Since 1962

Vivekanand Education Society's

Institute of Technology

An Autonomous Institute Affiliated to University of Mumbai,, Approved by AICTE & Recognized by Govt. of Maharashtra

Hashu Advani Memorial Complex, Collector Colony, Chembur East, Mumbai - 400074.

Department of Information Technology A.Y. 2024-25

Advance DevOps Lab
Experiment 11

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

Roll No.	22
Name	Sarthak Harade
Class	D15B
Subject	Advance DevOps Lab
LO Mapped	LO1: To understand the fundamentals of Cloud Computing and be fully proficient with Cloud based DevOps solution deployment options to meet your business requirements. LO6: To engineer a composition of nano services using AWS Lambda and Step Functions with the Serverless Framework.
Grade:	

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

<u>THEORY :</u>

AWS Lambda is a serverless computing service by AWS that allows you to run code without provisioning or managing servers. You create functions in supported languages like Python, Java, and Node.js, and these functions are executed in response to specific events such as API calls, file uploads to S3, or data changes in DynamoDB.

Key Features

● Automatic Scaling: Lambda automatically scales the infrastructure to handle incoming requests, reducing operational complexity.

● Cost-Efficiency: You only pay for the compute time you consume, with no upfront costs or server management fees.

● Security: Lambda integrates with AWS Identity and Access Management (IAM) to define roles and policies, ensuring secure execution.

● Fault Tolerance: AWS Lambda is designed to provide high availability and fault tolerance, handling server failures and maintaining continuous operation.

Execution Model

Lambda functions run in stateless containers fully managed by AWS. When an event triggers a function, AWS initiates a container to execute the function. If subsequent requests come in, additional containers are spun up to handle them. AWS may keep containers warm for a short period to reduce cold start latency.

Stateless Functions

Due to the stateless nature of Lambda, each function invocation is independent, running in a fresh environment. Code outside the main handler function runs once per container lifecycle, while the handler itself runs on every invocation.

Common Use Cases

● Scalable APIs: Lambda is ideal for building APIs that need to scale according to demand. Each API request can be routed to a specific Lambda function, and the service automatically adjusts to handle varying workloads.

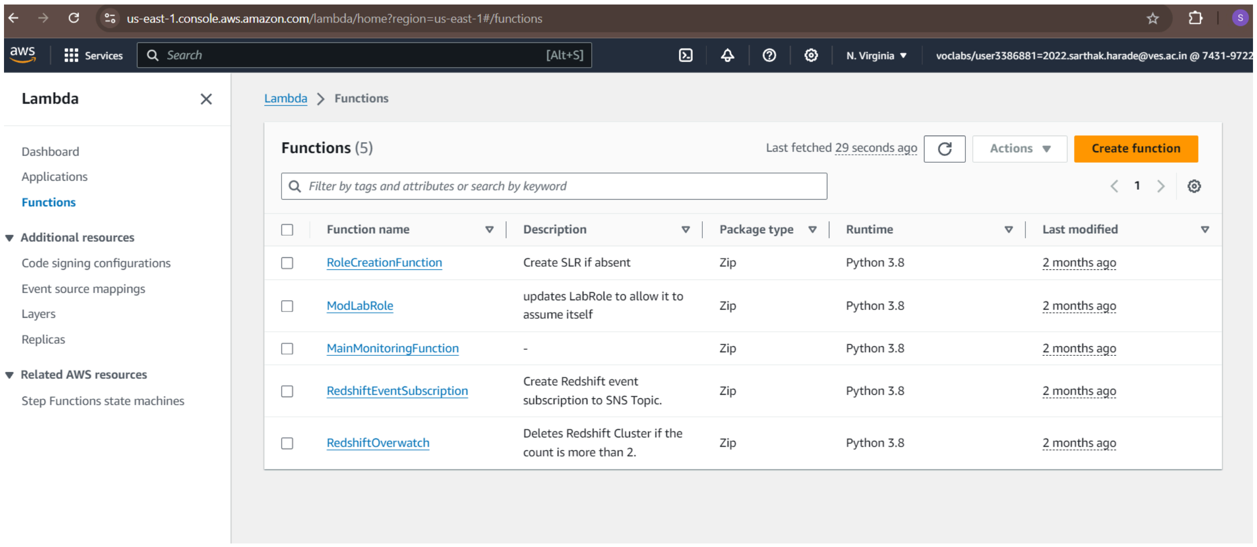
● Event-Driven Data Processing: Lambda excels in scenarios like real-time data processing, where functions are triggered by events from sources like S3 or DynamoDB, making it suitable for tasks like data transformation, analytics, and notifications.

