AWS Lambda Function with S3 Interaction

Overview

This document provides step-by-step instructions to set up an AWS environment for a Lambda function that lists all objects in a specified S3 bucket. This setup includes creating an S3 bucket, an IAM role, and configuring the Lambda function.

Best Practices

Prefer IAM Roles Over Users: For most operations, especially those performed by automated processes or AWS services, it's recommended to use IAM roles instead of static IAM user credentials. IAM roles provide temporary security credentials to access your AWS resources, which is a more secure approach.

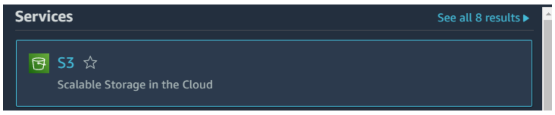
Use of IAM Users: If IAM roles cannot be used, create IAM users with permissions strictly limited to the necessary tasks. Avoid using the root user for day-to-day operations, as it has unrestricted access to all resources and services in the AWS account.

Least Privilege Principle: Whether using IAM roles or users, always adhere to the principle of least privilege by granting only the necessary permissions for a specific task or service

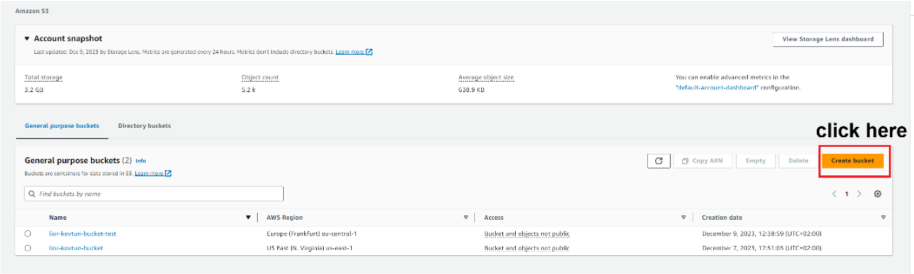
S3 Bucket Creation and Configuration

1. **Log in to AWS Management Console:** Ensure you're logged in with an IAM user with the necessary permissions.

2. **Open S3 Service:**
   - In the AWS Management Console, you can use the search bar at the top to quickly find the S3 service. Just type "S3" and click on the S3 result that appears.
   - Alternatively, you can find S3 under the "Services" menu.



3. **Create a New Bucket:** Once in the S3 dashboard, click on the "Create bucket" button.



Amazon S3

▾ Account snapshot

Last updated: Dec 9, 2023 by Storage Lens. Metrics are generated every 24 hours. Metrics don't include directory buckets. Learn more

View Storage Lens dashboard

Total storage	Object count	Average object size	You can enable advanced metrics in the "default-account-dashboard" configuration.
3.2 GB	5.2 k	638.9 KB	

General purpose buckets | Directory buckets

click here

General purpose buckets (2) Info

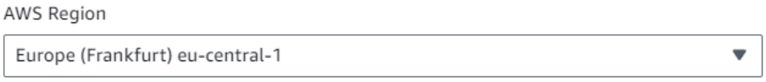
Buckets are containers for data stored in S3. Learn more

Copy ARN | Empty | Delete | Create bucket

Find buckets by name

	Name	AWS Region	Access	Creation date
○	lior-kovtun-bucket-test	Europe (Frankfurt) eu-central-1	Bucket and objects not public	December 9, 2023, 12:38:59 (UTC+02:00)
○	lior-kovtun-bucket	US East (N. Virginia) us-east-1	Bucket and objects not public	December 7, 2023, 17:51:05 (UTC+02:00)

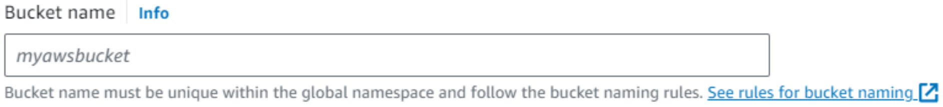
4. **Choose the AWS Region:**
   - Directly beneath the "Create bucket" header, you will find the "General configuration" section.
   - Locate the "AWS Region" dropdown menu.
   - Click on the dropdown and select an appropriate region. Choose a region that is geographically close to where the bucket's users will be, or one that meets specific data residency requirements for your project.



