

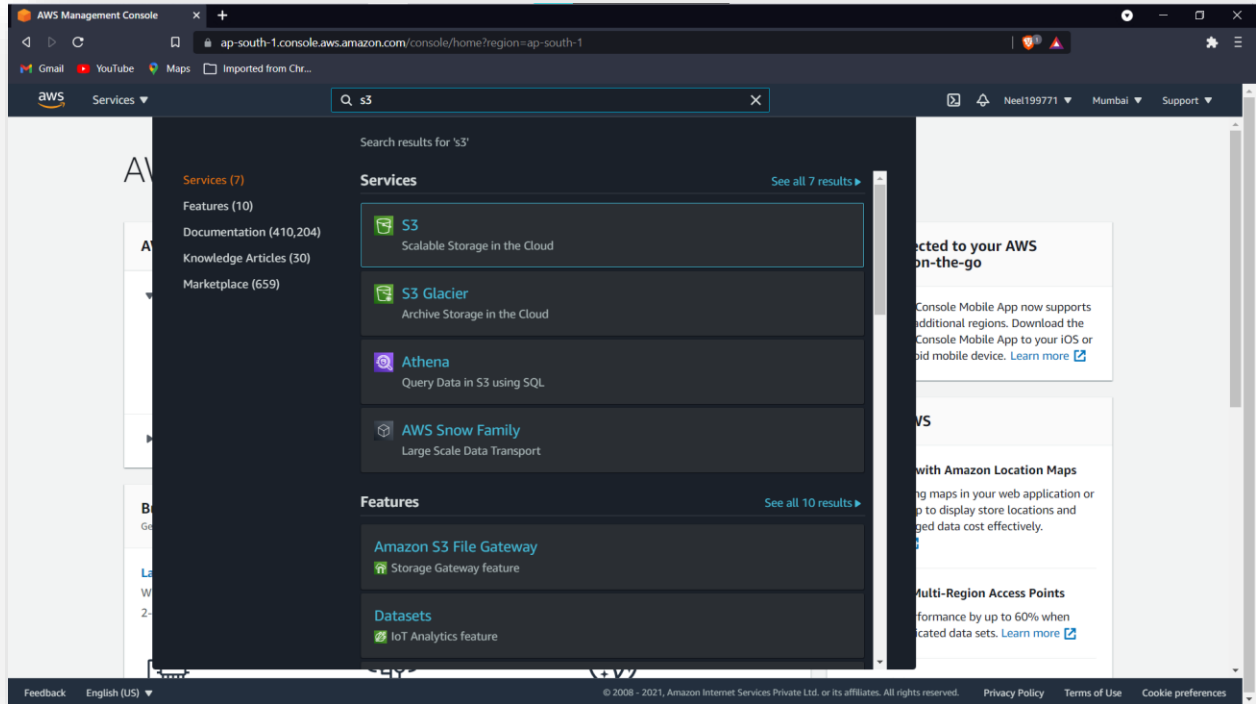
AWS Lambda Function

Lab-5

Aim- To implement lamda Function.

Login to Aws account-

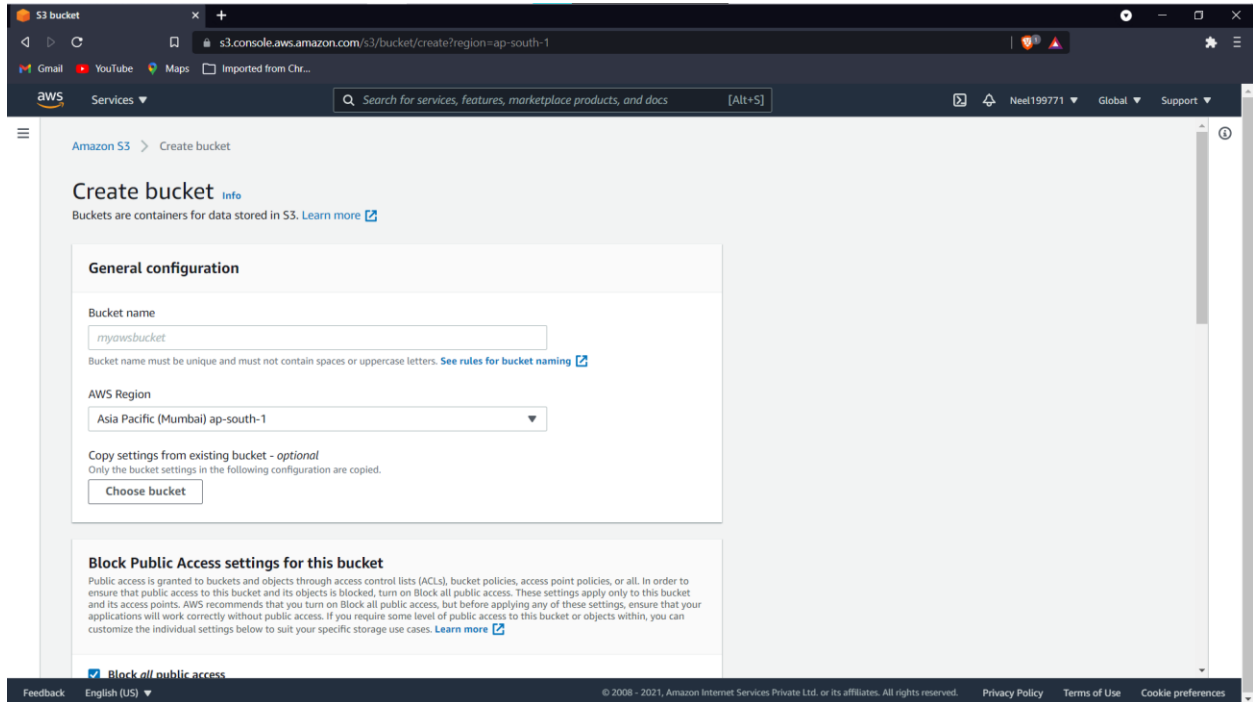
Search S3 ,click on the option below shown-



AWS Lambda Function

Lab-5

Create an S3 bucket by giving it a name



S3 bucket

s3.console.aws.amazon.com/s3/bucket/create?region=ap-south-1

Amazon S3 > Create bucket

Create bucket [Info](#)

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

General configuration

Bucket name

myawsbucket

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

AWS Region

Asia Pacific (Mumbai) ap-south-1

Copy settings from existing bucket - optional

Only the bucket settings in the following configuration are copied.

