

## ASSIGNMENT 6

AIM : To learn how to use Lamda in order to find the ContentType of Object uploaded in S3 Bucket.

### THEORY :

Create bucket:

The screenshot shows the AWS S3 console 'Create bucket' page. The browser address bar shows 's3.console.aws.amazon.com/s3/bucket/create?region=ap-south-1'. The page title is 'Create bucket' with an 'info' icon. Below the title is a sub-header 'Buckets are containers for data stored in S3. [Learn more](#)'. The main content area is divided into three sections: 'General configuration', 'Object Ownership', and 'Block Public Access settings for this bucket'. In the 'General configuration' section, the 'Bucket name' field contains 'mynewbucket' and the 'AWS Region' dropdown is set to 'Asia Pacific (Mumbai) ap-south-1'. There is a 'Choose bucket' button. The 'Object Ownership' section has two radio buttons: 'ACLs disabled (recommended)' (selected) and 'ACLs enabled'. The 'Block Public Access settings for this bucket' section has a heading and a paragraph of text.

create a new policy from iam dashboard;

while creating policy select json tab and paste the following code:

```
{  
  "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:::*/*"

}

]

}

```

The screenshot shows the AWS IAM console interface. The left sidebar contains the 'Identity and Access Management (IAM)' section with a search bar and a navigation menu. The main content area is titled 'Policies (1127)' and includes a search bar and a table of policies.

Policy name	Type	Used as	Description
AWSLambdaBasicExecutionRole-6939cbef-b679-49f4-9759-3e9eb1d3208c	Customer managed	Permissions policy (1)	
AWSLambdaBasicExecutionRole-bc3f17ef-7f7c5-4a0d-9852-4644e8dec70e	Customer managed	Permissions policy (1)	
AWSLambdaBasicExecutionRole-bf11a1ec-26a7-4e97-8e53-6e5c00213d96	Customer managed	Permissions policy (1)	
AWSLambdaS3ExecutionRole-77f8c449-cea9-4eaa-866a-ea6921b47727	Customer managed	Permissions policy (1)	
AWSLambdaS3ExecutionRole-c802badc-2a97-41c2-926f-e1115a9c0f6e	Customer managed	Permissions policy (1)	
AWSLambdaS3ExecutionRole-fedc33a5-6f18-425d-9560-b5c31236c294	Customer managed	Permissions policy (1)	
AdministratorAccess	AWS managed - job function	None	
PowerUserAccess	AWS managed - job function	None	
ReadOnlyAccess	AWS managed - job function	None	
AWSCloudFormationReadOnlyAccess	AWS managed	None	
CloudFrontFullAccess	AWS managed	None	
AWSCloudHSMFullAccess	AWS managed	None	
AWSCloudHSMReadOnlyAccess	AWS managed	None	
ResourceGroupsandTagEditorFullAccess	AWS managed	None	
ResourceGroupsandTagEditorReadOnlyAccess	AWS managed	None	
CloudFrontReadOnlyAccess	AWS managed	None	
CloudSearchFullAccess	AWS managed	None	
CloudSearchReadOnlyAccess	AWS managed	None	

create policy and name it:

The screenshot shows the 'Review and create' page in the AWS IAM console. The 'Policy details' section has a 'Policy name' field containing 's3-trigger-tutorial' and a 'Description' field. Below this, the 'Permissions defined in this policy' section shows a table with two permissions:

Service	Access level	Resource	Request condition
S3	Limited Read	BucketName   string like   All ObjectPath   string like   All	None
CloudWatch Logs	Limited Write	region   string like   All	None

The bottom section is 'Add tags - optional', which is currently empty.

