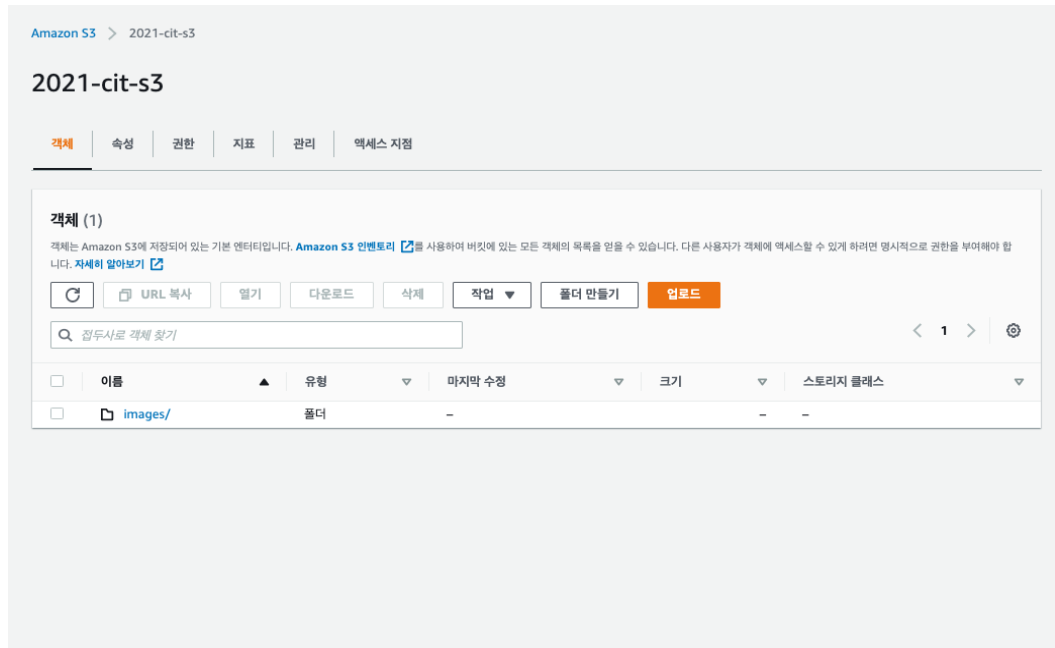


HW2_보고서

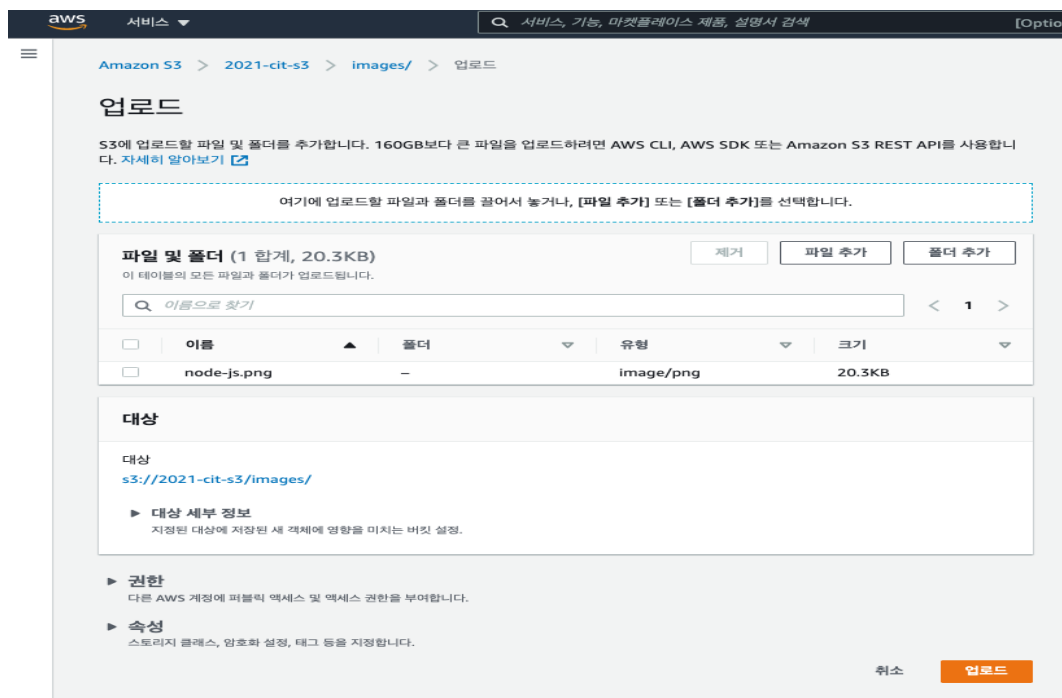
컴퓨터공학부 201814121 이연희

1. Task #1

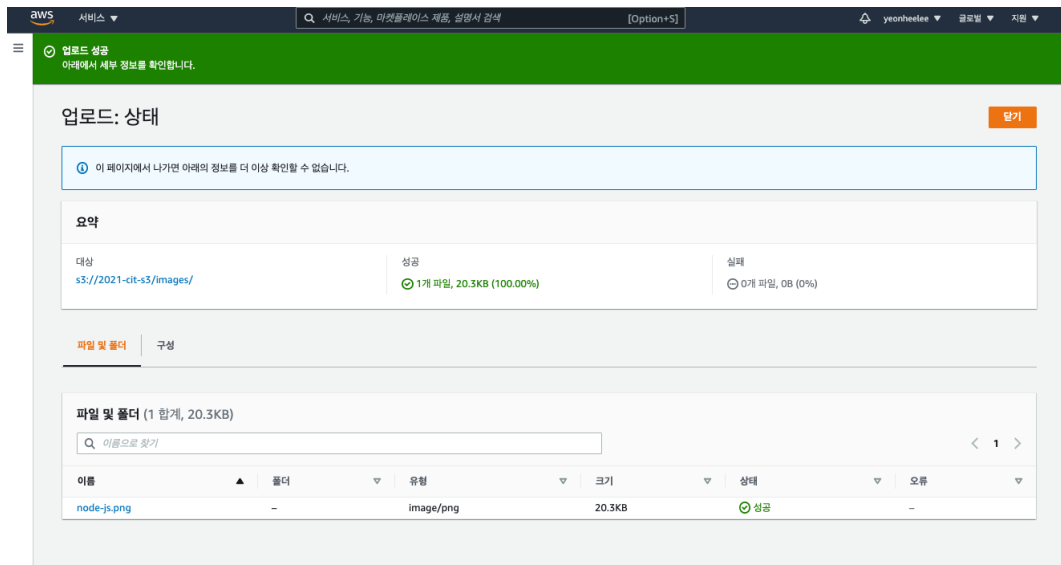
