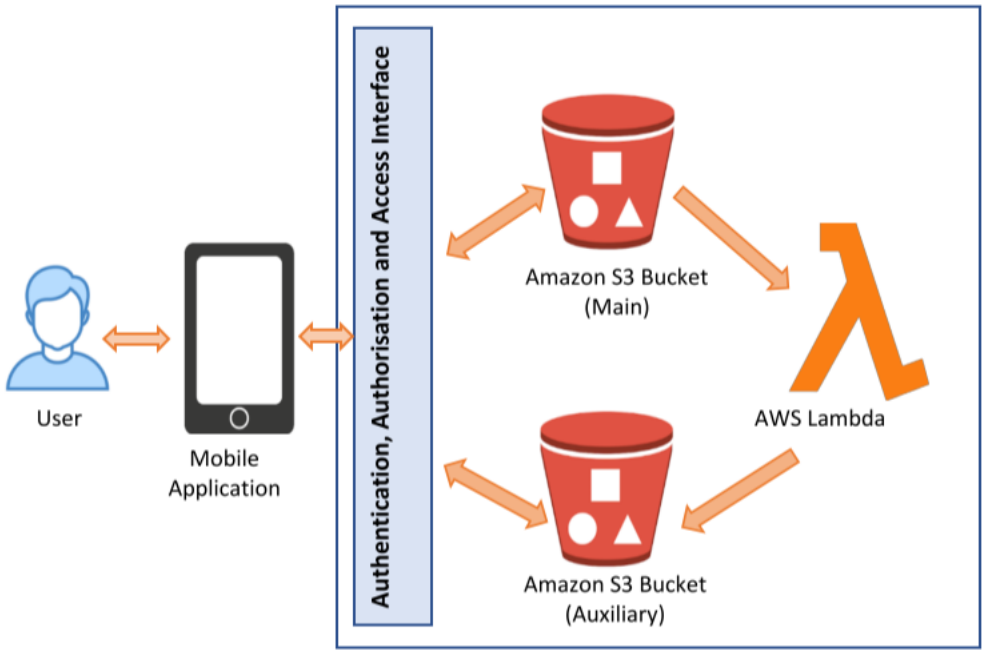
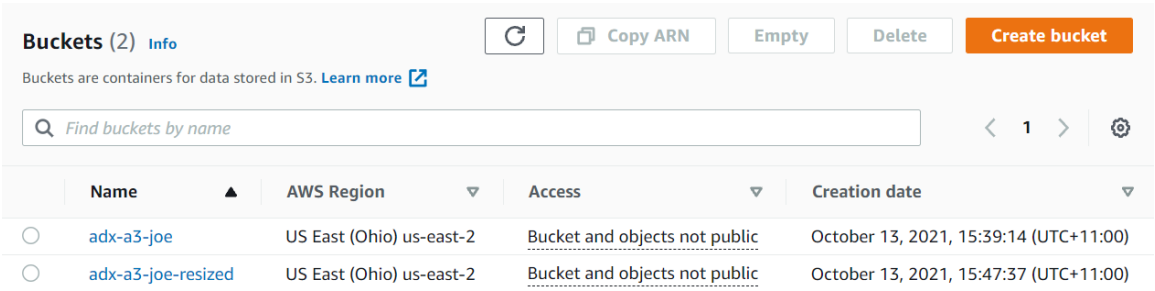
Designing a cloud-based infrastructure for object storage and serverless computing



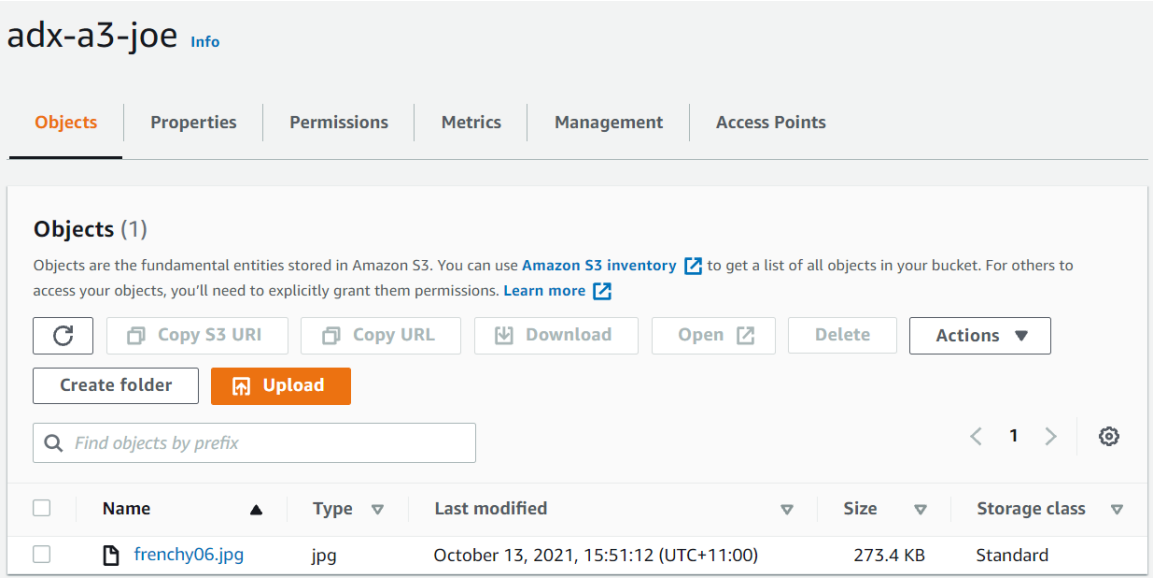
Please note that this guide does not include the configuration for the authentication, authorisation and access interface.

