

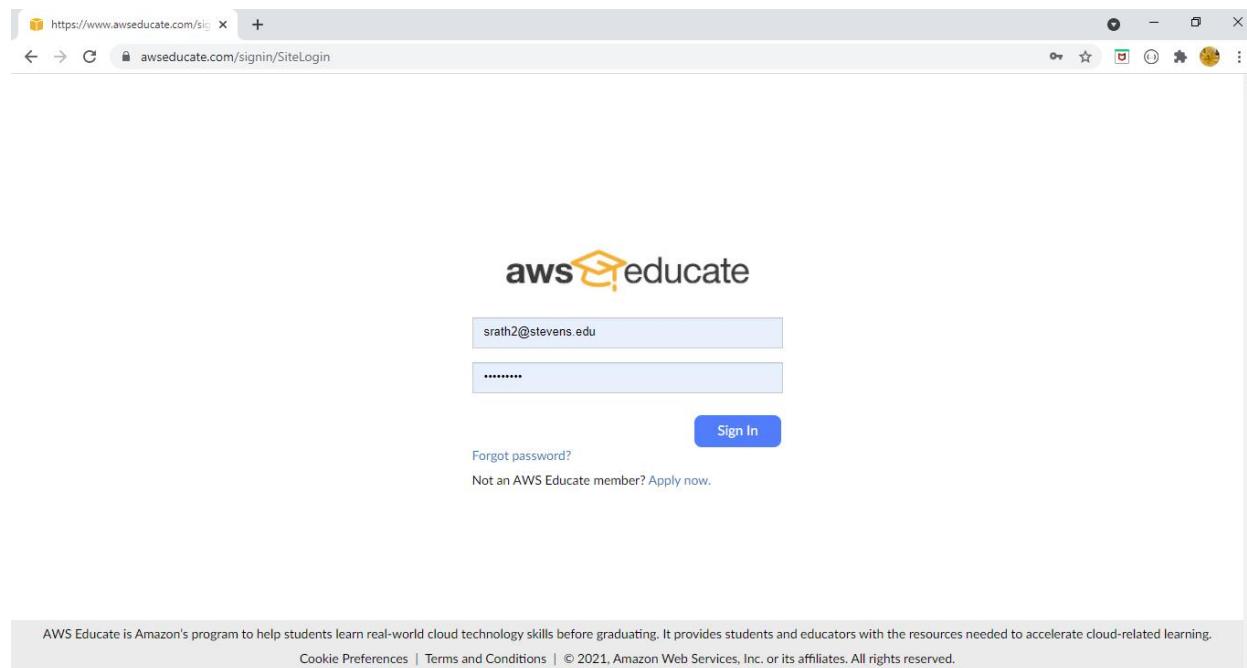
Microservice architecture – a variant of the [service-oriented architecture](#) (SOA) structural style – arranges an application as a collection of [loosely coupled](#) services. In a microservices architecture, services are fine-grained and the protocols are lightweight. This concept (i.e., a separate “small” service for each “simple” task) has been under consideration since early SOA times.

Application programming interface (API) is an [interface](#) that defines interactions between multiple [software applications](#) or mixed [hardware-software](#) intermediaries.^[1] It defines the kinds of [calls](#) or requests that can be made, how to make them, the [data formats](#) that should be used, the conventions to follow, etc. It can also provide extension mechanisms so that users can extend existing functionality in various ways and to varying degrees.

A **REST API** (also known as RESTful API) is an application programming interface (API or web API) that conforms to the constraints of REST architectural style and allows for interaction with RESTful web services.

To create AWS lambda function I have performed below steps

Login into AWS educate account



Screenshot of the AWS Educate student dashboard.

Header:

- Home
- awseducate.com/student/s/
- Logout

User Profile:

- Smita Rath
- Consecutive Days: 2
- Pathways Completed: 0
- Badges Earned: 0
- Preferred Language: English

Main Content:

Cloud technology is everywhere, creating over 18 million cloud jobs worldwide (source: Wanted Analytics). AWS Educate introduces you to lucrative cloud-enabled careers through more than 25 learning pathways, each with content from industry professionals, learning activities and labs, opportunities to earn AWS Educate Badges and Certificates of Completion, and access to the AWS Educate Job Board. Coupled with courses at your school or through online providers, AWS Educate puts you on the pathway to your dream job in the clouds.

Begin your journey today!

Search Thousands of Cloud Jobs and Internships on the AWS Educate Job Board

As an AWS Educate member, you now have access to a new job board experience meant to make finding a cloud career more streamlined. Search cloud-related jobs from Amazon and employers in the AWS Partner Network.

Learn More

Suggested Jobs:

- Associate Consultant at Slalom
- Deloitte Consulting Solutions Engineering Analyst at Deloitte Consulting

See More

Screenshot of the AWS Educate My Classrooms page.

Header:

- My Classrooms
- awseducate.com/student/s/classrooms
- Logout

User Profile:

- Smita Rath
- Consecutive Days: 2
- Pathways Completed: 0
- Badges Earned: 0
- Preferred Language: English

Section Header:

My Classrooms

View your list of Classroom invitations and accept or decline the invitation. Access a Classroom by clicking Go to my classroom.

| Course Name | Description | Educator | Course End Date | Credit Allocated Per Student | Status |
|---------------------------------|---|---------------|-----------------|------------------------------|---|
| Introduction to Cloud Computing | Full introduction to Cloud technologies and economics with the lab work on AWS. | Igor Faynberg | 05/19/2021 | \$100 | Accepted Go to classroom |

Page Footer:

AWS Support Forum | Contact Us | Cookie Preferences | Terms and Conditions

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

The screenshot shows the Vocareum Workbench interface. At the top, it says "Welcome to your AWS Educate Account". Below this, there's a section for "Your AWS Account Status" with three items: "Active" (full access), "\$95.05" (remaining credits estimated), and "2:60" (session time). There are two buttons at the bottom: "Account Details" and "AWS Console" (which is highlighted). A note below the status says: "Please use AWS Educate Account responsibly. Remember to shut down your instances when not in use to make the best use of your credits. And, don't forget to logout once you are done with your work!"

Go to AWS management console and select Lambda

The screenshot shows the AWS Management Console homepage. The left sidebar has sections for "Recently visited services" (S3, Lambda, CloudFront, Billing, CloudWatch, IAM, EC2, Simple Queue Service, API Gateway, CloudFormation, VPC) and "All services" (Compute, Quantum Technologies, Security, Identity, & Compliance, IAM, Resource Access, EC2, Lightsail, Lambda, Amazon Braket). The main content area features a "Stay connected to your AWS resources on-the-go" section about the AWS Console Mobile App, an "Explore AWS" section with "Free AWS Training" (with a link to 500+ courses) and "Build Serverless Apps with Infrastructure as Code", and a footer with links for Feedback, English (US), Privacy Policy, Terms of Use, and Cookie preferences.

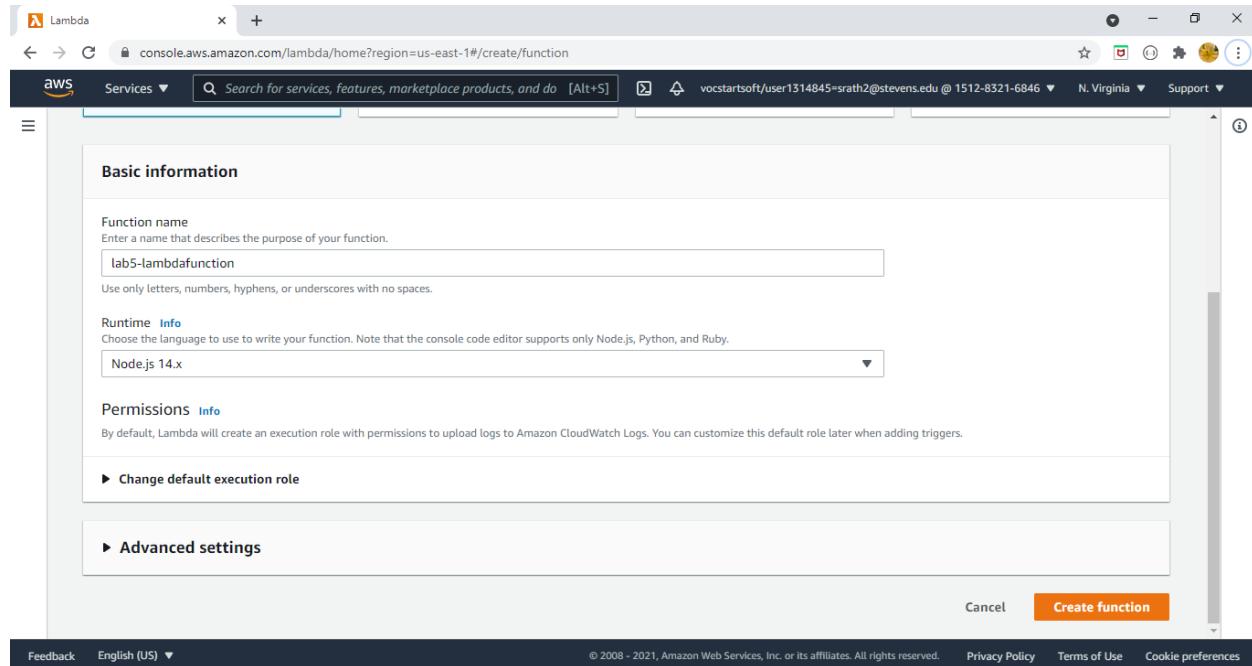
Click on click lambda

The screenshot shows the AWS Lambda Functions page. On the left, a sidebar menu includes 'Dashboard', 'Applications', 'Functions' (which is selected and highlighted in orange), 'Additional resources', and 'Related AWS resources'. The main content area is titled 'Functions (0)' and displays a message: 'There is no data to display.' A search bar at the top of the content area allows filtering by tags and attributes or searching by keyword. A prominent orange button labeled 'Create function' is located at the top right of the content area.

