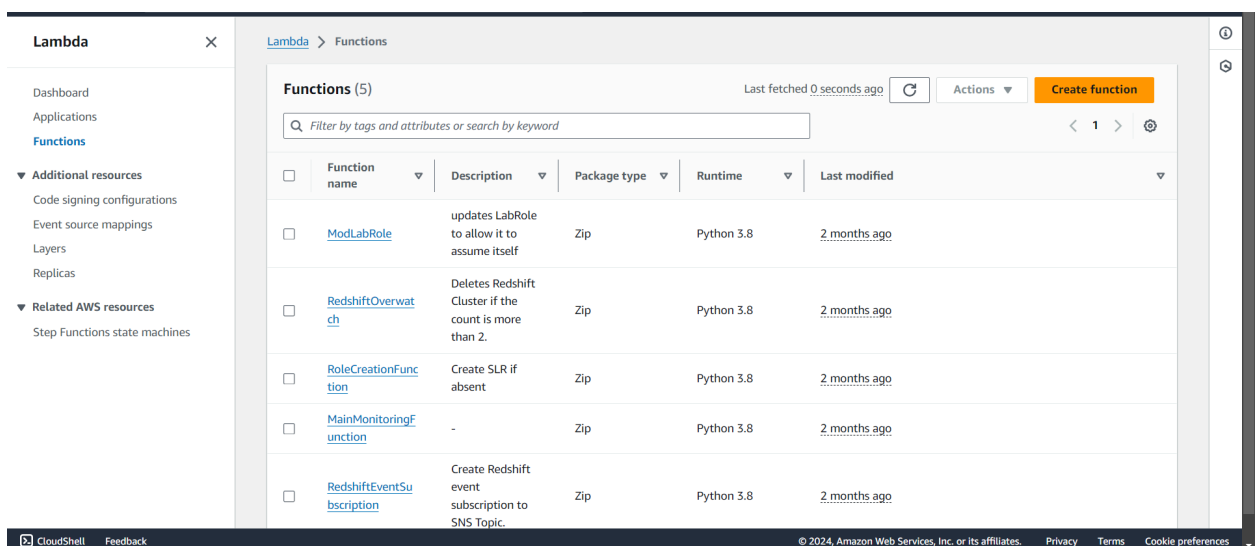
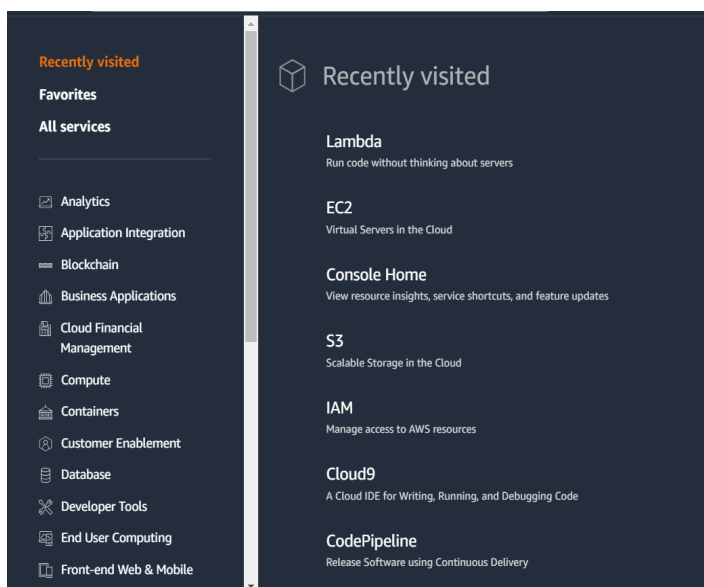


Experiment No: 11

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

CREATION OF LAMBDA FUNCTION:

Step1: Log in to your AWS Personal or Academy account. Navigate to Lambda, then select the 'Create Function' button



Step 2: Give your Lambda function a name and choose a programming language. The code editor only supports Node.js, Python, and Ruby, so in my case I have chosen **Python 3.12**. Set the **architecture to x86**. For the execution role, select '**Use an existing role**,' then pick '**Lab role**' from the dropdown menu under existing roles .