Packaging and Deployment

Lambda functions, along with their dependencies, are packaged and uploaded to AWS, often using an S3 bucket. AWS Lambda then uses this package to execute the function when an event occurs. Tools like the Serverless Stack Framework (SST) can simplify this process.

<u>Steps to create an AWS Lambda function</u>

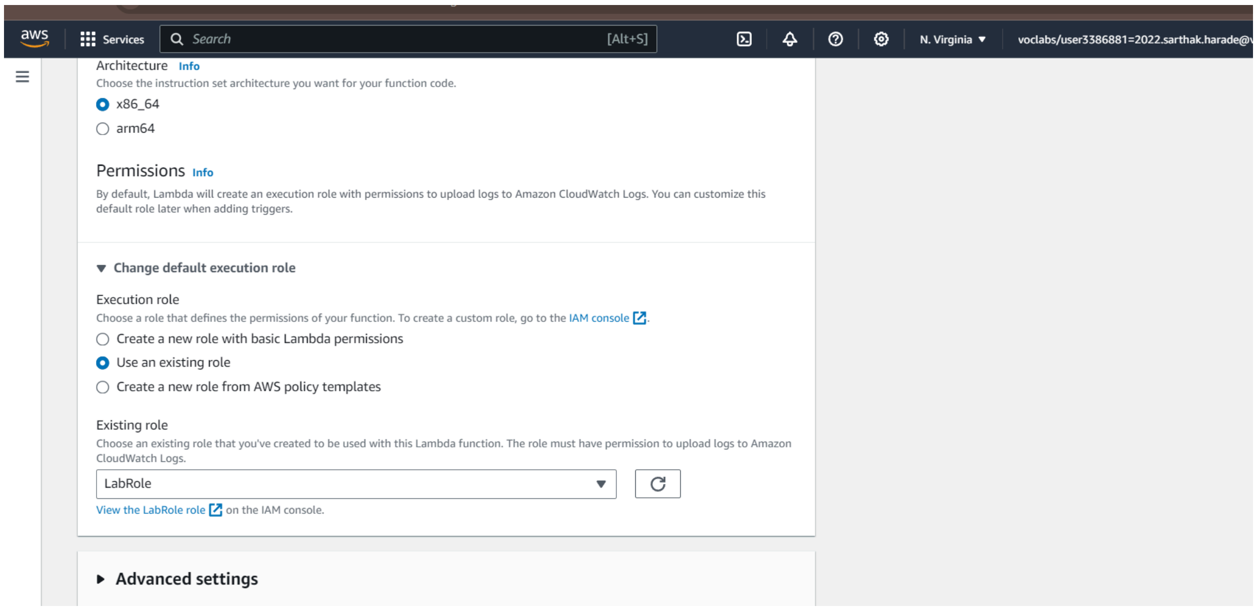
Open up the Lambda Console and click on the Create button.


us-east-1.console.aws.amazon.com/lambda/home?region=us-east-1#/functions
Services
Search
[Alt+S]
N. Virginia
voclabs/user3386881=2022.sarthak.harade@ves.ac.in @ 7431-9722
Lambda
Dashboard
Applications
Functions
Additional resources
Code signing configurations
Event source mappings
Layers
Replicas
Related AWS resources
Step Functions state machines
Lambda > Functions
Last fetched 29 seconds ago
Actions
Create function
Filter by tags and attributes or search by keyword

