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?

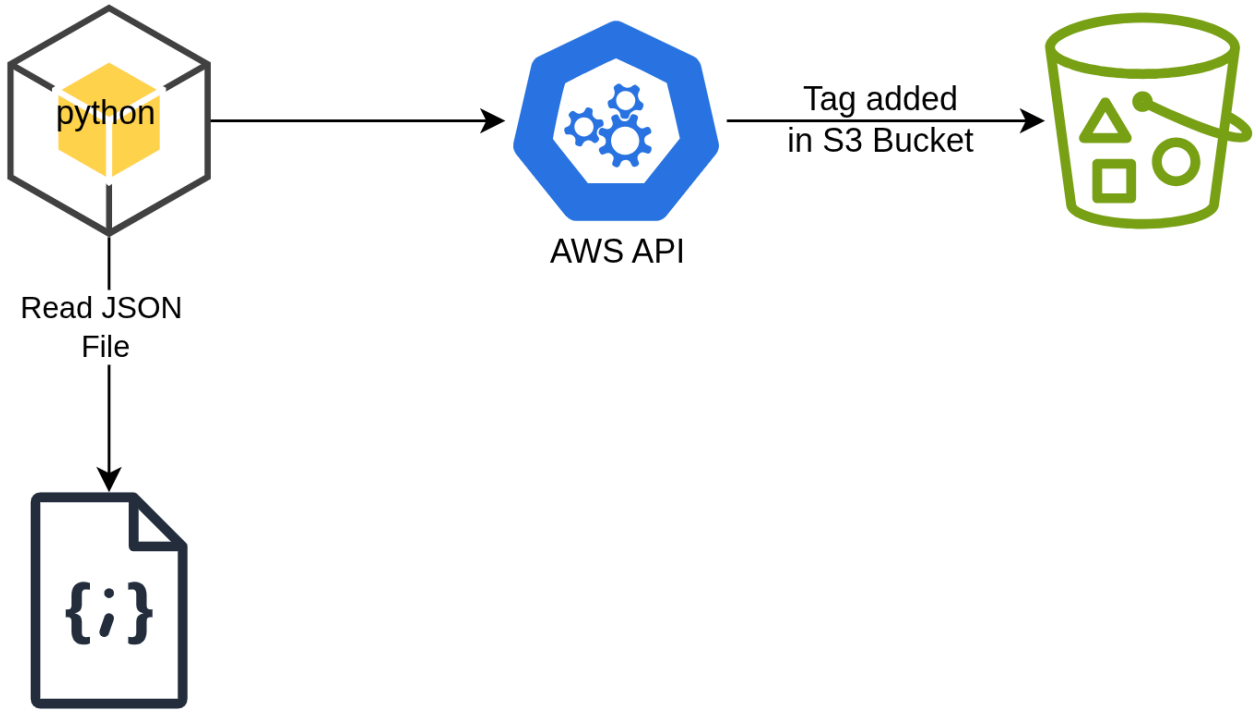
Prerequisites:
Before starting this project, ensure that the following are installed, set up, or understood.

Tools used:

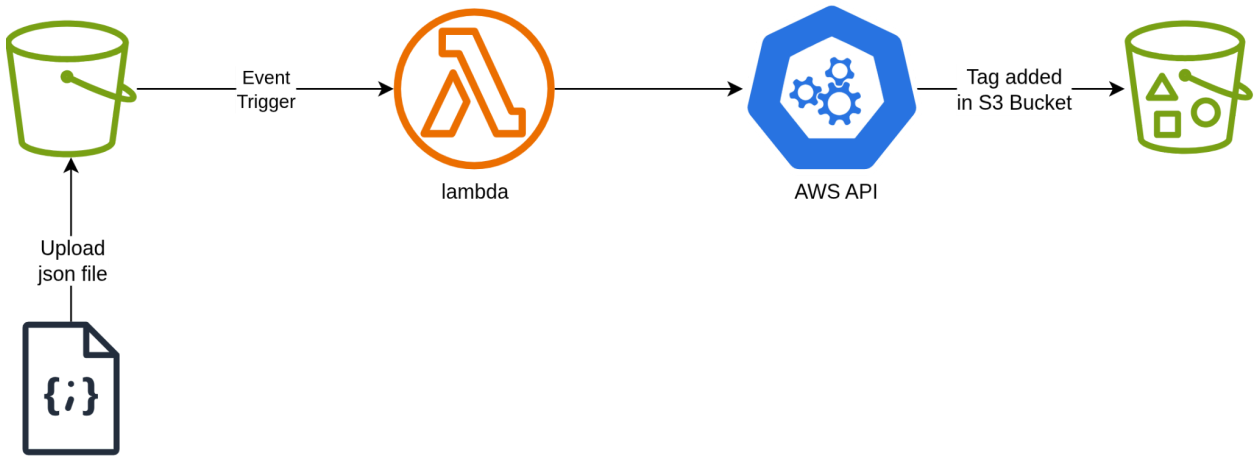
Tools	Installation link and source info
VS code	Code editor for the project
Python 3.12.3	Programming language used in the project
Github	To manage code and keep track of the project
AWS Boto3 SDK	for interacting with AWS services in Python
HTML	Supporting object
AWS account	To use AWS service like s3, 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:

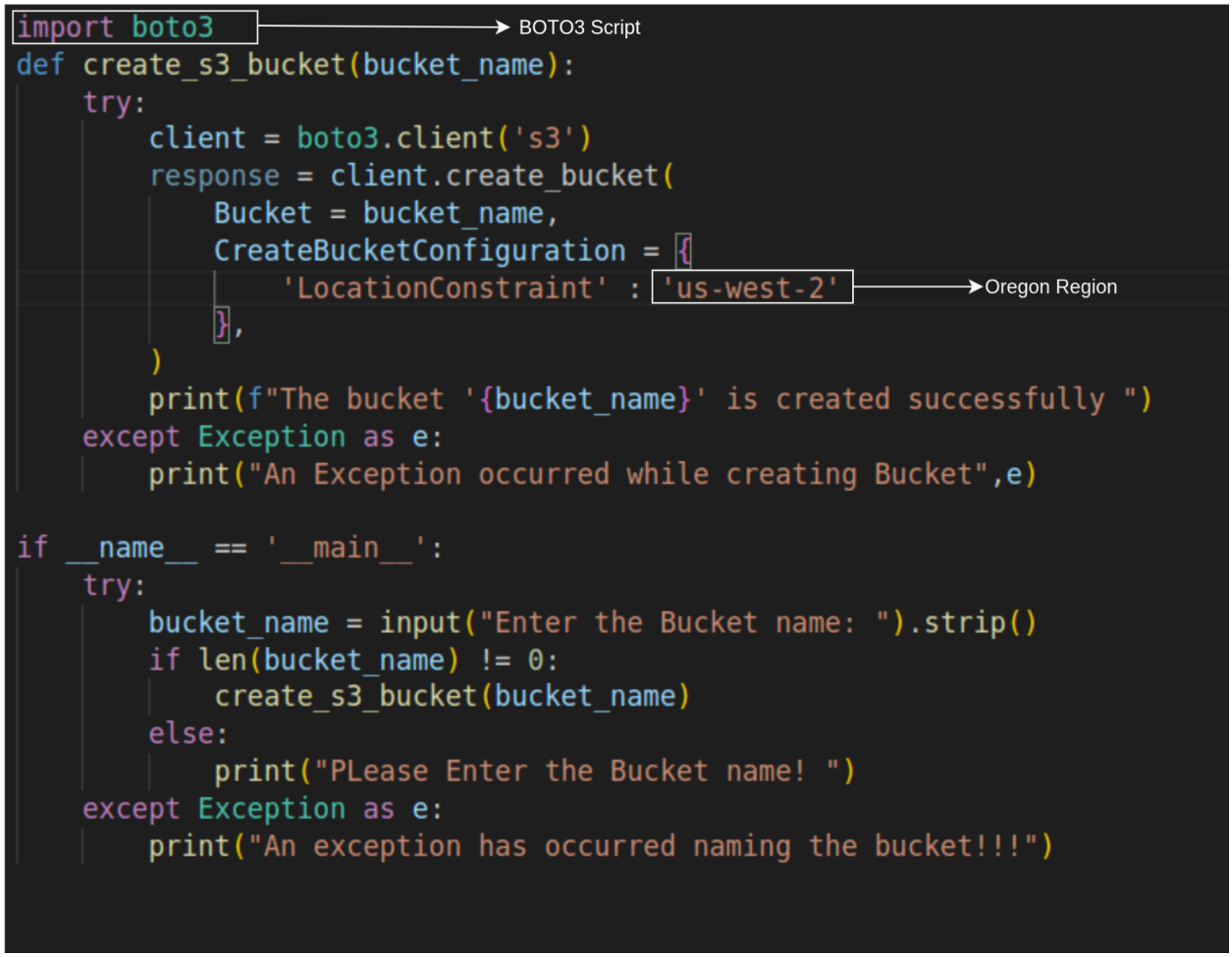


AWS flow of project:



Thought Process :

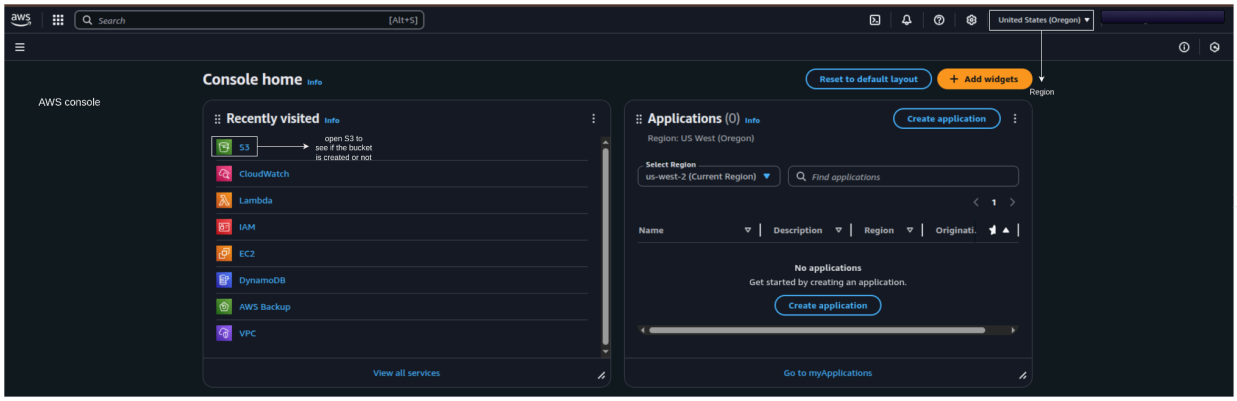
- First, write the code in VS code using Python, by installing the Boto3 library to create a S3 bucket.



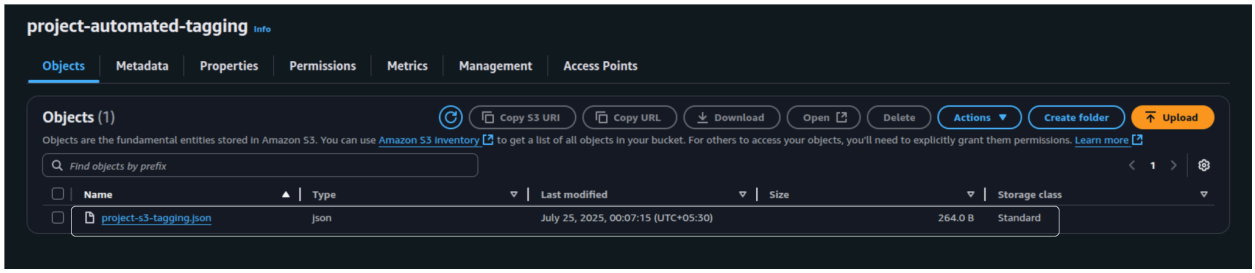
- Provide the unique bucket name
  Why Unique: S3 requires unique naming convention across the globe.

```
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
```

