# 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:

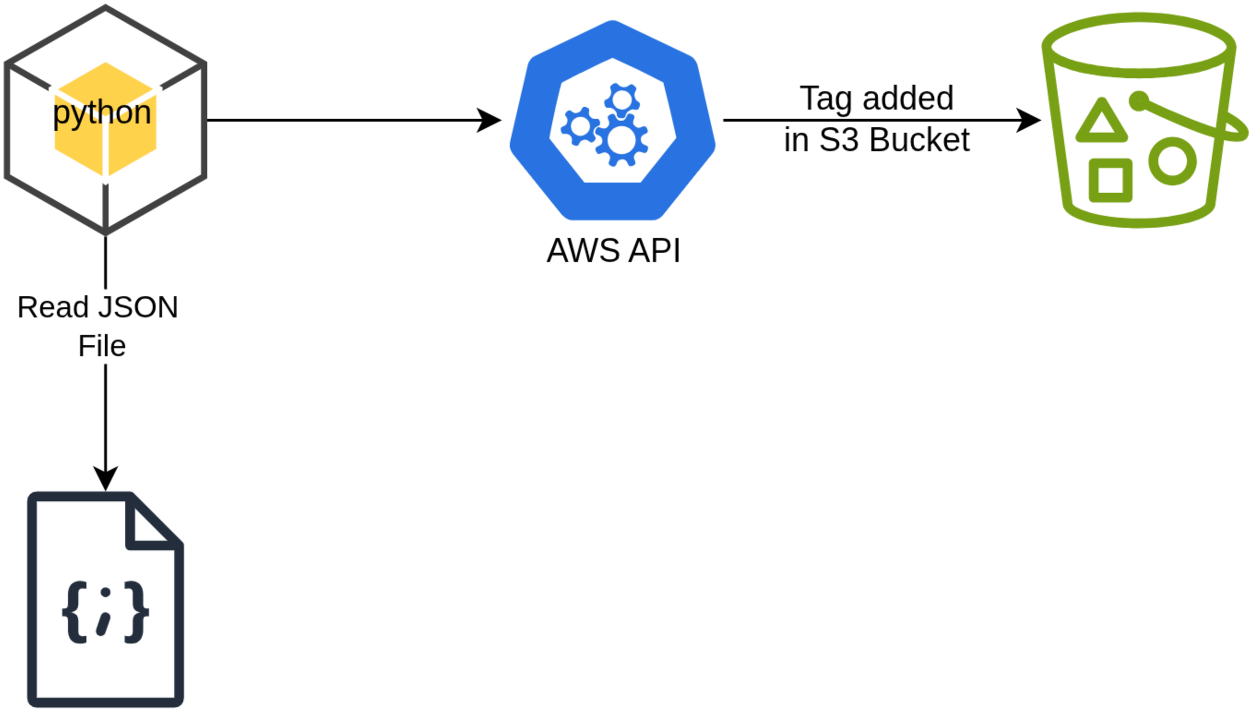
Before starting this project, ensure that the following are installed, set up, or understood.

### Tools used:

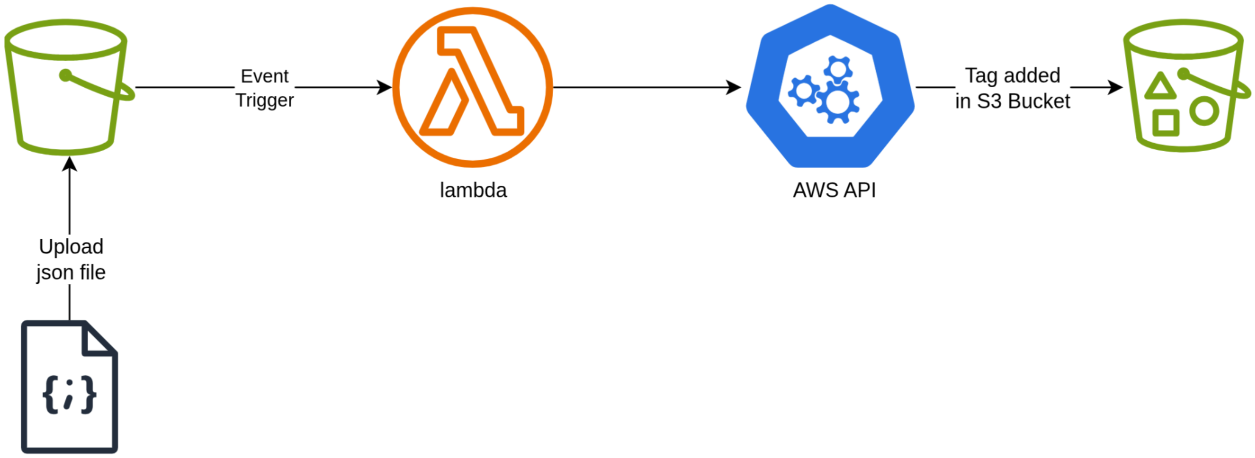
| Tools | Installation link and source info |
|---|---|
| VS code | Code editor for  the project |
| Python 3.12.3 | Programming language used in the project |
| Github | To manage code and keep track of the project |
| AWS Boto3 SDK | for interacting with AWS services in Python |
| HTML | Supporting 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:**