- 1) AWS 웹 콘솔에서 ‘2021-cit-s3’ 버킷과 이미지를 저장할 폴더 ‘images’를 생성했다.



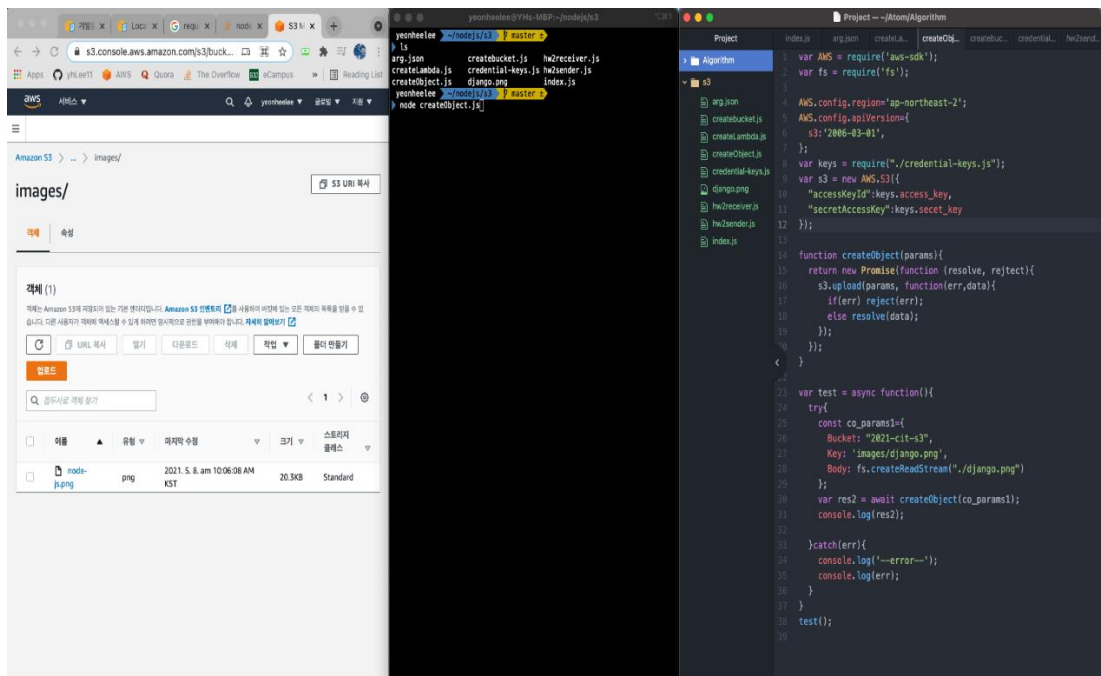
- 2) 웹 콘솔 상에서 노트북에 존재하는 node-js.png 파일을 추가했다.



