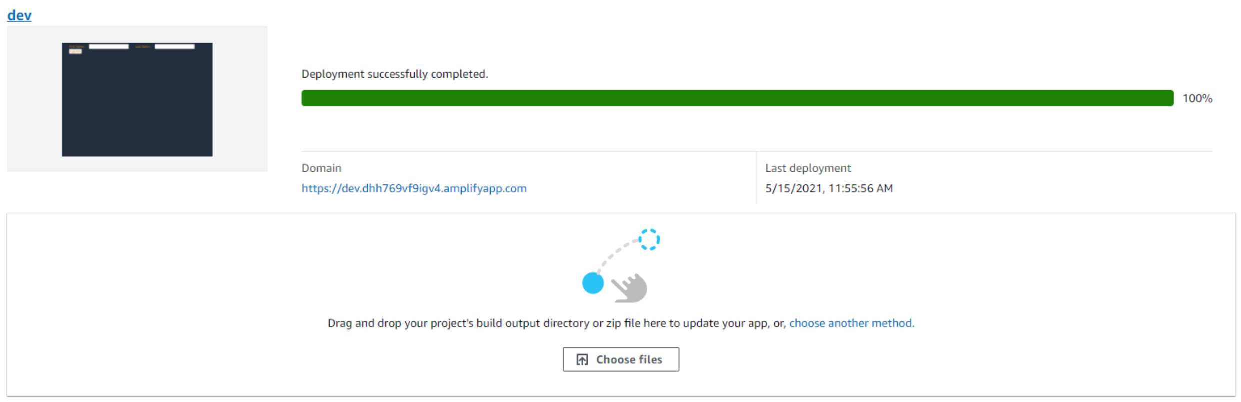
Introduction: Build a Basic Web App
Following the tutorial I deployed the web application:


dev

Deployment successfully completed.

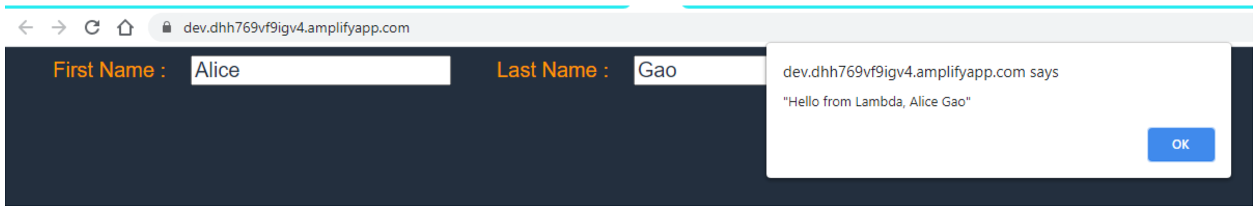
100%

Domain
https://dev.dhh769vf9igv4.amplifyapp.com

Last deployment
5/15/2021, 11:55:56 AM

Drag and drop your project's build output directory or zip file here to update your app, or, choose another method.

And got the correct response:


dev.dhh769vf9igv4.amplifyapp.com

First Name : Alice

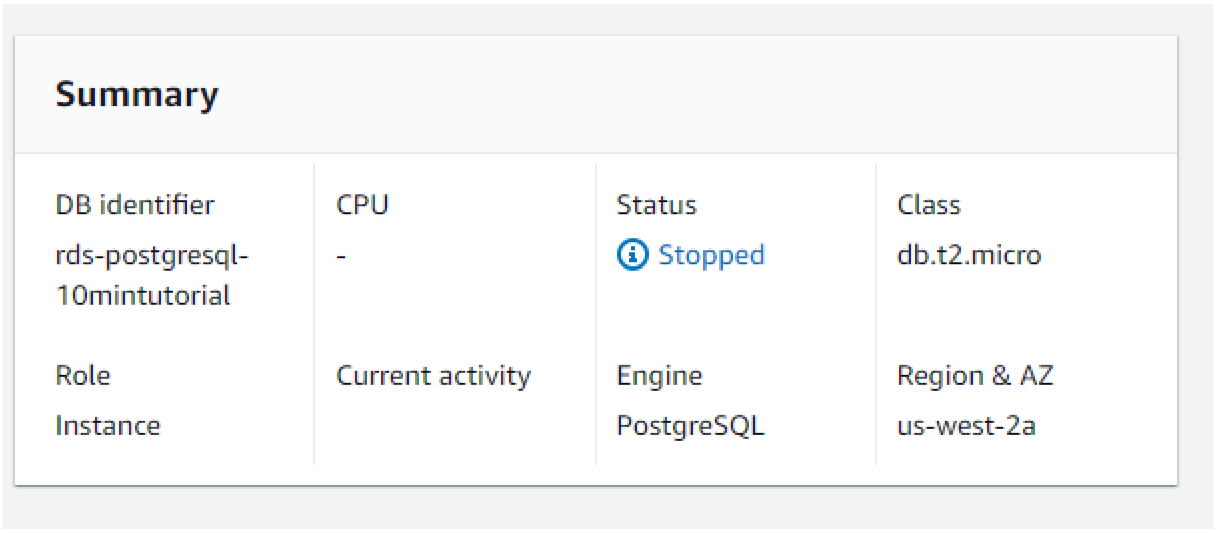
Last Name : Gao

dev.dhh769vf9igv4.amplifyapp.com says

"Hello from Lambda, Alice Gao"

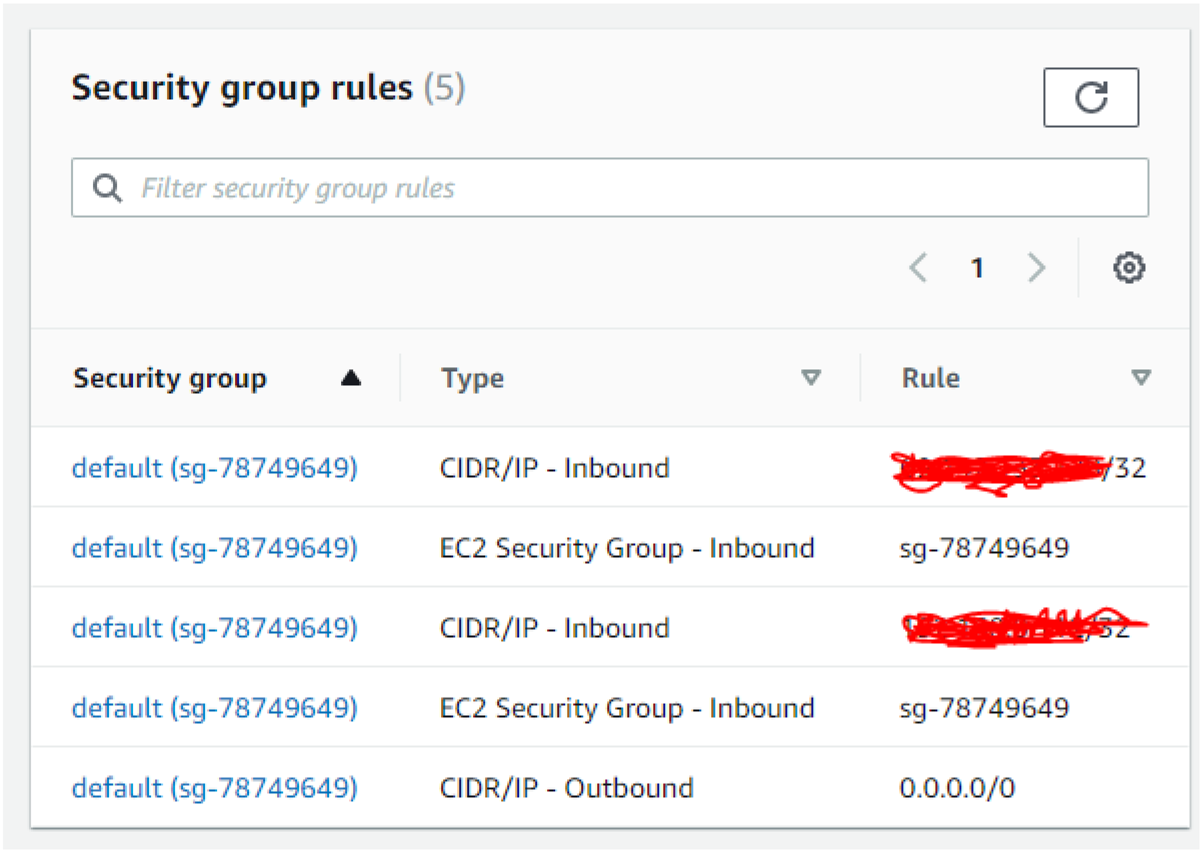
The code for this page is in *index.html*

Working with AWS Databases: Lambda and Postgres
I created a PostgreSQL database in the us-west-2a region with the identifier of rds-postgresql-10mintutorial.