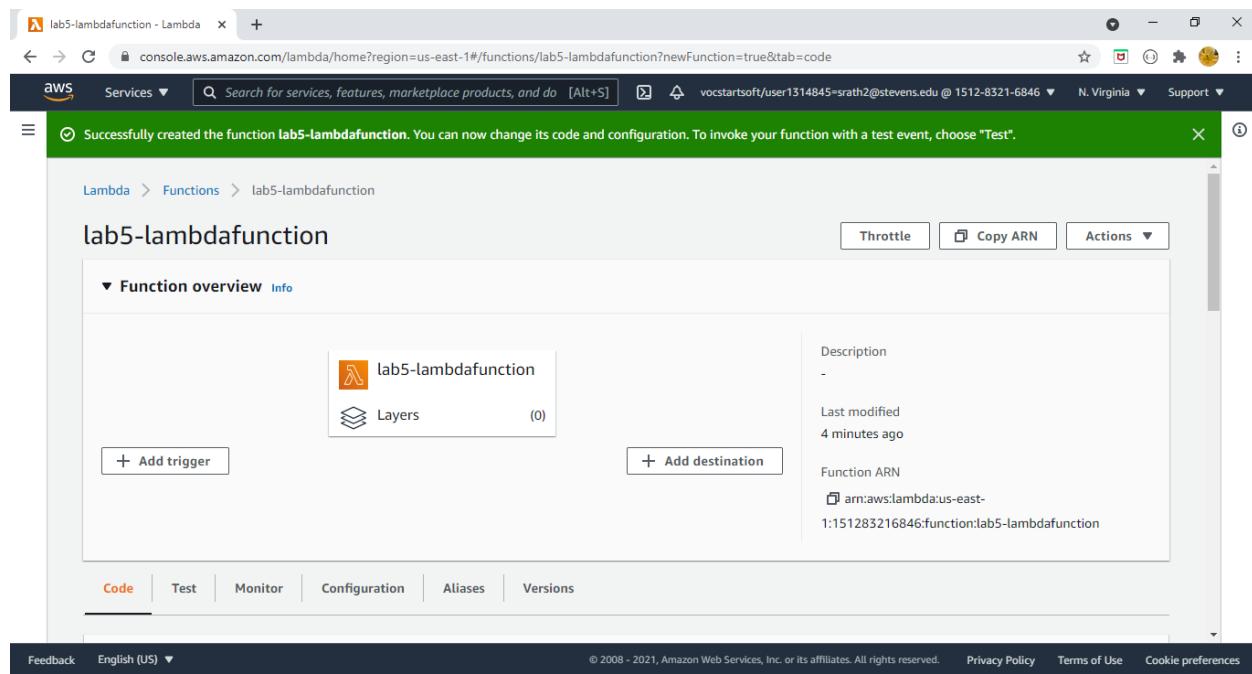
Different options available for creation I chose author from scratch and provided the function name

The screenshot shows the 'Create function' wizard. The first step, 'Choose one of the following options to create your function.', has four options: 'Author from scratch' (selected, indicated by a blue border), 'Use a blueprint', 'Container image', and 'Browse serverless app repository'. The 'Author from scratch' option includes a sub-instruction: 'Start with a simple Hello World example.' The second step, 'Basic information', requires entering a 'Function name' (e.g., 'lab5-lambdafunction') and selecting a 'Runtime' (e.g., 'Node.js 14.x'). Below these fields are detailed descriptions and notes. At the bottom of the screen, standard AWS navigation links like 'Feedback', 'English (US)', and 'Cookie preferences' are visible.

Click on create function.



Lambda function created



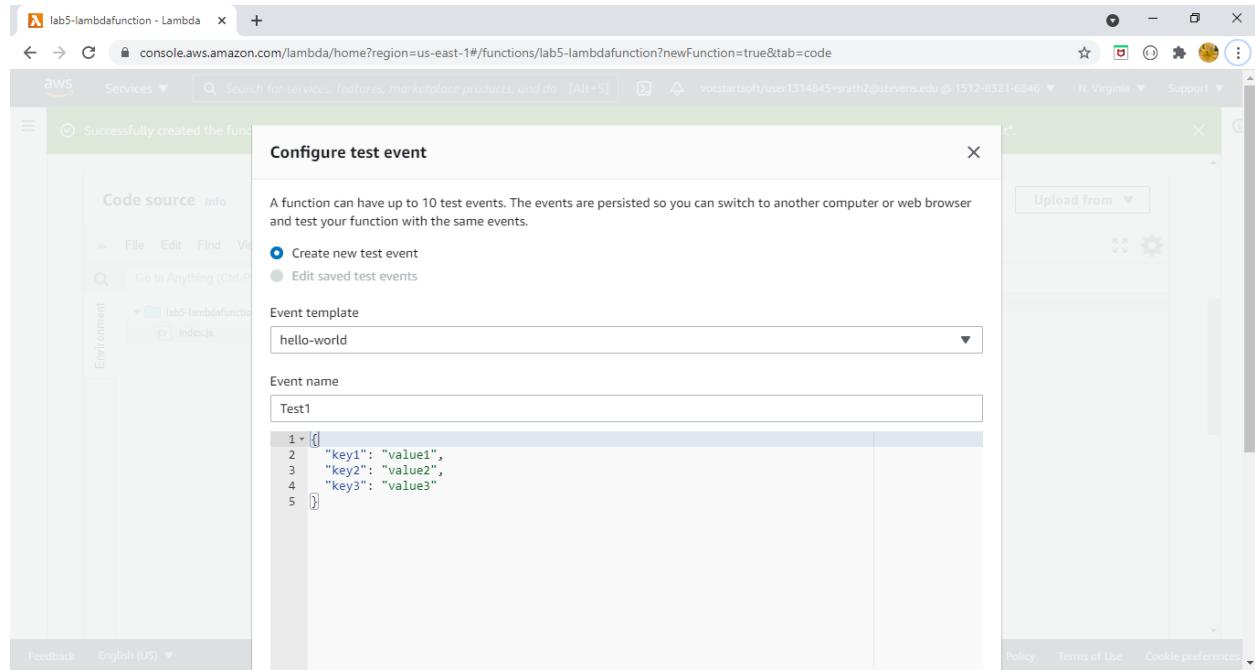
In the code source, code can be edited to changes the lambda function functionality

The screenshot shows the AWS Lambda console interface. The top navigation bar includes the AWS logo, Services dropdown, search bar, and user information. A green success message at the top states: "Successfully created the function lab5-lambdafunction. You can now change its code and configuration. To invoke your function with a test event, choose 'Test'." The main area is titled "Code source" with an "Info" tab. It features a file browser on the left showing a folder "lab5-lambdafunction" containing "index.js". The "index.js" file content is displayed in a code editor:

```
1 exports.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
```

Below the code editor are tabs for "Test" and "Deploy", with "Test" currently selected. A "Changes deployed" button is visible. The bottom of the screen includes standard AWS footer links: Feedback, English (US), Privacy Policy, Terms of Use, and Cookie preferences.

Under test, test event can be create to invoke the lambda function



The screenshot shows the AWS Lambda console interface. A modal window titled 'Test1' is open, displaying the following JSON code:

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

The 'Code source' tab is selected, showing 'index.js'. Below the code editor are buttons for 'Cancel', 'Format JSON', and a prominent orange 'Create' button.

After clicking on test, we can see the response of the test whether it is succeeded or not.

The screenshot shows the AWS Lambda console interface after a test event was run. A green success message at the top states: 'The test event Test1 was successfully saved.' The 'Execution result' section shows the following details:

Status: Succeeded | Max memory used: 65 MB | Time: 15.39 ms

Response:

```
{
  "statusCode": 200,
  "body": "\"Hello from Lambda!\""
}
```

Function Logs:

```
START RequestId: 4dca3c80-d729-46a5-a07e-6cc89a03ea01 Version: $LATEST
END RequestId: 4dca3c80-d729-46a5-a07e-6cc89a03ea01
REPORT RequestId: 4dca3c80-d729-46a5-a07e-6cc89a03ea01 Duration: 15.39 ms Billed Duration: 16 ms Memory Size: 128 MB Max Memory Used: 65 MB
```

Request ID: 4dca3c80-d729-46a5-a07e-6cc89a03ea01

Then we can see the logs under view logs in CloudWatch

The test event Test1 was successfully saved.

Last modified
11 minutes ago

Function ARN
arn:aws:lambda:us-east-1:151283216846:function:lab5-lambdafunction

Code | Test | **Monitor** | Configuration | Aliases | Versions

Metrics | Logs | Traces | View logs in CloudWatch | View X-Ray traces in ServiceLens | View Lambda Insights

CloudWatch metrics Info

Lambda sends runtime metrics for your functions to Amazon CloudWatch. The metrics shown are an **aggregate** view of all function runtime activity. To view metrics for the unqualified or **\$LATEST** resource, choose an option in the dropdown list. To view metrics for a specific function version or alias, choose the qualifier name on the Function details page, and select the Monitoring page.

Add to dashboard | 1h | 3h | 12h | 1d | 3d | 1w | custom | Refresh

Invocations | Duration | Error count and success rate (%)

Once I clicked on view logs in CloudWatch, it will give the details for request and response.