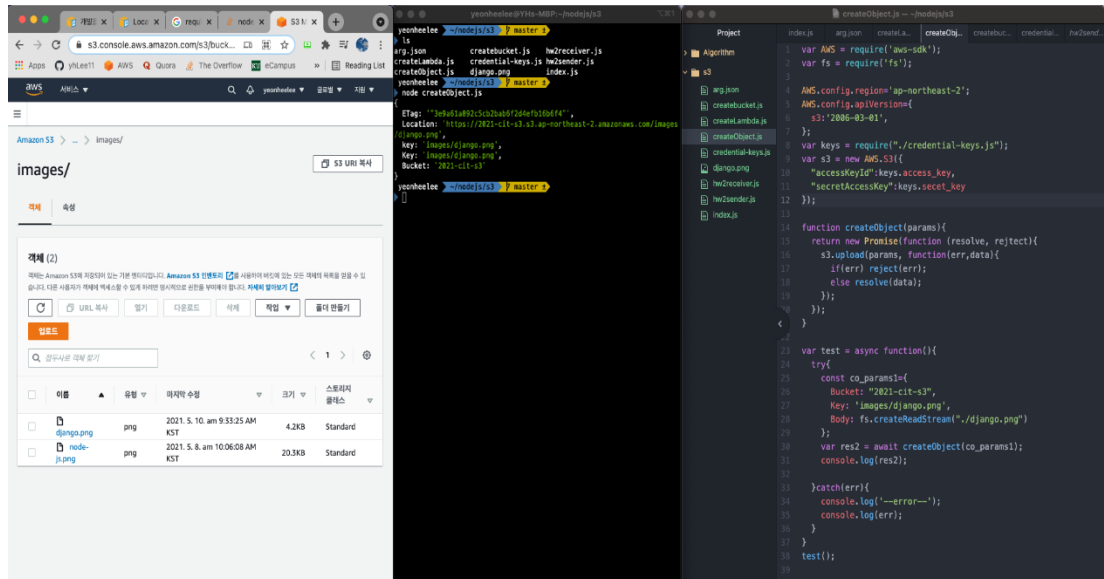
- 3) 2021-cit-s3 버킷의 images 폴더 안에 node-js.png 파일이 업로드 된 것을 확인할 수 있다.



- 4) 노트북의 로컬 환경에서 django.png 파일을 업로드하는 createObject.js를 실행시켰다. accessKeyId와 secretAccessKey 정보는 credential-keys.js를 생성하여 export하여 사용했다.



- 5) 2021-cit-s3 버킷 images 폴더 안에 django.png 파일이 업로드 된 것을 확인할 수 있다.



- 6) createObject.js 코드

```
var AWS = require('aws-sdk');
var fs = require('fs');

AWS.config.region='ap-northeast-2';
AWS.config.apiVersion={
  s3: '2006-03-01',
};

var keys = require("../credential-keys.js");
var s3 = new AWS.S3({
  "accessKeyId":keys.access_key,
  "secretAccessKey":keys.secet_key
});

function createObject(params){
  return new Promise(function (resolve, reject){
    s3.upload(params, function(err,data){
      if(err) reject(err);
      else resolve(data);
    });
  });
}

var test = async function(){
  try{
    const co_params1={
      Bucket: "2021-cit-s3",
      Key: 'images/django.png',
      Body: fs.createReadStream("../django.png")
    };
    var res2 = await createObject(co_params1);
    console.log(res2);
  }catch(err){
    console.log("error");
    console.log(err);
  }
}

test();
```

```

};
var res2 = await createObject(co_params1);
console.log(res2);
} catch(err){
  console.log('--error--');
  console.log(err);
}
}
test();

```

2. Task #2

- 1) index.js 파일을 생성한 후 credentials, node_module, package-lock.json 파일을 myLambda.zip으로 만들었다.

