

**Project Name:** Image Analysis using AWS Rekognition via Lambda Function and S3 bucket

**Description:** The goal of this project is to build a serverless image processing system using AWS Lambda, Amazon S3, and Amazon Rekognition. The system will automatically label detection and object detection for images uploaded to an S3 bucket.

By completing this project, you will have implemented a serverless image processing system that automatically analyzes and tags images using AWS Lambda, Amazon S3, and Amazon Rekognition. This project demonstrates your skills in serverless architecture, event-driven computing, and image processing using AWS services.

### **What is AWS Rekognition?**

Rekognition is one of the AWS services to perform image and video analysis. So here all we need to provide is the image or video to the AWS Rekognition service and it will help us to identify an object, people, text, activities, and scenes.

### **Benefits of using Amazon Rekognition are as follows:**

- Integrating powerful image and video analysis into your apps.
- Deep learning-based image and video analysis.
- Scalable image analysis.
- Integration with other AWS services.
- Low cost

### **Common use cases for using Amazon Rekognition mentioned in the following:**

- Searchable image and video libraries
- Face-based user verification
- Sentiment and demographic analysis
- Facial Search
- Unsafe content detection
- Celebrity recognition
- Text detection
- Custom labels

### **For Image analysis, we are using four services of AWS.**

- IAM
- S3
- Lambda
- Rekognition

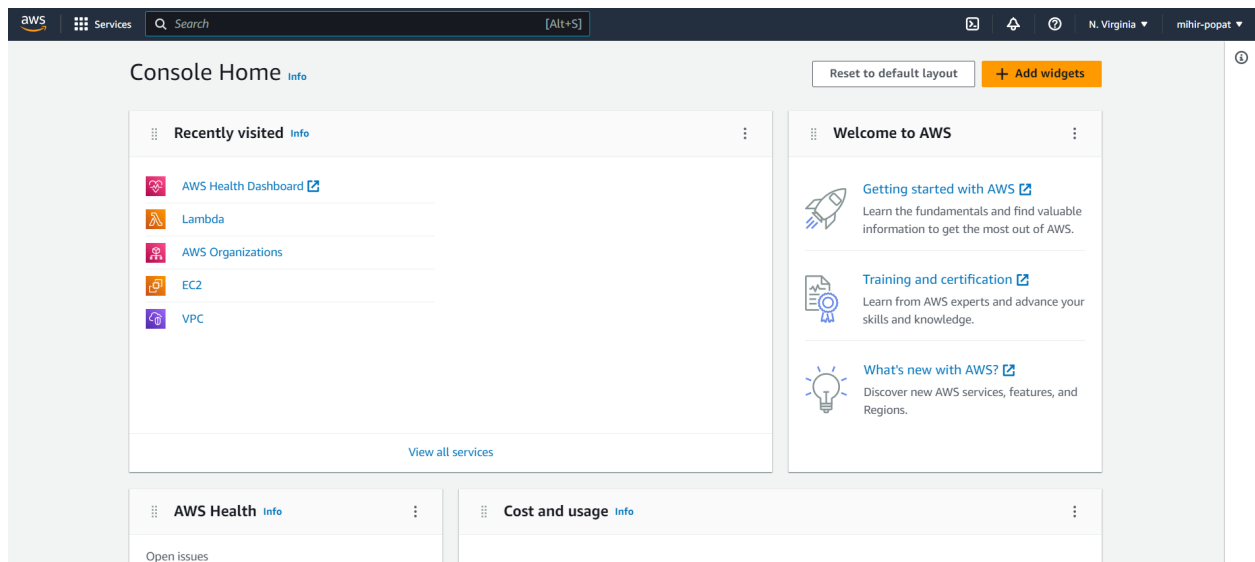
## Flow for image analysis will be

- Firstly, we are going to read an image from the S3 bucket via a lambda function.
- And in the second step we will pass that image to rekognition service via calling rekognition API. In response to this, rekognition API will return labels.

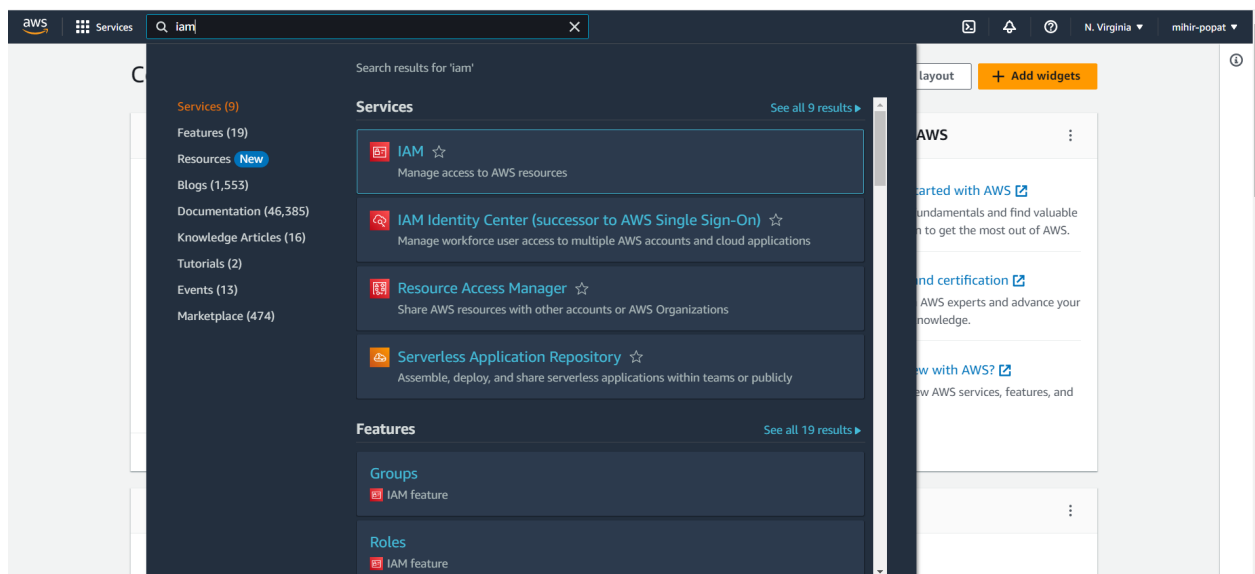
## Implementation Steps:

### Step 1: Creating an IAM role:

- Go to the AWS Management console.



- Search for the IAM service and enter.



The screenshot shows the AWS IAM dashboard. On the left, the 'Identity and Access Management (IAM)' sidebar is visible with a search bar and a list of navigation items: Dashboard, Access management (User groups, Users, Roles, Policies, Identity providers, Account settings), Access reports (Access analyzer), and a search bar. The main content area is titled 'IAM dashboard' and displays 'Security recommendations' with a red badge indicating 1 recommendation. The recommendations listed are:
 

- Root user has MFA**: Having multi-factor authentication (MFA) for the root user improves security for this account.
- Root user has no active access keys**: Using access keys attached to an IAM user instead of the root user improves security.
- Update your access permissions for AWS Billing, Cost Management, and Account consoles**: We are replacing the following IAM actions for Billing, Cost Management, and Account consoles with granular IAM actions: `aws-portal:ViewBilling`, `aws-portal:ModifyBilling`, `aws-portal:ViewAccount`, `aws-portal:ModifyAccount`, `aws-portal:ViewPaymentMethods`, `aws-portal:ModifyPaymentMethods`, `aws-portal:ViewUsage`, `purchase-orders:ViewPurchaseOrders`, and `purchase-orders:ModifyPurchaseOrders`. To ensure you don't lose access to AWS Billing, Cost Management, and Account console based features, update your existing IAM policies to include the new IAM actions before July 2023. Examples of features impacted include AWS Cost Explorer, AWS Budgets, Billing console, and more. For more information, please visit [blog](#).