AWS Region

Europe (Frankfurt) eu-central-1 ▾

5. **Enter a Bucket Name:**
   - Right below the "AWS Region" dropdown is the "Bucket name" field.
   - Type in your desired bucket name. This name must be unique across all of Amazon S3 and conform to the bucket naming rules, such as no uppercase letters and no underscores.

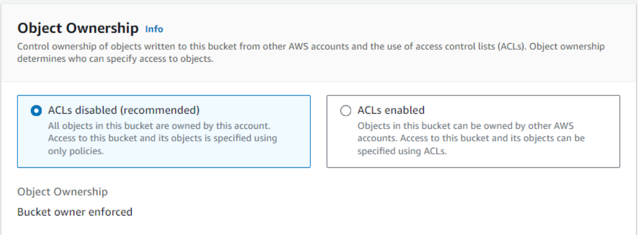


Bucket name Info

myawsbucket

Bucket name must be unique within the global namespace and follow the bucket naming rules. See rules for bucket naming

6. **Object Ownership:** The recommended setting is usually "ACLs disabled" which ensures that the bucket owner (you) retains ownership of uploaded objects.



Object Ownership Info

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

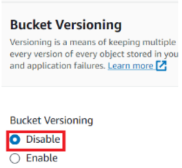
● ACLs disabled (recommended)
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

○ ACLs enabled
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Object Ownership
Bucket owner enforced

7. **Block Public Access Settings:** For security, it's recommended to keep the default setting which blocks all public access. Only uncheck these options if you are sure your bucket needs to be publicly accessible.
8. **Configure Bucket Versioning:**
   - **Bucket Versioning** helps preserve, retrieve, and restore every version of every object in your S3 bucket. It's crucial for data recovery.
   - Choose to "Enable" if you want to keep multiple variants of an object or "Disable" if this is not required for your use case.

     Current Setting: For the current setup, we have disabled versioning to simplify the process.

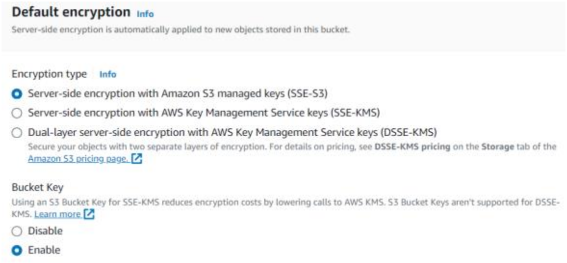


Bucket Versioning

Versioning is a means of keeping multiple
every version of every object stored in you
and application failures. Learn more

Bucket Versioning
● Disable
○ Enable

9. **Apply Tags (Optional):**
   - Tags help organize and track costs for your S3 bucket.
   - Click "Add tag" to create key-value pairs for the bucket. For example, you might add a tag with a key of "Environment" and a value of "Development".
10. **Set Default Encryption:**
    - **Default Encryption** adds a layer of security by encrypting objects at rest.
    - Select "Server-side encryption with Amazon S3 managed keys (SSE-S3)" for basic encryption needs.
    - For more advanced encryption options, select "Server-side encryption with AWS Key Management Service keys (SSE-KMS)" or "Dual-layer server-side encryption with AWS KMS (DSEE-KMS)" and specify the appropriate KMS key.
    - You can also choose to enable or disable a bucket key.
    - **Current Configuration:** We have set the bucket to use "Server-side encryption with Amazon S3 managed keys (SSE-S3)" for default encryption, balancing security with ease of management.



Default encryption Info

Server-side encryption is automatically applied to new objects stored in this bucket.

