

St. Francis Institute of Technology, Mumbai-400 103
Department Of Information Technology

A.Y. 2021-2022
Class: TE-ITA/B, Semester: V
Subject: Advanced DevOps Lab

Experiment – 12: To create AWS Lambda function to log “an object has been added” on adding an object to s3 bucket.

1. Aim: To write the Lambda function to take specific action when invoked.

2. Objectives: Aim of this experiment is that, the students will learn:

- Serverless cloud concept
- To create Lambda function for specific application
- Configuring Lambda bucket as a Lambda event source.
- Trigger a Lambda function.
- Monitoring AWS Lambda

3. Outcomes: After study of this experiment, the students will learn following:

- Introduction to Serverless computing
- Creating Lambda function
- Testing Lambda function
- Monitoring and Logging Lambda function

4. Prerequisite: Knowledge of AWS S3, lambda functions and AWS console. **5.**

Requirements: AWS account, browser, Personal Computer, Windows operating system, Internet Connection, Google Doc.

6. Pre-Experiment Exercise:

Brief Theory: Refer shared material

7. Laboratory Exercise

A. Procedure:

a. Answer the following:

- Explain AWS Lambda execution role?

A Lambda function's execution role is an AWS Identity and Access Management (IAM) role that grants the function permission to access AWS services and resources. You provide this role when you create a function, and Lambda assumes the role when your function is invoked. You can create an execution role for development that has permission to send logs to Amazon CloudWatch and to upload trace data to AWS X-Ray.

To view a function's execution role

1. Open the Functions page on the Lambda console.
2. Choose a function.
3. Choose Configuration and then choose Permissions.
4. Under Resource summary, view the services and resources that the function can access.

5. Choose a service from the dropdown list to see permissions related to that service.

You can add or remove permissions from a function's execution role at any time, or configure your function to use a different role. Add permissions for any services that your function calls with the AWS SDK, and for services that Lambda uses to enable optional features.

When you add permissions to your function, make an update to its code or configuration as well. This forces running instances of your function, which have out-of-date credentials, to stop and be replaced.

- What is serverless computing?

Serverless is a way to describe the services, practices, and strategies that enable you to build more agile applications so you can innovate and respond to change faster. With serverless computing, infrastructure management tasks like capacity provisioning and patching are handled by AWS, so you can focus on only writing code that serves your customers. Serverless services like AWS Lambda come with automatic scaling, built-in high availability, and a pay-for-value billing model. Lambda is an event-driven compute service that enables you to run code in response to events from over 200 natively-integrated AWS and SaaS sources - all without managing any servers.

Move from idea to market, faster

Lower your costs

Adapt at scale

Build better applications, easier

Serverless applications have built-in service integrations, so you can focus on building your application instead of configuring it.

b. Execute following and attach screenshots:

Step 1: Create an iam role with 3 policies: s3fullaccess,cloudwatchfullaccess and AwsLambdaBasicExecutionRole

The screenshot shows the AWS IAM Dashboard. On the left sidebar, under 'Access management', 'Roles' is selected. In the main content area, there's a 'Create role' button at the top. Below it, a section titled 'Select type of trusted entity' shows four options: 'AWS service (EC2, Lambda and others)', 'Another AWS account (Belonging to you or 3rd party)', 'Web identity (OpenID Connect or any OpenID provider)', and 'SAML 2.0 federation (Your corporate directory)'. The 'AWS service' option is highlighted. A note below says 'Allows AWS services to perform actions on your behalf. Learn more.' Below this, a section titled 'Choose a use case' lists 'Common use cases': EC2 (Allows EC2 instances to call AWS services on your behalf.) and Lambda (Allows Lambda functions to call AWS services on your behalf.). Under 'Or select a service to view its use cases', a grid of services is shown in two columns:

API Gateway	CloudWatch Events	EMR	IoT SiteWise	RDS
AWS Backup	CodeBuild	EMR Containers	IoT Things Graph	Redshift
AWS Chatbot	CodeDeploy	ElastiCache	KMS	Rekognition
AWS Marketplace	CodeGuru	Elastic Beanstalk	Kinesis	Robotmaker
AWS Support	CodeStar Notifications	Elastic Container Registry	Lake Formation	S3
Amplify	Comprehend	Elastic Container Service	Lambda	SMS
AppStream 2.0	Config	Elastic Transcoder	Lex	SNS
AppSync	Connect	Elastic Load Balancing	License Manager	SWF
Application Auto Scaling	DMS	EventBridge	MQ	SageMaker
Application Discovery Service	Data Lifecycle Manager	Forecast	Machine Learning	Security Hub
	Data Pipeline	GameLift	Macie	Service Catalog

At the bottom of the page, there are 'Required' and 'Next: Permissions' buttons.

Click on roles.Click on Create Role and choose a use case as Lambda and click on next permissions.

This screenshot shows the 'Create role' wizard, Step 1: 'Select type of trusted entity'. It's a continuation of the previous screenshot. The 'AWS service' option is selected. Below it, a note says 'Allows AWS services to perform actions on your behalf. Learn more.' A 'Choose a use case' section follows, with 'Common use cases' listed: EC2 and Lambda. Under 'Or select a service to view its use cases', the same grid of services is shown as in the previous screenshot. At the bottom, there are 'Required' and 'Next: Permissions' buttons.

Select policies : AWSLambdaBasicExecutionRole and CloudWatchFull Aceess and AmazonS3Full access.

The screenshot shows the AWS Lambda Role creation process. In the first step, 'Attach permissions policies', the 'AmazonS3FullAccess' policy is selected. In the second step, 'Set permissions boundary', the 'CloudWatchFullAccess' policy is selected. Both steps show a list of available policies with checkboxes and a search bar.

Step 1: Attach permissions policies

Policy name	Used as
AmazonDMSRedshiftS3Role	None
<input checked="" type="checkbox"/> AmazonS3FullAccess	None
AmazonS3ObjectLambdaExecutionRolePolicy	None
AmazonS3OutpostsFullAccess	None
AmazonS3OutpostsReadOnlyAccess	None
AmazonS3ReadOnlyAccess	None
IVSRecordToS3	None
QuickSightAccessForS3StorageManagementAnalyticsReadOnly	None

Step 2: Set permissions boundary

Policy name	Used as
CloudWatchEventsFullAccess	None
CloudWatchEventsInvocationAccess	None
CloudWatchEventsReadOnlyAccess	None
CloudWatchEventsServiceRolePolicy	None
<input checked="" type="checkbox"/> CloudWatchFullAccess	None
CloudWatchLambdaInsightsExecutionRolePolicy	None
CloudWatchLogsFullAccess	None
CloudWatchLogsReadOnlyAccess	None

The screenshot shows the AWS IAM Management Console with a search bar at the top containing 'awslambdaBasic'. Below the search bar, a table displays two policy results:

Policy name	Used as
AWSLambdaBasicExecutionRole	None
AWSLambdaBasicExecutionRole-448f660b-9c1f-40b8-bbaf-8e3971d39f48	Permissions policy (1)

At the bottom left, there is a link labeled 'Set permissions boundary'.

Give Role Name: s3-put-role

The screenshot shows the 'Create role' review step in the AWS IAM Management Console. The role has been named 's3-put-role'. The 'Role description' field contains the text: 'Allows Lambda functions to call AWS services on your behalf.' The 'Trusted entities' section lists 'AWS service: lambda.amazonaws.com'. The 'Policies' section includes three policies: 'AmazonS3FullAccess', 'CloudWatchFullAccess', and 'AWSLambdaBasicExecutionRole'. A note at the bottom states 'Permissions boundary' and 'Permissions boundary is not set'.

Role s3_put_object_role has been created successfully. Step 1 done.

