

ASSIGNMENT 10

AIM: To understand AWS Lambda Functions and create a Lambda function using Python to log "An Image has been added" message, once a file is added to an S3 Bucket.

THEORY:

AWS Lambda is a serverless computing service provided by Amazon Web Services (AWS). It allows you to run code in response to events without having to provision or manage servers. AWS Lambda functions are small, self-contained units of code that can be executed in the cloud, triggered by various AWS services or external events.

Key features of AWS Lambda functions include:

1. **Event-Driven:** Lambda functions are designed to be event-driven. They can be triggered by various events such as changes to data in Amazon S3, updates to a DynamoDB table, HTTP requests through Amazon API Gateway, and more.
2. **Scaling:** AWS Lambda automatically scales your functions in response to incoming traffic. You don't need to worry about server provisioning or capacity planning.
3. **Pay-as-You-Go:** With Lambda, you pay only for the compute time consumed during function execution. There are no upfront costs or infrastructure management fees.
4. **Support for Multiple Languages:** You can write Lambda functions in various programming languages, including Node.js, Python, Java, C#, Ruby, Go, and custom runtimes.
5. **Stateless and Isolated:** Each Lambda function is stateless and runs in isolation, ensuring that one function's execution doesn't affect another. Data can be stored and retrieved using other AWS services like S3 or DynamoDB.
6. **Easy Integration:** Lambda functions can be easily integrated with other AWS services, creating serverless architectures and event-driven workflows.
7. **Custom Runtimes:** You can use custom runtimes to execute functions written in languages not officially supported by AWS.

AWS Lambda is a powerful tool for building serverless applications, automating tasks, and responding to events in a highly scalable and cost-effective manner.

Name: Vignesh Pai

Roll no: 72

Batch: T21

The image consists of two screenshots of the Amazon S3 console interface.

Top Screenshot: Buckets List

The top screenshot shows the 'Buckets (1)' page in the Amazon S3 console. It displays a table with one bucket:

Name	AWS Region	Access	Creation date
codepipeline-eu-north-1-530481351972	Europe (Stockholm) eu-north-1	Bucket and objects not public	August 8, 2023 16:18:22 (UTC+05:30)

Bottom Screenshot: Create bucket wizard

The bottom screenshot shows the 'Create bucket' wizard. The 'General configuration' section is visible, showing the following details:

- Bucket name:** triggermagetuckets5
- AWS Region:** Europe (Stockholm) eu-north-1

The 'Object Ownership' section is partially visible below.

Name: Vignesh Pai

Roll no: 72

Batch: T21

The image shows two screenshots from the AWS Management Console. The top screenshot is the 'Create bucket' wizard, and the bottom screenshot is the 'Buckets' list page.

Top Screenshot: Create bucket wizard

Default encryption info
Server-side encryption is automatically applied to new objects stored in this bucket.

Encryption type
[info](#)

- ☒ Server-side encryption with Amazon S3-managed keys (SSE-S3)
- ☐ Server-side encryption with AWS Key Management Service keys (SSE-KMS)
- ☐ Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)
Secure your objects with two separate layers of encryption. For details on pricing, see DSSE-KMS getting on the Storage tab of the [Amazon S3 pricing page](#).

Bucket Key
Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for SSE-S3. [Learn more](#)

- ☐ Disable
- ☒ Enable

Advanced settings

After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.

[Cancel](#) [Create bucket](#)

Bottom Screenshot: Buckets list

Successfully created bucket "triggerimagebucket53"
To upload files and folders, or to configure additional bucket settings choose [View details](#).

Account snapshot
Storage lens provides visibility into storage usage and activity trends. [Learn more](#) [View Storage Lens dashboard](#)

Buckets (2)
Buckets are containers for data stored in S3. [Learn more](#)

[Refresh](#) [Copy ARN](#) [Empty](#) [Delete](#) [Create bucket](#)

Name	AWS Region	Access	Creation date
codepipeline-eu-north-1-130481351872	Europe (Stockholm) eu-north-1	Bucket and objects not public	August 8, 2023, 16:19:22 (UTC+05:30)
triggerimagebucket53	Europe (Stockholm) eu-north-1	Bucket and objects not public	August 29, 2023, 15:10:31 (UTC+05:30)

Name: Vignesh Pai

Roll no: 72

Batch: T21

The image displays three sequential screenshots of the AWS IAM console interface, showing the process of creating a new role.

