PROJECT :1

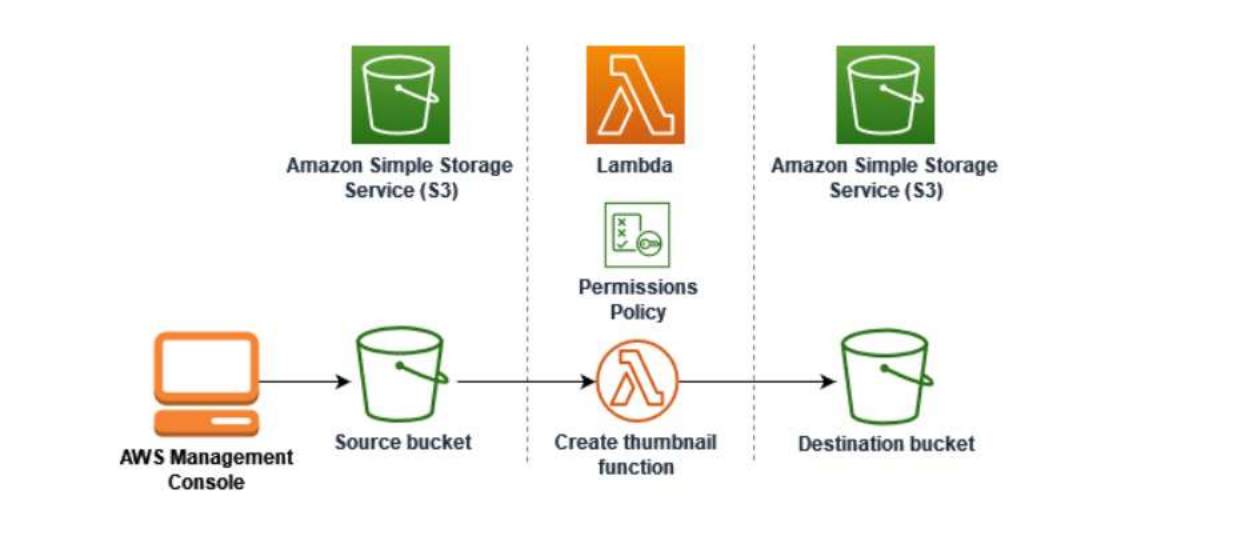
SERVERLESS IMAGE PROCESSING

CREATE A SERVERLESS IMAGE PROCESSING THAT AUTOMATICALLY RESIZE AND OPTIMIZE AN IMAGE UPLOADED TO AMAZON S3 BUCKET

TABLE OF CONTENT

OVERVIEW

- CREATE TWO S3 BUCKET
- CREATE IAM POLICY
- CREATE LAMBDA FUNCTION
- ADD TRIGGER
- TEST EVENT



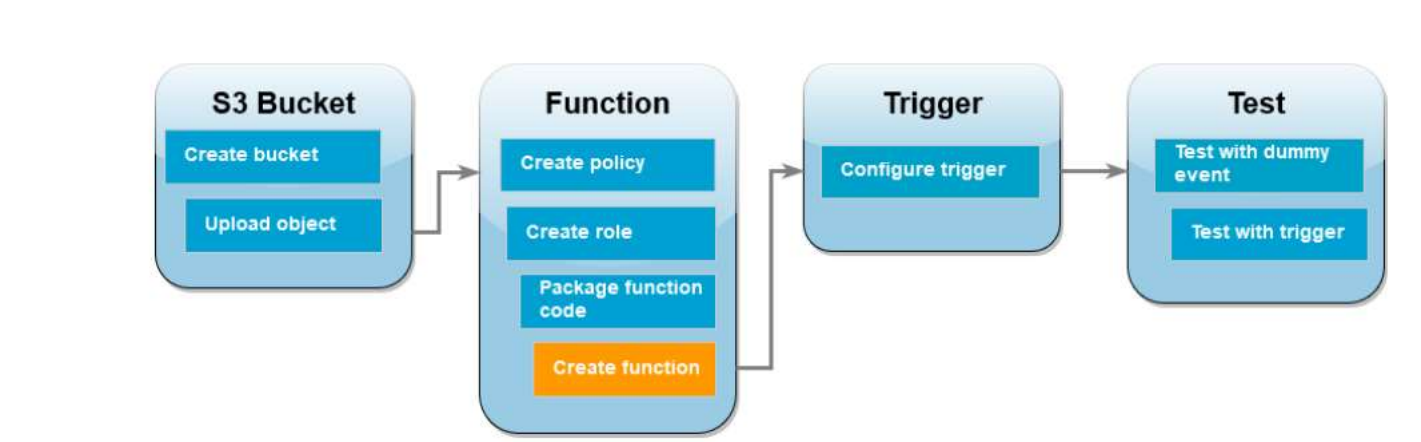
SERVERLESS FUNCTION

In the context of AWS (Amazon Web Services), lambda functions refer to AWS Lambda, which is a serverless computing service provided by Amazon. Here's a detailed explanation of what AWS Lambda is and how lambda functions relate to it.
AWS Lambda allows you to run code without provisioning or managing servers. It automatically scales your application by running code in response to triggers or events. Lambda functions are the units of code that you deploy and execute on AWS Lambda.

Key Feature of Lambda Function