[Choose bucket](#)

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

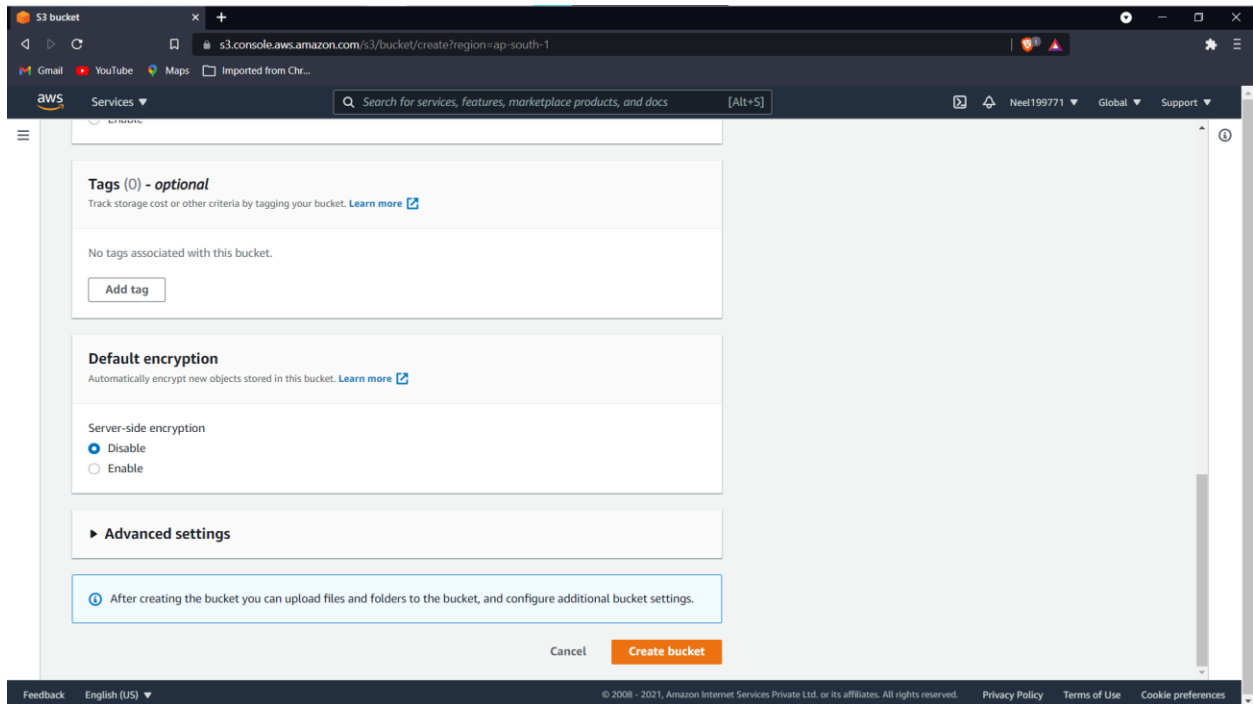
☒ **Block all public access**

Feedback English (US)

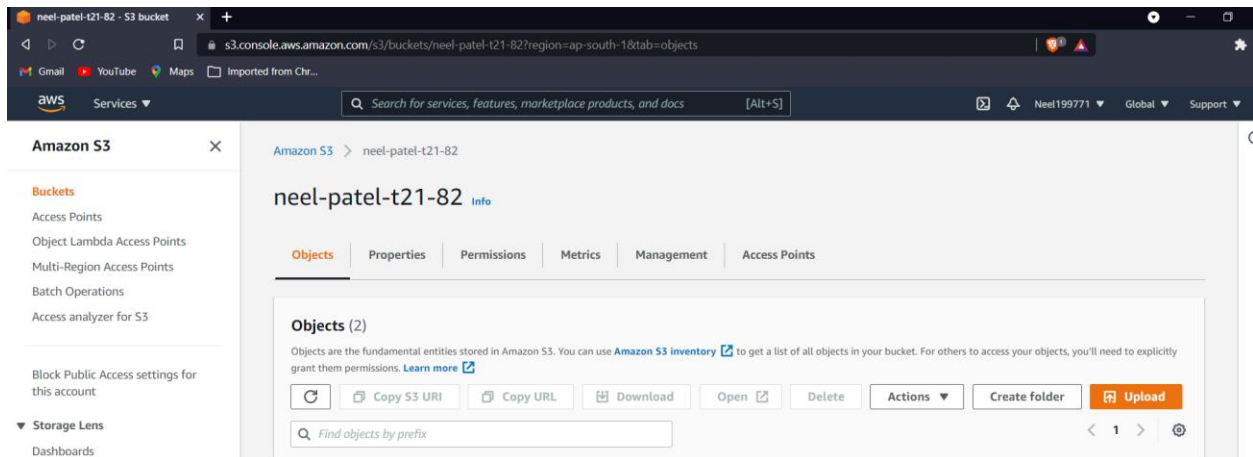
© 2008 - 2021, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#) [Cookie preferences](#)