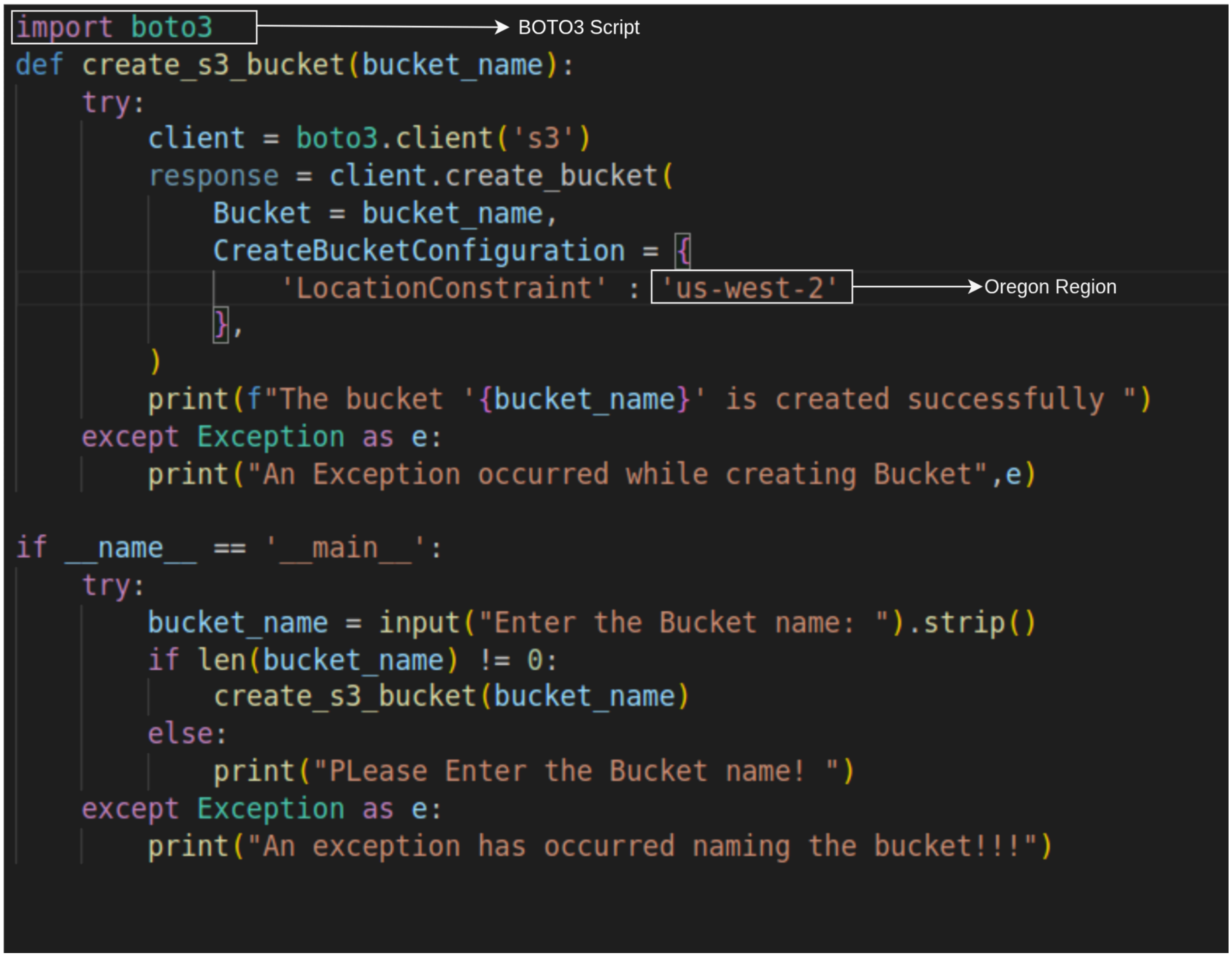


**AWS flow of project:**



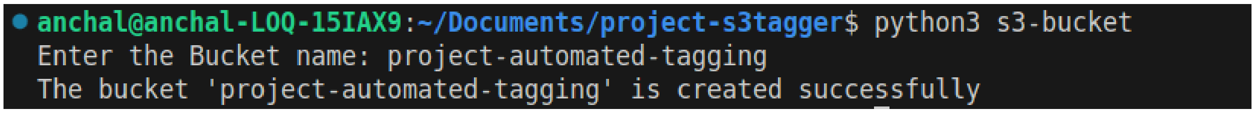
# Thought Process :

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



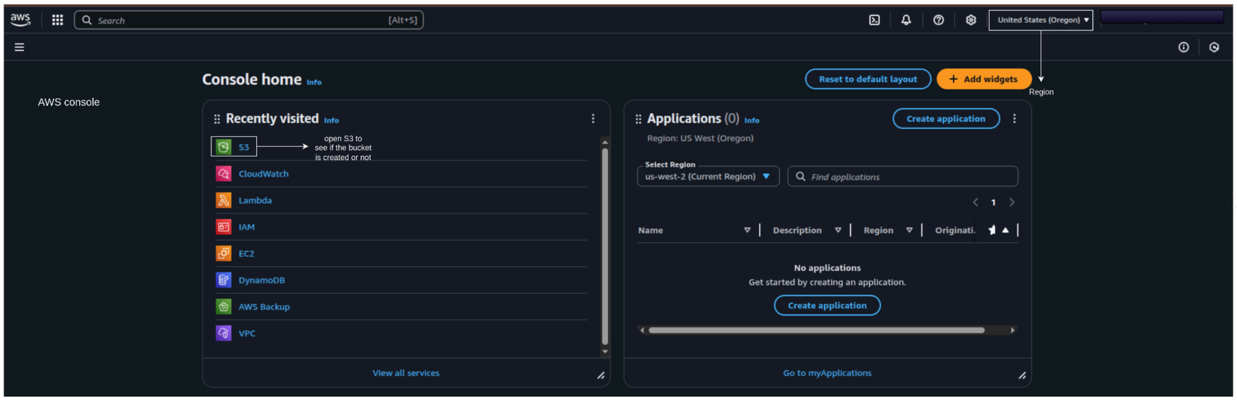
```python
import boto3
def create_s3_bucket(bucket_name):
    try:
        client = boto3.client('s3')
        response = client.create_bucket(
            Bucket = bucket_name,
            CreateBucketConfiguration = {
                'LocationConstraint' : 'us-west-2'
            },
        )
        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!!!")
```