Functions (5)

Function name	Description	Package type	Runtime	Last modified
RoleCreationFunction	Create SLR if absent	Zip	Python 3.8	2 months ago
ModLabRole	updates LabRole to allow it to assume itself	Zip	Python 3.8	2 months ago
MainMonitoringFunction	-	Zip	Python 3.8	2 months ago
RedshiftEventSubscription	Create Redshift event subscription to SNS Topic.	Zip	Python 3.8	2 months ago
RedshiftOverwatch	Deletes Redshift Cluster if the count is more than 2.	Zip	Python 3.8	2 months ago

Choose to create a function from scratch or use a blueprint, i.e templates defined by AWS for you with all configuration presets required for the most common use cases.



Architecture

Choose the instruction set architecture you want for your function code.

[✓ x86_64]
arm64

Permissions

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

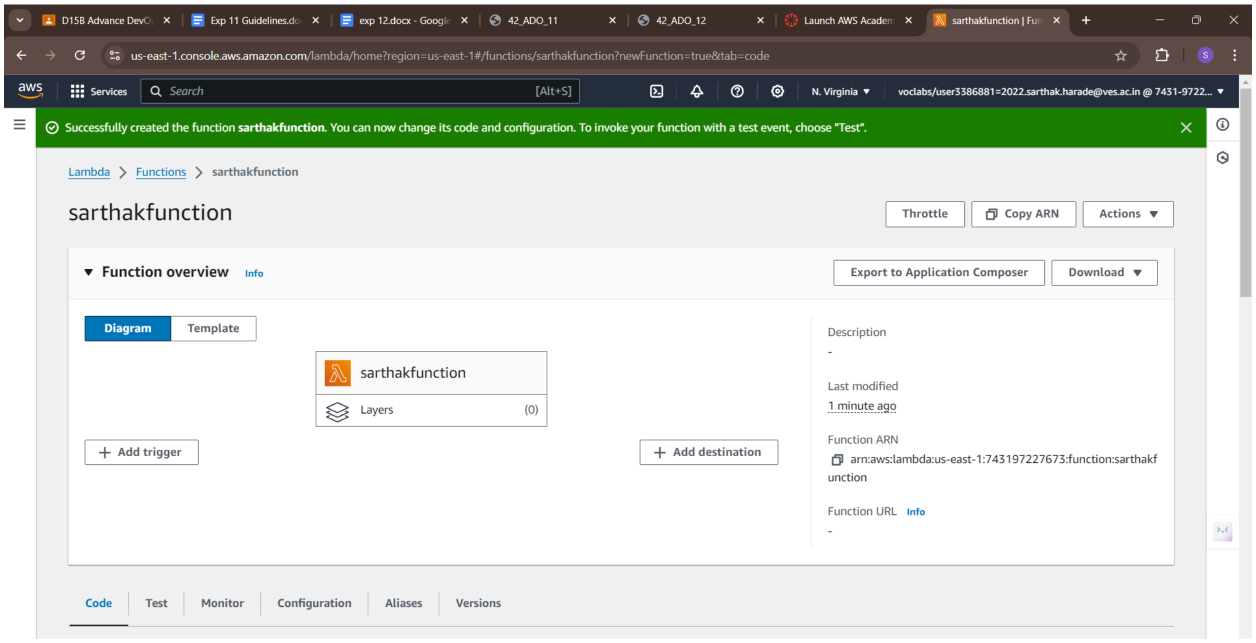
Existing role

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.

This process will take a while to finish and after that, you'll get a message that your function was successfully created.