CloudWatch Management Console

/aws/lambda/lab5-lambdafunction

Log group details

| Retention | Creation time | Stored bytes | ARN |
|--------------|----------------|----------------------|---|
| Never expire | 1 minute ago | - | arn:aws:logs:us-east-1:151283216846:log-group:/aws/lambda/lab5-lambdafunction:* |
| KMS key ID | Metric filters | Subscription filters | Contributor Insights rules |
| - | 0 | 0 | - |

Log streams | Metric filters | Subscription filters | Contributor Insights | Tags

Log streams (1)

Create log stream | Search all

Log group details

| | | | |
|--------------|----------------|----------------------|---|
| Retention | Creation time | Stored bytes | ARN |
| Never expire | 1 minute ago | - | arn:aws:logs:us-east-1:151283216846:log-group:/aws/lambda/lab5-lambdafunction:* |
| KMS key ID | Metric filters | Subscription filters | Contributor Insights rules |
| - | 0 | 0 | - |

Log streams (1)

| Log stream | Last event time |
|--|---------------------------------|
| 2021/04/24/[...LATEST]a3f758bf27e24973b82a41002bf67af9 | 2021-04-24 13:43:27 (UTC-04:00) |

Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. [Learn more about filter patterns](#)

| Timestamp | Message |
|-------------------------------|---|
| 2021-04-24T13:43:27.015-04:00 | No older events at this moment. Retry |
| 2021-04-24T13:43:27.031-04:00 | START RequestId: 4dcac80-d729-46a5-a07e-6cc89a03ea01 Version: \$LATEST |
| 2021-04-24T13:43:27.031-04:00 | END RequestId: 4dcac80-d729-46a5-a07e-6cc89a03ea01 |
| 2021-04-24T13:43:27.031-04:00 | REPORT RequestId: 4dcac80-d729-46a5-a07e-6cc89a03ea01 Duration: 15.39 ms Billed Duration: 16 ms Memory... |

Under code source I copied the provided code and deployed the changes.

The test event Test1 was successfully saved.

Code source Info

File Edit Find View Go Tools Window Test Deploy Changes not deployed

index.js Execution results

```
1 var json = {
2   "service": "lambda",
3   "reference": "https://aws.amazon.com/lambda/avengers/",
4   "questions": [
5     {"q": "What is the real name of the Scarlet Witch?", "a": "Wanda Maximoff"},
6     {"q": "Which film did Thor first appear in?", "a": "Thor: The Dark World"},
7     {"q": "Which of the infinity stones is hidden on Vormir?", "a": "Soul Stone"}, {"q": "What is Captain America's shield made of?", "a": "Vibranium"}, {"q": "Which country is Black Panther next in line to be King of?", "a": "T'Challa"}, {"q": "What is the real name of Black Widow?", "a": "Natasha Romanoff"}, {"q": "What is the name of the axe created for and then used by Thor in Avengers: Infinity War?", "a": "Stormbreaker"}, {"q": "What is Loki's title?", "a": "God of Mischief"}]
```

Feedback English (US) ▾ © 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use Cookie preferences

The test event Test1 was successfully saved.

Code Test Monitor Configuration Aliases Versions

Code source Info

File Edit Find View Go Tools Window Test Deploy Changes deployed Your changes have been deployed

index.js Execution results

```
1 var json = {
2   "service": "lambda",
3   "reference": "https://aws.amazon.com/lambda/avengers/",
4   "questions": [
5     {"q": "What is the real name of the Scarlet Witch?", "a": "Wanda Maximoff"}, {"q": "Which film did Thor first appear in?", "a": "Thor: The Dark World"}, {"q": "Which of the infinity stones is hidden on Vormir?", "a": "Soul Stone"}, {"q": "What is Captain America's shield made of?", "a": "Vibranium"}, {"q": "Which country is Black Panther next in line to be King of?", "a": "T'Challa"}, {"q": "What is the real name of Black Widow?", "a": "Natasha Romanoff"}, {"q": "What is the name of the axe created for and then used by Thor in Avengers: Infinity War?", "a": "Stormbreaker"}, {"q": "What is Loki's title?", "a": "God of Mischief"}]
```

Feedback English (US) ▾ © 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use Cookie preferences

The screenshot shows the AWS Lambda console interface. The top navigation bar includes tabs for Services, CloudWatch Management Console, and a search bar. Below the navigation is a green success message: "The test event Test1 was successfully saved." The main area is titled "Code source" and "Info". It features a code editor with tabs for File, Edit, Find, View, Go, Tools, Window, Test, Deploy, and Changes deployed. The "Test" tab is active. The code editor displays "index.js" with the following content:

```
function handler(event) {
  const response = {
    "body": `{"q":${event.q}}`
  };
  return response;
}
```

The "Execution results" section shows a successful execution with the following details:

- Status: Succeeded
- Max memory used: 64 MB
- Time: 25.29 ms

The "Function Logs" section provides detailed log output for the execution:

```
START RequestId: e6e46a86-8b80-42be-81c6-b7687d621c63 Version: $LATEST
2021-04-24T17:48:31.559Z e6e46a86-8b80-42be-81c6-b7687d621c63 INFO  Quote selected: 8
2021-04-24T17:48:31.561Z e6e46a86-8b80-42be-81c6-b7687d621c63 INFO  {
  "body": "{\"q\":\"What is the name of the organisation which is revealed to have infiltrated S.H.I.E.L.D. in Captain America: The Winter Soldier?\"}"
}
END RequestId: e6e46a86-8b80-42be-81c6-b7687d621c63
REPORT RequestId: e6e46a86-8b80-42be-81c6-b7687d621c63 Duration: 25.29 ms Billed Duration: 26 ms Memory Size: 128 MB Max Memory Used: 64 MB
Request ID
e6e46a86-8b80-42be-81c6-b7687d621c63
```

By clicking on add trigger I added the rest api to invoke the lambda function.

The screenshot shows the AWS Lambda console interface, specifically the "Function overview" page for the "lab5-lambdafunction" function. The top navigation bar includes tabs for Services, CloudWatch Management Console, and a search bar. Below the navigation is a green success message: "The test event Test1 was successfully saved." The main area is titled "Function overview" and "Info".

The function configuration includes:

- Name:** lab5-lambdafunction
- Description:** -
- Last modified:** 15 minutes ago
- Function ARN:** arn:aws:lambda:us-east-1:151283216846:function:lab5-lambdafunction

Below the configuration, there are buttons for "Throttle", "Copy ARN", and "Actions". The "Actions" dropdown menu is open, showing options like "Edit", "Delete", "Add trigger", "Add destination", "Edit environment variables", "Edit triggers", "Edit destinations", "Edit layers", and "Edit VPC settings".

The screenshot shows the 'Trigger configuration' page for a Lambda function. The 'API type' section is set to 'REST API'. The 'Security' section is set to 'Open'. A note at the bottom states: 'Lambda will add the necessary permissions for Amazon API Gateway to invoke your Lambda function from this trigger.'

Once API gateway is added, we can invoke the lambda function by going to the end point url.

The screenshot shows the 'Configuration' tab for the 'lab5-lambdafunction' Lambda function. Under the 'Triggers' section, there is one entry: 'API Gateway: lab5-lambdafunction-API'. The ARN for this trigger is listed as 'arn:aws:lambda:us-east-1:151283216846:function:lab5-lambdafunction'.

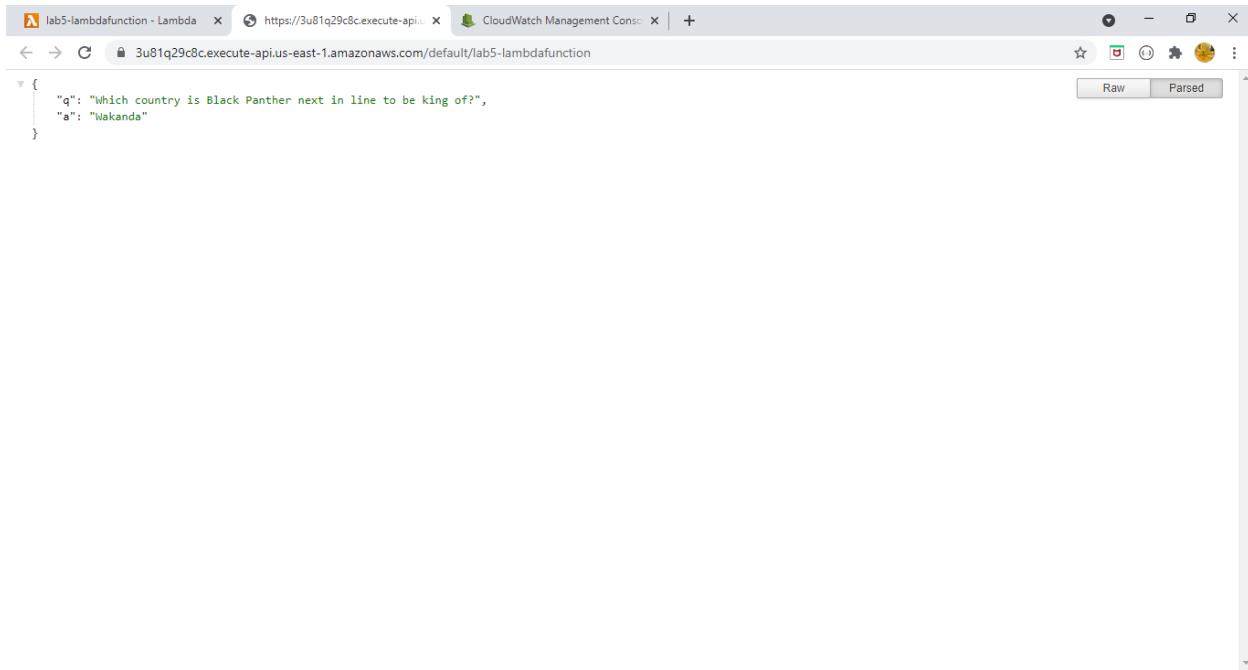
The screenshot shows the AWS Lambda Management Console. The left sidebar has a tree view with 'General configuration' expanded, and 'Triggers' is selected. The main area is titled 'Triggers (1)'. It shows a single trigger named 'Trigger' with the sub-type 'API Gateway: lab5-lambdafunction-API'. The details pane below it provides specific configuration details:

- API: api-gateway/3u81q29c8c/*/*/lab5-lambdafunction
- API endpoint: <https://3u81q29c8c.execute-api.us-east-1.amazonaws.com/default/lab5-lambdafunction>
- API name: lab5-lambdafunction-API
- API type: rest
- Authorization: NONE
- Enable metrics and error logging: No
- Method: ANY
- Resource path: /lab5-lambdafunction
- Security: NONE
- Stage: default

It displays random question on every invoked request.

