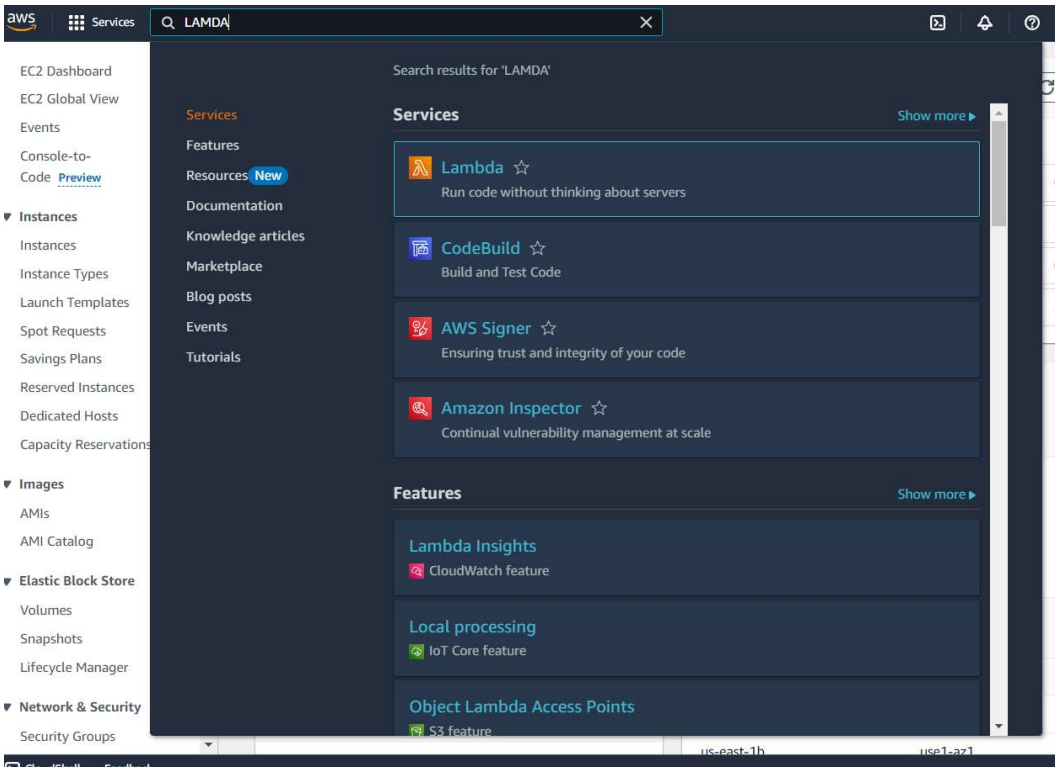
Name:Atharva Patil **Std:D15C** **ROLL:39**

