

# Serverless Application Development on AWS

Bob Reselman

**Please, take the pre-session survey:**

<https://www.surveymonkey.com/r/Q699JMF>

# Brief Bio

InformationWeek  
**NETWORK**Computing

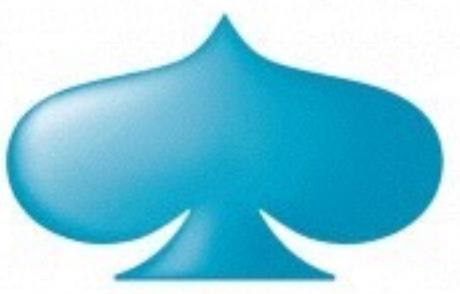
 DevOps.com

 **Gateway**™



SIMPSON  
COLLEGE

edmunds

  
**Capgemini**  
CONSULTING.TECHNOLOGY.OUTSOURCING

 logentries™

 Casting Networks  
INTERNATIONAL

# Brief Bio

## A Developer's Journey into Linux Containers

24 Sep 2015

Share: [Twitter](#) [Facebook](#) [Submit](#) [Link](#) 2



VMs? Containers?  
All I want to do is program!  
Jeesh.



I'll let you in on a secret: all that DevOps cloud stuff that goes into the world is still a bit of a mystery to me. But, over time I've come the ins and outs of large scale machine provisioning and application knowledge for a developer to have. It's akin to being a professional need know how to play your instrument. But, if you don't understand

ProgrammableWeb

API NEWS API DIRECTORY

LEARN ABOUT APIs

WHAT IS AN API?

API RESEARCH

WEATHER

MAPPING

## Why Messaging Queues Suck

API UNIVERSITY Analysis, Integration

Feb. 13 2017 By Bob Reselman CogArtTech



## DevOps testing: Never trust the world outside the enterprise



by  
Bob Reselman



Find out why a white-hat hacker claims "the biggest thing that keeps me up at night is the code DevOps is writing for the infrastructure," and what security pros can do about it.

### THIS ARTICLE COVERS

[DevOps](#)  
[Security](#)

### TECHNOLOGIES

[Code management](#) [DevOps](#)  
[Hacker](#) [Software testing & QA](#)  
[Test-Driven development](#)

### RELATED TOPICS

[Building a DevOps Culture](#)  
[DevOps and Software Development](#)

developer.com

Google Custom Search

Java Microsoft & .NET Mobile Android Open Source Cloud Database Architecture Other NEW

August 25, 2017 Hot Topics: [Android](#) [Java](#) [PHP](#) [Microsoft & .NET](#) [Cloud](#) [Open](#)

Developer.com [Architecture & Design](#)

[Read More in Architecture & Design »](#)

## Containers 101

September 25, 2015

By Bob Reselman

[Send Email »](#)

[More Articles »](#)

[Tweet](#)

About six months ago, I started to notice that there is a lot of hubbub going around in the tech-o-sphere about containers as a new way to approach virtual computing. I like exploring new technology, so I've spent the last few months getting the ins and outs of them. Here's what I can tell you: Containers are an important technology that is not going away anytime soon. There are a lot of players in the space, and new ones enter all the time. If you are a developer in the modern world, understanding and using containers are necessary skills to have in your professional life.

So, in the spirit of moving a good idea along, I am going to share with you the basics of container technology by answering the following questions:

- What are containers?
- What's so special about containers?
- How do I use them?

My desire in answering these question is to give you the basic knowledge and understanding that you need to start using containers when making and deploying code.

Let's get started.

### What Are Containers?

Container technology is a way to create a virtual environment by using an isolated process on a host computer. The isolated process, the container, has its own set of file system resources and subordinate processes. And the container does not intrude on the host system nor

[Post a comment](#)

@reselbob

# Agenda for Day 1

- **Session 1**
  - Setting Up An AWS Account
  - Creating the Class User & Assigning Permissions
  - Creating the Static Web Site on S3
- **Session 2**
  - Create Class Role
  - Create Lambda
  - View in CloudWatch
- **Session 3**
  - Create API Gateway Endpoints
  - Bind to Lambda
  - Testing the API
  - Creating the DynamoDB Table

# Agenda for Day 2

- **Session 4**

- Creating the POST Rating API Endpoint
- Creating the postRating Lambda Function
- Creating the Rating Index in DynamoDB

- **Session 5**

- Creating the getMovies Lambda Function
- Refactor the GET Movies API Endpoint
- Creating the getRating Lambda Function
- Creating the GET Ratings API Endpoint
- Publishing the API to a Test Deployment

- **Session 6**

- Refactoring the Movie Rater UI
- Creating and Binding the SNS Topic
- Subscribing to the MovieRater Topic

# Preliminaries

## GitHub Repo for Project

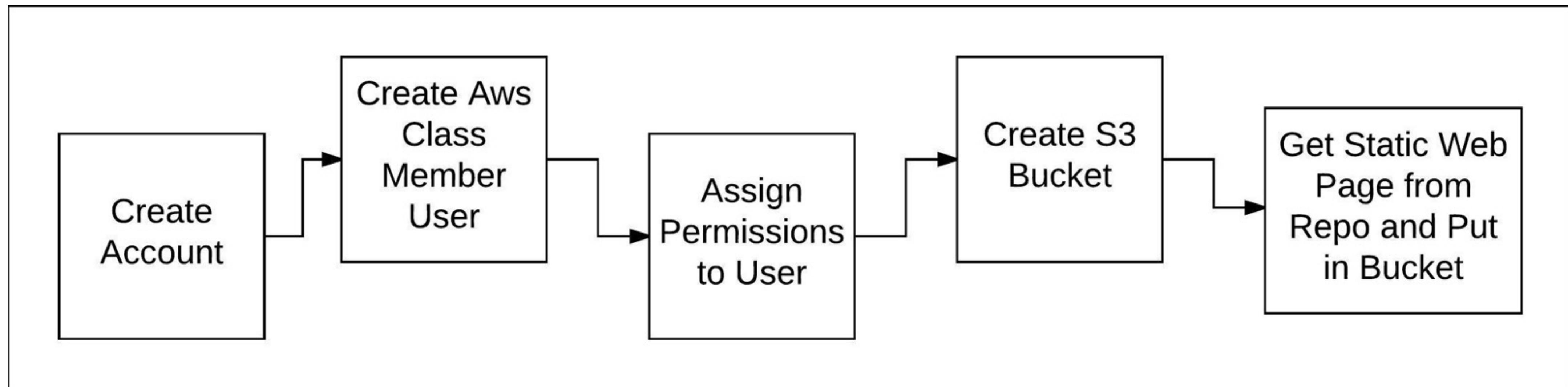
<https://github.com/reselbob/CDAwsClass>

