3. Serverless Data Processing Pipeline

Objective: Build a serverless pipeline for processing data (e.g., log processing or ETL jobs).

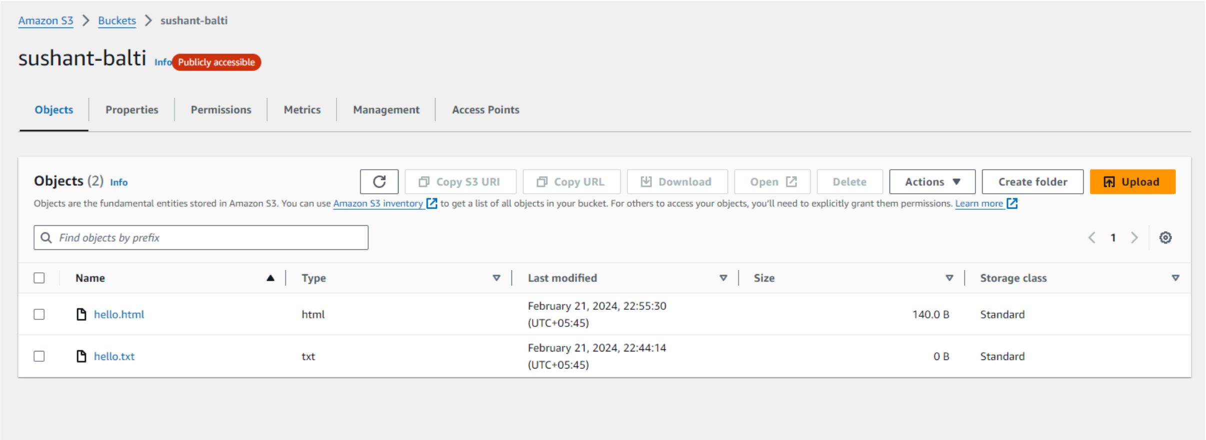
Approach:

- **Data Ingestion**: Use AWS services like S3 or Kinesis to ingest data.
- **Processing**: Create Lambda functions to process the ingested data.
- **Storage**: Store the processed data in an appropriate AWS service, like S3 or DynamoDB.
- **Monitoring**: Set up CloudWatch to monitor the pipeline's performance and to log any issues.

Goal: Learn to build a serverless data processing pipeline, understanding the flow of data through various AWS services.

Solution:

1. Bucket was created



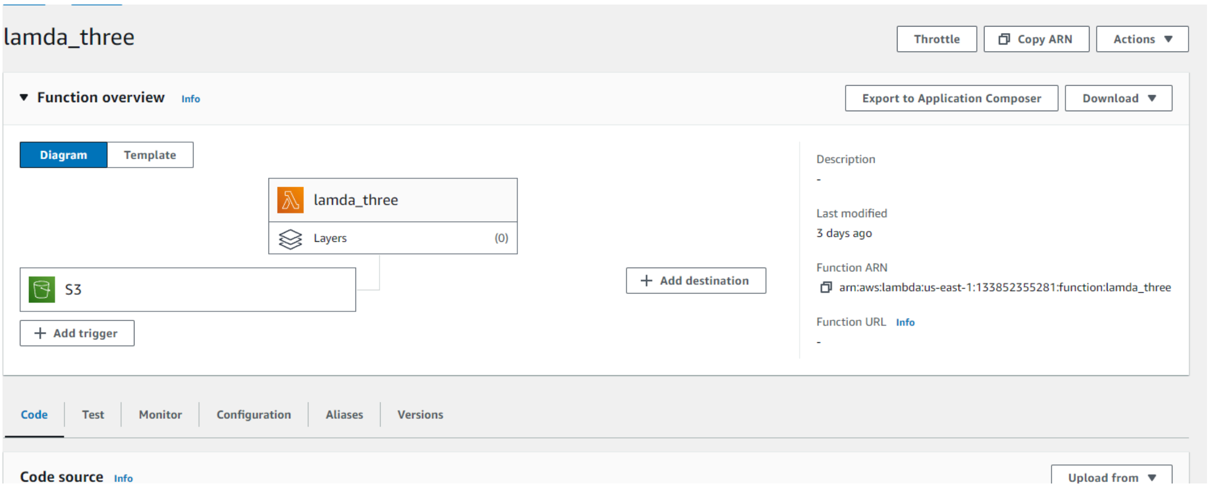
sushant-balti

Objects (2) Info

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
hello.html	html	February 21, 2024, 22:55:30 (UTC+05:45)	140.0 B	Standard
hello.txt	txt	February 21, 2024, 22:44:14 (UTC+05:45)	0 B	Standard