AWS Lambda Function

Lab-5

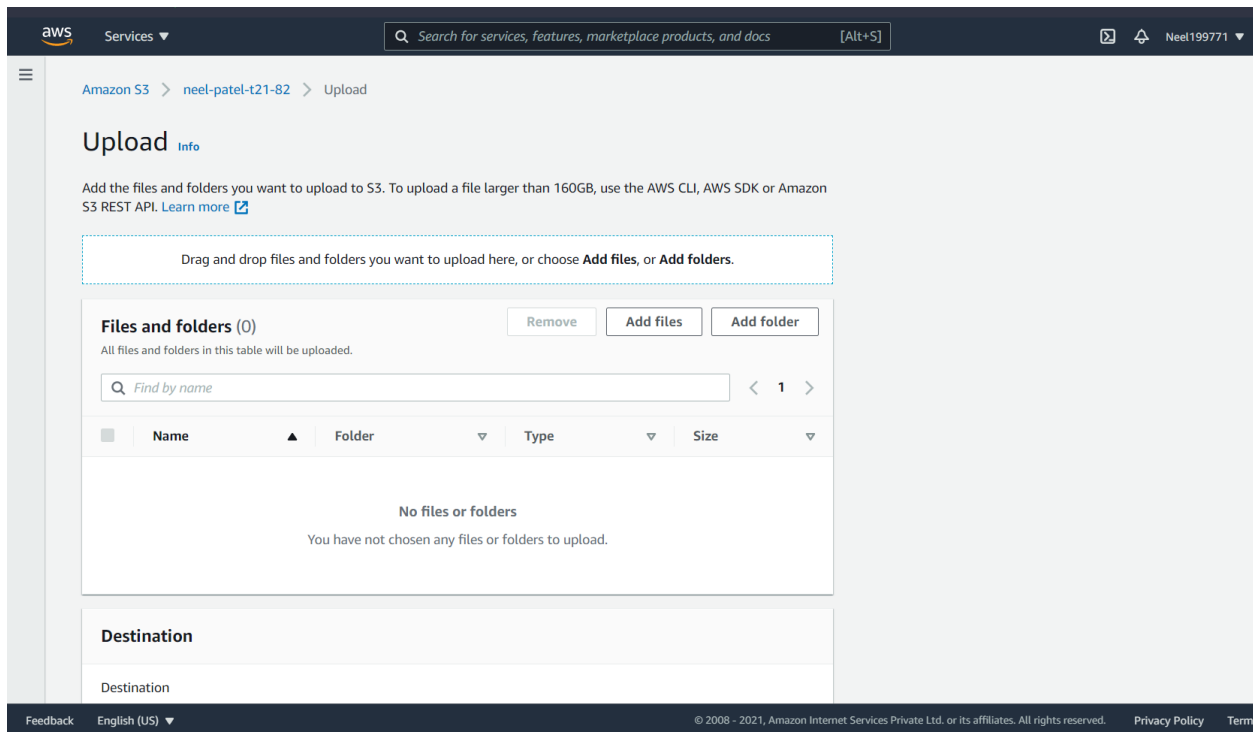


Click on upload button after the s3 bucket is created in the object section



