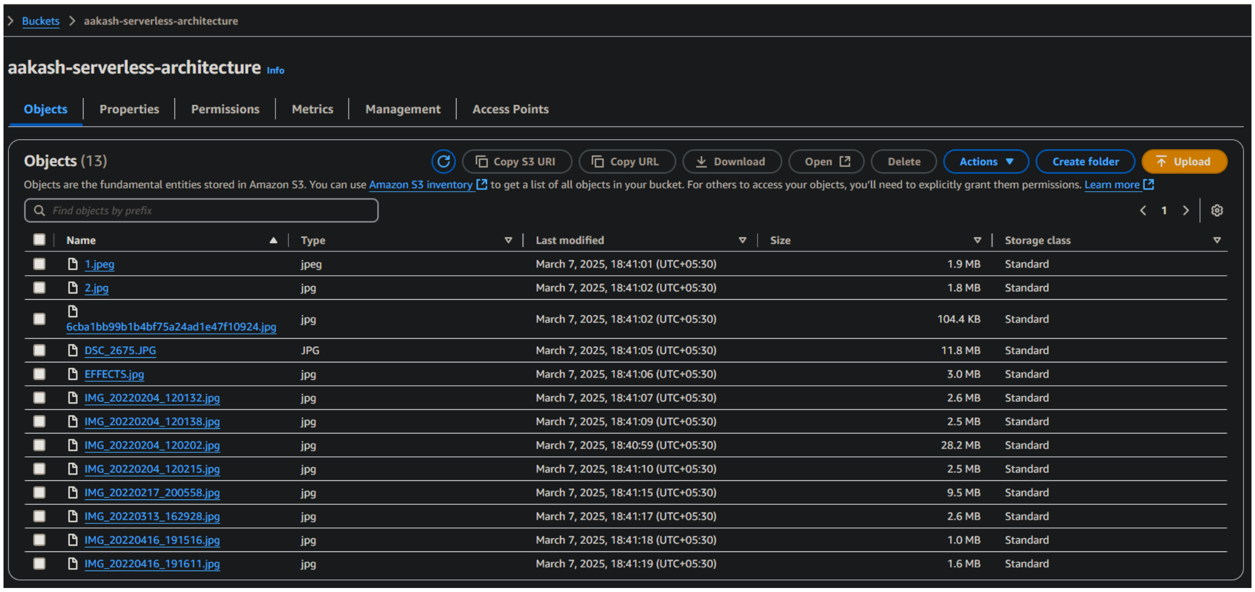
Graded Assignment on Serverless Architecture

Assignment 9: Archive Old Files from S3 to Glacier Using AWS Lambda and Boto3

Objective: Automate the archival of files older than a certain age from an S3 bucket to Amazon Glacier for cost-effective storage.

Task: Automatically move files in an S3 bucket older than 6 months to Glacier storage class.

S3 Setup:


Buckets > aakash-serverless-architecture
Info
Objects | Properties | Permissions | Metrics | Management | Access Points
Copy S3 URI
Copy URL
Download
Open
Delete
Actions ▼
Create folder
Upload
Find objects by prefix
< 1 >