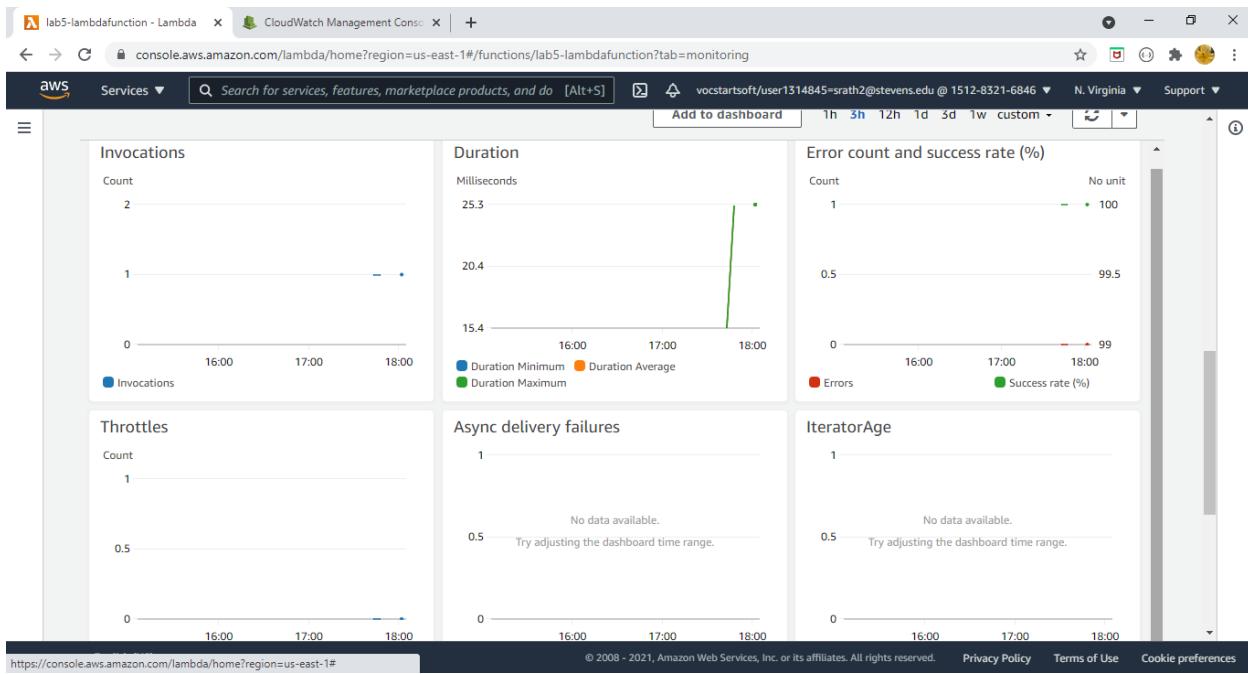
The screenshot shows a browser window with the URL <https://3u81q29c8c.execute-api.us-east-1.amazonaws.com/default/lab5-lambdafunction>. The page content is a JSON object:

```
{ "q": "What food do the Avengers go to eat after the Battle of New York in the first Avengers film at Tony Stark's suggestion?", "a": "Shawarma" }
```



```
Raw Parsed
{
  "q": "Which country is Black Panther next in line to be king of?",
  "a": "Wakanda"
}
```

Under monitor tab I can see the different logs for invocations, duration, error count and success rate.



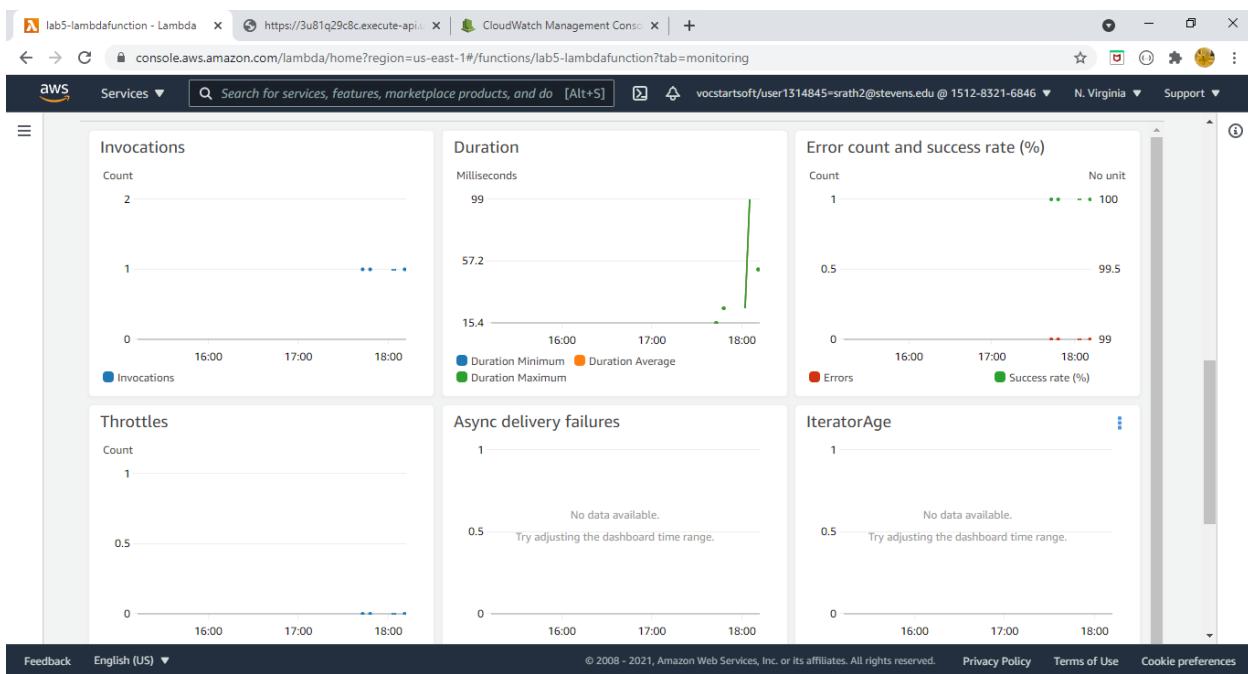
A screenshot of a web browser window showing a Lambda function log entry. The URL is <https://3u81q29c8c.execute-api.us-east-1.amazonaws.com/default/lab5-lambdafunction>. The log entry is as follows:

```

{
  "q": "What is Loki's title?",
  "a": "God of Mischief"
}

```

The browser has tabs for "lab5-lambdafunction - Lambda", "https://3u81q29c8c.execute-api.us-east-1.amazonaws.com/default/lab5-lambdafunction", and "CloudWatch Management Console". There are "Raw" and "Parsed" buttons at the top right.



Under logs in CloudWatch it will display the number of times the function is invoked with request time and ended time.

Screenshot of the AWS CloudWatch Management Console showing the Log streams page for the log group /aws/lambda/lab5-lambdafunction.

Log streams (4)

| Log stream | Last event time |
|--|---------------------------------|
| 2021/04/24/[SLATEST]b5251777d5d94a27ae3c0052348b574f | 2021-04-24 14:11:31 (UTC-04:00) |
| 2021/04/24/[SLATEST]off87c8050314811b9cf6b974b702dc | 2021-04-24 14:02:31 (UTC-04:00) |
| 2021/04/24/[SLATEST]51f68698b65e42879af6f0fa212e134c | 2021-04-24 13:48:31 (UTC-04:00) |
| 2021/04/24/[SLATEST]3f758bf27e24973b82a41002bf67af9 | 2021-04-24 13:43:27 (UTC-04:00) |

Screenshot of the AWS CloudWatch Management Console showing the Log events page for the log stream 2021/04/24/[SLATEST]b5251777d5d94a27ae3c0052348b574f.

Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns [\[?\]](#)

| Timestamp | Message |
|-------------------------------|--|
| 2021-04-24T14:11:31.861-04:00 | START RequestId: b2bcde6-c590-4049-8d33-726f60cfb28e Version: \$LATEST |
| 2021-04-24T14:11:31.872-04:00 | 2021-04-24T18:11:31.8722 b2bcde6-c590-4049-8d33-726f60cfb28e INFO Quote selected: 7 |
| 2021-04-24T14:11:31.874-04:00 | 2021-04-24T18:11:31.8742 b2bcde6-c590-4049-8d33-726f60cfb28e INFO { body: `{"q": "What is Loki's title?..` } |
| 2021-04-24T14:11:31.913-04:00 | END RequestId: b2bcde6-c590-4049-8d33-726f60cfb28e |
| 2021-04-24T14:11:31.913-04:00 | REPORT RequestId: b2bcde6-c590-4049-8d33-726f60cfb28e Duration: 51.55 ms Billed Duration: 52 ms Memory... |

The screenshot shows the AWS Lambda function configuration page. The top navigation bar includes tabs for Services, Search for services, features, marketplace products, and do [Alt+S], and a user session (vocstartsoft/user1314845=srath2@stevens.edu @ 1512-8321-6846). The main area displays the API Gateway section with a 'Code' tab selected. The code source editor shows the index.js file with the following content:

```
1 var json = {
2   "Name": "TV Maze API",
3   "shows": [
4     {
5       "score": 18.118694,
6       "show": {
7         "id": 139,
8         "url": "https://www.tvmaze.com/shows/139/girls",
9         "name": "Girls",
10        "type": "Scripted",
11        "language": "English",
12        "genres": [
13          "Drama",
14          "Romance"
15        ],
16      }
17    }
18  ]
19 }
```

I made changes in code source to make a TV maze api, so that I can invoke with rest api endpoint url and can use it for my program to display all the shows

The screenshot shows the AWS Lambda function configuration page with the code source editor displaying the updated index.js file. The content has been modified to include additional details for the TV show 'Rhyme and her friends'.

```
538   },
539   "dvdCountry": null,
540   "externals": {
541     "tvrage": null,
542     "thetvdb": 339854,
543     "imdb": null
544   },
545   "image": {
546     "medium": "https://static.tvmaze.com/uploads/images/medium_portrait/250/627433.jpg",
547     "original": "https://static.tvmaze.com/uploads/images/original_unouched/250/627433.jpg"
548   },
549   "summary": "<p>Rhyme and her friends – known by their 'ship name, \"The Chicken Girls\" – have been dancing together forever. But thi
550   "
551   "updated": 1618952664,
552   "links": {
553     "self": {
554       "href": "https://api.tvmaze.com/shows/32087"
555     },
556     "previousepisode": {
557       "href": "https://api.tvmaze.com/episodes/2053199"
558     },
559     "nextepisode": {
560       "href": "https://api.tvmaze.com/episodes/2053200"
561     }
562   }
563 }
564 }
565 }
566 }
567 }
568 }
569 }
570 }
571 }
572 }
573 }
```

```

{
  "score": 18.118694,
  "show": {
    "id": 139,
    "url": "https://www.tvmaze.com/shows/139/girls",
    "name": "Girls",
    "type": "Scripted",
    "language": "English",
    "genres": [
      "Drama",
      "Romance"
    ],
    "status": "Ended",
    "runtime": 30,
    "premiered": "2012-04-15",
    "officialSite": "http://www.hbo.com/girls",
    "schedule": {
      "time": "22:00",
      "days": [
        "Sunday"
      ]
    },
    "rating": {
      "average": 6.6
    },
    "weight": 92,
    "network": {
      "id": 8,
      "name": "HBO",
      "country": {
        "name": "United States",
        "code": "US",
        "timezone": "America/New_York"
      }
    }
  }
}

```

To display first show

```

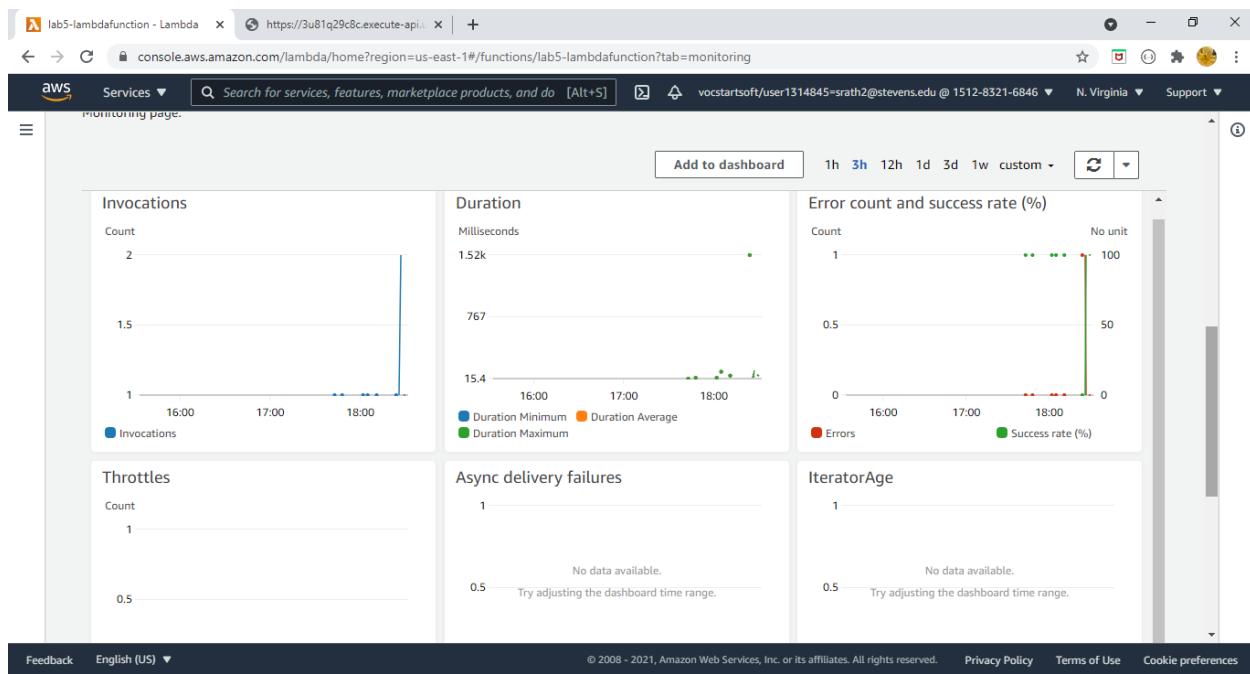
 551   "links": {
 552     "self": {
 553       "href": "https://api.tvmaze.com/shows/32087"
 554     },
 555     "previousepisode": {
 556       "href": "https://api.tvmaze.com/episodes/2853199"
 557     },
 558     "nextepisode": {
 559       "href": "https://api.tvmaze.com/episodes/2053200"
 560     }
 561   }
 562 }
 563 }
 564 ]
 565 }
 566
 567   exports.handler = function(event, context) {
 568     var response = {
 569       body: JSON.stringify(json.shows[1])
 570     };
 571     console.log(response);
 572     context.succeed(response);
 573   };

```

```

{
  "score": 15.28145,
  "show": {
    "id": 23542,
    "url": "https://www.tvmaze.com/shows/23542/good-girls",
    "name": "Good Girls",
    "type": "Scripted",
    "language": "English",
    "genres": [
      "Drama",
      "Comedy",
      "Crime"
    ],
    "status": "Running",
    "runtime": 60,
    "averageRuntime": 60,
    "premiered": "2018-02-26",
    "officialSite": "https://www.nbc.com/good-girls",
    "schedule": {
      "time": "22:00",
      "days": [
        "Sunday"
      ]
    },
    "rating": {
      "average": 7.4
    },
    "weight": 99,
    "network": {
      "id": 1,
      "name": "NBC",
      "country": {
        "name": "United States",
        "code": "US",
        "timezone": "America/New_York"
      }
    }
}

```



Now I am creating S3 bucket to attach it to lambda function. So that whenever I will upload some data in my s3 bucket, I can invoke the lambda function

The screenshot shows the AWS S3 Management Console. On the left, there's a sidebar with navigation links like 'Buckets', 'Access Points', 'Object Lambda Access Points', 'Batch Operations', 'Access analyzer for S3', 'Block Public Access settings for this account', 'Storage Lens' (which is expanded to show 'Dashboards' and 'AWS Organizations settings'), 'Feature spotlight', and 'AWS Marketplace for S3'. The main content area is titled 'Buckets (2)' and lists two buckets: 'cf-templates-13edpgj2uquad-us-east-1' and 'lab5-bucketlambda'. Each bucket entry includes its name, AWS Region (US East (N. Virginia) us-east-1), access level ('Objects can be public'), and creation date (April 10, 2021, 19:14:11 (UTC-04:00) and April 24, 2021, 14:40:42 (UTC-04:00)).

Uploaded image inside bucket

This screenshot shows the 'Objects' tab for the 'lab5-bucketlambda' bucket. The sidebar is identical to the previous screenshot. The main content area is titled 'lab5-bucketlambda' and shows one object named 'cloud.jpg'. The object details are: Name (cloud.jpg), Type (jpg), Last modified (April 24, 2021, 14:41:01 (UTC-04:00)), Size (202.7 KB), and Storage class (Standard). There are buttons for 'Copy URL', 'Delete', 'Actions', 'Create folder', and 'Upload'.