Screenshot 1: IAM Dashboard

- Left Sidebar:** Identity and Access Management (IAM) navigation menu including Dashboard, Access management (User groups, Users, Roles, Policies, Identity providers, Account settings), and Access reports (Access analyzer, Archive rules, Analytics).
- Main Content Area:**
 - IAM Dashboard:** Overview of IAM resources in the AWS account.
 - Security recommendations:** Alerts for adding MFA for the root user and ensuring the root user has no active access keys.
 - IAM resources:** Summary of resources in the AWS account: User groups (2), Users (1), Roles (8), Policies (2), and Identity providers (0).
 - AWS Account:** Account ID (558767473830), Account Alias, and Sign-in URL.
 - Quick Links:** My security credentials and Manage your access keys, multi-factor authentication (MFA) and other.

Screenshot 2: Select trusted entity

- Trusted entity type:** Selection screen for the role's trusted entity type.
 - AWS service:** Selected option. Description: Allow AWS services like EC2, Lambda, or others to perform actions in this account.
 - AWS account:** Allow entities in other AWS accounts belonging to you or a third party to perform actions in this account.
 - Web identity:** Allow users federated by the specified external web identity provider to assume this role to perform actions in this account.
 - SAML 2.0 federation:** Allow users federated with SAML 2.0 from a third-party identity provider to perform actions in this account.
 - Custom trust policy:** Create a custom trust policy to enable others to perform actions in this account.
- Use case:** Allow an AWS service like EC2, Lambda, or others to perform actions in this account.
- Common use cases:**
 - EC2:** Allow EC2 instances to call AWS services on your behalf.

Screenshot 3: Use case selection

- Common use cases:**
 - EC2:** Allow EC2 instances to call AWS services on your behalf.
 - Lambda:** Allow Lambda functions to call AWS services on your behalf.
- Use cases for other AWS services:** Create a service to view user cases.
- Buttons:** Cancel and Next.

Name: Vignesh Pai

Roll no: 72

Batch: T21

The image shows two screenshots of the AWS IAM console. The top screenshot shows the 'Set permissions boundary - optional' step, where a list of policies is displayed. The bottom screenshot shows the 'Attach permissions' step, where a list of policies is displayed.

Top Screenshot: Set permissions boundary - optional

Filter policies by property or policy name and press enter: 8 matches

Policy name	Type	Description
AmazonS3FullAccess	AWS m...	Provides full access to all buckets via the AWS Management Console.
AmazonS3ReadOnlyAccess	AWS m...	Provides read only access to all buckets via the AWS Management Console.
AmazonDMSRedshift	AWS m...	Provides access to manage S3 settings for Redshift endpoints for DMS.
QuickSightAccessForS3	AWS m...	Policy used by QuickSight team to access customer data produced by S3.
AmazonS3OutpostsFullAccess	AWS m...	Provides full access to Amazon S3 on Outposts via the AWS Management Console.
AmazonS3OutpostsReadOnlyAccess	AWS m...	Provides read only access to Amazon S3 on Outposts via the AWS Management Console.
AmazonS3OutpostsFullAccess	AWS m...	Provides AWS Lambda functions permissions to interact with Amazon S3 on Outposts.
AWSBackupServiceFullAccess	AWS m...	Policy containing permissions necessary for AWS Backup to restore a S3 bucket.
AWSBackupServiceReadOnlyAccess	AWS m...	Policy containing permissions necessary for AWS Backup to backup data from S3.

Bottom Screenshot: Attach permissions

Choose one or more policies to attach to your new role:

Filter policies by property or policy name and press enter: 20 matches

Policy name	Type	Description
CloudWatchFullAccess	AWS m...	Provides full access to CloudWatch.
CloudWatchReadOnlyAccess	AWS m...	Provides read only access to CloudWatch.
CloudWatchLogsFullAccess	AWS m...	Provides full access to CloudWatch Logs.
CloudWatchLogsReadOnlyAccess	AWS m...	Provides read only access to CloudWatch Logs.
CloudWatchActions	AWS m...	Provides read-only access to CloudWatch alarms and metrics as well as EC2 instances.
AmazonAPIGatewayPushToCloudWatchLogs	AWS m...	Allows API Gateway to push logs to user's account.
AmazonDMSReplicationTaskLogsToCloudWatch	AWS m...	Provides access to upload DMS replication logs to cloudwatch logs in cost.
CloudWatchEventsFullAccess	AWS m...	Provides read only access to Amazon CloudWatch Events.
CloudWatchEventsReadOnlyAccess	AWS m...	Allows built-in targets in Amazon CloudWatch Events to perform EC2 actions.
CloudWatchEventsFullAccess	AWS m...	Allows Amazon CloudWatch Events to relay events to the streams in AWS.