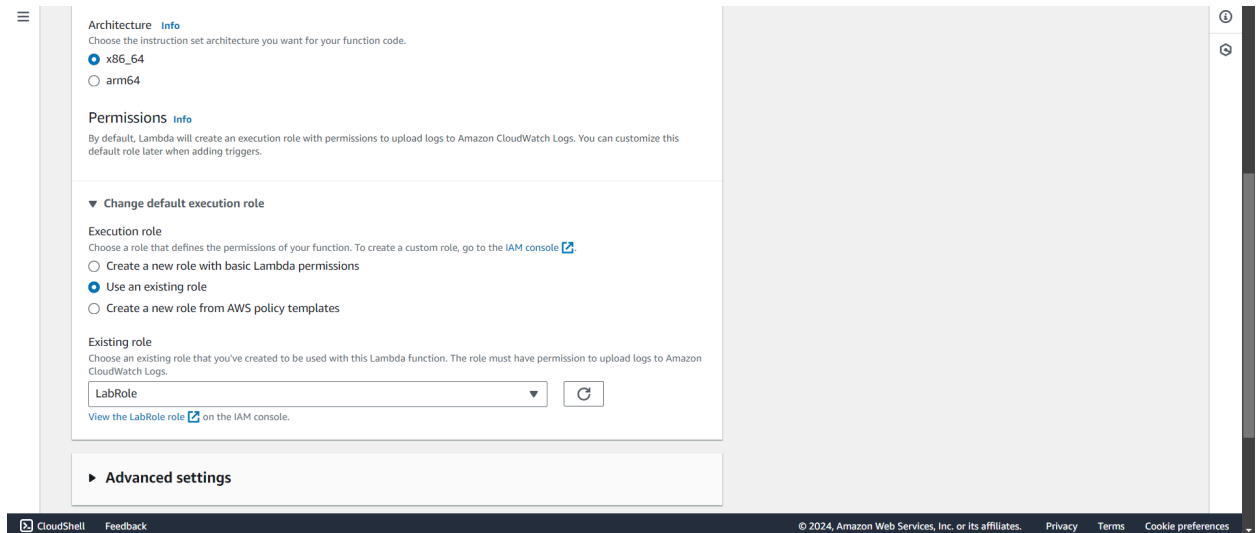
(This is because the Lab role already has the permissions needed for Lambda to run properly, so you don't need to create a new role from scratch. It's a quicker and more convenient option)

The screenshot shows the AWS Lambda 'Create function' page. At the top, there are three options to create a function: 'Author from scratch' (selected), 'Use a blueprint', and 'Container image'. Below these is the 'Basic information' section. It contains a 'Function name' field with the value 'Shivam_Lambda', a 'Runtime' dropdown menu set to 'Python 3.12', and an 'Architecture' dropdown menu set to 'x86_64'. The footer of the console shows 'CloudShell', 'Feedback', and copyright information for Amazon Web Services, Inc. or its affiliates.

