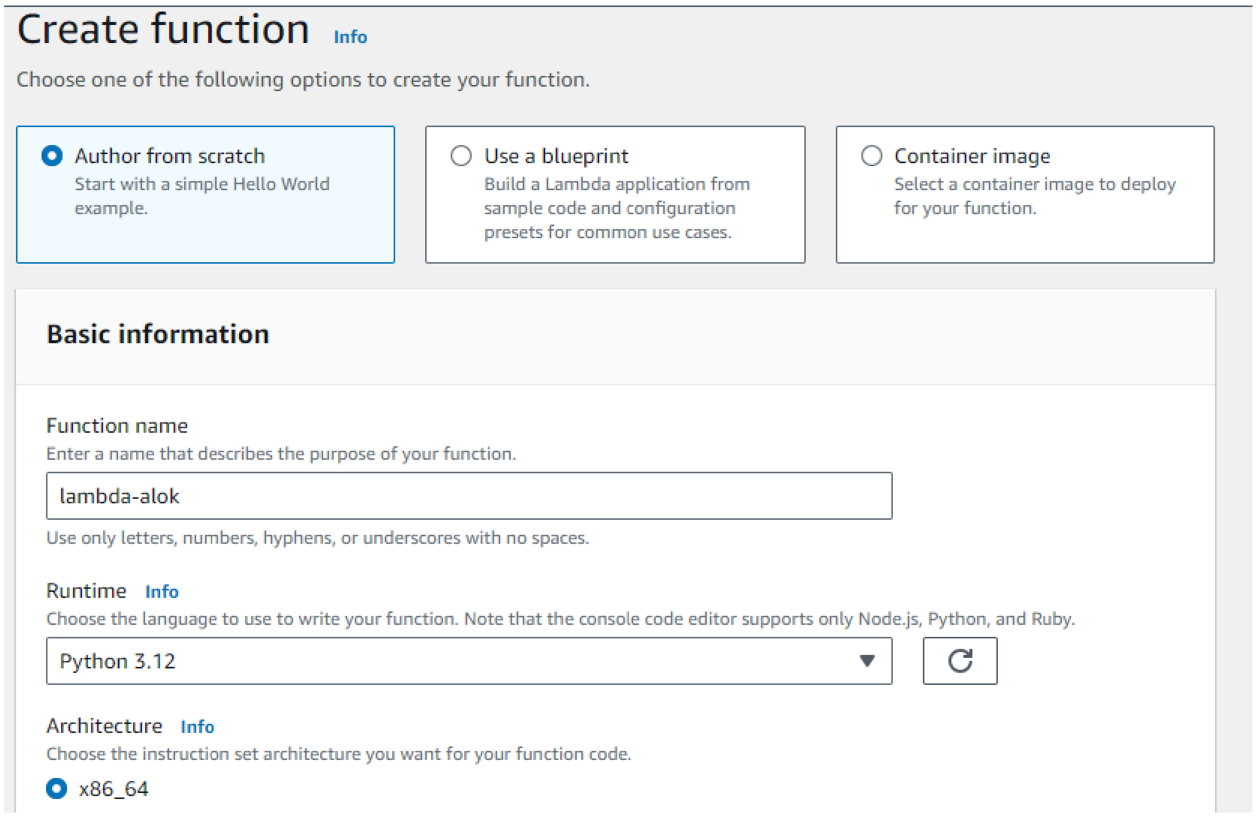
1. Open up the Lambda Console and click on the Create button.
   Be mindful of where you create your functions since Lambda is region-dependent.


Lambda > Functions
Functions (5)
Last fetched 0 seconds ago
Actions
Create function
Filter by tags and attributes or search by keyword
1

Function name	Description	Package type	Runtime	Last modified
RoleCreationFunction	Create SLR if absent	Zip	Python 3.8	2 months ago
RedshiftEventSubscription	Create Redshift event subscription to SNS Topic.	Zip	Python 3.8	2 months ago
	Deletes			

2. You can either create a function from scratch or select a blueprint, which is a pre-defined template by AWS that includes configuration settings for common use cases. Next, choose a runtime environment for your function; the dropdown will display all supported options, with Python, Node.js, .NET, and Java being the most popular. Finally, if you don't have an existing role, opt to create a new role with basic Lambda permissions.

