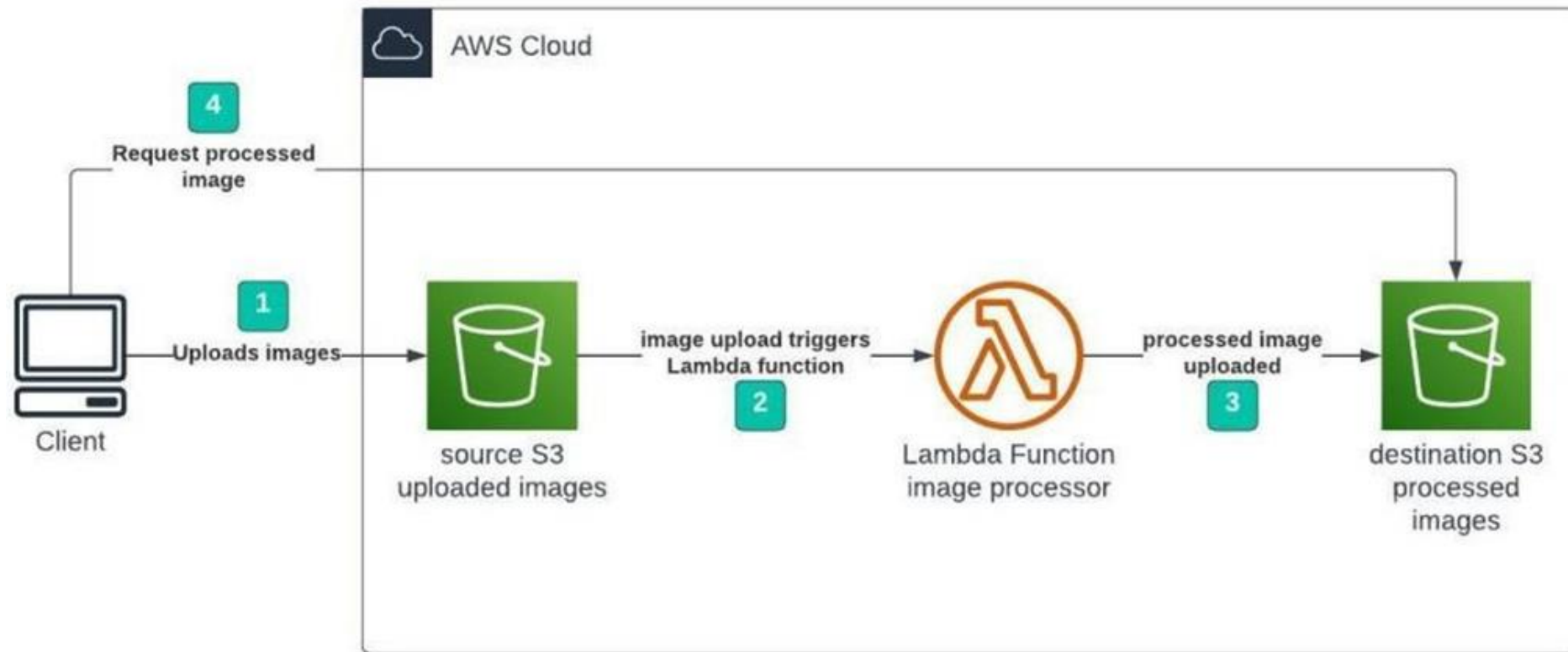


Project-1

Serverless Image Processing

Create a serverless image processing application that automatically resizes and optimizes images uploaded to an Amazon S3 bucket.

- ❖ The Serverless Image Handler solution helps you embed images on your websites and mobile applications to drive user engagement. It uses the SHARP Node.js library to provide high- speed image processing without sacrificing image quality.
- ❖ To minimize your costs of image optimization, manipulation, and processing, this solution automates version control and provides flexible storage and compute options for file reprocessing.



Serverless Image Processor

- **LAB STEPS:-**

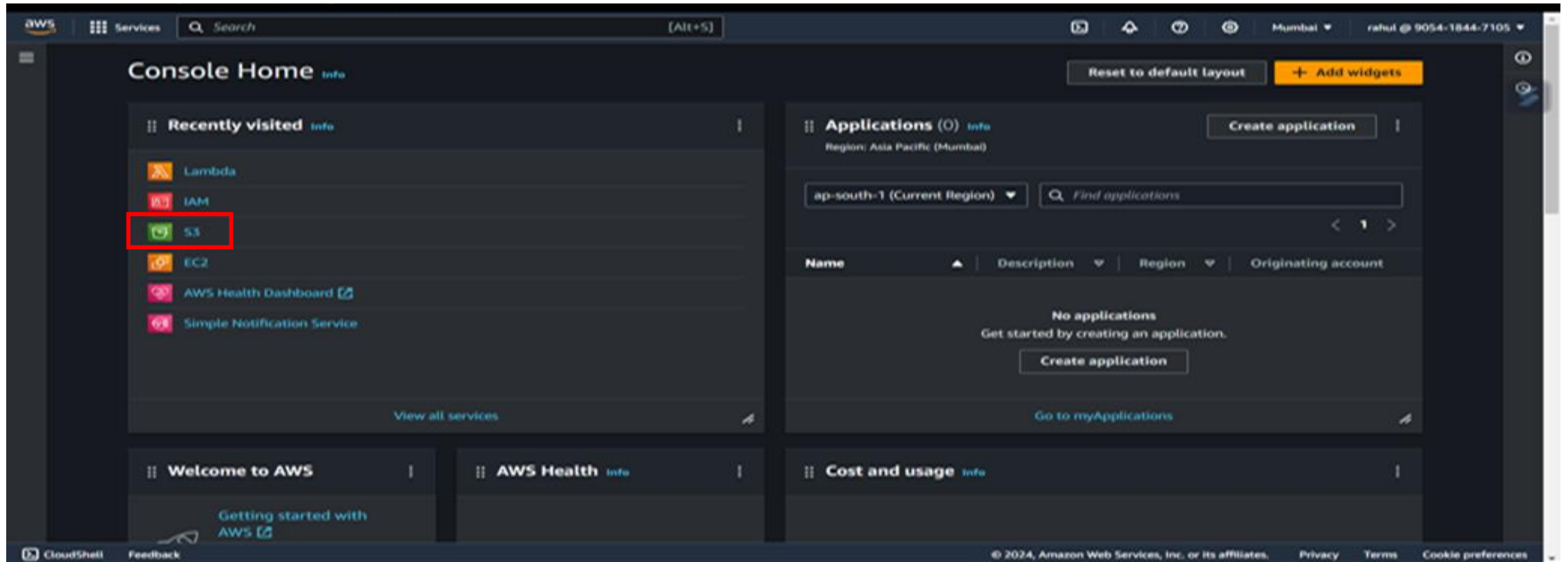
Task 1: Sign in to AWS Management Console