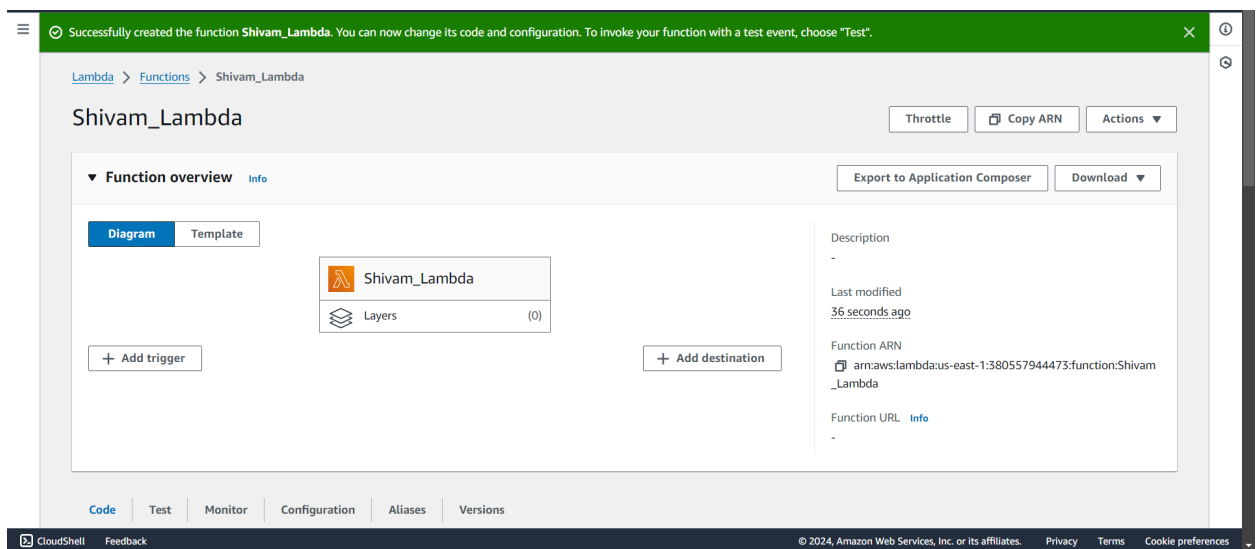
Give the function name and select required language for lambda function

This screenshot is a closer view of the 'Basic information' section from the previous image. It shows the 'Function name' field with 'Shivam_Lambda', the 'Runtime' dropdown with 'Python 3.12', and the 'Architecture' dropdown with 'x86_64'. Below the architecture section, there is a 'Permissions' section with a note about default permissions.