- Serverless Execution
- Event-driven
- Pay-per-use
- Integration with AWS Services

HOW LAMBDA WORK

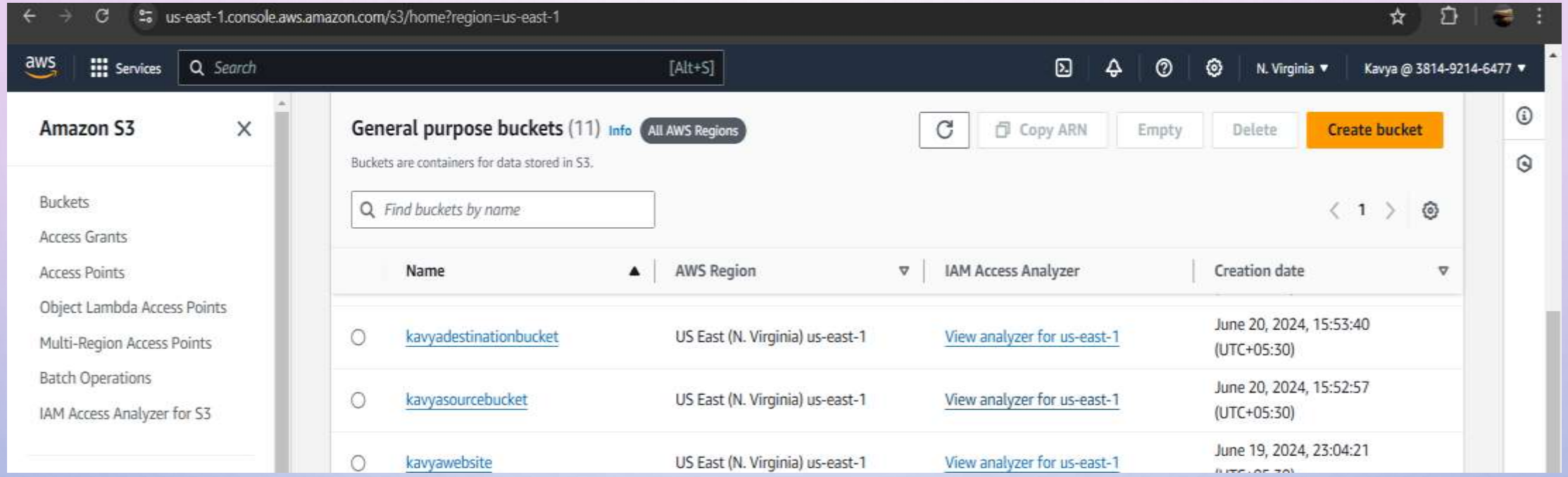


STEPS TO CREATE SERVERLESS IMAGE PROCESSING

STEP 1: create s3 buckets

- Go to AWS management console and sign in with your credentials
- Once logged in, you'll be in the AWS Management Console dashboard. Find and click on **"S3"** under **"Storage"**, or you can type "S3" in the search bar and select it
- In the S3 dashboard, click on the **"Create bucket"** button.
- Enter a unique name for your bucket. Bucket names must be globally unique across all of AWS, so if your desired name is taken, you'll need to choose another.
- In the option **Block Public Access Setting**, uncheck the option
- Click on I acknowledge the current setting might result in this bucket and the object it becoming public checkbox
- Keep everything default and click **on Create Bucket button**