1. Click on the Open Console button, and you will get redirected to AWS Console in a new browser tab.
2. On the AWS sign-in page,
 - Leave the Account ID as default. Never edit/remove the 12 digit Account ID present in the AWS Console. otherwise, you cannot proceed with the lab.
 - Now copy your User Name and Password in the Lab Console to the IAM Username and Password in AWS Console and click on the Sign in button.
3. Once Signed In to the AWS Management Console, Make the default AWS Region as US East (N. Virginia) us-east-1.

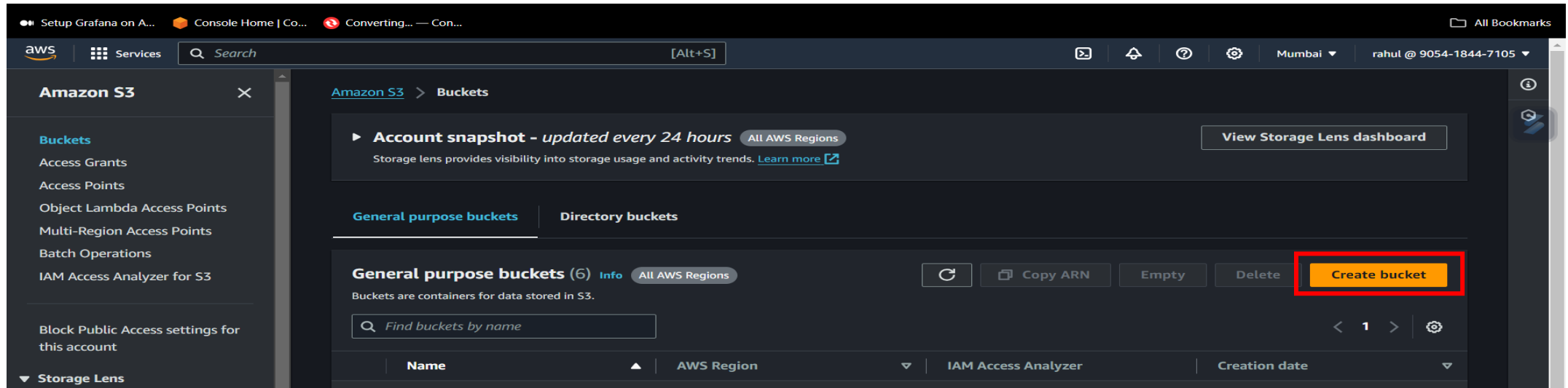
❖ Create Two Amazon S3 Buckets

In this task, we will create two AWS S3 buckets i.e the source bucket and the destination bucket by providing the required configurations like name, region etc.

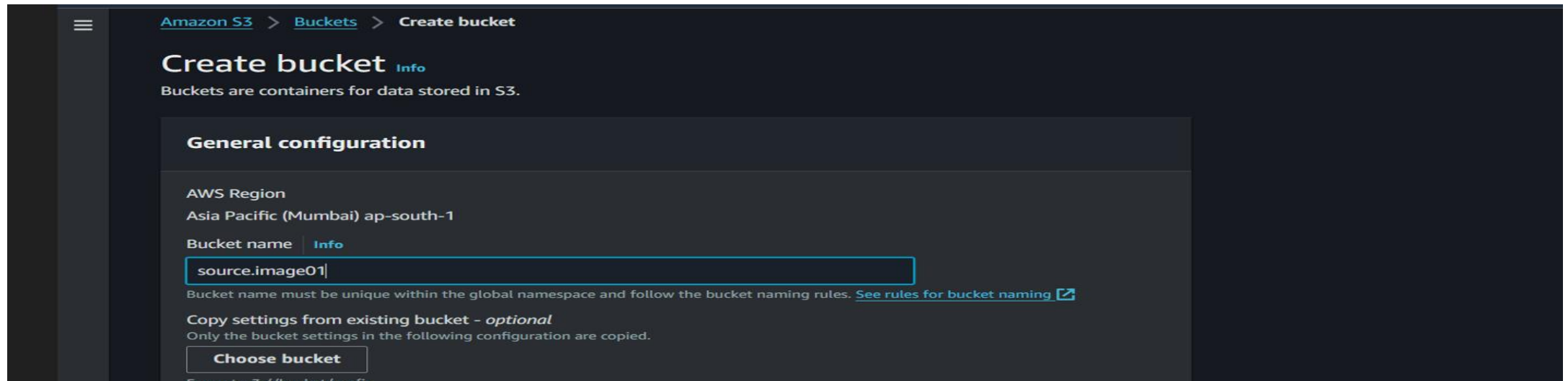
1. Navigate to the **Services** menu in the Top, then click on **S3** in the storage section.