## API for IMDB Movie Data

<https://www.omdbapi.com/>

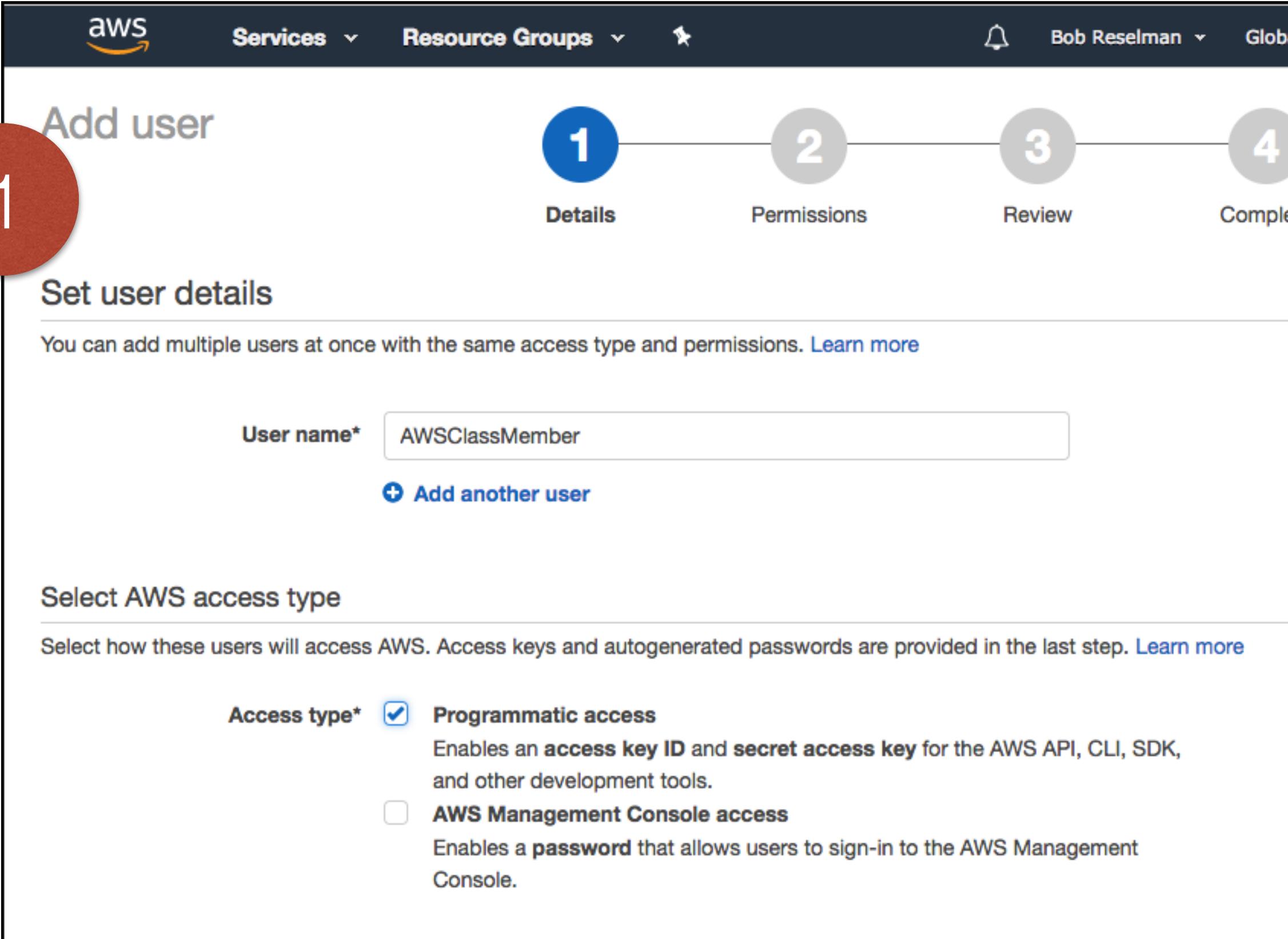
Sign up and get an API key

# Session 1



# Setting Up An AWS Account

# Creating the Class User & Assigning Permissions



1

Add user

1 Details    2 Permissions    3 Review    4 Complete

**Set user details**

You can add multiple users at once with the same access type and permissions. [Learn more](#)

User name\* AWSClassMember

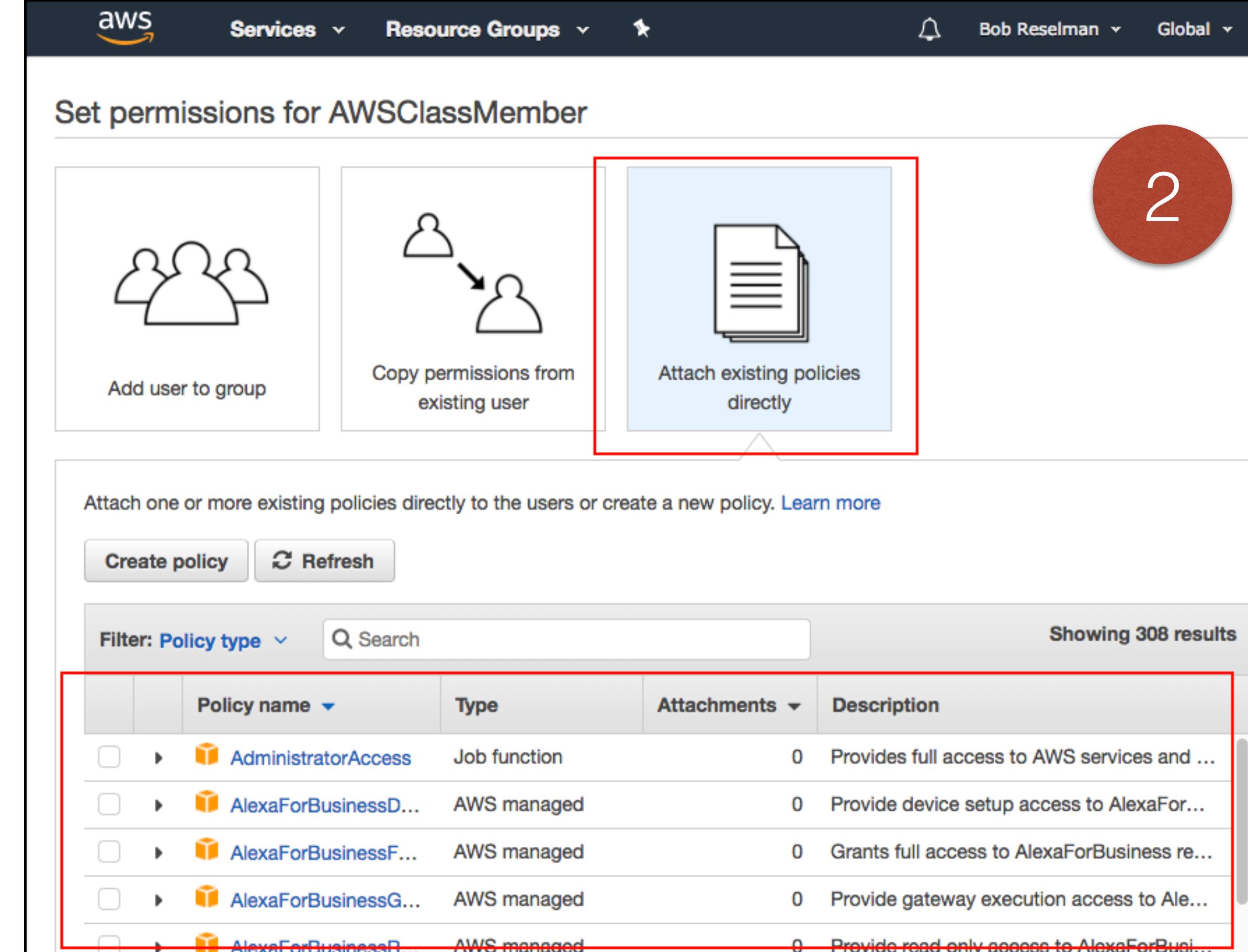
+ Add another user

**Select AWS access type**

Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. [Learn more](#)

Access type\*  **Programmatic access**  
Enables an access key ID and secret access key for the AWS API, CLI, SDK, and other development tools.