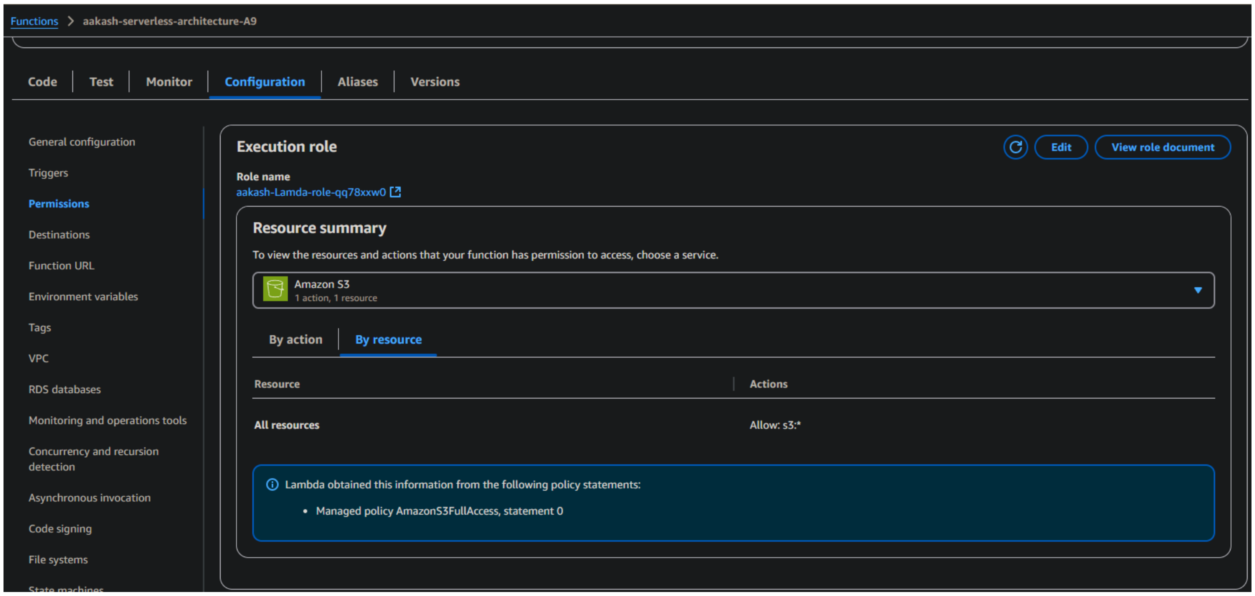
aakash-serverless-architecture

Objects (13)

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

Name	Type	Last modified	Size	Storage class
1.jpeg	jpeg	March 7, 2025, 18:41:01 (UTC+05:30)	1.9 MB	Standard
2.jpg	jpg	March 7, 2025, 18:41:02 (UTC+05:30)	1.8 MB	Standard
6cba1bb99b1b4bf75a24ad1e47f10924.jpg	jpg	March 7, 2025, 18:41:02 (UTC+05:30)	104.4 KB	Standard
DSC_2675.JPG	JPG	March 7, 2025, 18:41:05 (UTC+05:30)	11.8 MB	Standard
EFFECTS.jpg	jpg	March 7, 2025, 18:41:06 (UTC+05:30)	3.0 MB	Standard
IMG_20220204_120132.jpg	jpg	March 7, 2025, 18:41:07 (UTC+05:30)	2.6 MB	Standard
IMG_20220204_120138.jpg	jpg	March 7, 2025, 18:41:09 (UTC+05:30)	2.5 MB	Standard
IMG_20220204_120202.jpg	jpg	March 7, 2025, 18:40:59 (UTC+05:30)	28.2 MB	Standard
IMG_20220204_120215.jpg	jpg	March 7, 2025, 18:41:10 (UTC+05:30)	2.5 MB	Standard
IMG_20220217_200558.jpg	jpg	March 7, 2025, 18:41:15 (UTC+05:30)	9.5 MB	Standard
IMG_20220313_162928.jpg	jpg	March 7, 2025, 18:41:17 (UTC+05:30)	2.6 MB	Standard
IMG_20220416_191516.jpg	jpg	March 7, 2025, 18:41:18 (UTC+05:30)	1.0 MB	Standard
IMG_20220416_191611.jpg	jpg	March 7, 2025, 18:41:19 (UTC+05:30)	1.6 MB	Standard

Lambda IAM Role:



Execution role

Role name
aakash-Lamda-role-qq78xxw0

Resource summary

To view the resources and actions that your function has permission to access, choose a service.