1. Create S3 buckets and upload a sample image in the bucket

- Navigate to **Amazon S3**, click **Create bucket**
- Bucket 1
  - Bucket Name: adx-a3-joe
  - AWS Region: US East (Ohio) us-east-2
  - Block Public Access: Block all public access
  - Bucket Versioning: Disable
  - Default encryption: Disable
- Bucket 2
  - Bucket Name: adx-a3-joe-resized
  - AWS Region: US East (Ohio) us-east-2
  - Block Public Access: Block all public access
  - Bucket Versioning: Disable
  - Default encryption: Disable

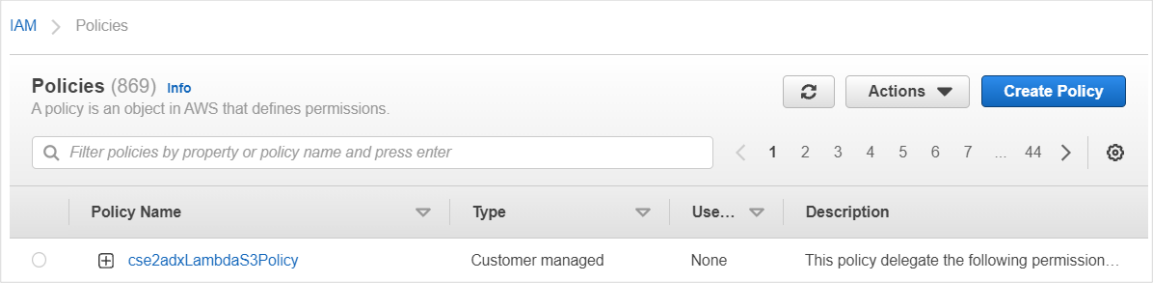


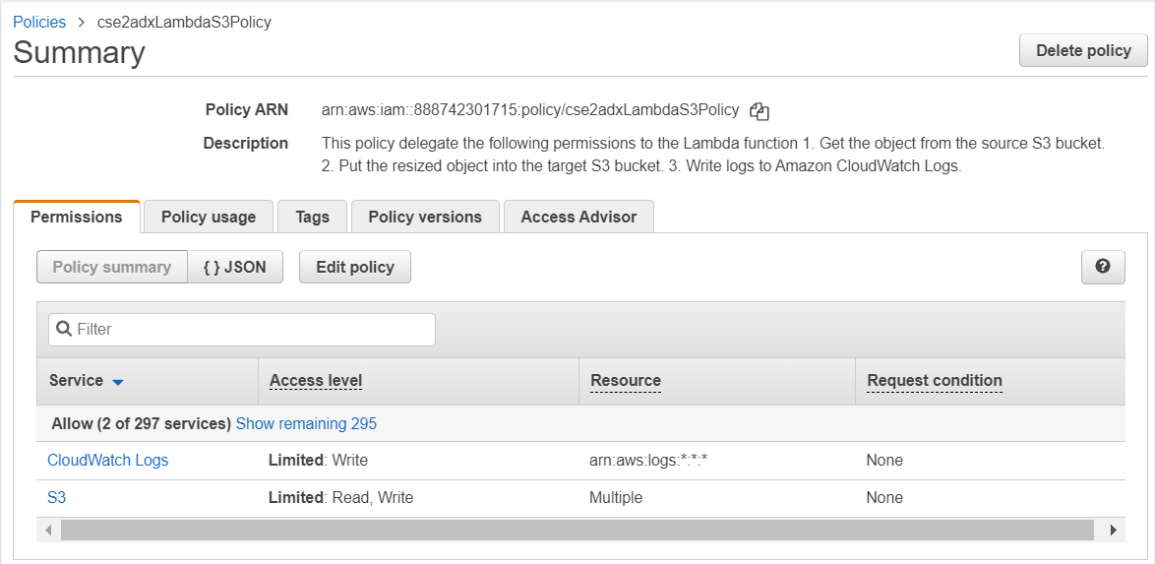
- Select **adx-a3-joe** and click **Upload**