**AWS Management Console access**  
Enables a password that allows users to sign-in to the AWS Management Console.



2

Set permissions for AWSClassMember

Add user to group    Copy permissions from existing user    Attach existing policies directly

Attach one or more existing policies directly to the users or create a new policy. [Learn more](#)

Create policy    Refresh

Filter: Policy type ▾    Search    Showing 308 results

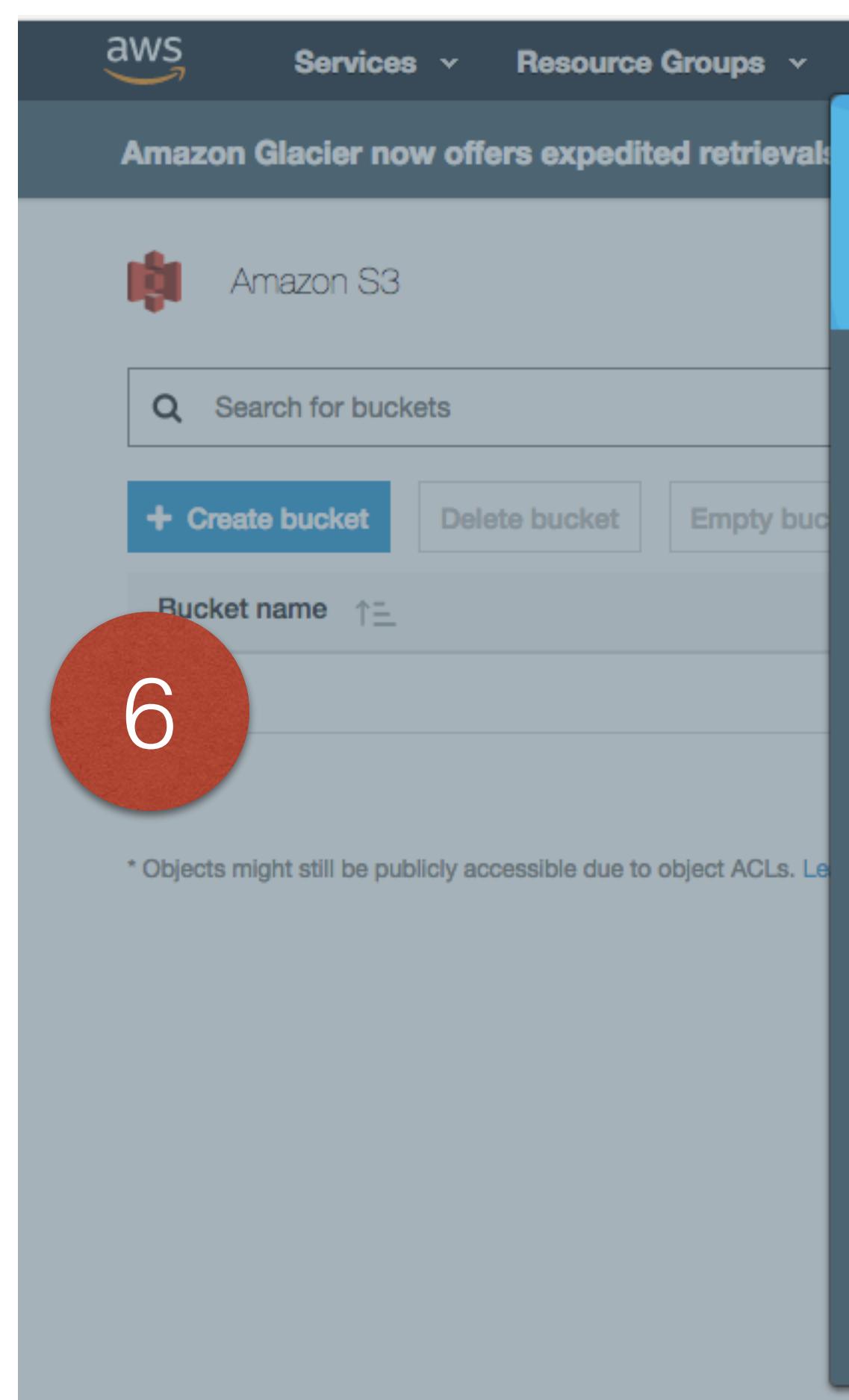
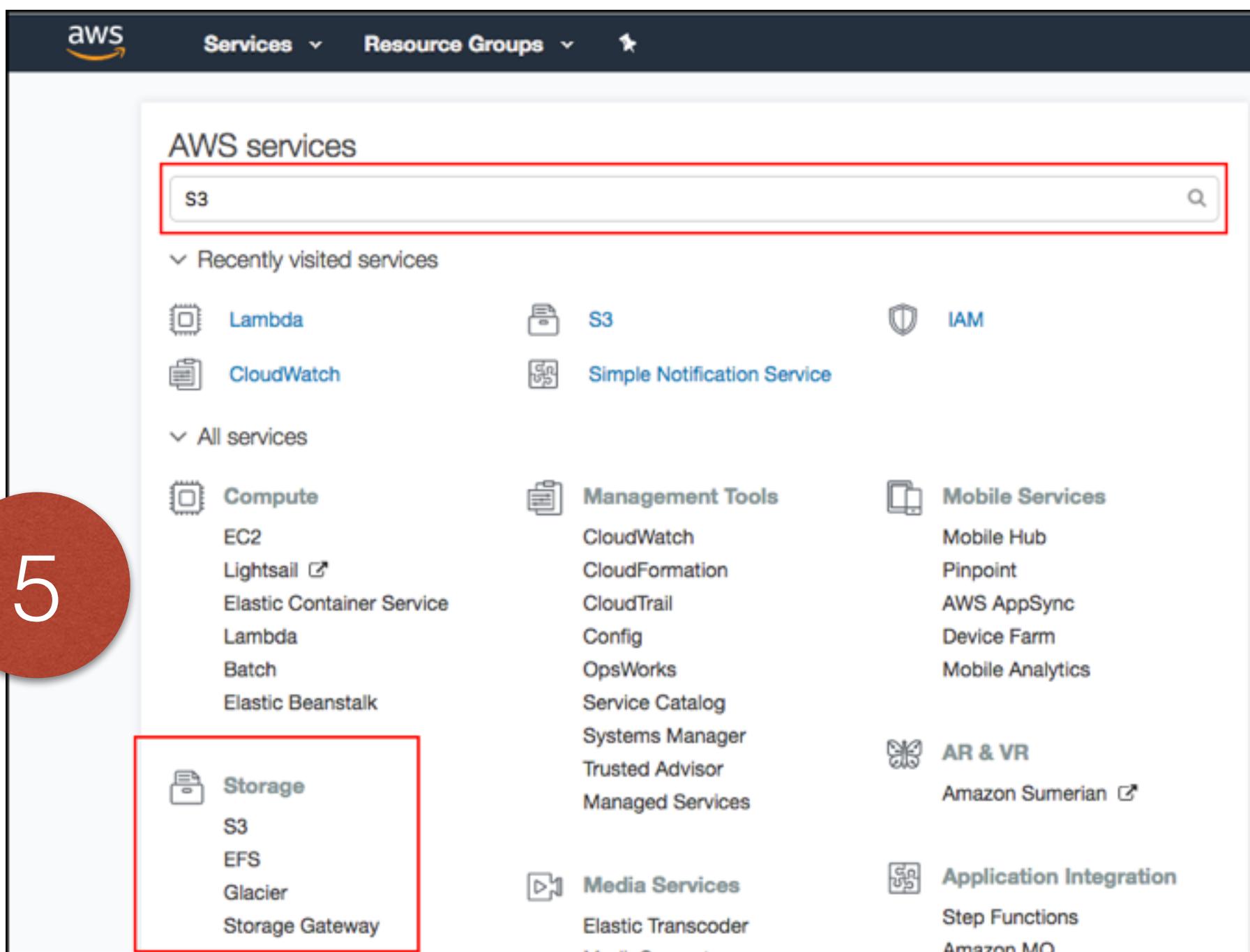
	Policy name ▾	Type	Attachments ▾	Description
<input type="checkbox"/>	AdministratorAccess	Job function	0	Provides full access to AWS services and ...
<input type="checkbox"/>	AlexaForBusinessD...	AWS managed	0	Provide device setup access to AlexaFor...
<input type="checkbox"/>	AlexaForBusinessF...	AWS managed	0	Grants full access to AlexaForBusiness re...
<input type="checkbox"/>	AlexaForBusinessG...	AWS managed	0	Provide gateway execution access to Ale...
<input type="checkbox"/>	AlexaForBusinessP...	AWS managed	0	Provide read-only access to AlexaForBusi...