AWS Lambda Function

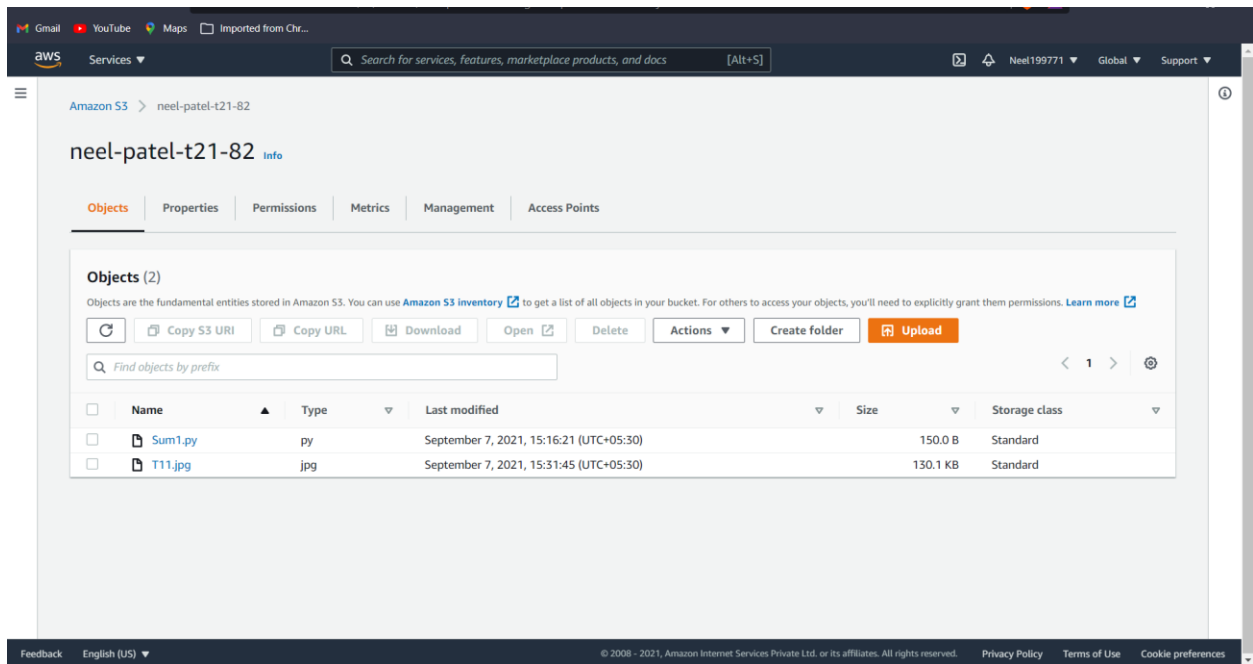
Lab-5



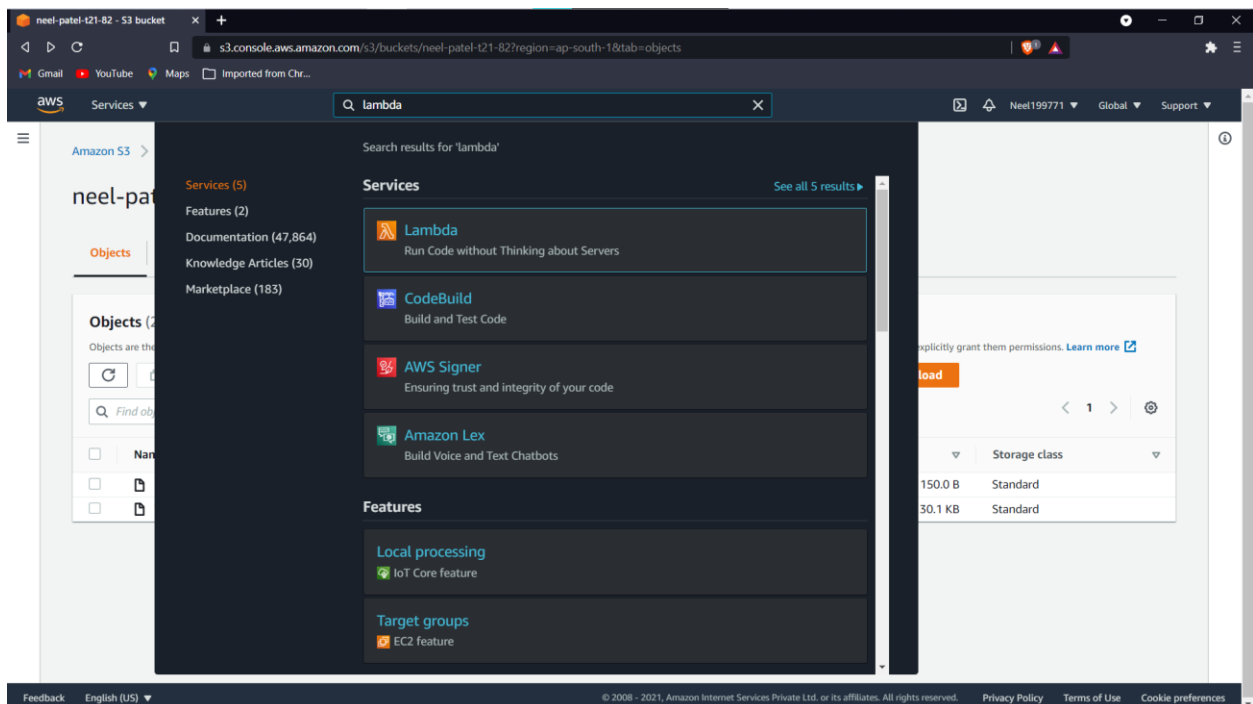
Add any .py or .java extension file and click on upload