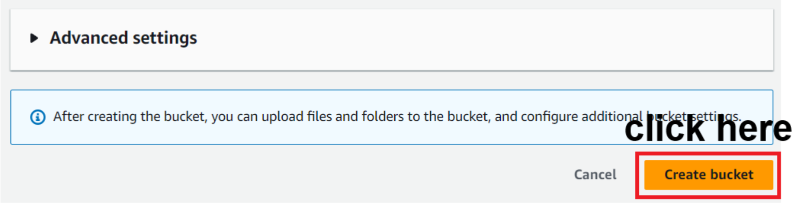
Encryption type Info
● Server-side encryption with Amazon S3 managed keys (SSE-S3)
○ Server-side encryption with AWS Key Management Service keys (SSE-KMS)
○ Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)
Secure your objects with two separate layers of encryption. For details on pricing, see **DSSE-KMS pricing** on the **Storage** tab of the Amazon S3 pricing page.

Bucket Key

Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSSE-KMS. Learn more

○ Disable
● Enable

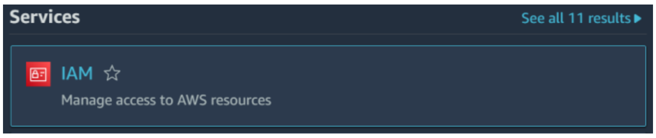
11. **Enable Object Lock (If applicable):**
    - **Object Lock** provides an additional layer of protection by preventing objects from being deleted or overwritten for a fixed amount of time or indefinitely.
    - This feature only works if versioning is enabled. You can enable it if you require immutability for compliance or data protection reasons.
    - **Current Configuration:** We have set the Object Lock to Disable.
12. **Finalize Bucket Creation:**
    - Review all the settings you have configured.
    - Once satisfied with the configuration, click the "Create bucket" button at the bottom of the page.



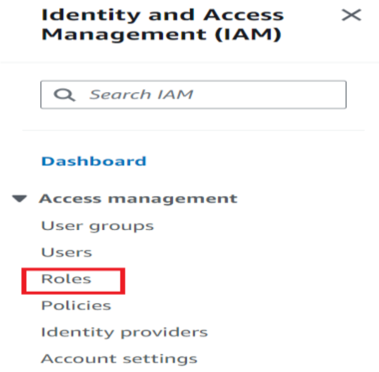
ⓘ After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.

Creating an IAM Role for AWS Lambda

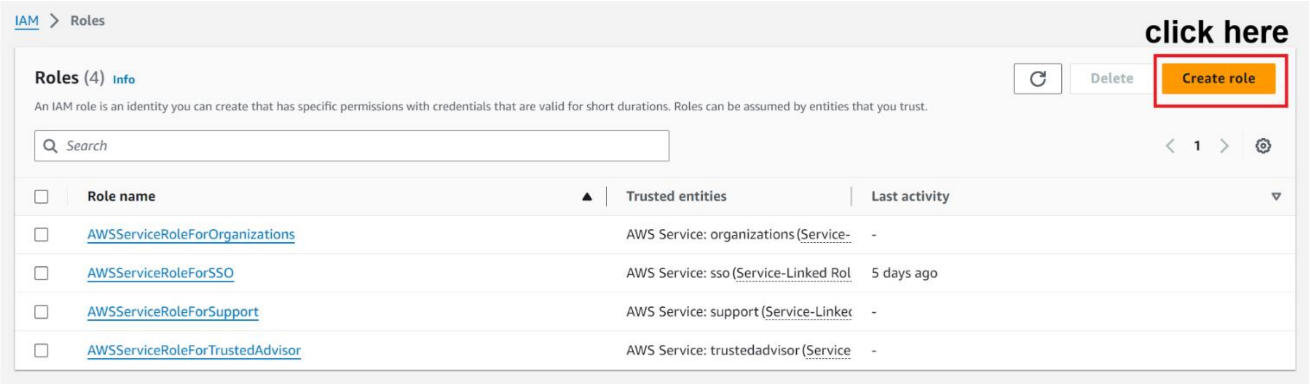
1. **Access IAM Management:** Navigate to the IAM service by typing "IAM" in the search bar or selecting it from the "Services" menu.