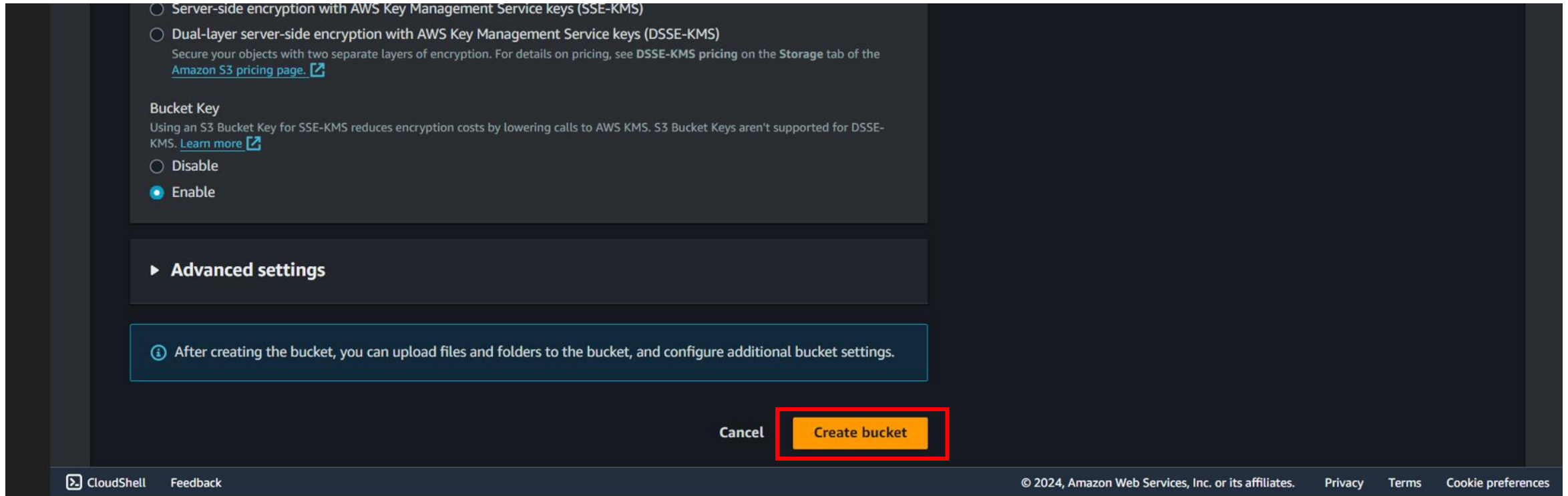
2. Click on Create Bucket button.



3. Create Source Bucket



4. Leave Other settings as Default and click on the **Create Bucket** button



5. Once the Bucket is created successfully, Select your S3 bucket.

- Click on the Copy ARN button to copy the ARN.
- Save the source bucket ARN in a text file for later use.
- `arn:aws:s3:::source.bucket01`

6. Create Destination Bucket

Create bucket [Info](#)
Buckets are containers for data stored in S3.

General configuration

AWS Region
Asia Pacific (Mumbai) ap-south-1

Bucket name [Info](#)

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

Copy settings from existing bucket - *optional*
Only the bucket settings in the following configuration are copied.

Choose bucket

Format: s3://bucket/prefix

Object Ownership [Info](#)
Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

☐ ACLs disabled (recommended)
All objects in this bucket are owned by this account.

☒ ACLs enabled
Objects in this bucket can be owned by other AWS

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

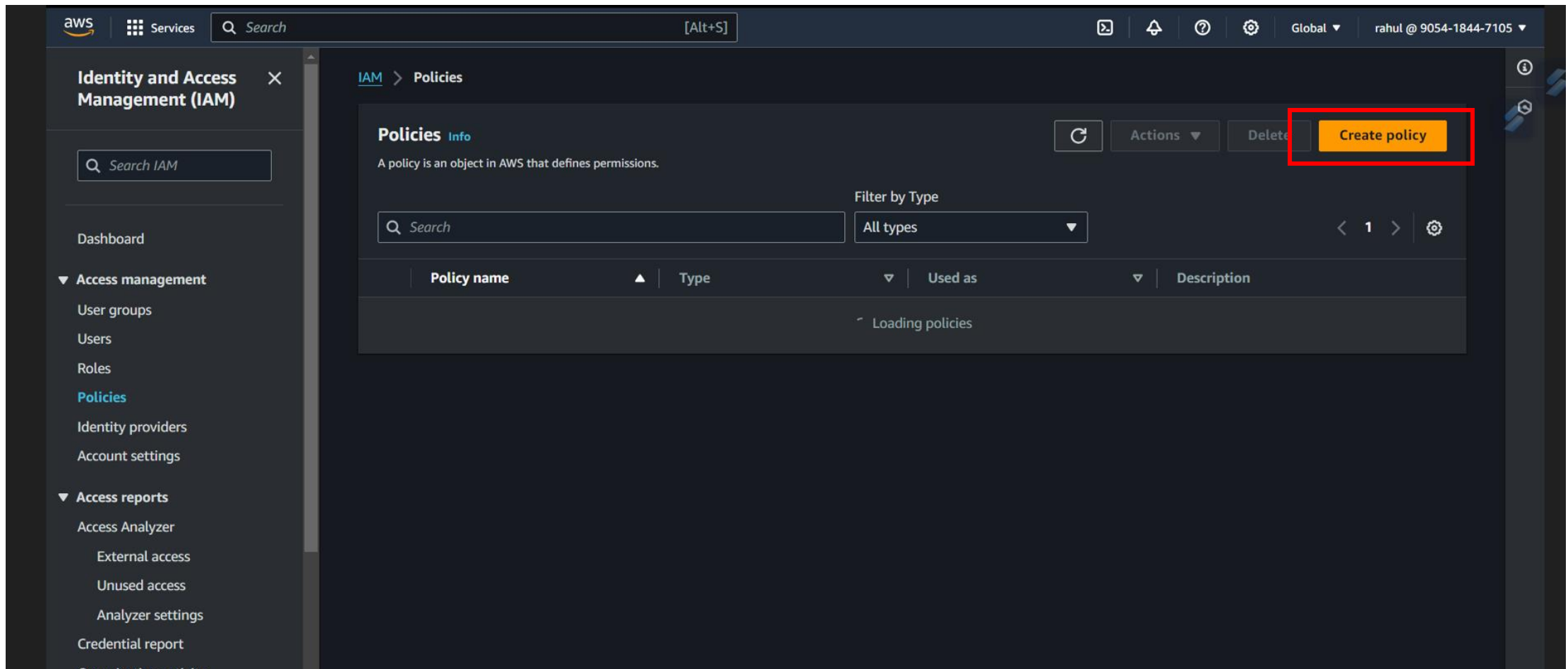
7. Leave Other settings as Default and click on the **Create Bucket** button