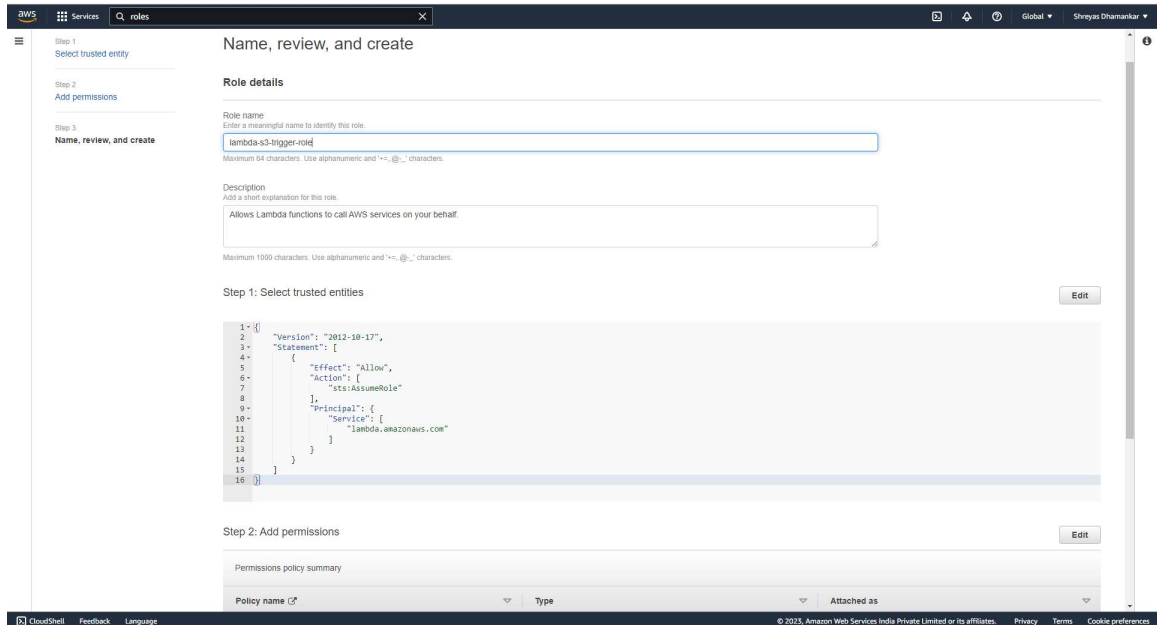
go to roles page and select create a new role

The screenshot shows the 'Roles' page in the AWS IAM console. It displays a list of roles with columns for 'Role name', 'Trusted entities', and 'Last activity'. The roles listed are:

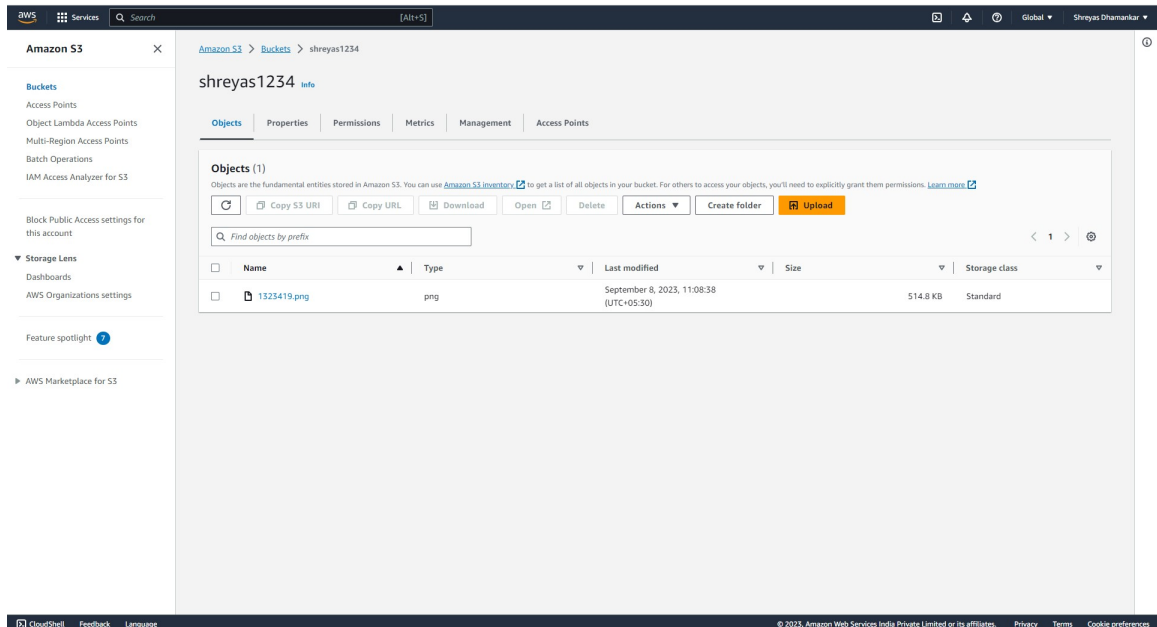
Role name	Trusted entities	Last activity
AWSCloudSSMAccessRole	AWS Service: cloud9 and 1 more	33 days ago
AWSServiceRoleForAWSCloud9	AWS Service: cloud9 (Service-Linked Role)	33 days ago
AWSServiceRoleForSupport	AWS Service: support (Service-Linked Role)	-
AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service-Linked Role)	-
shreyas	AWS Service: lambda	36 minutes ago
shreyas1	AWS Service: lambda	32 minutes ago
shreyas2	AWS Service: lambda	27 minutes ago

Below the list, there are three sections: 'Access AWS from your non AWS workloads', 'X.509 Standard', and 'Temporary credentials'. The 'Create role' button is visible in the top right corner.

under policies for role select the policy that you have created and click next. Then name the role as follows:



Upload an image file in the S3 bucket.