The screenshot shows a terminal window with the following commands and output:

```

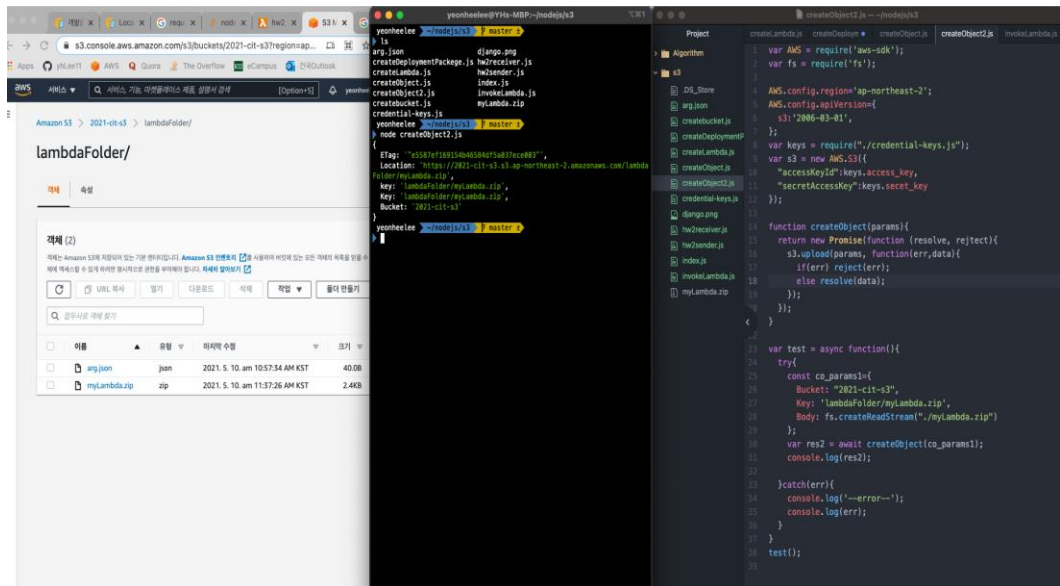
yeonheelee@YHs-MBP:~/nodejs/s3
ls
arg.json      credential-keys.js index.js
createLambda.js credentials      invokeLambda.js
createObject.js django.png      node_modules
createPackage.js hw2receiver.js package-lock.json
createbucket.js hw2sender.js

yeonheelee@YHs-MBP:~/nodejs/s3
zip myLambda.zip index.js package-lock.json credentials node_modules
adding: index.js (deflated 51%)
adding: package-lock.json (deflated 81%)
adding: credentials (deflated 9%)
adding: node_modules/ (stored 0%)

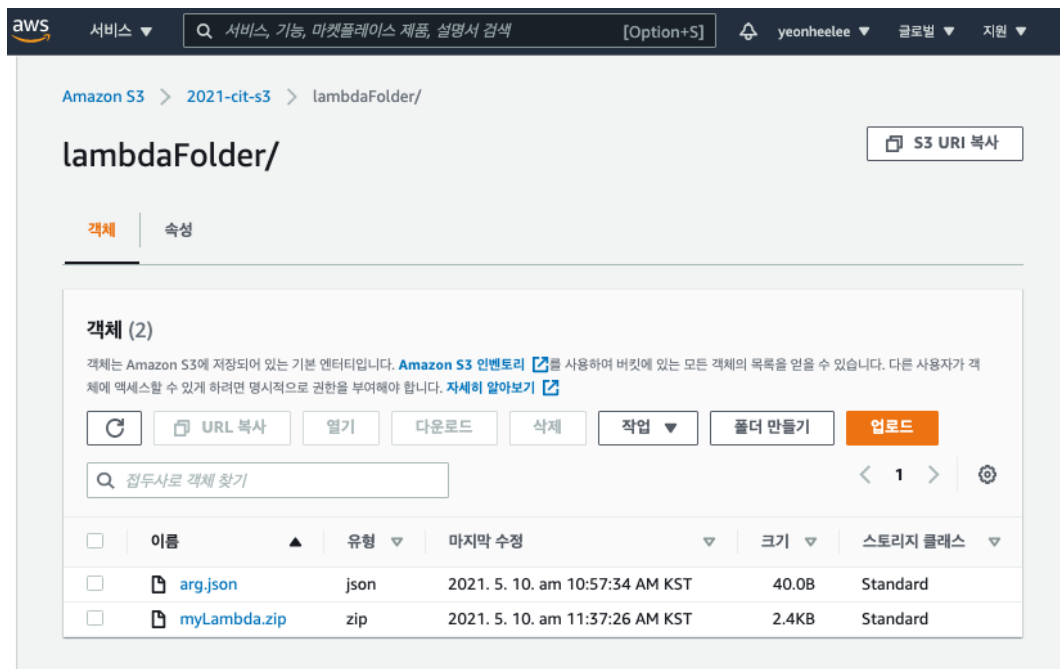
yeonheelee@YHs-MBP:~/nodejs/s3
ls
arg.json      credential-keys.js index.js
createLambda.js credentials      invokeLambda.js
createObject.js django.png      myLambda.zip
createPackage.js hw2receiver.js node_modules
createbucket.js hw2sender.js package-lock.json