Summary

DB identifier	CPU	Status	Class
rds-postgresql-10mintutorial	-	Stopped	db.t2.micro
Role	Current activity	Engine	Region & AZ
Instance		PostgreSQL	us-west-2a

I added a couple of new rules to the security group to allow my IP address as the ocnnection was not working otherwise



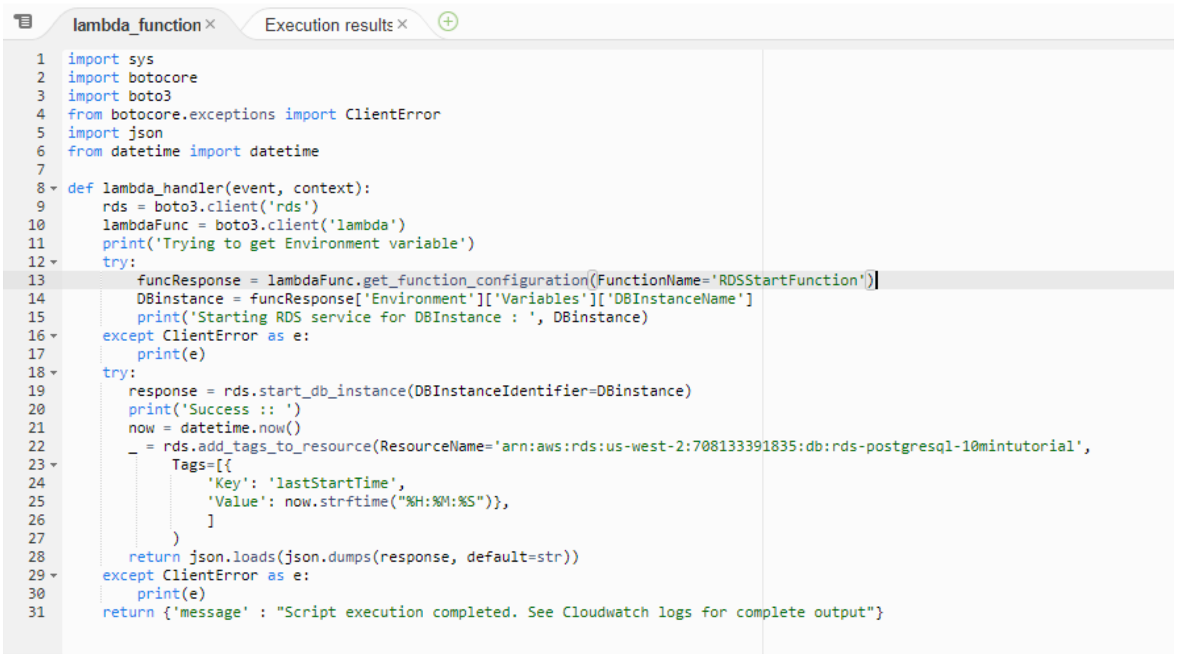
Security group rules (5)

Filter security group rules

Security group	Type	Rule
default (sg-78749649)	CIDR/IP - Inbound	[redacted]/32
default (sg-78749649)	EC2 Security Group - Inbound	sg-78749649
default (sg-78749649)	CIDR/IP - Inbound	[redacted]
default (sg-78749649)	EC2 Security Group - Inbound	sg-78749649
default (sg-78749649)	CIDR/IP - Outbound	0.0.0.0/0

I also created two lambda functions, RDSStartFunction and RDSStopFunction, to start/stop the database automatically. The RDSStartFunction also adds tags to the cluster before it starts, as specified by one of the bonus tasks. It adds the last time the cluster was started