 A 'View affected policies' button is present next to the third recommendation.

- In the IAM service on the left side click on Roles In that click on Create Role button.

The screenshot shows the 'Roles' page in the AWS IAM console. The left sidebar is the same as the previous screenshot. The main content area is titled 'Roles (2)' and includes a search bar and a table of existing roles. The table has columns for 'Role name', 'Trusted entities', and 'Last activity'. The roles listed are:
 

Role name	Trusted entities	Last activity
<a href="#">AWSServiceRoleForSupport</a>	AWS Service: support (Service-Linked Role)	-
<a href="#">AWSServiceRoleForTrustedAdvisor</a>	AWS Service: trustedadvisor (Service-Linked Role)	-

 Below the table, there is a section for 'Roles Anywhere' with a 'Manage' button.

- Select the type of trusted entity as an AWS service by default.

The screenshot shows the 'Select trusted entity' step in the 'Create role' wizard. The left sidebar shows the progress: Step 1 (Select trusted entity), Step 2 (Add permissions), and Step 3 (Name, review, and create). The main content area is titled 'Select trusted entity' and displays 'Trusted entity type' with five options:
 

- AWS service** (selected): 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.

- In Choose a use case select Lambda and then click on Next: Permission button.

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.

**Common use cases**

☐ **EC2**  
Allows EC2 instances to call AWS services on your behalf.

☒ **Lambda**  
Allows Lambda functions to call AWS services on your behalf.

Use cases for other AWS services:  
Choose a service to view use case

[Cancel](#) [Next](#)

- In the Attach permissions policies select two policies :
- **AmazonRekognitionFullAccess**
- **AWSLambdaExecute**

IAM > Roles > Create role

Step 1  
[Select trusted entity](#)

Step 2  
**Add permissions**

Step 3  
Name, review, and create

## Add permissions [Info](#)

**Permissions policies (845)** [Info](#)

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

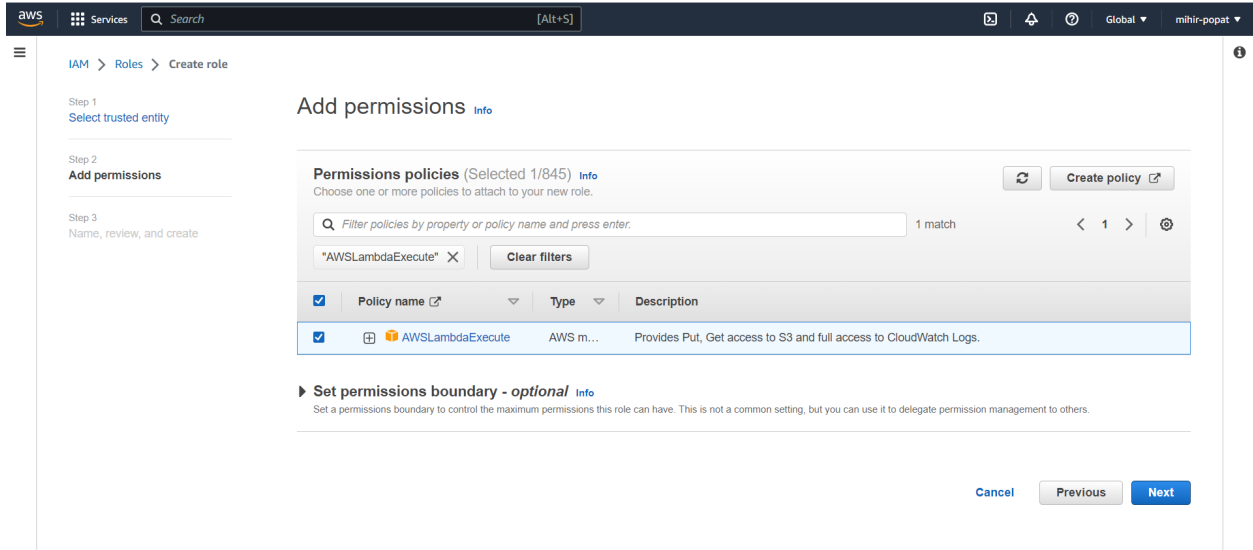
Filter policies by property or policy name and press enter. 1 match

\*AmazonRekognitionFullAccess\* X Clear filters

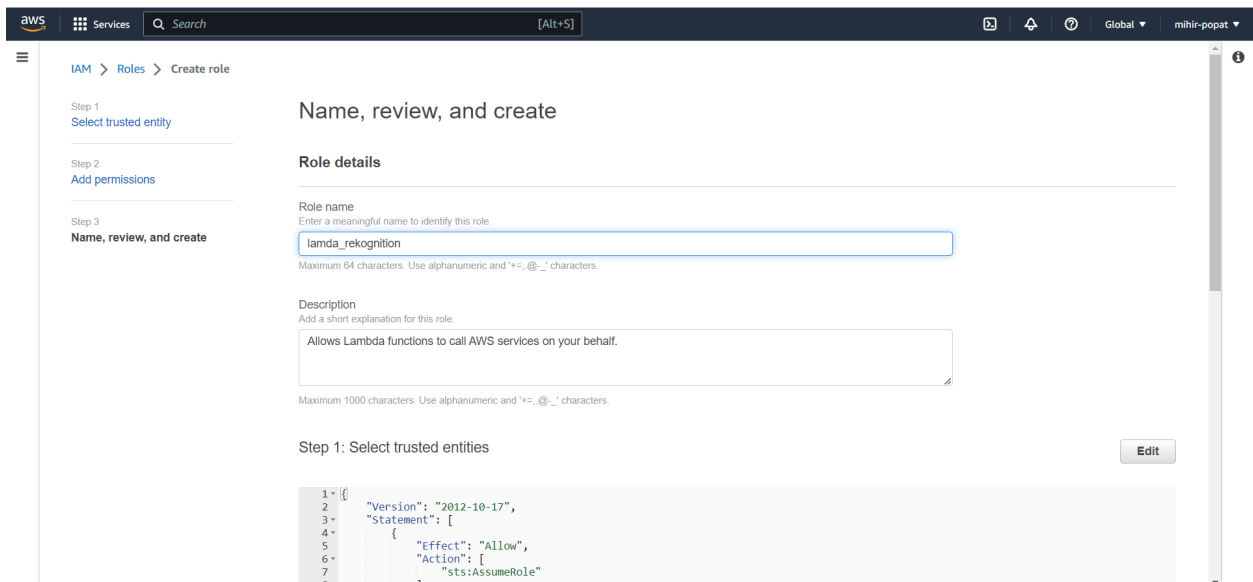
<input type="checkbox"/>	Policy name <a href="#">↗</a>	Type	Description
<input type="checkbox"/>	AmazonRekognition...	AWS m...	Access to all Amazon Rekognition APIs