Architecture will be x86_64

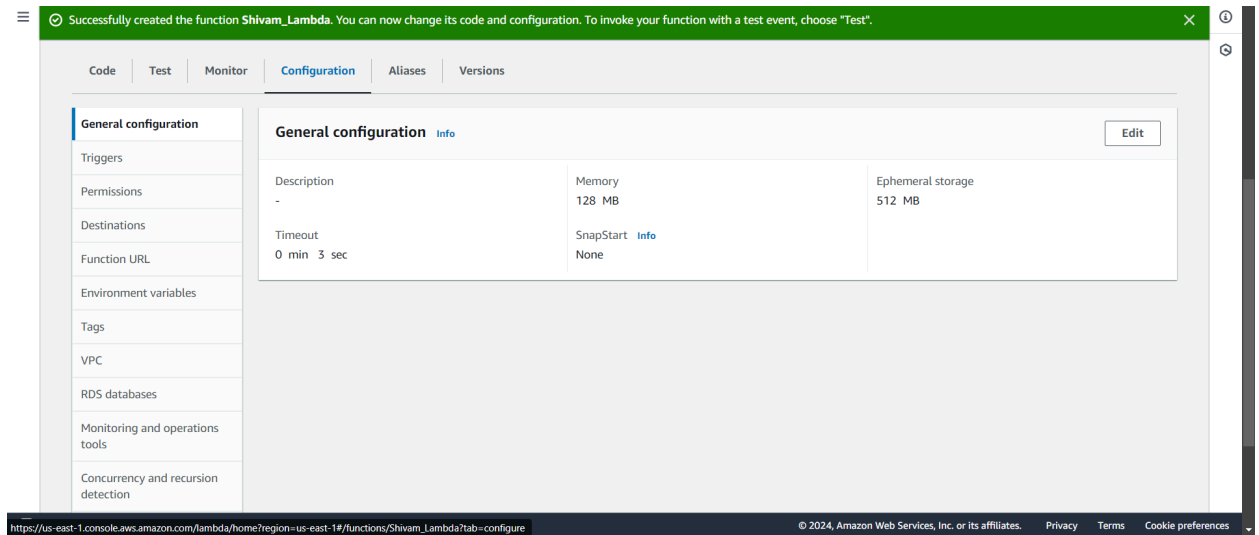


Select proper Execution role

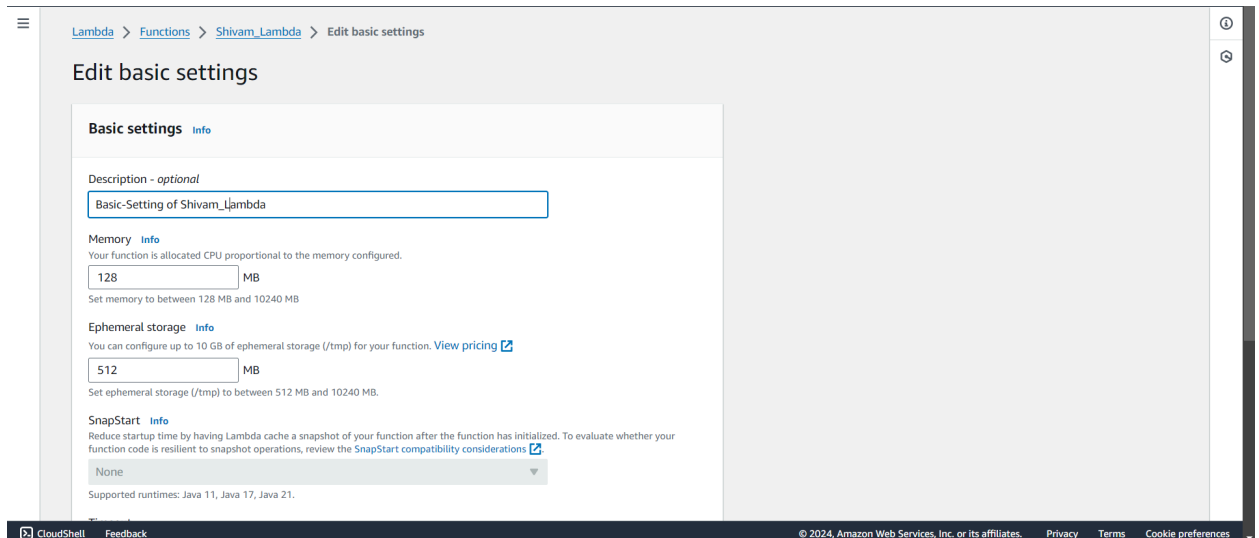


Successfully created Lambda function

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



You can add a description and adjust the memory and timeout settings. I've changed the timeout to 1 second, as that's enough for now.



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

SnapStart [Info](#)
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](#) [↗](#).

Supported runtimes: Java 11, Java 17, Java 21.

Timeout
 min sec

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#) [↗](#).
☒ 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.
 [↻](#)
[View the LabRole role](#) [↗](#) on the IAM console.

[Cancel](#) [Save](#)

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Click on Save

Step 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 basically create a new event to test and verify your Lambda function; setting Event Sharing to private keeps it secure and choosing the "hello-world" template provides a simple structure for testing without complex inputs.

[Code](#) [Test](#) [Monitor](#) [Configuration](#) [Aliases](#) [Versions](#)

Test event [Info](#) [Save](#) [Test](#)

To invoke your function without saving an event, configure the JSON event, then choose Test.