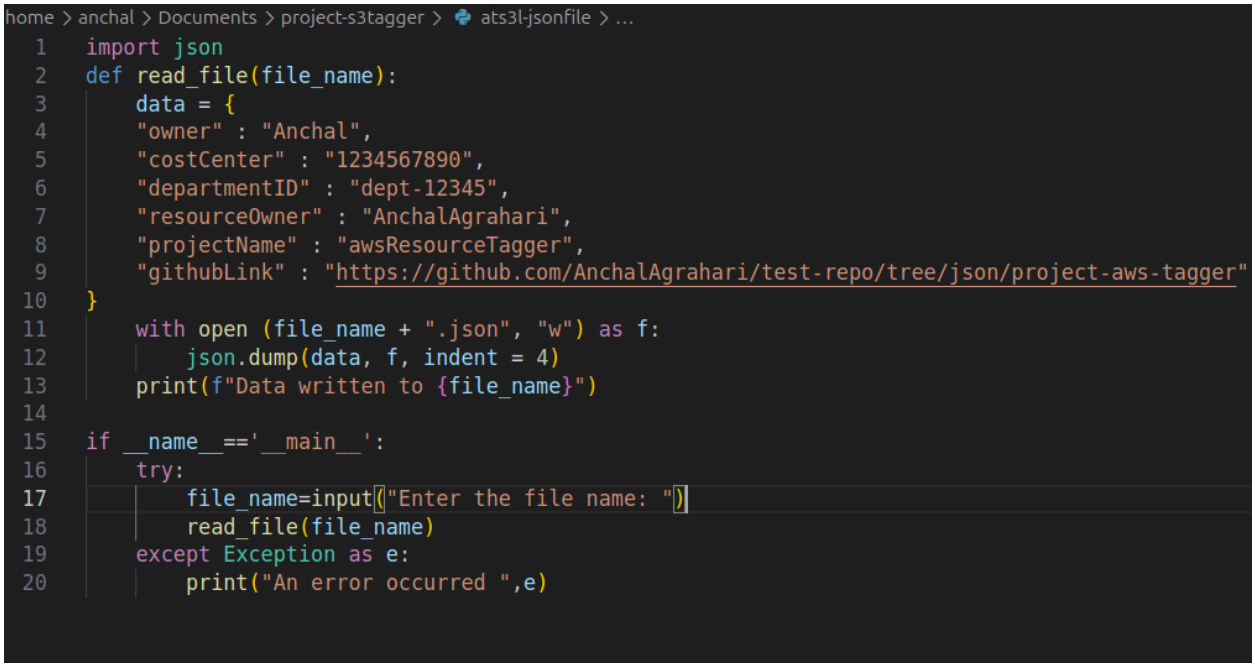
- Check if the bucket is created in AWS S3 or not .



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



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



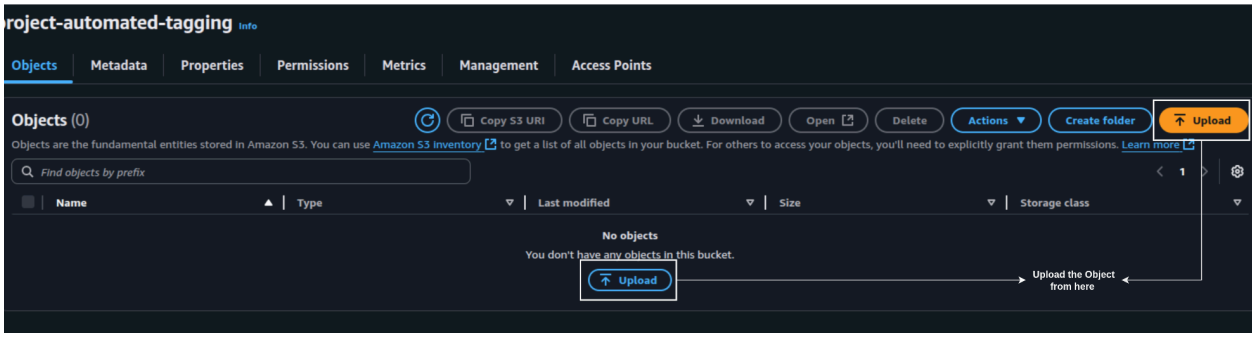
- After running the json code.

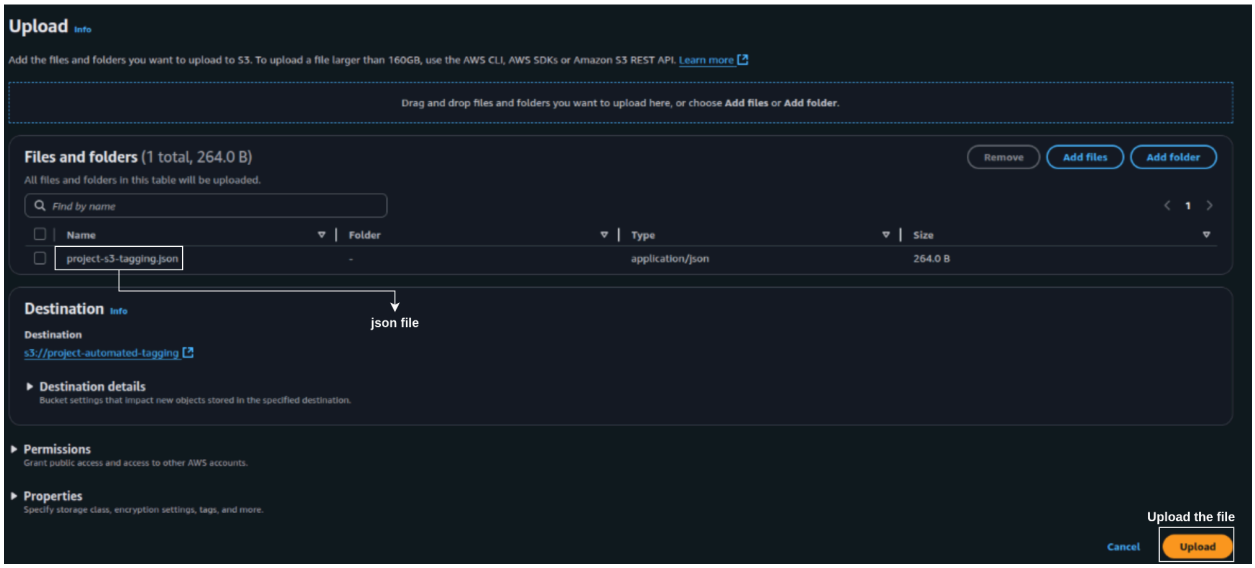
```
anchal@anchal-LOQ-15IAX9:~/Documents/project-s3tagger$ python3 ats3l-jsonfile
Enter the file name: project-s3-tagging
Data written to project-s3-tagging.json
```

- We get the json file.