# Creating the Class User & Assigning Permissions

The screenshot shows the 'Add user' process at the 'Review' step. A red circle labeled '3' highlights the 'Review' button. The navigation bar shows steps 1 (Details), 2 (Permissions), and 3 (Review). The 'User details' section shows a user named 'AWSClassMember' with 'Programmatic access - with an access key'. The 'Permissions summary' section, highlighted by a red box, lists five managed policies: AmazonSNSFullAccess, AmazonSQSFullAccess, AWSLambdaExecute, AmazonAPIGatewayAdministrator, and AmazonS3FullAccess.

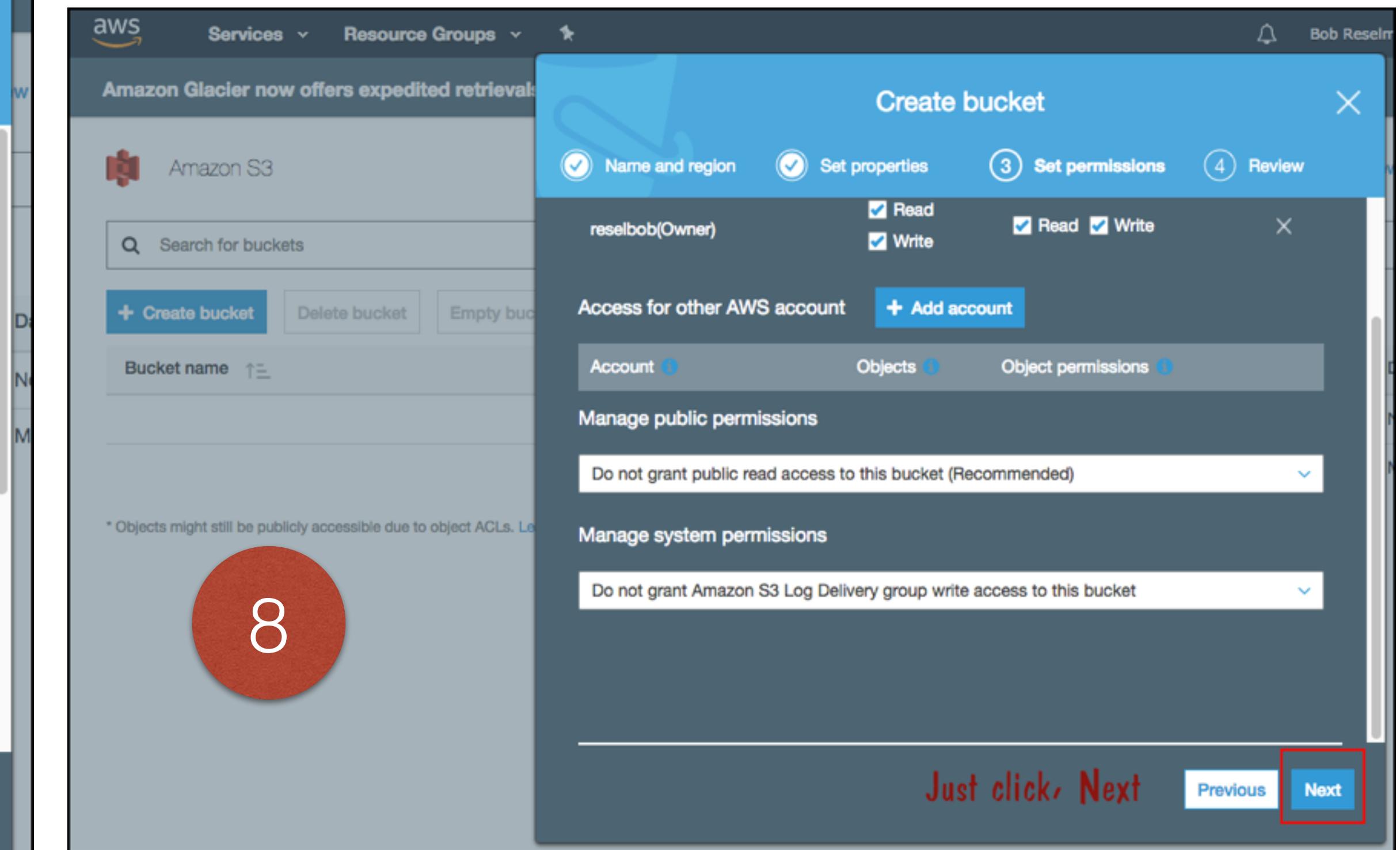
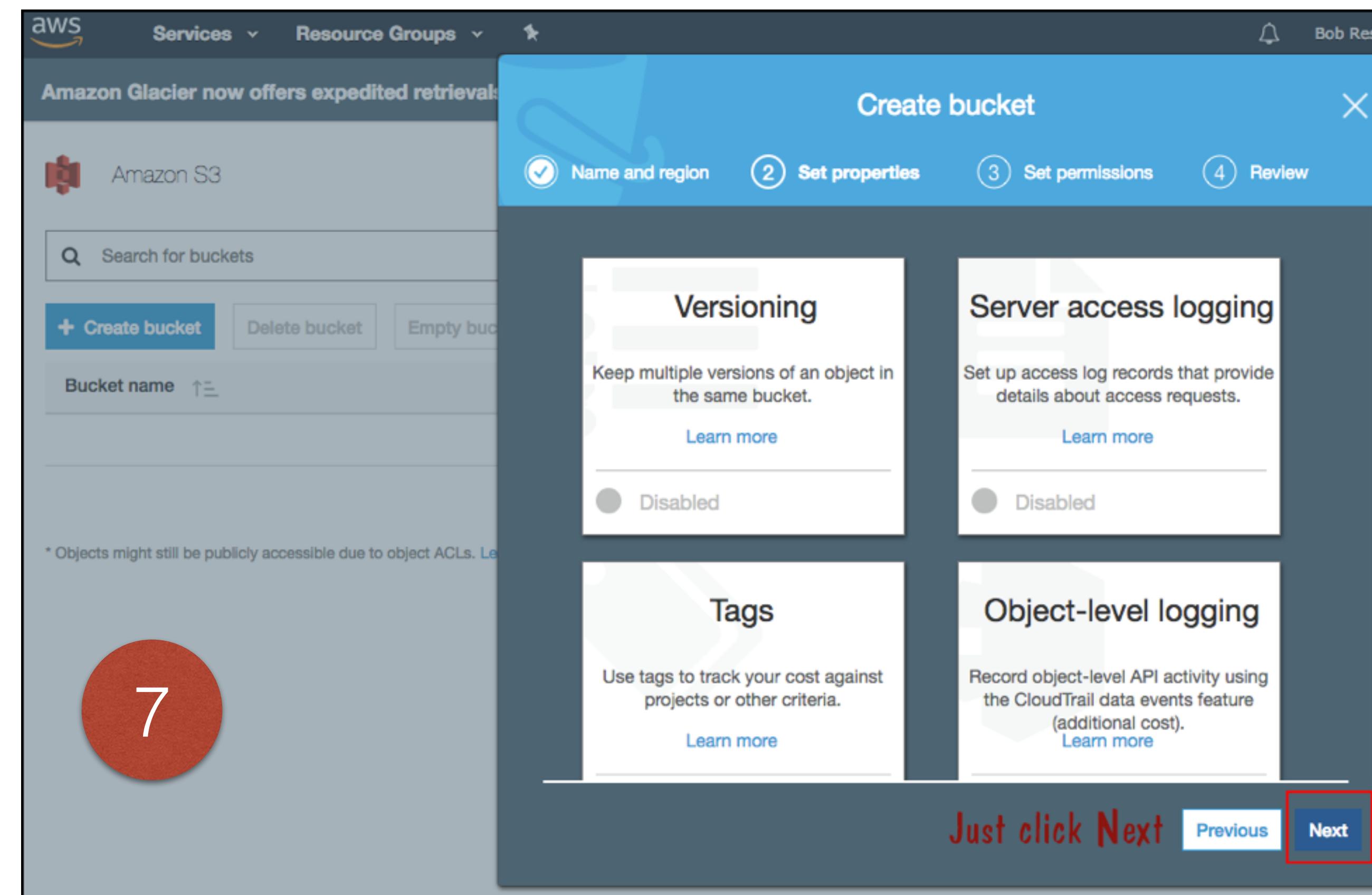
The screenshot shows the 'Add user' process at the 'Complete' step. A red circle labeled '4' highlights the 'Success' message. The message states: 'You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.' Below the message is a 'Download .csv' button. A table displays the created user 'AWSClassMember' with its 'Access key ID' (AKIAINEOK4BKP3PIQGXQ) and 'Secret access key' (\*\*\*\*\* Show). The 'User' column and the 'Access key ID' and 'Secret access key' columns are highlighted by red boxes.