```

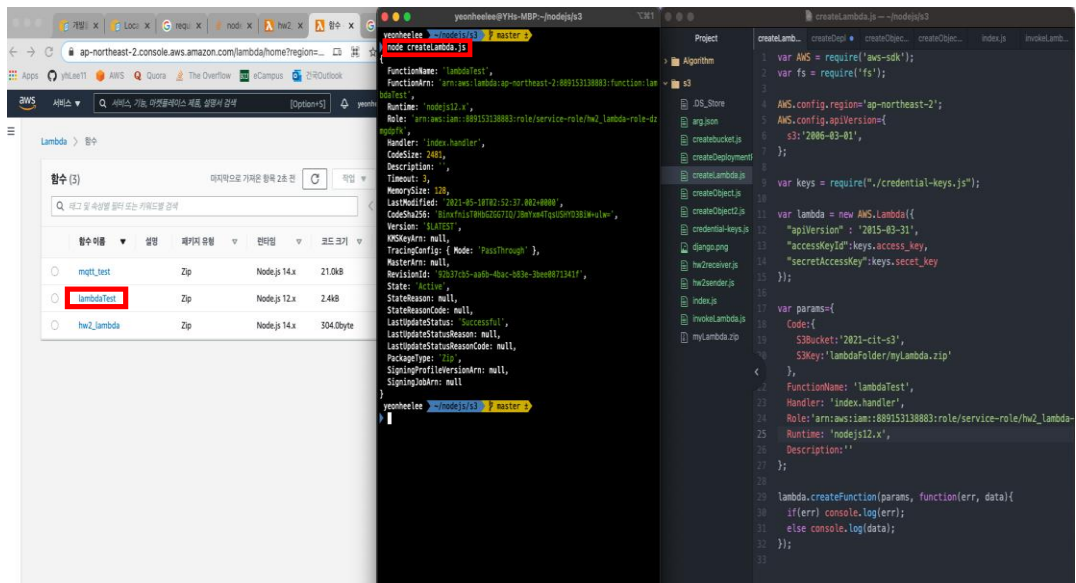
- 2) myLambda.zip을 버킷에 업로드하기 위해 createObject2.js를 생성하고 코드를 실행시켰다. accessKeyId와 secretAccessKey 정보는 credential-keys.js를 생성하여 export하여 사용했다.



- 3) lambdaFolder안에 myLambda.zip이 업로드 된 것을 확인할 수 있다.



- 4) 로컬 환경에서 lambdaTest 함수를 생성하기 위해 createLambda.js를 실행시켰다.
AWS lambda 페이지에 lambdaTest가 생성된 것을 확인할 수 있다.