8. Once the Bucket is created successfully, Select your S3 bucket.

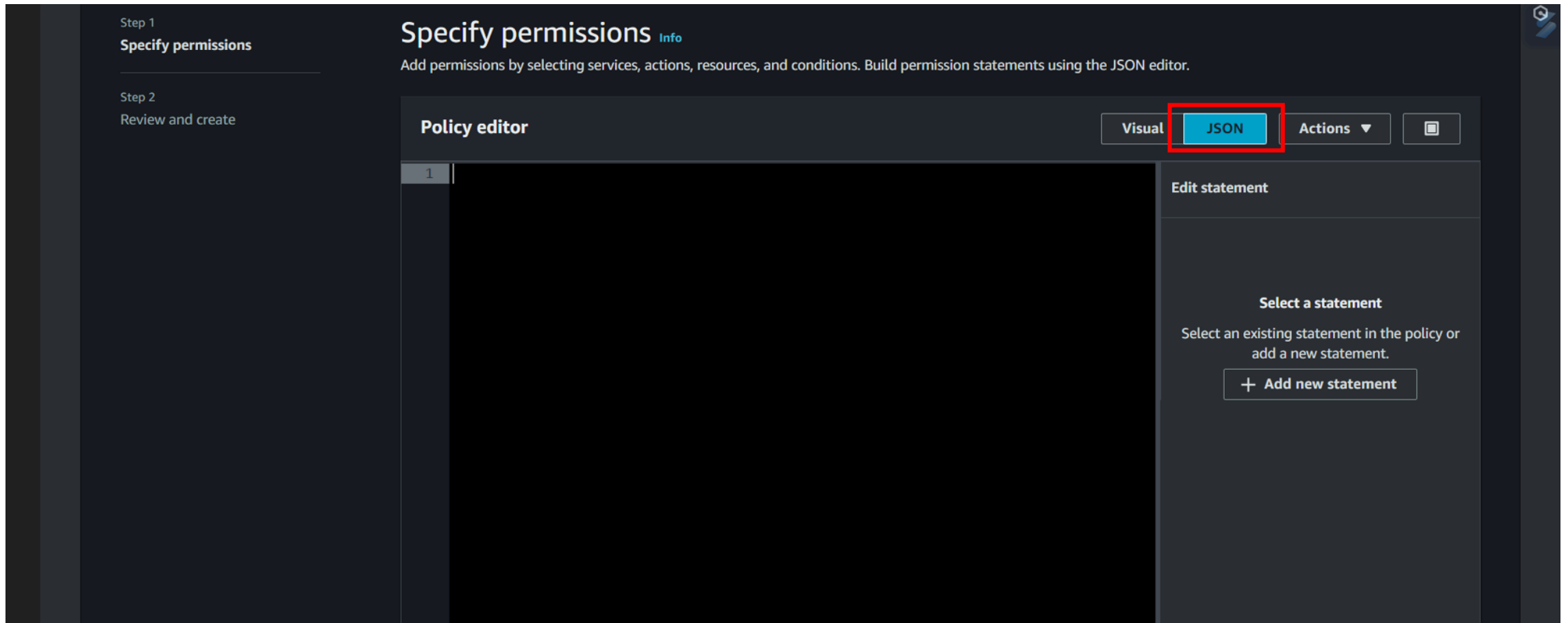
- Click on the Copy ARN button to copy the ARN.
- Save the source bucket ARN in a text file for later use.
- `arn:aws:s3:::destination.bucket01`

Task 3: Create an IAM Policy

1. Go to **Services** and Select **IAM** under **Security, Identity and Compliance**.
2. Click on **Policies** in the left navigation bar and click on the **Create policy** button.



3. Click on the **JSON** tab, Remove the existing code and copy-paste the below policy statement into the editor:



- **Policy JSON:**

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow", "Action": [ "logs:PutLogEvents",  
        "logs:CreateLogGroup",  
        "logs:CreateLogStream"  
      ],  
      "Resource": "arn:aws:logs:*:*:*"  
    },  
    {  
      "Effect": "Allow",  
      "Action": ["s3:GetObject"],  
      "Resource": "arn:aws:s3:::source.bucket01/*"  
    },  
  ],  
}
```

