

# Automated Resource Tagging in AWS using Lambda

## Problem Statement

When we use a huge amount of S3 buckets in an organisation, It becomes difficult to differentiate which project or department or cost center is contributing what amount in our aws consolidated bills.

## Objective

In order to solve the problem statement, We will create multiple tags and attach it to S3 buckets. It will help us to maintain better organization, improve searchability, and reduce manual effort.

## Brainstorming:

- How to add tags?
- How to fill the knowledge gap in terms of AWS, Python, JSON?
- How does it work?
- Where should I start to add tags?
- How to write the code?
- Which programming language would be more efficient and suitable for writing the code?
- Where should I learn the languages from?

## Prerequisites:

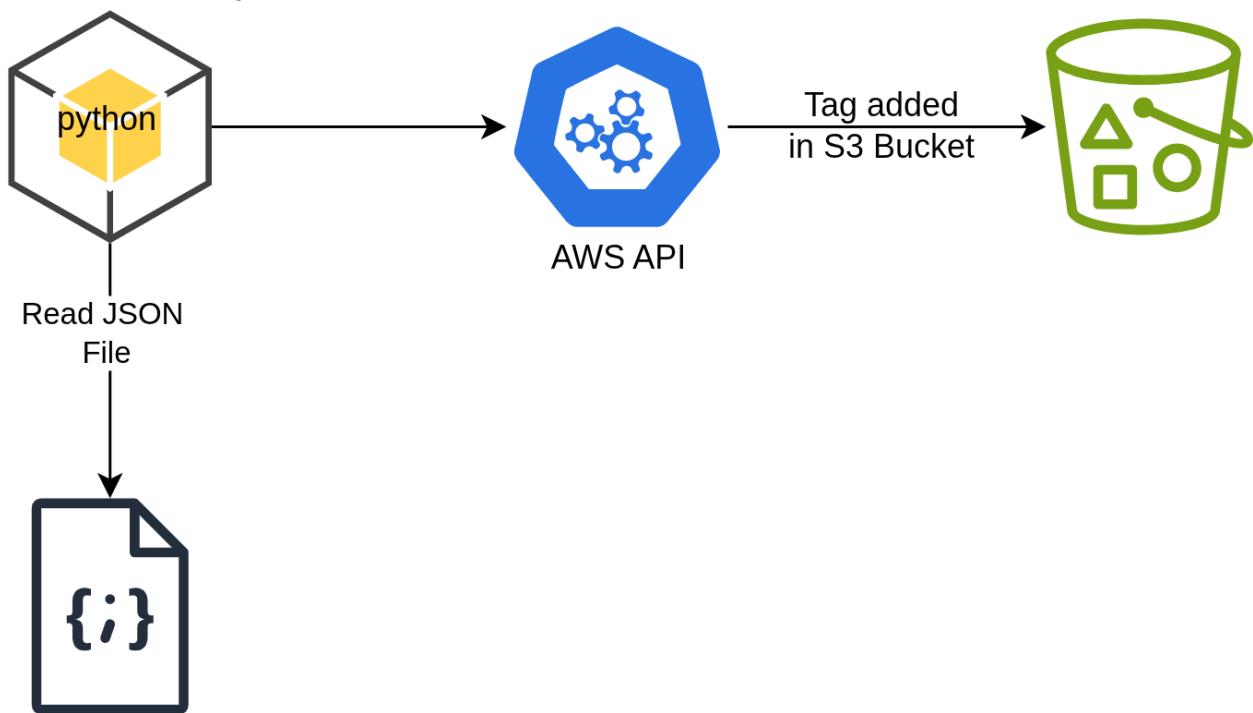
Before starting this project, ensure that the following are installed, set up, or understood.

