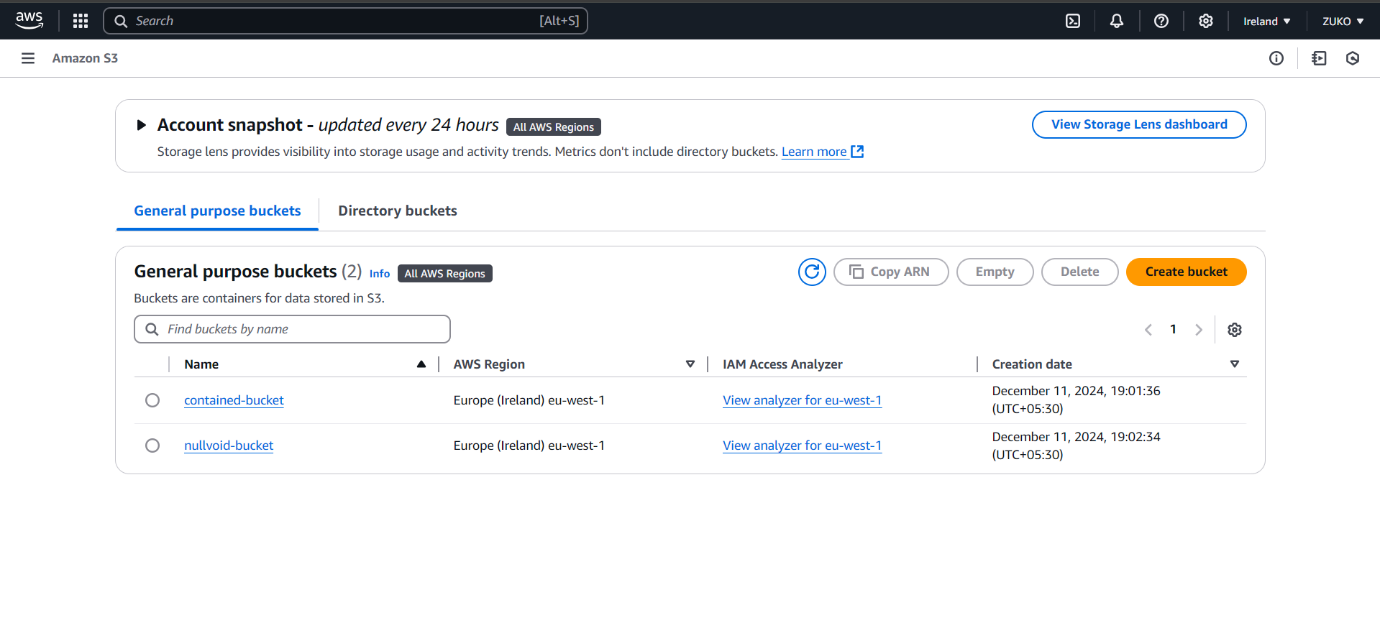
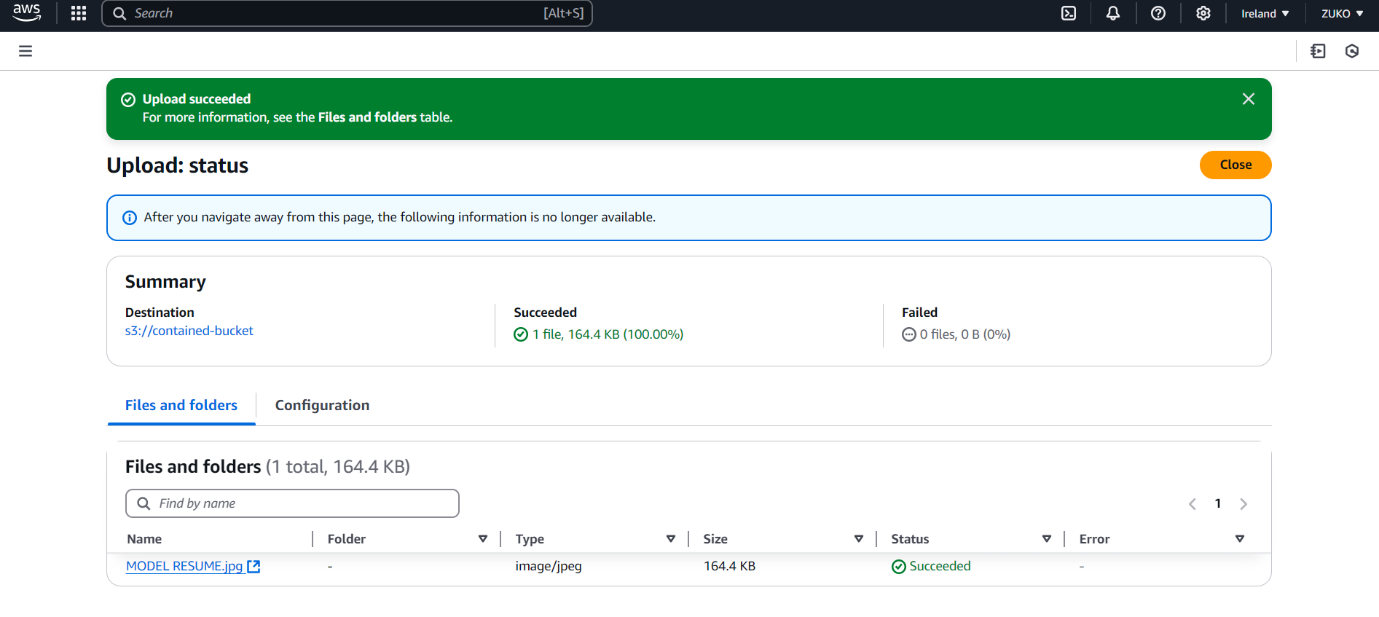
**NANO PROJECT- S3 COST OPTIMIZATION WITH AWS LAMBDA**