2. Create the IAM policy for granting permissions to the Lambda function

- Navigate to **IAM Dashboard**, click **Policies**, **Create Policy**, **JSON**

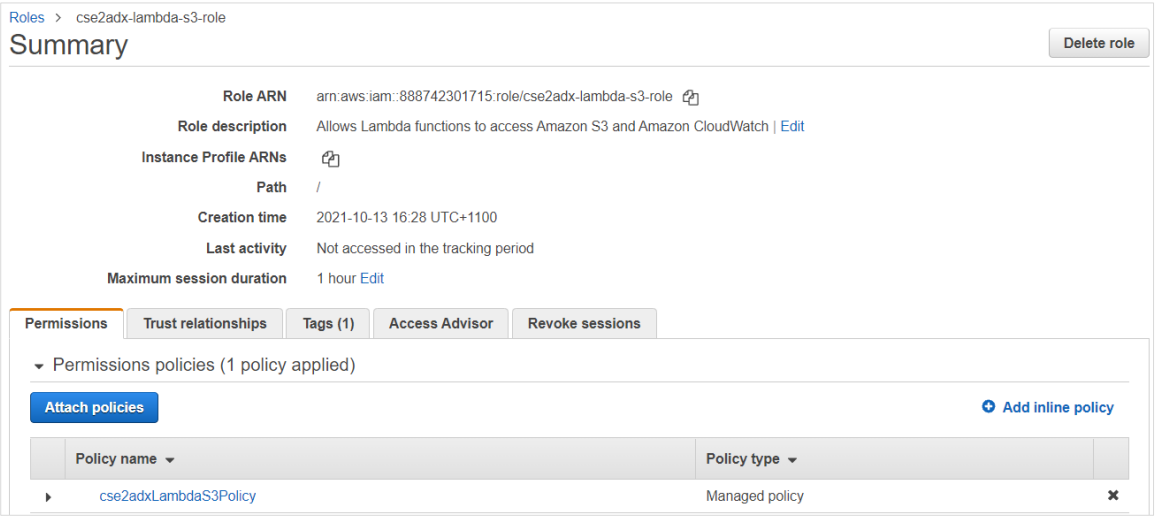




```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents",
                "logs:CreateLogGroup",
                "logs:CreateLogStream"
            ],
            "Resource": "arn:aws:logs:*:*:*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::adx-a3-joe/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::adx-a3-joe-resized/*"
        }
    ]
}
```

3. Create the execution role

- Navigate to **IAM Dashboard**, click **Roles**, **Create role**
    - Select type of trusted entity: AWS service
    - Choose a use case: Lambda
    - Attach permissions policies: cse2adxLambdaS3Policy
    - Role name: cse2adx-lambda-s3-role
    - Role description: Allows Lambda functions to access Amazon S3 and Amazon CloudWatch



4. Create the function code

- Create a python code in a text editor

lambda_function - Notepad

File Edit Format View Help

```
# Code for creating thumbnails from S3 bucket (CSE2ADX)
import boto3
import os
import sys
import uuid
from PIL import Image
import PIL.Image

s3_client = boto3.client('s3')

def resize_image(image_path, resized_path):
    with Image.open(image_path) as image:
        image.thumbnail((128, 128))
        image.save(resized_path)

def lambda_handler(event, context):
    for record in event['Records']:
        bucket = record['s3']['bucket']['name']
        key = record['s3']['object']['key']
        download_path = '/tmp/{}{}'.format(uuid.uuid4(), key)
        upload_path = '/tmp/resized-{}'.format(key)

        s3_client.download_file(bucket, key, download_path)
        resize_image(download_path, upload_path)
        s3_client.upload_file(upload_path, '{}-resized'.format(bucket), key)
```

Ln 1, Col 56 | 100% | Unix (LF) | UTF-8

5. Create the deployment package

- I launched an EC2 instance and access it via SSH PuTTY.

yum install pip	// install **pip**
sudo rm /usr/bin/python	
sudo ln -s /usr/bin/python3.7 /usr/bin/python	// run **Python3.7**
pip install virtualenv	// install the **virtualenv** package
python3 -m venv a3lambda	// create a virtual environment
cd a3lambda	// go into **a3lambda**
dir	// check directories

```
[root@ip-172-31-34-152 ec2-user]# python3 -m venv a3lambda
[root@ip-172-31-34-152 ec2-user]# cd a3lambda
[root@ip-172-31-34-152 a3lambda]# ls
bin  include  lib  lib64  pyvenv.cfg
[root@ip-172-31-34-152 a3lambda]#
```

pip install boto3	// install **boto3**
pip install docutils	// install **docutils**
pip install pillow	// install **pillow**