## Tools used:

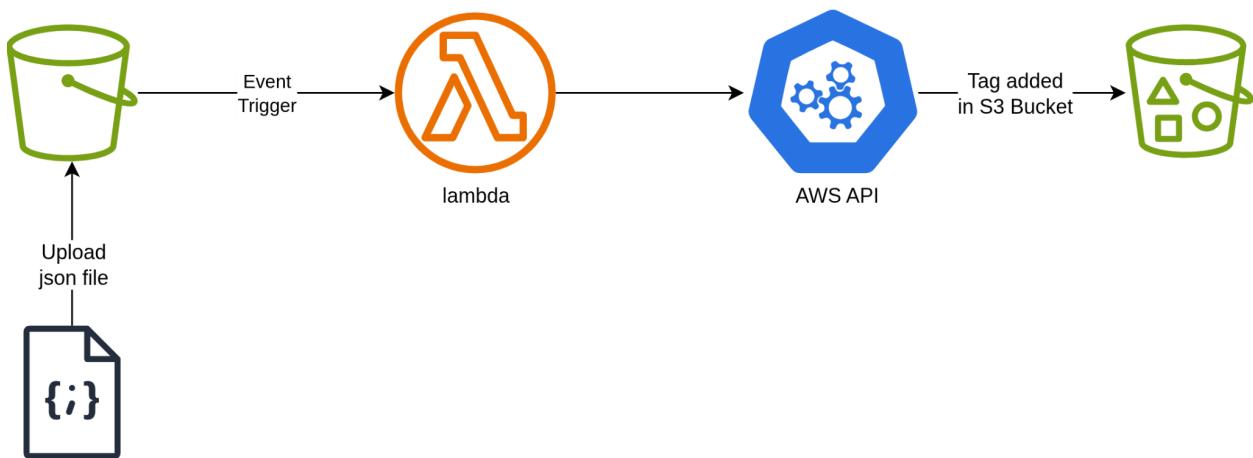
Tools	Installation link and source info
VS code	Code editor for the project
Python 3.12.3	Programming language used in the project
Github	To manage code and keep track of the project
AWS Boto3 SDK	for interacting with AWS services in Python
HTML	Supporting object
AWS account	To use AWS service like s3, lambda, IAM, cloudWatch
AWS s3	To store the uploaded file
AWS lambda	To automatically add tag on the uploaded files
IAM roles	To manage roles and permission
AWS cloudwatch	To monitor the logs of the lambda service

## Architecture :

### Local flow of project:



### AWS flow of project:



## Thought Process :

- First, write the code in VS code using Python, by installing the Boto3 library to create a S3 bucket.

```

import boto3 → BOTO3 Script
def create_s3_bucket(bucket_name):
    try:
        client = boto3.client('s3')
        response = client.create_bucket(
            Bucket = bucket_name,
            CreateBucketConfiguration = [
                'LocationConstraint' : 'us-west-2' → Oregon Region
            ],
        )
        print(f"The bucket '{bucket_name}' is created successfully ")
    except Exception as e:
        print("An Exception occurred while creating Bucket",e)

if __name__ == '__main__':
    try:
        bucket_name = input("Enter the Bucket name: ").strip()
        if len(bucket_name) != 0:
            create_s3_bucket(bucket_name)
        else:
            print("Please Enter the Bucket name! ")
    except Exception as e:
        print("An exception has occurred naming the bucket!!!")

```

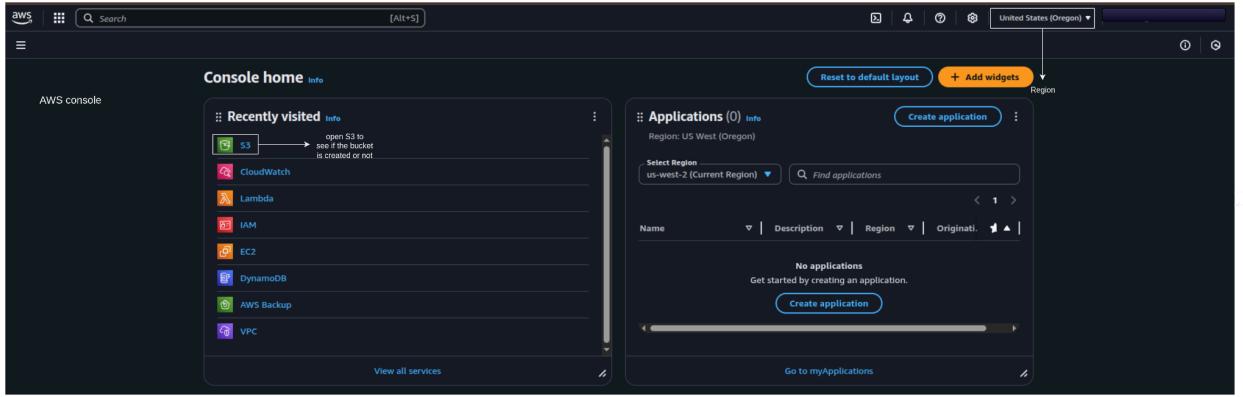
- Provide the unique bucket name  
Why Unique: S3 requires unique naming convention across the globe.

```

● anchal@anchal-L0Q-15IAX9:~/Documents/project-s3tagger$ python3 s3-bucket
Enter the Bucket name: project-automated-tagging
The bucket 'project-automated-tagging' is created successfully

```

- Check if the bucket is created in AWS S3 or not .