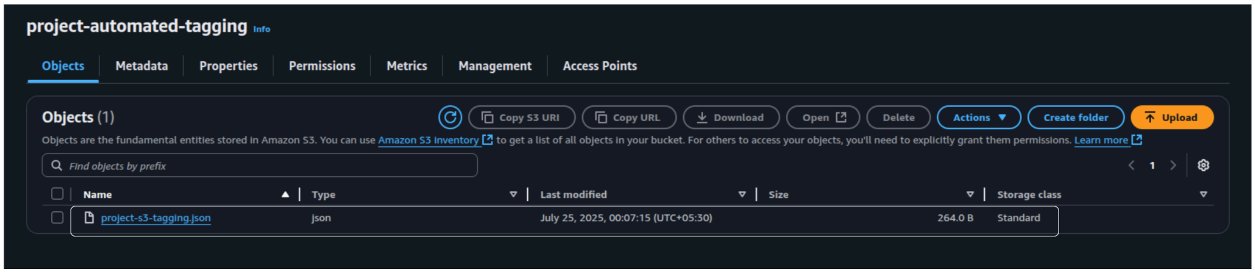
- Provide the unique bucket name
  Why Unique: S3 requires unique naming convention across the globe.



```
anchal@anchal-LOQ-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 .



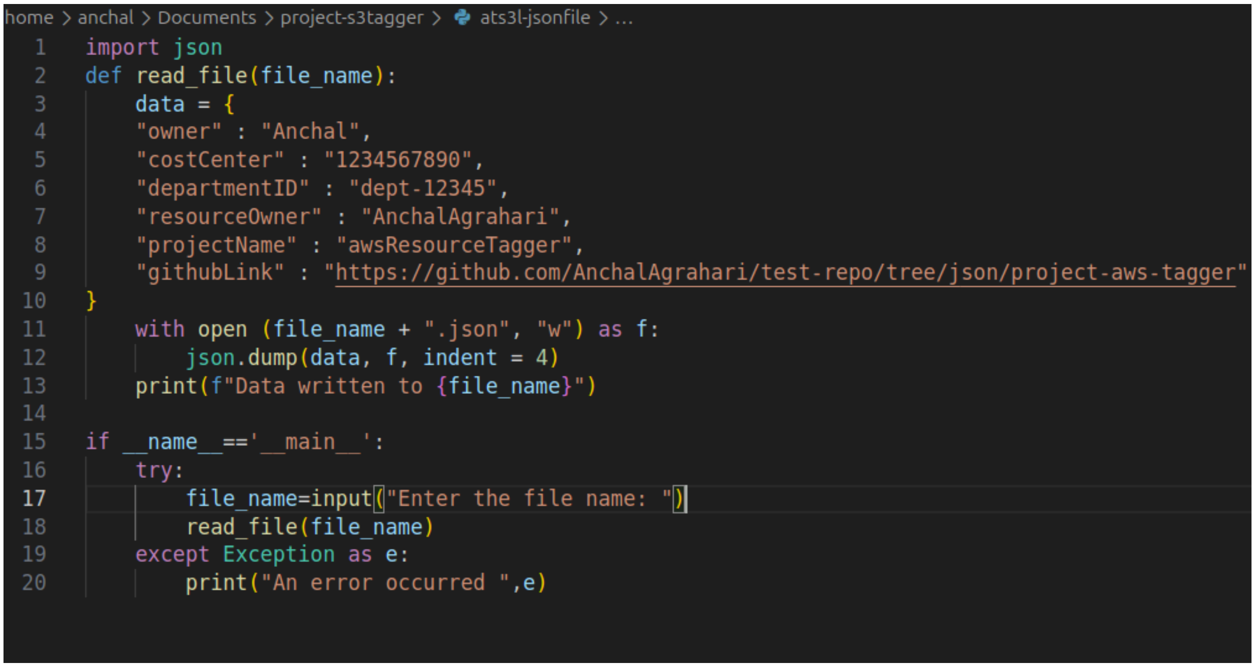


Objects (1)

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

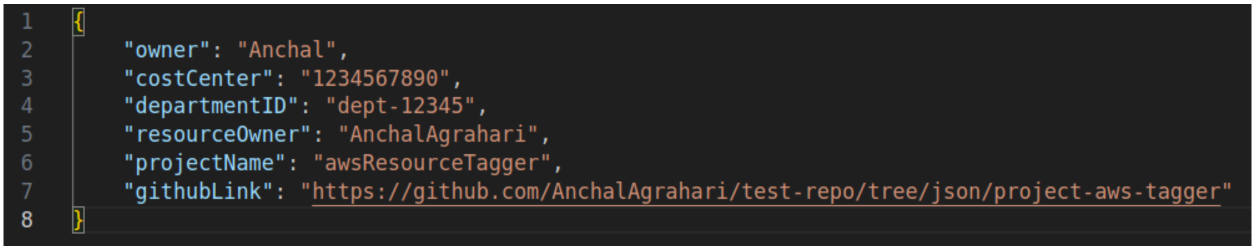
| □ | Name | Type | Last modified | Size | Storage class |
|---|---|---|---|---|---|
| □ | project-s3-tagging.json | json | July 25, 2025, 00:07:15 (UTC+05:30) | 264.0 B | Standard |

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



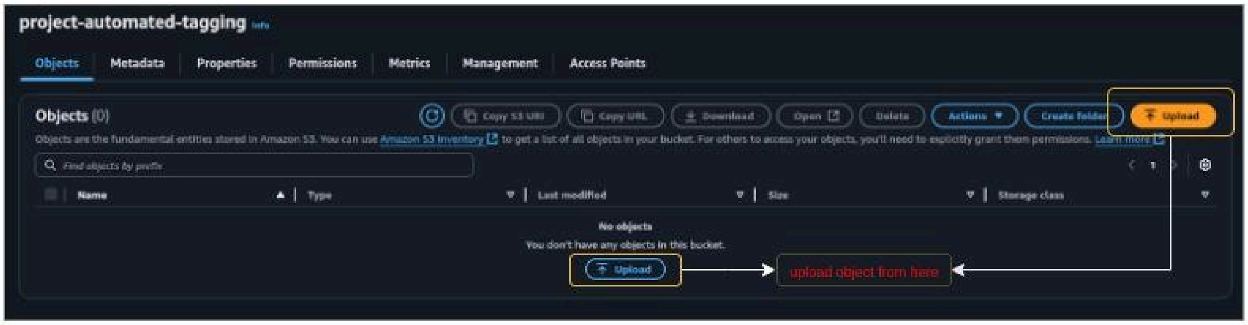
home > anchal > Documents > project-s3tagger > ats3l-jsonfile > ...

```
import json
def read_file(file_name):
    data = {
    "owner" : "Anchal",
    "costCenter" : "1234567890",
    "departmentID" : "dept-12345",
    "resourceOwner" : "AnchalAgrahari",
    "projectName" : "awsResourceTagger",
    "githubLink" : "https://github.com/AnchalAgrahari/test-repo/tree/json/project-aws-tagger"
}
    with open (file_name + ".json", "w") as f:
        json.dump(data, f, indent = 4)
    print(f"Data written to {file_name}")

if __name__=='__main__':
    try:
        file_name=input("Enter the file name: ")
        read_file(file_name)
    except Exception as e:
        print("An error occurred ",e)
```