AWS Lambda Function

Lab-5



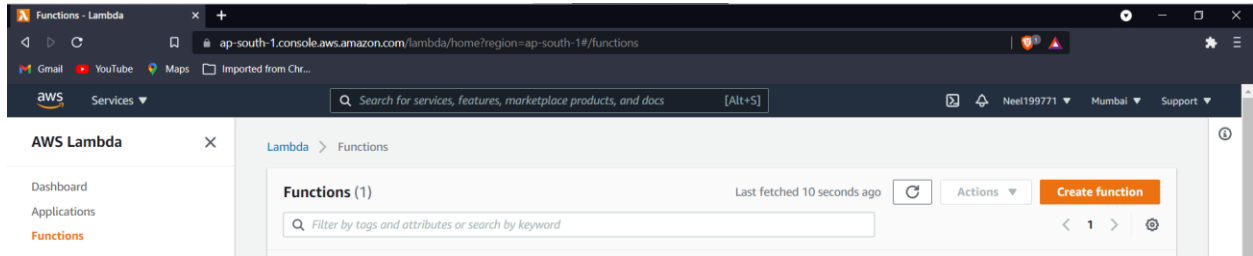
Now search Lamda



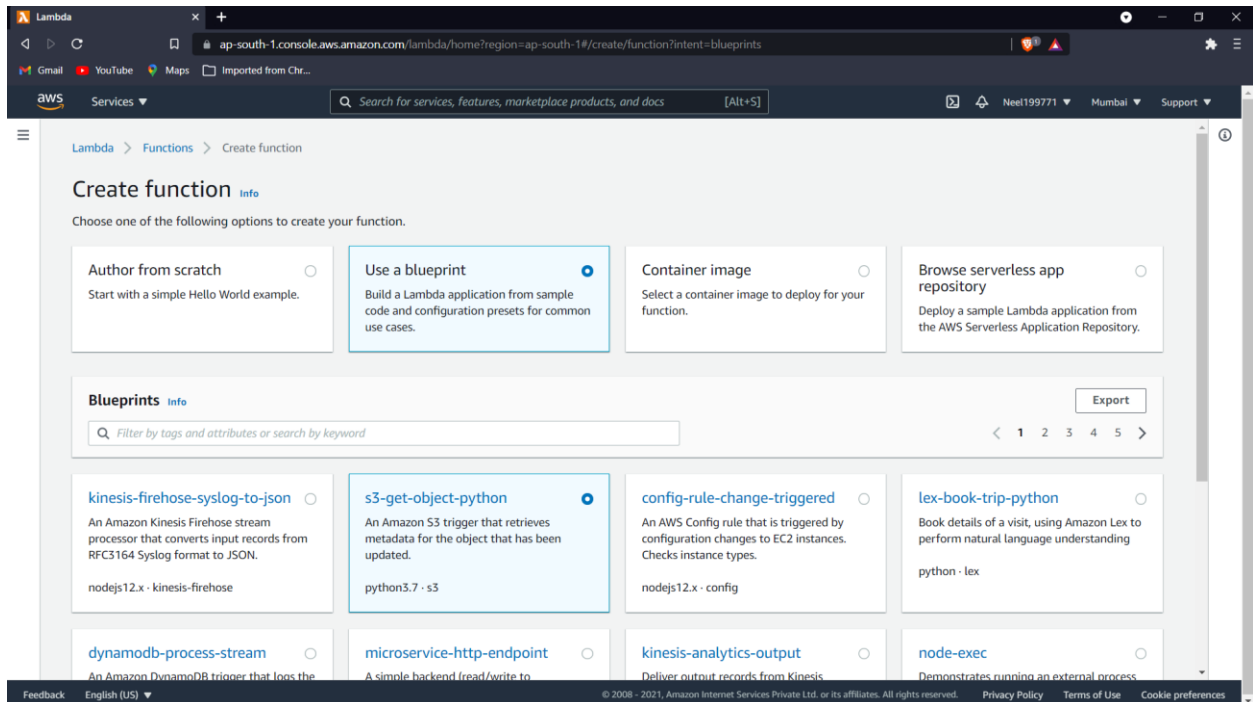
AWS Lambda Function

Lab-5

Click create function



Click on below options and click on configure



AWS Lambda Function

Lab-5

The screenshot shows the AWS Lambda console in the 'Create function' wizard, specifically the 'Configure blueprint' step for the 's3-get-object-python' blueprint. The 'Function name' field is filled with 'Neelt21'. Under 'Execution role', the option 'Create a new role from AWS policy templates' is selected. A message box 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 'Role name' field contains 'myRoleName'. Under 'Policy templates - optional', the 'Amazon S3 object read-only permissions' template is selected. The breadcrumb trail at the top reads: 'Lambda > Functions > Create function > Configure blueprint s3-get-object-python'.

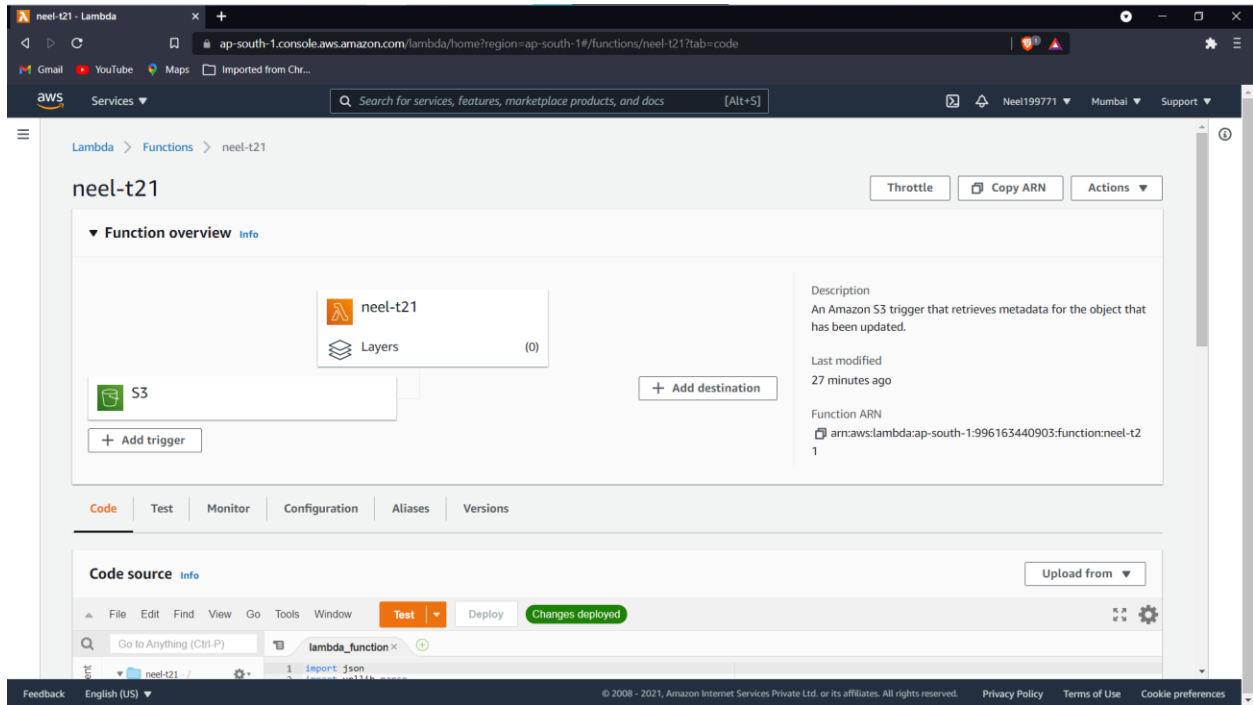
Select the bucket created and create trigger ,click on create function-

The screenshot shows the 'S3 trigger' configuration step in the AWS Lambda console. The 'Bucket' dropdown is set to 'neel-patel-t21-82'. The 'Event type' dropdown is set to 'All object create events'. There are optional fields for 'Prefix' (containing 'e.g. images/') and 'Suffix' (containing 'e.g. .jpg'). A note at the bottom states: 'Lambda will add the necessary permissions for Amazon S3 to invoke your Lambda function from this trigger. [Learn more](#) about the Lambda permissions model.' Below this is the 'Lambda function code' section, which notes that the code is preconfigured by the chosen blueprint. The breadcrumb trail at the top reads: 'Lambda > Functions > Create function > S3 trigger'.