Test event action
☒ Create new event ☐ Edit saved event

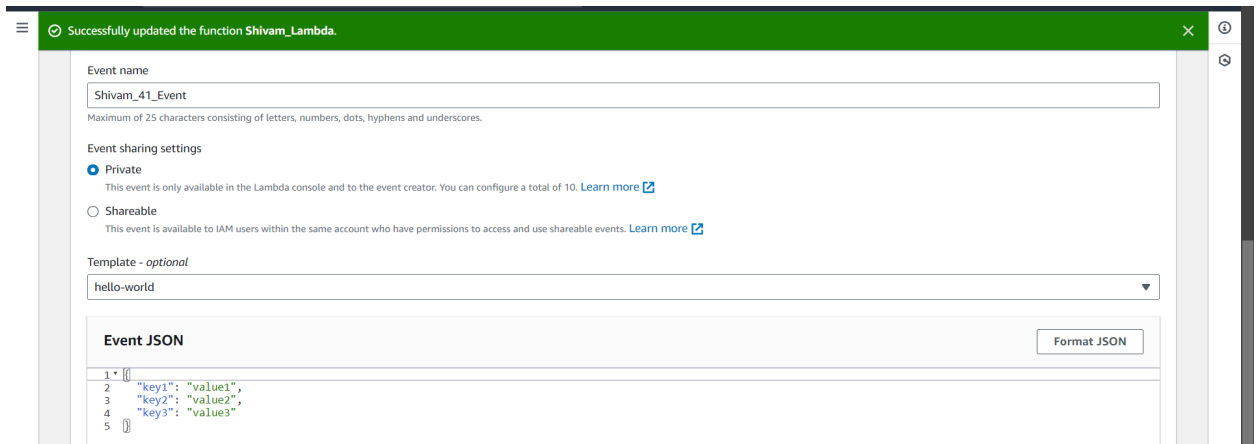
Event name

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

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences



Successfully updated the function Shivam_Lambda.

Event name
Shivam_41_Event
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](#)