Batch: T21



Name: Vignesh Pai

Roll no: 72

Batch: T21

The image shows two screenshots. The top screenshot is the AWS Lambda website landing page. It features a dark blue header with the text "AWS Lambda lets you run code without thinking about servers." and a "Get started" section with a "Create a function" button. Below this is a "How it works" section with a "Run" button and a dropdown menu for selecting a runtime (Python, Node.js, etc.). The bottom screenshot is the AWS Lambda console dashboard for the Stockholm region. It displays "Resources for Europe (Stockholm)" with metrics for Lambda functions (0), Code storage (0 byte), Full account concurrency (10), and Unreserved account concurrency (10). It also shows "Account-level metrics" with charts for Error count and success rate, Throttles, and Invocations, all of which currently show "No data available".

AWS Lambda website landing page:

- Header: Compute
- Main heading: **AWS Lambda**
- Subheading: lets you run code without thinking about servers.
- Text: You pay only for the compute time that you consume — there is no charge when your code is not running. With Lambda, you can run code for virtually any type of application or backend service, all with zero administration.
- Get started section: Author a Lambda function from scratch, or choose from one of many preconfigured examples. [Create a function](#)
- How it works section: [Run](#) Next: Lambda responds to events
- Runtime selection: .NET, Go, Java, **Node.js**, Python, Ruby, Custom runtime

AWS Lambda console dashboard:

- Region: Stockholm
- Resources for Europe (Stockholm): [Create function](#)
- Metrics:
 - Lambda functions: 0
 - Code storage: 0 byte (0% of 75.0 GB)
 - Full account concurrency: 10
 - Unreserved account concurrency: 10
- Account-level metrics: The charts below show metrics across all your Lambda functions in the AWS Region.
- Time range: 1h, 3h, 12h, 1d, 3d, 1w, Custom
- Charts:
 - Error count and success rate: No unit, No data available. Try adjusting the dashboard time range.
 - Throttles: No unit, No data available. Try adjusting the dashboard time range.
 - Invocations: No unit, No data available. Try adjusting the dashboard time range.

Name: Vignesh Pai

Roll no: 72

Batch: T21

The screenshot shows the 'Create function' page in the AWS Lambda console. The 'Author from scratch' option is selected. The function name is 'triggerimagelambda'. The runtime is set to 'Node.js 18.x'. The architecture is set to 'x86_64'.

Create function [info](#)

Choose one of the following options to create your function:

- ☒ **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
- ☐ **Browse serverless app repository**
Deploy a sample Lambda application from the AWS Serverless Application Repository

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.

Architecture [info](#)
Choose the instruction set architecture you want for your function code.
☒ **x86_64**
☐ **arm64**

Permissions [info](#)
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default behavior when adding triggers.

[Change default execution role](#)

[Advanced settings](#)

Name: Vignesh Pai

Roll no: 72

Batch: T21

Permissions [info](#)

By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default behavior when setting 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.

[View the triggerimgelambda53 role](#) on the IAM console

Advanced settings

[Cancel](#) [Create Function](#)

CloudShell Feedback Language © 2023, Amazon Web Services India Private Limited or its affiliates. Privacy Terms Cookie preferences 31°C Mostly cloudy 3:24 PM 8/24/2023

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

triggerimgelambda [Throttle](#) [Copy ARN](#) [Actions](#)

Function overview [info](#)

triggerimgelambda

Layers (0)

[+ Add trigger](#) [+ Add destination](#)

Description
-

Last modified
15 seconds ago

Function ARN
[arn:aws:lambda:eu-north-1:538767473030:function:triggerimgelambda](#)

Function URL [info](#)

CloudShell Feedback Language © 2023, Amazon Web Services India Private Limited or its affiliates. Privacy Terms Cookie preferences 31°C Mostly cloudy 3:25 PM 8/24/2023

Name: Vignesh Pai

Roll no: 72

Batch: T21

The screenshot displays the AWS Lambda console interface. At the top, there's a configuration window for adding a trigger. It includes fields for 'Key' (with a dropdown), 'Prefix' (optional, e.g., 'images/'), and 'Suffix' (optional, e.g., '.jpg'). A 'Recursive invocation' section has a checkbox checked, with a warning that using the same S3 bucket for both input and output is not recommended. Below this, a 'Cancel' button and an 'Add' button are visible.