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

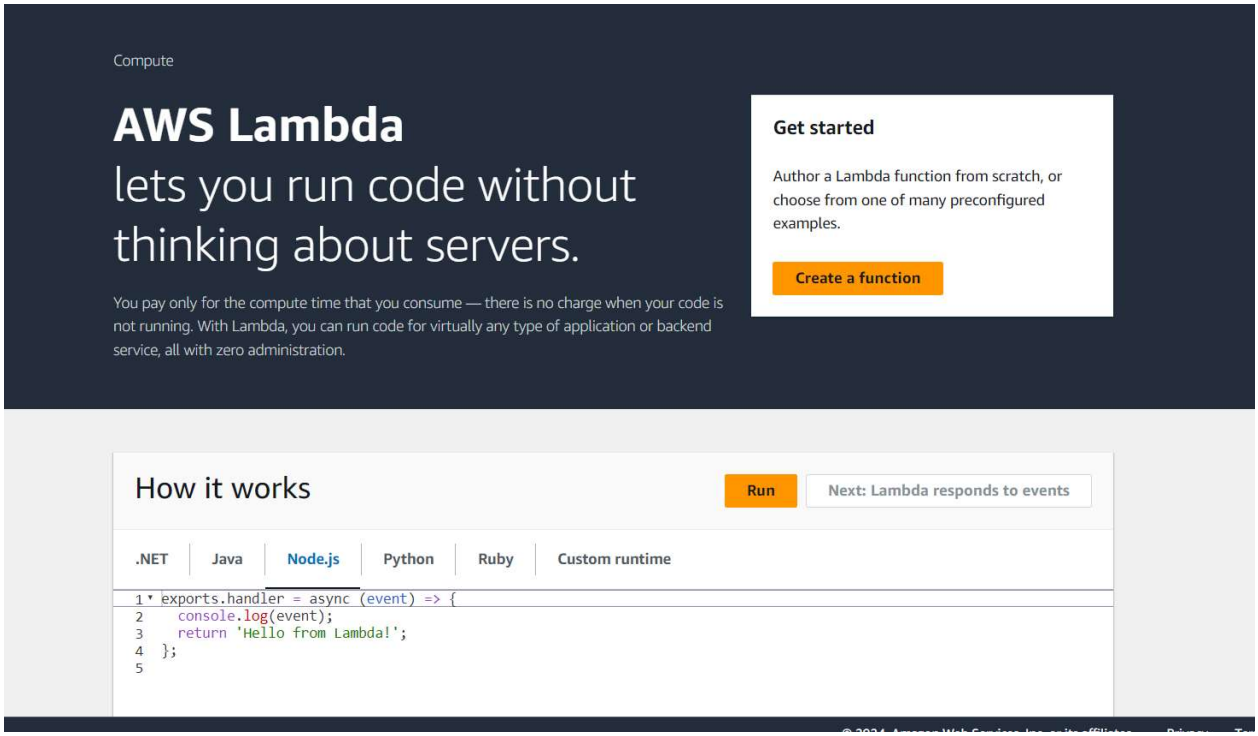
Step 1: Accessing AWS

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



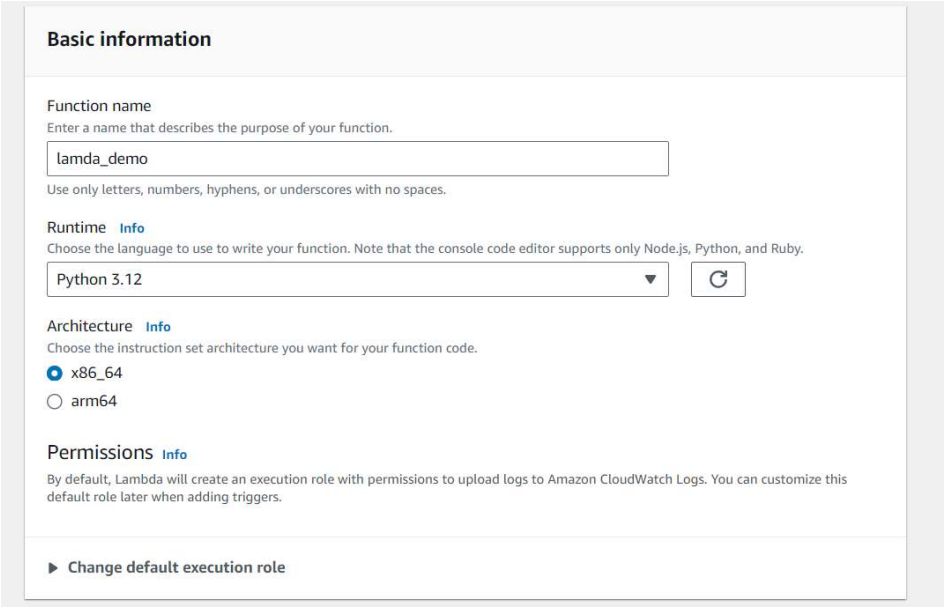
Step 2: Creating a New Lambda Function

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.