- We have no files in the Object (project-s3-tagger).

The screenshot shows the AWS S3 'Objects' list for the 'project-automated-tagging' bucket. There is one object named 'project-s3-tagging.json' which is a 'json' file uploaded on July 25, 2025, at 00:07:15 (UTC+05:30). The file size is 264.0 B and it is stored in the Standard storage class.

Name	Type	Last modified	Size	Storage class
project-s3-tagging.json	json	July 25, 2025, 00:07:15 (UTC+05:30)	264.0 B	Standard

- Create a JSON file , to upload it as an Object in the bucket (project-automated-tagging).

```
home > anchal > Documents > project-s3tagger > ats3l-jsonfile > ...
1  import json
2  def read_file(file_name):
3      data = {
4          "owner" : "Anchal",
5          "costCenter" : "1234567890",
6          "departmentID" : "dept-12345",
7          "resourceOwner" : "AnchalAgrahari",
8          "projectName" : "awsResourceTagger",
9          "githubLink" : "https://github.com/AnchalAgrahari/test-repo/tree/json/project-aws-tagger"
10     }
11     with open (file_name + ".json", "w") as f:
12         json.dump(data, f, indent = 4)
13         print(f"Data written to {file_name}")
14
15     if __name__=='__main__':
16         try:
17             file_name=input("Enter the file name: ")
18             read_file(file_name)
19         except Exception as e:
20             print("An error occurred ",e)
```

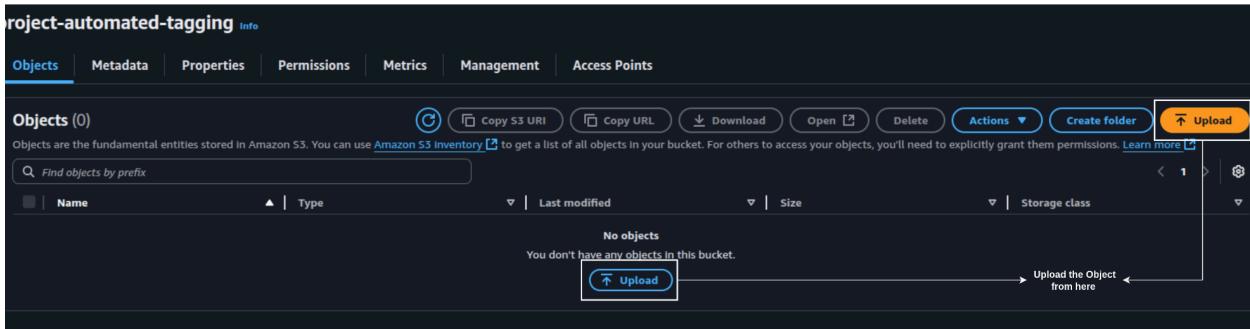
- After running the json code.

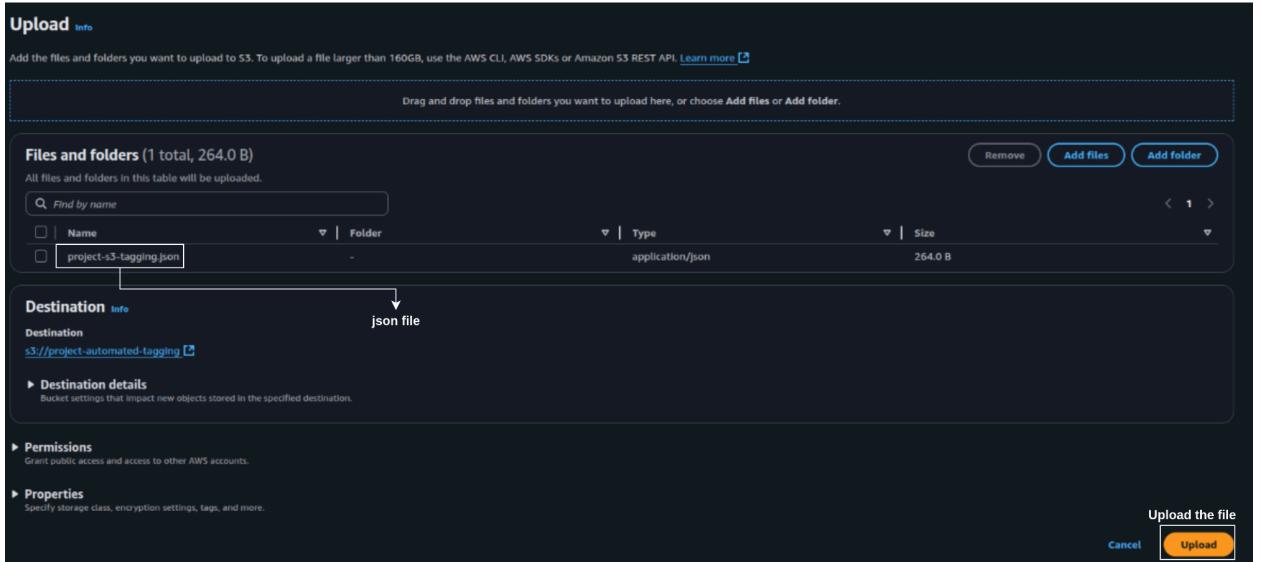
```
anchal@anchal-L0Q-15IAIX9:~/Documents/project-s3tagger$ python3 ats3l-jsonfile
Enter the file name: project-s3-tagging
Data written to project-s3-tagging.json
```