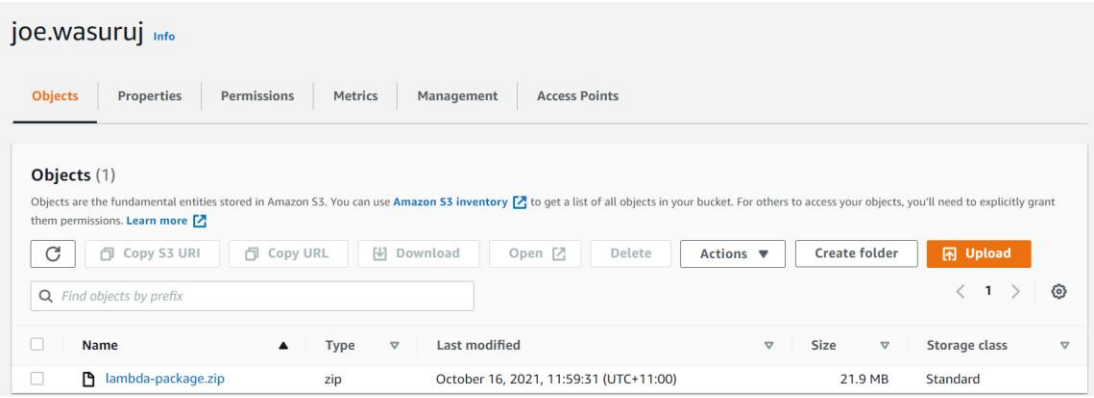
sudo find / -type d -iname “xxx”	// find where the packages are
yes | cp -ar /source/* /destination/	// copy the packages to **a3lambda**

/usr/local/lib/python3.7/site-packages/*	// path for the **boto3** package

```
/usr/local/lib64/python3.7/site-packages/*          // path for the PIL package
/usr/lib/python3.7/site-packages/*                  // path for the docutils package

/home/ec2-user/a3lambda/lib/python3.7/site-packages  // path for the destination

touch lambda_function.py                            // create lambda_function.py
cat << EOF > lambda_function.py                     // write code to lambda_function.py
// Code from Task 2.4 //
EOF

find . -exec zip lambda-package.zip {} +            // create lambda-package.zip
```