# Create the S3 Bucket

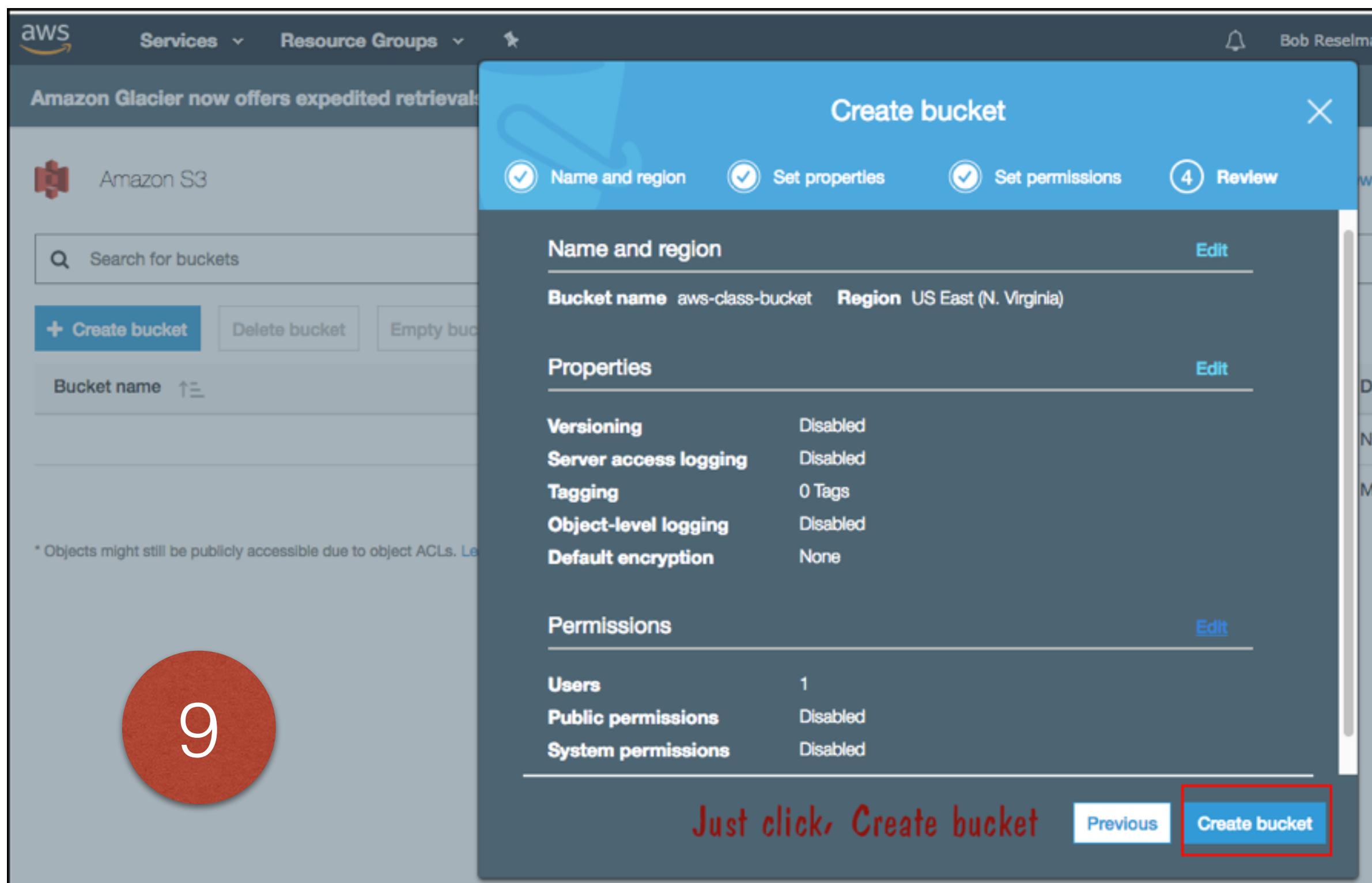


The screenshot shows the "Create bucket" wizard, Step 1: Name and region. The "Bucket name" field contains "aws-class-bucket", which is highlighted with a red box. The "Region" dropdown is set to "US East (N. Virginia)". The "Copy settings from an existing bucket" section is shown below, with a "Select bucket (optional)" dropdown containing "2 Buckets".

# Create the S3 Bucket



# Create the S3 Bucket



10

The screenshot shows the AWS S3 buckets list. It displays three buckets: 'aws-class-bucket' (Not public, US East (N. Virginia), Nov 30, 2017 5:14:09 PM), 'Public' (Public, US East (N. Virginia), Nov 7, 2017 7:42:33 PM), and another bucket (Not public, US West (Oregon), May 26, 2016 8:58:18 AM). A red circle with the number '10' is overlaid on the top center of the screen. The 'aws-class-bucket' row is highlighted with a red border.

Bucket name	Access	Region	Date created
aws-class-bucket	Not public *	US East (N. Virginia)	Nov 30, 2017 5:14:09 PM
Public	Public	US East (N. Virginia)	Nov 7, 2017 7:42:33 PM
	Not public *	US West (Oregon)	May 26, 2016 8:58:18 AM

# Create the S3 Bucket

This screenshot shows the 'Properties' tab of an S3 bucket named 'aws-class-bucket'. The tab is highlighted with a red box. The page displays the location as 'US East (N. Virginia)'. A message at the top says, 'This bucket is empty. Upload new objects to get started.' Below the message are three icons: a bucket icon labeled 'Upload an object', a user icon labeled 'Set object', and a database icon labeled 'Set object'.

11

Amazon S3 > aws-class-bucket

Properties

Overview Permissions Management

Upload Create folder More

US East (N. Virginia)

This bucket is empty. Upload new objects to get started.

Upload an object Set object Set object

This screenshot shows the 'Properties' tab of the same S3 bucket. It highlights the 'Static website hosting' feature with a red box. Other features shown include 'Versioning' (disabled), 'Server access logging' (disabled), 'Object-level logging' (disabled), and 'Static website hosting' (disabled). Each feature has a 'Learn more' link.

12

Amazon S3 > aws-class-bucket

Overview Properties Permissions Management

Versioning

Keep multiple versions of an object in the same bucket.

Learn more

Disabled

Server access logging

Set up access log records that provide details about access requests.

Learn more

Disabled

Static website hosting

Host a static website, which does not require server-side technologies.