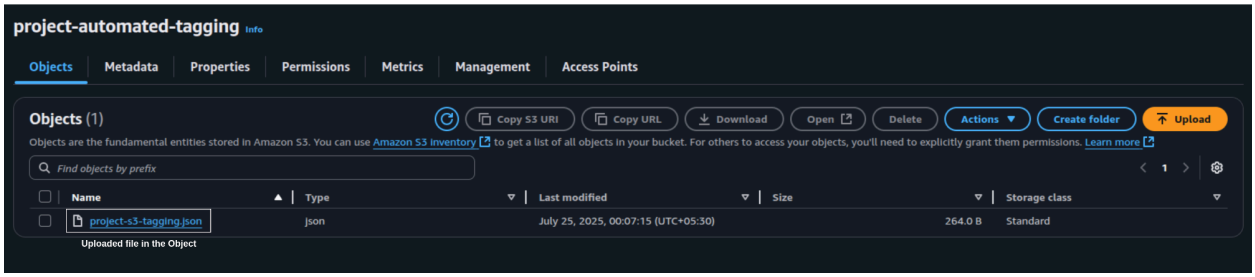
```
{
    "owner": "Anchal",
    "costCenter": "1234567890",
    "departmentID": "dept-12345",
    "resourceOwner": "AnchalAgrahari",
    "projectName": "awsResourceTagger",
    "githubLink": "https://github.com/AnchalAgrahari/test-repo/tree/json/project-aws-tagger"
}
```

- Upload it to an s3 bucket.