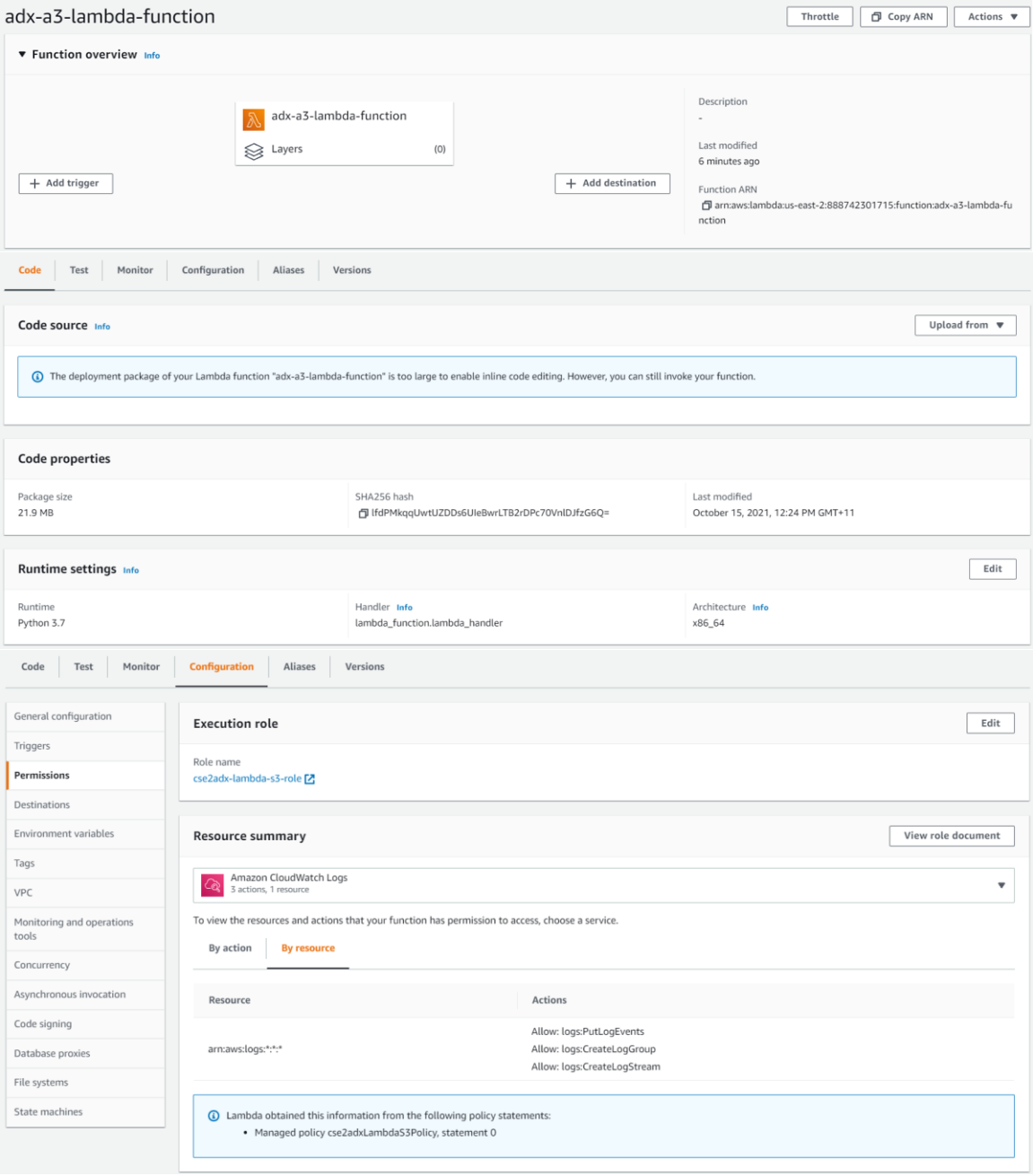
```
[root@ip-172-31-34-152 site-packages]# ls
aws_cfn_bootstrap-2.0-py3.7.egg-info  lambda_function.py              pystache-0.5.4-py3.7.egg-info
boto3                                 lambda-package.zip              python_daemon-2.2.3-py3.7.egg-info
boto3-1.18.62.dist-info               lockfile                        python_dateutil-2.8.2.dist-info
botocore                              lockfile-0.11.0-py3.7.egg-info  s3transfer
botocore-1.21.62.dist-info            PIL                             s3transfer-0.5.0.dist-info
cfnbootstrap                          Pillow-8.3.2.dist-info          setuptools
daemon                                Pillow.libs                     setuptools-47.1.0.dist-info
dateutil                              pip                             setuptools-49.1.3.dist-info
docutils                              pip-20.1.1.dist-info            six-1.16.0.dist-info
docutils-0.14-py3.7.egg-info          pip-20.2.2.dist-info            six.py
easy_install.py                       pkg_resources                   urllib3
jmespath                              __pycache__                     urllib3-1.26.7.dist-info
jmespath-0.10.0.dist-info             pystache
[root@ip-172-31-34-152 site-packages]#
```

```
zip -sf lambda-package.zip                          // check content of lambda-package.zip

pip install awscli                                  // install AWS CLI
aws configure                                       // enter Access Key ID & Secret Access Key
aws s3 cp /home/ec2-user/a3lambda/lib/python3.7/site-packages/lambda-package.zip
s3://joe.wasuruj/ --acl public-read                 // upload lambda-package.zip to S3
```



6. Create the Lambda function

- Navigate to **Lambda Dashboard**, click **Functions**, **Create function**
- Function name: adx-a3-lambda-function
- Runtime: Python 3.7
- Architecture: x86_64
- Permissions: Use an existing role: **cse2adx-lambda-s3-role**
- In **Code source**, Upload from: Amazon S3 location
  https://s3.us-east-2.amazonaws.com/joe.wasuruj/lambda-package.zip