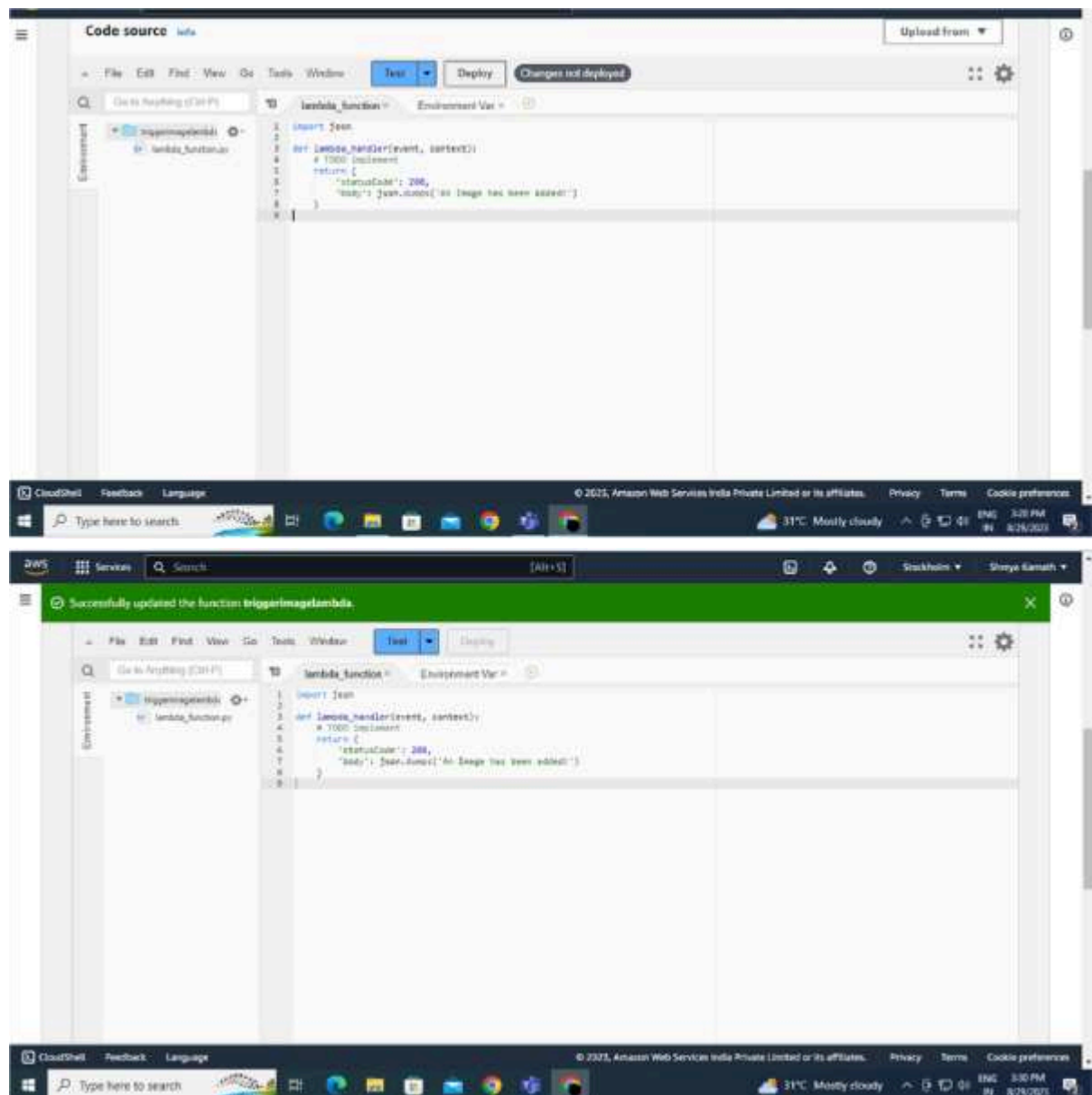
The main console area shows the 'triggerimagemagicklambda' function. A green notification banner states: 'The trigger triggerimagebucketS3 was successfully added to function triggerimagemagicklambda. The function is now receiving events from the trigger.' Below this, the 'Function overview' section shows a diagram of the function 'triggerimagemagicklambda' with a layer and an S3 trigger. A '+ Add trigger' button is present. To the right, a sidebar provides details: 'Description', 'Last modified: 2 minutes ago', 'Function ARN: arn:aws:lambda:us-east-1:538767473830:function:triggerimagemagicklambda', and 'Function URL: test'.