Learn more

Disabled

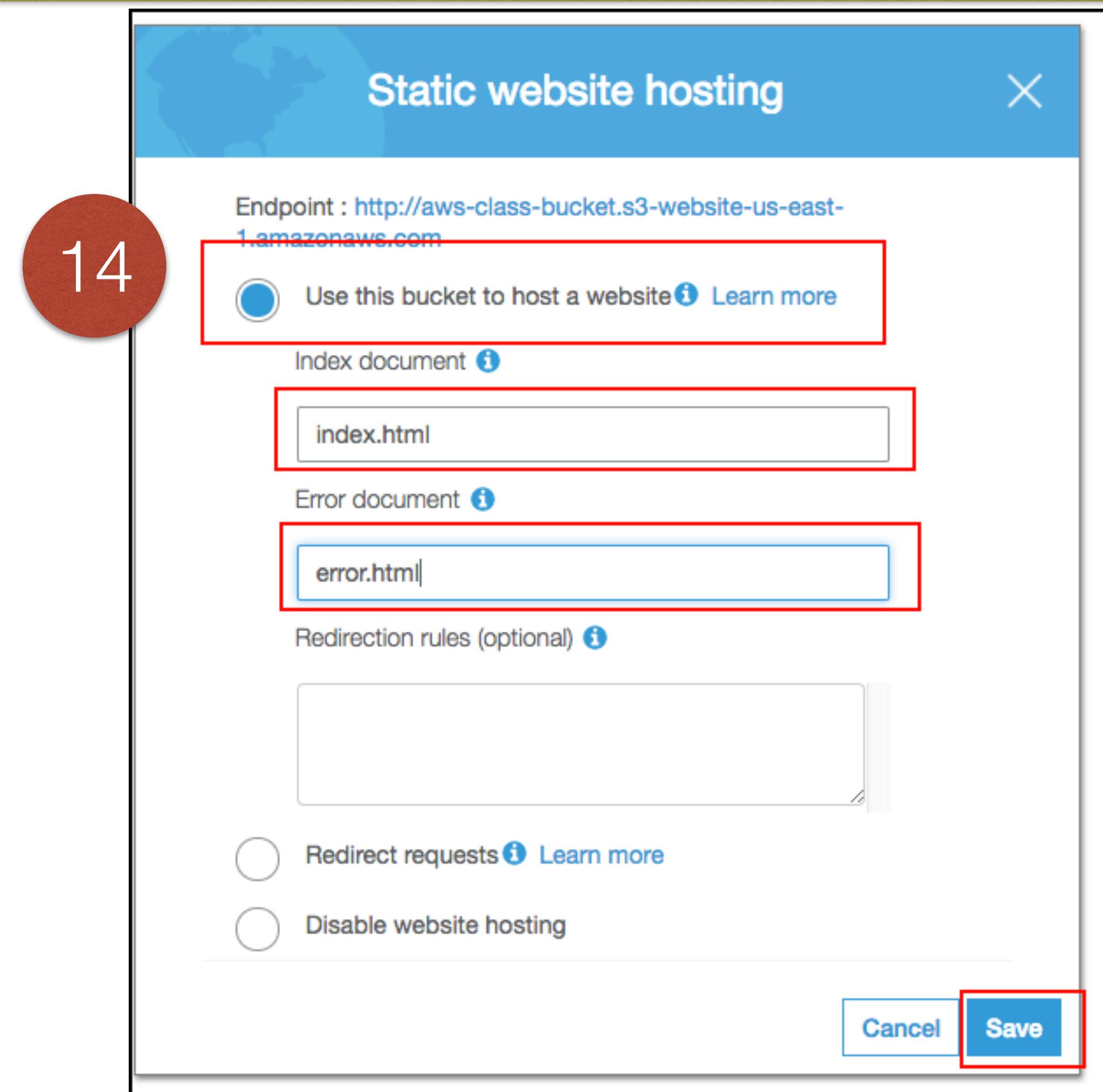
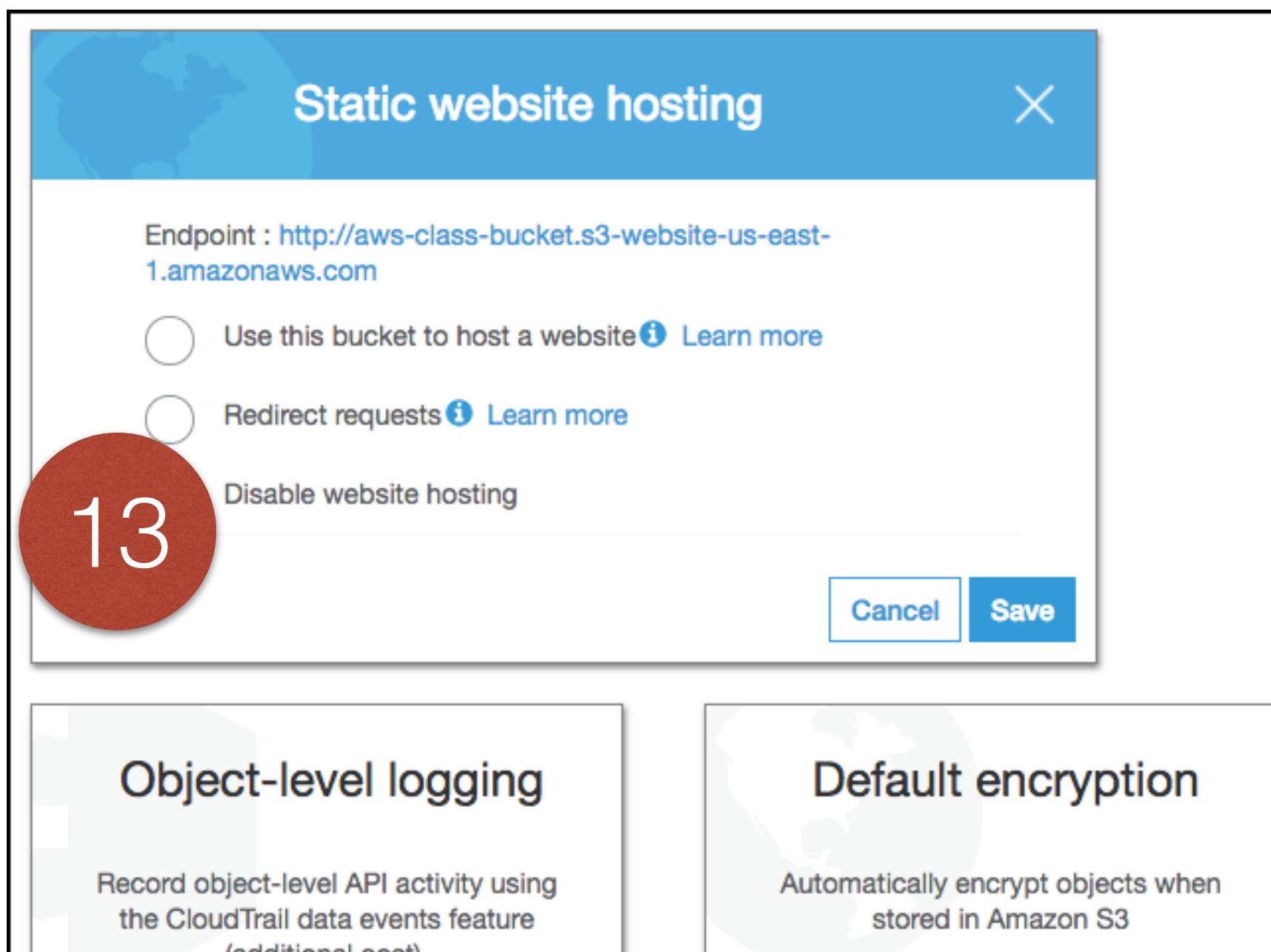
Object-level logging

Record object-level API activity using the CloudTrail data events feature (additional cost).