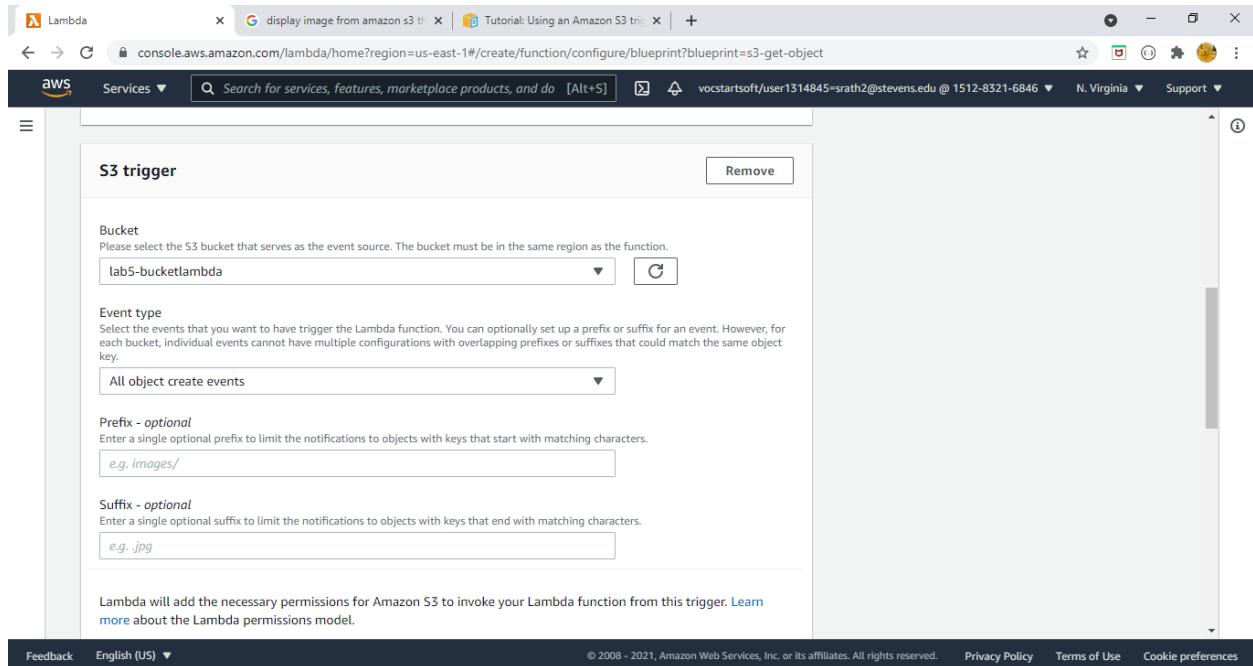
For creating a lambda function, we have to create the function by using a blueprint.

The screenshot shows the AWS Lambda 'Create function' interface. At the top, there are four options: 'Author from scratch', 'Use a blueprint' (which is selected), 'Container image', and 'Browse serverless app repository'. Below this, a section titled 'Blueprints' lists three items: 's3-get-object-python', 'rekognition-python', and 's3-get-object'. The 's3-get-object' item is selected and highlighted with a blue border. The status bar at the bottom indicates the browser is at 40% zoom and the date is 4/24/2021.

Under lambda function I have created a new role for AWS policy.

The screenshot shows the 'Configure blueprint s3-get-object' page under the 'Basic information' tab. It includes fields for 'Function name' (set to 'my-s3-function'), 'Execution role' (set to 'Create a new role from AWS policy templates'), and 'Role name' (set to 'my-s3-function-role'). A note at the bottom states: 'Role creation might take a few minutes. Please do not delete the role or edit the trust or permissions policies in this role.' The status bar at the bottom indicates the browser is at 40% zoom and the date is 4/24/2021.

Under triggers, I have newly added S3 bucket



Then it will the code source before creation of lambda function.

```
1  console.log('Loading function');
2
3  const aws = require('aws-sdk');
4
5  const s3 = new aws.S3({ apiVersion: '2006-03-01' });
6
7
8  exports.handler = async (event, context) => {
9      //console.log('Received event:', JSON.stringify(event, null, 2));
10
11     // Get the object from the event and show its content type
12     const bucket = event.Records[0].s3.bucket.name;
13     const key = decodeURIComponent(event.Records[0].s3.object.key.replace(/\+/g, ' '));
14
15     const params = {
16         Bucket: bucket,
17         Key: key,
18     };
19     try {
20         const { ContentType } = await s3.getObject(params).promise();
21         console.log('CONTENT TYPE:', ContentType);
22         return ContentType;
23     } catch (err) {
24         console.log(err);
25         const message = `Error getting object ${key} from bucket ${bucket}. Make sure they`;
26         console.log(message);
27         throw new Error(message);
28     }
29 }
```

The screenshot shows the AWS Lambda code editor with the above JavaScript code. The code handles an S3 object creation event, logs the event details, retrieves the object content type, and returns it. The 'Create function' button is visible at the bottom right of the editor.

Lambda function created.

The screenshot shows the AWS Lambda console with the function 'my-s3-function' selected. The top navigation bar includes tabs for Services, Search for services, features, marketplace products, and do [Alt+S], and a user profile. The main content area displays the function's name, a success message about creation, and its configuration details. The configuration pane shows an S3 trigger named 'S3' with an ARN of arn:aws:lambda:us-east-1:151283216846:function:my-s3-function. Buttons for Throttle, Copy ARN, and Actions are visible at the top right.

Checking the source code.

The screenshot shows the 'Code source' tab of the AWS Lambda console. The interface includes a toolbar with Code, Test, Monitor, Configuration, Aliases, and Versions. The main area displays the 'index.js' file content, which is a Node.js script for an S3 trigger. The code uses the AWS SDK to log the loading function, get the object from the event, and return its content type. The AWS SDK is required for the function.

```
1 console.log('Loading function');
2
3 const aws = require('aws-sdk');
4
5 const s3 = new aws.S3({ apiVersion: '2006-03-01' });
6
7
8 exports.handler = async (event, context) => {
9   //console.log('Received event:', JSON.stringify(event, null, 2));
10
11  // Get the object from the event and show its content type
12  const bucket = event.Records[0].s3.bucket.name;
13  const key = decodeURIComponent(event.Records[0].s3.object.key.replace(/\+/g, ' '));
14  const params = {
15    Bucket: bucket,
16    Key: key,
17  };
18  try {
19    const { ContentType } = await s3.getObject(params).promise();
20    console.log('CONTENT TYPE:', ContentType);
21    return ContentType;
22  } catch (err) {
23    console.log(err);
24    const message = `Error getting object ${key} from bucket ${bucket}. Make sure they exist and your bucket is in the same region as your Lambda function.`;
25    console.log(message);
26    throw new Error(message);
27  }
28}
```

For invoking lambda function, where the event-template will be s3-put and provide a name.

The screenshot shows the AWS Lambda console. In the top navigation bar, there are tabs for 'my-s3-function - Lambda', 'S3 Management Console', and 'Tutorial: Using an Amazon S3 trigger'. The main area is titled 'Event template' with a dropdown menu showing 's3-put'. Below it, the 'Event name' field contains 'mys3testevent'. The code editor shows the following JSON configuration:

```
mys3testevent
10 "principalId": "EXAMPLE"
11 },
12 "requestParameters": {
13   "sourceIPAddress": "127.0.0.1"
14 },
15 "responseElements": {
16   "x-amz-request-id": "EXAMPLE123456789",
17   "x-amz-id-2": "EXAMPLE123/5678abcdefghijklmnaaaaaaaaaaaaaaaaaaaaa"
18 },
19 "s3": {
20   "s3SchemaVersion": "1.0",
21   "configurationId": "testConfigRule",
22   "bucket": {
23     "name": "lab5-bucketlambda",
24     "ownerIdentity": {
25       "principalId": "EXAMPLE"
26     },
27     "arn": "arn:aws:s3:::example-bucket"
28   },
29   "object": {
30     "key": "cloud.jpg",
31     "size": 1024,
32     "eTag": "0123456789abcdef0123456789abcdef",
33     "sequencer": "0A1B2C3D4E5F678901"
34   }
35 }
36 }
```

In the code source I have provided the S3 bucket name and key which is the uploaded image name "cloud.jpg"

The screenshot shows the AWS Lambda console with the same interface as the previous one. The event template is still 's3-put' and the event name is 'mys3testevent'. The code editor now displays the updated JSON configuration with the 'key' value changed to 'cloud.jpg':

```
mys3testevent
10 "principalId": "EXAMPLE"
11 },
12 "requestParameters": {
13   "sourceIPAddress": "127.0.0.1"
14 },
15 "responseElements": {
16   "x-amz-request-id": "EXAMPLE123456789",
17   "x-amz-id-2": "EXAMPLE123/5678abcdefghijklmnaaaaaaaaaaaaaaaaaaaaa"
18 },
19 "s3": {
20   "s3SchemaVersion": "1.0",
21   "configurationId": "testConfigRule",
22   "bucket": {
23     "name": "lab5-bucketlambda",
24     "ownerIdentity": {
25       "principalId": "EXAMPLE"
26     },
27     "arn": "arn:aws:s3:::example-bucket"
28   },
29   "object": {
30     "key": "cloud.jpg",
31     "size": 1024,
32     "eTag": "0123456789abcdef0123456789abcdef",
33     "sequencer": "0A1B2C3D4E5F678901"
34   }
35 }
36 }
```

At the bottom right of the code editor, there are three buttons: 'Cancel', 'Format JSON', and a large orange 'Create' button.

Changes are deployed.

```
1 console.log('Loading function');
2
3 const aws = require('aws-sdk');
4
5 const s3 = new aws.S3({apiVersion: '2006-03-01'});
6
7
8 exports.handler = async (event, context) => {
9     //console.log('Received event:', JSON.stringify(event, null, 2));
10    // Get the object from the event and show its content type
11    const bucket = event.Records[0].s3.bucket.name;
12    const key = decodeURIComponent(event.Records[0].s3.object.key.replace(/\+/g, ' '));
13    const params = {
14        Bucket: bucket,
15        Key: key,
16    };
17}
```

On clicking test, it will invoke the lambda function which will return success.

