

ASSIGNMENT-15

Create a serverless computing service.

Step 1: Sign in to your AWS Account as Root User. Search for “Lambda” and open it. Then click on “Create function”.

Resources for Asia Pacific (Mumbai)				Create function
Lambda function(s)	Code storage	Full account concurrency	Unreserved account concurrency	
0	0 byte (0% of 75.0 GB)	10	10	

Step 2: Select “Author from scratch”. Give Function name and Runtime. Then click on “Create function”.

Lambda > Functions > Create function

Create function [Info](#)

AWS Serverless Application Repository applications have moved to [Create application](#).

☒ 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.

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.

Node.js 18.x

Architecture [Info](#)
Choose the instruction set architecture you want for your function code.

☒ x86_64

☐ arm64

► Advanced settings

Cancel Create function

Step 3: The function is successfully created. Now scroll down to the code section below.


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


Lambda > Functions > AkarFunction

AkarFunction

Throttle Copy ARN Actions

▼ Function overview Info

 AkarFunction

 Layers (0)

+ Add trigger

+ Add destination


Description

-

Last modified

6 minutes ago

Function ARN

 arn:aws:lambda:ap-south-1:782829530958:function:AkarFunction

Function URL [Info](#)

-

Code Test Monitor Configuration Aliases Versions

Step 4: Write down your code in index.mjs.

Code source Info Upload from

File Edit Find View Go Tools Window Test Deploy

Go to Anything (Ctrl-P)

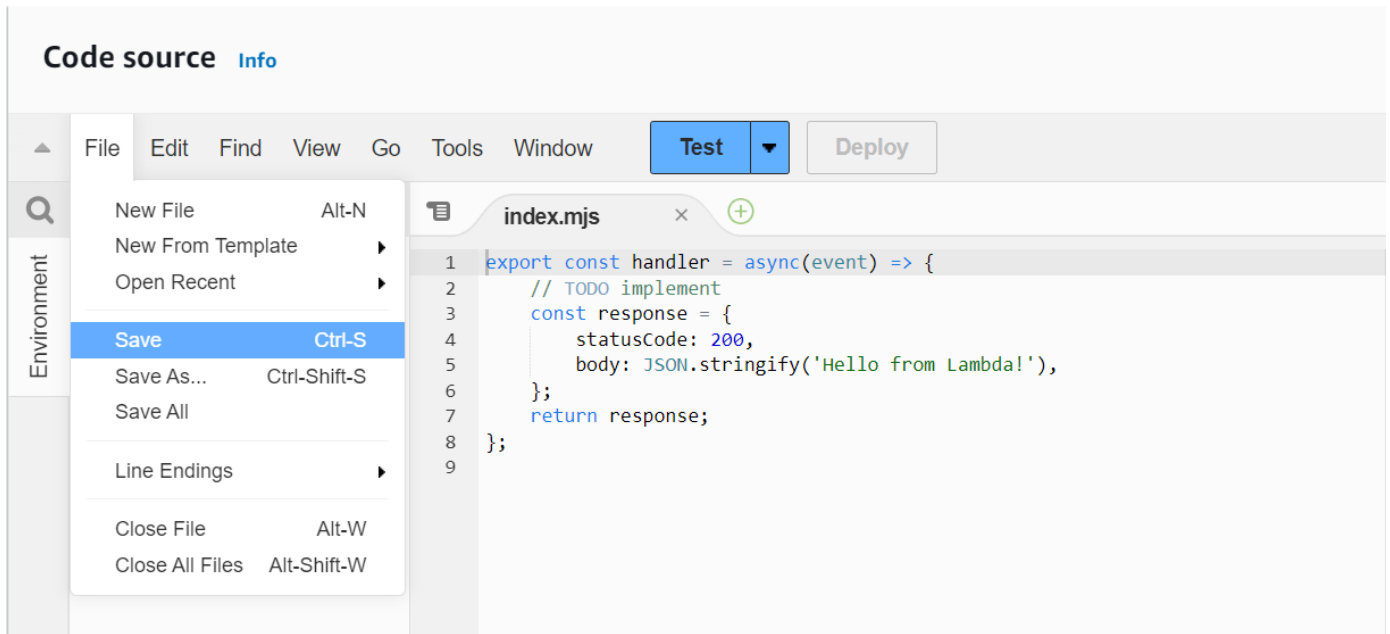
Environment

- AkarFunction
 - index.mjs

index.mjs

```
1 export const handler = async(event) => {
2   // TODO implement
3   const response = {
4     statusCode: 200,
5     body: JSON.stringify('Hello from Lambda!'),
6   };
7   return response;
8 };
9
```

Step 5: Then save the file and click on Test.