This runbook demonstrates how to create an AWS Lambda function to automatically delete empty S3 buckets for cost optimization. The Lambda function checks for empty S3 buckets and deletes them to save on storage costs

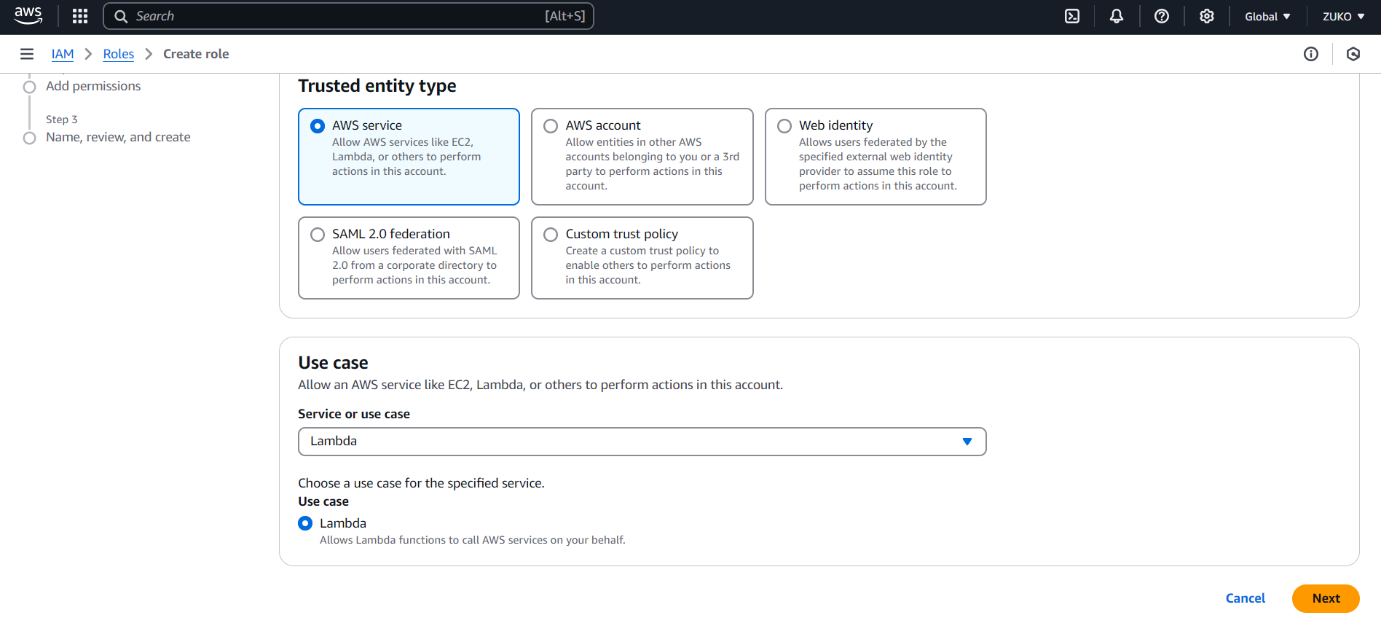
**CREATE TWO BUCKETS**



**UPLOAD AN OBJECT INTO ONE BUCKET**



**CREATE IAM ROLE LAMBDA AS USE CASE**



**ADD PERMISSIONS BY CREATING INLINE POLICY (JSON)**

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Action": [

"s3:ListBucket",

"s3:GetBucketLocation",