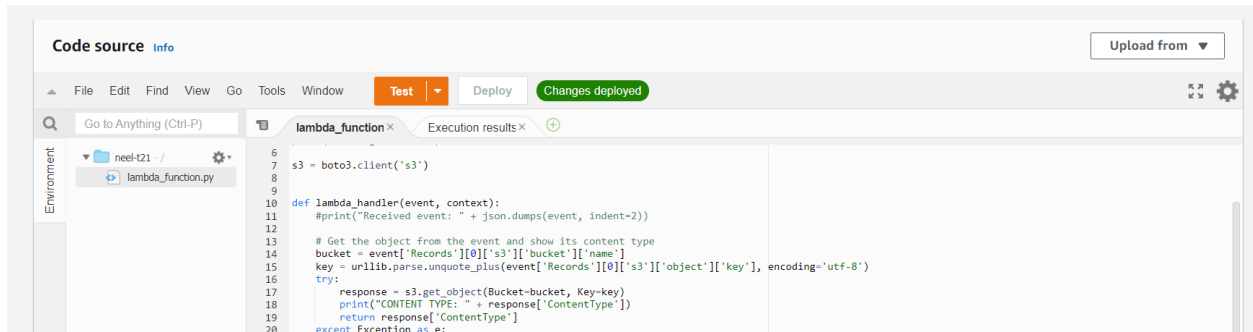
AWS Lambda Function

Lab-5

Check the given trigger is created



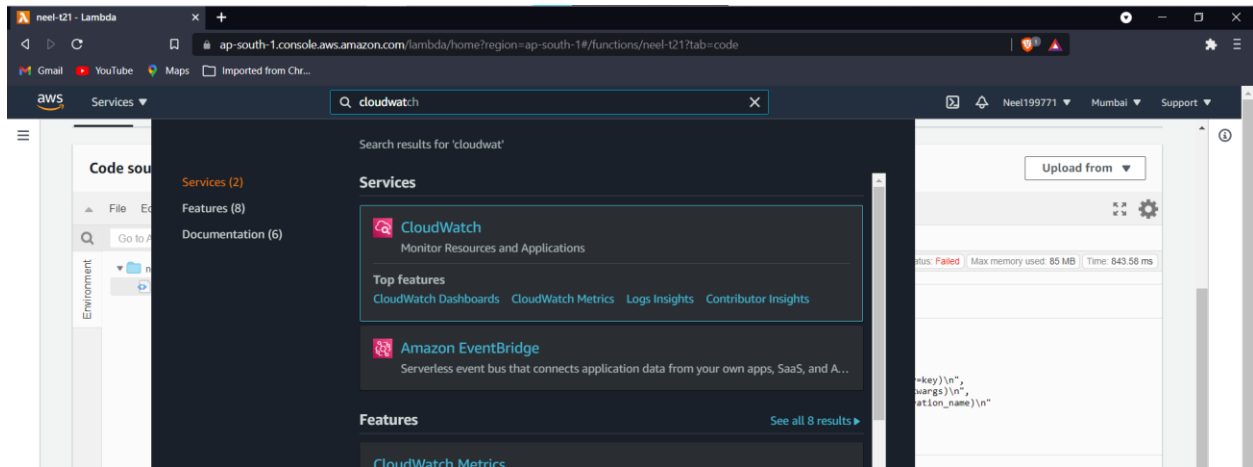
Click on the orange test button-



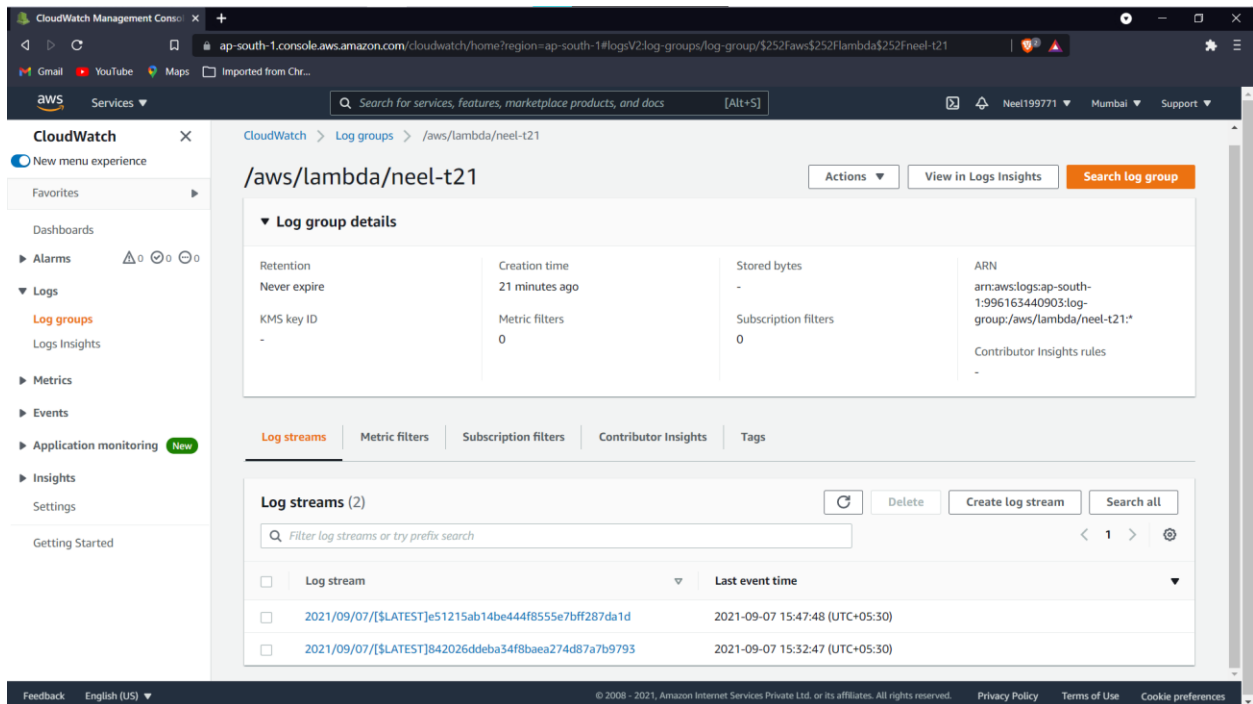