Name: Alok Yadav   Div: D15C   Roll No. 59



Create function Info

Choose one of the following options to create your function.

[✓ Author from scratch] Start with a simple Hello World example.

Use a blueprint — Build a Lambda application from sample code and configuration presets for common use cases.

Container image — Select a container image to deploy for your function.

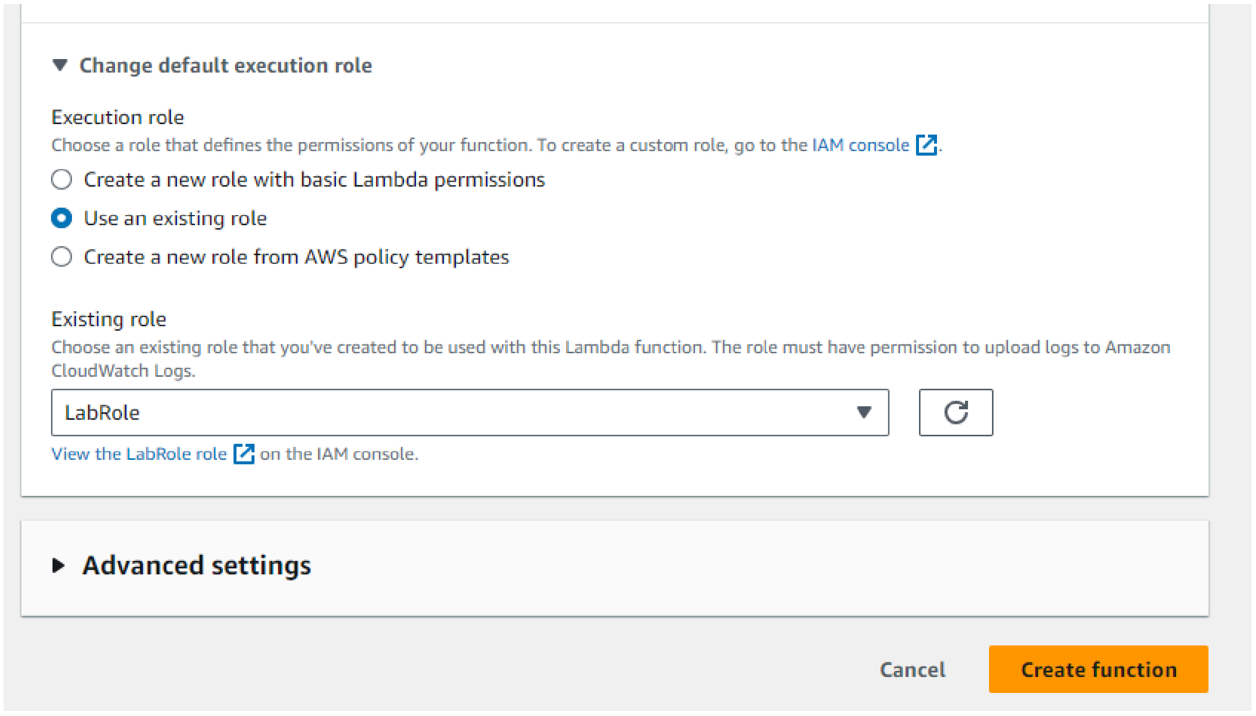
Basic information

Function name
Enter a name that describes the purpose of your function.
lambda-alok
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]