2. **Create a New IAM Role:**
    - In the IAM dashboard, click on "Roles" on the left-hand sidebar.



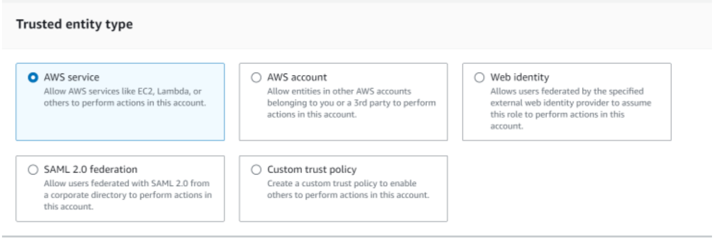
    - Click the "Create role" button.



An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

□	Role name ▲	Trusted entities	Last activity
□	AWSServiceRoleForOrganizations	AWS Service: organizations (Service-	-
□	AWSServiceRoleForSSO	AWS Service: sso (Service-Linked Rol	5 days ago
□	AWSServiceRoleForSupport	AWS Service: support (Service-Linkec	-
□	AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service	-

3. **Choose AWS Service as the Trusted Entity:**
   - On the "Create role" page, for the "Trusted entity type," select "AWS service."



Trusted entity type

● AWS service
Allow AWS services like EC2, Lambda, or others to perform actions in this account.

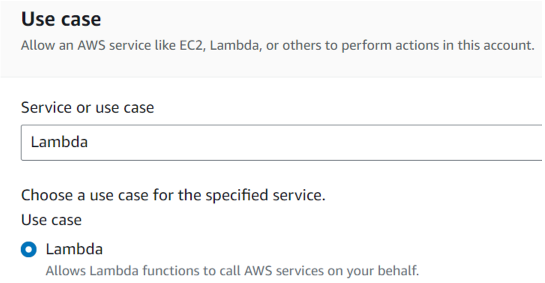
○ AWS account
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.

○ Web identity
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.

○ SAML 2.0 federation
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.

○ Custom trust policy
Create a custom trust policy to enable others to perform actions in this account.

   - For the "Use case," select "Lambda" to create a role for your Lambda function. This allows the Lambda service to assume the role.



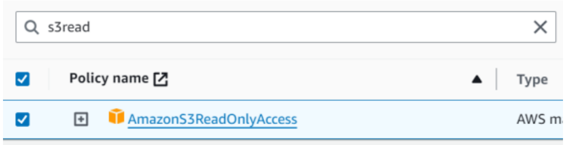
Use case
Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Service or use case
Lambda

Choose a use case for the specified service.
Use case
● Lambda
Allows Lambda functions to call AWS services on your behalf.

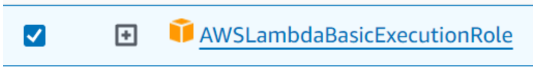
   - Click the “Next” button.

4. **Attach Permissions Policies:**
   - After selecting Lambda as the use case, you'll be directed to attach permission policies.
   - In the list of policies, find and tick the checkbox next to AmazonS3ReadOnlyAccess. This policy grants the role read-only permissions to S3 resources.
   - Since the Lambda function code utilizes logging, also select the AWSLambdaBasicExecutionRole policy. This grants the role permissions to write logs to Amazon CloudWatch, which is necessary for capturing log outputs from the function.



s3read

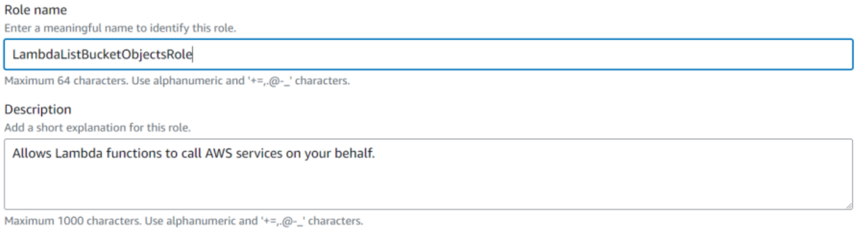
☑	Policy name	Type
☑	AmazonS3ReadOnlyAccess	AWS m



☑ AWSLambdaBasicExecutionRole

   - Click "Next" to proceed.