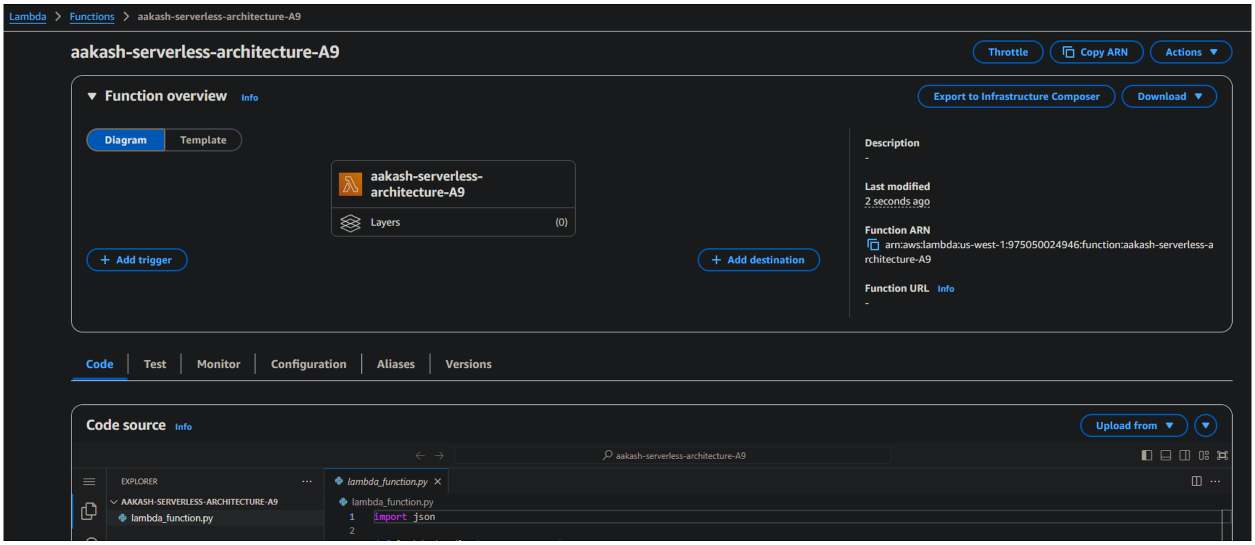
Amazon S3
1 action, 1 resource

Resource	Actions
All resources	Allow: s3:*

Lambda obtained this information from the following policy statements:

- Managed policy AmazonS3FullAccess, statement 0

Lambda Function:



aakash-serverless-architecture-A9

Function overview

aakash-serverless-architecture-A9

Description
-

Last modified
2 seconds ago

Function ARN
arn:aws:lambda:us-west-1:975050024946:function:aakash-serverless-architecture-A9