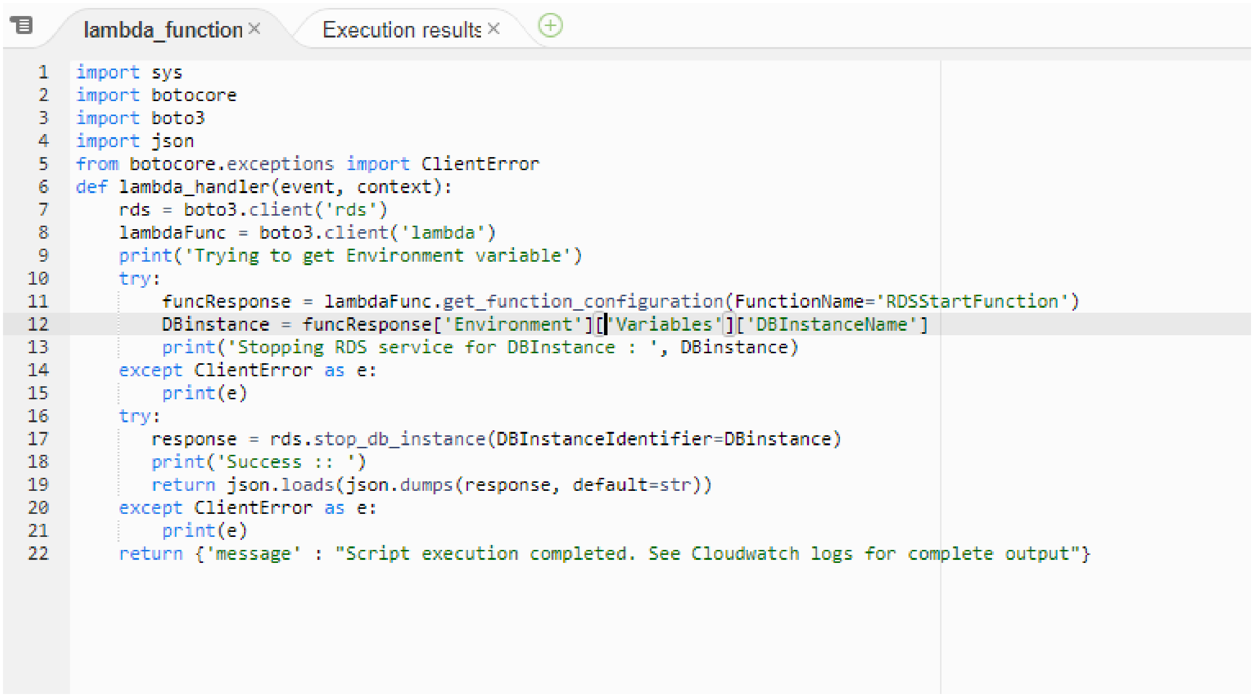
RDSStartFunction:



```python
import sys
import botocore
import boto3
from botocore.exceptions import ClientError
import json
from datetime import datetime

def lambda_handler(event, context):
    rds = boto3.client('rds')
    lambdaFunc = boto3.client('lambda')
    print('Trying to get Environment variable')
    try:
        funcResponse = lambdaFunc.get_function_configuration(FunctionName='RDSStartFunction')
        DBinstance = funcResponse['Environment']['Variables']['DBInstanceName']
        print('Starting RDS service for DBInstance : ', DBinstance)
    except ClientError as e:
        print(e)
    try:
       response = rds.start_db_instance(DBInstanceIdentifier=DBinstance)
       print('Success :: ')
       now = datetime.now()
       _ = rds.add_tags_to_resource(ResourceName='arn:aws:rds:us-west-2:708133391835:db:rds-postgresql-10mintutorial',
            Tags=[{
                'Key': 'lastStartTime',
                'Value': now.strftime("%H:%M:%S")},
                ]
            )
       return json.loads(json.dumps(response, default=str))
    except ClientError as e:
        print(e)
    return {'message' : "Script execution completed. See Cloudwatch logs for complete output"}
```

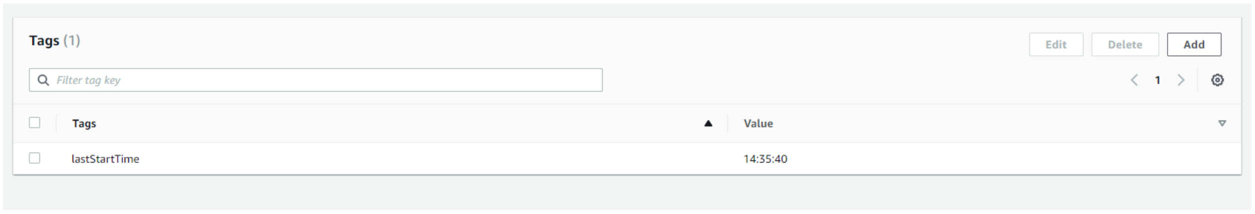
RDSStopFunction:



```python
import sys
import botocore
import boto3
import json
from botocore.exceptions import ClientError
def lambda_handler(event, context):
    rds = boto3.client('rds')
    lambdaFunc = boto3.client('lambda')
    print('Trying to get Environment variable')
    try:
        funcResponse = lambdaFunc.get_function_configuration(FunctionName='RDSStartFunction')
        DBinstance = funcResponse['Environment']['Variables']['DBInstanceName']
        print('Stopping RDS service for DBInstance : ', DBinstance)
    except ClientError as e:
        print(e)
    try:
       response = rds.stop_db_instance(DBInstanceIdentifier=DBinstance)
       print('Success :: ')
       return json.loads(json.dumps(response, default=str))
    except ClientError as e:
        print(e)
    return {'message' : "Script execution completed. See Cloudwatch logs for complete output"}
```