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

Lambda > Functions > sarthakfunction

sarthakfunction

Function overview

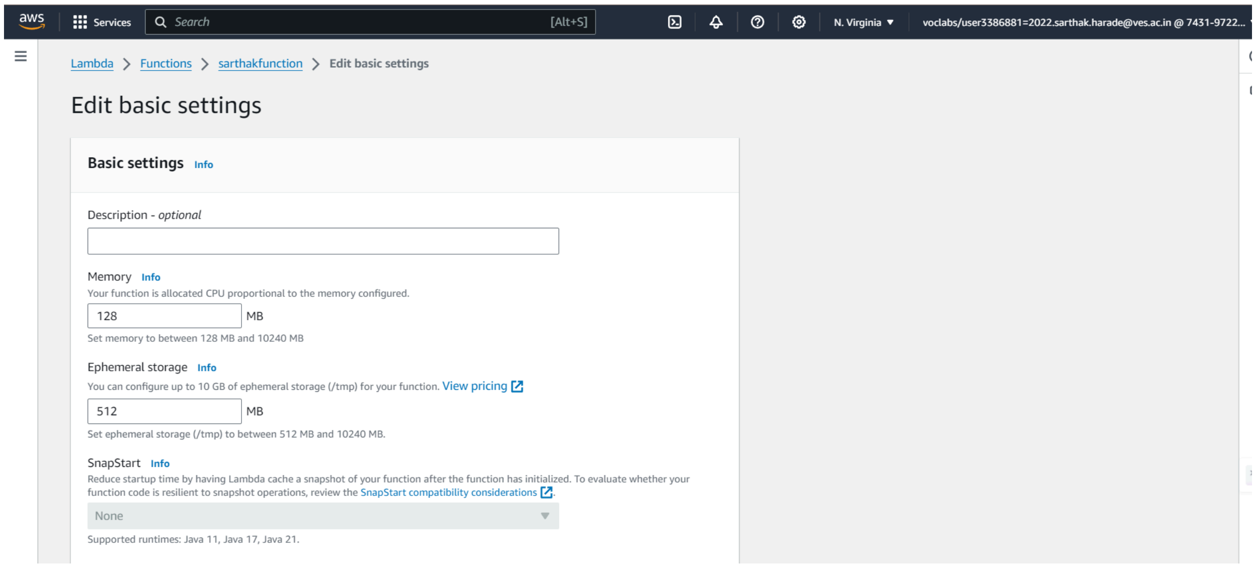
Description
-

Last modified
1 minute ago

Function ARN
arn:aws:lambda:us-east-1:743197227673:function:sarthakfunction

Function URL
-

To change the configuration, open up the Configuration tab and under General Configuration, choose Edit.