2. **Lambda function was created**



lamda_three

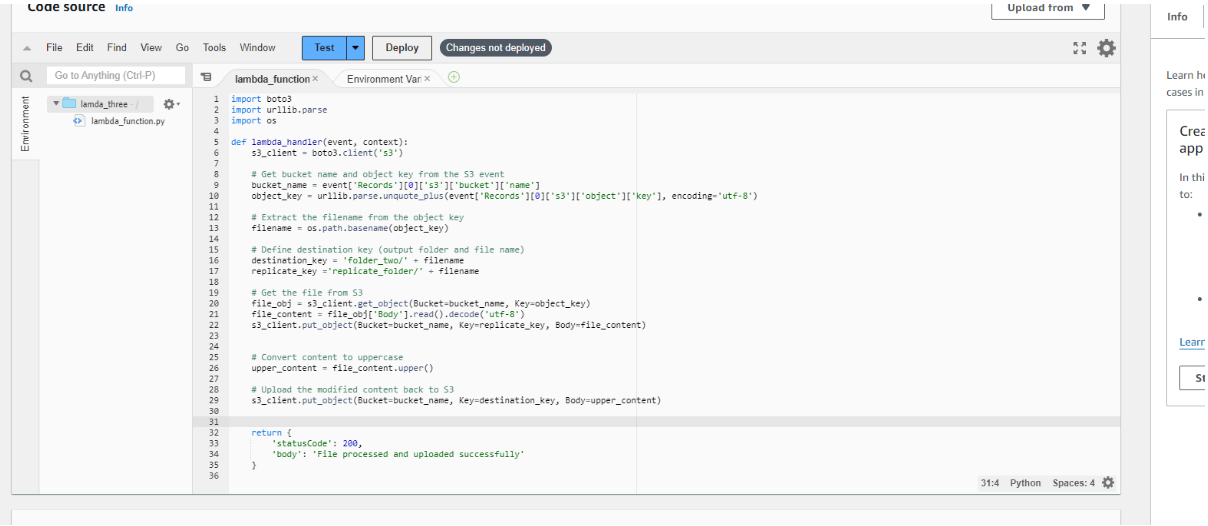
Description
-

Last modified
3 days ago

Function ARN
arn:aws:lambda:us-east-1:133852355281:function:lamda_three

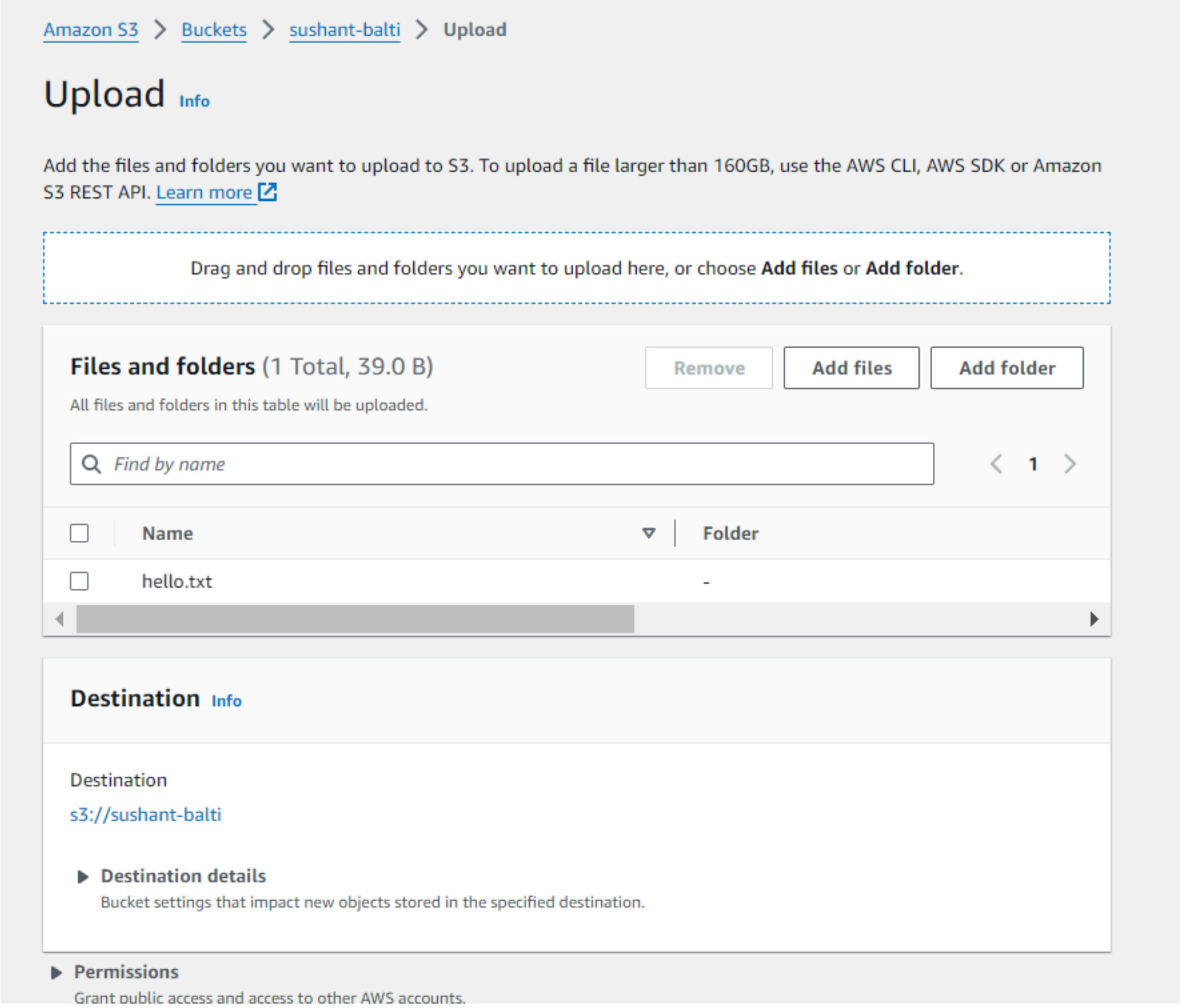
Function URL Info
-

3. **Code was placed to transform the received data to upper case and it triggers when the file is uploaded to s3 bucket**



```python
import boto3
import urllib.parse
import os

def lambda_handler(event, context):
    s3_client = boto3.client('s3')

    # Get bucket name and object key from the S3 event
    bucket_name = event['Records'][0]['s3']['bucket']['name']
    object_key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'], encoding='utf-8')

    # Extract the filename from the object key
    filename = os.path.basename(object_key)

    # Define destination key (output folder and file name)
    destination_key = 'folder_two/' + filename
    replicate_key ='replicate_folder/' + filename

    # Get the file from S3