Function URL
-

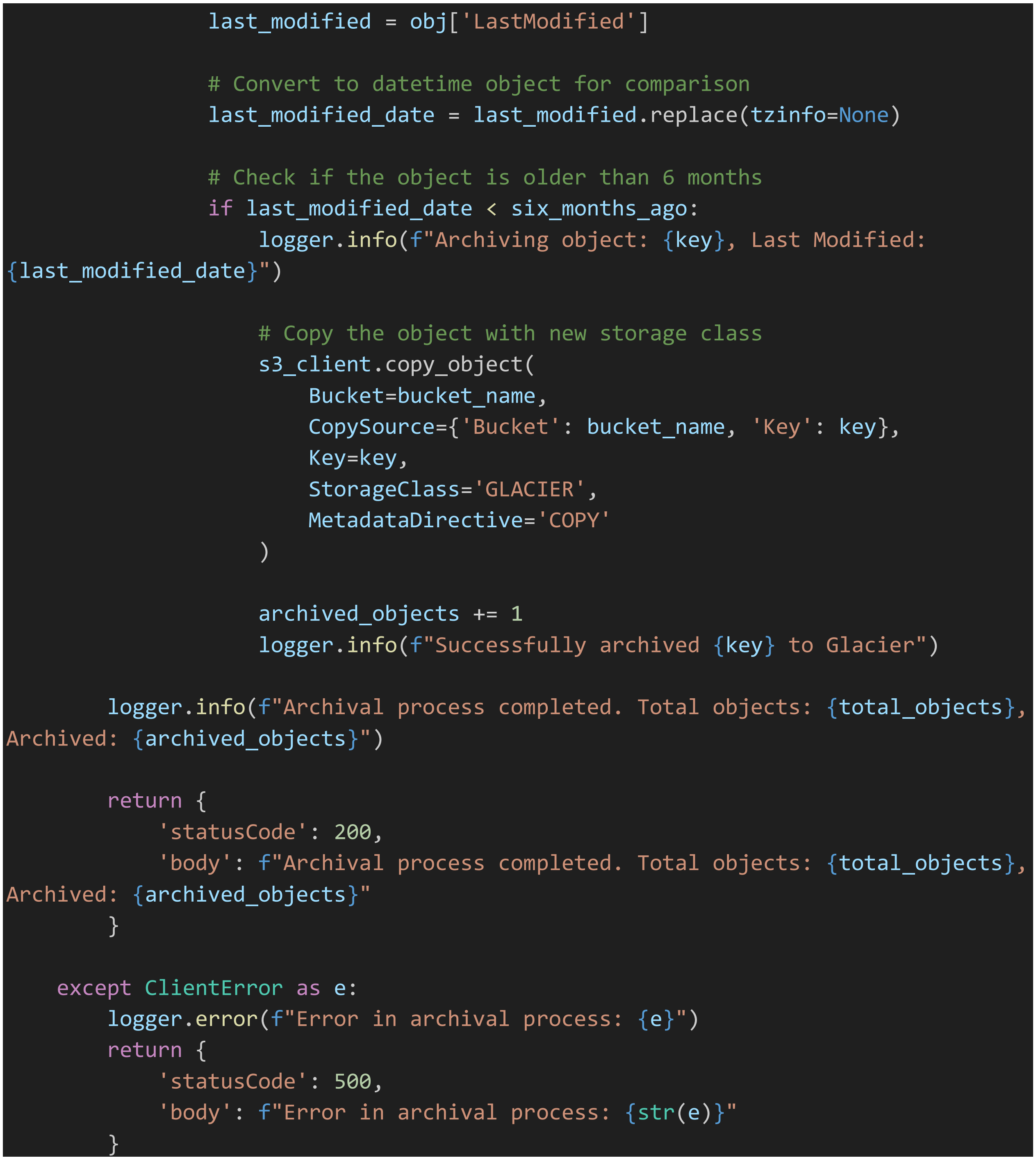
Code source

AAKASH-SERVERLESS-ARCHITECTURE-A9
lambda_function.py

```python
import json

```

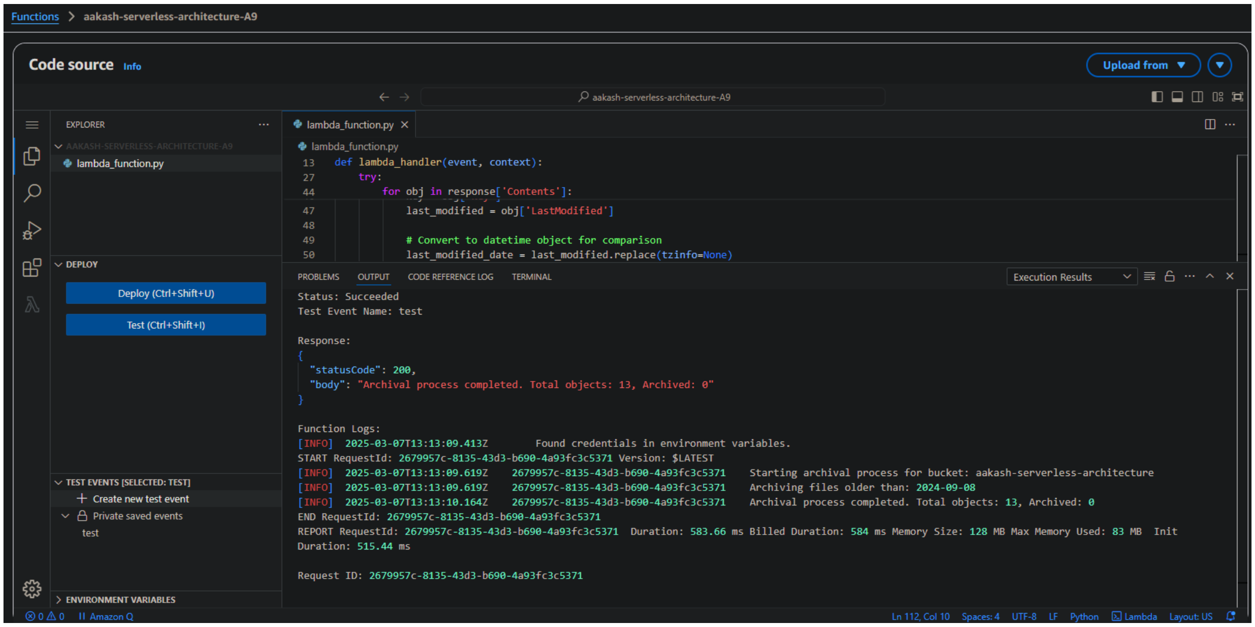
Write the Boto3 Python script:

```python
import boto3
import logging
import datetime
from botocore.exceptions import ClientError

# Configure logging
logger = logging.getLogger()
logger.setLevel(logging.INFO)

# S3 client
s3_client = boto3.client('s3')

def lambda_handler(event, context):
    """
    Lambda function to archive S3 objects older than 6 months to Glacier storage class
    """
    # Set the bucket name - you can also pass this via environment variables
    bucket_name = 'aakash-serverless-architecture'

    # Calculate the date 6 months ago
    today = datetime.datetime.now()
    six_months_ago = today - datetime.timedelta(days=180)

    logger.info(f"Starting archival process for bucket: {bucket_name}")
    logger.info(f"Archiving files older than: {six_months_ago.strftime('%Y-%m-%d')}")

    try:
        # Get list of objects in the bucket
        response = s3_client.list_objects_v2(Bucket=bucket_name)

        # If no objects are found
        if 'Contents' not in response:
            logger.info(f"No objects found in bucket {bucket_name}")
            return {
                'statusCode': 200,
                'body': 'No objects found in bucket'
            }

        # Count statistics
        total_objects = 0
        archived_objects = 0

        # Process each object in the bucket
        for obj in response['Contents']:
            total_objects += 1
            key = obj['Key']
            last_modified = obj['LastModified']

            # Convert to datetime object for comparison
            last_modified_date = last_modified.replace(tzinfo=None)

            # Check if the object is older than 6 months
            if last_modified_date < six_months_ago:
                logger.info(f"Archiving object: {key}, Last Modified:
{last_modified_date}")

                # Copy the object with new storage class
                s3_client.copy_object(
                    Bucket=bucket_name,
                    CopySource={'Bucket': bucket_name, 'Key': key},
                    Key=key,
                    StorageClass='GLACIER',
                    MetadataDirective='COPY'
                )

                archived_objects += 1
                logger.info(f"Successfully archived {key} to Glacier")

        # Process additional pages if the response was truncated
        while response.get('IsTruncated', False):
            continuation_token = response.get('NextContinuationToken')
            response = s3_client.list_objects_v2(
                Bucket=bucket_name,
                ContinuationToken=continuation_token
            )

            for obj in response.get('Contents', []):
                total_objects += 1
                key = obj['Key']
```



```
                last_modified = obj['LastModified']

                # Convert to datetime object for comparison
                last_modified_date = last_modified.replace(tzinfo=None)

                # Check if the object is older than 6 months
                if last_modified_date < six_months_ago:
                    logger.info(f"Archiving object: {key}, Last Modified:
{last_modified_date}")

                    # Copy the object with new storage class
                    s3_client.copy_object(
                        Bucket=bucket_name,
                        CopySource={'Bucket': bucket_name, 'Key': key},
                        Key=key,
                        StorageClass='GLACIER',
                        MetadataDirective='COPY'
                    )

                    archived_objects += 1
                    logger.info(f"Successfully archived {key} to Glacier")

        logger.info(f"Archival process completed. Total objects: {total_objects},
Archived: {archived_objects}")

        return {
            'statusCode': 200,
            'body': f"Archival process completed. Total objects: {total_objects},
Archived: {archived_objects}"
        }

    except ClientError as e:
        logger.error(f"Error in archival process: {e}")
        return {
            'statusCode': 500,
            'body': f"Error in archival process: {str(e)}"
        }
```