Select proper Execution role



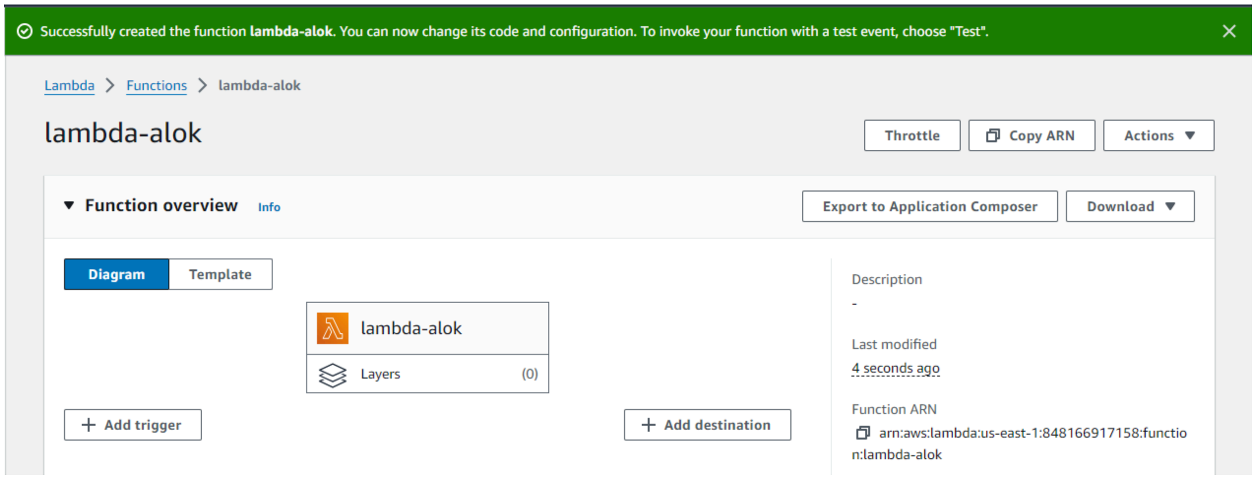
Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.

Create a new role with basic Lambda permissions
[✓ Use an existing role]
Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
LabRole
View the LabRole role on the IAM console.