    file_obj = s3_client.get_object(Bucket=bucket_name, Key=object_key)
    file_content = file_obj['Body'].read().decode('utf-8')
    s3_client.put_object(Bucket=bucket_name, Key=replicate_key, Body=file_content)


    # Convert content to uppercase
    upper_content = file_content.upper()

    # Upload the modified content back to S3
    s3_client.put_object(Bucket=bucket_name, Key=destination_key, Body=upper_content)


    return {
        'statusCode': 200,
        'body': 'File processed and uploaded successfully'
    }

```

4. **Upload file to s3 bucket**



Upload

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 folder**.

Files and folders (1 Total, 39.0 B)

All files and folders in this table will be uploaded.

□	Name	Folder
□	hello.txt	-

Destination

Destination

s3://sushant-balti

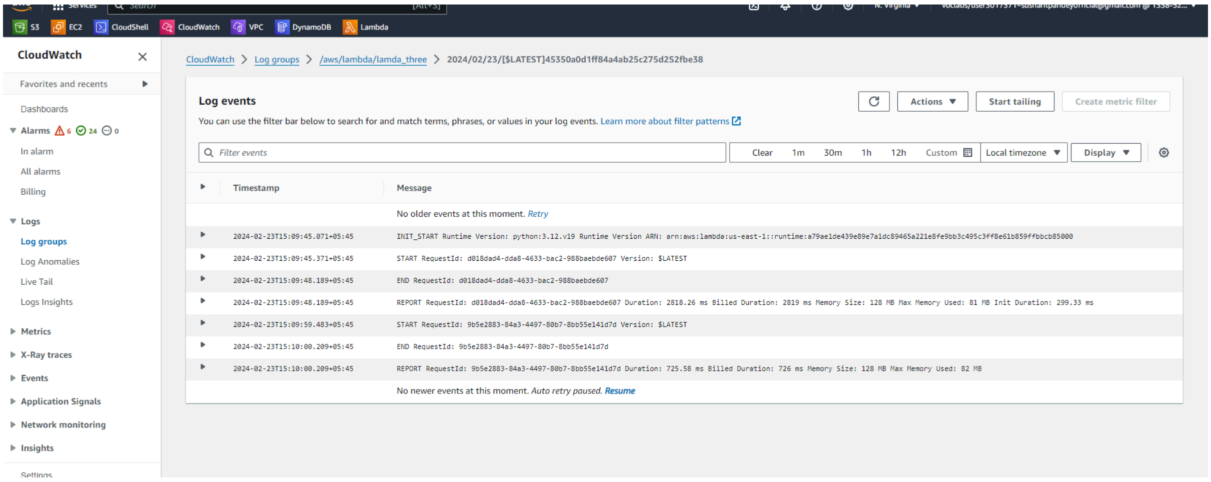
▶ **Destination details**

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

▶ **Permissions**

Grant public access and access to other AWS accounts.