Lambda > Functions > sarthakfunction > Edit basic settings

Edit basic settings

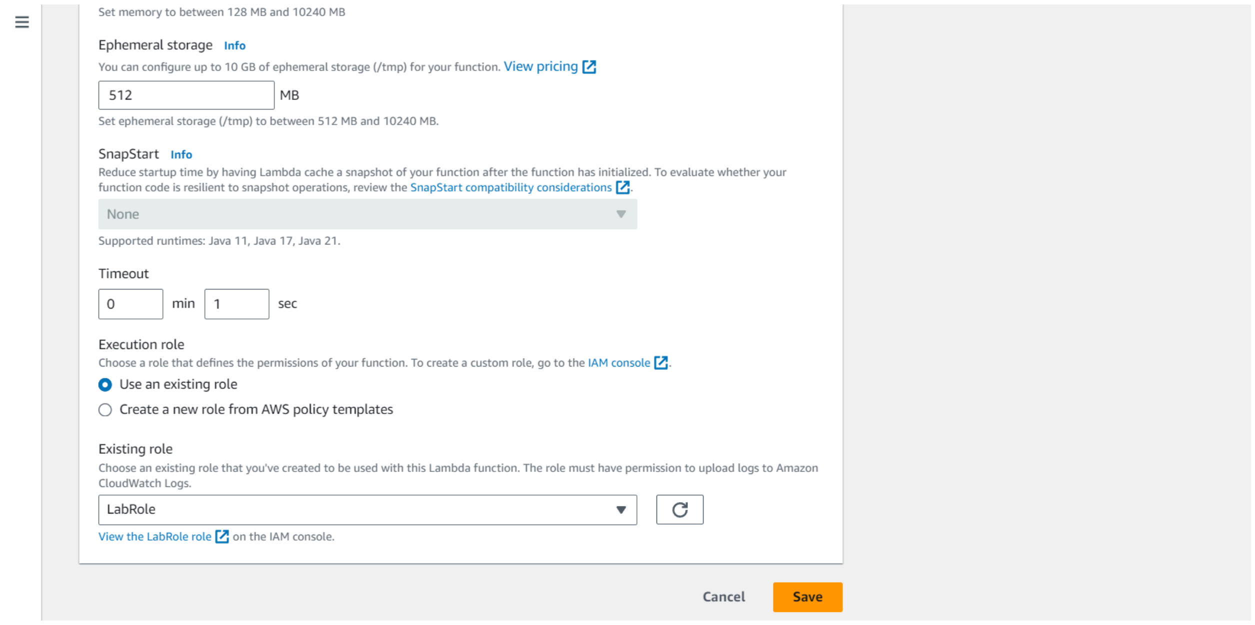
Basic settings

Description - *optional*

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.
None
Supported runtimes: Java 11, Java 17, Java 21.


Set memory to between 128 MB and 10240 MB

Ephemeral storage Info

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 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.

None

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

Timeout

0 min 1 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

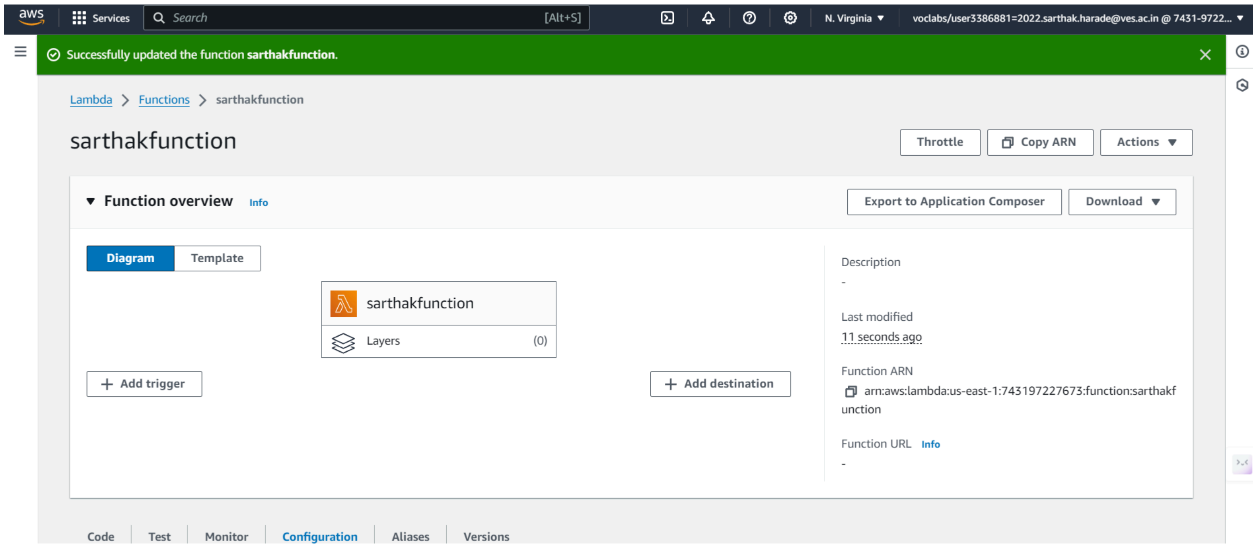
Existing role

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.

LabRole

View the LabRole role on the IAM console.

Cancel Save


aws
Services
Search [Alt+S]
N. Virginia
voclabs/user3386881=2022.sarthak.harade@ves.ac.in @ 7431-9722...

Successfully updated the function **sarthakfunction**.