```
{
    "owner": "Anchal",
    "costCenter": "1234567890",
    "departmentID": "dept-12345",
    "resourceOwner": "AnchalAgrahari",
    "projectName": "awsResourceTagger",
    "githubLink": "https://github.com/AnchalAgrahari/test-repo/tree/json/project-aws-tagger"
}
```

- Upload it to an s3 bucket.

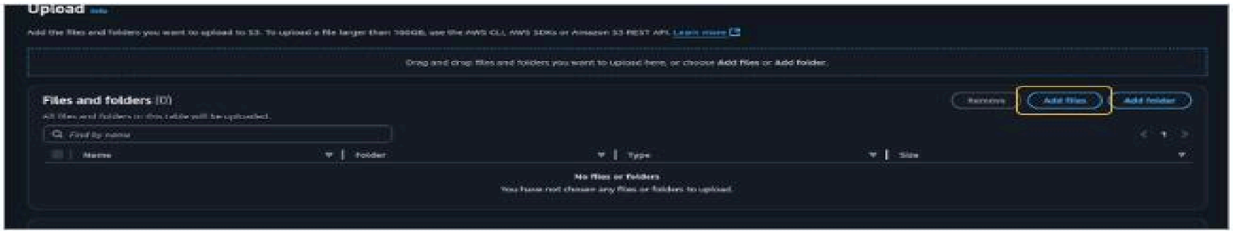


Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

| Name | Type | Last modified | Size | Storage class |
|---|---|---|---|---|

No objects

You don't have any objects in this bucket.



Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDKs 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**.

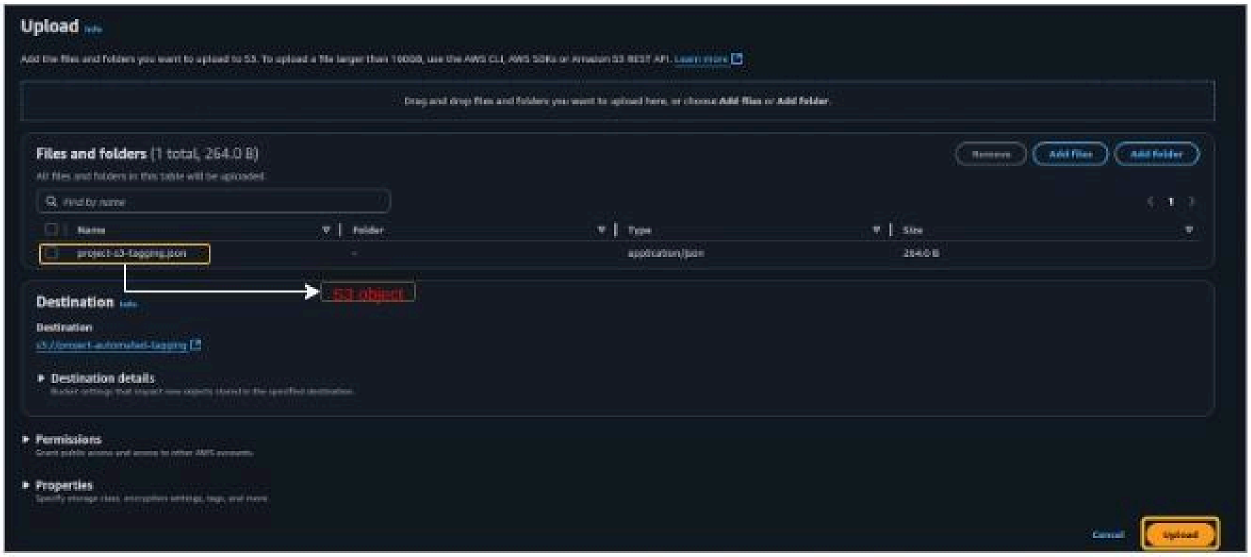
**Files and folders** (0)

All files and folders in this table will be uploaded.

| Name | Folder | Type | Size |
|---|---|---|---|

No files or folders

You have not chosen any files or folders to upload.



Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDKs 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, 264.0 B)

All files and folders in this table will be uploaded.

| Name | Folder | Type | Size |
|---|---|---|---|
| project-s3-tagging.json | - | application/json | 264.0 B |

Destination

s3://project-automated-tagging

