* **Lambda Code for resizing the image**

import boto3

from PIL import Image

import io

s3\_client = boto3.client('s3')

def lambda\_handler(event, context):

for record in event['Records']:

bucket = record['s3']['bucket']['name']

key = record['s3']['object']['key']

response = s3\_client.get\_object(Bucket=bucket, Key=key)

content = response['Body'].read()

image = Image.open(io.BytesIO(content))

image.thumbnail((480, 640))

buffer = io.BytesIO()

image.save(buffer, format='JPEG')

buffer.seek(0)

s3\_client.put\_object(Bucket='shreyas002', Key=key, Body=buffer.getvalue())

**Explanation of lambda code -**

**1. import boto3:** This line imports the Boto3 library, which is a software development kit (SDK) provided by Amazon Web Services (AWS) for building applications on top of their services.

**2. from PIL import Image:** This line imports the Image module from the Python Imaging Library (PIL). The Image module provides a way to represent and manipulate images in Python.

**3. import io:** This line imports the io module which provides a way to work with stream-based input/output operations.

**4. s3\_client = boto3.client('s3'):** This line creates an S3 client object which is used to communicate with the Amazon S3 service. The client is created by calling the boto3.client method and passing in the name of the service, 's3', as an argument.

**5. def lambda\_handler(event, context):** This line defines a function called lambda\_handler that takes two arguments, event and context. This is the handler function for an AWS Lambda function, which is a serverless compute service provided by AWS.

**6. for record in event['Records']:** This line sets up a for loop that iterates over the list of S3 event records contained in the Records key of the event dictionary.

**7. bucket = record['s3']['bucket']['name']:** This line retrieves the name of the S3 bucket associated with the current event record.

**8. key = record['s3']['object']['key']:** This line retrieves the object key for the S3 object that triggered the event.

**9. response = s3\_client.get\_object(Bucket=bucket, Key=key):** This line retrieves the contents of the S3 object by calling the get\_object method of the S3 client object and passing in the bucket name and object key as arguments.

**10. content = response['Body'].read():** This line reads the contents of the S3 object by accessing the Body key of the response dictionary and calling its read method.

**11. image = Image.open(io.BytesIO(content)):** This line opens the image from the S3 object by calling the open method of the Image module and passing in a BytesIO object that contains the image data.

**12. image.thumbnail((480, 640)):** This line resizes the image to fit within a bounding box of 480x640 pixels by calling the thumbnail method of the Image object and passing in a tuple of the desired dimensions.

**13. buffer = io.BytesIO():** This line creates a new BytesIO object that will be used to store the resized image.

**14. image.save(buffer, format='JPEG'):** This line saves the resized image to the BytesIO object by calling the save method of the Image object and passing in the buffer object and the desired file format (JPEG in this case).

**15. buffer.seek(0):** This line resets the buffer's read/write position to the beginning of the file.

**16. s3\_client.put\_object(Bucket='shreyas002', Key=key, Body=buffer.getvalue()):** This line uploads the resized image to another S3 bucket by calling the put\_object method of the S3 client object and passing in the name of the target bucket, the object key, and the contents of the BytesIO buffer as arguments.