Lambda > Functions > sarthakfunction

sarthakfunction

Function overview Info

Description

-

Last modified

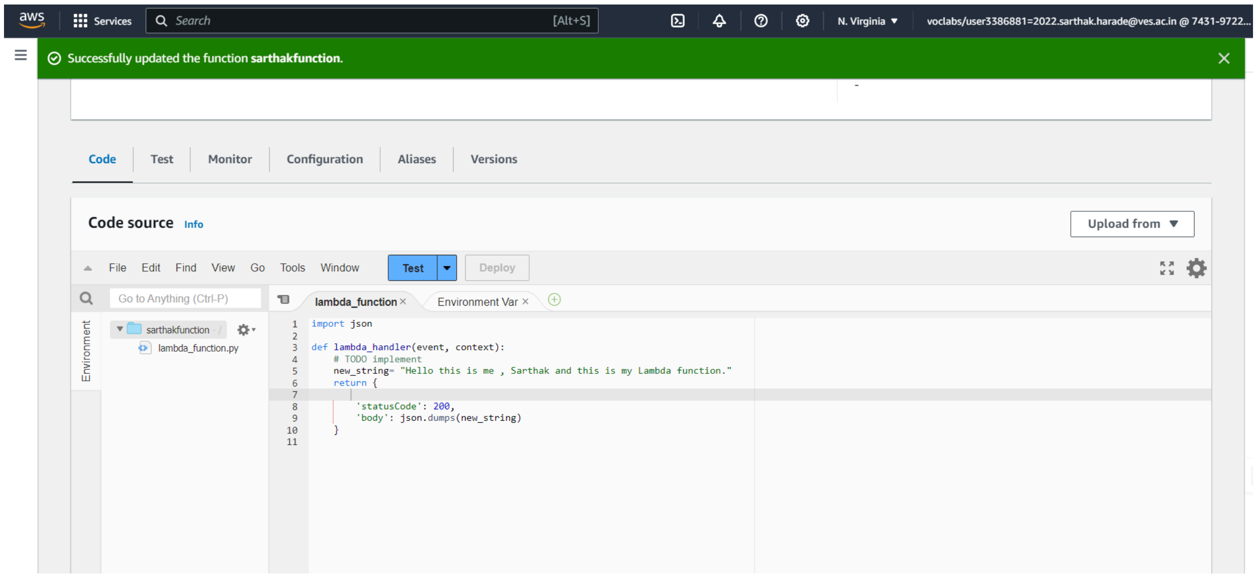
11 seconds ago

Function ARN

arn:aws:lambda:us-east-1:743197227673:function:sarthakfunction

Function URL Info

-



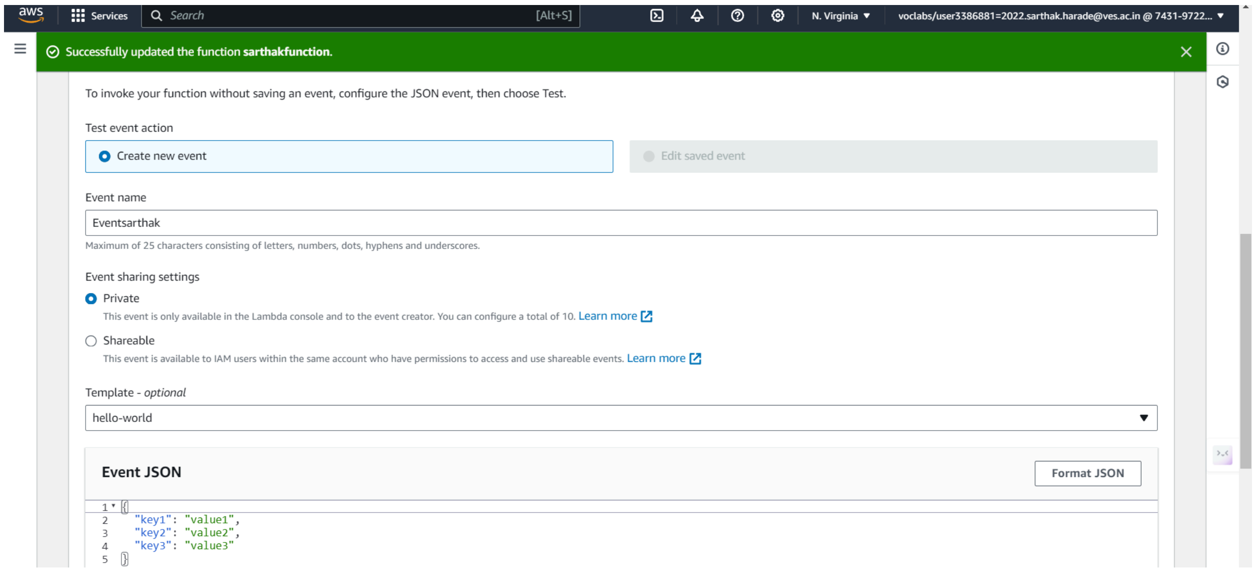
Successfully updated the function **sarthakfunction**.