The screenshot shows the AWS IAM Management Console. In the top navigation bar, there are tabs for 'Meet - lab-share-ece', 'IAM Management Console', and 'EC2 Management Console'. Below the tabs, there's a search bar and a user dropdown. The main content area is titled 'Identity and Access Management (IAM)'. Under 'Access management', the 'Roles' section is selected, indicated by an orange highlight. A green success message at the top says 'The role s3-put-object-role has been created'. Below this, the 'Roles [4] Info' section displays four roles: 'AWSServiceRoleForSupport', 'AWSServiceRoleForTrustedAdvisor', 'lambda-basic-execution-role', and 's3-put-role'. The 's3-put-role' row shows it was last updated 25 minutes ago. At the bottom right of the table, there are 'Delete' and 'Create role' buttons. The left sidebar contains links for 'Identity providers', 'Account settings', 'Access reports', 'Access analyzer', 'Archive rules', 'Analyzers', 'Settings', 'Credential report', 'Organization activity', and 'Service control policies (SCPs)'. The bottom of the screen shows the Windows taskbar with various pinned icons like File Explorer, Task View, Mail, and Edge.

Step 2: Create an empty bucket using s3.

Go to services , s3, give globally unique bucket name, bucket versioning enable, create bucket.

The screenshot shows the AWS S3 Management Console. In the top navigation bar, there are tabs for 'Meet - lab-share-ece', 'S3 Management Console', and 'EC2 Management Console'. Below the tabs, there's a search bar and a user dropdown. The main content area is titled 'Amazon S3'. On the left, a sidebar lists 'Buckets', 'Access Points', 'Object Lambda Access Points', 'Multi-Region Access Points', 'Batch Operations', 'Access analyzer for S3', 'Block Public Access settings for this account', 'Storage Lens', 'Dashboards', 'AWS Organizations settings', 'Feature spotlight', and 'AWS Marketplace for S3'. A blue banner at the top says 'We're continuing to improve the S3 console to make it faster and easier to use. If you have feedback on the updated experience, choose Provide feedback.' and 'AWS Snow Family is a suite of highly-secure, portable devices equipped to transfer petabytes of data into Amazon S3.' Below this, the 'Buckets [0] Info' section shows a table with columns 'Name', 'AWS Region', 'Access', and 'Creation date'. A message says 'No buckets' and 'You don't have any buckets.' A 'Create bucket' button is located at the bottom of the table. The bottom of the screen shows the Windows taskbar with various pinned icons like File Explorer, Task View, Mail, and Edge.

The screenshot shows the AWS S3 'Create bucket' wizard. The first section, 'General configuration', includes fields for 'Bucket name' (set to 'bucket-lambda-s3'), 'AWS Region' (set to 'Asia Pacific (Mumbai) ap-south-1'), and a 'Copy settings from existing bucket - optional' section with a 'Choose bucket' button. The second section, 'Block Public Access settings for this bucket', contains a 'Add tag' button. The third section, 'Default encryption', has a 'Server-side encryption' dropdown where 'Disable' is selected. A '► Advanced settings' button is also present. A note at the bottom states: 'After creating the bucket you can upload files and folders to the bucket, and configure additional bucket settings.' At the bottom right are 'Cancel' and 'Create bucket' buttons.

Step 2 done i.e. empty Bucket created successfully.

The screenshot shows two consecutive screenshots of the AWS S3 Management Console.

Screenshot 1 (Top): The user has just created a new bucket named "bucket-lambda-s3". A green success message at the top of the page states: "Successfully created bucket 'bucket-lambda-s3' To upload files and folders, or to configure additional bucket settings choose View details." Below this, there is an "Account snapshot" section and a table showing the newly created bucket.

Name	AWS Region	Access	Creation date
bucket-lambda-s3	Asia Pacific (Mumbai) ap-south-1	Bucket and objects not public	September 7, 2021, 14:31:50 (UTC+05:30)

Screenshot 2 (Bottom): The user is viewing the contents of the "bucket-lambda-s3" bucket. The "Objects" tab is selected. The page displays a message: "No objects". There is a prominent "Upload" button at the bottom of the object list area.

Step 3: Create a Lambda Function

The screenshot shows the 'Create function' wizard in the AWS Lambda console. The first step, 'Basic information', is displayed. In the 'Function name' field, the value 'myfunctionname' is entered. The 'Runtime' dropdown is set to 'Node.js 14.x'. The 'Permissions' section indicates that Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs.