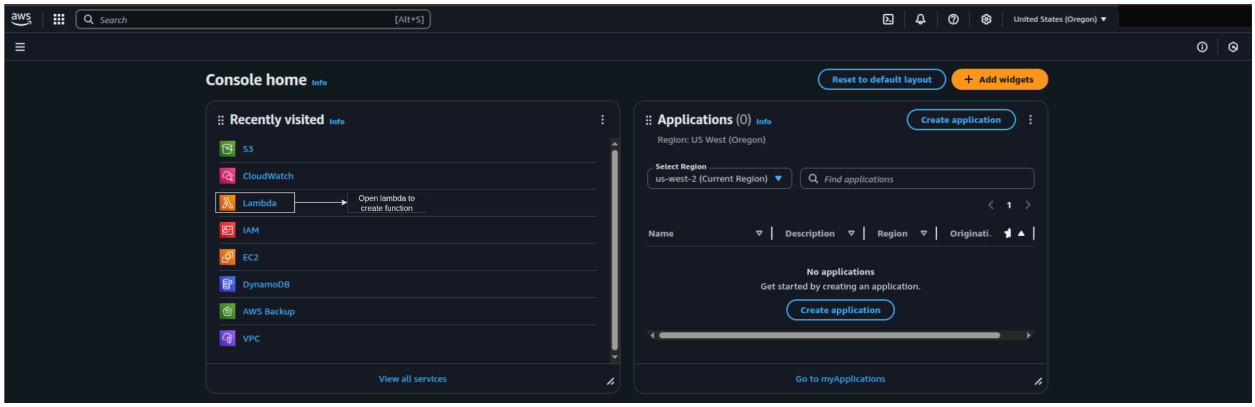


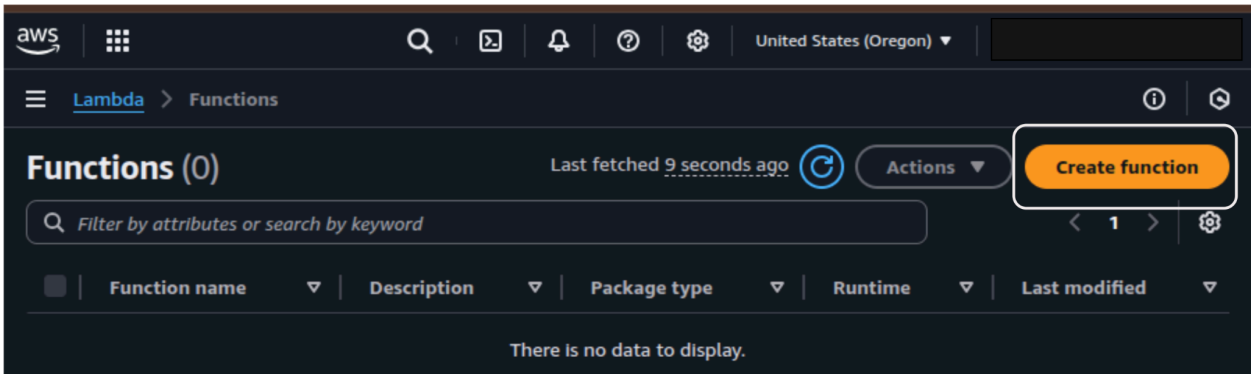


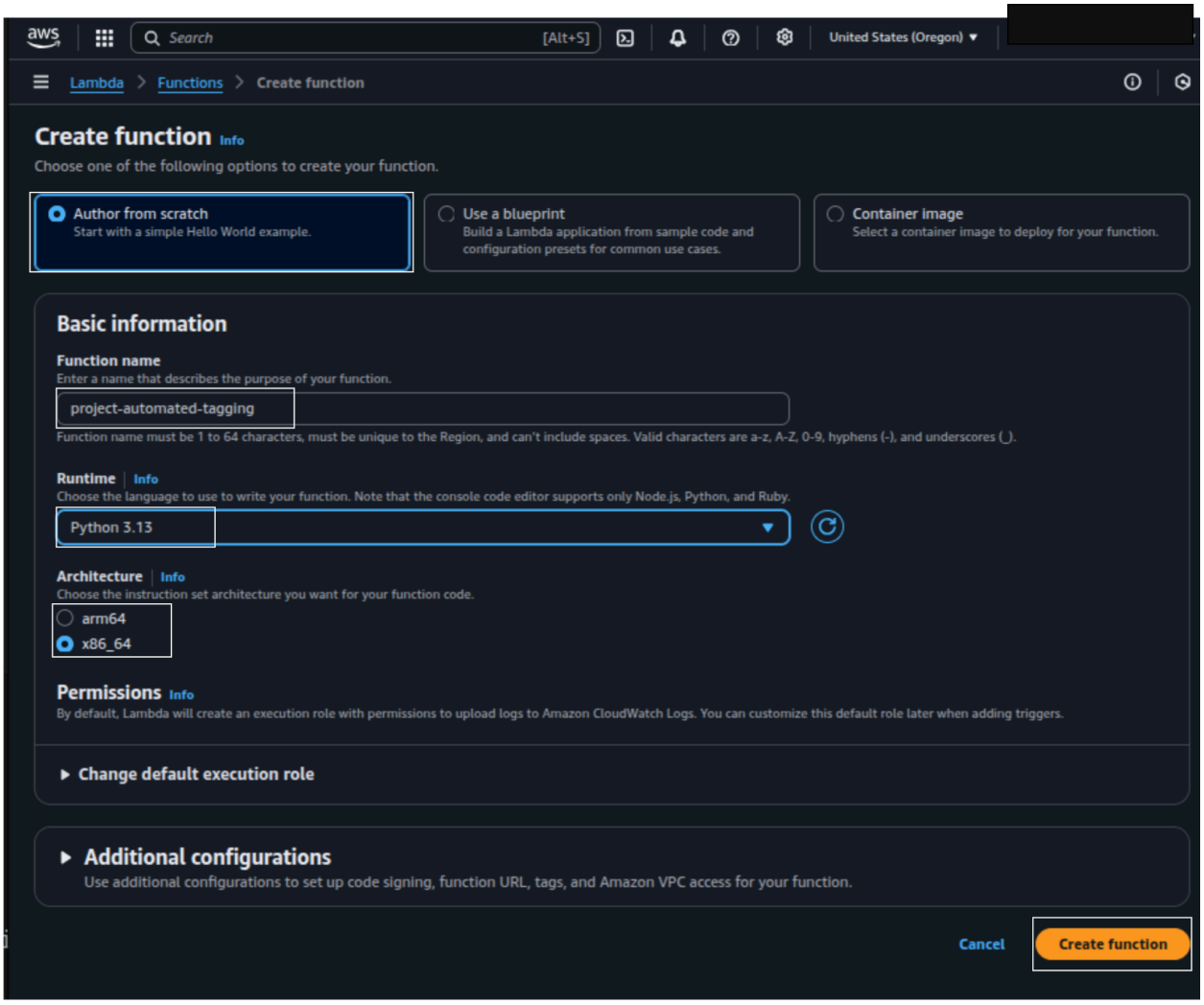
- Check if the Object is uploaded to the S3 bucket.