I also created two CloudWatch rules, called startrds and stoprds, to start and stop the cluster automatically. The startrds rule targets the RDSStartFunction Lambda function and is scheduled to be called once a day at 14:35 UTC. The stoprds similarly targets the RDSStopFunction and is scheduled once a day at 5:00 UTC.

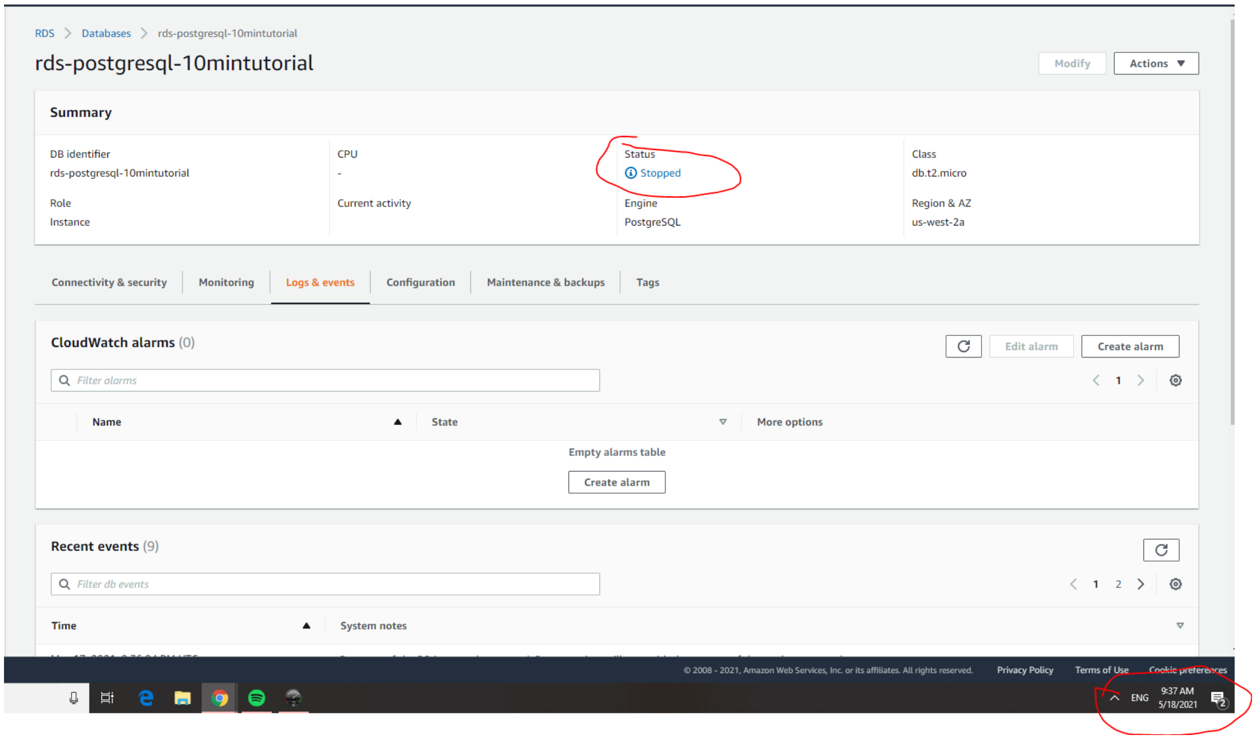
Through the tags we can see that the last time the database started was indeed 14:35 UTC:



Tags (1)

Tags	Value
lastStartTime	14:35:40

Currently, at the time of writing this report, it is 9:37AM ET. We can see how the RDS has been stopped:



RDS > Databases > rds-postgresql-10mintutorial

rds-postgresql-10mintutorial

Summary

DB identifier rds-postgresql-10mintutorial	CPU -	Status Stopped	Class db.t2.micro
Role Instance	Current activity	Engine PostgreSQL	Region & AZ us-west-2a