- We get the json file.

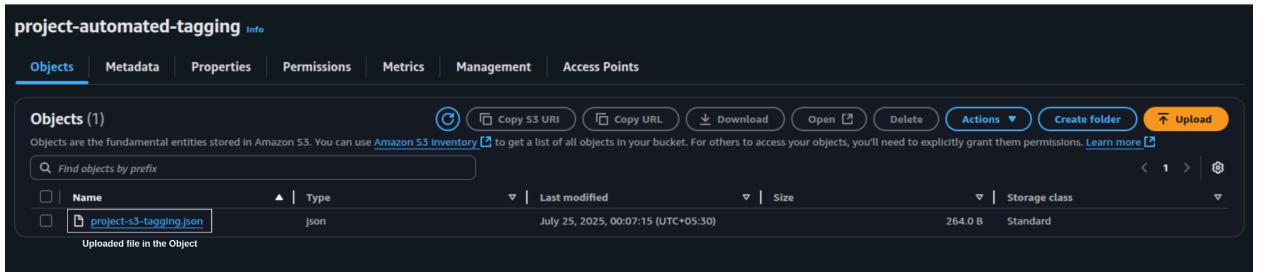
```
1 [ {
2   "owner": "Anchal",
3   "costCenter": "1234567890",
4   "departmentID": "dept-12345",
5   "resourceOwner": "AnchalAgrahari",
6   "projectName": "awsResourceTagger",
7   "githubLink": "https://github.com/AnchalAgrahari/test-repo/tree/json/project-aws-tagger"
8 } ]
```

- Upload it to an s3 bucket.

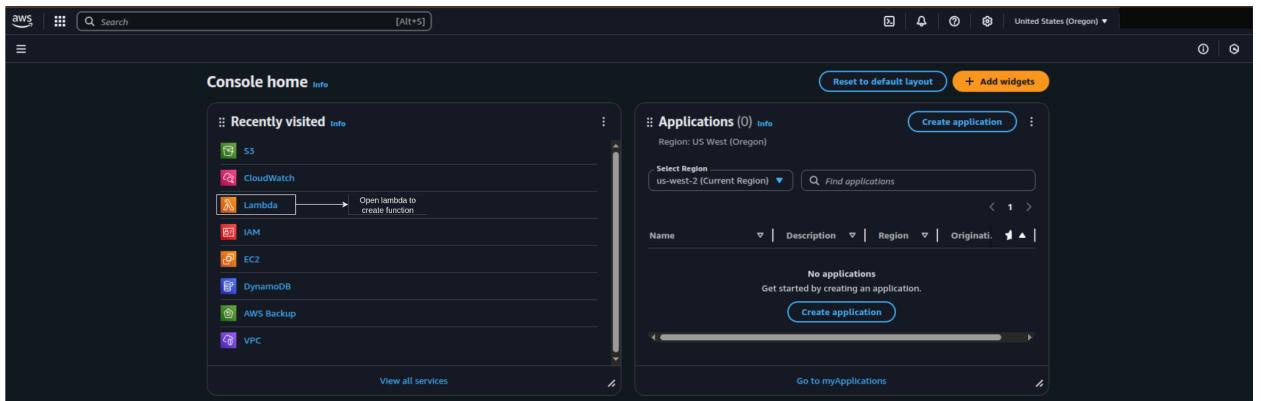




- Check if the Object is uploaded to the S3 bucket.