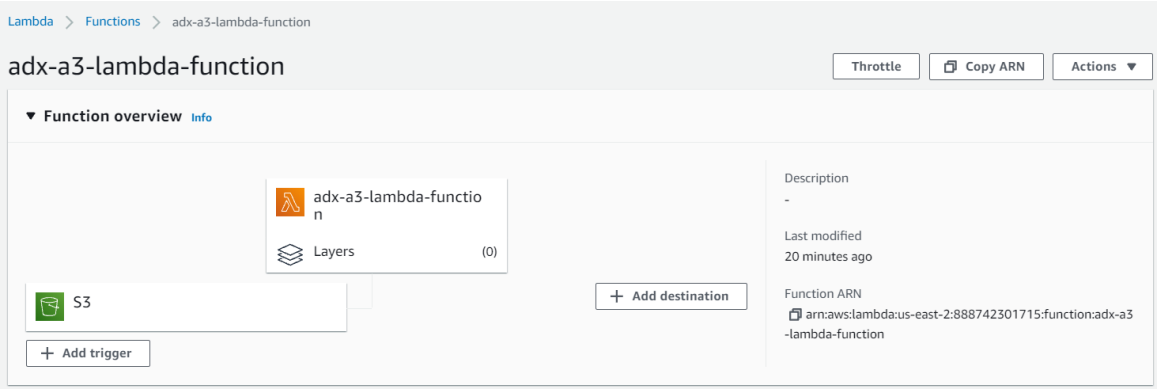


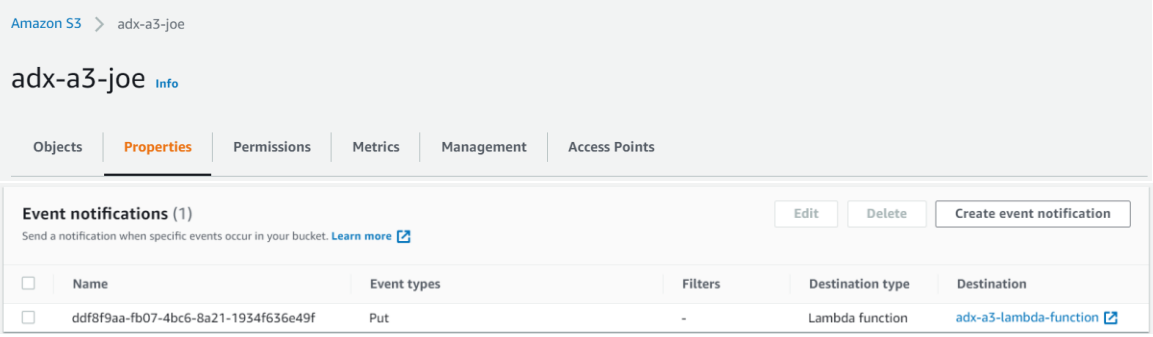
- The OS base that runs Python in AWS does not use the forked libraries such as Pillow that are deprecated. I need to add a layer in Lambda so that it can run Pillow:
- In **adx-a3-lambda-fucntion**, **Code**, **Layers**, click **Add a layer**
- Choose a layer: Specify an ARN

arn:aws:lambda:us-east-2:113088814899:layer:Klayers-python37-Pillow:11

7. Configure Amazon S3 to publish events and invoke the Lambda function

- Creating a trigger for the Lambda Function from **AWS Console** can be done in two ways:
- [1] In **Lambda Dashboard**, click **Functions**, **adx-a3-lambda-function**
    - In **Configurations**, click **Triggers**, **Add trigger**
    - Service: S3
    - Bucket: adx-a3-joe
    - Event type: PUT
- [2] In **S3 Dashboard**, click **Buckets**, **adx-a3-joe**, **Properties**
    - In **Event notifications**, click **Create event notification**
    - Event name: S3-trigger
    - Event type: PUT
    - Destination: Lambda function: adx-a3-lambda-function
- I only needed to do one of the two methods. The trigger will automatically appear in both the Lambda Dashboard and the S3 Dashboard.

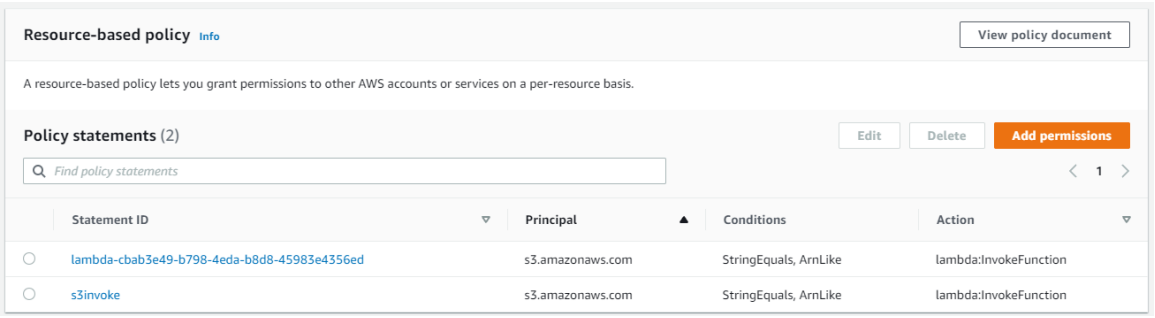




- Add permissions to the function access policy to allow S3 to invoke the function