Connectivity & security | Monitoring | Logs & events | Configuration | Maintenance & backups | Tags

CloudWatch alarms (0)

Name	State	More options

Empty alarms table

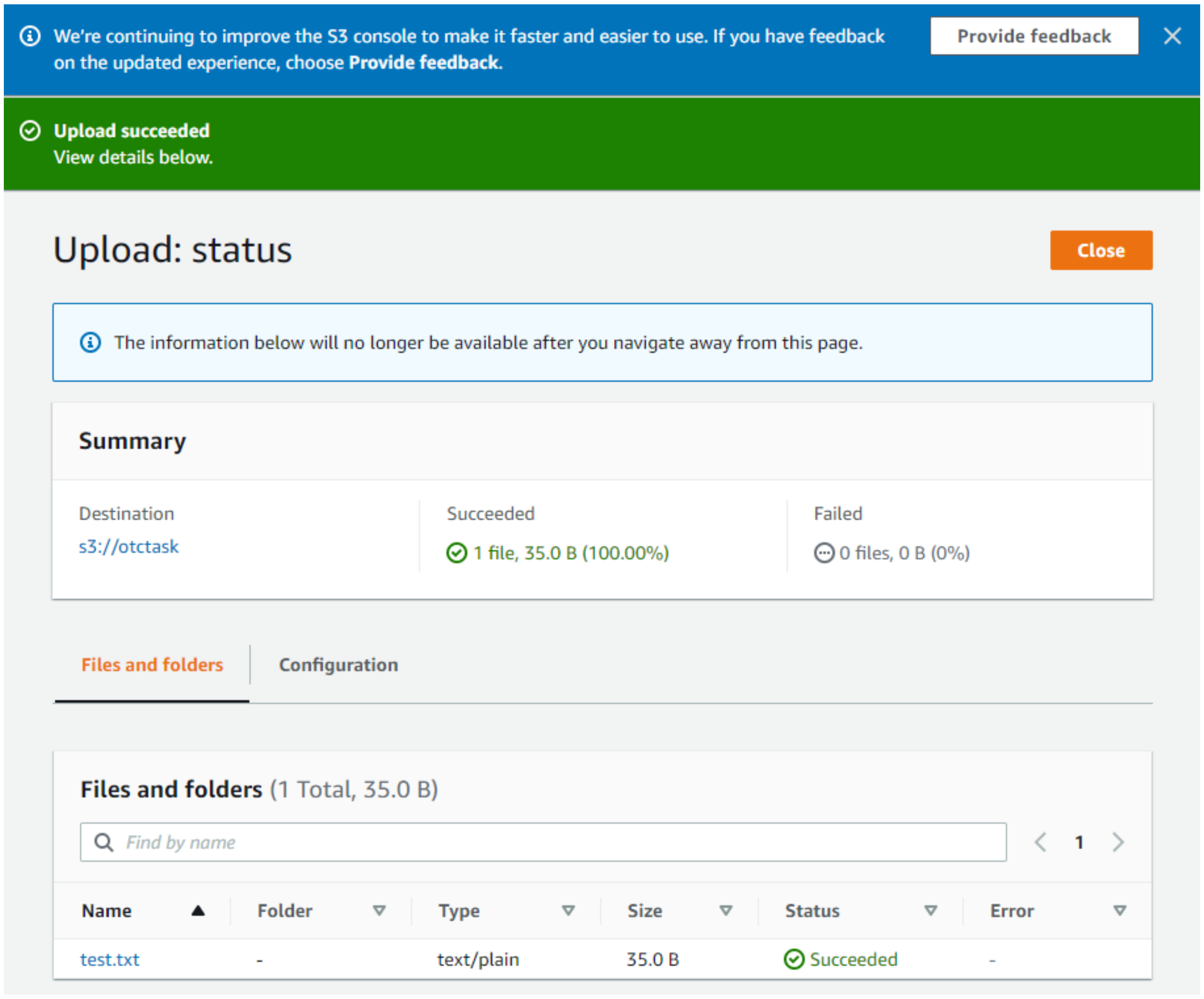
Recent events (9)

Time	System notes

© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Working with S3 and SNS

Uploading a FIle:


We're continuing to improve the S3 console to make it faster and easier to use. If you have feedback on the updated experience, choose **Provide feedback**.

Upload succeeded
View details below.