Name: Alok Yadav Div: D15C Roll No. 59



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

Lambda > Functions > lambda-alok

lambda-alok

Function overview

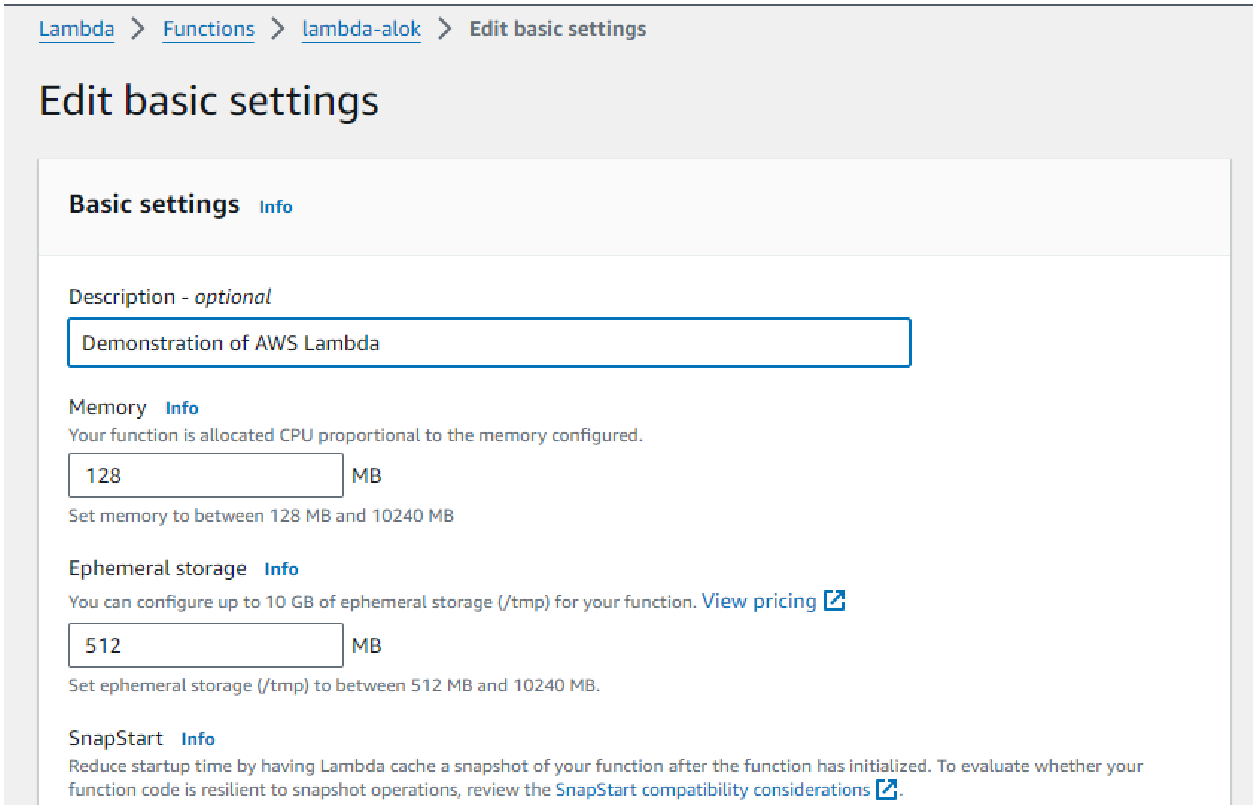
Description
-

Last modified
4 seconds ago

Function ARN
arn:aws:lambda:us-east-1:848166917158:function:lambda-alok

Successfully created Lambda function

3. To view or change the basic settings, go to the 'Configuration' tab and click 'Edit' under 'General settings.' (THIS STEP IS OPTIONAL)



Lambda > Functions > lambda-alok > Edit basic settings

Edit basic settings

Basic settings

Description - *optional*
Demonstration of AWS Lambda

Memory
Your function is allocated CPU proportional to the memory configured.
128 MB
Set memory to between 128 MB and 10240 MB

Ephemeral storage
You can configure up to 10 GB of ephemeral storage (/tmp) for your function. View pricing
512 MB
Set ephemeral storage (/tmp) to between 512 MB and 10240 MB.

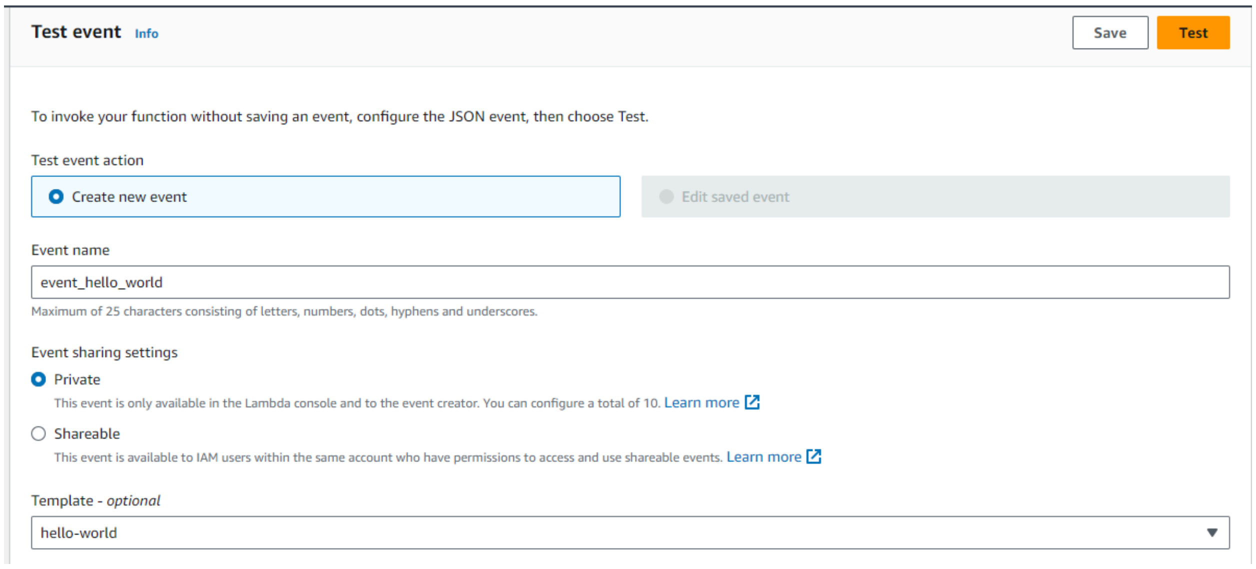
SnapStart
Reduce startup time by having Lambda cache a snapshot of your function after the function has initialized. To evaluate whether your function code is resilient to snapshot operations, review the SnapStart compatibility considerations.