```
{  
  "Effect": "Allow", "Action": ["s3:PutObject"],  
  "Resource": "arn:aws:s3:::destination.bucket01/*"  
}  
]  
}
```

4. Leave Everything as default and click on **Next** button.

5. On the Review Policy page:

6. Enter **Policy Name** and Click on the **Create policy** button

Step 2
Review and create

Policy details

Policy name
Enter a meaningful name to identify this policy.
rahull
Maximum 128 characters. Use alphanumeric and '+, @, _' characters.

Description - optional
Add a short explanation for this policy.

Maximum 1,000 characters. Use alphanumeric and '+, @, _' characters.

Permissions defined in this policy [Info](#)

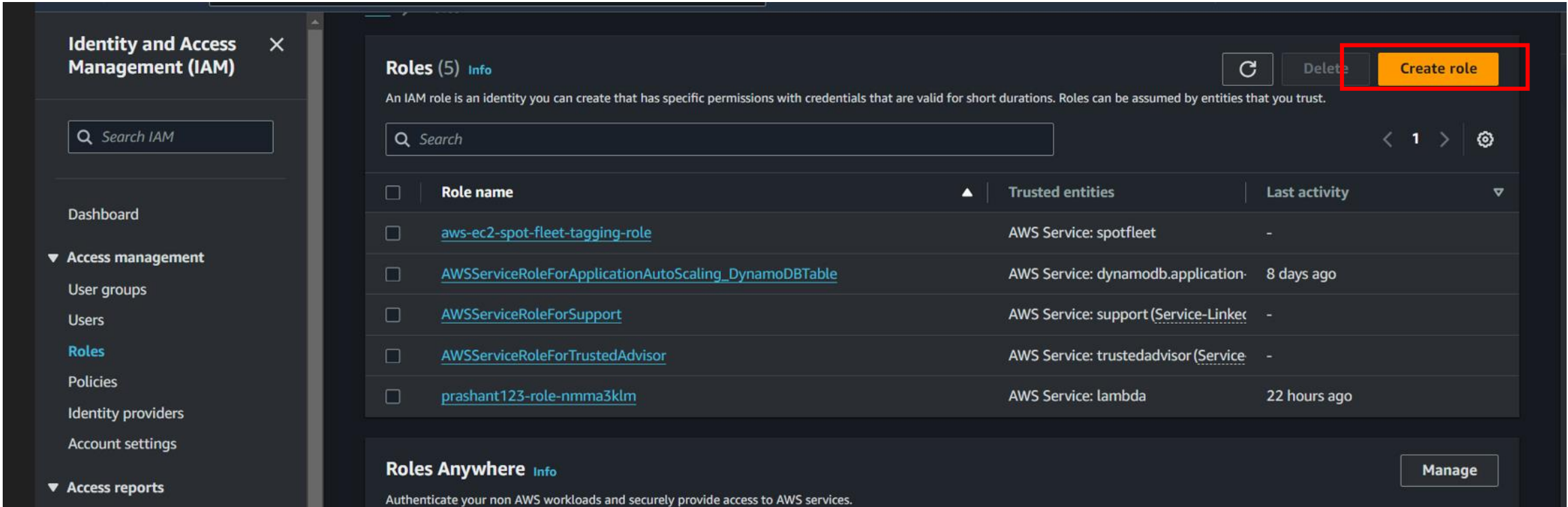
Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it

Allow (2 of 418 services) ☐ Show remaining 416 services

Service	Access level	Resource	Request condition
CloudWatch Logs	Limited: Write	region string like All	None
S3	Limited: Read, Write	Multiple	None

Task 4: Create an IAM Role

1. In the left menu, click on **Roles** and click on the **Create Role** button.



The screenshot displays the AWS IAM console interface. On the left, the 'Identity and Access Management (IAM)' sidebar is visible, with the 'Roles' link highlighted under the 'Access management' section. The main content area shows the 'Roles (5)' page. At the top right of this section, there are buttons for 'Refresh', 'Delete', and 'Create role'. The 'Create role' button is highlighted with a red rectangular box. Below the buttons is a search bar and a table listing existing roles. The table has columns for 'Role name', 'Trusted entities', and 'Last activity'. The roles listed are 'aws-ec2-spot-fleet-tagging-role', 'AWSServiceRoleForApplicationAutoScaling_DynamoDBTable', 'AWSServiceRoleForSupport', 'AWSServiceRoleForTrustedAdvisor', and 'prashant123-role-nmma3klm'. At the bottom of the console, there is a 'Roles Anywhere' section with a 'Manage' button.

Identity and Access Management (IAM)

Search IAM

Dashboard

▼ Access management

- User groups
- Users
- Roles**
- Policies
- Identity providers
- Account settings

▼ Access reports

Roles (5) Info

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

Search

<input type="checkbox"/>	Role name	Trusted entities	Last activity
<input type="checkbox"/>	aws-ec2-spot-fleet-tagging-role	AWS Service: spotfleet	-
<input type="checkbox"/>	AWSServiceRoleForApplicationAutoScaling_DynamoDBTable	AWS Service: dynamodb.application	8 days ago
<input type="checkbox"/>	AWSServiceRoleForSupport	AWS Service: support (Service-Linker)	-
<input type="checkbox"/>	AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service-Linker)	-
<input type="checkbox"/>	prashant123-role-nmma3klm	AWS Service: lambda	22 hours ago

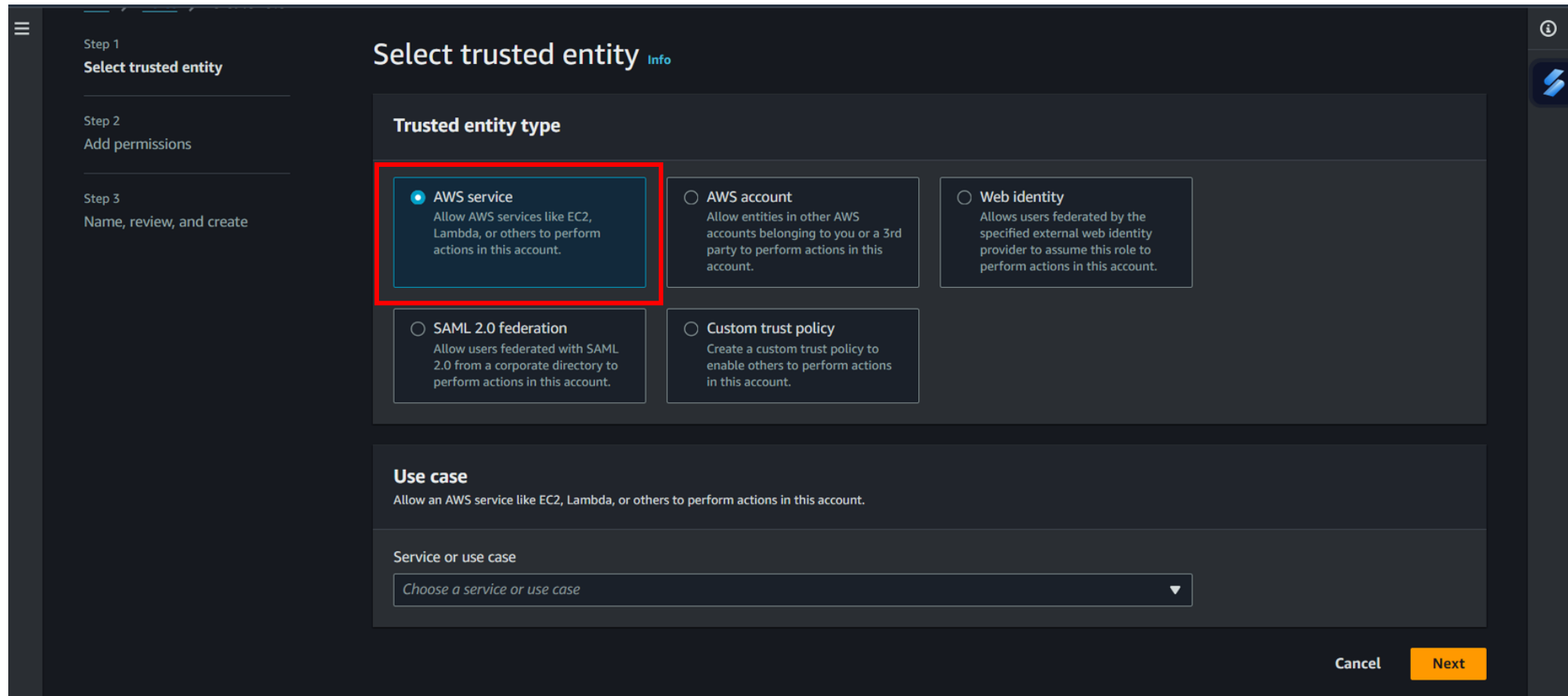
Roles Anywhere Info

Authenticate your non AWS workloads and securely provide access to AWS services.

Manage

2. Select Lambda from AWS Services list.

- From Trusted Entity Type: Select AWS Service