"s3:CreateBucket",

"s3:DeleteBucket",

"s3:PutObject",

"s3:GetObject",

"s3:DeleteObject",

"s3:PutLifecycleConfiguration",

"s3:ListBucket"

],

"Resource": [

"arn:aws:s3:::contained-bucket",

"arn:aws:s3:::contained-bucket/\*",

"arn:aws:s3:::nullvoid-bucket",

"arn:aws:s3:::nullvoid-bucket/\*"

]

},

{

"Effect": "Allow",

"Action": [

"logs:CreateLogGroup",

"logs:CreateLogStream",

"logs:PutLogEvents"

],

"Resource": "arn:aws:logs:\*:\*:log-group:/aws/lambda/\*"

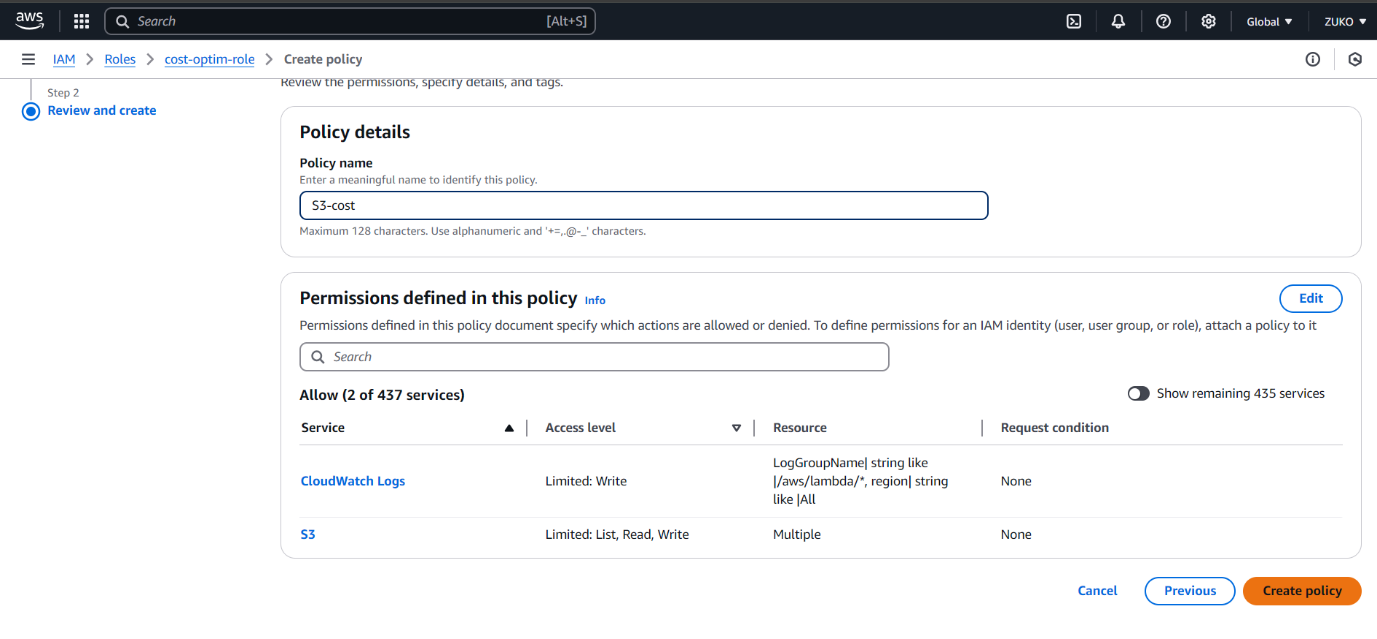
}

]