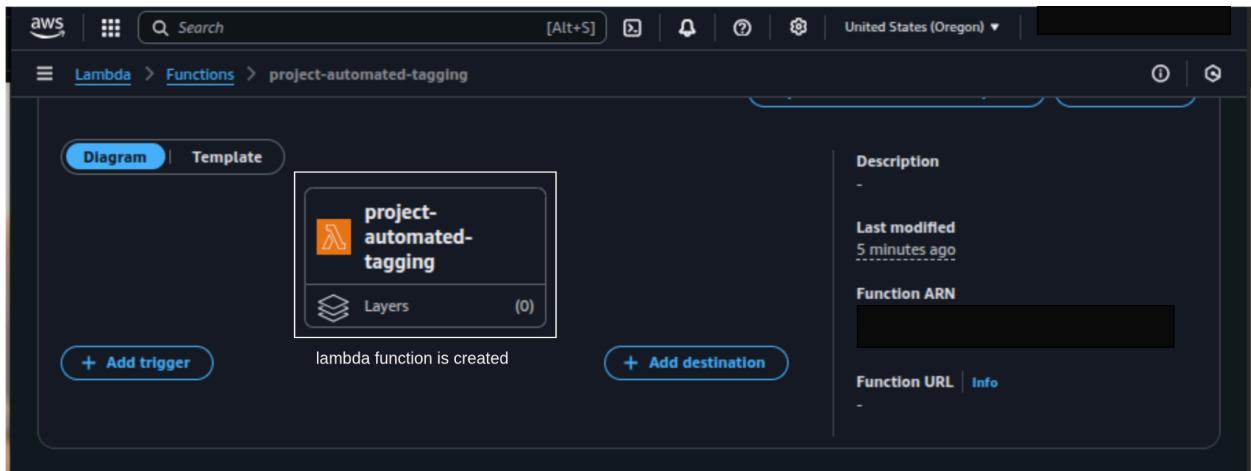


- Then create a Function in lambda

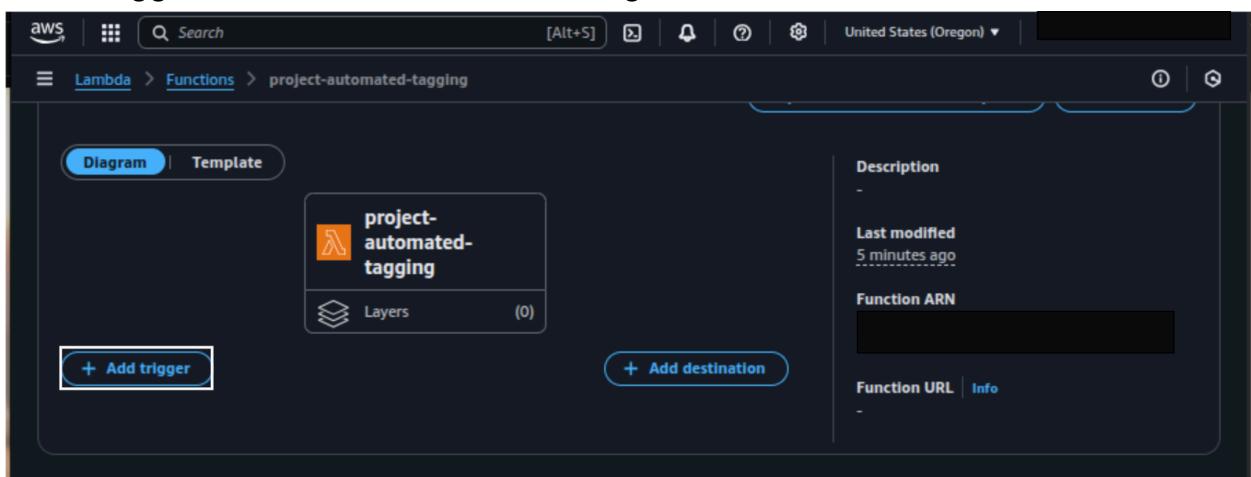


The screenshot shows the AWS Lambda Functions page. At the top, there's a navigation bar with the AWS logo, search, notifications, and a dropdown for 'United States (Oregon)'. Below the navigation is a breadcrumb trail: 'Lambda > Functions'. On the left, a sidebar has a 'Create function' button. The main area is titled 'Functions (0)' and includes a search bar and a table header with columns: 'Function name', 'Description', 'Package type', 'Runtime', and 'Last modified'. A message at the bottom says 'There is no data to display.'