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**

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

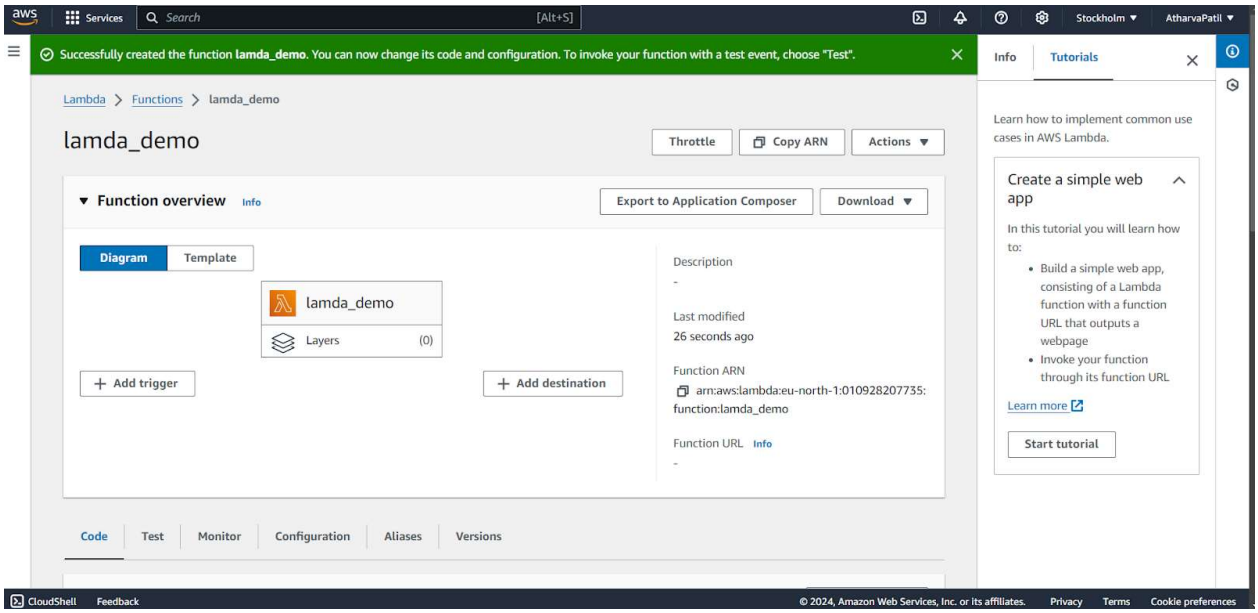
ⓘ Role creation might take a few minutes. Please do not delete the role or edit the trust or permissions policies in this role.

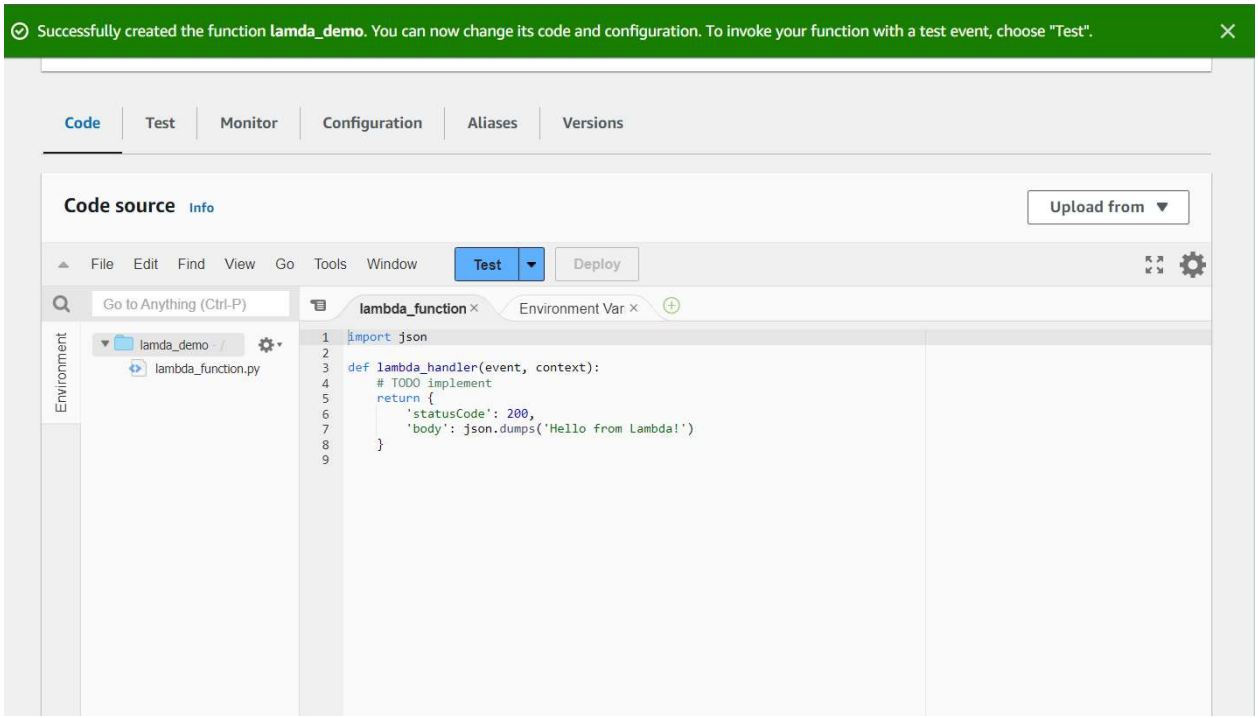
Lambda will create an execution role named ATHARV_LAMDA-role-0u7c9ooi, with permission to upload logs to Amazon CloudWatch Logs.