- Create two buckets ***source*** ***and*** ***destination*** by above steps.


us-east-1.console.aws.amazon.com/s3/home?region=us-east-1
aws
Services
Search
[Alt+S]
N. Virginia
Kavya @ 3814-9214-6477
Amazon S3
Buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
General purpose buckets (11) Info
All AWS Regions
Buckets are containers for data stored in S3.
Copy ARN
Empty
Delete
Create bucket
Find buckets by name
1
Name
AWS Region
IAM Access Analyzer
Creation date
kavyadestinationbucket
US East (N. Virginia) us-east-1
View analyzer for us-east-1
June 20, 2024, 15:53:40 (UTC+05:30)
kavyasourcebucket
US East (N. Virginia) us-east-1
View analyzer for us-east-1
June 20, 2024, 15:52:57 (UTC+05:30)
kavyawebsite
US East (N. Virginia) us-east-1
View analyzer for us-east-1
June 19, 2024, 23:04:21

CREATING IAM POLICY AND ROLE

- Navigate to the IAM console.
- Click on "Policies" in the left-hand menu.
- Click on "Create policy".
- Select the "JSON" tab to create your policy from scratch.
- Paste the JSON policy document given below

```
{
"Version": "2012-10-17",
"Statement": [
{
"Effect": "Allow",
"Action": [
"logs:CreateLogGroup",
"logs:CreateLogStream",
"logs:PutLogEvents"
],
"Resource": "arn:aws:logs:*:*:*"
},
{
"Effect": "Allow",
"Action": [
"s3:GetObject"
],
"Resource": "arn:aws:s3:::your-bucket-name/*"
}
]
}
```

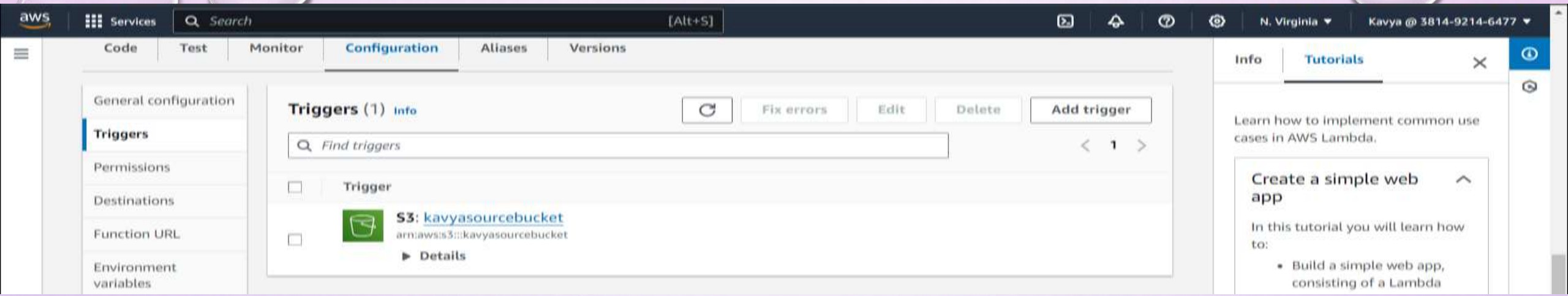
- Make sure that you update arn of your source bucket.
- Give policy name resizer.
- Click on "create policy" to save and create the IAM policy.
- Now on left panel select **Roles.**
- Click on create role and click on next.
- Select **Lambda** in **Use Case** dialog box.
- **Click on next**
- Select policy we created.
- Give role name and clickmon **Create Role**