The bottom of the image shows the Windows taskbar with various application icons and the system clock indicating 3:27 PM on 8/24/2023.

Name: Vignesh Pai

Roll no: 72

Batch: T21



Go back to s3

Name: Vignesh Pai

Roll no: 72

Batch: T21

The image consists of two screenshots of the AWS CloudShell interface, showing the process of uploading files to an Amazon S3 bucket.

Top Screenshot: Upload Configuration

The top screenshot shows the "Files and folders" upload configuration window. At the top, it says "Files and folders (1 Total, 840.9 KB)" with buttons for "Remove", "Add files", and "Add folder". Below this is a search bar labeled "Find by name". A table header shows columns for "Name", "Folder", "Type", and "Size". The "Destination" section shows the target S3 bucket: "s3://vigneshmagetbucket53". Below this are expandable sections for "Destination details", "Permissions", and "Properties". At the bottom right are "Cancel" and "Upload" buttons.

Bottom Screenshot: Upload Status

The bottom screenshot shows the "Upload: status" page after a successful upload. A green banner at the top says "Upload succeeded" with a "View details below" link. Below this is a summary table:

Summary		
Destination s3://vigneshmagetbucket53	Successful 1 file, 840.9 KB (100.00%)	Failed 0 files, 0 B (0%)

Below the summary are tabs for "Files and folders" (selected) and "Configuration". The "Files and folders" tab shows the same "Files and folders (1 Total, 840.9 KB)" header.

Name: Vignesh Pai

Roll no: 72

Batch: T21

The image consists of two screenshots from the AWS IAM console, showing the process of adding permissions to a role.

Top Screenshot: The 'Permissions' tab is selected. It shows a list of 'Permissions policies (2)' with columns for Policy name, Type, and Description. The policies listed are 'CloudWatchFullAccess' and 'AmazonS3FullAccess'. A dropdown menu is open, showing options: 'Add permissions', 'Attach policy', and 'Create new policy'. The 'Attach policy' option is highlighted.

Bottom Screenshot: The 'Add permissions' page is shown. It displays the 'Current permissions policies (2)' and a list of 'Other permissions policies (Selected 1/674)'. The 'AmazonEventBridgeFullAccess' policy is selected. The 'Attach' button is visible. At the bottom, there are 'Cancel' and 'Add permissions' buttons.

Name: Vignesh Pai

Roll no: 72

Batch: T21

The screenshot shows the AWS IAM console interface. The left sidebar contains navigation links for Identity and Access Management (IAM), including Dashboard, Access management (User groups, Users, Roles, Policies, Identity providers, Account settings), and Access reports (Access analyzer, Archive rules, Analyzers). The main content area is titled 'Permissions' and shows 'Permissions policies (3)'. It lists three AWS managed policies:

Policy name	Type	Description
CloudWatchFullAccess	AWS managed	Provides full access to CloudWatch
AmazonS3FullAccess	AWS managed	Provides full access to all buckets
AmazonEventBridgeFullAccess	AWS managed	Provides full access to Amazon EventBridge

Below the policy list, the 'Permissions boundary' is shown as '(not set)'. The top of the console shows the user 'Shreya Kamath' and the 'Global' region. The bottom of the image shows a Windows taskbar with various application icons and system information like 31°C and 4:02 PM on 8/28/2023.

Name: Vignesh Pai

Roll no: 72

Batch: T21

The image consists of two screenshots from the AWS Management Console, showing the configuration of a Lambda function and its destination.