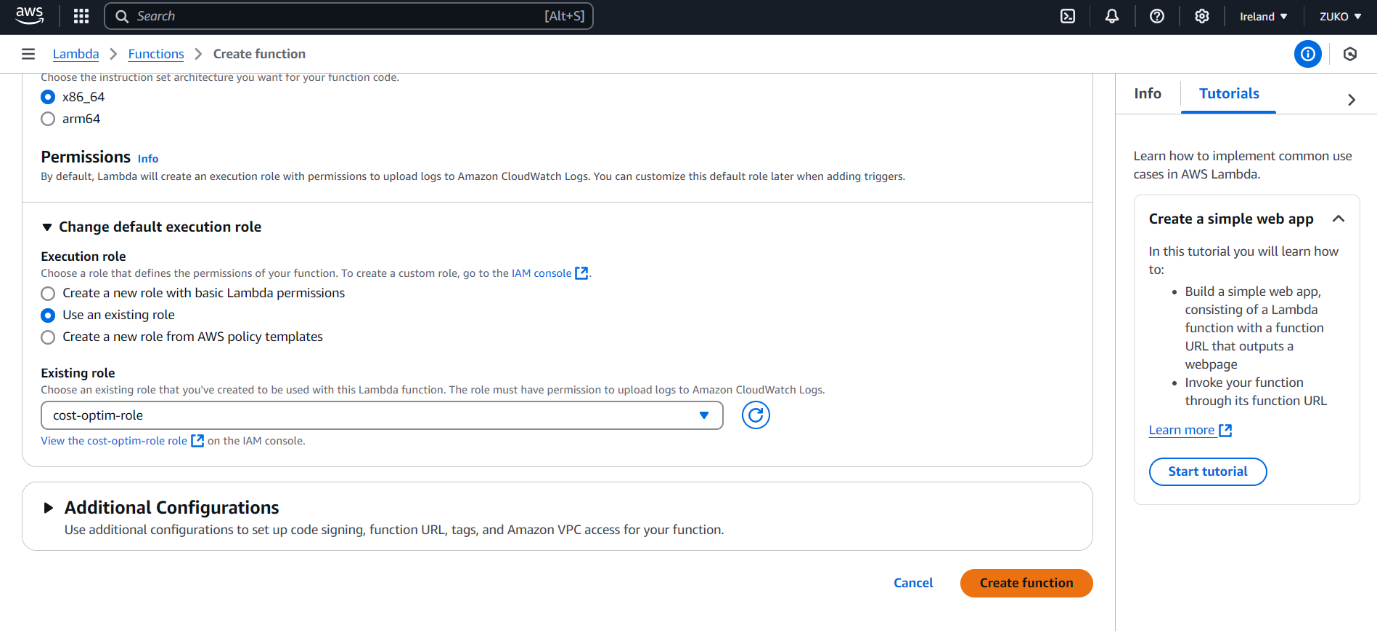
}

**Permissions Explanation:**

* s3:ListBucket: List objects in a bucket to check if it's empty.
* s3:DeleteBucket: Permission to delete the bucket.
* **s3:PutObject: Add an object to the bucket (if needed).**
* logs:CreateLogGroup, logs:PutlogEvents: For logging Lambda execution.



**CREATE LAMBDA FUNCTION CHOOSE IAM ROLE**



**PYTHON CODE**

import boto3

import logging

# Initialize the S3 client

s3 = boto3.client('s3')

def lambda\_handler(event, context):

    # Bucket names

    bucket\_names = ['contained-bucket', 'nullvoid-bucket']

    # Iterate over each bucket to check if it is empty

    for bucket in bucket\_names:

        try:

            # List the objects in the bucket to check if it is empty

            response = s3. list\_objects\_v2(Bucket=bucket)

            # If no objects are found in the bucket, delete the bucket

            if 'Contents' not in response:

                logging.info (f"The bucket {bucket} is empty. Deleting...")

                s3. delete\_bucket(Bucket=bucket)

                logging.info (f"Bucket {bucket} deleted successfully.")

            else:

                logging.info (f"The bucket {bucket} is not empty. Skipping deletion.")

        except Exception as e:

            logging.error(f"Error while processing bucket {bucket}: {e}")

            return {

                'statusCode': 500,

                'body': f"Error while processing bucket {bucket}: {str(e)}"