- From Use case: Select Lambda
- Click on Next button.



The screenshot shows the 'Select trusted entity' step in the AWS IAM console. The interface is dark-themed. On the left, a sidebar shows the progress: Step 1 (Select trusted entity), Step 2 (Add permissions), and Step 3 (Name, review, and create). The main area is titled 'Select trusted entity' with an 'Info' link. It contains two sections: 'Trusted entity type' and 'Use case'. In the 'Trusted entity type' section, five options are listed: 'AWS service' (selected and highlighted with a red box), 'AWS account', 'Web identity', 'SAML 2.0 federation', and 'Custom trust policy'. Each option has a radio button and a description. The 'Use case' section has a description and a dropdown menu labeled 'Service or use case' with the placeholder text 'Choose a service or use case'. At the bottom right, there are 'Cancel' and 'Next' buttons.

Step 1
Select trusted entity

Step 2
Add permissions

Step 3
Name, review, and create

Select trusted entity [Info](#)

Trusted entity type

- ☒ **AWS service**
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 3rd party to perform actions in this account.
- ☐ **Web identity**
Allows 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 corporate directory 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.

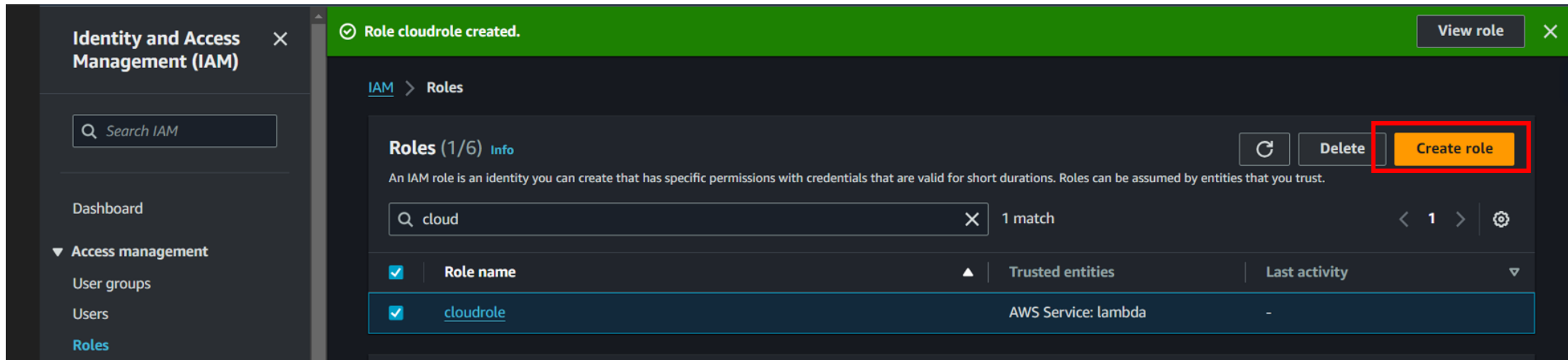
Service or use case

Cancel Next

4.Role Name: Enter **cloudrole**

5.Click on the **Create Role** button.

- You have successfully created an IAM role by name cloudrole.



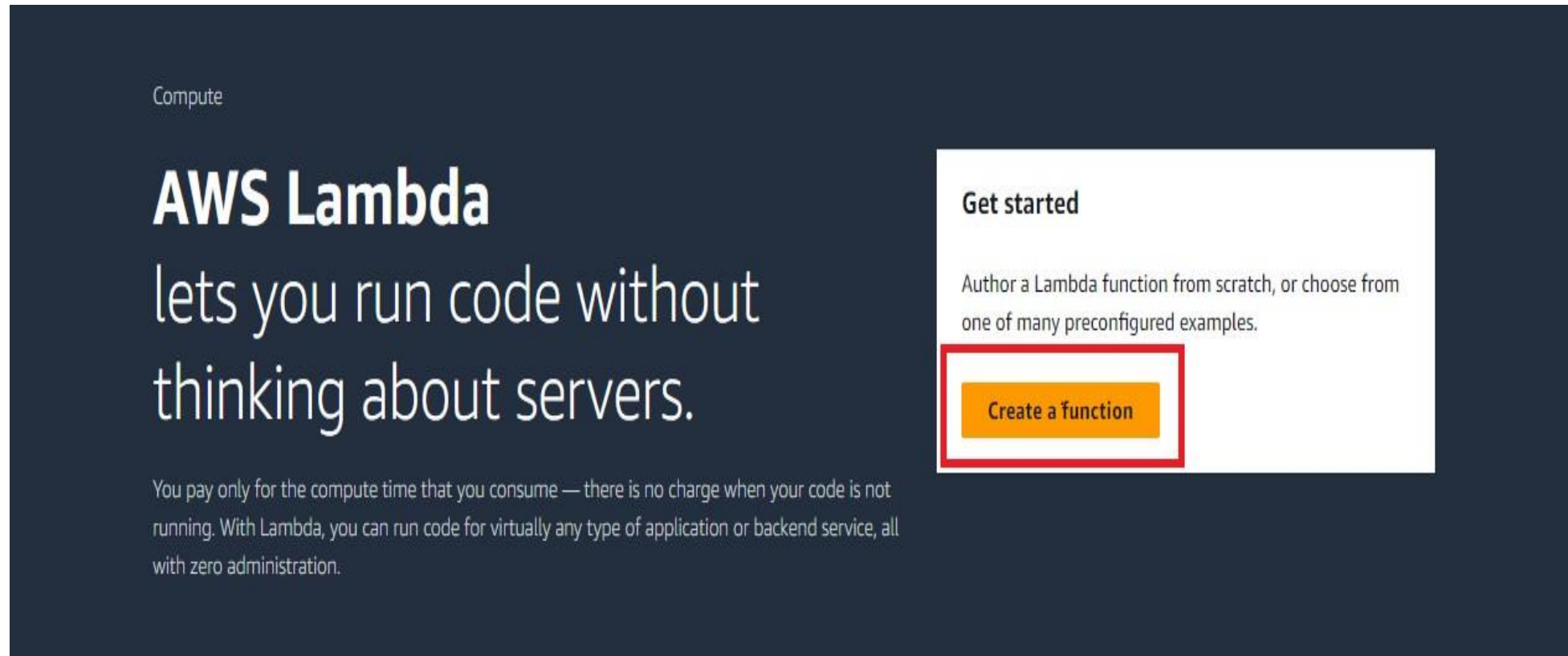
The screenshot displays the AWS Identity and Access Management (IAM) console. A green notification banner at the top states "Role cloudrole created." with a "View role" button. The left sidebar shows the "Identity and Access Management (IAM)" menu with options for "Dashboard", "Access management", "User groups", "Users", and "Roles". The main content area is titled "Roles (1/6)" and includes a search bar with "cloud" entered, showing "1 match". Below the search bar is a table of roles:

<input checked="" type="checkbox"/>	Role name	Trusted entities	Last activity
<input checked="" type="checkbox"/>	cloudrole	AWS Service: lambda	-

The "Create role" button is highlighted with a red rectangle.

Task 5: Creating Lambda function

1. Go to AWS Lambda Console, Navigate to functions section . Click **Create function**



2. Name it and select runtime and Leave all other settings as default.

The screenshot shows the AWS Lambda 'Create function' console page. The breadcrumb navigation at the top reads 'Lambda > Functions > Create function'. The main heading is 'Create function' with an 'Info' link. Below the heading, it says 'Choose one of the following options to create your function.' There are three options: 'Author from scratch' (selected and highlighted with a red box), 'Use a blueprint', and 'Container image'. The 'Author from scratch' option includes the text 'Start with a simple Hello World