Code source

```
import json

def lambda_handler(event, context):
    # TODO implement
    new_string= "Hello this is me , Sarthak and this is my Lambda function."
    return {

        'statusCode': 200,
        'body': json.dumps(new_string)
    }
```


aws
Services
Search
[Alt+S]
N. Virginia
voclabs/user3386881=2022.sarthak.harade@ves.ac.in @ 7431-9722...
Format JSON

Successfully updated the function **sarthakfunction**.

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

Eventsarthak

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

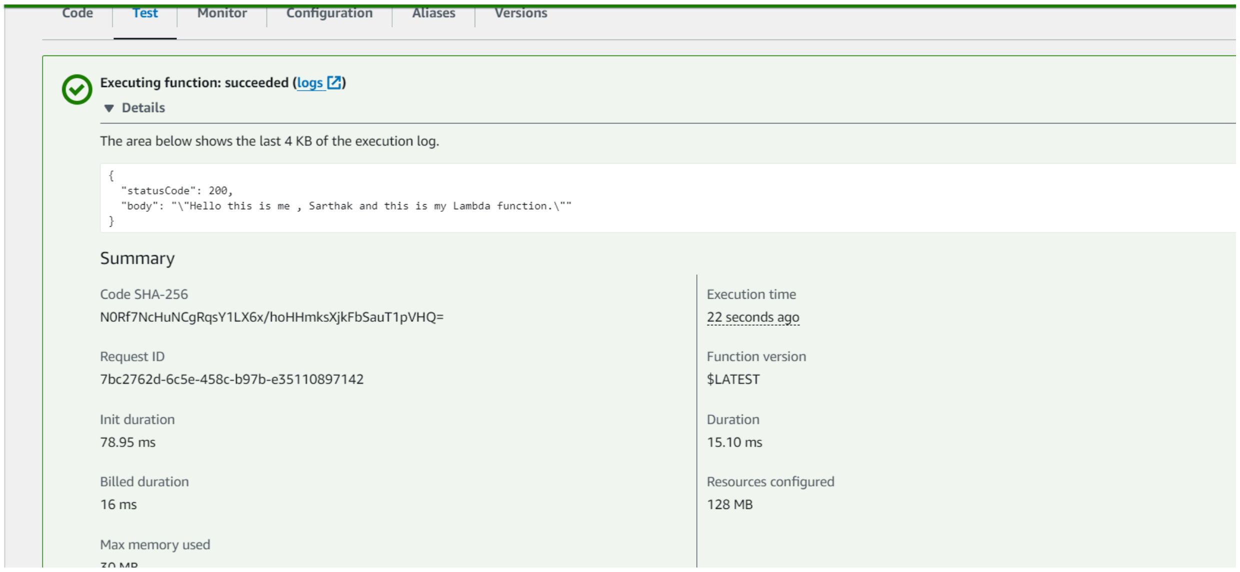
Template - *optional*

hello-world

Event JSON

```
{
  "key1": "value1",
  "key2": "value2",
  "key3": "value3"
}
```