Now,

AWS Lambda Function

Lab-5



Check the logs of the test-

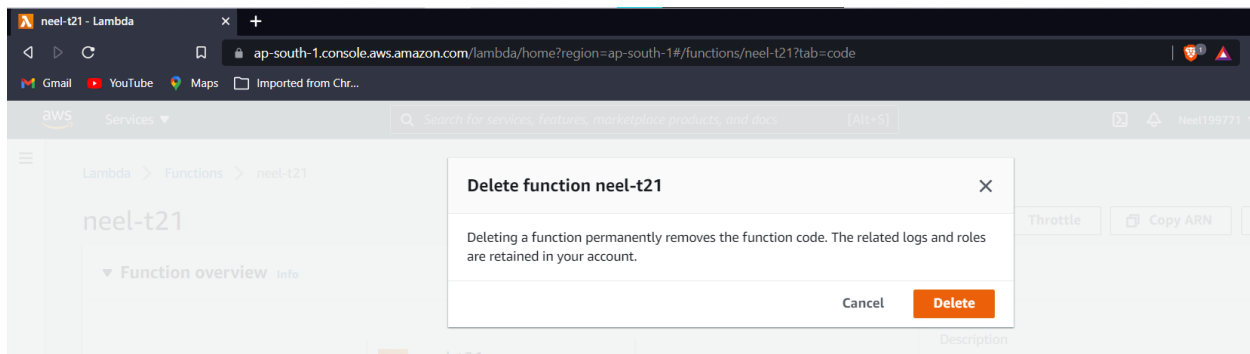
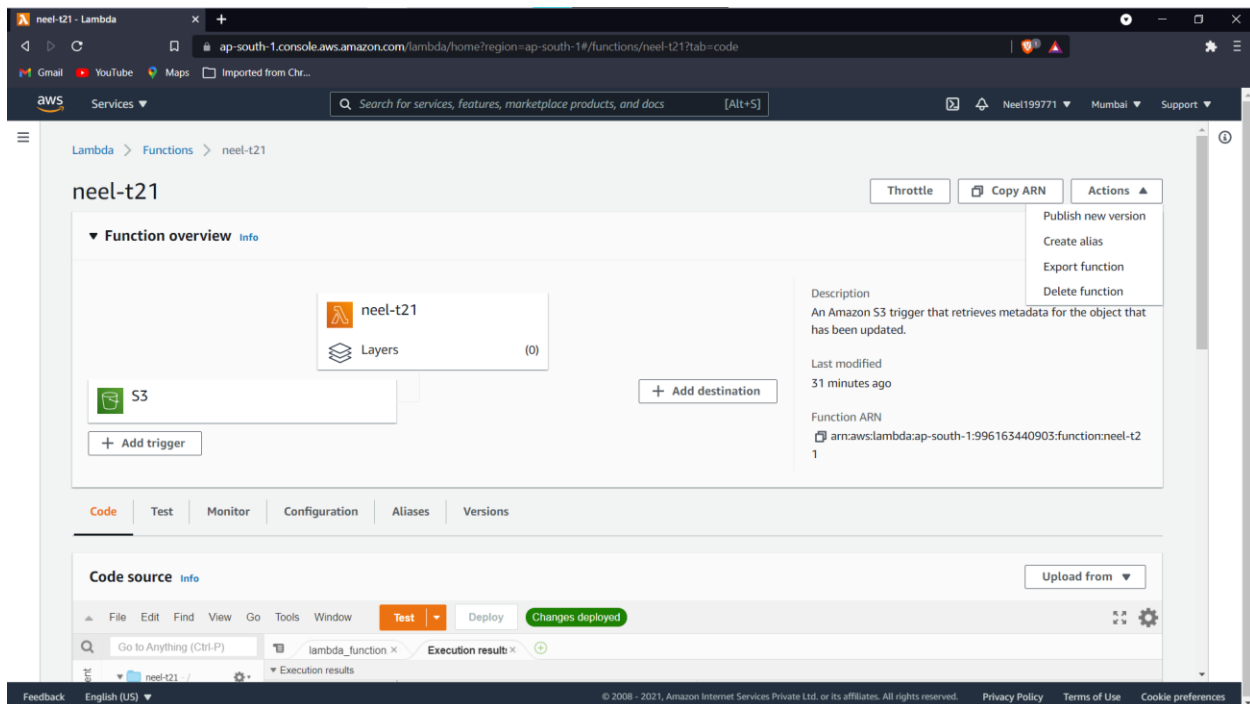


Now terminate-

Click on delete function.