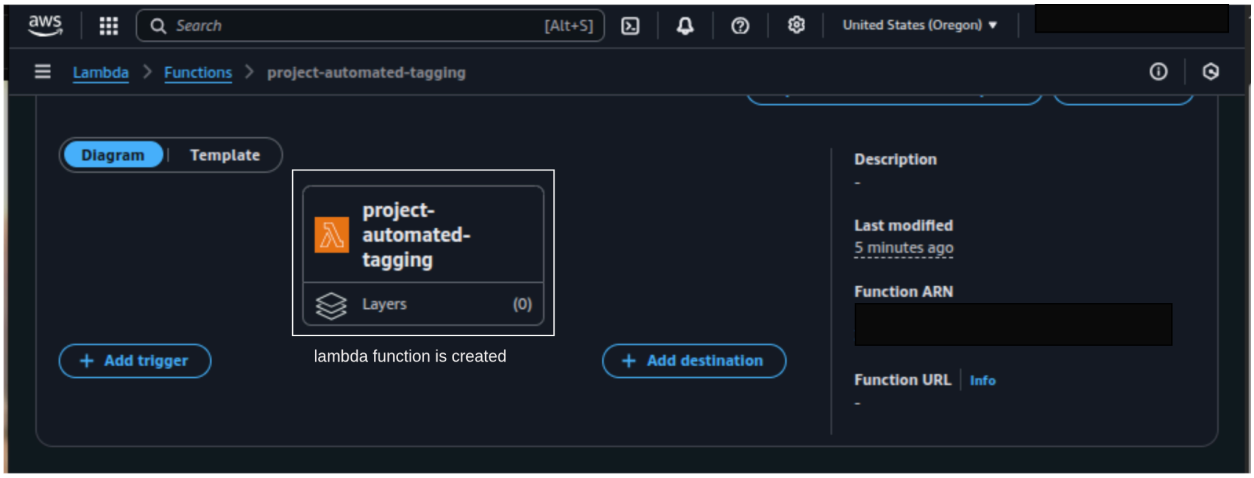


- Then create a Function in lambda



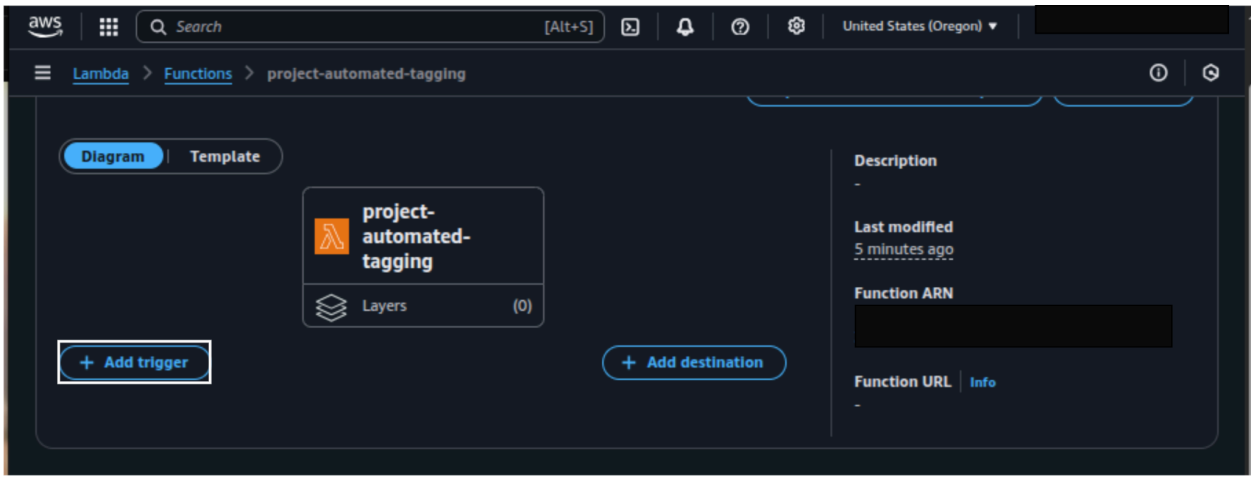

United States (Oregon)
Lambda > Functions
Functions (0)
Last fetched 9 seconds ago
Actions
Create function
Filter by attributes or search by keyword
1
Function name
Description
Package type
Runtime
Last modified
There is no data to display.

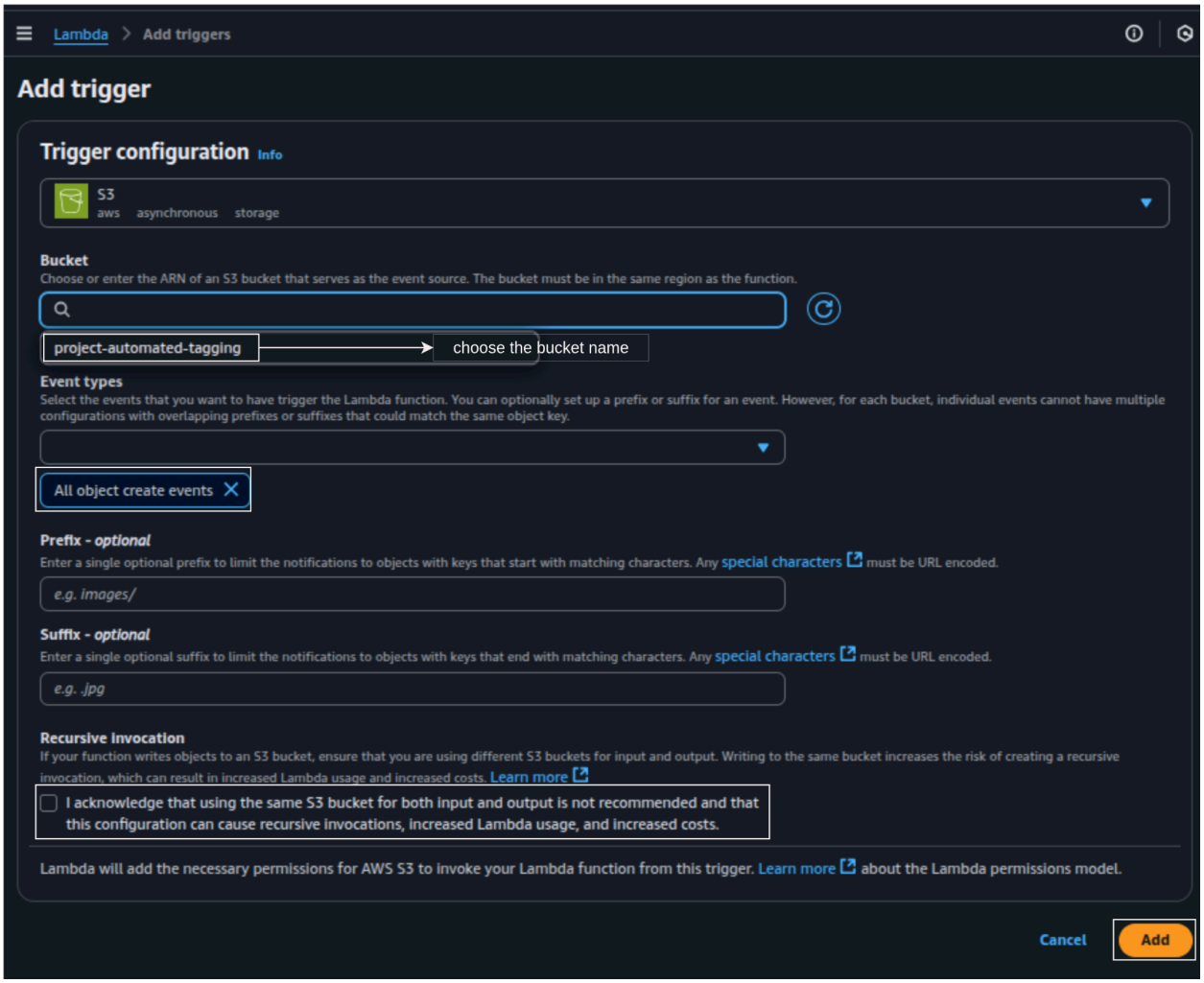

Search
[Alt+S]
United States (Oregon)
Lambda > Functions > Create function
Create function Info
Choose one of the following options to create your function.
Author from scratch
Start with a simple Hello World example.
Use a blueprint
Build a Lambda application from sample code and configuration presets 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.
Cancel
Create function