Go to lambda dashboard in aws and create a new function named s3-trigger tutorial. select change execution code and choose the option use existing role. Select lambda-s3-trigger-role.

Once function is created go to code panel and paste the following code.

```
import json
```

```
import urllib.parse
```

```

import boto3

print('Loading function')

s3 = boto3.client('s3')

def lambda_handler(event, context):

    #print("Received event: " + json.dumps(event, indent=2))

    # Get the object from the event and show its content type

    bucket = event['Records'][0]['s3']['bucket']['name']

    key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'], encoding='utf-8')

    try:

        response = s3.get_object(Bucket=bucket, Key=key)

        print("An object : "+response['ContentType']+" has been added to S3 Bucket")

        print("CONTENT TYPE: " + response['ContentType'])

        return response['ContentType']

    except Exception as e:

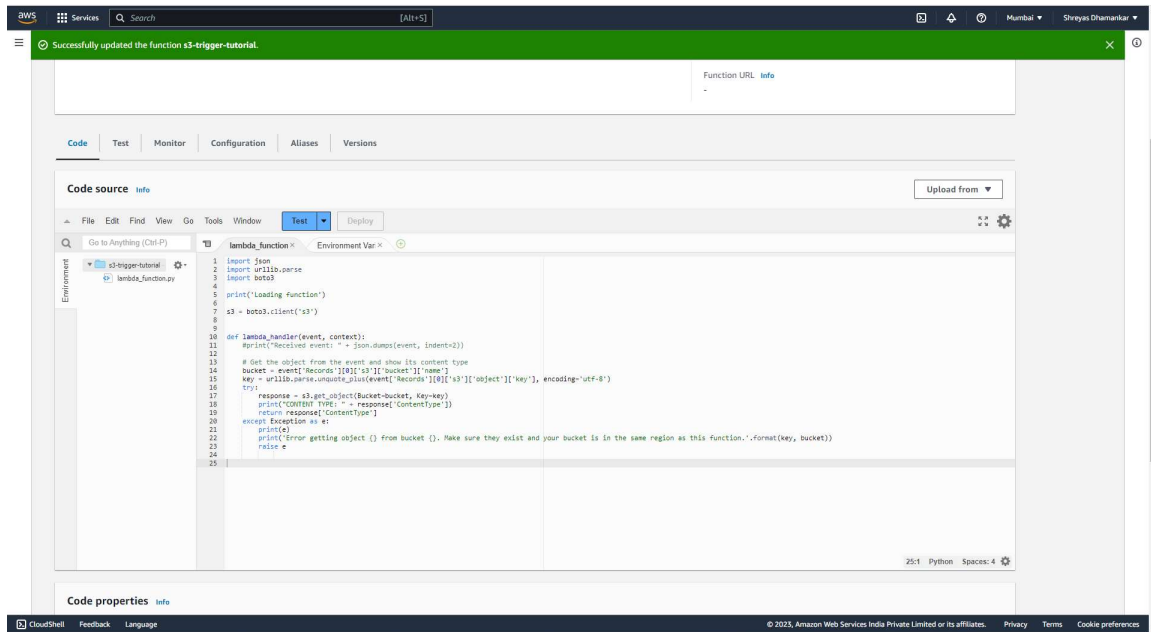
        print(e)

        print('Error getting object {} from bucket {}. Make sure they exist and your bucket is in the
        same region as this function.'.format(key, bucket))

        raise e

then select deploy changes.

```



then click on test and create a new custom event named MyTestEvent.

For Template, choose S3 Put.

In the Event JSON, replace the following values:

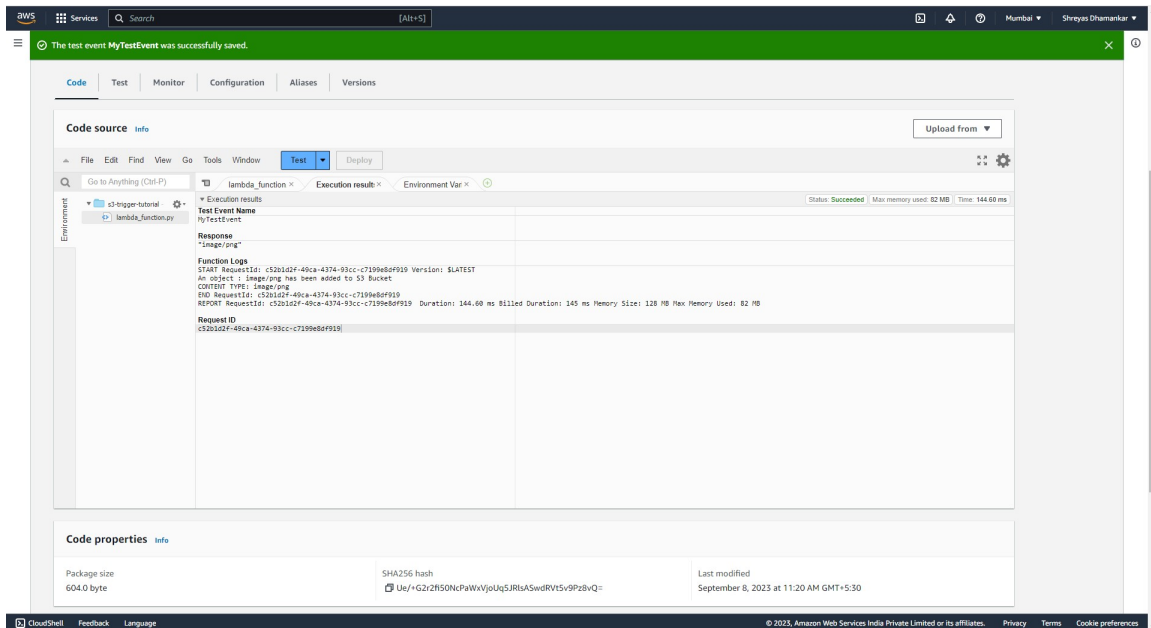
Replace us-east-1 with the region you created your Amazon S3 bucket in.

Replace both instances of example-bucket with the name of your own Amazon S3 bucket.

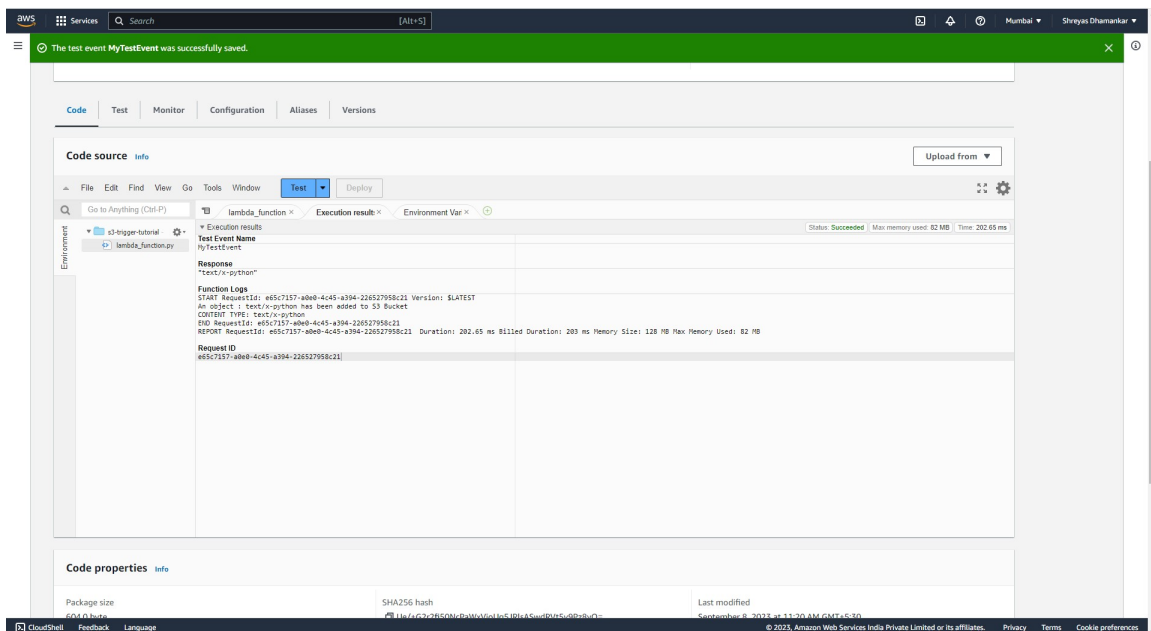
Replace test%2Fkey with the name of the test object you uploaded to your bucket earlier (for example, factorial.py).

Click on save and press Test. You will see the results in Execution tab.

Check execution Tab for the CONTENT TYPE. If you have uploaded the file properly you will get the file type that you have uploaded in the S3 Bucket. Check Function logs where the line has been printed that "An Image has been Uploaded to S3 Bucket".



Next I have uploaded a python file too and this is the output we get in the execution tab.



**CONCLUSION :** In this assignment we learnt how to use Lambda function to log what

type of object has been uploaded to S3 Bucket.