Basic information

Function name: myfunctionname

Runtime: Node.js 14.x

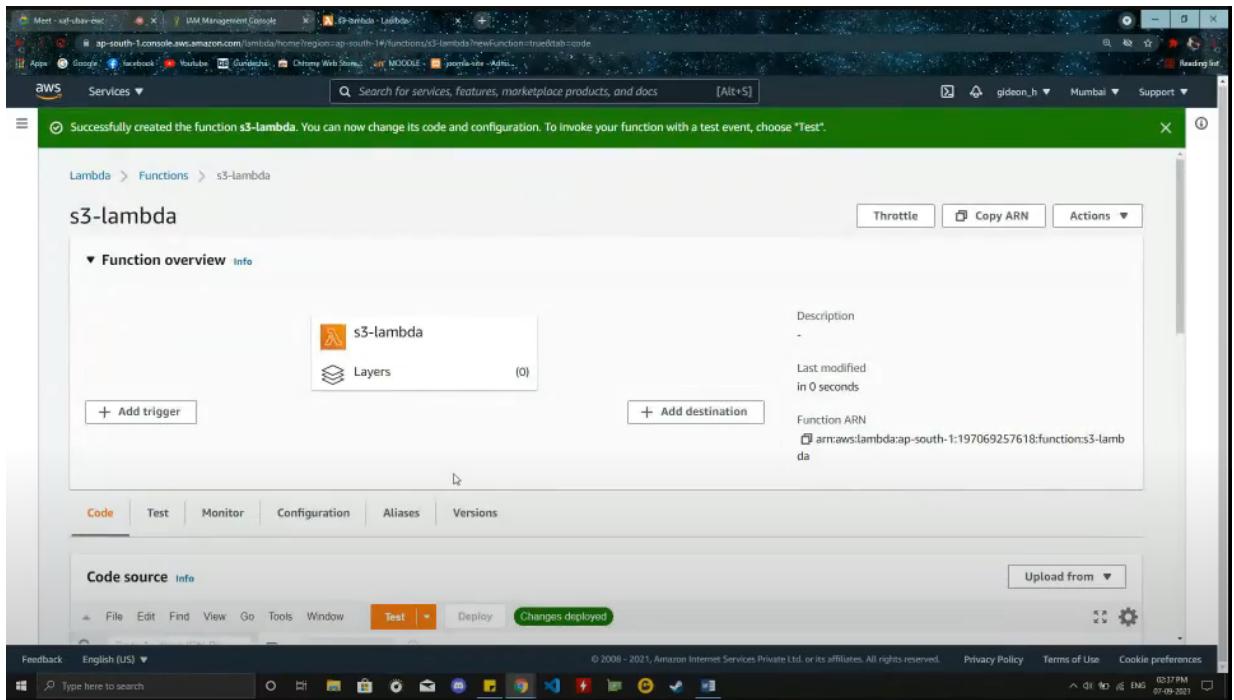
Permissions: By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

The screenshot shows the 'Advanced settings' step of the wizard. Under 'Execution role', the 'Use an existing role' option is selected, and the role 's3-put-role' is chosen from the dropdown. The 'Create function' button is visible at the bottom right.