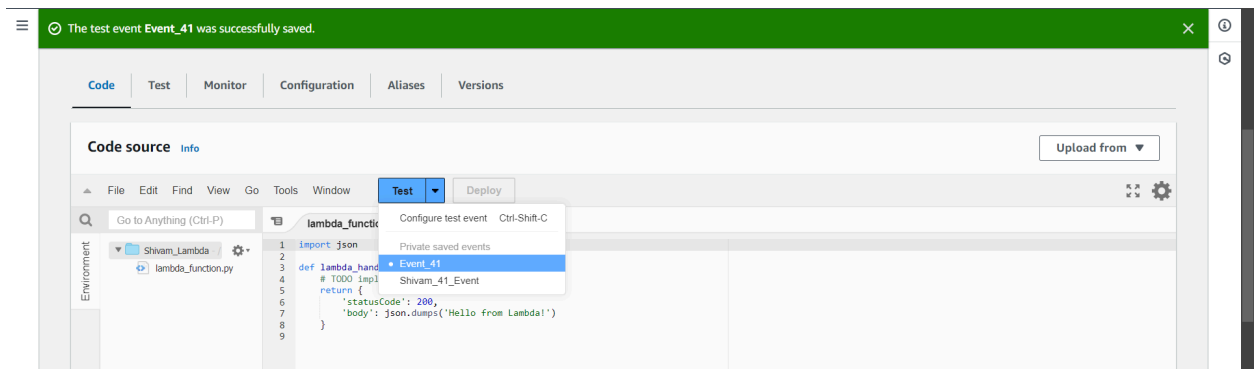
Template - optional
hello-world

Event JSON
Format JSON

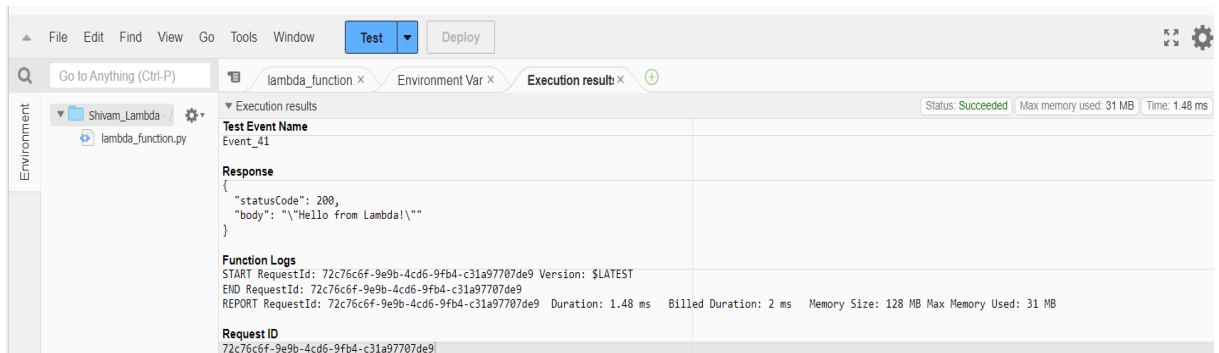
```
1 {  
2   "key1": "value1",  
3   "key2": "value2",  
4   "key3": "value3",  
5 }
```

Click on save button above

Step 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."

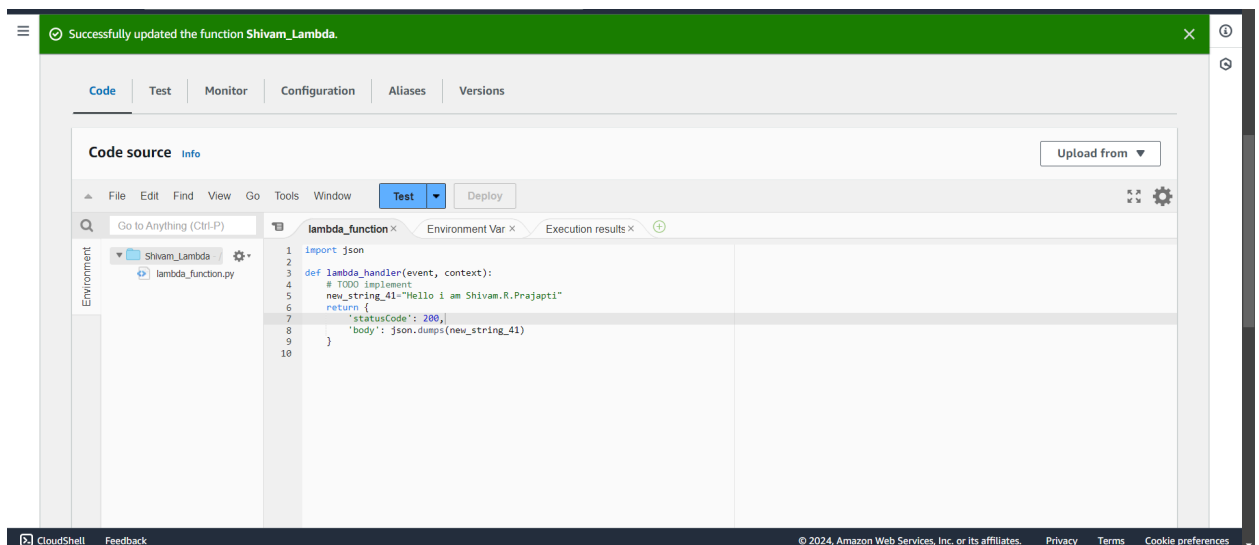
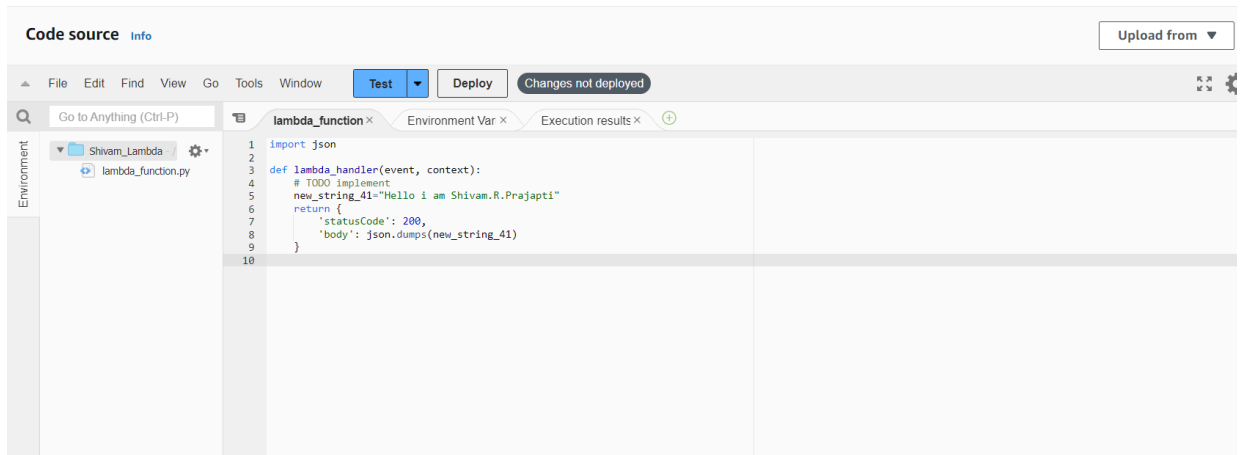