AWS Lambda Function

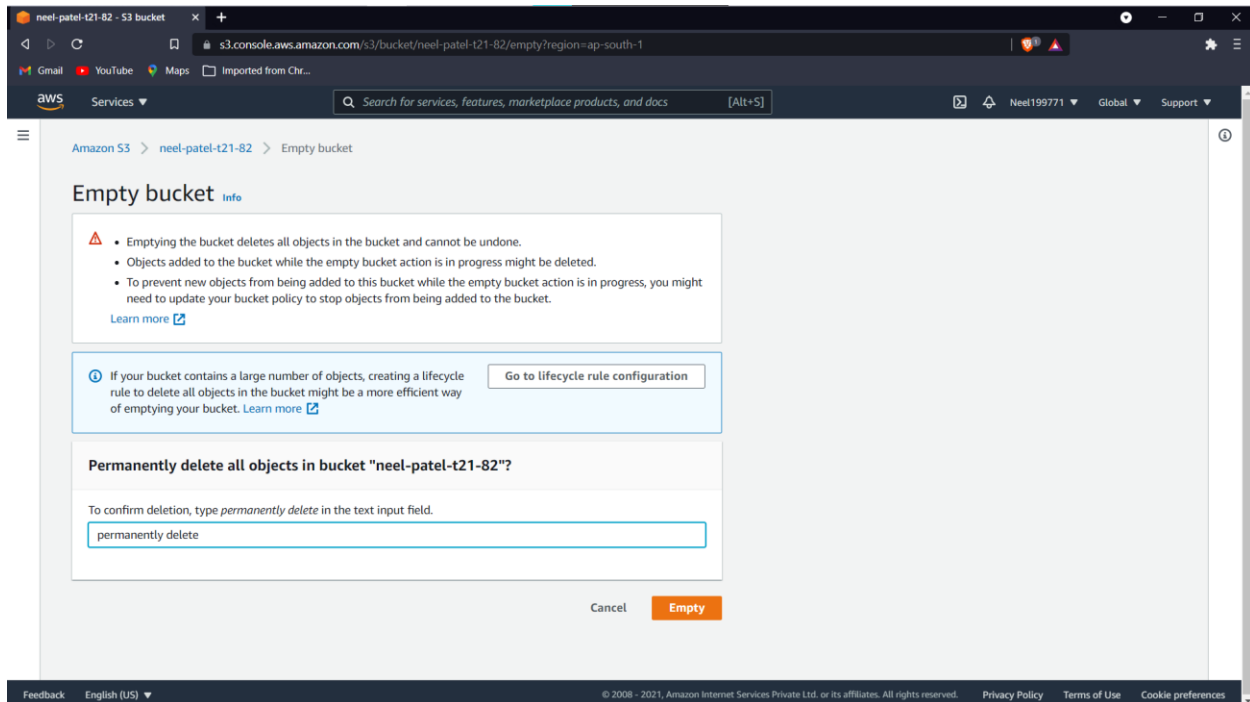
Lab-5



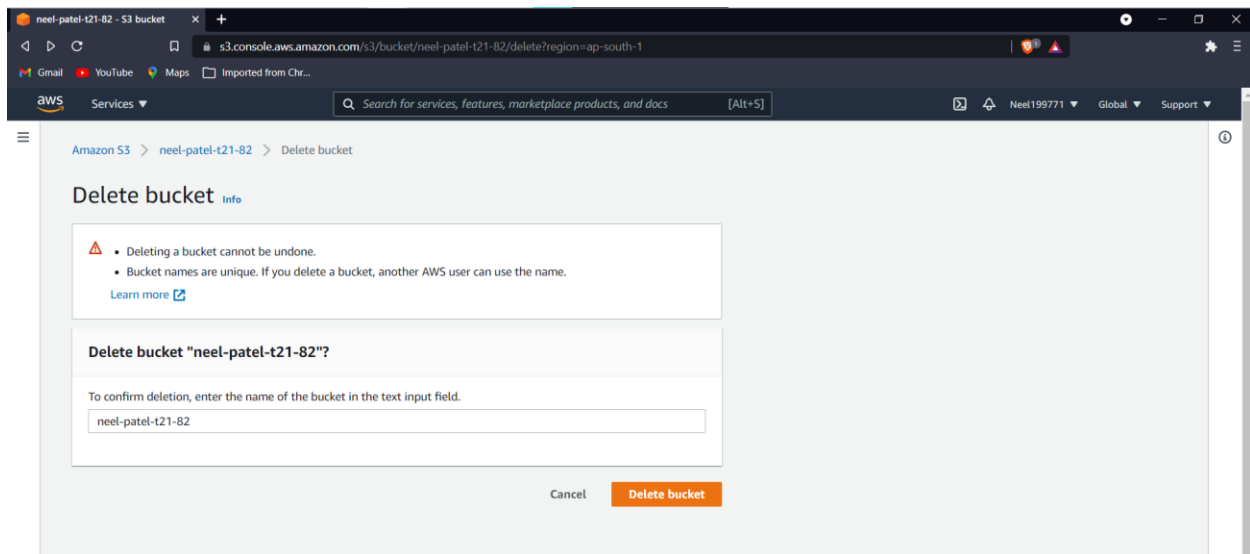
Empty bucket

AWS Lambda Function

Lab-5



Delete bucket-



Step 1: Create a Bucket

- In AWS S3, create a new bucket where you will upload objects.

Step 2: Create a New Policy

AWS Lambda Function

Lab-5

- Go to the IAM (Identity and Access Management) dashboard in AWS.
- Create a new policy.
- When creating the policy, select the JSON tab and paste the provided JSON code. This policy grants necessary permissions for your Lambda function to interact with S3 and CloudWatch Logs.

Step 3: Create a Policy Name

- Name your newly created policy.

Step 4: Create a New Role

- Go to the "Roles" page in AWS IAM.
- Select "Create a new role."
- Under "Policies for Role," select the policy you created earlier.
- Name the role.

Step 5: Upload an Image File

- Upload an image file to the S3 bucket you created in step 1.

Step 6: Create a Lambda Function

- Go to the Lambda dashboard in AWS.
- Create a new function and name it "s3-trigger tutorial."
- Choose the option to "Use an existing role" and select the Lambda execution role you created earlier.
- In the code panel, paste the provided Python code. This code is the Lambda function that will check the content type of the object uploaded to the S3 bucket.

Step 7: Test the Lambda Function

- Create a custom test event named "MyTestEvent."
- For the template, choose "S3 Put."
- In the event JSON, replace values as mentioned in the instructions (region, bucket name, and object key).
- Save the test event and press "Test." The Lambda function will execute, and you'll see the results in the Execution tab.

AWS Lambda Function

Lab-5

Step 8: Check Execution Tab

- In the Execution tab, you can check the "CONTENT TYPE" to see the content type of the uploaded object.
- The function logs should also contain a message like "An object has been added to the S3 Bucket."

```
{
  "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:::*/*"
    }
  ]
}
```

```
import json
import urllib.parse
import boto3
```

AWS Lambda Function

Lab-5

```
print('Loading function')
```

```
s3 = boto3.client('s3')
```

```
def lambda_handler(event, context):
```

```
    # 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 the 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
```