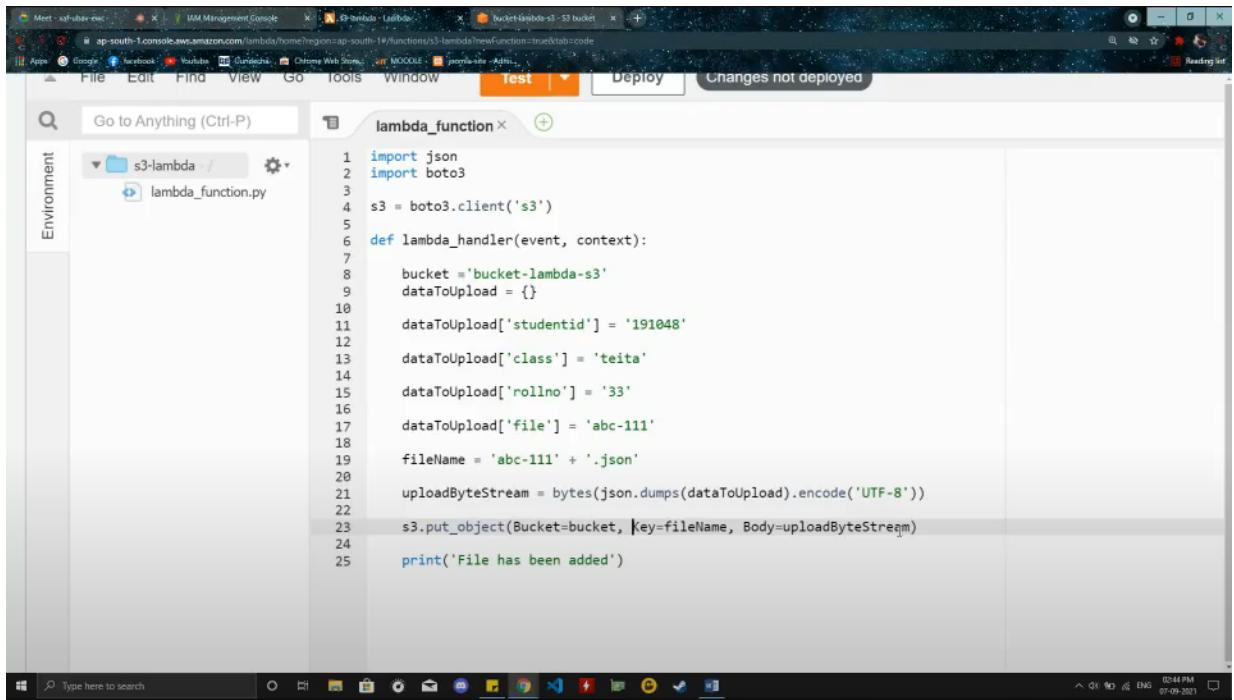
Execution role: Use an existing role: s3-put-role