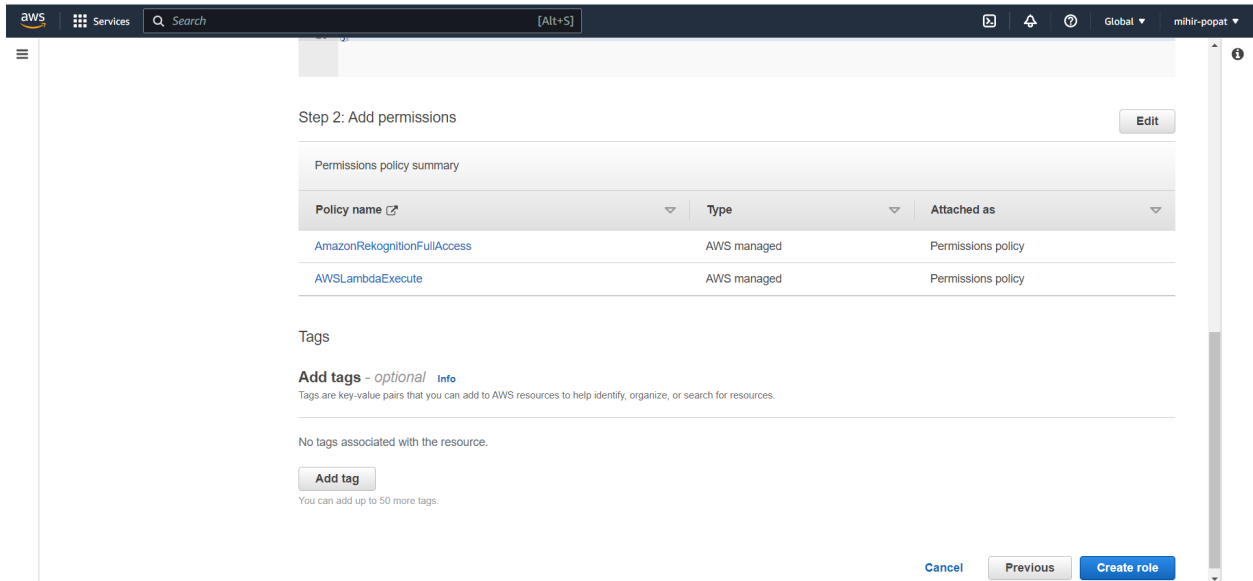
► **Set permissions boundary - optional** [Info](#)  
Set a permissions boundary to control the maximum permissions this role can have. This is not a common setting, but you can use it to delegate permission management to others.

[Cancel](#) [Previous](#) [Next](#)

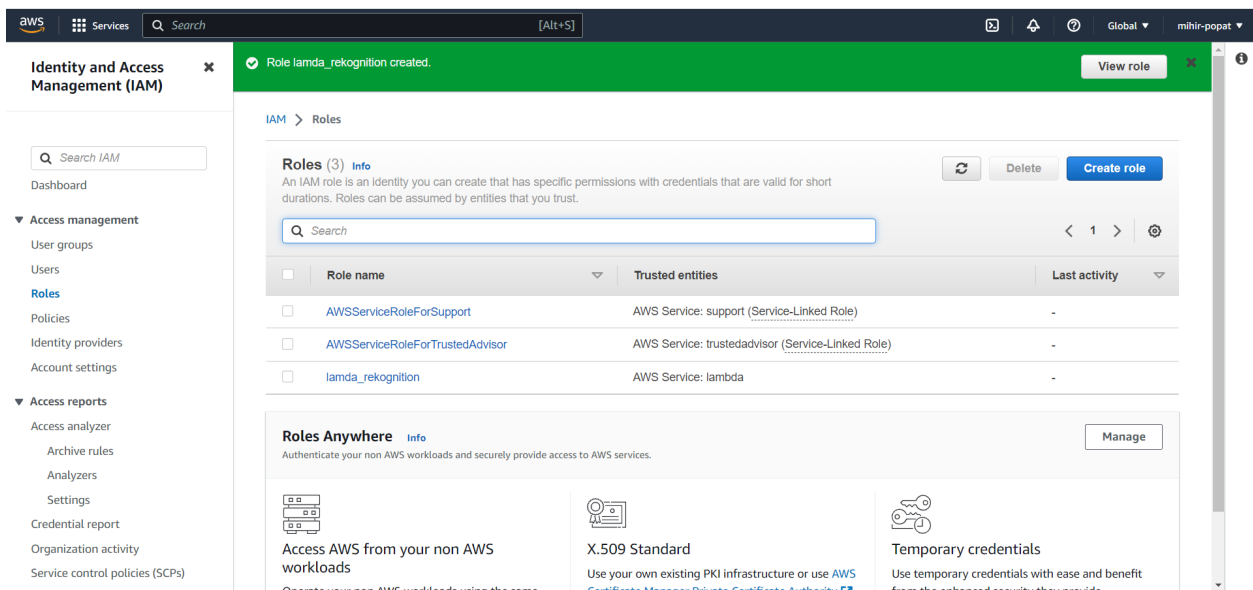


- Click on Next: Tags button.
- Add tags part is optional so click on Next: Review button.
- Give a name to your role. You can give any name to your role [for eg.lamda\_rekognition ] and click on the Create role button.



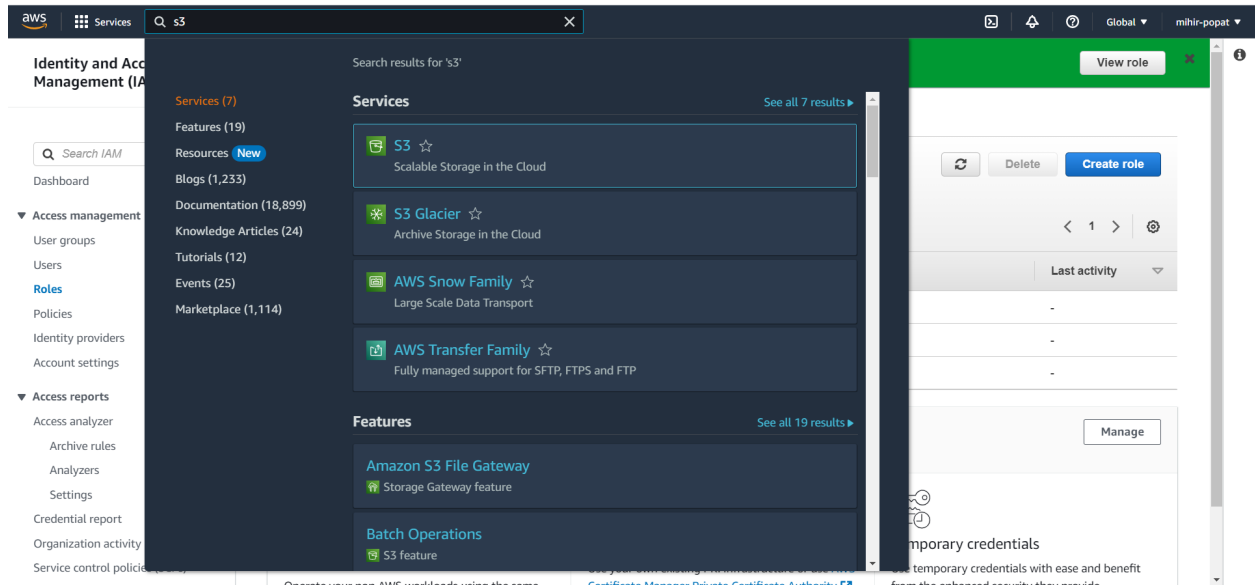


- Your role is ready.