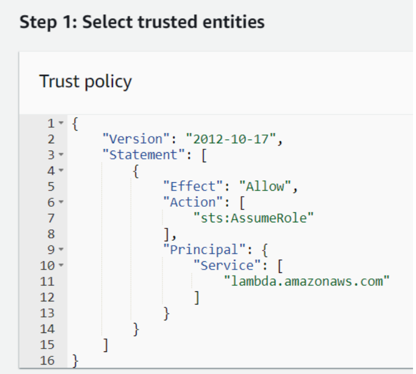
5. **Name, review, and create IAM Role:**
   - **Role Name** Enter a unique, descriptive name for the new role, such as ‘LambdaListBucketObjectsRole’.
   - **Role Description** It's helpful to provide a description that details the role's purpose, like "Role for Lambda function to list objects in S3."



Role name
Enter a meaningful name to identify this role.
LambdaListBucketObjectsRole
Maximum 64 characters. Use alphanumeric and '+=,.@-_' characters.

Description
Add a short explanation for this role.
Allows Lambda functions to call AWS services on your behalf.
Maximum 1000 characters. Use alphanumeric and '+=,.@-_' characters.

- **Review Trust Policy** Verify the trust policy is correct, which should look something like this:

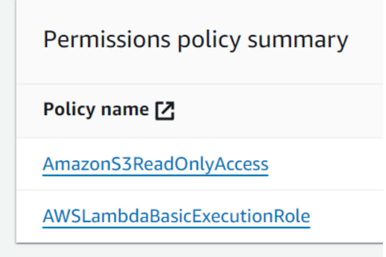


Step 1: Select trusted entities

Trust policy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRole"
            ],
            "Principal": {
                "Service": [
                    "lambda.amazonaws.com"
                ]
            }
        }
    ]
}
```

- **Review Permissions Policy Ensure** In the Permissions policy summary, ensure the following policies are listed:
  -AmazonS3ReadOnlyAccess for read-only access to S3 resources.
  -AWSLambdaBasicExecutionRole for Lambda logging to CloudWatch.



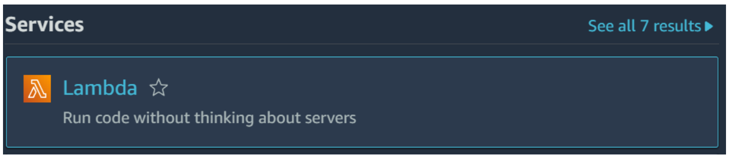
Permissions policy summary

Policy name
AmazonS3ReadOnlyAccess
AWSLambdaBasicExecutionRole

- **Create Role** After confirming the details are correct, click "Create role”.

Lambda Function Configuration and Deployment

1. **Access Lambda Service**
   - Log into the AWS Management Console with your IAM user credentials.
   - Open the Lambda service by typing "Lambda" in the search bar or selecting it from the "Services" menu.



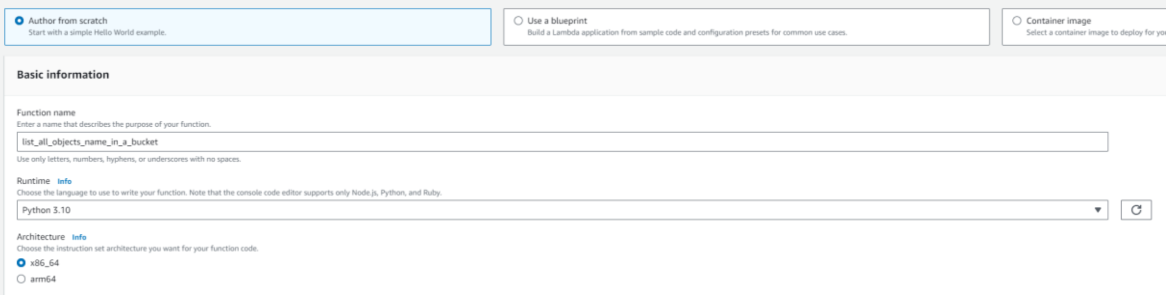
Services

See all 7 results ▸

Lambda ☆

Run code without thinking about servers

2. **Create a New Lambda Function**
   - Click on the “Create function” button.
   - Choose “Author from scratch.”