▸ Destination details

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

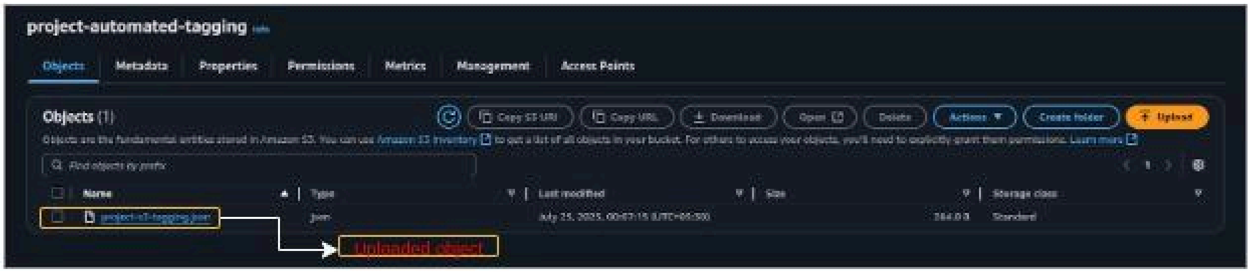
▸ Permissions

Grant public access and access to other AWS accounts.

▸ Properties

Specify storage class, encryption settings, tags, and more.

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



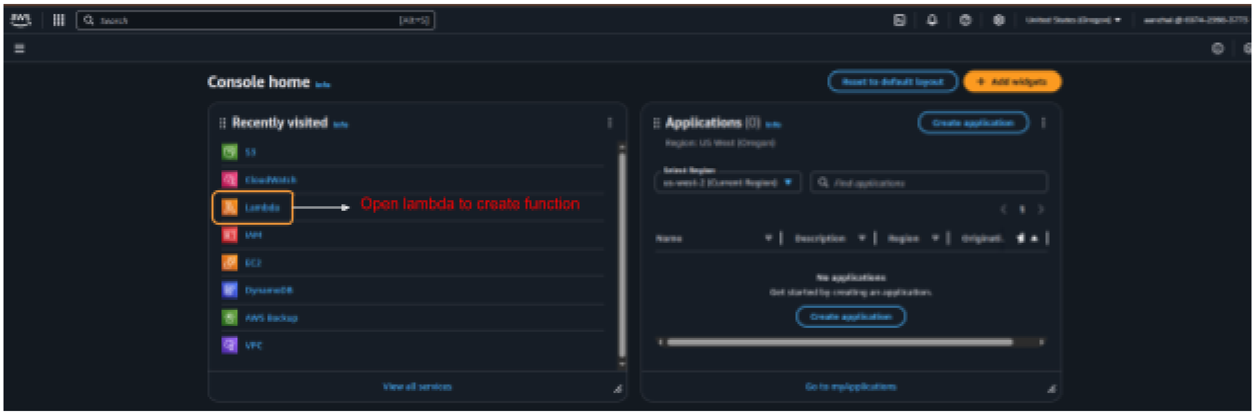
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 Inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Find objects by prefix

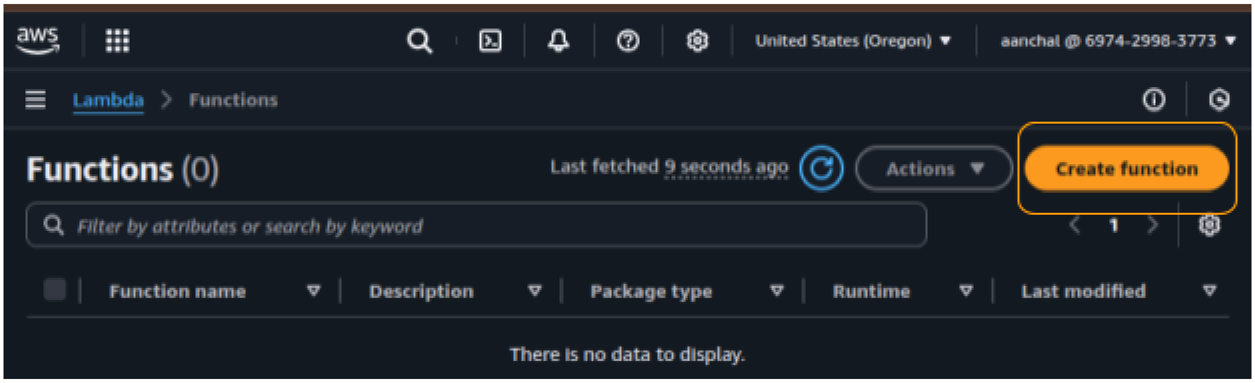
| □ | Name | Type | Last modified | Size | Storage class |
|---|---|---|---|---|---|
| □ | project-s3-tagging.json | json | July 25, 2025, 00:07:15 (UTC+05:30) | 264.0 B | Standard |

Uploaded object

- Then create a Function in lambda



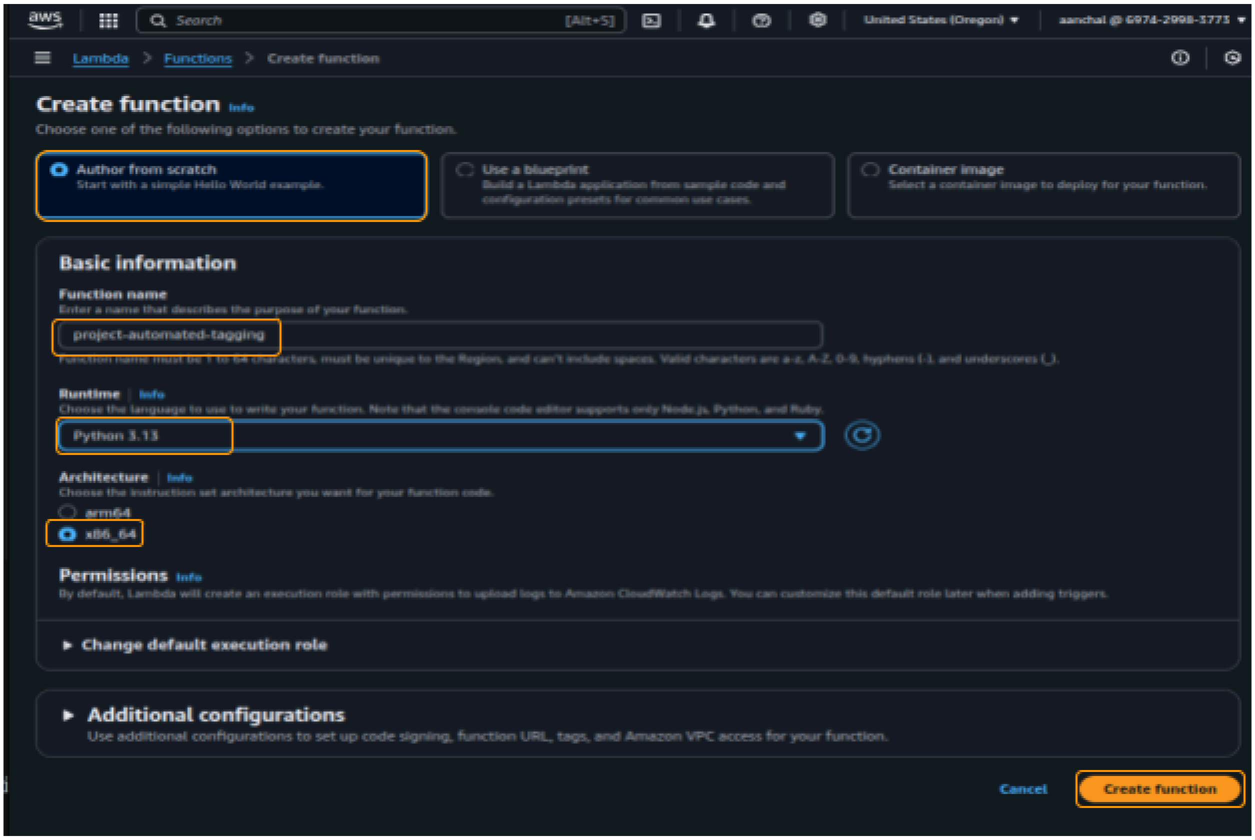
Open lambda to create function



aws
United States (Oregon) ▼
aanchal @ 6974-2998-3773 ▼
Lambda
Functions
Functions (0)
Last fetched 9 seconds ago
Actions ▼
Create function
Filter by attributes or search by keyword
1

| □ | Function name | Description | Package type | Runtime | Last modified |
|---|---|---|---|---|---|

There is no data to display.



aws
Search [Alt+S]
United States (Oregon) ▾
aanchal @ 6974-2998-3773 ▾