            }

    return {

        'statusCode': 200,

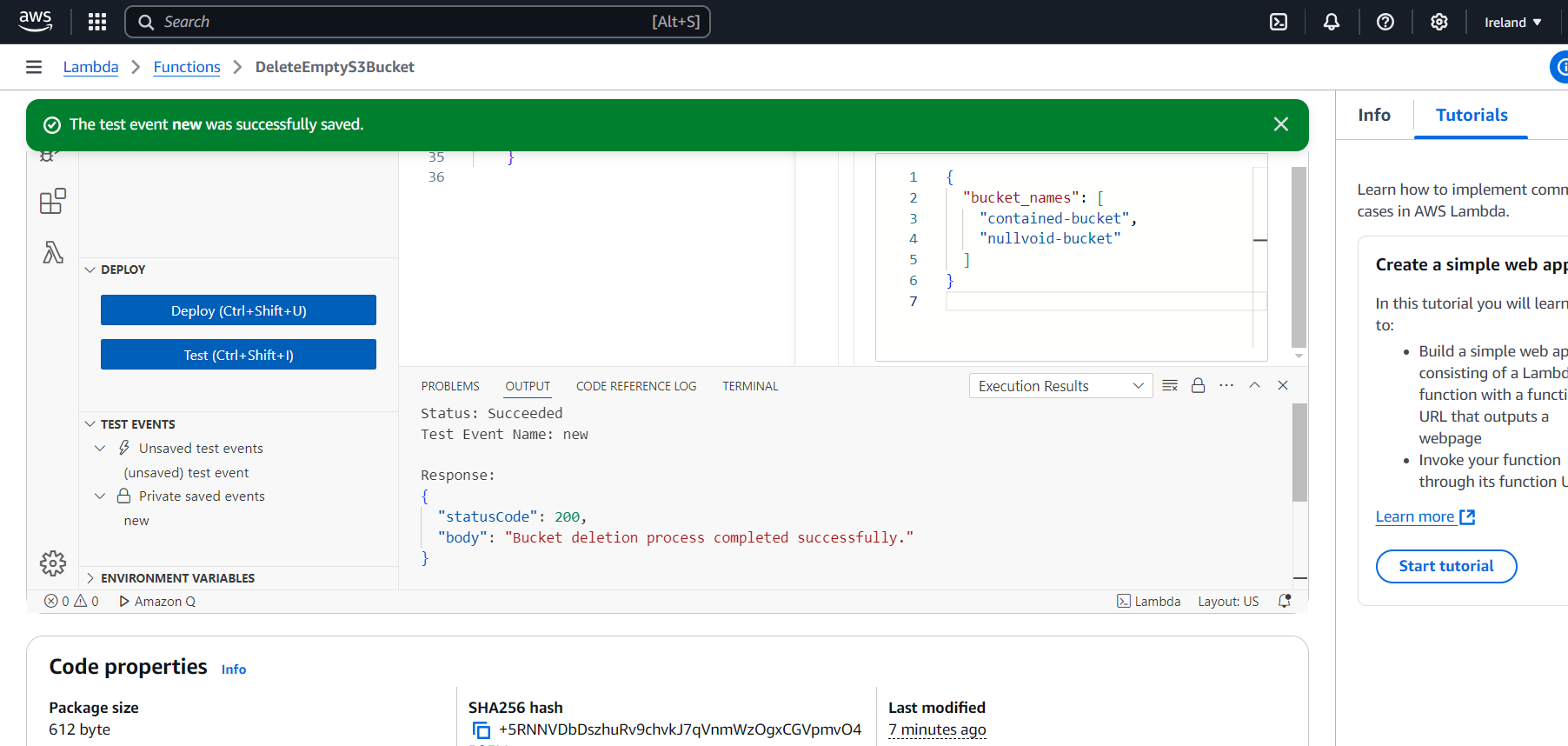
        'body': 'Bucket deletion process completed successfully.'

    }

**Code Explanation:**

* The lambda handler function is the entry point.
* It lists all the S3 buckets using the list buckets () method.
* It checks if the bucket contains any objects using list\_objects\_v2().
* If a bucket is empty, it is deleted using the delete bucket () method.
* It uses logging to capture the status of each operation and any errors.

**DEPLOY & TEST**



**STALE RESOURCE GOT DELETED.**