▶ **Additional Configurations**

Use additional configurations to set up code signing, function URL, tags, and Amazon VPC access for your function.

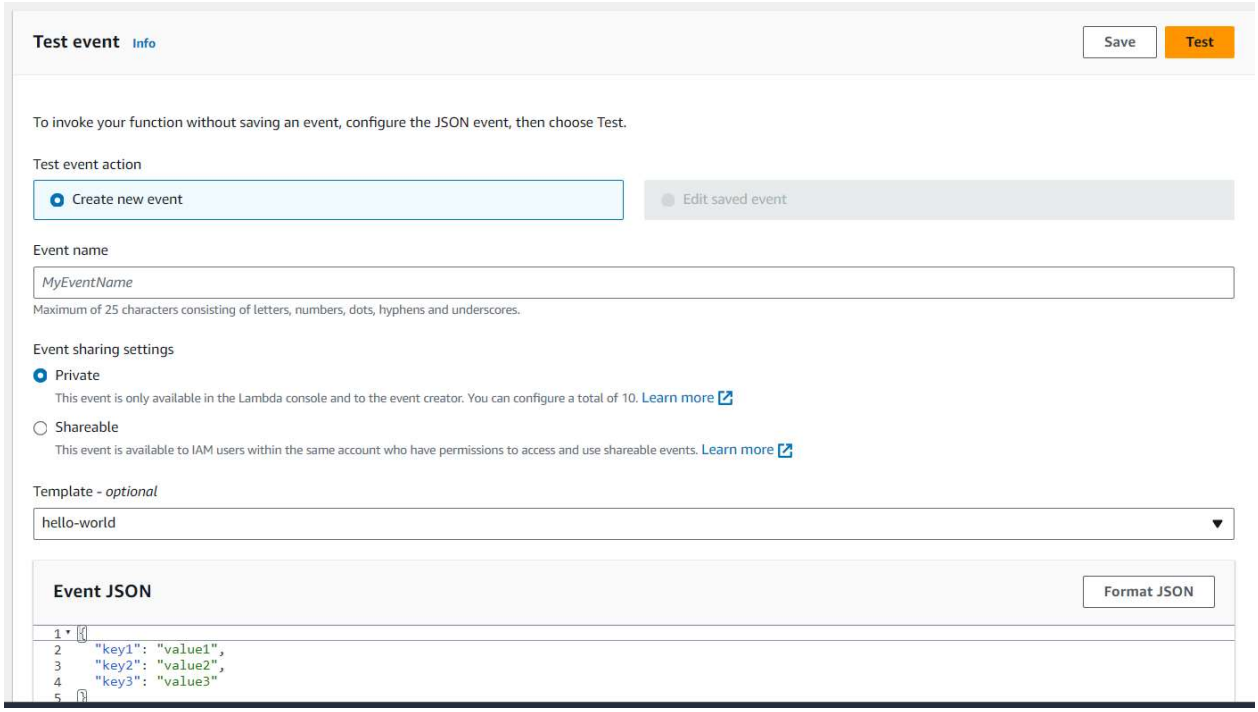
Cancel | **Create function**

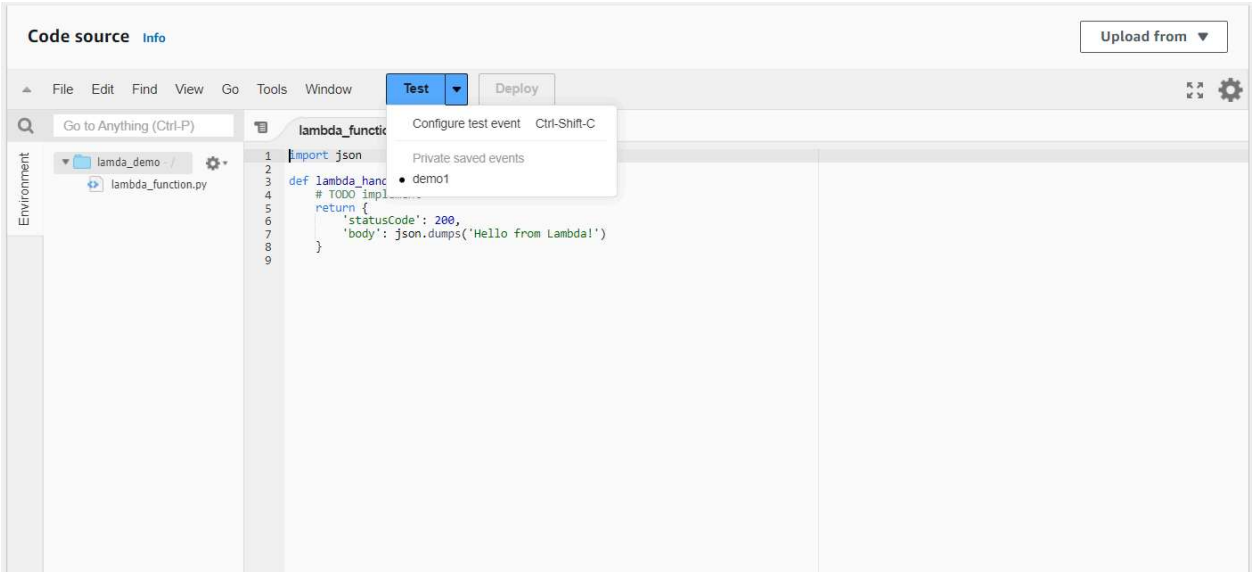


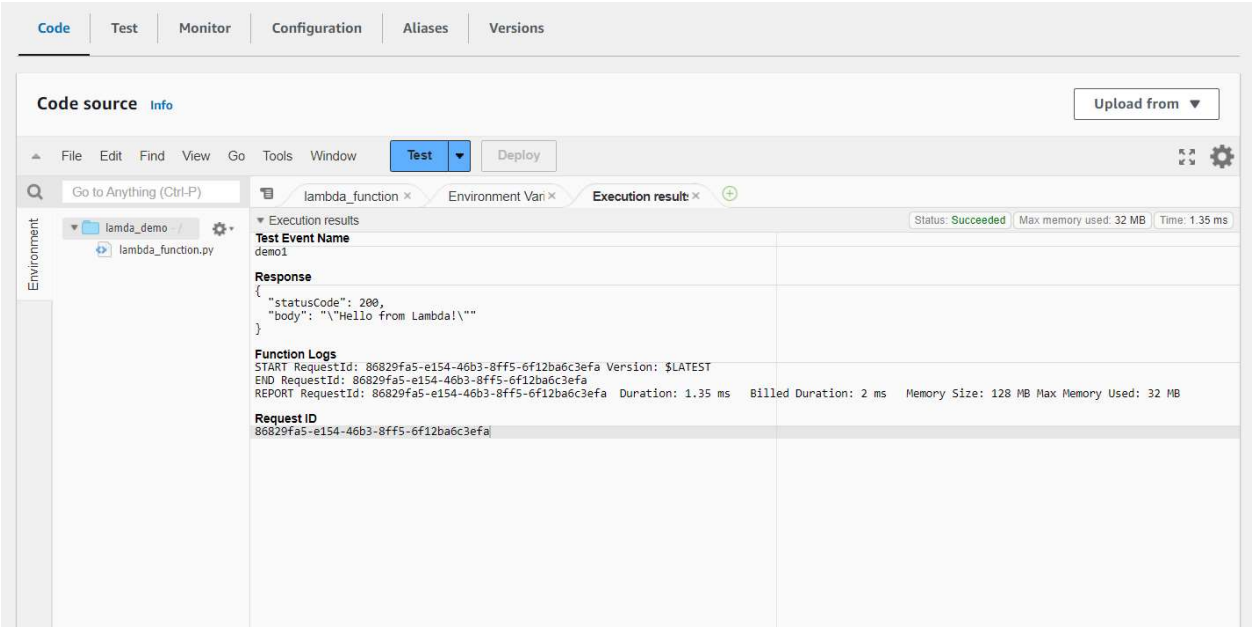