The screenshot shows the 'Create function' wizard. The top navigation bar includes a search bar and a dropdown for 'United States (Oregon)'. The breadcrumb trail is 'Lambda > Functions > Create function'. The first step, 'Create function', is selected. It offers three options: 'Author from scratch' (selected), 'Use a blueprint', and 'Container image'. The 'Basic information' step is shown next, with fields for 'Function name' (set to 'project-automated-tagging'), 'Runtime' (set to 'Python 3.13'), and 'Architecture' (set to 'x86\_64'). Other steps like 'Permissions' and 'Additional configurations' are also visible.



- Add a trigger to the lambda function using the S3 bucket.



Lambda > Add triggers

## Add trigger

**Trigger configuration** [Info](#)

S3 aws asynchronous storage

**Bucket**  
Choose or enter the ARN of an S3 bucket that serves as the event source. The bucket must be in the same region as the function.

project-automated-tagging [choose the bucket name](#)

**Event types**  
Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.

All object create events

**Prefix - optional**  
Enter a single optional prefix to limit the notifications to objects with keys that start with matching characters. Any [special characters](#) must be URL encoded.

e.g. images/

**Suffix - optional**  
Enter a single optional suffix to limit the notifications to objects with keys that end with matching characters. Any [special characters](#) must be URL encoded.