Name: Alok Yadav Div: D15C Roll No. 59

Added description.

4. Go to the 'Test' tab and click 'Create a new event.' Give the event a name, set 'Event Sharing' to private, and choose the 'hello-world' template.

   We can create a new event to test and validate your Lambda function. By setting Event Sharing to private, we ensure its security, and selecting the "hello-world" template offers a straightforward framework for testing with minimal input complexity.



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

event_hello_world

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

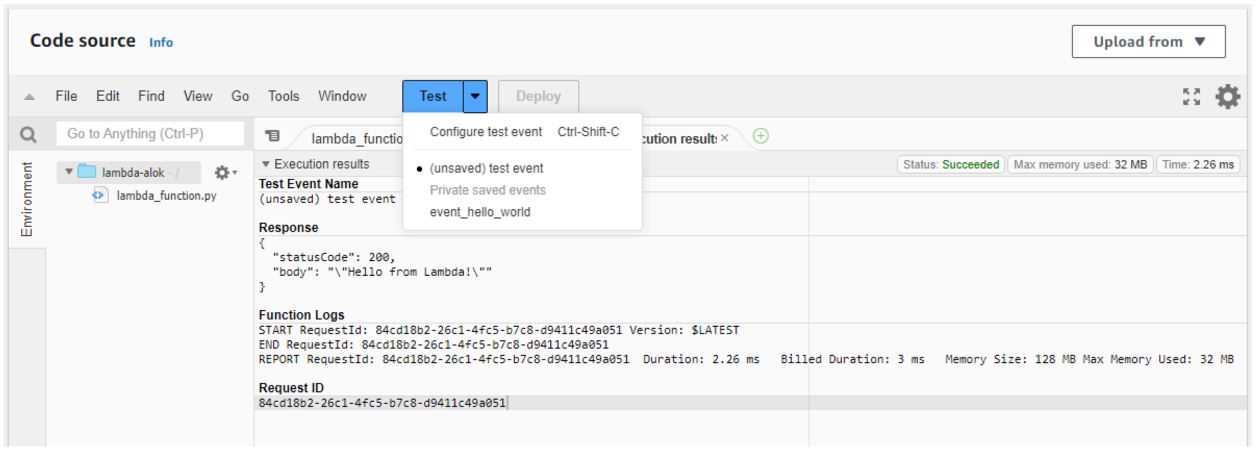
Event sharing settings

- [✓ 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

5. In the Code section, select the event you created from the dropdown menu under 'Test,' then click 'Test.' You should see the output below."