   - Enter a name for your function, for example, “list_all_objects_name_in_a_bucket”.
   - For "Runtime," choose the Python version you're using, like Python 3.10.



● 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 yo

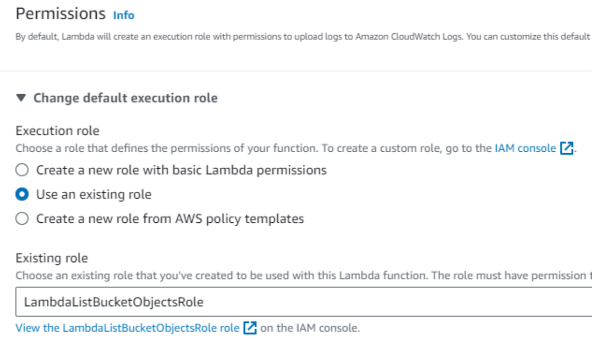
Basic information

Function name
Enter a name that describes the purpose of your function.
list_all_objects_name_in_a_bucket
Use only letters, numbers, hyphens, or underscores with no spaces.

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 3.10

Architecture Info
Choose the instruction set architecture you want for your function code.
● x86_64
○ arm64

3. **Assign the Execution Role**
   - In the "Permissions" section of the Lambda function creation process, select the option to "Use an existing role".
   - From the list of available roles, choose the role you created earlier, in our case it’s the ‘LambdaListBucketObjectsRole’



Permissions Info

By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default

▼ Change default execution role

Execution role

Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.

○ Create a new role with basic Lambda permissions

● Use an existing role

○ Create a new role from AWS policy templates

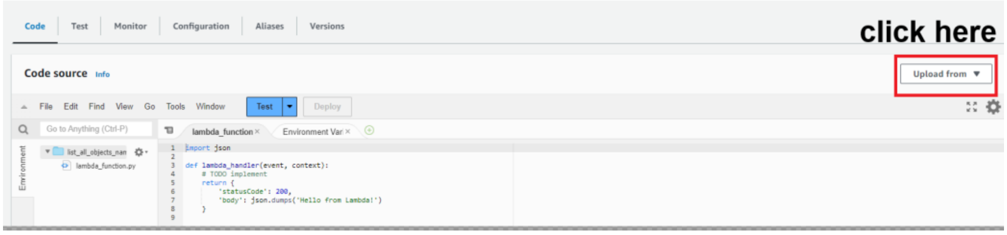
Existing role

Choose an existing role that you've created to be used with this Lambda function. The role must have permission t

LambdaListBucketObjectsRole

View the LambdaListBucketObjectsRole role on the IAM console.

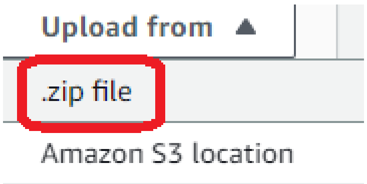
4. **Create the Function:** Once all settings are correctly configured, click on the "Create function" button at the bottom of the page.
5. **Uploading the lambda function:**
   - On the function's main page, you'll see the "Code source" section. This is where your Lambda function's code resides.
   - Above the code editor window, there is a button labeled "Upload from". Click on this button to reveal a dropdown menu.



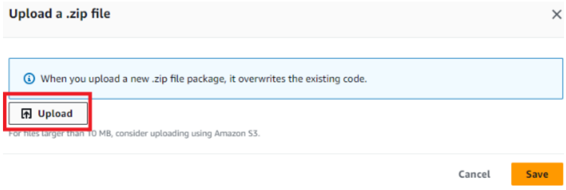
```
import json

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

   - From the dropdown menu, select “.zip file”.