5. **Here is the log report from the cloud watch that lambda function was triggered successfully**

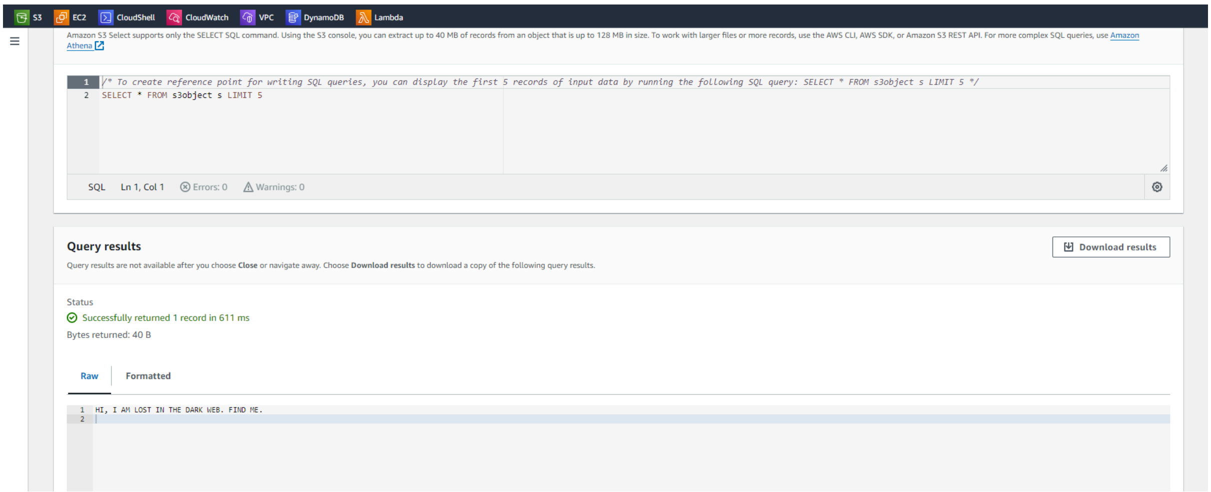


Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

```
Timestamp                       Message
                                No older events at this moment. Retry
2024-02-23T15:09:45.071+05:45   INIT_START Runtime Version: python:3.12.v19 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:a79ae1de439e89e7a1dc89465a221e8fe9bb3c495c3ff8e61b859ffbbcb85000
2024-02-23T15:09:45.371+05:45   START RequestId: d018dad4-dda8-4633-bac2-988baebde607 Version: $LATEST
2024-02-23T15:09:48.189+05:45   END RequestId: d018dad4-dda8-4633-bac2-988baebde607
2024-02-23T15:09:48.189+05:45   REPORT RequestId: d018dad4-dda8-4633-bac2-988baebde607 Duration: 2818.26 ms Billed Duration: 2819 ms Memory Size: 128 MB Max Memory Used: 81 MB Init Duration: 299.33 ms
2024-02-23T15:09:59.483+05:45   START RequestId: 9b5e2883-84a3-4497-80b7-8bb55e141d7d Version: $LATEST
2024-02-23T15:10:00.209+05:45   END RequestId: 9b5e2883-84a3-4497-80b7-8bb55e141d7d
2024-02-23T15:10:00.209+05:45   REPORT RequestId: 9b5e2883-84a3-4497-80b7-8bb55e141d7d Duration: 725.58 ms Billed Duration: 726 ms Memory Size: 128 MB Max Memory Used: 82 MB
                                No newer events at this moment. Auto retry paused. Resume
```

6. **Here is the query that our lower case data was successfully transformed to upper case**



Amazon S3 Select supports only the SELECT SQL command. Using the S3 console, you can extract up to 40 MB of records from an object that is up to 128 MB in size. To work with larger files or more records, use the AWS CLI, AWS SDK, or Amazon S3 REST API. For more complex SQL queries, use Amazon Athena

```sql
/* To create reference point for writing SQL queries, you can display the first 5 records of input data by running the following SQL query: SELECT * FROM s3object s LIMIT 5 */
SELECT * FROM s3object s LIMIT 5
```

Query results

Query results are not available after you choose **Close** or navigate away. Choose **Download results** to download a copy of the following query results.

Status

Successfully returned 1 record in 611 ms

Bytes returned: 40 B

```
HI, I AM LOST IN THE DARK WEB. FIND ME.
```