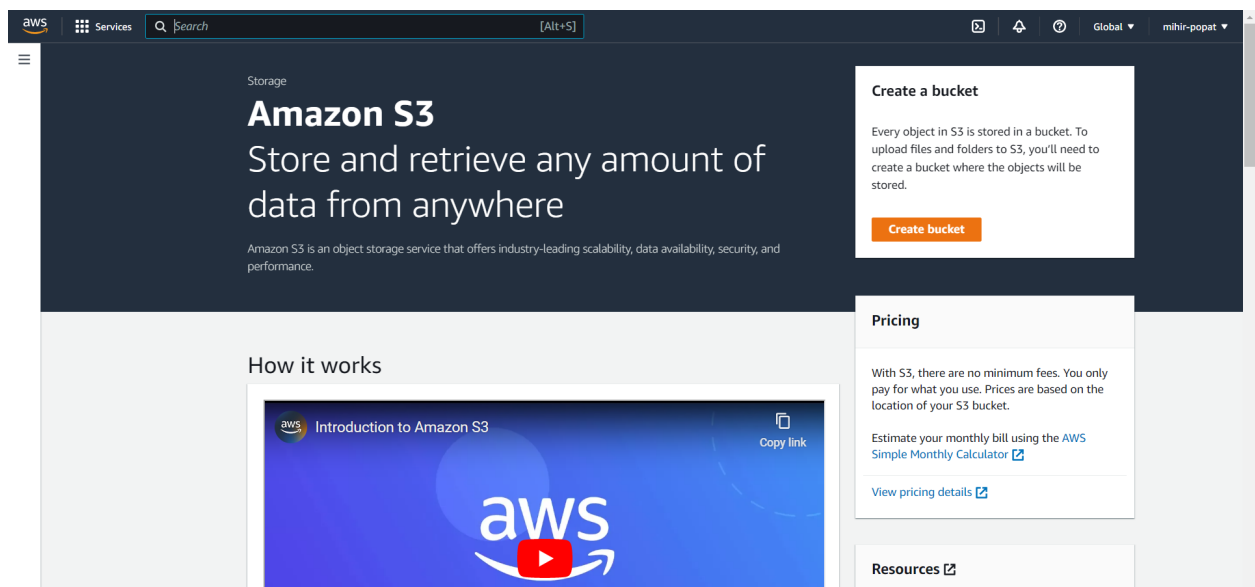


## Step 2: Create an S3 bucket to store images:

- Go to the AWS Management console.
- Search for the S3 service and enter.



- Click on the Create bucket button.



- Give any unique name to you bucket [for eg rekognition].

**Create bucket** [Info](#)

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

**General configuration**

Bucket name

Bucket name must be globally unique and must not contain spaces or uppercase letters. See rules for bucket naming [?](#)

AWS Region  
 US East (N. Virginia) us-east-1

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

**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. Access to this bucket and its objects is specified using

☐ **ACLs enabled**  
 Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be

**Amazon S3** [×](#)

**Successfully created bucket "rekognition-mihir"** [View details](#) [×](#) [?](#)

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

Amazon S3 > Buckets

**Account snapshot** [View Storage Lens dashboard](#)

Storage lens provides visibility into storage usage and activity trends. [Learn more](#)

**Buckets (1)** [Info](#) [Refresh](#) [Copy ARN](#) [Empty](#) [Delete](#) [Create bucket](#)

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

	Name	AWS Region	Access	Creation date
<input type="radio"/>	rekognition-mihir	US East (N. Virginia) us-east-1	Bucket and objects not public	May 14, 2023, 10:47:02 (UTC+05:30)

**Storage Lens**

Dashboards

AWS Organizations settings

Feature spotlight [3](#)

[AWS Marketplace for S3](#)

- Keep all default settings as it is and click on the Create bucket button.
- Once your bucket is created click on your bucket name. In that click on the upload button and drag and drop any image that you want and click on the upload button directly. Once the image is uploaded you can see the image as follows



Amazon S3

Buckets

Access Points

Object Lambda Access Points

Multi-Region Access Points

Batch Operations

IAM Access Analyzer for S3

Block Public Access settings for this account

Storage Lens

Dashboards

AWS Organizations settings

Feature spotlight 3

AWS Marketplace for S3

Amazon S3 > Buckets > rekognition-mihir

rekognition-mihir Info

Objects | Properties | Permissions | Metrics | Management | Access Points

Objects (0)

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](#)

Refresh

Copy S3 URI

Copy URL

Download

Open

Delete

Actions

Create folder

Upload

Find objects by prefix

< 1 >

	Name	Type	Last modified	Size	Storage class
No objects					
You don't have any objects in this bucket.					
<div>Upload</div>					

Amazon S3 > Buckets > rekognition-mihir > Upload

Upload Info

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](#)

Drag and drop files and folders you want to upload here, or choose **Add files**, or **Add folders**.

Files and folders (1 Total, 171.2 KB)

Remove

Add files

Add folder

All files and folders in this table will be uploaded.