Step 6: This screen will pop-up. Give an event name and select its sharing settings to be Private. After that save it.

Configure test event

A test event is a JSON object that mocks the structure of requests emitted by AWS services to invoke a Lambda function. Use it to see the function's invocation result.

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

Event1

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

Event sharing settings

☒ Private ☐ Shareable

This event is only available in the Lambda console and to the event creator. You can configure a total of 10. [Learn more](#)

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

Cancel Save

Step 7: Again Test your code. Then, click on “Deploy” to Deploy it.

The screenshot shows the AWS Lambda console interface. At the top, there's a 'Code source' section with an 'Info' link and an 'Upload from' dropdown. Below this is a toolbar with 'File', 'Edit', 'Find', 'View', 'Go', 'Tools', 'Window', 'Test', and 'Deploy' buttons. A 'Changes not deployed' status bar is also present. The main area displays the 'Execution results' for a test event named 'Event1'. The 'Response' is shown as a JSON object: `{ "statusCode": 200, "body": "\"Hello from Lambda!\"" }`. Below the response, the 'Function Logs' are visible, showing the start, end, and report of the function execution with details like RequestId, Version, Duration, Billed Duration, Memory Size, and Max Memory Used. The 'Request ID' is also displayed at the bottom.

Step 8: After deploying, go to Configuration. And click on “Function URL”.

The screenshot shows the AWS Lambda console interface with the 'Configuration' tab selected. A green notification bar at the top indicates 'Successfully updated the function AkarFunction.' The left sidebar shows the 'General configuration' section with sub-items: Triggers, Permissions, Destinations, Function URL, and Environment variables. The main area displays the 'General configuration' for the function, including a table with columns for Description, Memory, and Ephemeral storage. The 'Function URL' section is highlighted, showing a 'Create function URL' button.

Step 9: Then click on “Create function URL”.

The screenshot shows the AWS Lambda console interface with the 'Function URL' tab selected. The left sidebar shows the 'Function URL' section with sub-items: General configuration, Triggers, Permissions, Destinations, and Function URL. The main area displays the 'Function URL' configuration, including a 'Create function URL' button and a message stating 'No Function URL is configured.' Below this message, there is a 'Create function URL' button.

Step 10: Choose “NONE” and Save it.

Lambda > Functions > AkarFunction > Configure Function URL

Configure Function URL

Function URL [Info](#)

Use function URLs to assign HTTP(S) endpoints to your Lambda function.

Auth type

Choose the auth type for your function URL. [Learn more](#) [🔗](#)

☐ AWS_IAM
Only authenticated IAM users and roles can make requests to your function URL.

☒ NONE
Lambda won't perform IAM authentication on requests to your function URL. The URL endpoint will be public unless you implement your own authorization logic in your function.

Function URL permissions

i When you choose auth type **NONE**, Lambda automatically creates the following resource-based policy and attaches it to your function. This policy makes your function public to anyone with the function URL. You can edit the policy later. To limit access to authenticated IAM users and roles, choose auth type **AWS_IAM**.

► Additional settings

Cancel **Save**

Step 11: Now click on the Function URL.

🟢 Your changes have been saved. ✕


Lambda > Functions > AkarFunction


AkarFunction

Throttle Copy ARN Actions ▼

▼ **Function overview** [Info](#)

+ Add trigger


 AkarFunction


 Layers (0)

+ Add destination

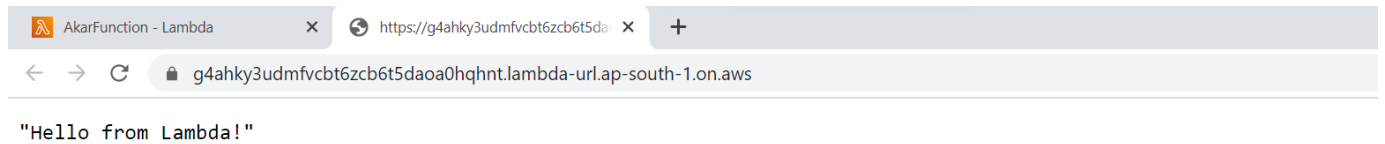
Description
-

Last modified
6 minutes ago

Function ARN
 arn:aws:lambda:ap-south-1:782829530958:function:AkarFunction

Function URL [Info](#)
 <https://g4ahky3udmfvcbt6zcb6t5daaa0hqhnt.lambda-url.ap-south-1.on.aws/> [🔗](#)

Step 12: We can see the content of the file.



Thus, we have successfully created a serverless computing service.