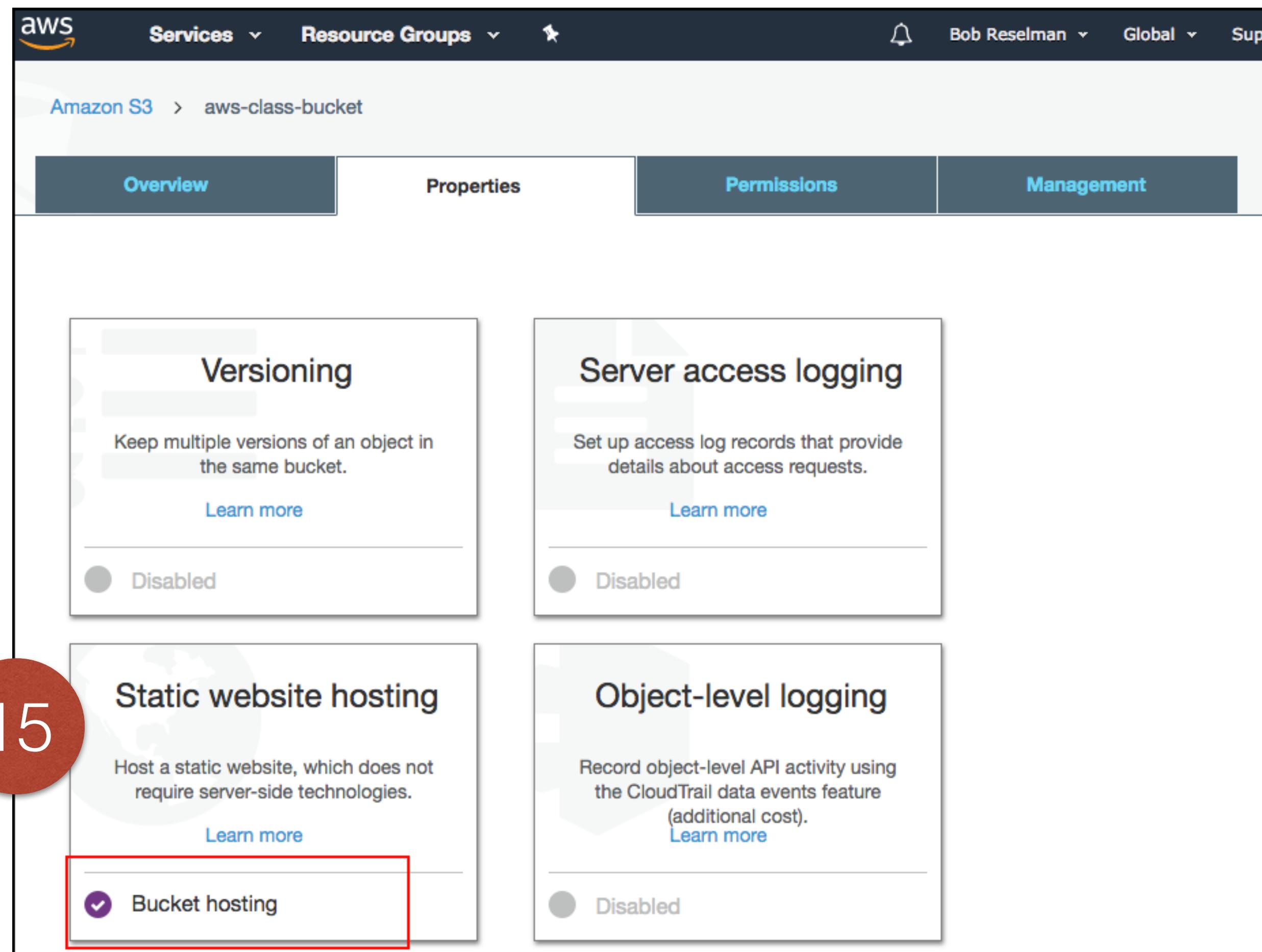
Learn more

Disabled

# Create the S3 Bucket



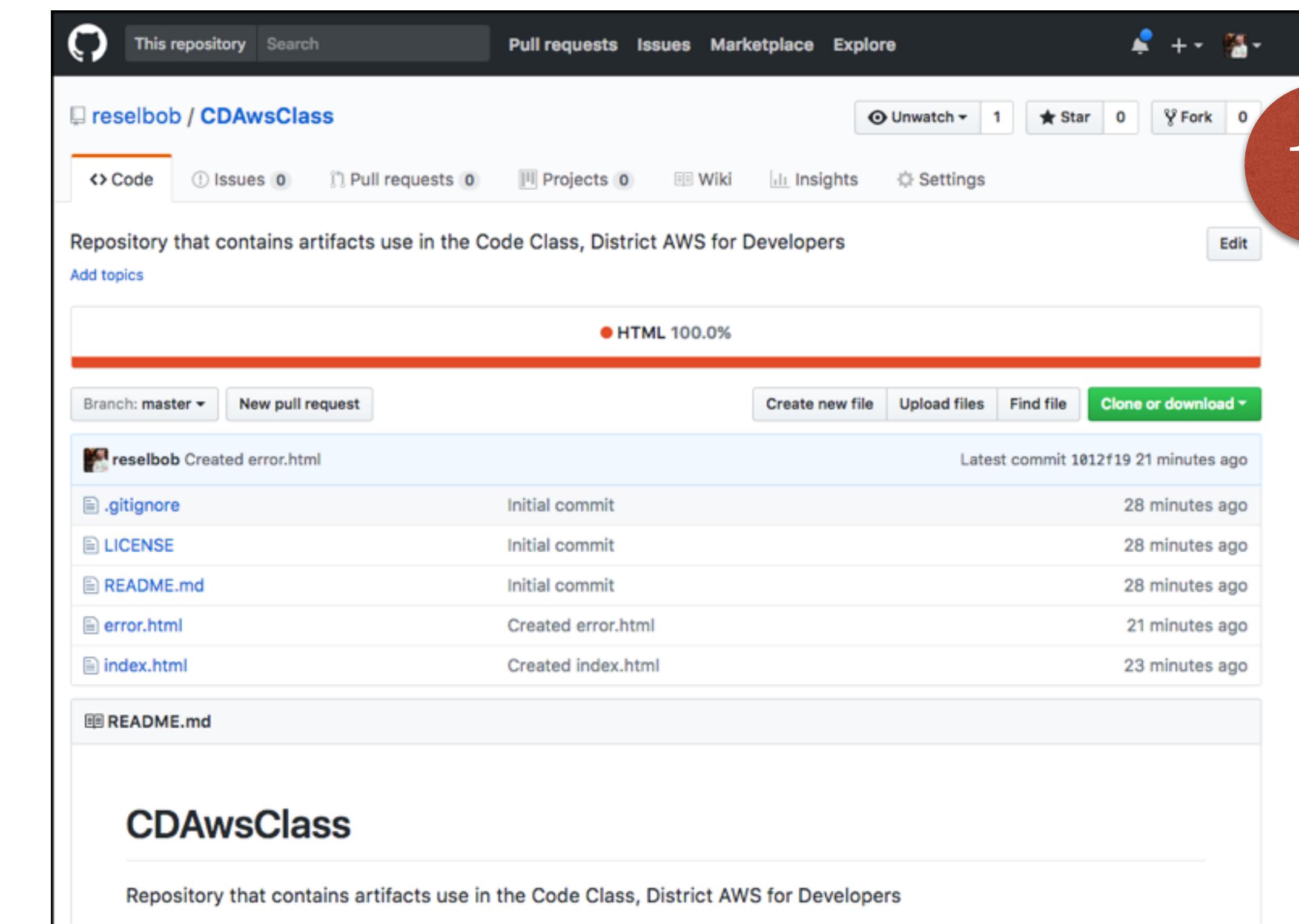
# Create the S3 Bucket



The screenshot shows the AWS S3 console for a bucket named "aws-class-bucket". The "Overview" tab is selected. There are four main sections: "Versioning", "Server access logging", "Static website hosting", and "Object-level logging". The "Bucket hosting" option under "Static website hosting" is highlighted with a red box. The "Versioning" and "Server access logging" sections are also visible.