example.' Below the options is the 'Basic information' section. It contains a 'Function name' field with the value 'lamdafunction' and a note 'Use only letters, numbers, hyphens, or underscores with no spaces.' The 'Runtime' section is highlighted with a red box; it shows a dropdown menu set to 'Node.js 18.x' and a note: 'Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.' Below the runtime is the 'Architecture' section, which has a dropdown menu set to 'x86_64'. At the bottom, the 'Permissions' section is partially visible. On the right side of the console, there is a 'Tutorials' sidebar with a 'Create a simple web app' tutorial card, which includes a list of steps and a 'Start tutorial' button.

[Lambda](#) > [Functions](#) > **Create function**

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.

Basic information

Function name
Enter a name that describes the purpose of your function.

lamdafunction

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

Permissions [Info](#)

Info **Tutorials** ×

Learn how to implement common use cases in AWS Lambda.

Create a simple web app ^

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

[Learn more](#)

Start tutorial

3. Change Default execution role and create function

▼ 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.

cloudrole ▼

[View the cloudrole role](#) on the IAM console.

► Advanced settings

[Cancel](#) **Create function**

app

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

[Learn more](#)

Start tutorial

4. Edit Environment Variables

Code

Test

Monitor

Configuration

Aliases

Versions

General configuration

Triggers

Permissions

Destinations

Function URL

Environment variables

Tags

VPC

RDS databases

General configuration

Info

Edit

Description	Memory	Ephemeral storage
-	128 MB	512 MB
Timeout	SnapStart	Info
0 min 3 sec	None	

Create a simple web app

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

[Learn more](#)

Start tutorial

Edit environment variables

Environment variables

You can define environment variables as key-value pairs that are accessible from your function code. These are useful to store configuration settings without the need to change function code. [Learn more](#)

Key

bucket_key

Value

destination.bucket01

Remove

Add environment variable

► Encryption configuration

Cancel

Save

Learn how to implement common use cases in AWS Lambda.

Create a simple web app ^

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

[Learn more](#)

Start tutorial

Task 6: Test Lambda Function

- *Go to AWS Lambda console. Navigate to Functions section.
- *open function then will be created
- *open test console
- *template=s3-put

Test event [Info](#)

Save Test

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

event123

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

s3-put

Create a simple web app

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

[Learn more](#)

Start tutorial

EVENT JSON:

```
{  
  "Records": [  
    {  
      "eventVersion": "2.0",  
      "eventSource": "aws:s3",  
      "awsRegion": "us-east-1",  
      "eventTime": "1970-01-01T00:00:00.000Z",  
      "eventName": "ObjectCreated:Put", "userIdentity": {  
        "principalId": "EXAMPLE"  
      },  
      "requestParameters": { "sourceIPAddress": "127.0.0.1"
```

```
"sourceIPAddress": "127.0.0.1"
},
"responseElements": {
  "x-amz-request-id": "EXAMPLE123456789", "x-amz-id-2":
  "EXAMPLE123/5678abcdefghijklmbdaisawesome/mnopqrstuvwxyzAB
  CDEFGH"
}, "s3": {"s3SchemaVersion": "1.0",
"configurationId": "testConfigRule", "bucket": {
  "name": "arn:aws:s3:::source.bucket01", "ownerIdentity": {
    "principalId": "EXAMPLE"
  },
  "arn": "arn:aws:s3:::source.bucket01"
},
"object": {
  "key": "18981044.jpg",
  "size": 1024,
  "eTag": "0123456789abcdef0123456789abcdef", "sequencer": "0A1B2C3D4E5F678901" } } } ]
}
```


Now We can Test:

The screenshot displays the AWS Lambda console interface. At the top, a navigation bar includes tabs for 'Code', 'Test', 'Monitor', 'Configuration', 'Aliases', and 'Versions'. The 'Test' tab is currently selected. Below the navigation bar, a green banner indicates a successful function execution with a checkmark icon and the text 'Executing function: succeeded (logs [link])'. A 'Details' link is also present. The main section is titled 'Test event' with an 'Info' link. On the right side of this section are 'Save' and 'Test' buttons; the 'Test' button is highlighted with a red rectangle. Below the title, a text block explains: 'To invoke your function without saving an event, configure the JSON event, then choose Test.' The 'Test event action' section contains two radio buttons: 'Create new event' (selected) and 'Edit saved event'. The 'Event name' section has a text input field containing 'event123' and a note: 'Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.' The 'Event sharing settings' section features two radio buttons: 'Private' (selected) and 'Shareable'. Descriptive text for each setting is provided, along with 'Learn more' links. At the bottom, the text 'Template - optional' is visible. On the right side of the console, a sidebar contains a 'Tutorials' section with the title 'Create a simple web app', a description of the tutorial's goals, a list of bullet points, a 'Learn more' link, and a 'Start tutorial' button.

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

✓ Executing function: succeeded (logs [link])
▶ Details

Test event Info

Save Test

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

event123

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

Info | **Tutorials** | [Close]

Learn how to implement common use cases in AWS Lambda.

Create a simple web app ^

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

[Learn more](#)

Start tutorial

Task 7: Creating S3 Trigger

*Add trigger

*Select s3

*choose source Bucket name

*Now Add

The screenshot displays the AWS Lambda console interface for a function named 'lamdafunction'. The breadcrumb navigation at the top indicates the path: [Lambda](#) > [Functions](#) > [lamdafunction](#). The main header area includes the function name 'lamdafunction' and buttons for 'Throttle', 'Copy ARN', and 'Actions'. Below this, the 'Function overview' tab is selected, showing a diagram of the function's architecture. The diagram consists of a box labeled 'lamdafunction' with a Lambda icon, connected to an 'S3' bucket icon. A red rectangular box highlights the '+ Add trigger' button located below the S3 icon. To the right of the diagram, there is a '+ Add destination' button. The right sidebar contains a 'Tutorials' section with the title 'Create a simple web app' and a description: 'Learn how to implement common use cases in AWS Lambda.' Below this, there is a list of bullet points: 'Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage' and 'Invoke your function through its function URL'. A 'Start tutorial' button is at the bottom of the sidebar. The bottom navigation bar includes tabs for 'Code', 'Test', 'Monitor', 'Configuration', 'Aliases', and 'Versions'.

Function overview

Diagram Template

lamdafunction

Layers (0)

S3

+ Add trigger

+ Add destination

Export to Application Composer

Download

Description

-

Last modified

2 minutes ago

Function ARN

arn:aws:lambda:ap-south-1:905418447105:function:lamdafunction

Function URL [Info](#)

-

Info Tutorials

Learn how to implement common use cases in AWS Lambda.

Create a simple web app

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

[Learn more](#)

Start tutorial

Code Test Monitor Configuration Aliases Versions



[Lambda](#) > Add triggers

Add trigger

Trigger configuration [Info](#)



S3

aws

asynchronous

storage



Bucket

Choose or enter the ARN of an S3 bucket that serves as the event source. The bucket must be in the same region as the function.



s3/source.bucket01



Bucket region: ap-south-1

Event types

Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.



All object create events

Prefix - optional

Enter a single optional prefix to limit the notifications to objects with keys that start with matching characters.

e.g. images/

Suffix - optional

Enter a single optional suffix to limit the notifications to objects with keys that end with matching characters.

Info

Tutorials



Learn how to implement common use cases in AWS Lambda.

Create a simple web app ^

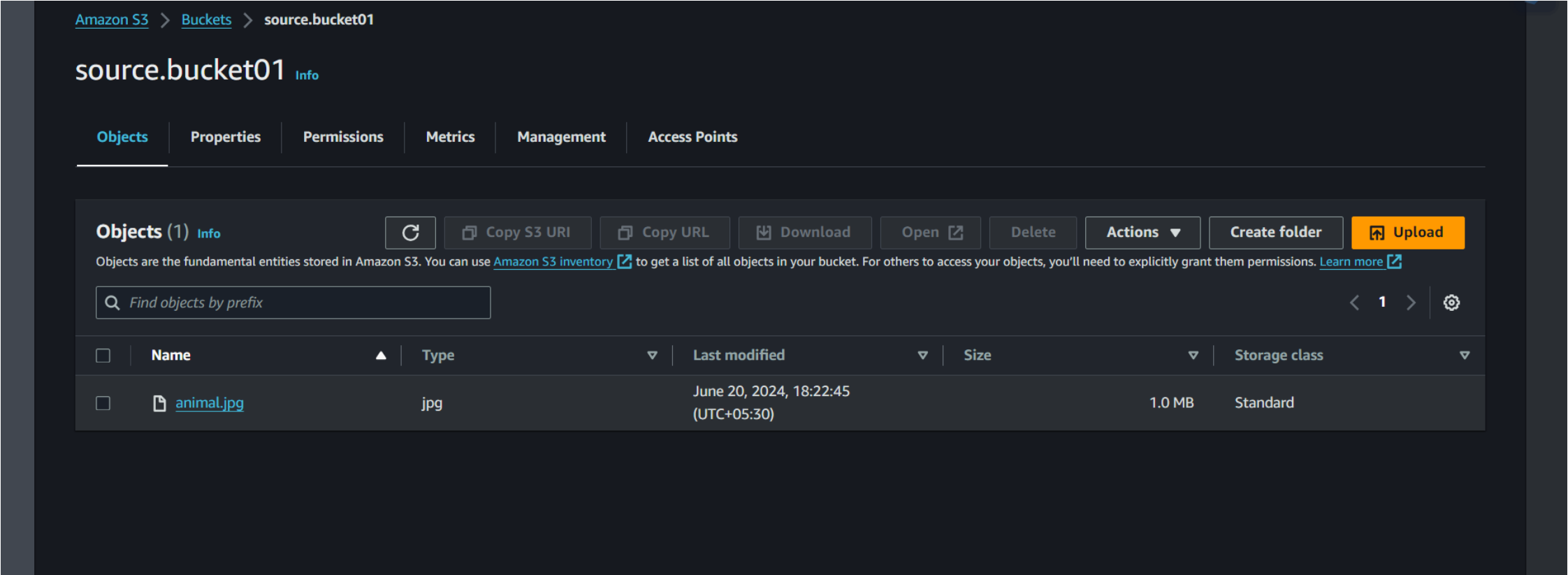
In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

[Learn more](#)

Start tutorial

Task 8: Upload image in Source Bucket



❖ Original Image



❖ Destination Bucket

[Amazon S3](#) > [Buckets](#) > destination.bucket01

destination.bucket01

Info

Objects

Properties

Permissions

Metrics

Management

Access Points

Objects (1)

Info

↺

Copy S3 URI

Copy URL

Download

Open

Delete

Actions

Create folder


Upload

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Find objects by prefix

< 1 >

⚙

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	 animal.jpg	jpg	June 20, 2024, 18:31:30 (UTC+05:30)	8.4 KB	Standard

❖ Resize Image