Lambda > Functions > Create function

## Create function Info

Choose one of the following options to create your function.

● Author from scratch
Start with a simple Hello World example.

○ Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.

○ Container image
Select a container image to deploy for your function.

### Basic information

**Function name**
Enter a name that describes the purpose of your function.

project-automated-tagging

Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (_).

**Runtime** | Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Python 3.13

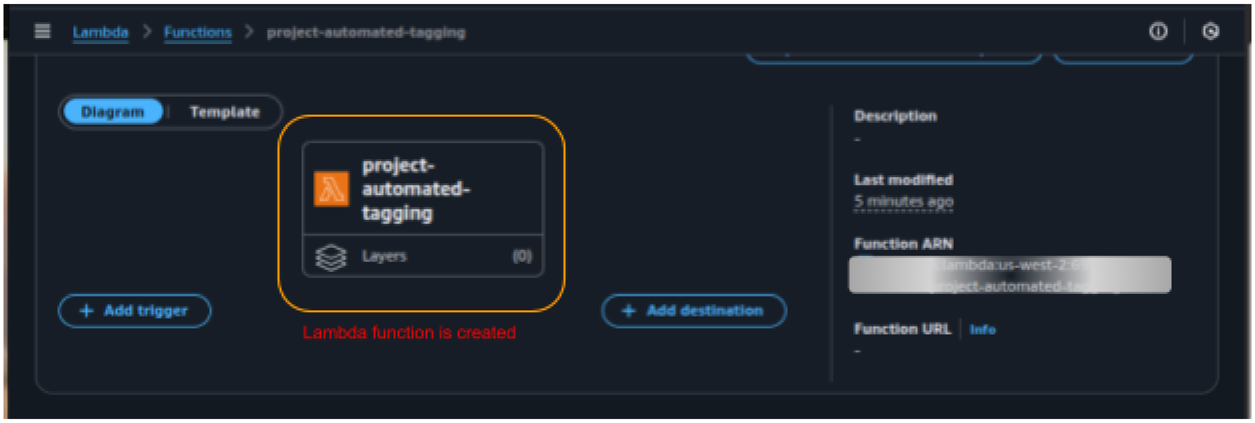
**Architecture** | Info
Choose the instruction set architecture you want for your function code.

○ arm64
● x86_64

**Permissions** Info
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

▸ Change default execution role

▸ **Additional configurations**
Use additional configurations to set up code signing, function URL, tags, and Amazon VPC access for your function.



Lambda > Functions > project-automated-tagging
Diagram
Template
project-automated-tagging
Layers (0)
+ Add trigger
+ Add destination

Lambda function is created

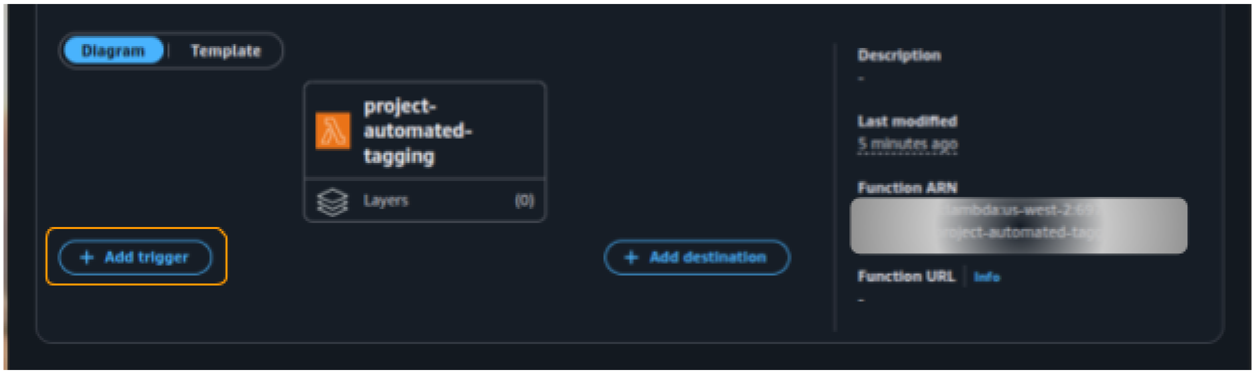
**Description**
-

**Last modified**
5 minutes ago