Upload: status

The information below will no longer be available after you navigate away from this page.

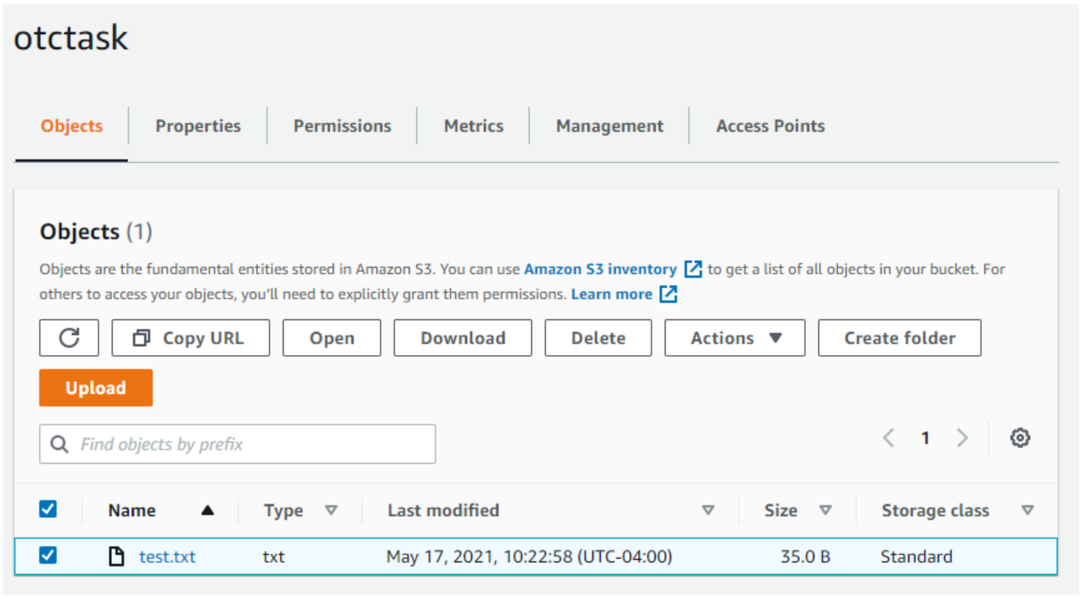
Summary

Destination	Succeeded	Failed
s3://otctask	✓ 1 file, 35.0 B (100.00%)	0 files, 0 B (0%)

Files and folders (1 Total, 35.0 B)

Name	Folder	Type	Size	Status	Error
test.txt	-	text/plain	35.0 B	✓ Succeeded	-

Downloading an Object:



otctask

Objects | Properties | Permissions | Metrics | Management | Access Points

Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use **Amazon S3 inventory** to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. **Learn more**

Copy URL | Open | Download | Delete | Actions | Create folder

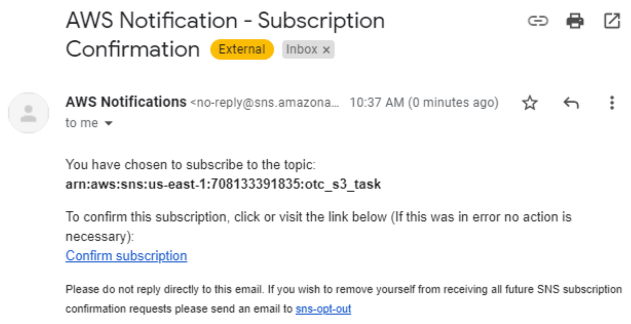
Upload

Find objects by prefix

✓	Name	Type	Last modified	Size	Storage class
✓	test.txt	txt	May 17, 2021, 10:22:58 (UTC-04:00)	35.0 B	Standard

This was saved as *test-dl.txt*

Creating Email Notifications:



AWS Notification - Subscription Confirmation External Inbox ×

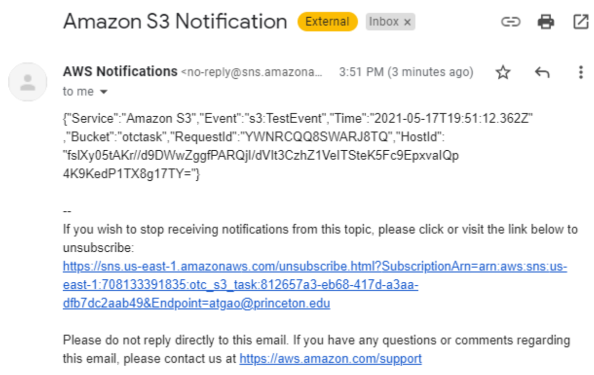
AWS Notifications <no-reply@sns.amazona... 10:37 AM (0 minutes ago)
to me