Find by name

< 1 >

	Name	Folder	Type	Size
<input type="checkbox"/>	man.jpg	-	image/jpeg	171.2 KB

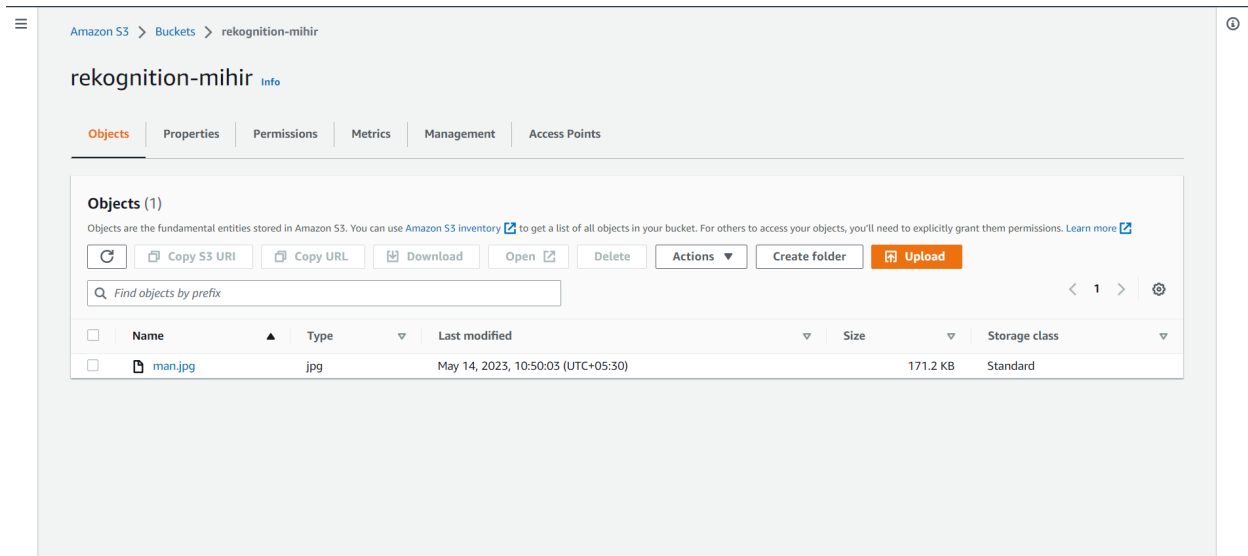
Destination

Destination

s3://rekognition-mihir

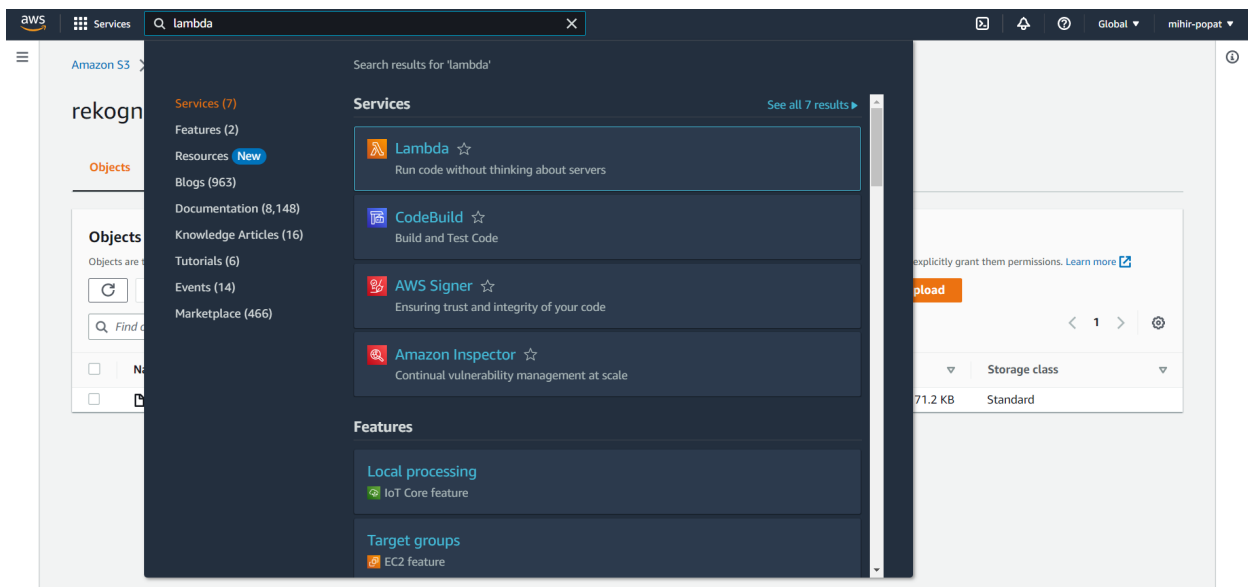
Destination details

Bucket settings that impact new objects stored in the specified destination.

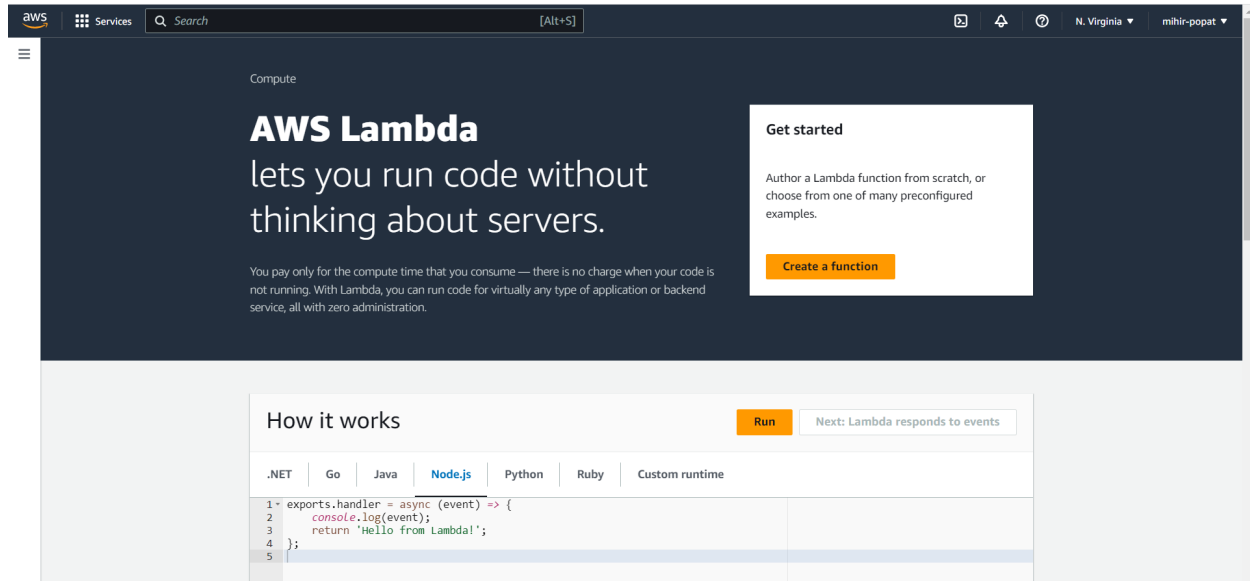


### Step 3: Create a Lambda Function:

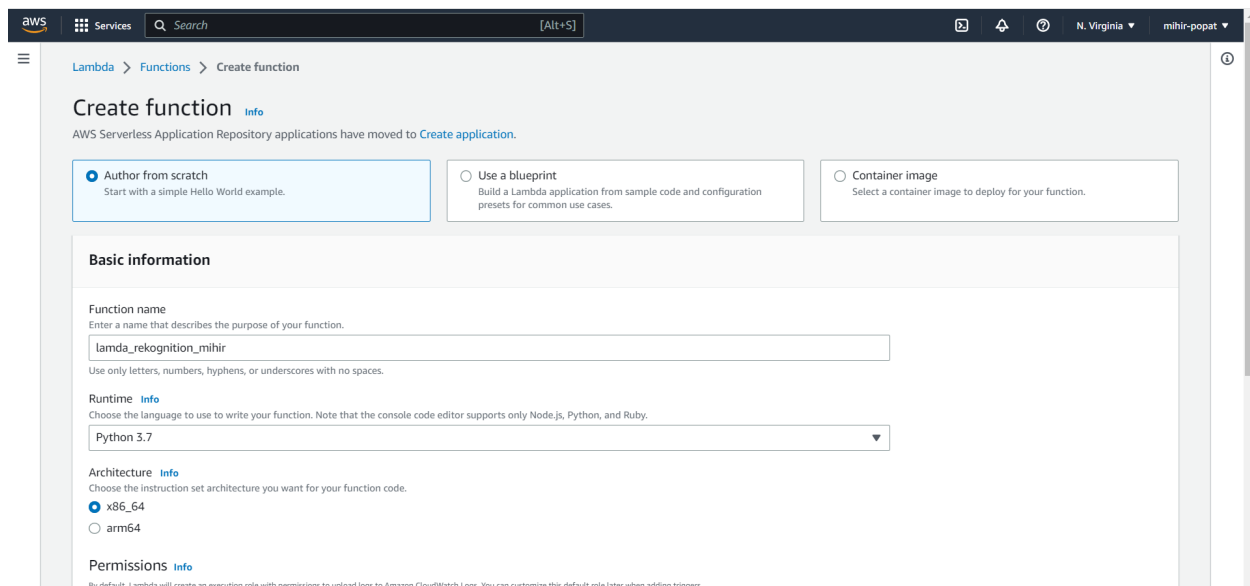
- Go to the AWS Management console.
- Search for the Lambda service and enter.