**Function ARN**
[redacted]

**Function URL** | Info
-

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



**Description**

-

**Last modified**

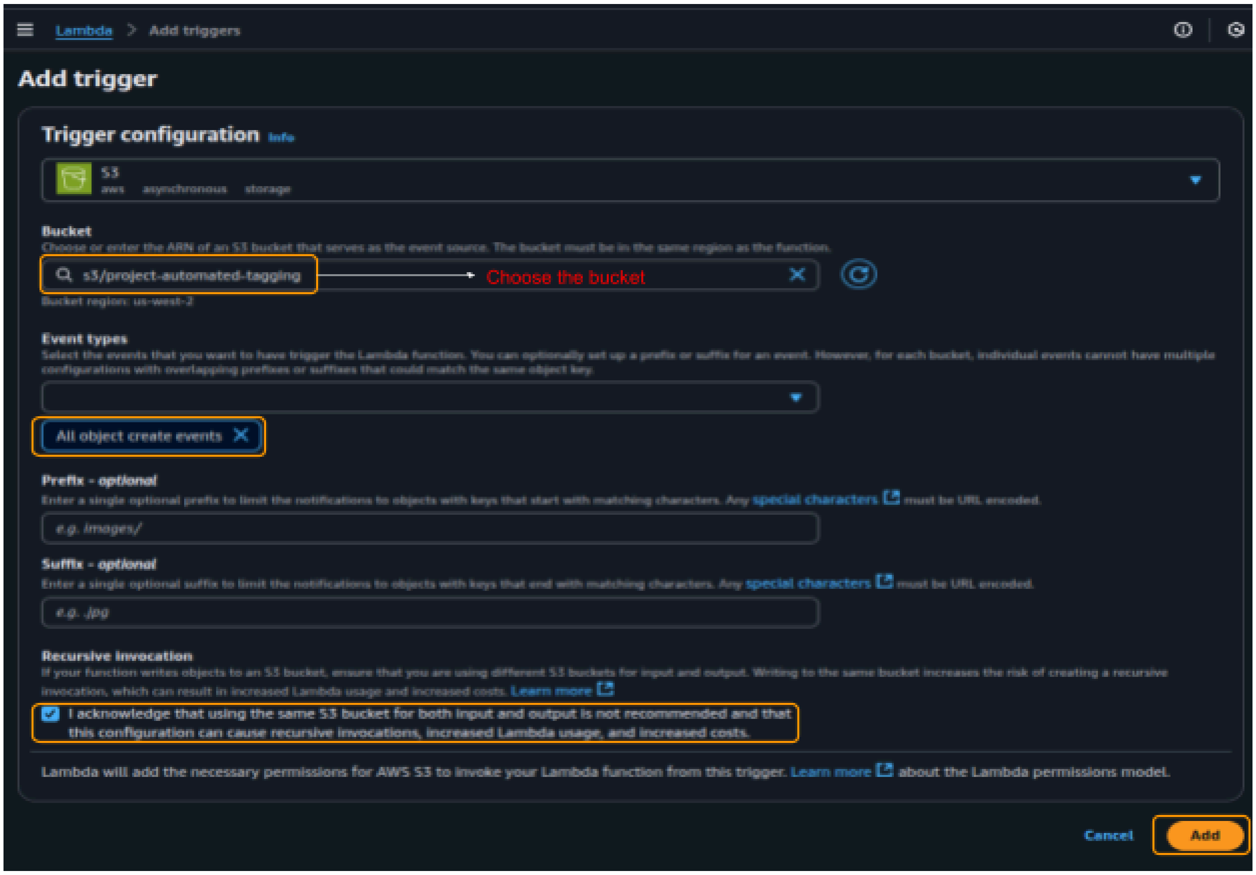
5 minutes ago

**Function ARN**

[redacted]

**Function URL** Info

-



## 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.

s3/project-automated-tagging → Choose the bucket

Bucket region: us-west-2

**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.

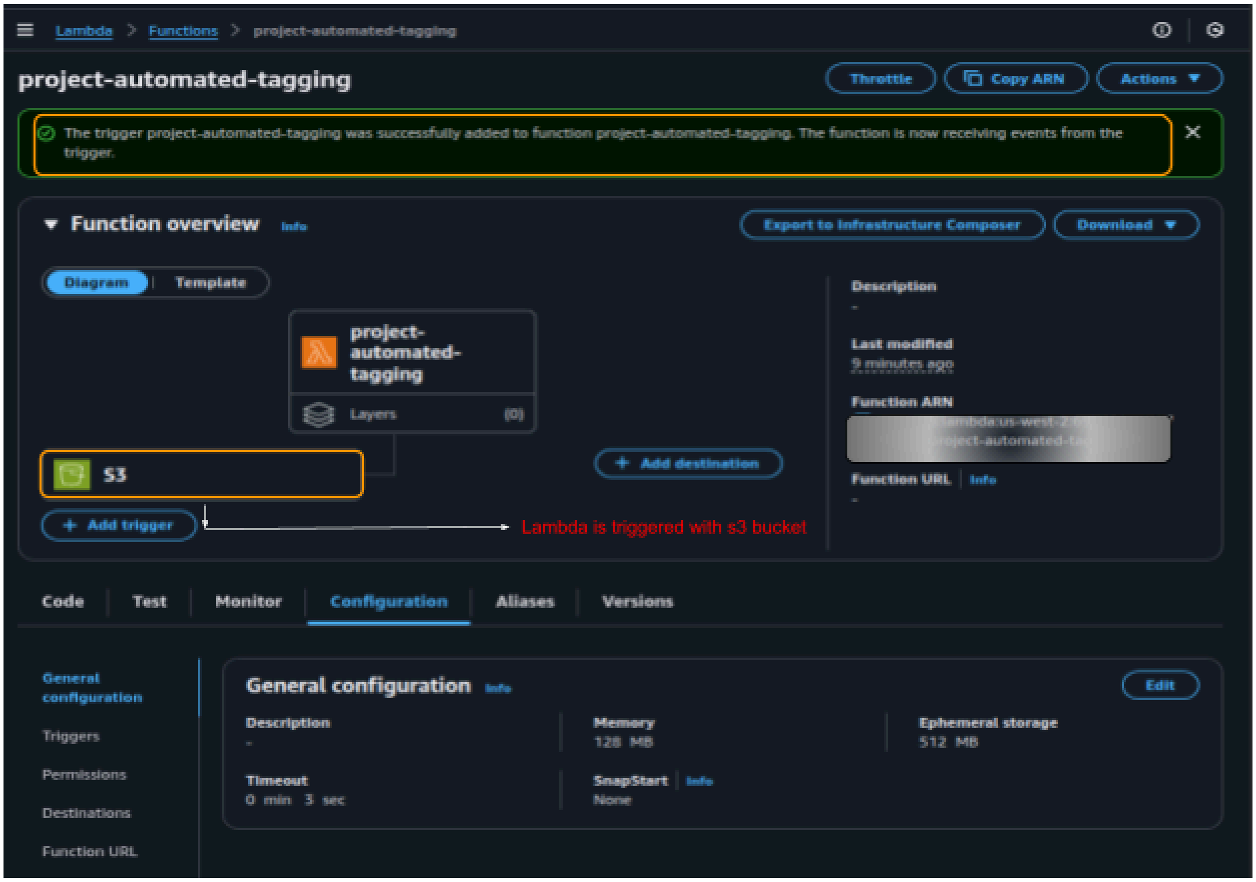
*e.g. .jpg*

**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.



project-automated-tagging

The trigger project-automated-tagging was successfully added to function project-automated-tagging. The function is now receiving events from the trigger.

Function overview Info

project-automated-tagging

Lambda is triggered with s3 bucket

Description
-

Last modified
9 minutes ago

Function ARN
[redacted]

Function URL Info
-

General configuration Info

Description
-

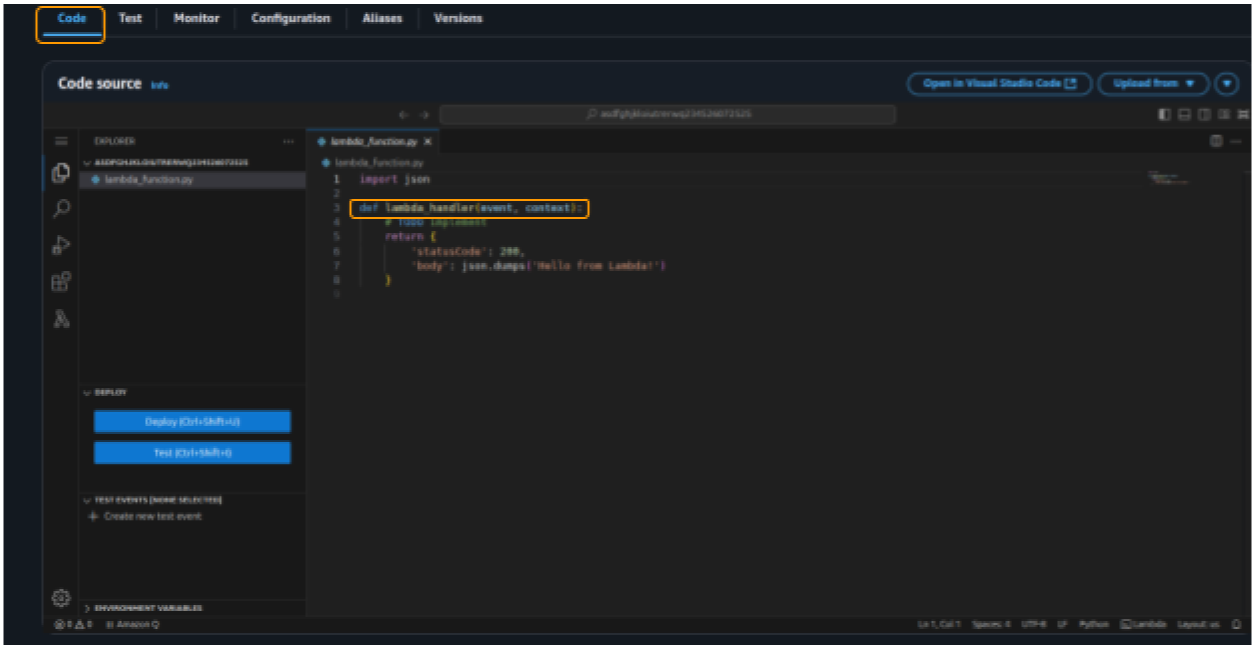
Memory
128 MB

Ephemeral storage
512 MB

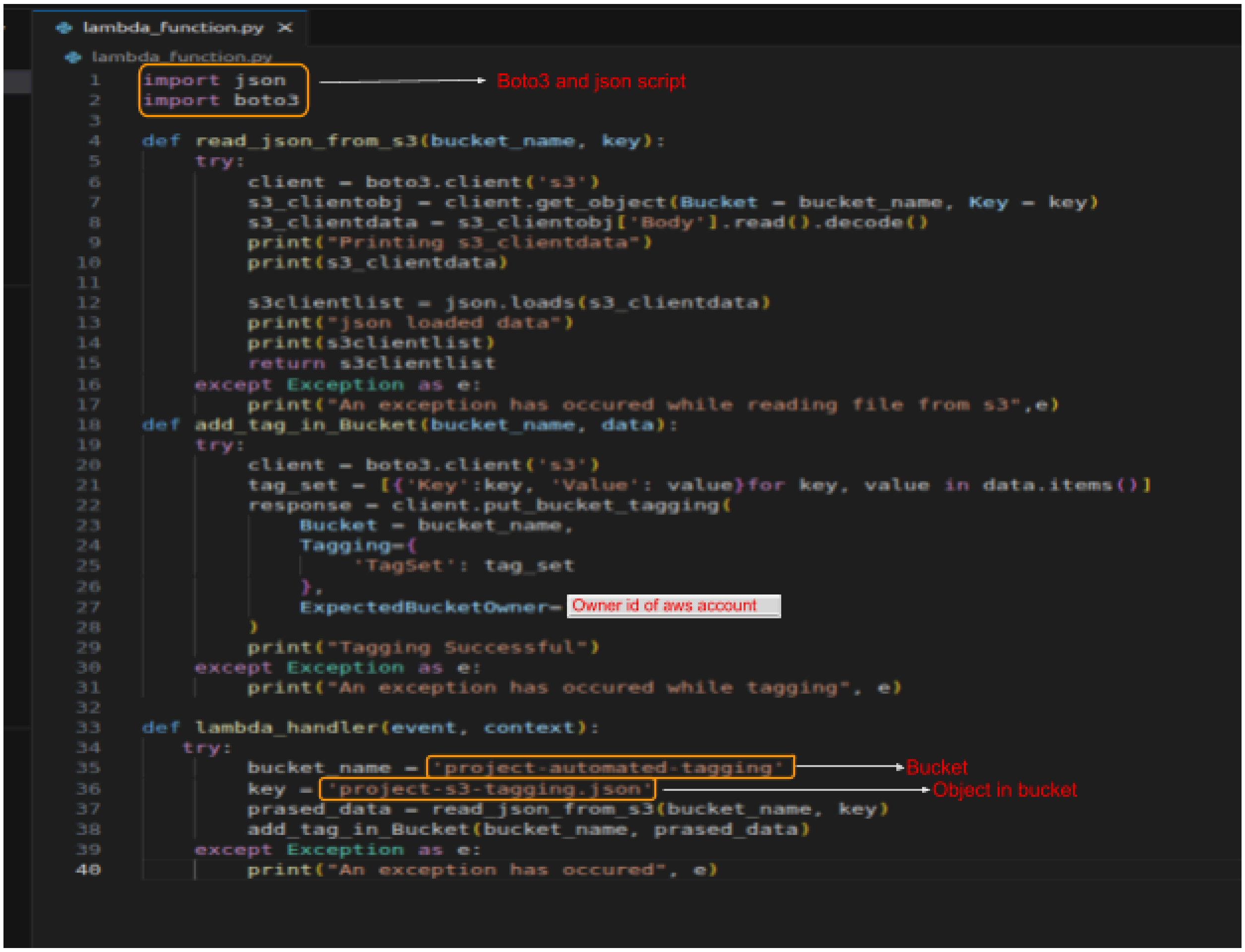
Timeout
0 min 3 sec

SnapStart Info
None

- Write the python and boto3 script in lambda handler

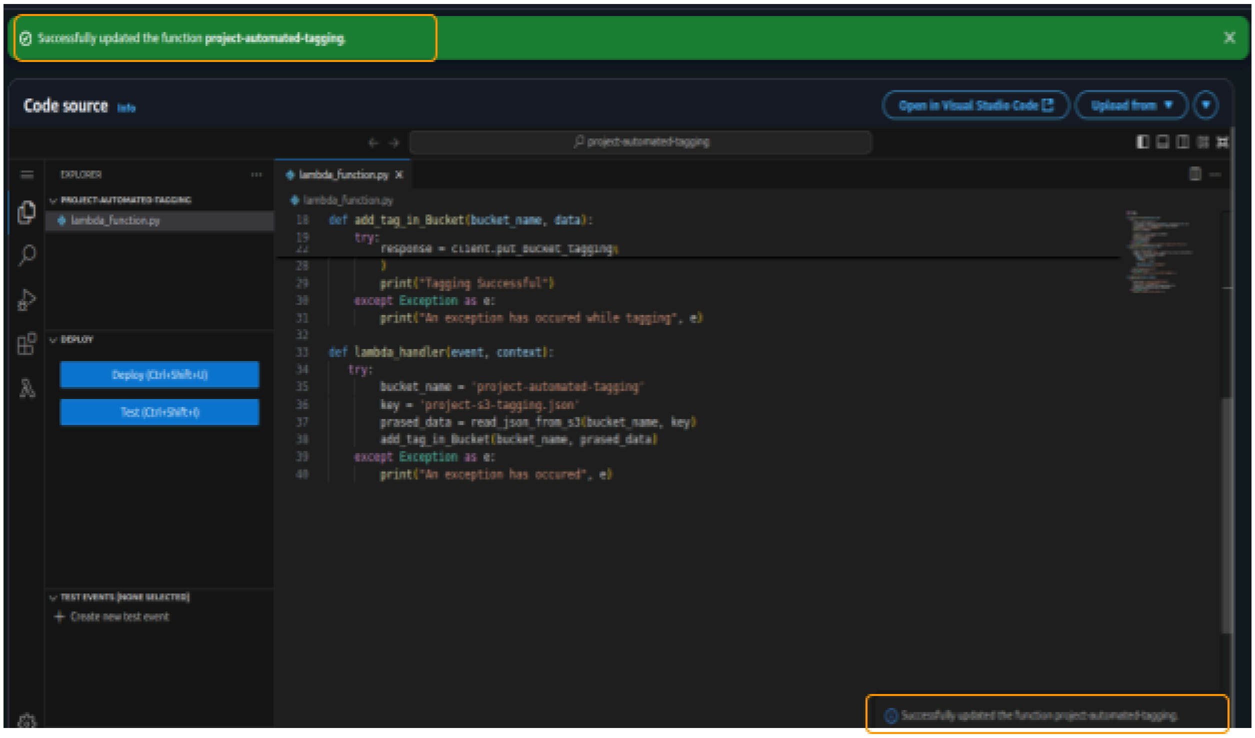