You have chosen to subscribe to the topic:
arn:aws:sns:us-east-1:708133391835:otc_s3_task

To confirm this subscription, click or visit the link below (If this was in error no action is necessary):
Confirm subscription

Please do not reply directly to this email. If you wish to remove yourself from receiving all future SNS subscription confirmation requests please send an email to sns-opt-out

I then uploaded a txt file called *email.txt*. Afterwards I deleted this file and got the following notification:



Amazon S3 Notification External Inbox ×

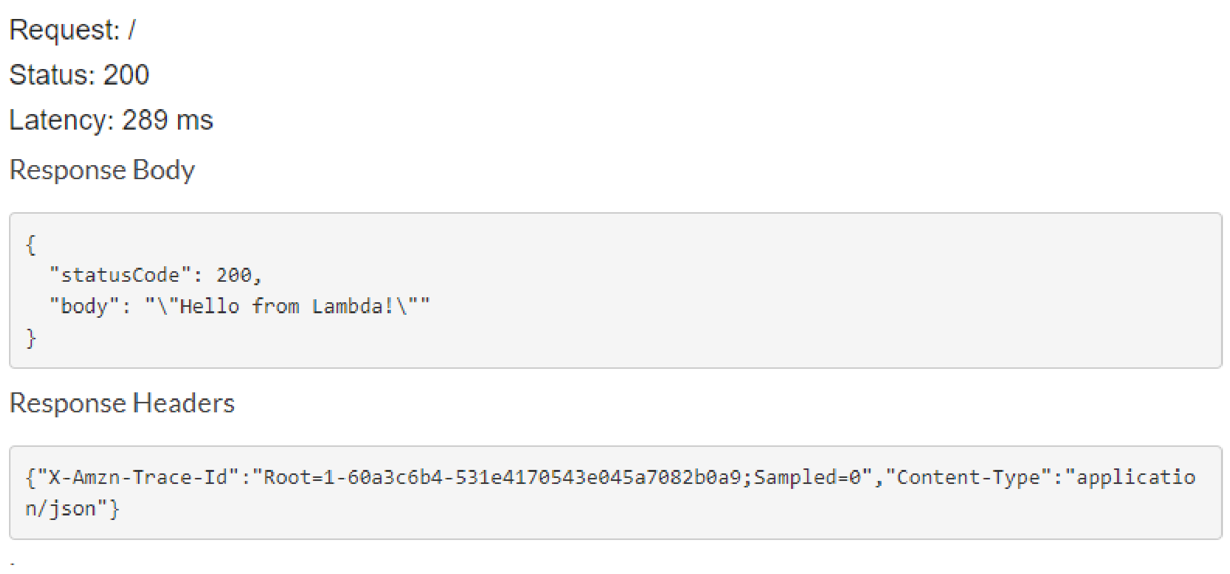
AWS Notifications <no-reply@sns.amazona... 3:51 PM (3 minutes ago)
to me

```
{"Service":"Amazon S3","Event":"s3:TestEvent","Time":"2021-05-17T19:51:12.362Z"
,"Bucket":"otctask","RequestId":"YWNRCQQ8SWARJ8TQ","HostId":
"fslXy05tAKr//d9DWwZggfPARQjl/dVlt3CzhZ1VeITSteK5Fc9EpxvalQp
4K9KedP1TX8g17TY="}
```

--
If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
https://sns.us-east-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-east-1:708133391835:otc_s3_task:812657a3-eb68-417d-a3aa-dfb7dc2aab49&Endpoint=atgao@princeton.edu

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at https://aws.amazon.com/support

Working with API Gateway and Lambda:
Following the tutorial I created an API called Python Function API. Here is the successful test result:



Request: /

Status: 200

Latency: 289 ms

Response Body

```
{
  "statusCode": 200,
  "body": "\"Hello from Lambda!\""
}
```

Response Headers

```
{"X-Amzn-Trace-Id":"Root=1-60a3c6b4-531e4170543e045a7082b0a9;Sampled=0","Content-Type":"applicatio
n/json"}
```

Gathering Website Information with Python:
I put the code to scrap the MIDI files into *parser.py*. It can be simply run calling python *parser.py*.