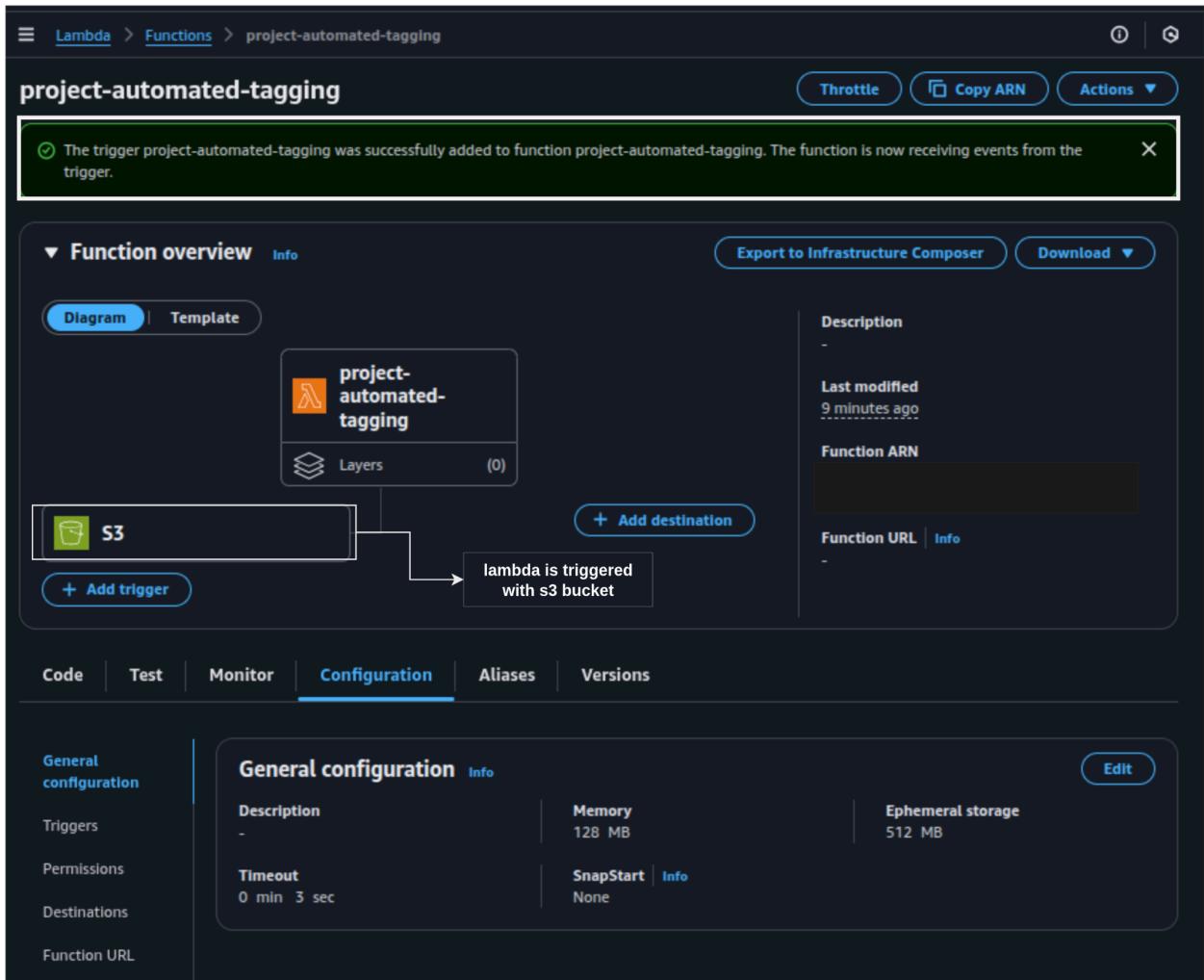
e.g. .jpg

**Recursive invocation**  
If your function writes objects to an S3 bucket, ensure that you are using different S3 buckets for input and output. Writing to the same bucket increases the risk of creating a recursive invocation, which can result in increased Lambda usage and increased costs. [Learn more](#)

I acknowledge that using the same S3 bucket for both input and output is not recommended and that this configuration can cause recursive invocations, increased Lambda usage, and increased costs.

Lambda will add the necessary permissions for AWS S3 to invoke your Lambda function from this trigger. [Learn more](#) about the Lambda permissions model.

[Cancel](#) [Add](#)



- Write the python and boto3 script in lambda handler

Code Test Monitor Configuration Aliases Versions

Code source [Ints](#)

EXPLORER ASDFDJKLJSKJTRWQJSHJH2022072222 lambda\_function.py

lambda\_function.py

```
1 import json
2
3 def lambda_handler(event, context):
4     # TODO implement
5     return {
6         'statusCode': 200,
7         'body': json.dumps('Hello from Lambda!')
8     }
```

DEPLOY Deploy (Ctrl+Shift+F10) Test (Ctrl+Shift+F11)

TEST EVENTS [None selected] Create new test event

ENVIRONMENT VARIABLES

Open in Visual Studio Code [Upload from...](#)

File Edit View Insert Run Tools Help

Ctrl + Alt + F11 Amazon Q

Ctrl + Shift + F11 Lambda Lambda

```
lambda_function.py X
lambda function.py
1 import json
2 import boto3 → boto3 and json script
3
4 def read_json_from_s3(bucket_name, key):
5     try:
6         client = boto3.client('s3')
7         s3_clientobj = client.get_object(Bucket = bucket_name, Key = key)
8         s3_clientdata = s3_clientobj['Body'].read().decode()
9         print("Printing s3_clientdata")
10        print(s3_clientdata)
11        print(type(s3_clientdata))
12
13        s3clientlist = json.loads(s3_clientdata)
14        print("json loaded data")
15        print(s3clientlist)
16        print(type(s3clientlist))
17        return s3clientlist
18    except Exception as e:
19        print("An exception has occurred while reading file from s3",e)
20
21 def add_tag_in_Bucket(bucket_name, data):
22     try:
23         client = boto3.client('s3')
24         tag_set = [{ 'Key':key, 'Value': value}for key, value in data.items()]
25         response = client.put_bucket_tagging(
26             Bucket = bucket_name,
27             Tagging={
28                 'TagSet': tag_set
29             },
30             ExpectedBucketOwner=' Owner ID'
31         )
32         print("Tagging Successful")
33     except Exception as e:
34         print("An exception has occurred while tagging", e)
35
36 def lambda_handler(event, context):
37     try:
38         bucket_name = 'project-ats3l' → bucket name
39         key = 'project-ats3l.json' → file in S3 Object
40         prased_data = read_json_from_s3(bucket_name, key)
41         print("Let's check data", prased_data)
42         add_tag_in_Bucket(bucket_name, prased_data)
43     except Exception as e:
44         print("An exception has occurred", e)
```

- Deploy the script with deploy button in left sidebar or **ctrl+shift+U**

The screenshot shows the AWS Lambda function editor interface. At the top, a green banner displays the message "Successfully updated the function project-automated-tagging.". The main area shows the code for `lambda_function.py`:

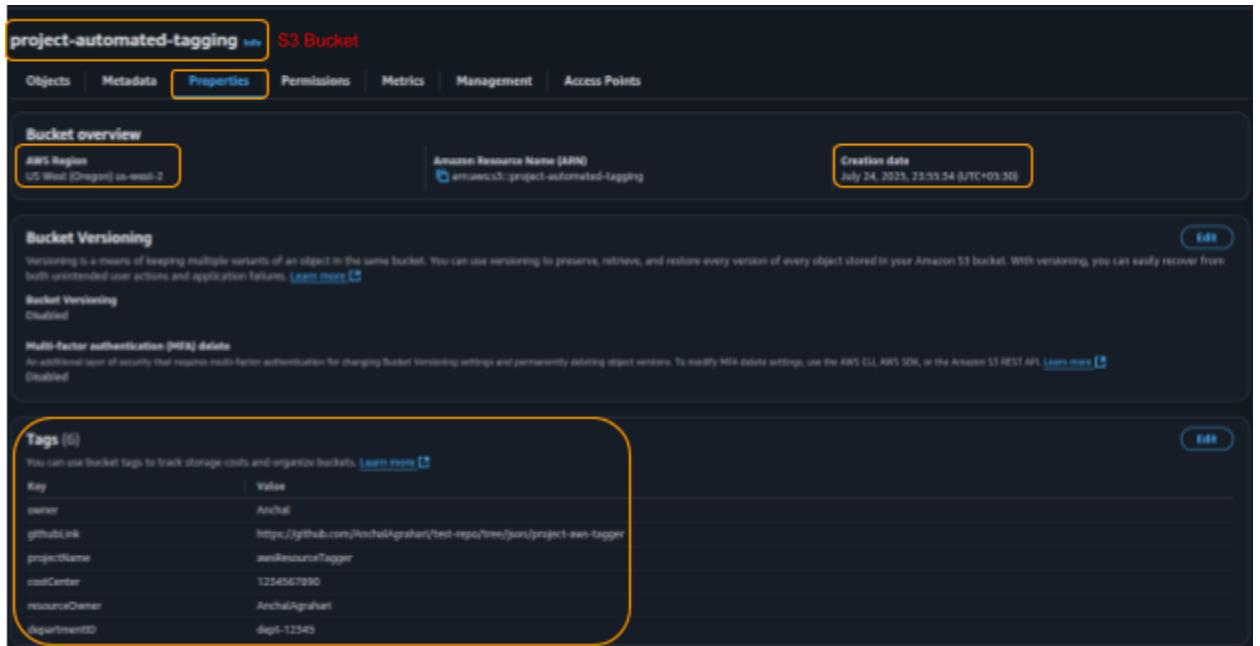
```

lambda_function.py X
lambda_function.py
18 def add_tag_in_Bucket(bucket_name, data):
19     try:
20         response = client.get_bucket_tagging()
21     except Exception as e:
22         print("Tagging Successful")
23     except Exception as e:
24         print("An exception has occurred while tagging", e)
25
26 def lambda_handler(event, context):
27     try:
28         bucket_name = 'project-automated-tagging'
29         key = 'project-s3-tagging.json'
30         parsed_data = read_json_from_s3(bucket_name, key)
31         add_tag_in_Bucket(bucket_name, parsed_data)
32     except Exception as e:
33         print("An exception has occurred", e)

```

The left sidebar shows the project structure: `PROJECT-AUTOMATED-TAGGING` containing `lambda_function.py`. Below it, under `DEPLOY`, are two buttons: `Deploy (Ctrl+Shift+U)` and `Test (Ctrl+Shift+T)`. At the bottom left, there's a section for `TEST EVENTS [NONE SELECTED]` with a button to `Create new test event`.

- Now in the s3 bucket, re-upload the file and refresh the page .
- In the bucket's properties, you'll see the added tags .



**Source:**

Aws boto3	<a href="https://boto3.amazonaws.com/v1/documentation/api/latest/index.html">https://boto3.amazonaws.com/v1/documentation/api/latest/index.html</a>
Article for stepwise guide to create bucket	<a href="https://medium.com/@techjunction.info/step-by-step-guide-how-to-create-an-s3-bucket-in-aws-84cfb158f405">https://medium.com/@techjunction.info/step-by-step-guide-how-to-create-an-s3-bucket-in-aws-84cfb158f405</a>
Guiding to create lambda function	<a href="https://medium.com/@selhorma/the-complete-beginners-guide-to-creating-an-aws-lambda-function-from-scratch-d03e6fa7e2b2">https://medium.com/@selhorma/the-complete-beginners-guide-to-creating-an-aws-lambda-function-from-scratch-d03e6fa7e2b2</a>
Basic JSON Learning	<a href="https://medium.com/@catherineisonline/what-is-json-ba631eeb0f32">https://medium.com/@catherineisonline/what-is-json-ba631eeb0f32</a>