```python
import json

def lambda_handler(event, context):
    # TODO implement
    return {
        'statusCode': 200,
        'body': json.dumps('Hello from Lambda!')
    }
```



```python
import json
import boto3

def read_json_from_s3(bucket_name, key):
    try:
        client = boto3.client('s3')
        s3_clientobj = client.get_object(Bucket = bucket_name, Key = key)
        s3_clientdata = s3_clientobj['Body'].read().decode()
        print("Printing s3_clientdata")
        print(s3_clientdata)

        s3clientlist = json.loads(s3_clientdata)
        print("json loaded data")
        print(s3clientlist)
        return s3clientlist
    except Exception as e:
        print("An exception has occured while reading file from s3",e)
def add_tag_in_Bucket(bucket_name, data):
    try:
        client = boto3.client('s3')
        tag_set = [{'Key':key, 'Value': value}for key, value in data.items()]
        response = client.put_bucket_tagging(
            Bucket = bucket_name,
            Tagging={
                'TagSet': tag_set
            },
            ExpectedBucketOwner= Owner id of aws account
        )
        print("Tagging Successful")
    except Exception as e:
        print("An exception has occured while tagging", e)

def lambda_handler(event, context):
    try:
        bucket_name = 'project-automated-tagging'
        key = 'project-s3-tagging.json'
        prased_data = read_json_from_s3(bucket_name, key)
        add_tag_in_Bucket(bucket_name, prased_data)
    except Exception as e:
        print("An exception has occured", e)
```

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



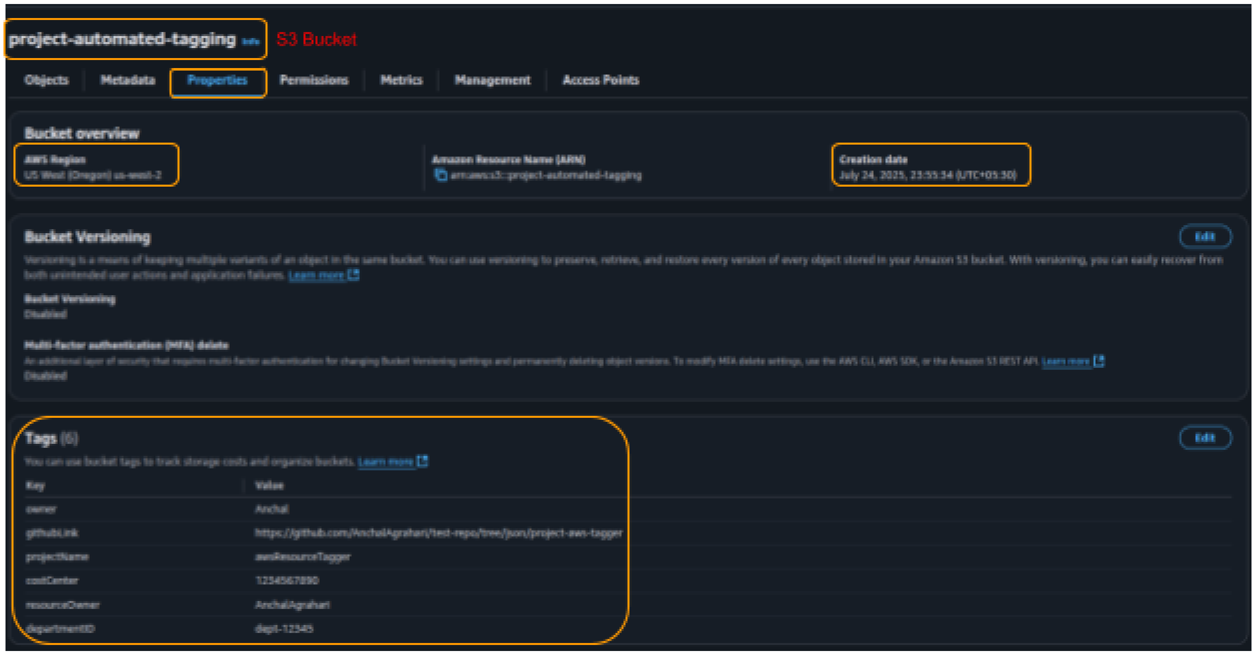
Successfully updated the function **project-automated-tagging**.

**Code source** Info

```python
def add_tag_in_Bucket(bucket_name, data):
    try:
        response = client.put_bucket_tagging(
        )
        print("Tagging Successful")
    except Exception as e:
        print("An exception has occured while tagging", e)

def lambda_handler(event, context):
   try:
        bucket_name = 'project-automated-tagging'
        key = 'project-s3-tagging.json'
        prased_data = read_json_from_s3(bucket_name, key)
        add_tag_in_Bucket(bucket_name, prased_data)
    except Exception as e:
        print("An exception has occured", e)
```

Successfully updated the function project-automated-tagging

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



**project-automated-tagging** Info S3 Bucket

**Bucket overview**

**AWS Region**
US West (Oregon) us-west-2

**Amazon Resource Name (ARN)**
arn:aws:s3:::project-automated-tagging

**Creation date**
July 24, 2025, 23:55:34 (UTC+05:30)

**Bucket Versioning**

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. Learn more

**Bucket Versioning**
Disabled

**Multi-factor authentication (MFA) delete**
An additional layer of security that requires multi-factor authentication for changing Bucket Versioning settings and permanently deleting object versions. To modify MFA delete settings, use the AWS CLI, AWS SDK, or the Amazon S3 REST API. Learn more
Disabled

**Tags** (6)

You can use bucket tags to track storage costs and organize buckets. Learn more

| Key | Value |
|---|---|
| owner | Anchal |
| githubLink | https://github.com/AnchalAgrahari/test-repo/tree/json/project-aws-tagger |
| projectName | awsResourceTagger |
| costCenter | 1234567890 |
| resourceOwner | AnchalAgrahari |
| departmentID | dept-12345 |

## Source:

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