Create Lambda Function

- Navigation to lambda.
- Create a function. Set function name Lambda function.
- Set runtime to Node.js 18.x
- Change default execution role to select existing one.
- Select image-resize-policy.
- Create function . lambda function is created

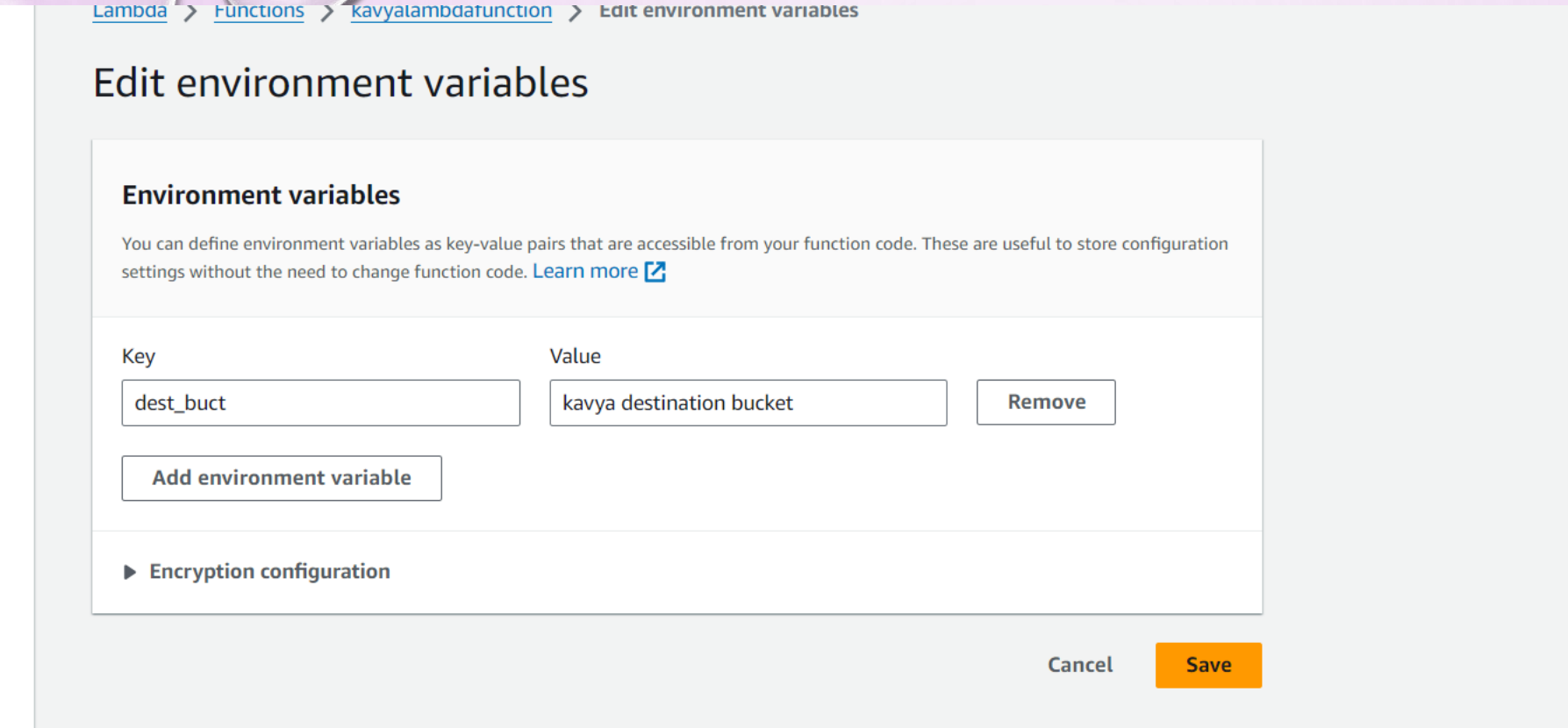
Add trigger

- Click on add trigger.
- Select **s3** in dialog box.
- Select **source bucket name**.
- Click on I acknowledge…….. check box. And leave all as default.
- Click on **ADD** button



Create environmental variable

- Click on environmental variable in left panel.
- Click on edit
- IN dialog box
- Key : dest-buct
- Value : destination bucket.
- Click on **save** button



Upload the zip file by clicking on code then upload.

Test the function

- Click on test tabs on the tabs
- On **Template :** select s3 Put

Successfully updated the function **kavyalambdafunction**.

Event JSON

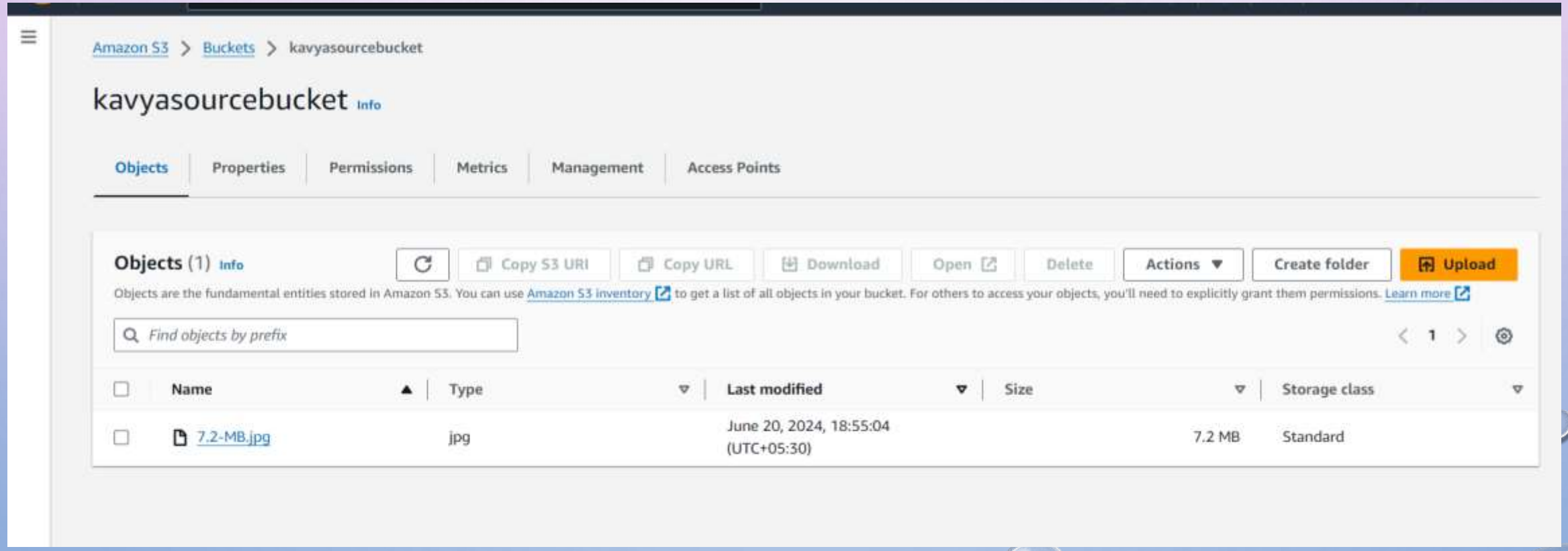
Format JSON


```
{
  "Records": [
    {
      "eventVersion": "2.0",
      "eventSource": "aws:s3",
      "awsRegion": "us-east-1",
      "eventTime": "1970-01-01T00:00:00.000Z",
      "eventName": "ObjectCreated:Put",
      "userIdentity": {
        "principalId": "EXAMPLE"
      },
      "requestParameters": {
        "sourceIPAddress": "127.0.0.1"
      },
      "responseElements": {
        "x-amz-request-id": "EXAMPLE123456789",
        "x-amz-id-2": "EXAMPLE123/5678abcdefghijklambdaisawesome/mnopqrstuvwxyzABCDEFGH"
      },
      "s3": {
        "s3SchemaVersion": "1.0",
        "configurationId": "testConfigRule",
        "bucket": {
          "name": "kavyasourcebucket",
          "ownerIdentity": {
            "principalId": "EXAMPLE"
          },
          "arn": "arn:aws:s3:::kavyasourcebucket"
        },
        "object": {
          "key": "7.2-MB.jpg",
          "size": 1024,
          "eTag": "0123456789abcdef0123456789abcdef",
          "sequencer": "0A1B2C3D4E5F678901"
        }
      }
    }
```

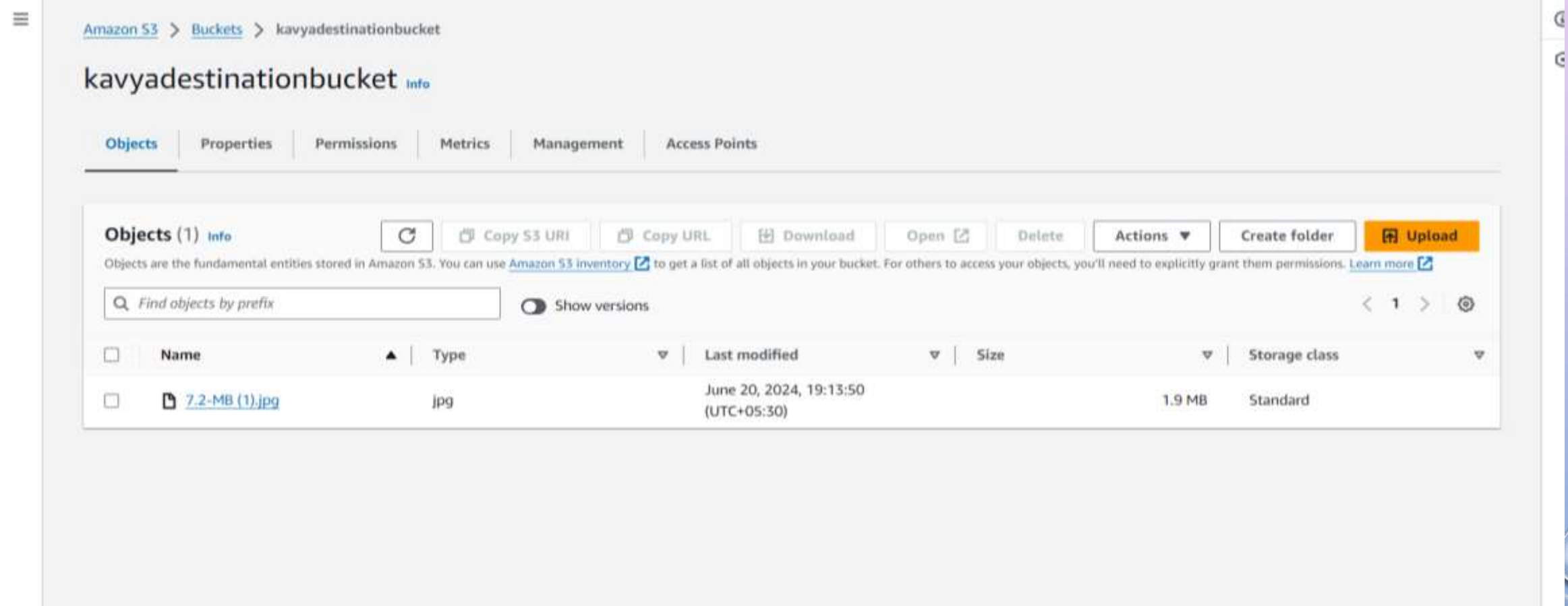

CloudShell Feedback

RESULT

Source Bucket



Destination Bucket



CONCLUSION

lambda functions offer a concise and efficient way to resize images programmatically. Their ability to be defined inline and used directly where needed reduces the need for separate named functions or methods, streamlining code and improving readability. By leveraging lambda functions for image resizing, developers can achieve faster implementation times and maintain cleaner code bases, ultimately enhancing the overall efficiency and maintainability of image processing tasks in their applications.