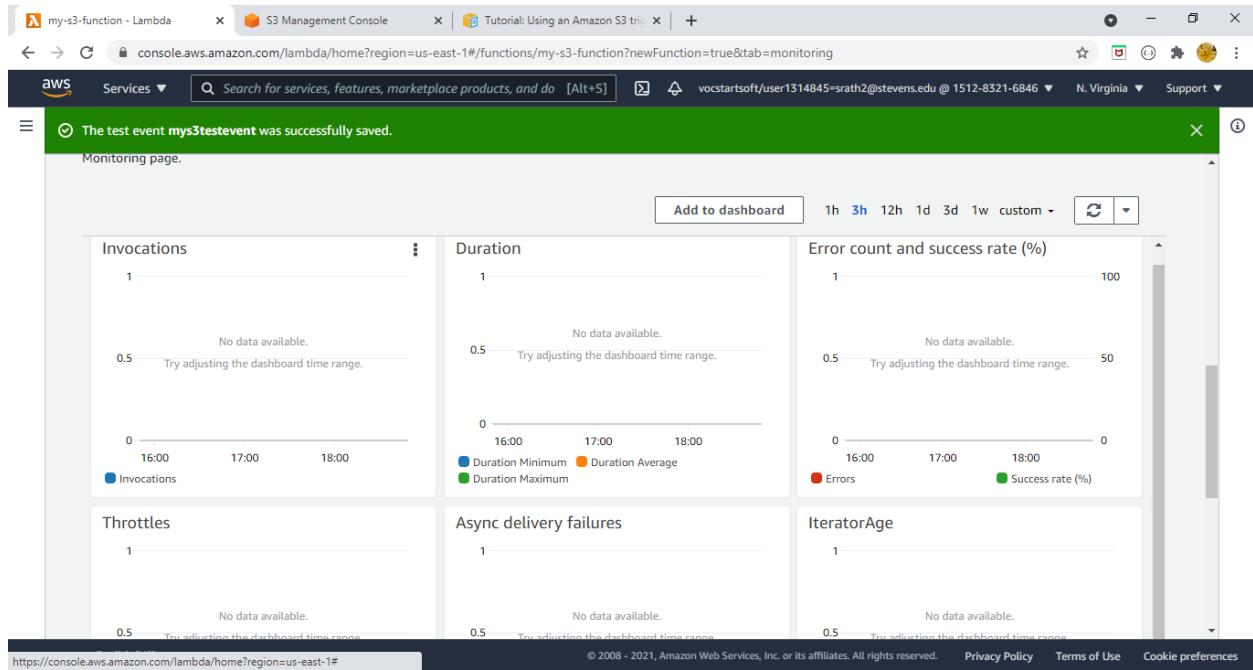
Execution result: Status: Succeeded Max memory used: 88 MB Time: 853.46 ms

Response
"image/jpeg"

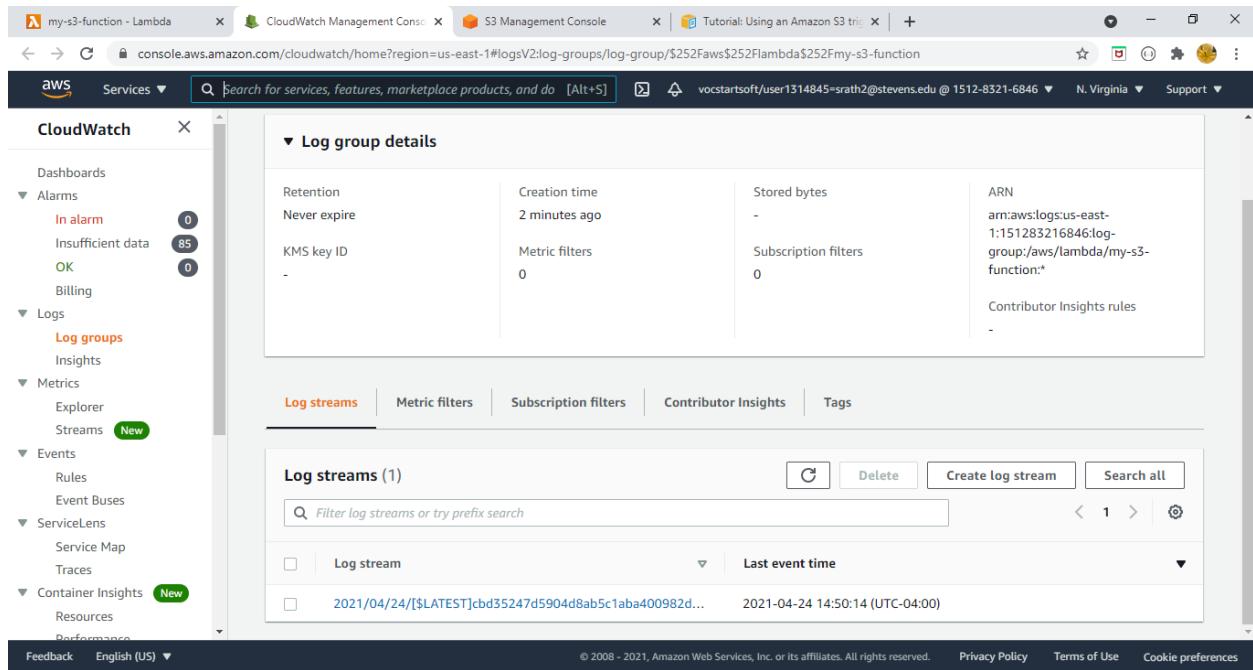
Function Logs

```
START RequestId: 9d133c87-224d-479c-962c-434a4b7e2f1e Version: $LATEST
2021-04-24T18:50:15.187Z 9d133c87-224d-479c-962c-434a4b7e2f1e INFO  CONTENT TYPE: image/jpeg
END RequestId: 9d133c87-224d-479c-962c-434a4b7e2f1e
REPORT RequestId: 9d133c87-224d-479c-962c-434a4b7e2f1e Duration: 853.46 ms Billed Duration: 854 ms Memory Size: 128 MB Max Memory Used: 88 MB
```

Request ID
9d133c87-224d-479c-962c-434a4b7e2f1e



Log generated after invoking Lambda function.



Uploading the image in S3 bucket.

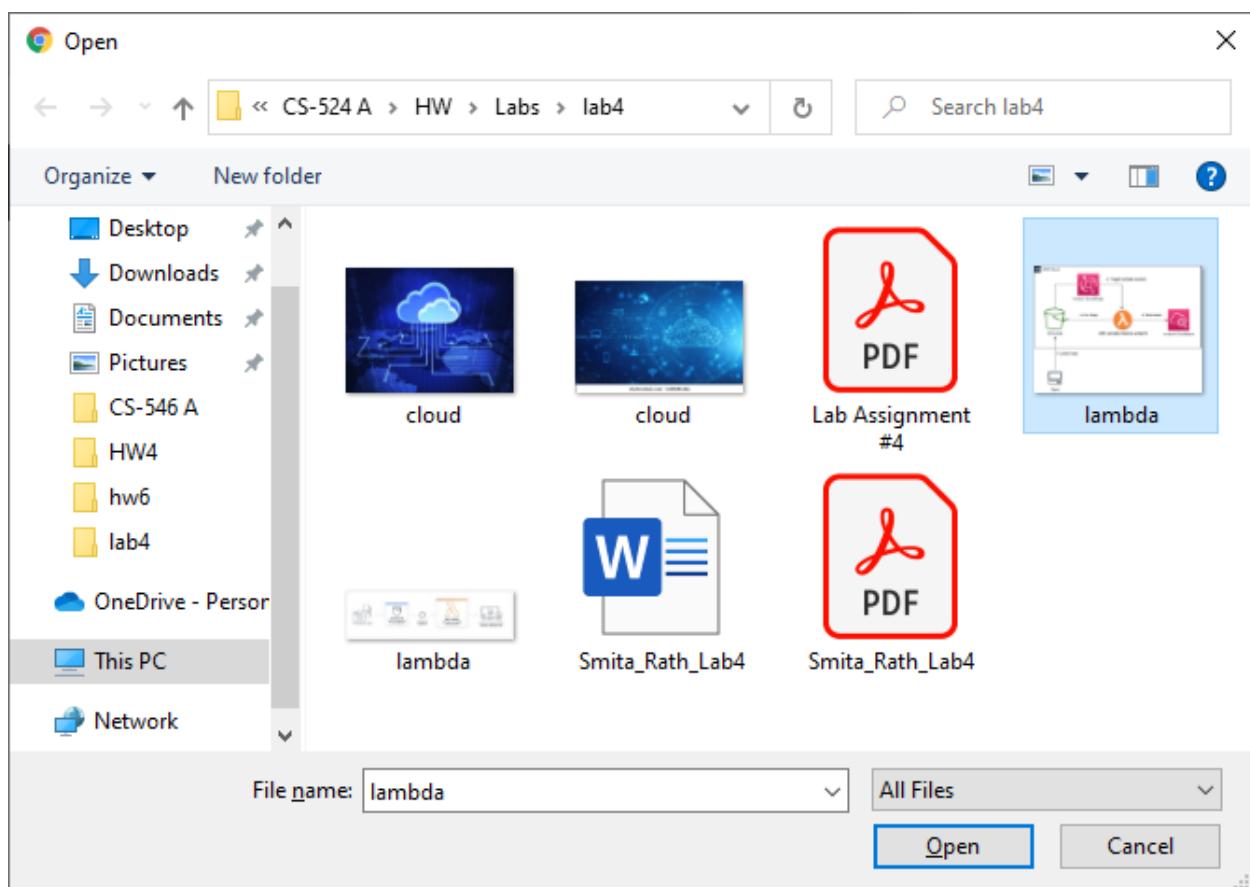
Screenshot of the AWS S3 Management Console showing the 'Upload' interface for the 'lab5-bucketlambda' bucket.