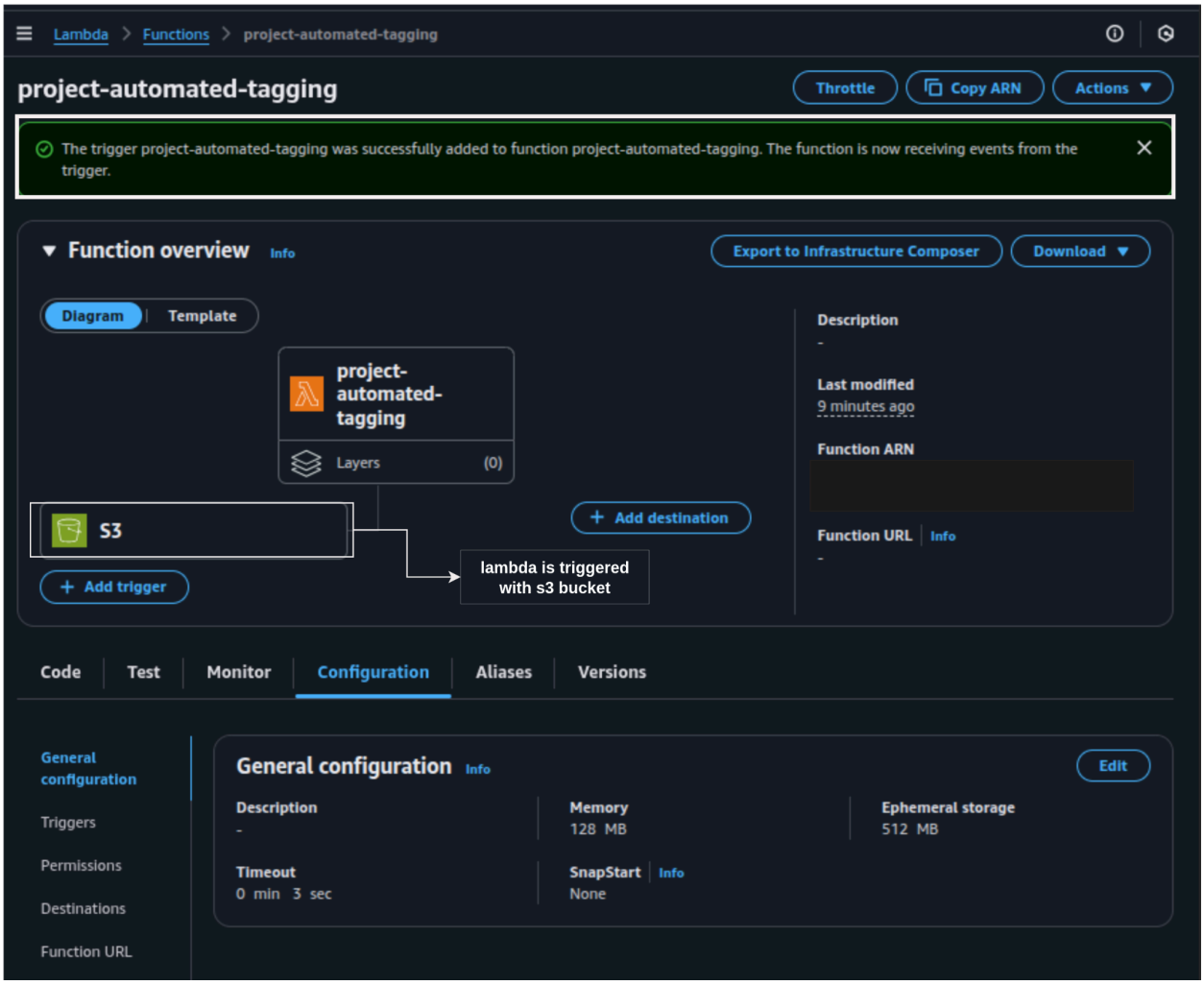


lambda function is created

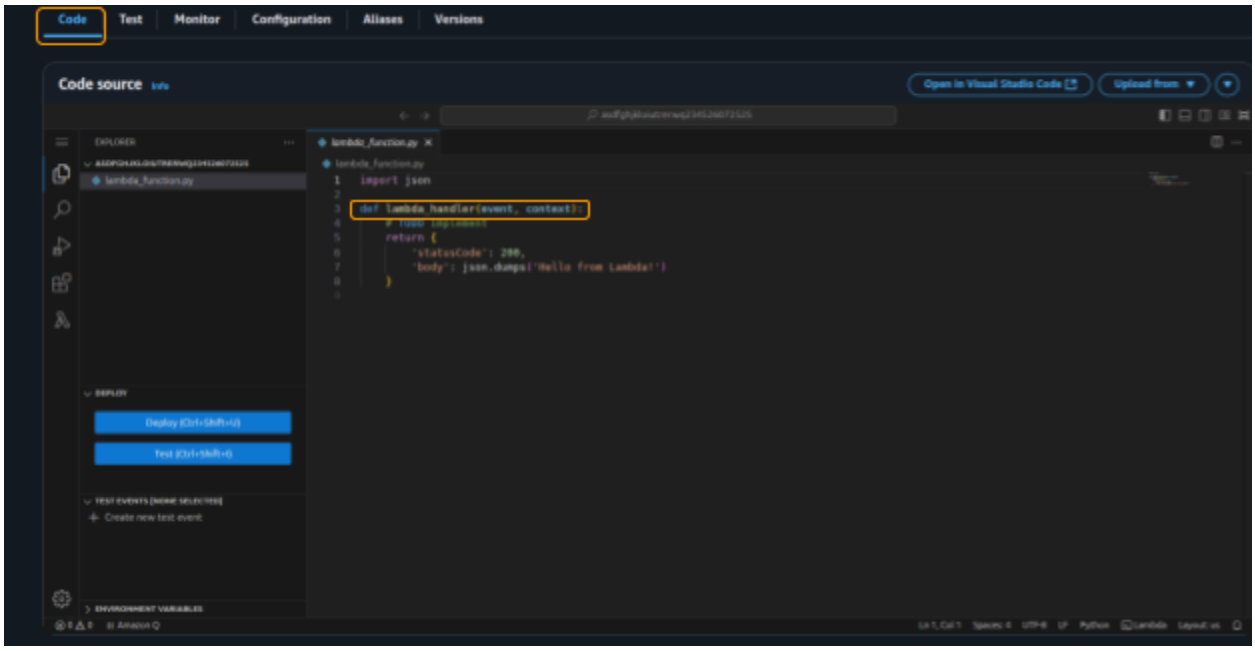
- Add a trigger to the lambda function using the S3 bucket.