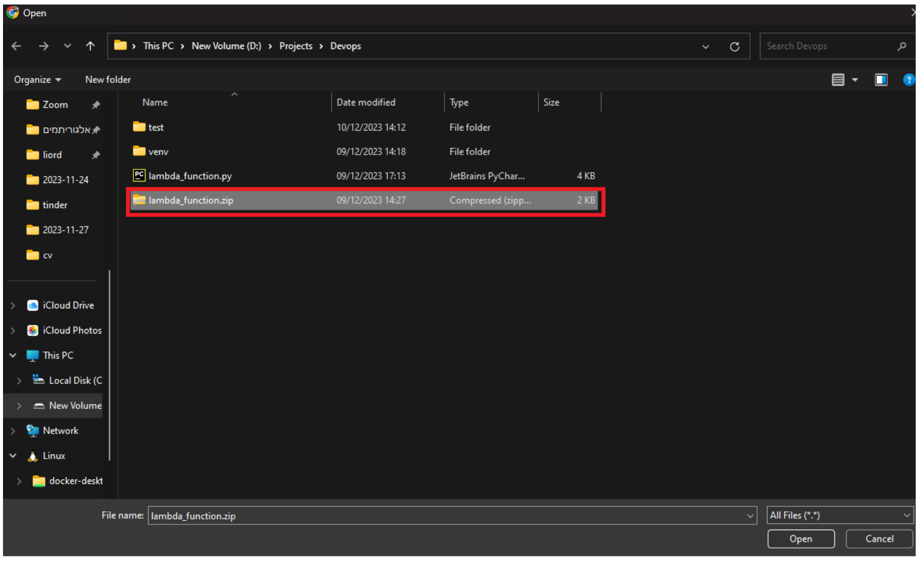
   - In the popup, click "Upload". Navigate to the location on your computer where lambda_function.zip is stored, select it, and confirm the upload.



Upload a .zip file

When you upload a new .zip file package, it overwrites the existing code.

For files larger than 10 MB, consider uploading using Amazon S3.



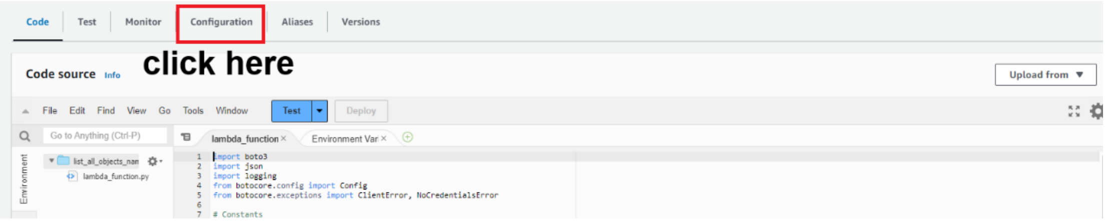
Name	Date modified	Type	Size
test	10/12/2023 14:12	File folder	
venv	09/12/2023 14:18	File folder	
lambda_function.py	09/12/2023 17:13	JetBrains PyChar...	4 KB
lambda_function.zip	09/12/2023 14:27	Compressed (zipp...	2 KB

File name: lambda_function.zip

All Files (*.*)

   - Once the upload is complete, click "Save" to apply the uploaded code to your Lambda function.