Testing:



```python
def lambda_handler(event, context):
    try:
        for obj in response['Contents']:
            last_modified = obj['LastModified']

            # Convert to datetime object for comparison
            last_modified_date = last_modified.replace(tzinfo=None)
```

```
Status: Succeeded
Test Event Name: test

Response:
{
  "statusCode": 200,
  "body": "Archival process completed. Total objects: 13, Archived: 0"
}

Function Logs:
[INFO]  2025-03-07T13:13:09.413Z        Found credentials in environment variables.
START RequestId: 2679957c-8135-43d3-b690-4a93fc3c5371 Version: $LATEST
[INFO]  2025-03-07T13:13:09.619Z    2679957c-8135-43d3-b690-4a93fc3c5371    Starting archival process for bucket: aakash-serverless-architecture
[INFO]  2025-03-07T13:13:09.619Z    2679957c-8135-43d3-b690-4a93fc3c5371    Archiving files older than: 2024-09-08
[INFO]  2025-03-07T13:13:10.164Z    2679957c-8135-43d3-b690-4a93fc3c5371    Archival process completed. Total objects: 13, Archived: 0
END RequestId: 2679957c-8135-43d3-b690-4a93fc3c5371
REPORT RequestId: 2679957c-8135-43d3-b690-4a93fc3c5371  Duration: 583.66 ms Billed Duration: 584 ms Memory Size: 128 MB Max Memory Used: 83 MB  Init
Duration: 515.44 ms

Request ID: 2679957c-8135-43d3-b690-4a93fc3c5371
```