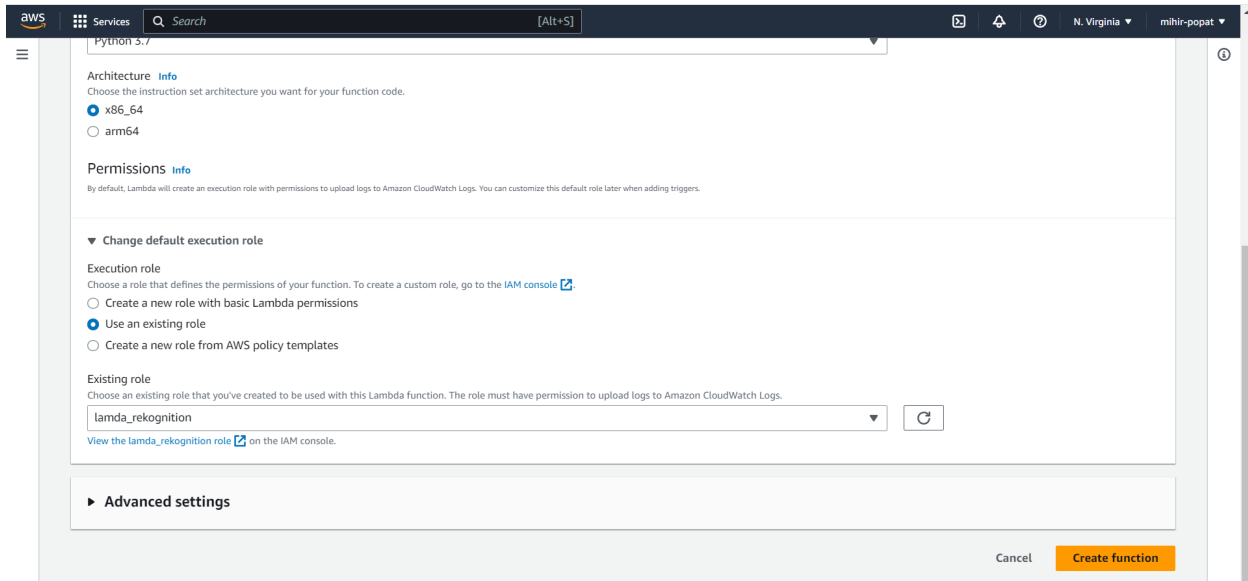
- After coming onto the lambda service page click on the Create function button.



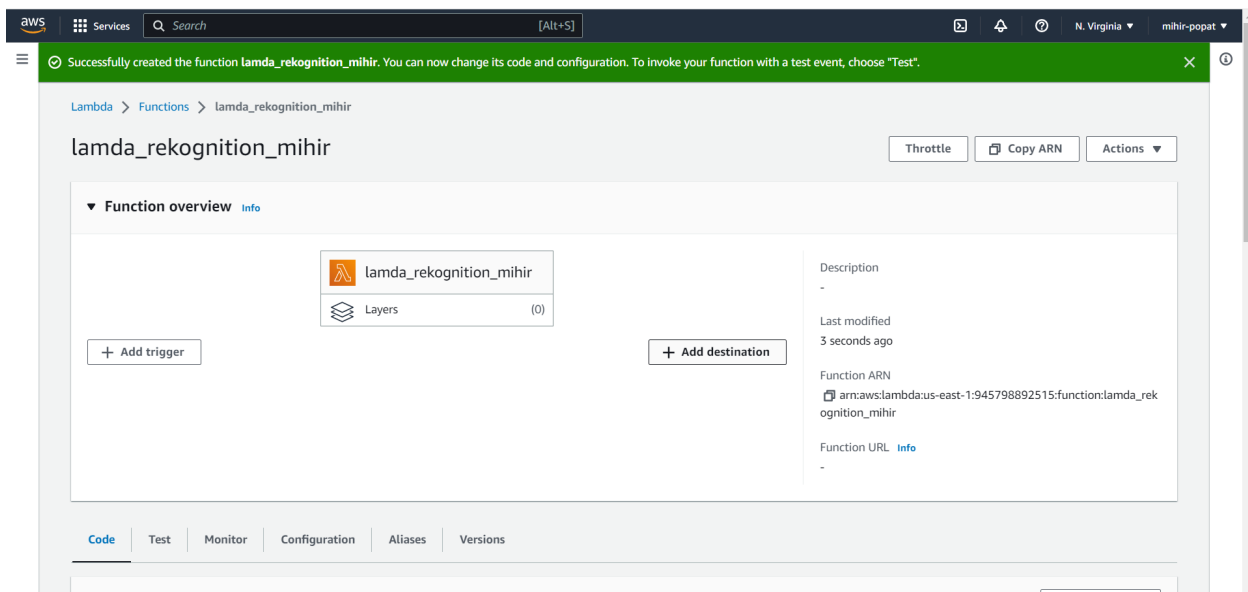
- Select the Author from Scratch default option.
- For function name give any name of your choice[for eg lamda\_rekognition].
- In the Runtime select python 3.7