Output :



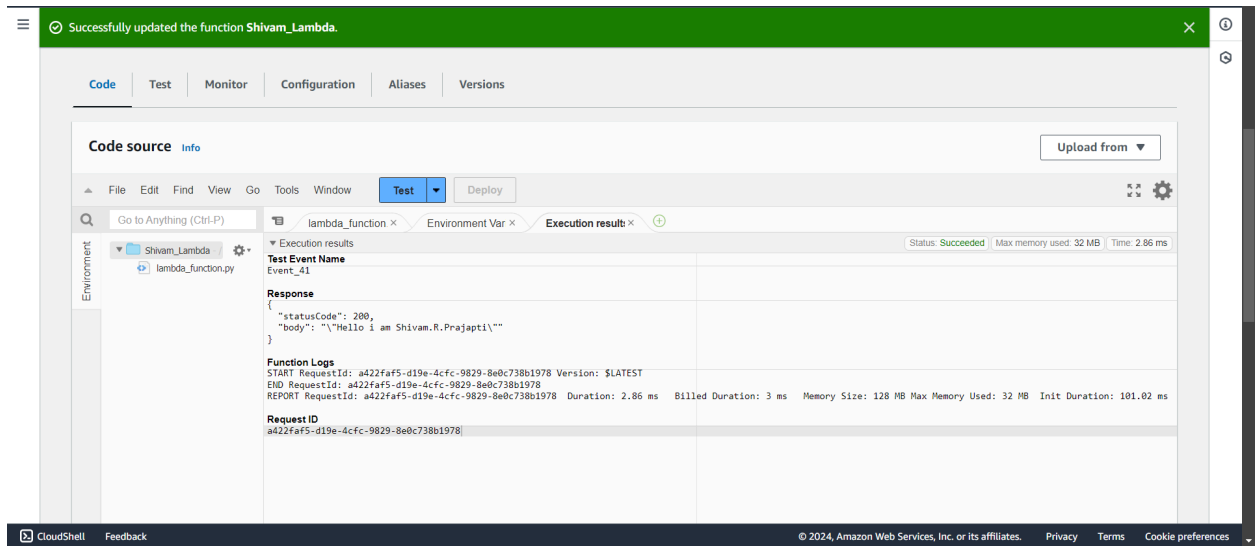
You select the created event to run the specific test you set up, and clicking 'Test' executes your Lambda function to check if it works as expected and produces the desired output.

Step 6: You can edit your lambda function code. I have changed the code to display the new String. After Changing save it by Control + S and click on Deploy . Make sure you have internet connectivity while deploying or else it will show failed deployment



Successfully changed the function.

Step 7: Click on 'Test' to see the output. You'll get a status code of **200** which means "OK" and indicates that the request was successful, your string output, and the function logs, showing that it was deployed successfully.



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.