Execution results

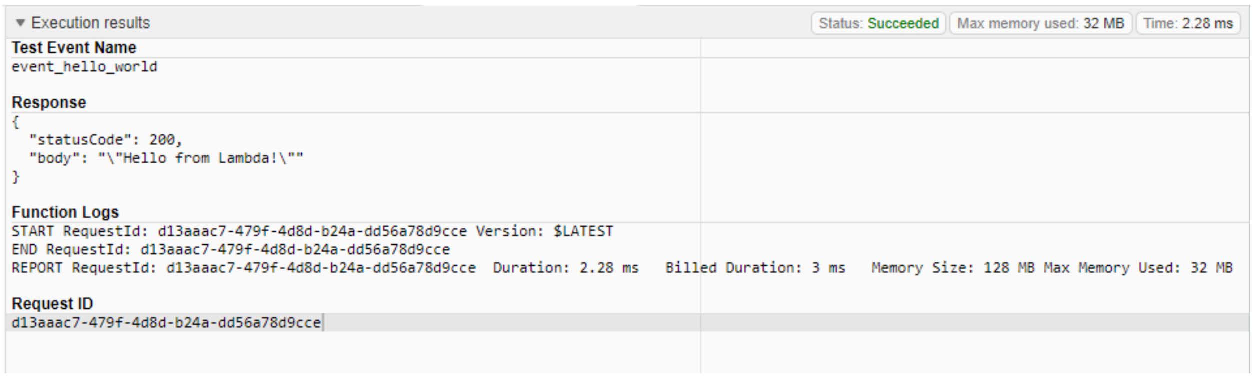
Status: Succeeded | Max memory used: 32 MB | Time: 2.26 ms

Test Event Name
```
(unsaved) test event
```

Response
```
{
  "statusCode": 200,
  "body": "\"Hello from Lambda!\""
}
```

Function Logs
```
START RequestId: 84cd18b2-26c1-4fc5-b7c8-d9411c49a051 Version: $LATEST
END RequestId: 84cd18b2-26c1-4fc5-b7c8-d9411c49a051
REPORT RequestId: 84cd18b2-26c1-4fc5-b7c8-d9411c49a051  Duration: 2.26 ms   Billed Duration: 3 ms   Memory Size: 128 MB Max Memory Used: 32 MB
```

Request ID
```
84cd18b2-26c1-4fc5-b7c8-d9411c49a051
```



Execution results — Status: Succeeded | Max memory used: 32 MB | Time: 2.28 ms

Test Event Name
```
event_hello_world
```

Response
```
{
  "statusCode": 200,
  "body": "\"Hello from Lambda!\""
}
```

Function Logs
```
START RequestId: d13aaac7-479f-4d8d-b24a-dd56a78d9cce Version: $LATEST
END RequestId: d13aaac7-479f-4d8d-b24a-dd56a78d9cce
REPORT RequestId: d13aaac7-479f-4d8d-b24a-dd56a78d9cce  Duration: 2.28 ms   Billed Duration: 3 ms   Memory Size: 128 MB Max Memory Used: 32 MB
```

Request ID
```
d13aaac7-479f-4d8d-b24a-dd56a78d9cce
```

Got above output from Lambda function

6. You can modify your Lambda function code. I've updated it to display the current Date and Time of AWS Server. After making your changes, save them using Control + S and then click on Deploy.

   Function to get current Date and Time of Server:

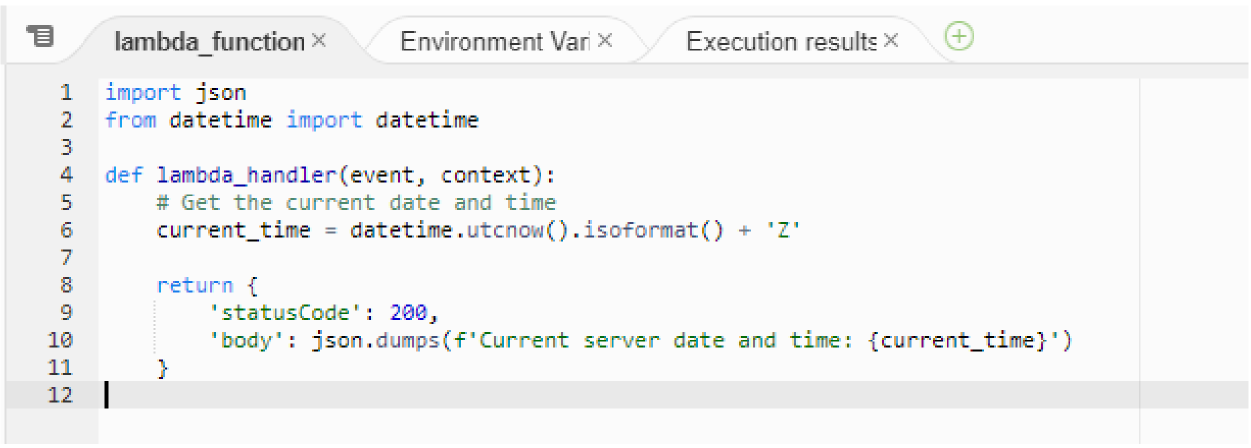
```
import json
from datetime import datetime

def lambda_handler(event, context):
   # Get the current date and time
   current_time = datetime.utcnow().isoformat() + 'Z'

   return {
      'statusCode': 200,
      'body': json.dumps(f'Current server date and time: {current_time}')
   }
```