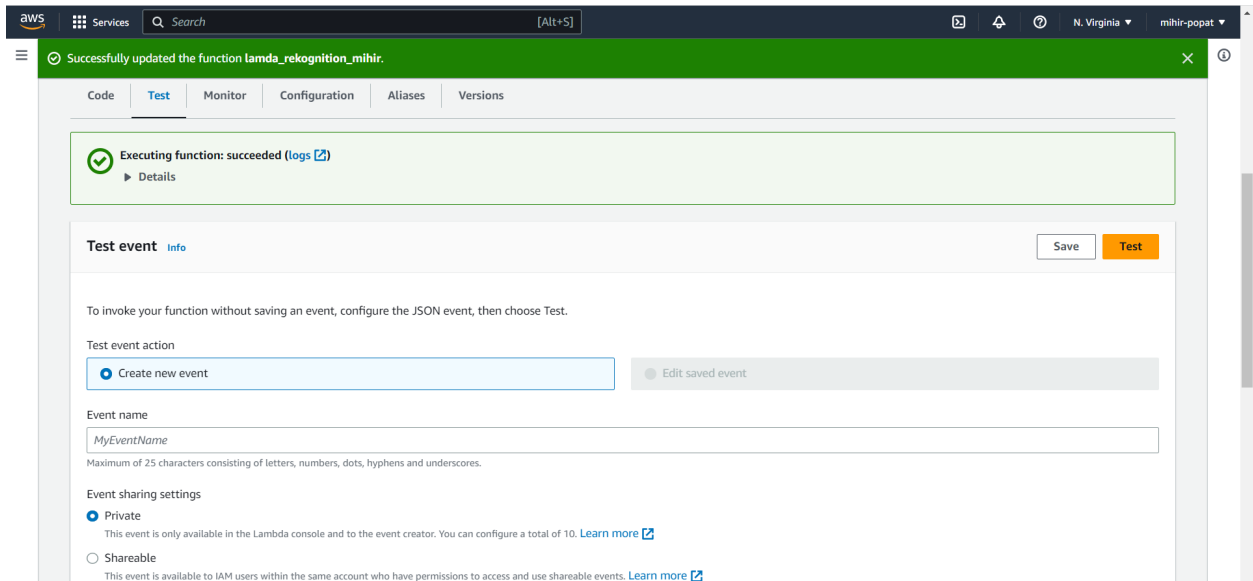
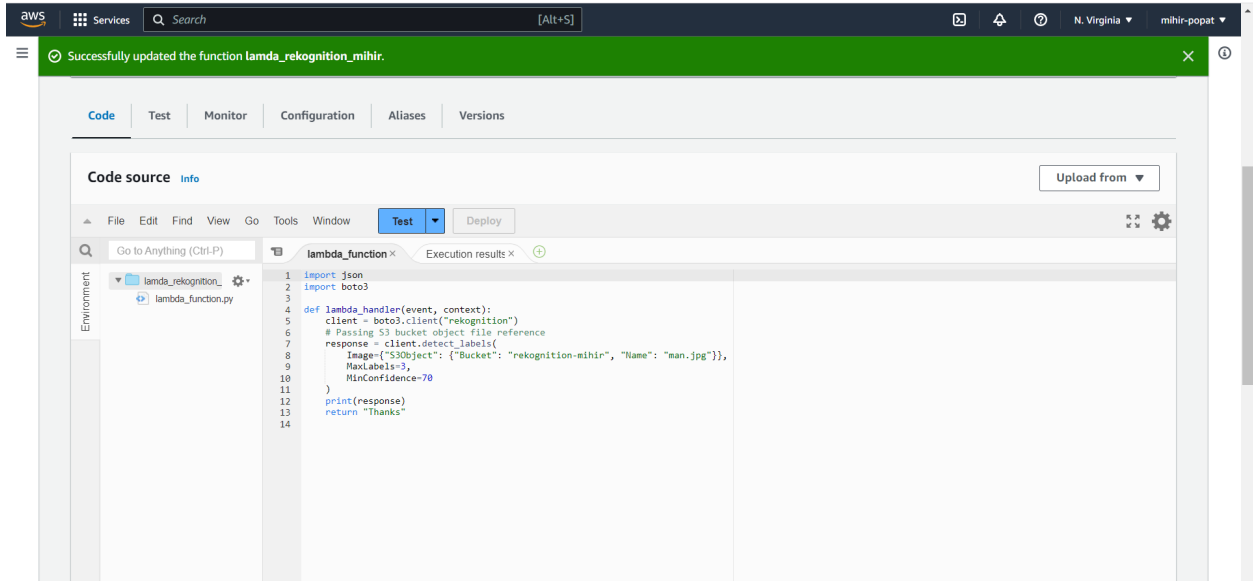
- Expand the Choose or create an execution role.
- In that select Use an existing role. And in the existing role select the role that we created in our first step[I have given the name for a role is lamda\_rekognition].

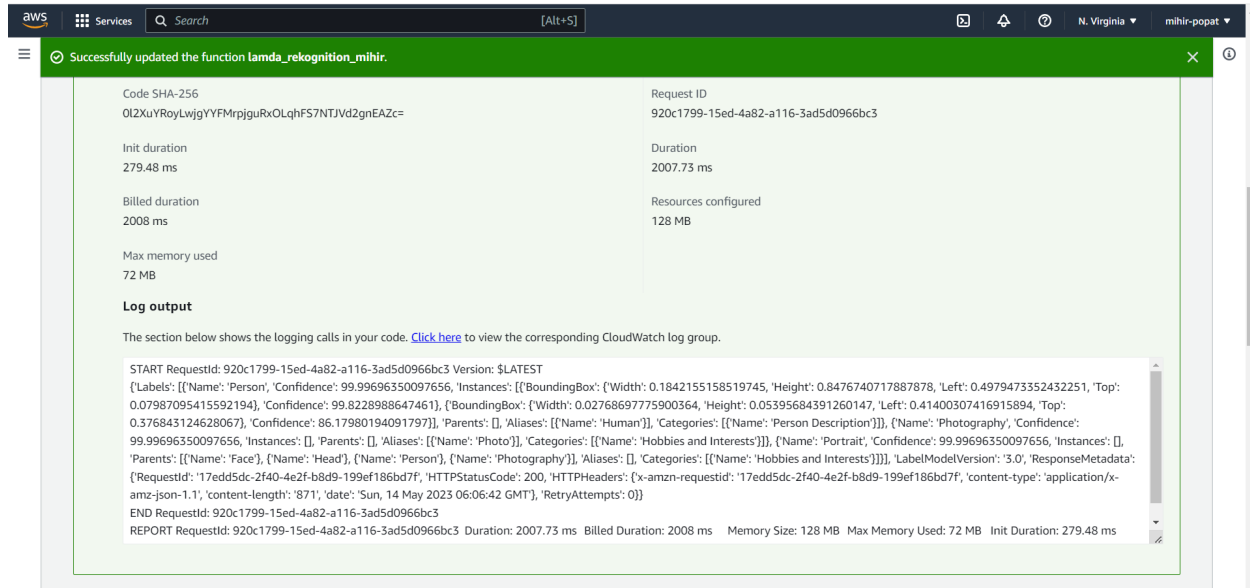


- Finally, click on the create function button.
- Once you created a lambda function then click on your function name.



- In the function, code editor type the function that I have given in the following:
- In the following code, you can directly pass the S3 references in the response using recognition client and you will get a response."MaxLables=3" term is optional using this you can able to see only three labels for the image if we did not mention the name you get more label names for your images





- Note: In place of bucket\_name and image\_name please mention your S3 bucket name and uploaded image name.

```
import json
import boto3
```

```
def lambda_handler(event, context):
    client = boto3.client("rekognition")
    # Passing S3 bucket object file reference
    response = client.detect_labels(
        Image={"S3Object": {"Bucket": "bucket-name", "Name": "image-name"}},
        MaxLabels=3,
        MinConfidence=70
    )
    print(response)
    return "Thanks"
```

- If you want to pass the byte data in the function then also you can pass then prefer the following code.