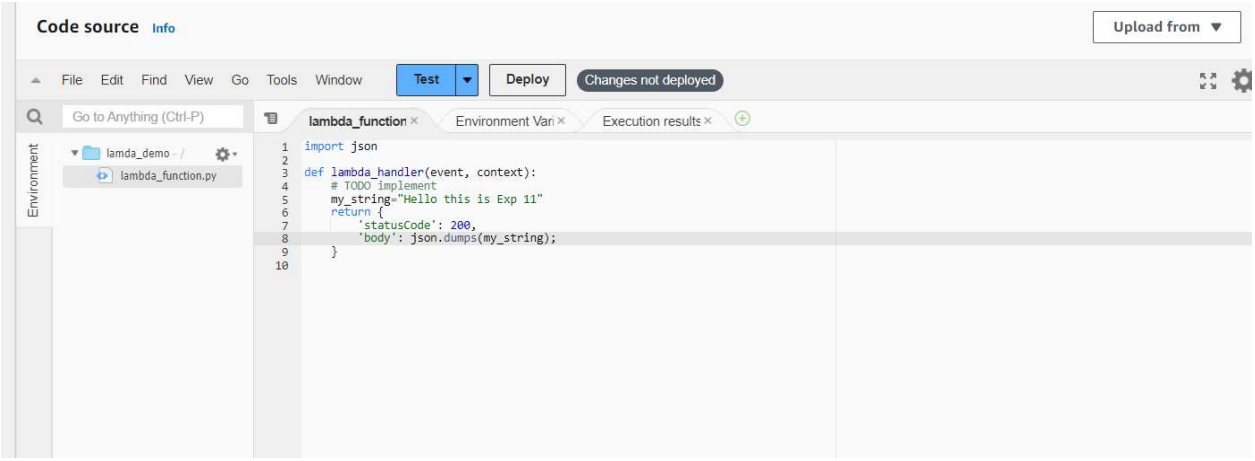
Step 4: Testing the Function

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.