The browser tabs include: my-s3-function - Lambda, CloudWatch Management Console, S3 Management Console, Tutorial: Using an Amazon S3 trig..., and s3.console.aws.amazon.com/s3/upload/lab5-bucketlambda?region=us-east-1.

The main content area shows the 'Upload' step. A large text box says: "Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. Learn more." Below it is a dashed border box with the instruction: "Drag and drop files and folders you want to upload here, or choose Add files, or Add folders."

A table titled "Files and folders (0)" shows the uploaded items. It has columns: Name, Folder, Type, and Size. A search bar at the top of the table says "Find by name". The message "No files or folders" is displayed below the table, followed by the note "You have not chosen any files or folders to upload."

At the bottom of the page are links: Feedback, English (US) ▾, © 2008 – 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved., Privacy Policy, Terms of Use, and Cookie preferences.



All files and folders in this table will be uploaded.

| Name | Folder | Type | Size |
|------------|--------|------------|---------|
| lambda.jpg | - | image/jpeg | 62.2 KB |

Destination

Destination
s3://lab5-bucketlambda

▶ 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.

Cancel **Upload**

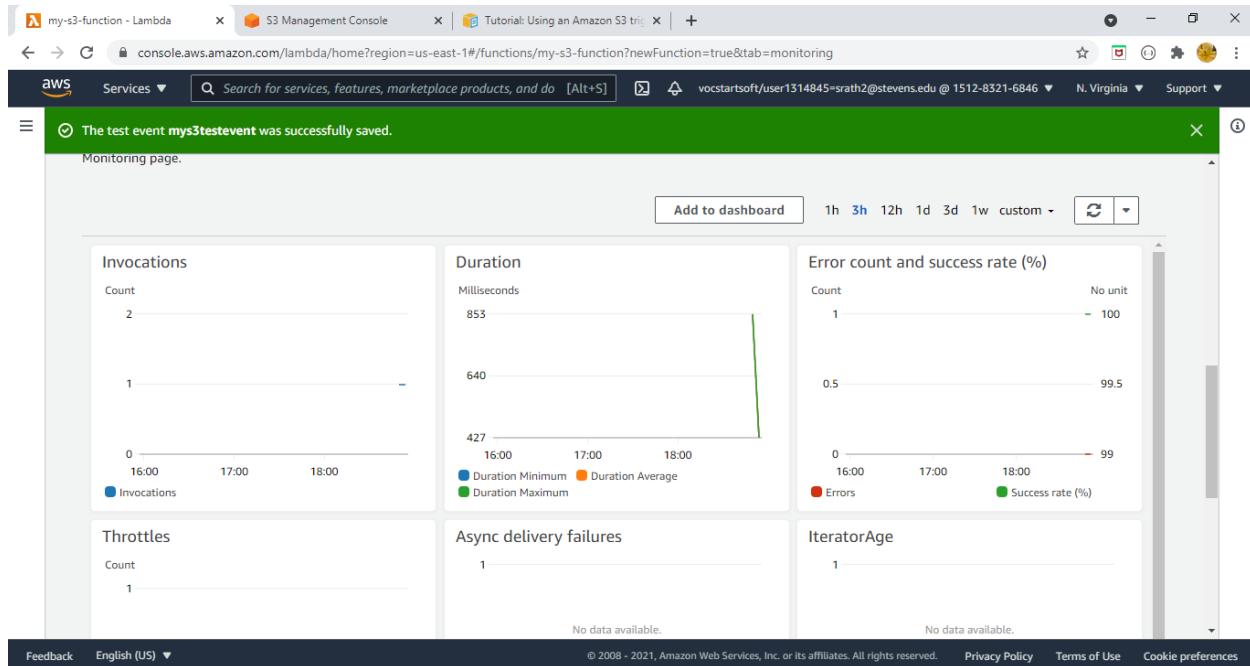
Once uploaded the lambda function will be invoked.

Upload succeeded
View details below.

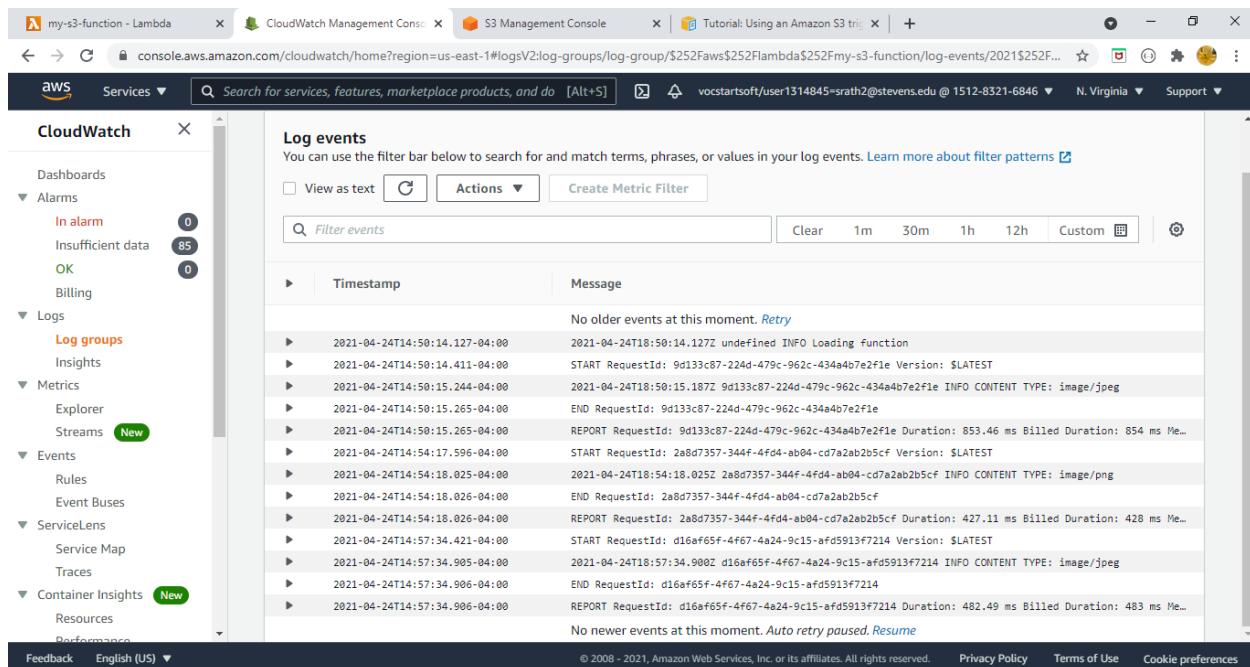
| Destination | Succeeded | Failed |
|------------------------|---------------------------|-------------------|
| s3://lab5-bucketlambda | 1 file, 62.2 KB (100.00%) | 0 files, 0 B (0%) |

Files and folders (1 Total, 62.2 KB)

| Name | Type | Size |
|------------|------------|---------|
| lambda.jpg | image/jpeg | 62.2 KB |



Logs generated after uploading the new image in S3 bucket.



The screenshot shows the AWS Lambda Functions page. There are two functions listed:

| Function name | Description | Package type | Runtime | Code size | Last modified |
|---------------------|--|--------------|--------------|-----------|----------------|
| lab5-lambdafunction | An Amazon S3 trigger that retrieves metadata for the object that has been updated. | Zip | Node.js 14.x | 3.4 kB | 9 seconds ago |
| my-s3-function | An Amazon S3 trigger that retrieves metadata for the object that has been updated. | Zip | Node.js 12.x | 668 bytes | 18 minutes ago |

The screenshot shows the AWS Lambda Functions page after the "my-s3-function" was deleted. A green message box displays:

Your Lambda function "my-s3-function" was successfully deleted.

The table now shows only one function:

| Function name | Description | Package type | Runtime | Code size | Last modified |
|---------------------|--|--------------|--------------|-----------|----------------|
| lab5-lambdafunction | An Amazon S3 trigger that retrieves metadata for the object that has been updated. | Zip | Node.js 14.x | 3.4 kB | 24 seconds ago |

So, in this way lambda function can be used for uploading images in S3 bucket which can be used for processing images. Also, lambda function can be invoked with the Lambda API, the AWS SDK, the AWS CLI, and AWS toolkits. We can also configure other AWS services to invoke our function, or you can configure Lambda to read from a stream or queue and invoke your function.

When we invoke a function, we can choose to invoke it synchronously or asynchronously. With synchronous invocation, we wait for the function to process the event and return a response. With asynchronous invocation, Lambda queues the event for processing and returns a response immediately. For asynchronous invocation, Lambda handles retries and can send invocation records to a destination.

To use function to process data automatically, add one or more triggers. A trigger is a Lambda resource or a resource in another service that we configure to invoke our function in response to lifecycle events, external requests, or on a schedule. Function can have multiple triggers. Each trigger acts as a client invoking function independently. Each event that Lambda passes to the function only has data from one client or trigger.

<https://docs.aws.amazon.com/lambda/latest/dg/lambda-invocation.html>

<https://docs.aws.amazon.com/lambda/latest/dg/getting-started-create-function.html>

<https://docs.aws.amazon.com/lambda/latest/dg/services-apigateway.html>

<https://docs.aws.amazon.com/lambda/latest/dg/with-s3-example.html>