Create function

Successfully created a function

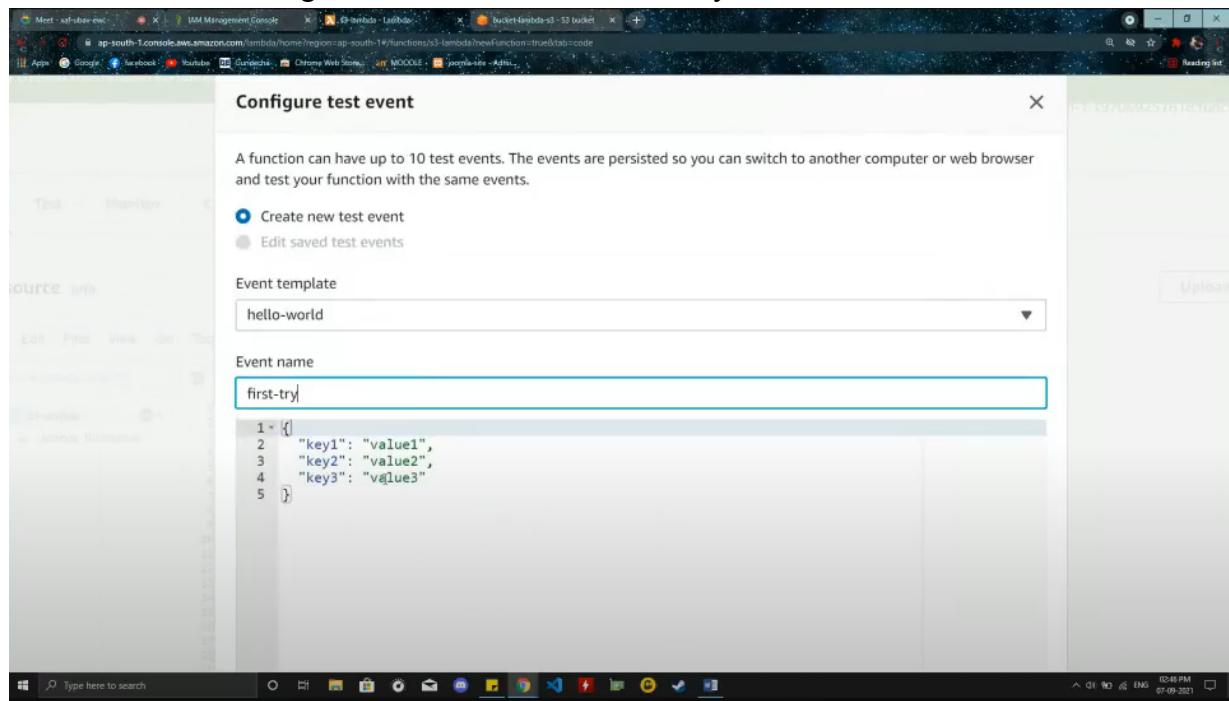


Write following script as lambda function

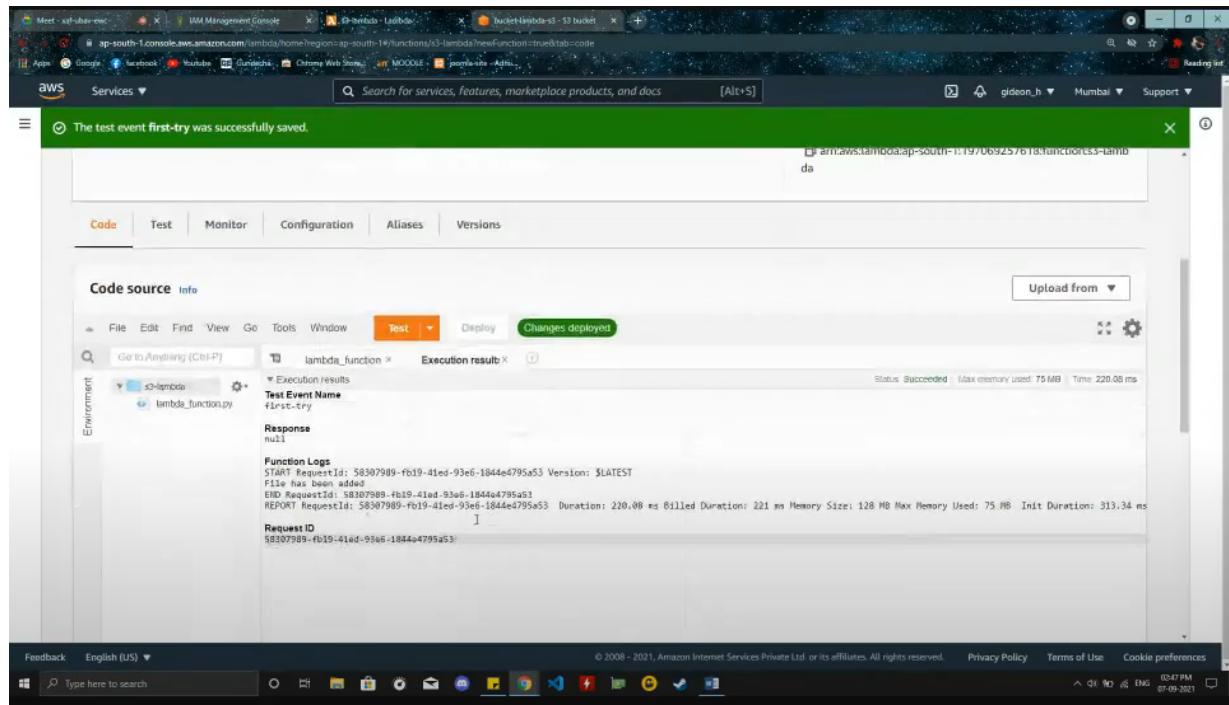


Go to File save and then click on deploy.

Now click on Test and give the event name as secondary.



Click again on Test , it will generate an execution report of function.



Step 4: Services, go to cloudwatch,logs,log groups. Click on the log group. Click on the latest log stream and check the output.

The screenshot shows the AWS CloudWatch Log Groups page. On the left, there's a sidebar with 'Log groups' selected. The main area displays two log groups: '/aws/lambda/gideon-helloWorld' and '/aws/lambda/s3-lambda'. The second log group, '/aws/lambda/s3-lambda', has a cursor pointing at it. At the top right, there are buttons for 'Actions', 'View in Logs Insights', and 'Create log group'.