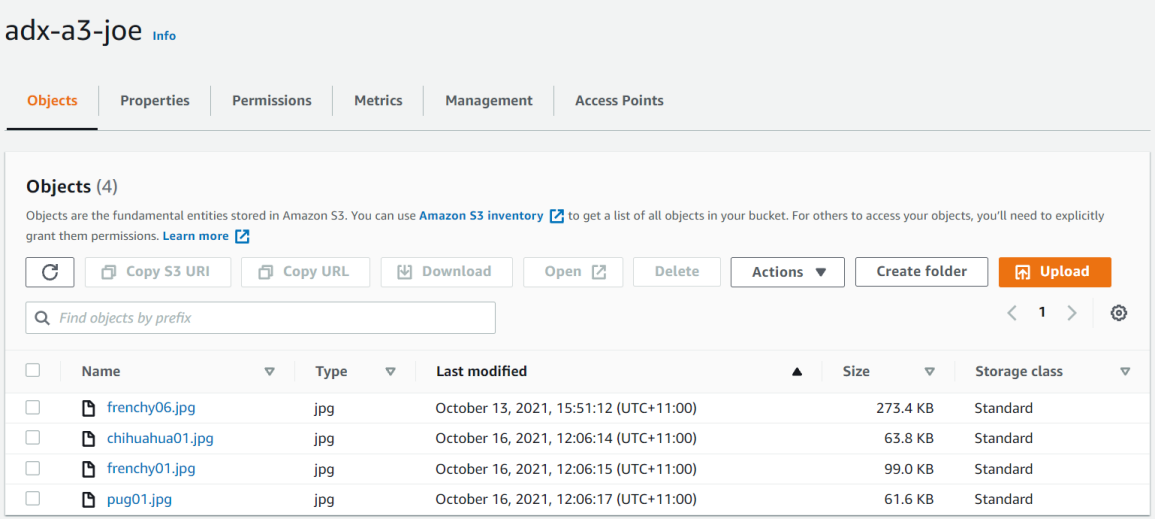
aws lambda add-permission --function-name adx-a3-lambda-function --principal s3.amazonaws.com --statement-id s3invoke --action "lambda:InvokeFunction" --source-arn arn:aws:s3:::adx-a3-joe --source-account **********

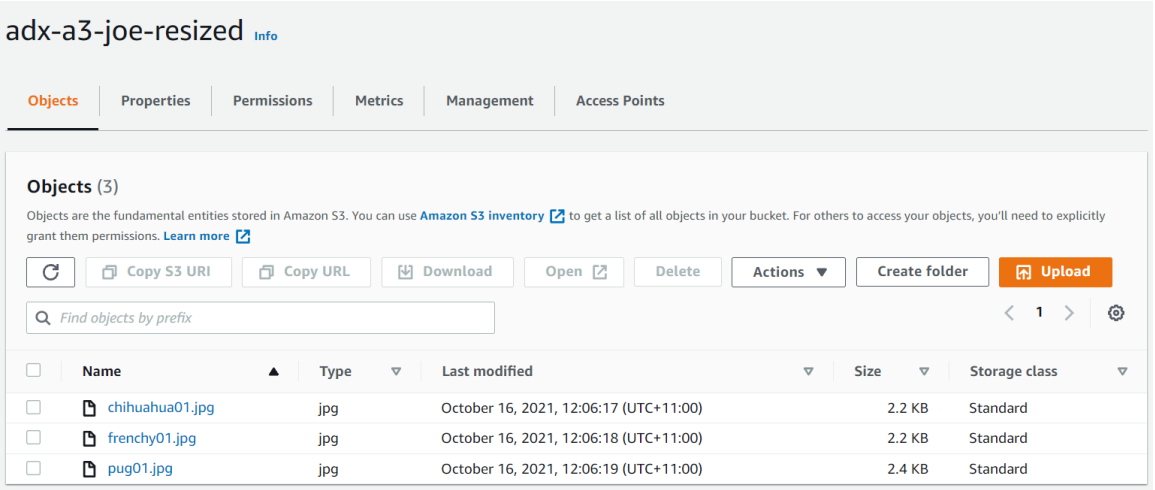
```
[ec2-user@ip-172-31-34-152 ~]$ aws lambda add-permission --function-name adx-a3-lambda-function
--principal s3.amazonaws.com --statement-id s3invoke --action "lambda:InvokeFunction" --source-a
rn arn:aws:s3:::adx-a3-joe --source-account [redacted]
{"Sid":"s3invoke","Effect":"Allow","Principal":{"Service":"s3.amazonaws.com"},"Action":"lambda:I
nvokeFunction","Resource":"arn:aws:lambda:us-east-2:[redacted]:function:adx-a3-lambda-function
","Condition":{"StringEquals":{"AWS:SourceAccount":"[redacted]"},"ArnLike":{"AWS:SourceArn":"a
rn:aws:s3:::adx-a3-joe"}}}
[ec2-user@ip-172-31-34-152 ~]$
```

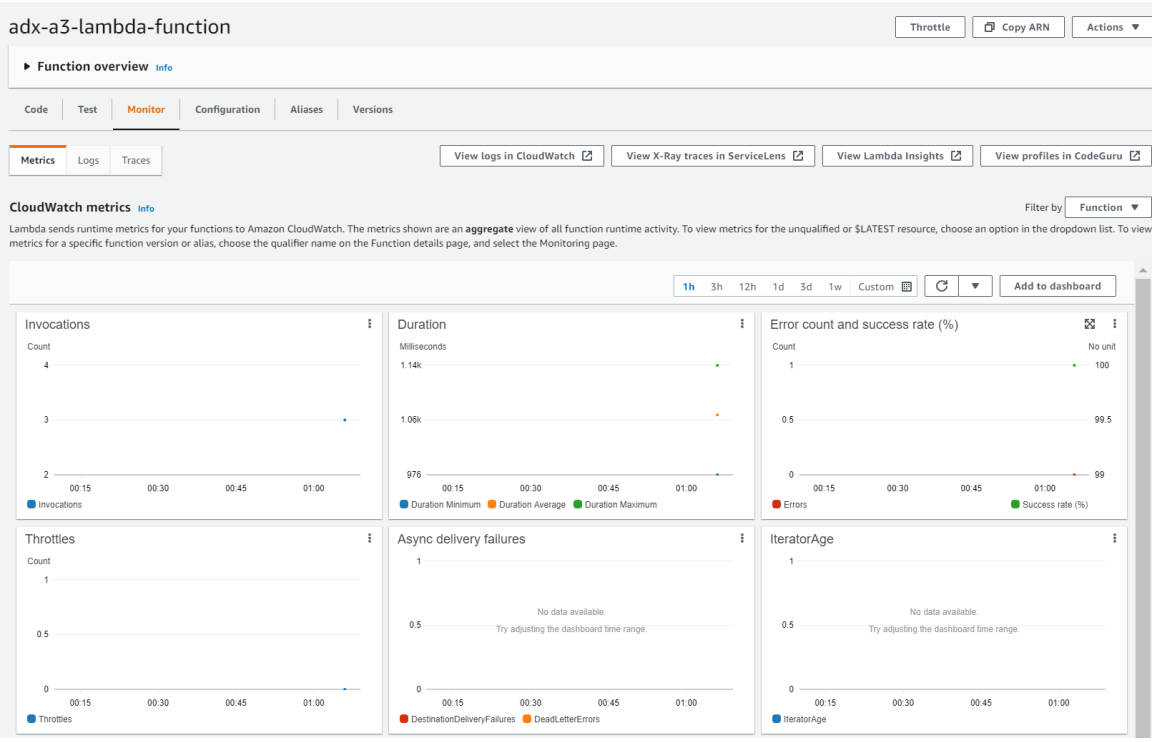


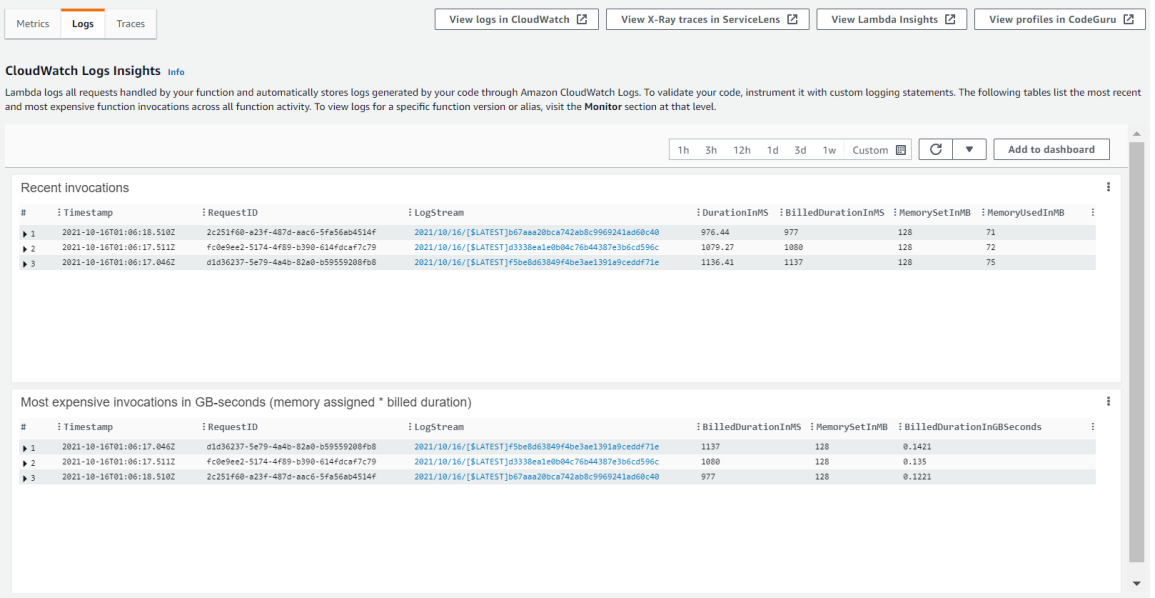
8. Test using the S3 trigger

- I uploaded 3 photos in the source bucket **adx-a3-joe**
  - chihuahua01.jpg
  - frenchy01.jpg
  - pug01.jpg









https://github.com/joe-wasuruj/aws-step-by-step-guides