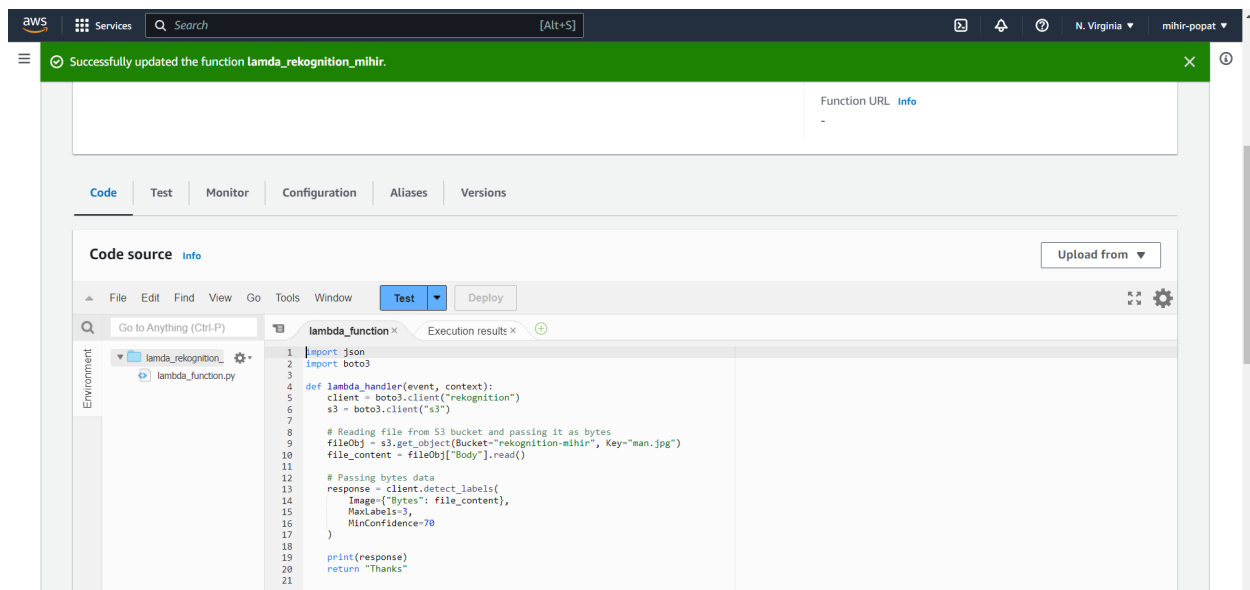
```
import json
import boto3
```

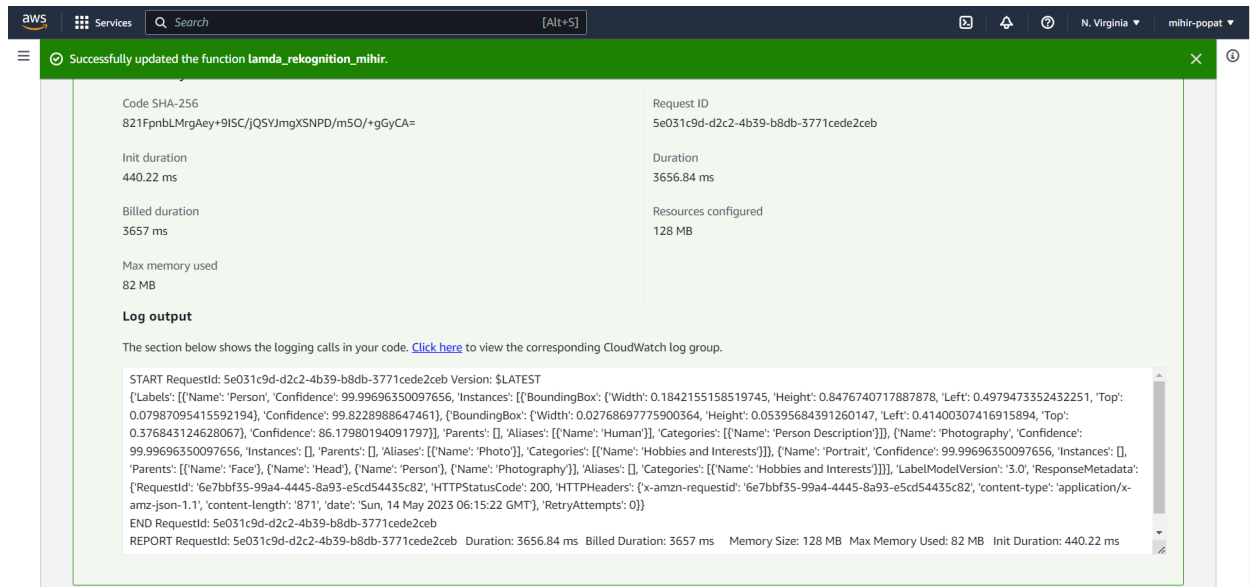
```
def lambda_handler(event, context):
    client = boto3.client("rekognition")
    s3 = boto3.client("s3")
```

```
# Reading file from S3 bucket and passing it as bytes
fileObj = s3.get_object(Bucket="bucket_name", Key="image_name")
file_content = fileObj["Body"].read()

# Passing bytes data
response = client.detect_labels(
    Image={"Bytes": file_content},
    MaxLabels=3,
    MinConfidence=70
)

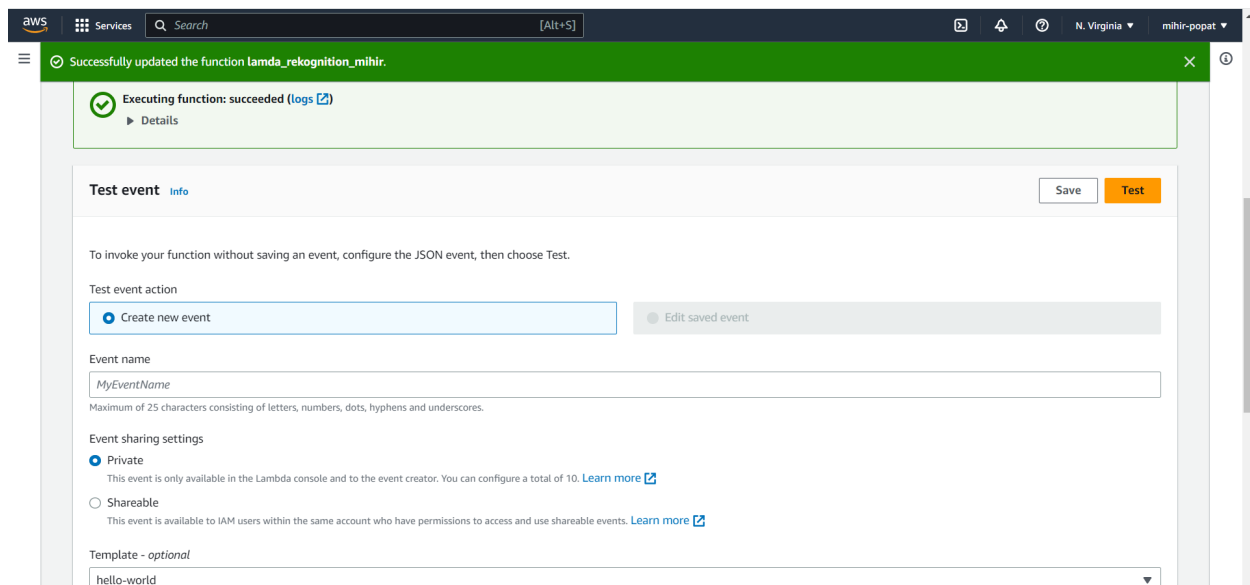
print(response)
return "Thanks"
```





You can add any one of the codes that I have given above and you will get the same response. I have just shown you two different kinds of code for the lambda function.

- After adding the code click on the save button.
- To test the function for image analysis. Click on the Test button given on the upper right side. Once you click on the test button it will pop up one window.
- In that select Create new test event and in Event Template give any name that you want.



- And finally, click on the Create button.
- Once you create the test event it will show you the test event name. So now click on the Test button.
- Once you click on the test the response will look like below:



