
Create S3 bucket from AWS CLI

- Configure aws with the terminal

Pip3 install aws

Pip3 install awscli

```
~ [aws configure
AWS Access Key ID [None]: AKIAZI2LHWBXGJ67XRXA
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]: json]
```

- Create iam role:

```
~ [aws iam create-role --role-name mys3fullaccess --assume-role-policy-document
file:///Users/jayantasudhani/Documents/AWS/AWS_Assignment/trust_policy.json
{
    "Role": {
        "Path": "/",
        "RoleName": "mys3fullaccess",
        "RoleId": "AROAZI2LHWBXPB4T5HNMR",
        "Arn": "arn:aws:iam::637423562862:role/mys3fullaccess",
        "CreateDate": "2024-05-03T07:27:11Z",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "Service": "lambda.amazonaws.com"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        }
    }
}
```

Trust-policy.json

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

- Attach S3 Full Access policy to the above created role:

```
aws iam attach-role-policy --role-name mys3fullaccess --policy-arn arn:aws:iam::aws:policy/AmazonS3FullAccess
```

The screenshot shows the AWS IAM Roles page. On the left, the navigation menu includes Identity and Access Management (IAM), Dashboard, Access management (User groups, Users, Roles), Policies, Identity providers, Account settings, and Access reports (Access Analyzer, External access, Unused access, Analyzer settings). The main content area displays the 'mys3fullaccess' role under the 'Roles' section. The 'Summary' tab shows the role was created on May 03, 2024, at 12:57 (UTC+05:30). The ARN is listed as arn:aws:iam::637423562862:role/mys3fullaccess. The 'Permissions' tab is selected, showing one attached policy: 'AmazonS3FullAccess'. This policy is described as an AWS managed policy. The bottom of the screen shows the Mac OS X dock with various application icons.

- Create an IAM instance profile and attach it to the above role:

```
~ 13:06:08
└─ aws iam add-role-to-instance-profile --role-name mys3fullaccess --instance-profile-name MyInstanceProfile

~ 13:00:58
└─ aws iam create-instance-profile --instance-profile-name MyInstanceProfile

{
  "InstanceProfile": {
    "Path": "/",
    "InstanceProfileName": "MyInstanceProfile",
    "InstanceProfileId": "AIPAZI2LHWBXHY2FFNPUL",
    "Arn": "arn:aws:iam::637423562862:instance-profile/MyInstanceProfile",
    "CreateDate": "2024-05-03T07:36:08Z",
    "Roles": []
  }
}
```

- List IAM instance profiles:

```
~ 13:06:54
└─ aws iam list-instance-profiles

{
  "InstanceProfiles": [
    {
      "Path": "/",
      "InstanceProfileName": "AmazonEMR-InstanceProfile-20240417T154613",
      "InstanceProfileId": "AIPAZI2LHWBXCS2YIMAWO",
      "Arn": "arn:aws:iam::637423562862:instance-profile/AmazonEMR-InstanceProfile-20240417T154613",
      "CreateDate": "2024-04-17T10:16:16Z",
      "Roles": [
        {
          "Path": "/service-role/",
          "RoleName": "AmazonEMR-InstanceProfile-20240417T154613",
          "RoleId": "AROAZI2LHWBXBCMPUEQME",
          "Arn": "arn:aws:iam::637423562862:role/service-role/AmazonEMR-InstanceProfile-20240417T154613",
          "CreateDate": "2024-04-17T10:16:15Z",
          "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
              {
                "Effect": "Allow",
                "Principal": {
                  "Service": "ec2.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
              }
            ]
          }
        }
      ],
      "Path": "/",
      "InstanceProfileName": "AmazonEMR-InstanceProfile-20240418T121701",
      "InstanceProfileId": "AIPAZI2LHWBXPC52AWH5F",
      "Arn": "arn:aws:iam::637423562862:instance-profile/AmazonEMR-InstanceProfile-20240418T121701",
    }
  ]
}
```

- Create an EC2 instance using above role and key:

```
~ 13:07:45
aws ec2 run-instances --image-id ami-07caf09b362be10b8 --count 1 --instance-type t2.micro --key-name key_assignment --iam-instance-profile Name=MyInstanceProfile --region us-east-1
{
  "Groups": [],
  "Instances": [
    {
      "AmiLaunchIndex": 0,
      "ImageId": "ami-07caf09b362be10b8",
      "InstanceId": "i-076c805fb8d8e1ba2",
      "InstanceType": "t2.micro",
      "KeyName": "key_assignment",
      "LaunchTime": "2024-05-03T07:41:23.000Z",
      "Monitoring": {
        "State": "disabled"
      },
      "Placement": {
        "AvailabilityZone": "us-east-1a",
        "GroupName": "",
        "Tenancy": "default"
      },
      "PrivateDnsName": "ip-172-31-23-92.ec2.internal",
      "PrivateIpAddress": "172.31.23.92",
      "ProductCodes": [],
      "PublicDnsName": "",
      "State": {
        "Code": 0,
        "Name": "pending"
      },
      "StateTransitionReason": "",
      "SubnetId": "subnet-058b45b2fadec518f",
      "VpcId": "vpc-0e59ae9bc7fbe4092",
      "Architecture": "x86_64",
      "BlockDeviceMappings": [],
      "ClientToken": "ea405ad0-f12e-4e71-b2a2-ea54c7ceb026",
      "EbsOptimized": false,
      "EnaSupport": true,
    }
  ]
}
```

The screenshot shows the AWS Management Console interface for the EC2 service. The left sidebar navigation bar includes links for EC2 Dashboard, EC2 Global View, Events, Console-to-Code (Preview), Instances, Images, Elastic Block Store, and CloudShell. The main content area displays the 'Instance summary for i-076c805fb8d8e1ba2' page. The summary table contains the following details:

Attribute	Value
Instance ID	i-076c805fb8d8e1ba2
IPV6 address	-
Hostname type	IP name: ip-172-31-23-92.ec2.internal
Answer private resource DNS name	-
Auto-assigned IP address	54.196.162.122 [Public IP]
IAM Role	mys3fullaccess
IMDSv2 Required	
Public IPv4 address	54.196.162.122 open address
Instance state	Running
Private IP DNS name (IPv4 only)	ip-172-31-23-92.ec2.internal
Instance type	t2.micro
VPC ID	vpc-0e59ae9bc7fbe4092
Subnet ID	subnet-058b45b2fadec518f
Private IPv4 addresses	172.31.23.92
Public IPv4 DNS	ec2-54-196-162-122.compute-1.amazonaws.com open address
Elastic IP addresses	-
AWS Compute Optimizer finding	Opt-in to AWS Compute Optimizer for recommendations. Learn more
Auto Scaling Group name	-

Below the summary table, there are tabs for Details, Status and alarms (New), Monitoring, Security, Networking, Storage, and Tags. The bottom of the screen shows the Mac OS X dock with various application icons.

- Create s3 bucket:

```
aws s3api create-bucket --bucket mybucket9882 --region us-east-1

{
  "Location": "/mybucket9882"
}
```

The screenshot shows a web browser window with the AWS S3 console. The URL in the address bar is `us-east-1.console.aws.amazon.com/s3/buckets/mybucket9882?region=us-east-1&bucketType=standard`. The page title is "mybucket9882". The main content area shows the bucket details with "Objects (0)". The left sidebar has a tree view with "Amazon S3" expanded, showing "Buckets" and other services like "CloudWatch Metrics" and "AWS Lambda". The bottom of the screen shows the macOS dock with various application icons.

Put files in S3 bucket from lambda

- Create policy to put object in S3
putObjectPolicy.json

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::vrabuckvra/*"
    }
  ]
}
```

```
~ 13:13:03
└─ aws iam create-policy --policy-name LambdaPutPolicy --policy-document file:///Users/jayantasudhani/Documents/AWS/AWS_Assignment/s3putobjectpolicy.json
{
  "Policy": {
    "PolicyName": "LambdaPutPolicy",
    "PolicyId": "ANPAZI2LHWBXE3EFRGWHZ",
    "Arn": "arn:aws:iam::637423562862:policy/LambdaPutPolicy",
    "Path": "/",
    "DefaultVersionId": "v1",
    "AttachmentCount": 0,
    "PermissionsBoundaryUsageCount": 0,
    "IsAttachable": true,
    "CreateDate": "2024-05-03T07:50:47Z",
    "UpdateDate": "2024-05-03T07:50:47Z"
  }
}
```

- Create a new IAM role and attach the above policy:

```
~ 13:20:47
└─ aws iam create-role --role-name lambdaS3PutRole --assume-role-policy-document file:///Users/jayantasudhani/Documents/AWS/AWS_Assignment/trust_policy.json
{
  "Role": {
    "Path": "/",
    "RoleName": "lambdaS3PutRole",
    "RoleId": "AROAZI2LHWBXLRCOTYYD",
    "Arn": "arn:aws:iam::637423562862:role/lambdaS3PutRole",
    "CreateDate": "2024-05-03T07:53:34Z",
    "AssumeRolePolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Principal": {
            "Service": "lambda.amazonaws.com"
          },
          "Action": "sts:AssumeRole"
        }
      ]
    }
  }
}
```

The screenshot shows the AWS IAM LambdaPutPolicy configuration page. The left sidebar is titled 'Identity and Access Management (IAM)' and includes sections for Dashboard, Access management (User groups, Users, Roles), Policies (selected), Identity providers, Account settings, and Access reports (Access Analyzer, External access, Unused access, Analyzer settings, Credential report, Organization activity). The main content area shows the 'LambdaPutPolicy' details. It has tabs for Policy details, Permissions (selected), Entities attached, Tags, Policy versions (1), and Access Advisor. Under 'Permissions defined in this policy', it shows 'Allow (1 of 411 services)' for S3 with a Limited: Write access level and a Resource ARN of arn:aws:s3:::your-bucket-name. There are tabs for Edit, Summary (selected), and JSON.

- Create an IAM policy that allows logging to CloudWatch Logs and attach to the above created role:
cloudWatchPolicies.json

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "lambda:InvokeFunction"
            ],
            "Resource": "*"
        }
    ]
}
  
```

```
aws iam create-policy --policy-name CloudWatchPolicy --policy-document file:///Users/jayantasudhani/Documents/AWS/AWS_Assignment/cloudWatchPolicies.json
{
  "Policy": {
    "PolicyName": "CloudWatchPolicy",
    "PolicyId": "ANPAZI2LHWBXBAIXAPMB4",
    "Arn": "arn:aws:iam::637423562862:policy/CloudWatchPolicy",
    "Path": "/",
    "DefaultVersionId": "v1",
    "AttachmentCount": 0,
    "PermissionsBoundaryUsageCount": 0,
    "IsAttachable": true,
    "CreateDate": "2024-05-03T08:01:54Z",
    "UpdateDate": "2024-05-03T08:01:54Z"
  }
}
```

- Create lambda function to put file in the bucket
Lambda_function.py

```
import json
import boto3
import os
from datetime import datetime

def lambda_handler(event, context):
    # Get current execution count from environment variable
    execution_count = int(os.environ.get('EXECUTION_COUNT', 0))

    # Increment execution count
    execution_count += 1

    # Update execution count in environment variable
    os.environ['EXECUTION_COUNT'] = str(execution_count)

    # Stop execution after 3 runs
    if execution_count > 3:
        return {
            'statusCode': 200,
            'body': 'Execution stopped after 3 runs'
        }

    # Generate JSON data
    json_data = {
```

```

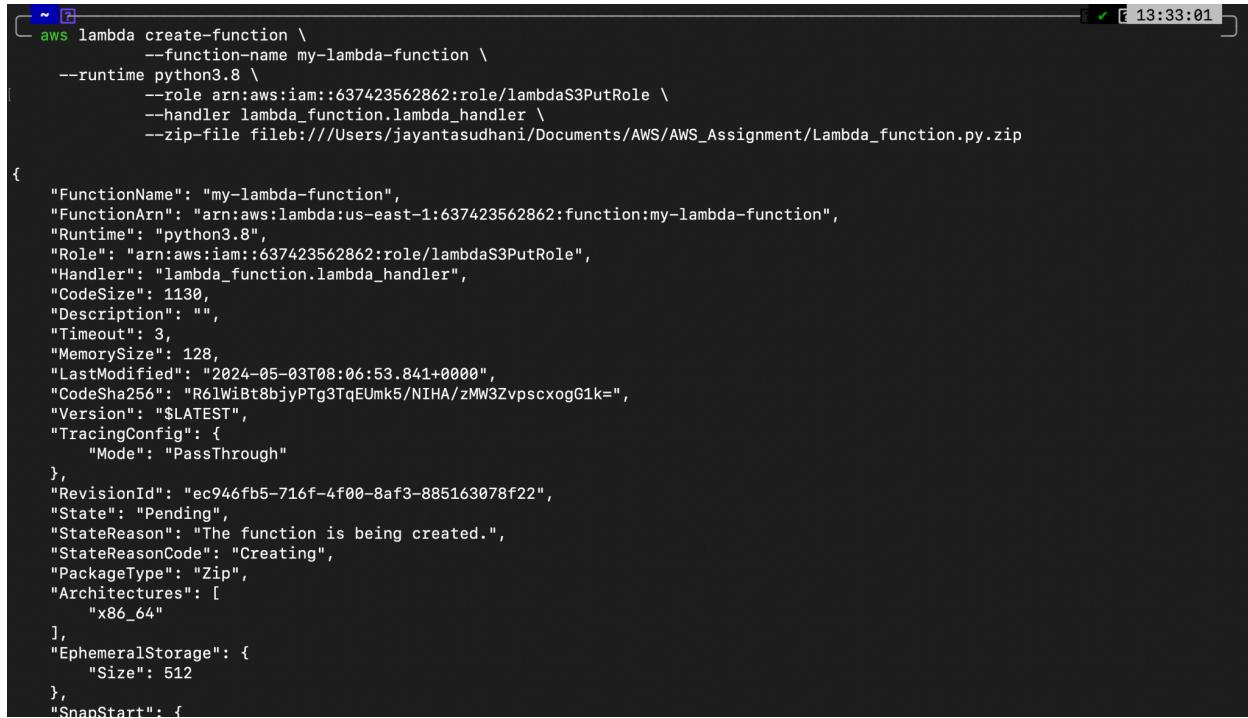
    "transaction_id": 12345,
    "payment_mode": "card/netbanking/upi",
    "amount": 200.0,
    "customer_id": 101,
    "timestamp": str(datetime.now())
}

# Save JSON data to S3
bucket_name = 'vrabuckvra'
file_name = f"transaction_{datetime.now().strftime('%Y-%m-%d_%H-%M-%S')}.json"

s3 = boto3.client('s3')
s3.put_object(Bucket=bucket_name, Key=file_name, Body=json.dumps(json_data))

return {
    'statusCode': 200,
    'body': json.dumps('JSON file saved successfully')
}

```



The screenshot shows a terminal window with the AWS Lambda command-line interface (CLI) running. The command used is:

```
aws lambda create-function \
  --function-name my-lambda-function \
  --runtime python3.8 \
  --role arn:aws:iam::637423562862:role/lambdaS3PutRole \
  --handler lambda_function.lambda_handler \
  --zip-file fileb:///Users/jayantasudhani/Documents/AWS/AWS_Assignment/Lambda_function.py.zip
```

The output of the command is displayed below, showing the configuration of the newly created Lambda function:

```
{
  "FunctionName": "my-lambda-function",
  "FunctionArn": "arn:aws:lambda:us-east-1:637423562862:function:my-lambda-function",
  "Runtime": "python3.8",
  "Role": "arn:aws:iam::637423562862:role/lambdaS3PutRole",
  "Handler": "lambda_function.lambda_handler",
  "CodeSize": 1130,
  "Description": "",
  "Timeout": 3,
  "MemorySize": 128,
  "LastModified": "2024-05-03T08:06:53.841+0000",
  "CodeSha256": "R61WiBt8bjyPTg3TqEUmk5/NIHA/zMW3ZvpscXogG1k=",
  "Version": "$LATEST",
  "TracingConfig": {
    "Mode": "PassThrough"
  },
  "RevisionId": "ec946fb5-716f-4f00-8af3-885163078f22",
  "State": "Pending",
  "StateReason": "The function is being created.",
  "StateReasonCode": "Creating",
  "PackageType": "Zip",
  "Architectures": [
    "x86_64"
  ],
  "EphemeralStorage": {
    "Size": 512
  },
  "SnapStart": {}
}
```

my-lambda-function

Description
-

Last modified
1 minute ago

Function ARN
arn:aws:lambda:us-east-1:637423562862:function:my-lambda-function

Function URL: Info
-

Tutorials

Create a simple web app

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

[Learn more](#)

[Start tutorial](#)

- Create event to trigger the lambda_function every minute for 3 times

Create event rule:

```
aws events put-rule --name "LambdaExecutionScheduleRule" \
--schedule-expression "rate(1 minute)" \
--state "ENABLED" \
--description "Trigger Lambda function every minute" \
--event-pattern "{\"source\":[\"aws.events\"],\"detail-type\":[\"Scheduled Event\"],\"resources\":[\"arn:aws:events:us-east-1:637423562862:rule/LambdaExecutionScheduleRule\"]}" \
{ \
    "RuleArn": "arn:aws:events:us-east-1:637423562862:rule/LambdaExecutionScheduleRule" \
}
```

Add rule to the function:

```
aws events put-targets --rule "LambdaExecutionScheduleRule" --targets "Id="1","Arn"="arn:aws:lambda:us-east-1:637423562862:function:my-lambda-function"
{
    "FailedEntryCount": 0,
    "FailedEntries": []
}
```

- On execution of the lambda_function files got loaded in S3 bucket every minute for 3 times

Brave File Edit View History Bookmarks People Tab Window Help Fri 3 May 3:44 PM

Welcome to Stop loading this page AWS_Assignme AWS_Assignme AWS assignmen my-lambda-func Policies | IAM mybucket9882 JayantAtSigmoid COVID-19 Data ChatGPT + DE Fresher Induct... JayantAtSigmoid (... Python Guides: It... Services Search [Option+S] N. Virginia Jayant

Amazon S3

Buckets

- Access Grants
- Access Points
- Object Lambda Access Points
- Multi-Region Access Points
- Batch Operations
- IAM Access Analyzer for S3

Block Public Access settings for this account

Storage Lens

- Dashboards
- Storage Lens groups
- AWS Organizations settings

Feature spotlight 7

AWS Marketplace for S3

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Brave File Edit View History Bookmarks People Tab Window Help Fri 3 May 10:49 PM

Welcome to Sign... ChatGPT DE Fresher Induct... JayantAtSigmoid (... Python Guides: It... Services Search [Option+S] N. Virginia Jayant

CloudWatch

Favorites and recents Dashboards Alarms ▲ 0 ○ 0 ○ 0 Logs Log groups Log Anomalies Live Tail Logs Insights Metrics X-Ray traces Events Application Signals Network monitoring Insights Settings Getting Started What's new

CloudWatch Log groups /aws/lambda/my-lambda-function 2024/05/03/[LATEST]283ec53e84d14b77834fc5ed32b2c3e2

Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

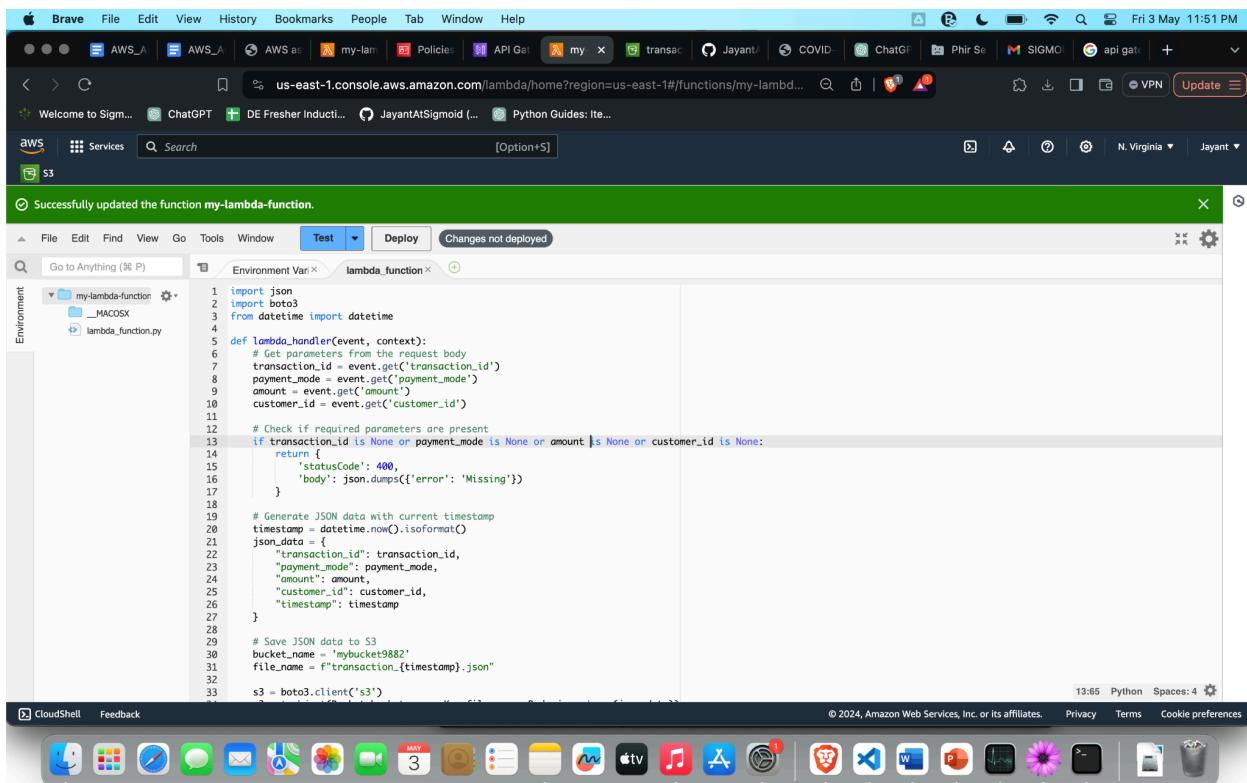
Filter events - press enter to search Clear 1m 30m 1h 12h Custom Local timezone Display

Timestamp	Message
2024-05-03T15:40:07.911+05:30	INIT_START Runtime Version: python:3.8.v48 Runtime Version ARN: arn:aws:lambda:us-east-1:runtim... RequestID: b81a8be2-f0df-432b-9c7c-ad7820bc3420 Version: \$LATEST
2024-05-03T15:40:08.202+05:30	START RequestID: b81a8be2-f0df-432b-9c7c-ad7820bc3420 Version: \$LATEST
2024-05-03T15:40:10.671+05:30	END RequestID: b81a8be2-f0df-432b-9c7c-ad7820bc3420
2024-05-03T15:40:10.671+05:30	REPORT RequestID: b81a8be2-f0df-432b-9c7c-ad7820bc3420 Duration: 2469.18 ms Billed Duration: 2470 ms Memory Size: 128 MB Max ... RequestID: 707d11d8-4474-4d4b-86e6-88ce7b06ab0f Version: \$LATEST
2024-05-03T15:42:15.736+05:30	START RequestID: 707d11d8-4474-4d4b-86e6-88ce7b06ab0f Version: \$LATEST
2024-05-03T15:42:16.010+05:30	END RequestID: 707d11d8-4474-4d4b-86e6-88ce7b06ab0f
2024-05-03T15:42:16.010+05:30	REPORT RequestID: 707d11d8-4474-4d4b-86e6-88ce7b06ab0f Duration: 273.94 ms Billed Duration: 274 ms Memory Size: 128 MB Max ... RequestID: 5a517603-ef8e-4c03-ac5c-73890ed0e393 Version: \$LATEST
2024-05-03T15:43:16.064+05:30	START RequestID: 5a517603-ef8e-4c03-ac5c-73890ed0e393
2024-05-03T15:43:16.370+05:30	END RequestID: 5a517603-ef8e-4c03-ac5c-73890ed0e393
2024-05-03T15:43:16.370+05:30	REPORT RequestID: 5a517603-ef8e-4c03-ac5c-73890ed0e393 Duration: 305.27 ms Billed Duration: 306 ms Memory Size: 128 MB Max ...

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

API gateway - Lambda integration

- Modify the lambda function to accept parameters and return file name.
`My_lambda_function.py`:



The screenshot shows the AWS Lambda console interface. The top navigation bar includes 'File', 'Edit', 'View', 'History', 'Bookmarks', 'People', 'Tab', 'Window', and 'Help'. Below the navigation is a toolbar with icons for 'Test' (selected), 'Deploy', and 'Changes not deployed'. The main area displays the code for 'lambda_function.py' under the environment 'my-lambda-function'. The code is as follows:

```
1 import json
2 import boto3
3 from datetime import datetime
4
5 def lambda_handler(event, context):
6     # Get parameters from the request body
7     transaction_id = event.get('transaction_id')
8     payment_mode = event.get('payment_mode')
9     amount = event.get('amount')
10    customer_id = event.get('customer_id')
11
12    # Check if required parameters are present
13    if transaction_id is None or payment_mode is None or amount is None or customer_id is None:
14        return {
15            'statusCode': 400,
16            'body': json.dumps({'error': 'Missing'})
17        }
18
19    # Generate JSON data with current timestamp
20    timestamp = datetime.now().isoformat()
21    json_data = {
22        "transaction_id": transaction_id,
23        "payment_mode": payment_mode,
24        "amount": amount,
25        "customer_id": customer_id,
26        "timestamp": timestamp
27    }
28
29    # Save JSON data to S3
30    bucket_name = 'mybucket9882'
31    file_name = f"transaction_{timestamp}.json"
32
33    s3 = boto3.client('s3')
34    s3.put_object(Bucket=bucket_name, Key=file_name, Body=json.dumps(json_data))
```

The bottom of the screen shows the Mac OS X dock with various application icons.

```
import json
import boto3
from datetime import datetime

def lambda_handler(event, context):
    # Get parameters from the request body
    transaction_id = event.get('transaction_id')
    payment_mode = event.get('payment_mode')
    amount = event.get('amount')
    customer_id = event.get('customer_id')

    # Check if required parameters are present
    if transaction_id is None or payment_mode is None or amount is None or customer_id is None:
```

```

    return {
        'statusCode': 400,
        'body': json.dumps({'error': 'Missing'})
    }

# Generate JSON data with current timestamp
timestamp = datetime.now().isoformat()
json_data = {
    "transaction_id": transaction_id,
    "payment_mode": payment_mode,
    "amount": amount,
    "customer_id": customer_id,
    "timestamp": timestamp
}

# Save JSON data to S3
bucket_name = 'mybucket9882'
file_name = f"transaction_{timestamp}.json"

s3 = boto3.client('s3')
s3.put_object(Bucket=bucket_name, Key=file_name, Body=json.dumps(json_data))

return {
    'statusCode': 200,
    'body': json.dumps({'file_name': file_name})
}

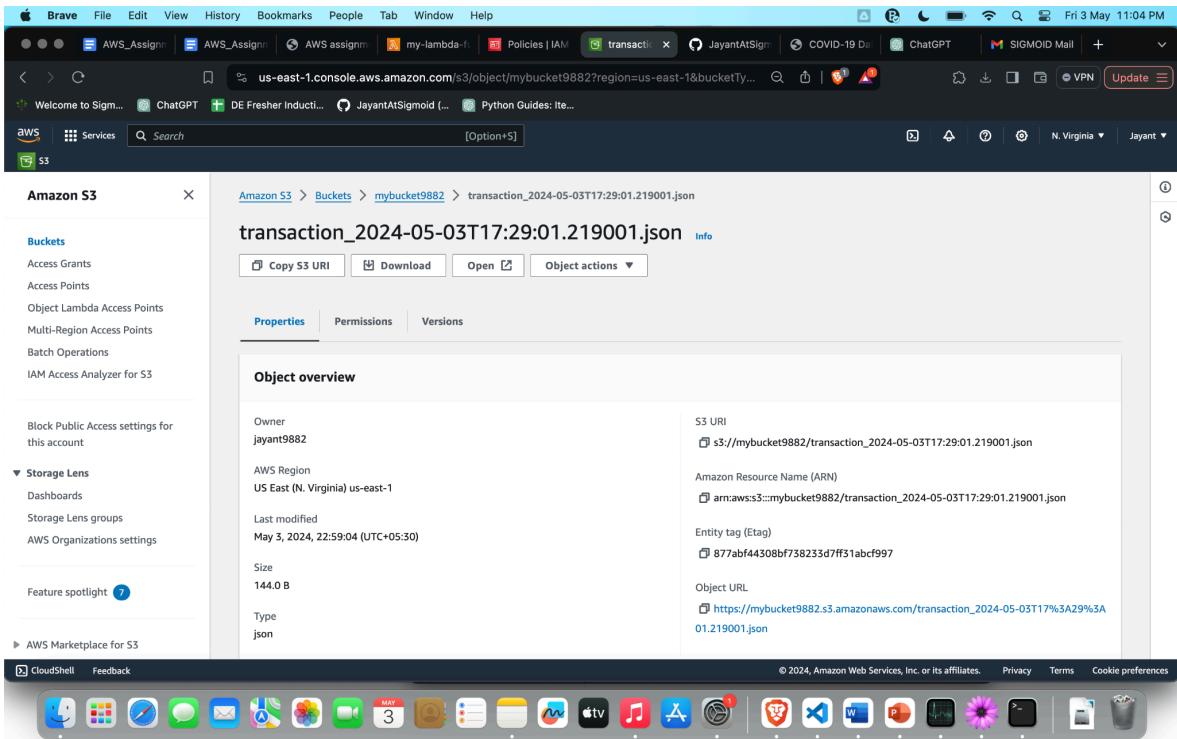
```

Deploying and invoking this function

```

[~] 22:57:18
aws lambda invoke --function-name my-lambda-function --payload '{"transaction_id": 12345, "payment_mode": "card/netbanking/upi", "amount": 200.0,"customer_id": 101}' output.json
{
    "StatusCode": 200,
    "ExecutedVersion": "$LATEST"
}

```



On checking this function independently, file successfully uploaded in bucket.

b. Create a POST API from API Gateway, pass parameters as request body to Lambda job. Return filename and status code as response.

Creating rest api from api gateway

```
~ [23:04:20]
aws apigateway create-rest-api --name 'FileAPI' --region us-east-1
{
  "id": "iqjl6tod3j",
  "name": "FileAPI",
  "createdDate": 1714757999,
  "apiKeySource": "HEADER",
  "endpointConfiguration": {
    "types": [
      "EDGE"
    ]
  },
  "disableExecuteApiEndpoint": false,
  "rootResourceId": "6jf7k9jm8g"
}

~ [23:10:21]
aws apigateway get-resources --rest-api-id iqjl6tod3j --region us-east-1
{
  "items": [
    {
      "id": "6jf7k9jm8g",
      "path": "/"
    }
  ]
}
```

```
~ 255 23:11:01
aws apigateway create-resource --rest-api-id iqjl6tod3j --parent-id 6jf7k9jm8g --path-part files --region us-east-1
{
  "id": "0cus3y",
  "parentId": "6jf7k9jm8g",
  "pathPart": "files",
  "path": "/files"
}
```

Assigning POST request.

```
~ 23:11:22
aws apigateway put-method --rest-api-id iqjl6tod3j --resource-id 6jf7k9jm8g --http-method POST --authorization-type NONE --region us-east-1
{
  "httpMethod": "POST",
  "authorizationType": "NONE",
  "apiKeyRequired": false
}

~ 23:12:09
aws apigateway put-integration --rest-api-id iqjl6tod3j --resource-id 6jf7k9jm8g --http-method POST --type AWS_PROXY --integration-type AWS_PROXY --uri 'arn:aws:apigateway:us-east-1:lambda:path/2015-03-31/functions/arn:aws:lambda:us-east-1:637423562862:function:my-lambda-function/invocations' --region us-east-1
{
  "type": "AWS_PROXY",
  "httpMethod": "POST",
  "uri": "arn:aws:apigateway:us-east-1:lambda:path/2015-03-31/functions/arn:aws:lambda:us-east-1:637423562862:function:my-lambda-function/invocations",
  "passthroughBehavior": "WHEN_NO_MATCH",
  "timeoutInMillis": 29000,
  "cacheNamespace": "6jf7k9jm8g",
  "cacheKeyParameters": []
}
```

Deploying in prod

```
~ 23:13:49
aws apigateway create-deployment --rest-api-id iqjl6tod3j --stage-name prod --region us-east-1
{
  "id": "7h4oqd",
  "createdDate": 1714758278
}
```

c. Consume API from local machine and pass unique data to lambda. Using curl to check the api

```
~ 23:44:27
curl -X POST 'https://iqjl6tod3j.execute-api.us-east-1.amazonaws.com/prod/my-lambda-function' \
-d '{"operation": "create", "payload": {"Item": { "transaction_id": 12345, "payment_mode": "card/netbanking/upi", "amount": 200.0, "customer_id": 101, "timestamp": "77" }}, {"file_name": "transaction_2024-05-03T18:20:48.469263.json"}'
```

<input type="checkbox"/>	transaction_2024-05-03T18:20:48.469263.json	json	May 3, 2024, 23:50:49 (UTC+05:30)	126.0 B	Standard
--------------------------	-------------------------------------------------------------	------	--------------------------------------	---------	----------

d. Check if cloudwatch logs are generated Cloudwatch logs:

The screenshot shows the AWS CloudWatch Log Events interface. The left sidebar navigation includes CloudWatch, Favorites and recents, Dashboards, Alarms, Logs (selected), Log groups, Metrics, X-Ray traces, Events, Application Signals, Network monitoring, and Insights. The main content area displays log events for the path: CloudWatch > Log groups > /aws/lambda/my-lambda-function > 2024/05/03/[\$.LATEST]0ec06ba15d4647d1b0c46b21896bd2b5. The log events table has columns for Timestamp and Message. The first message is "No older events at this moment. [Retry](#)". Subsequent messages show Lambda startup and execution details, such as INIT_START, START, REPORT, and END requests, with timestamps ranging from 2024-05-03T23:49:40 to 2024-05-03T23:50:48. The interface includes a search bar, filter controls (Clear, 1m, 30m, 1h, 12h, Custom, Local timezone, Display), and action buttons (Actions, Start tailing, Create metric filter).