Now click on Test and you should be able to see the results



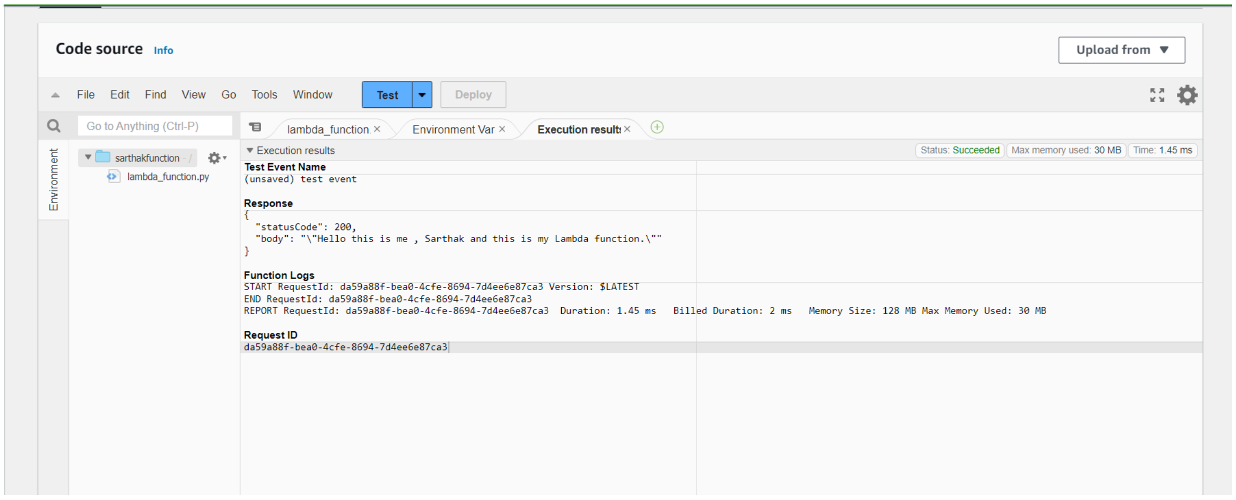
Executing function: succeeded (logs)

The area below shows the last 4 KB of the execution log.

```
{
  "statusCode": 200,
  "body": "\"Hello this is me , Sarthak and this is my Lambda function.\""
}
```

Summary

Code SHA-256	N0Rf7NcHuNCgRqsY1LX6x/hoHHmksXjkFbSauT1pVHQ=
Request ID	7bc2762d-6c5e-458c-b97b-e35110897142
Init duration	78.95 ms
Billed duration	16 ms
Max memory used	30 MB
Execution time	22 seconds ago
Function version	$LATEST
Duration	15.10 ms
Resources configured	128 MB



Status: Succeeded | Max memory used: 30 MB | Time: 1.45 ms

```
Test Event Name
(unsaved) test event

Response
{
  "statusCode": 200,
  "body": "\"Hello this is me , Sarthak and this is my Lambda function.\""
}

Function Logs
START RequestId: da59a88f-bea0-4cfe-8694-7d4ee6e87ca3 Version: $LATEST
END RequestId: da59a88f-bea0-4cfe-8694-7d4ee6e87ca3
REPORT RequestId: da59a88f-bea0-4cfe-8694-7d4ee6e87ca3  Duration: 1.45 ms   Billed Duration: 2 ms   Memory Size: 128 MB Max Memory Used: 30 MB

Request ID
da59a88f-bea0-4cfe-8694-7d4ee6e87ca3
```

CONCLUSION : AWS Lambda simplifies the process of running code in the cloud by handling server management, scaling, and security for you. Its flexibility and cost-efficiency make it an ideal choice for a wide range of applications, from scalable APIs to real-time data processing. By leveraging AWS Lambda, you can focus on writing code while AWS takes care of the rest.