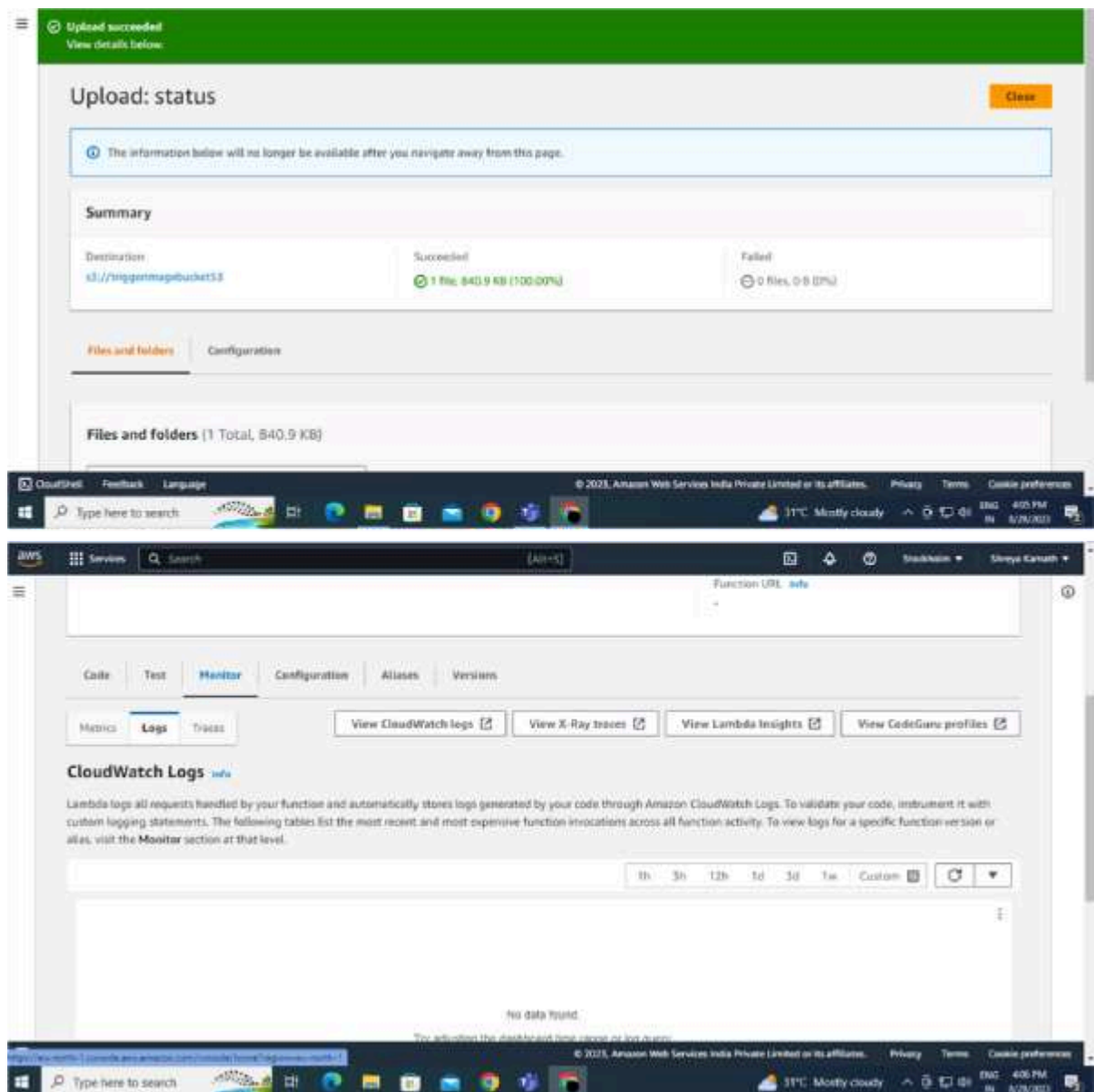
Top Screenshot: AWS Lambda Console - Function Overview

- Function Name:** triggerimagelambda
- Runtime:** Python 3.9
- Architecture:** x86_64
- Region:** us-east-1
- Triggers:** One trigger is configured using Amazon EventBridge.
- Destinations:** One destination is configured using Amazon S3.
- Function ARN:** arn:aws:lambda:us-east-1:528767473830:function:triggerimagelambda
- Function URL:** [Info](#)

Bottom Screenshot: AWS S3 Console - Destination Details

- Destination:** s3://triggerimagebucket55
- Destination details:** Bucket settings that impact new objects stored in the specified destination.
- Permissions:** Grant public access and access to other AWS accounts.
- Properties:** Specify storage class, encryption settings, tags, and more.

Name: Vignesh Pai
Roll no: 72
Batch: T21



CONCLUSION:

Hence, I have understood AWS Lambda Functions and created a Lambda function using Python to log "An Image has been added" message, once a file is added to an S3 Bucket.