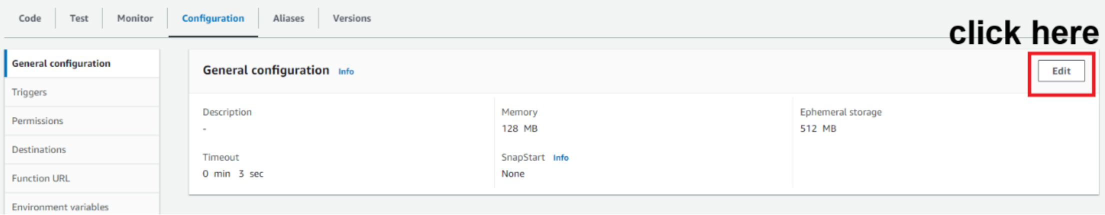
6. **Change Function Timeout:** If your Lambda function interacts with large S3 buckets or performs operations that may take longer, it's crucial to adjust the timeout setting. This ensures that the function has sufficient time to list all items without being prematurely terminated, Follow these steps:
   - **Edit Configuration**: Click on "Configuration" tab, then "General configuration" section."



```
import boto3
import json
import logging
from botocore.config import Config
from botocore.exceptions import ClientError, NoCredentialsError

# Constants
```

   - **Adjust Timeout:** Click "Edit", find "Timeout" setting, and set for 30 seconds.



Description
-

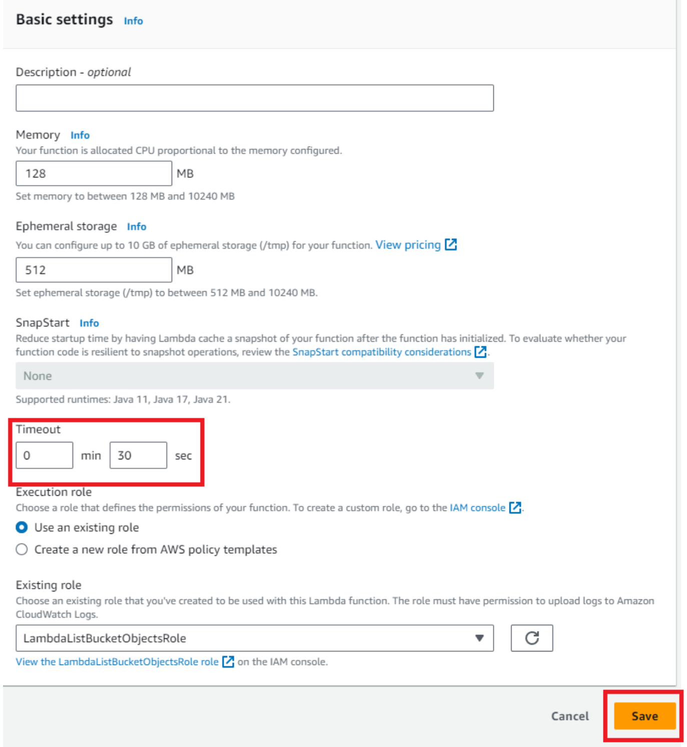
Memory
128 MB

Ephemeral storage
512 MB

Timeout
0 min 3 sec

SnapStart Info
None

   - **Save Changes:** Click "Save" to apply the new timeout setting.



Description - *optional*

Memory Info
Your function is allocated CPU proportional to the memory configured.
128 MB
Set memory to between 128 MB and 10240 MB

Ephemeral storage Info
You can configure up to 10 GB of ephemeral storage (/tmp) for your function. View pricing
512 MB
Set ephemeral storage (/tmp) to between 512 MB and 10240 MB.

SnapStart Info
Reduce startup time by having Lambda cache a snapshot of your function after the function has initialized. To evaluate whether your function code is resilient to snapshot operations, review the SnapStart compatibility considerations.
None
Supported runtimes: Java 11, Java 17, Java 21.

Timeout
0 min 30 sec

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.
● Use an existing role
○ Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
LambdaListBucketObjectsRole
View the LambdaListBucketObjectsRole role on the IAM console.