- 5) index.js 코드

```
exports.handler=async(event)=>{
  switch (event.op) {
    case '+': event.result = Number(event.la)+Number(event.ra);
    case '-': event.result = Number(event.la)-Number(event.ra);
    case '*': event.result = Number(event.la)*Number(event.ra);
    case '/': event.result = Number(event.la)/Number(event.ra);
  }
  const response = {
    statusCode: 200,
    body: JSON.stringify(event)
  };
  return response;
};
```

- 6) createObject2.js 코드

```
var AWS = require('aws-sdk');
var fs = require('fs');
AWS.config.region='ap-northeast-2';
AWS.config.apiVersion={
  s3:'2006-03-01',
};
var keys = require("./credential-keys.js");
```

```

var s3 = new AWS.S3({
  "accessKeyId":keys.access_key,
  "secretAccessKey":keys.secet_key
});

function createObject(params){
  return new Promise(function (resolve, reject){
    s3.upload(params, function(err,data){
      if(err) reject(err);
      else resolve(data);
    });
  });
}

var test = async function(){
  try{
    const co_params1={
      Bucket: "2021-cit-s3",
      Key: 'lambdaFolder/myLambda.zip',
      Body: fs.createReadStream("./myLambda.zip")
    };
    var res2 = await createObject(co_params1);
    console.log(res2);
  }catch(err){
    console.log('--error--');
    console.log(err);
  }
}
test();

```

7) createLambda.js 코드

```

var AWS = require('aws-sdk');
var fs = require('fs');

AWS.config.region='ap-northeast-2';
AWS.config.apiVersion={
  s3:'2006-03-01',
};

var keys = require("./credential-keys.js");

var lambda = new AWS.Lambda({
  "apiVersion" : '2015-03-31',
  "accessKeyId":keys.access_key,
  "secretAccessKey":keys.secet_key
});

```

```

var params={
  Code:{
    S3Bucket:'2021-cit-s3',
    S3Key:'lambdaFolder/myLambda.zip'
  },
  FunctionName: 'lambdaTest',
  Handler: 'index.handler',
  Role:'arn:aws:iam::889153138883:role/service-role/hw2_lambda-role-dzmgdpgfk',
  Runtime: 'nodejs12.x',
  Description:''
};

lambda.createFunction(params, function(err, data){
  if(err) console.log(err);
  else console.log(data);
});

```

3. Task #3

- 1) 위에서 생성한 람다 함수를 사용하기 위한 invokeLambda.js 코드를 생성하였다. 인자 값은 const exp로 선언하여 전달하였다. op:/ la: 2 ra:3의 값을 찾을 때 실행 결과이다.

The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with a folder named 's3' containing several files, including 'invokeLambda.js'. The code editor shows the content of 'invokeLambda.js', which is a JavaScript file that uses the AWS SDK to invoke a Lambda function. The code defines a constant 'exp' with the value 'op:/ la: 2 ra: 3' and a 'params' object with 'FunctionName' 'lambdaTest', 'InvocationType' 'RequestResponse', and 'Payload' 'JSON.stringify(exp)'. The code then calls 'lambda.invoke(params, function(err, data){...})'. The terminal on the left shows the command 'node invokeLambda.js' being executed, and the output is a JSON object: {'statusCode': 200, 'body': '{"op":"/", "la": 2, "ra": 3, "result": 0.6666666666666666}'}. The code editor on the right shows the following code:

```

1 var AWS = require('aws-sdk');
2 var fs = require('fs');
3
4 AWS.config.region='ap-northeast-2';
5 AWS.config.apiVersion={
6   s3: '2006-03-01',
7 };
8
9 var keys = require("./credential-keys.js");
10
11 var lambda = new AWS.Lambda({
12   "apiVersion" : '2015-03-31',
13   "accessKeyId":keys.access_key,
14   "secretAccessKey":keys.sectet_key
15 });
16
17 const exp={"op":"/", "la":2, "ra":3};
18 var params={
19   FunctionName: "lambdaTest",
20   InvocationType: "RequestResponse",
21   Payload: JSON.stringify(exp)
22 };
23 lambda.invoke(params, function(err,data){
24   if(err) console.log(err);
25   else console.log(JSON.parse(data.Payload))
26 });

```


2) invokeLambda.js 코드

```
var AWS = require('aws-sdk');
var fs = require('fs');
|
AWS.config.region='ap-northeast-2';
AWS.config.apiVersion={
  s3:'2006-03-01',
};
|
var keys = require("./credential-keys.js");
|
var lambda = new AWS.Lambda({
  "apiVersion" : '2015-03-31',
  "accessKeyId":keys.access_key,
  "secretAccessKey":keys.secet_key
});
|
const exp={"op":"+", "la":2, "ra":1};
var params={
  FunctionName: "lambdaTest",
  InvocationType: "RequestResponse",
  Payload: JSON.stringify(exp)
};
lambda.invoke(params, function(err,data){
  if(err) console.log(err);
  else console.log(JSON.parse(data.Payload))
});
```