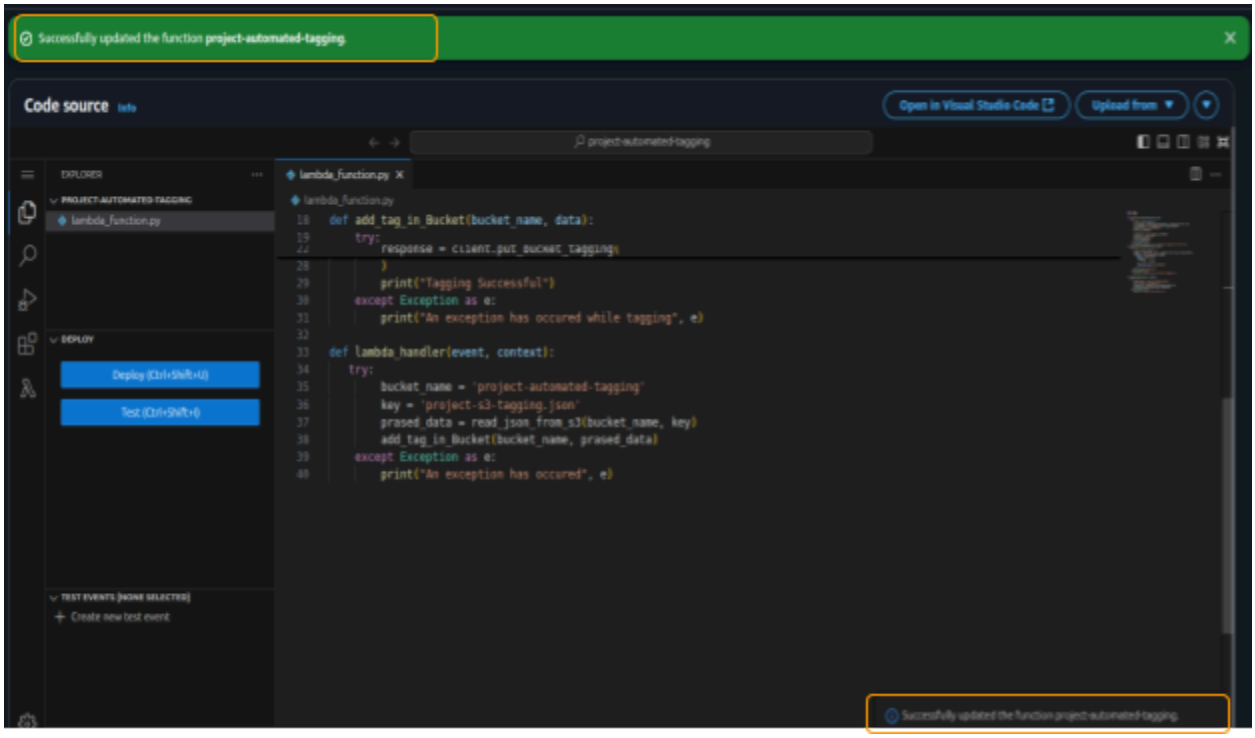

Lambda > Add triggers
Add trigger
Trigger configuration Info
S3
aws asynchronous storage
Bucket
Choose or enter the ARN of an S3 bucket that serves as the event source. The bucket must be in the same region as the function.
project-automated-tagging
choose the bucket name
Event types
Select the events 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



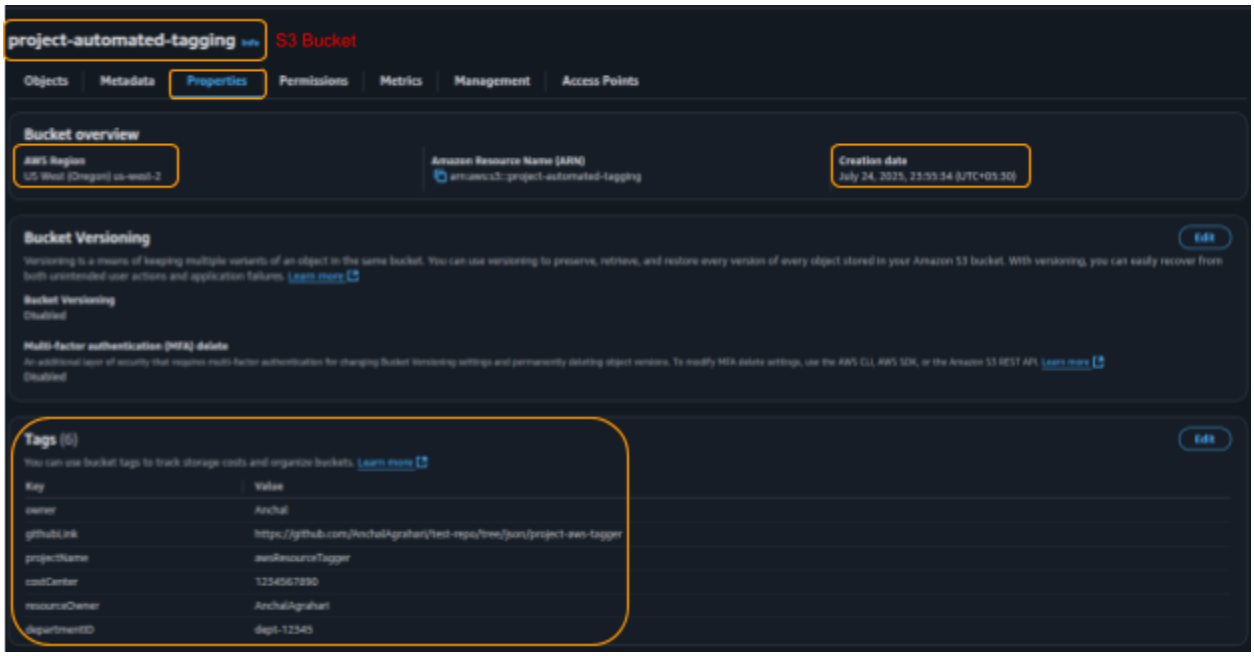
- Write the python and boto3 script in lambda handler


Code
Test
Monitor
Configuration
Aliases
Versions
Code source
Open in Visual Studio Code
Upload from
lambda_function.py
import json
def lambda_handler(event, context):
return {
'statusCode': 200,
'body': json.dumps('Hello from Lambda!')
}
Deploy (Ctrl+Shift+U)
Test (Ctrl+Shift+I)
Create new test event

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



- 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	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