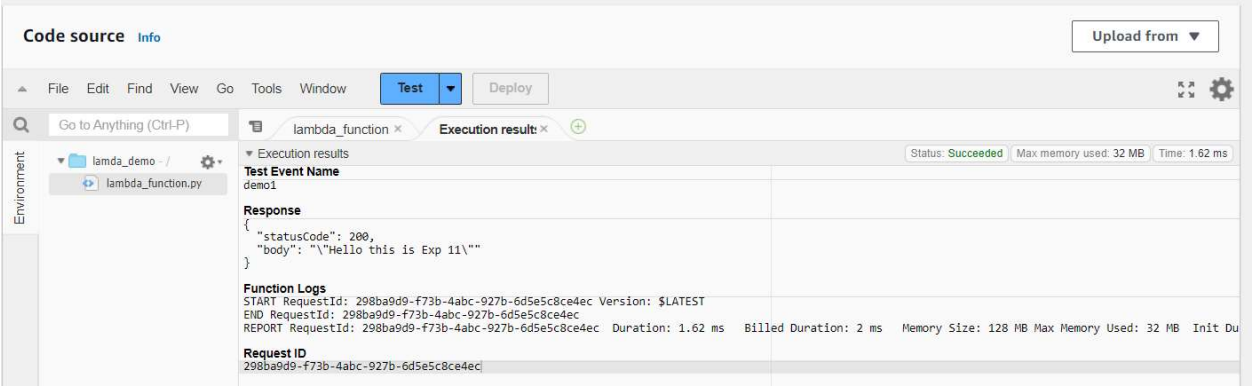

Code source Info
Upload from
File Edit Find View Go Tools Window
Test
Deploy
Configure test event Ctrl-Shift-C
Private saved events
demo1
Go to Anything (Ctrl-P)
lambda_functio
Environment
lamda_demo
lambda_function.py
import json
def lambda_hand
# TODO impl
return {
'statusCode': 200,
'body': json.dumps('Hello from Lambda!')
}


Code
Test
Monitor
Configuration
Aliases
Versions
Code source Info
Upload from
File Edit Find View Go Tools Window
Test
Deploy
Go to Anything (Ctrl-P)
lambda_function
Environment Vari
Execution result
Environment
lamda_demo
lambda_function.py
Execution results
Status: Succeeded
Max memory used: 32 MB
Time: 1.35 ms
Test Event Name
demo1
Response
{
"statusCode": 200,
"body": "\"Hello from Lambda!\""
}
Function Logs
START RequestId: 86829fa5-e154-46b3-8ff5-6f12ba6c3efa Version: $LATEST
END RequestId: 86829fa5-e154-46b3-8ff5-6f12ba6c3efa
REPORT RequestId: 86829fa5-e154-46b3-8ff5-6f12ba6c3efa Duration: 1.35 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 32 MB
Request ID
86829fa5-e154-46b3-8ff5-6f12ba6c3efa



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.

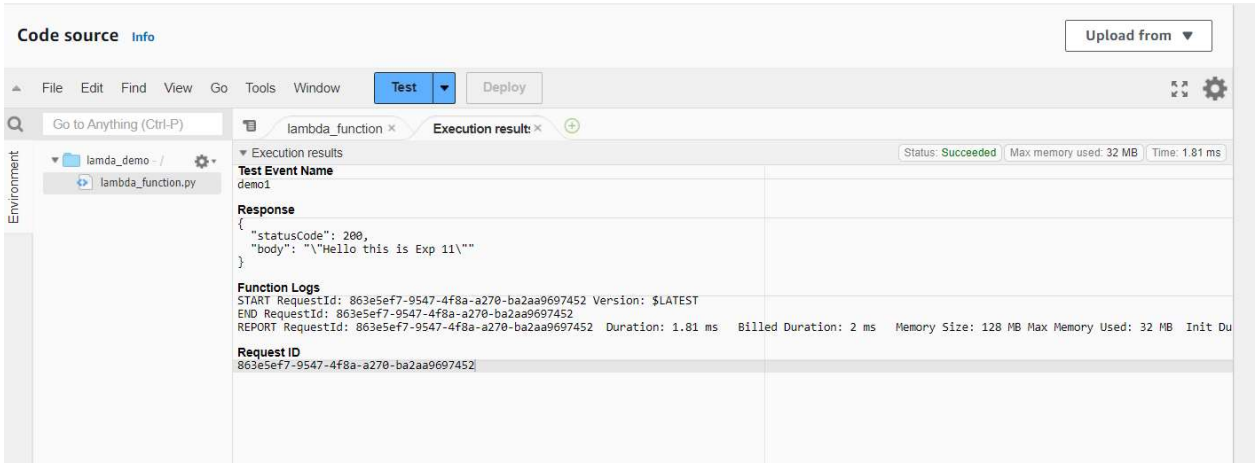


Step 6: Editing and Deploying the Code

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.