Now we have to check bucket

The screenshot shows the AWS CloudWatch Log Events page for the '/aws/lambda/s3-lambda' log group. The sidebar on the left is identical to the previous screenshot. The main content area shows log events for file uploads. One event is expanded, showing the message 'File has been added'. There are 'Copy' buttons next to each event message. A note at the bottom says 'No newer events at this moment. Auto retry paused. Resume'.

Step 5 : Go to s3...now check our created empty bucket. abc-111.json object is there in our bucket.

Name	AWS Region	Access	Creation date
bucket-lambda-s3	Asia Pacific (Mumbai) ap-south-1	Bucket and objects not public	September 7, 2021, 14:31:50 (UTC+05:30)

Click on object abc-111 and right most corner , click on open
It will start downloading and open that file with notepad ...check the output...all fields will have values.

Name	Type	Last modified
abc-111.json	json	September 7, 2021, 14:47:13 (UTC+05:30)

The screenshot shows a Windows desktop environment with three main windows:

- AWS Management Console (Top Window):** The user is viewing the "Object overview" for the file "abc-111.json" located in the "bucket-lambda-s3" bucket. The file is a JSON file with the following content:

```
{"studentid": "191048", "class": "teita", "rollno": "33", "file": "abc-111"}
```
- File Explorer Context Menu (Middle Window):** A context menu is open over a file named "abc-111.json" in the "AWS Marketplace" folder. The menu options include "Open when done", "Always open files of this type", "Pause", and "Show in folder".
- Notepad (Bottom Window):** The Notepad application is open, displaying the same JSON content as the S3 object.

The taskbar at the bottom of the screen shows various pinned icons and the system clock indicating 02:49 PM on 07-09-2021.

Now delete lambda functions,s3 bucket and free up all other resources.

The screenshot shows the AWS Lambda Functions page. The left sidebar has a tree view with 'AWS Lambda' selected, showing 'Functions' (selected), 'Additional resources', and 'Related AWS resources'. The main content area shows a table titled 'Functions (0)'. The table has columns: Function name, Description, Package type, Runtime, Code size, and Last modified. A message at the bottom of the table says 'There is no data to display.'

8. Post-Experiments Exercise

A. Extended Theory:

Nil

B. Questions:

GIDEON HARPANAHALLI

DOMS Page No.

Date / /

AWS ADL EXP 12 TEITA [33]

B] Questions: What are the components of AWS Lambda?

① Building Blocks of Lambda Function

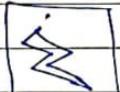
Ans: The components are as follows:

→ Event source

→ Functions

→ Services

Event Source



- Request to API endpoints
- Change in data state
- Change in resource state

Function



Services → Lambda function interacts with databases or AWS services or third-party services.



Lambda function interacts with databases or AWS services or third-party services.

(i) Event Source :- They are known as trigger. It is the starting point of an AWS serverless function and workflow. It triggers execution of function. It can be S3, SNS, SQS, CloudWatch logs, etc.

(ii) Lambda Function :- It is the core component of an AWS serverless architecture.

(iii) Services :- A Lambda function is not sufficient to fulfill the needs of a business use case. It requires services like DB, auth, queues and third party system.

DOMS	Page No.
Date	/ /

TEITA [33]

EXP. 10: AWS

- Q) What are the limitations of AWS Lambda
- i) The disk space is limited to 512 MB
 - ii) The default deployment package size is 50 MB
 - iii) Memory range is 128 to 1536 MB
 - iv) Maximum execution time for a function is 15 minutes
 - ✓ Event request body can be up to 6 MB

C. Conclusion:

- Write what was performed in the experiment.
- Write the significance of the topic studied in the experiment.

TEITA [33]

D CONCLUSION

①

- We created an AWS Lambda function to log an object that has been added or added an object to S3 bucket.
- We created S3 bucket, a Lambda function & then tested the function log.

②

- Lambda function helps us to run code without creating, managing or paying for servers.
- Helps to integrate security model, connect to shared file system, automatic scaling & other features.

D. References:

<https://intellipaat.com/blog/tutorial/amazon-web-services-aws-tutorial/aws-lambda-tutorial/>