   Note: After making change in lambda function, ensure you deploy it before running test

Name: Alok Yadav Div: D15C Roll No. 59

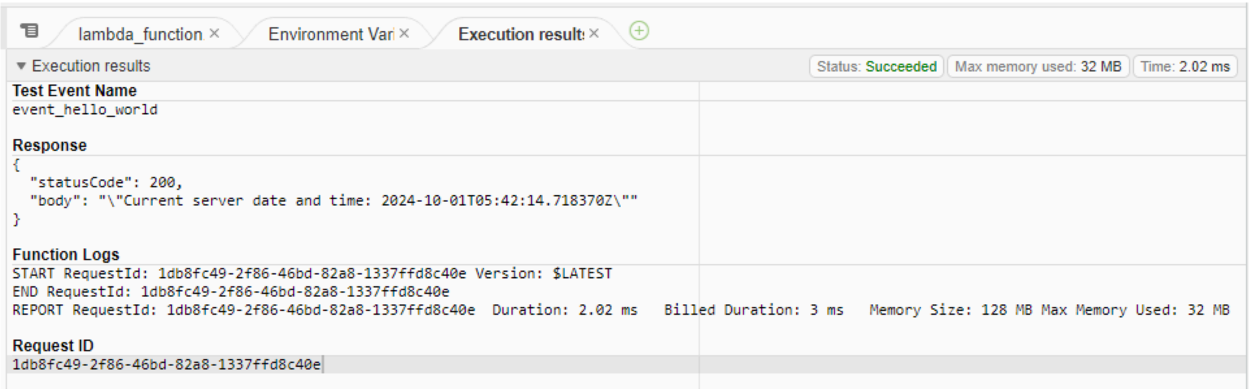


```python
import json
from datetime import datetime

def lambda_handler(event, context):
    # Get the current date and time
    current_time = datetime.utcnow().isoformat() + 'Z'

    return {
        'statusCode': 200,
        'body': json.dumps(f'Current server date and time: {current_time}')
    }

```



Status: Succeeded | Max memory used: 32 MB | Time: 2.02 ms

Test Event Name

```
event_hello_world
```

Response

```
{
  "statusCode": 200,
  "body": "\"Current server date and time: 2024-10-01T05:42:14.718370Z\""
}
```

Function Logs

```
START RequestId: 1db8fc49-2f86-46bd-82a8-1337ffd8c40e Version: $LATEST
END RequestId: 1db8fc49-2f86-46bd-82a8-1337ffd8c40e
REPORT RequestId: 1db8fc49-2f86-46bd-82a8-1337ffd8c40e  Duration: 2.02 ms   Billed Duration: 3 ms   Memory Size: 128 MB Max Memory Used: 32 MB
```

Request ID

```
1db8fc49-2f86-46bd-82a8-1337ffd8c40e
```

After changing the lambda function, we are getting Date and Time from the Server.

Conclusion:

In conclusion, I conducted an experiment using the **Hello World** template and achieved the expected results. Initially, I encountered issues with my custom Lambda function, which was designed to retrieve the server's date and time, due to it not being deployed. After deploying the function, it executed correctly and returned the desired output. This experience underscored the critical importance of deploying changes to ensure they are properly reflected and take effect in production.