15

<https://github.com/reselbob/CDAwsClass>



The screenshot shows a GitHub repository page for "reselbob / CDAwsClass". The repository description is "Repository that contains artifacts use in the Code Class, District AWS for Developers". The commit history shows the following entries:

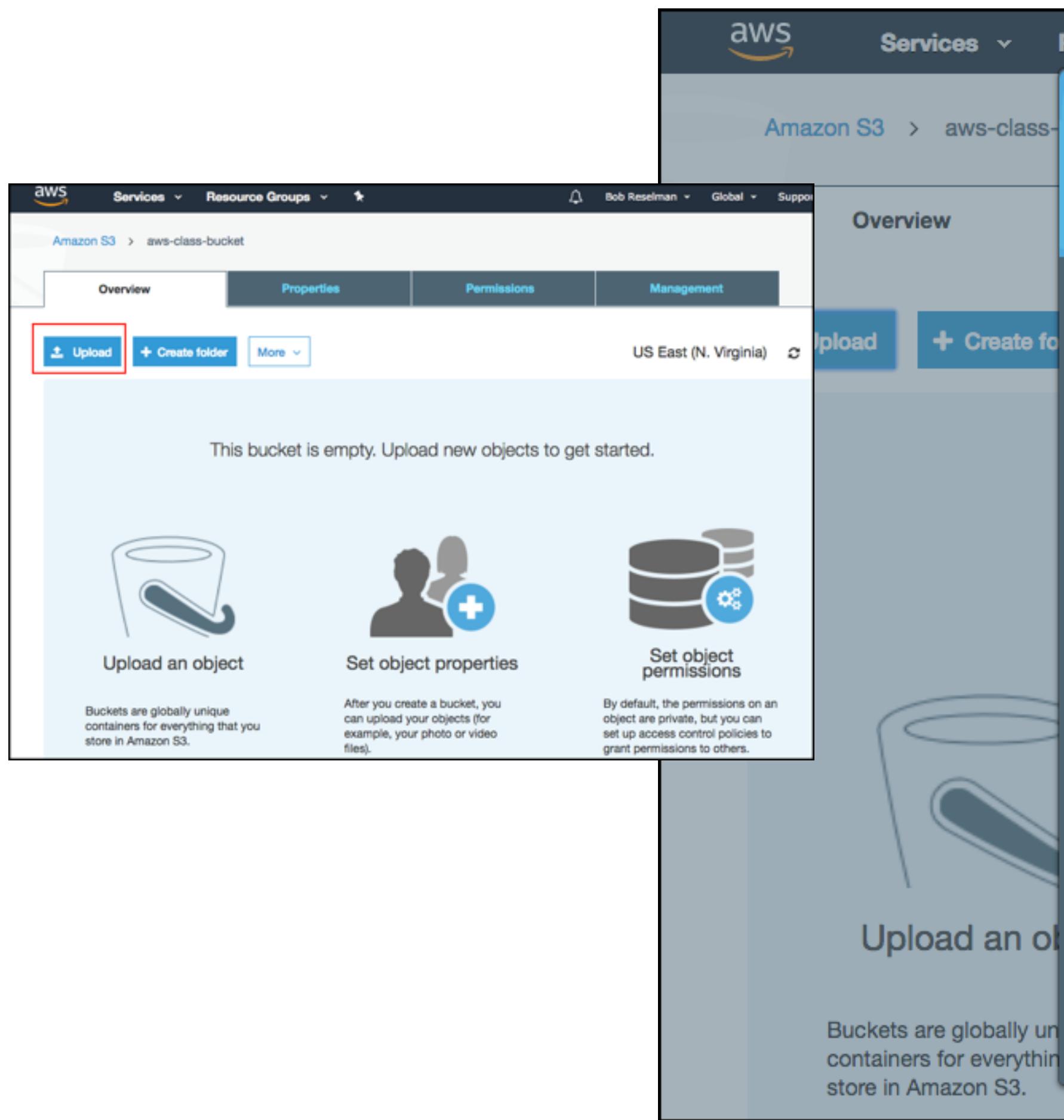
File	Commit Type	Time Ago
.gitignore	Initial commit	28 minutes ago
LICENSE	Initial commit	28 minutes ago
README.md	Initial commit	28 minutes ago
error.html	Created error.html	21 minutes ago
index.html	Created index.html	23 minutes ago

A red circle with the number 16 is overlaid on the top right corner of the GitHub screenshot.

16

# Create the S3 Bucket

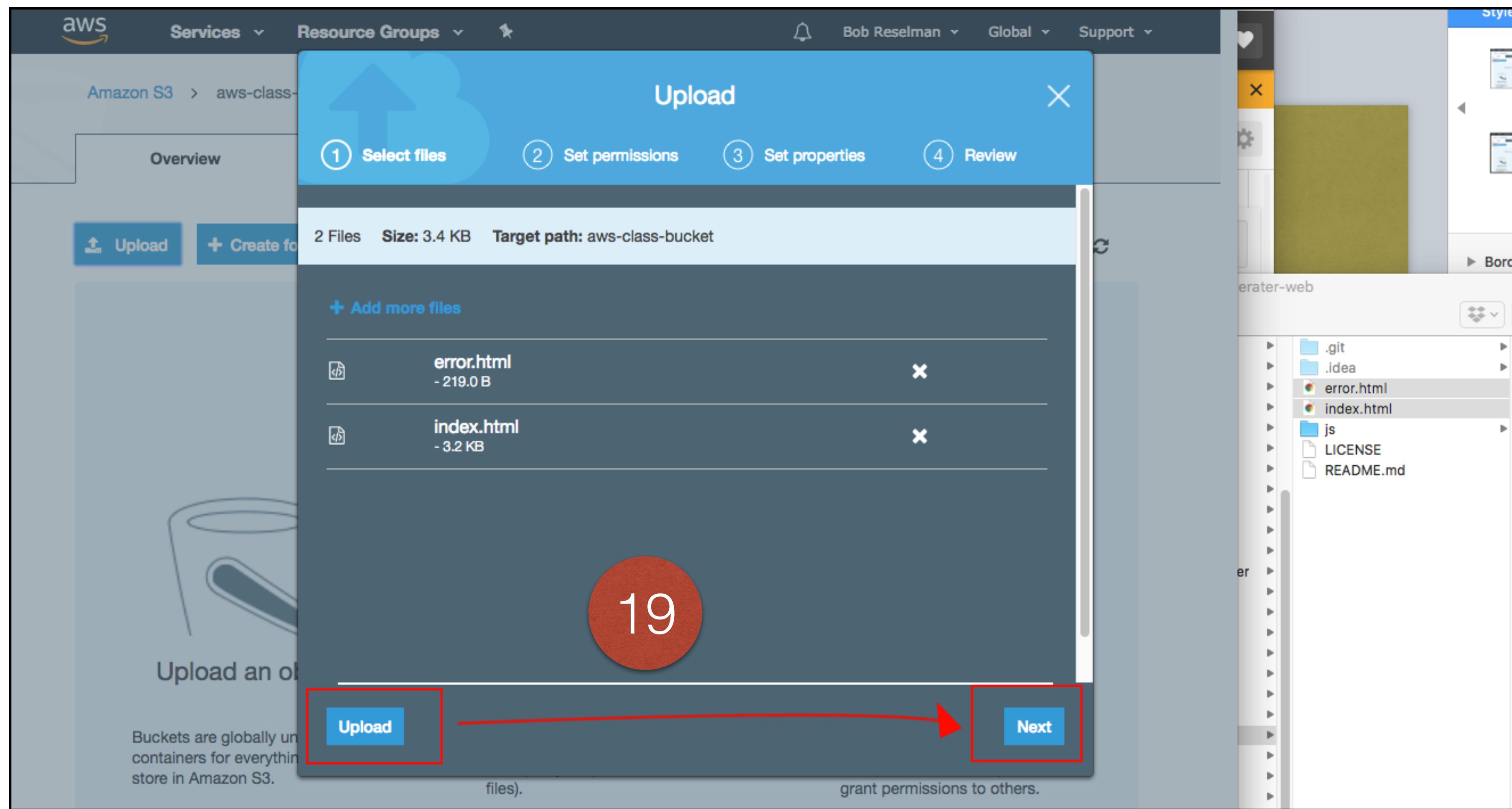
17



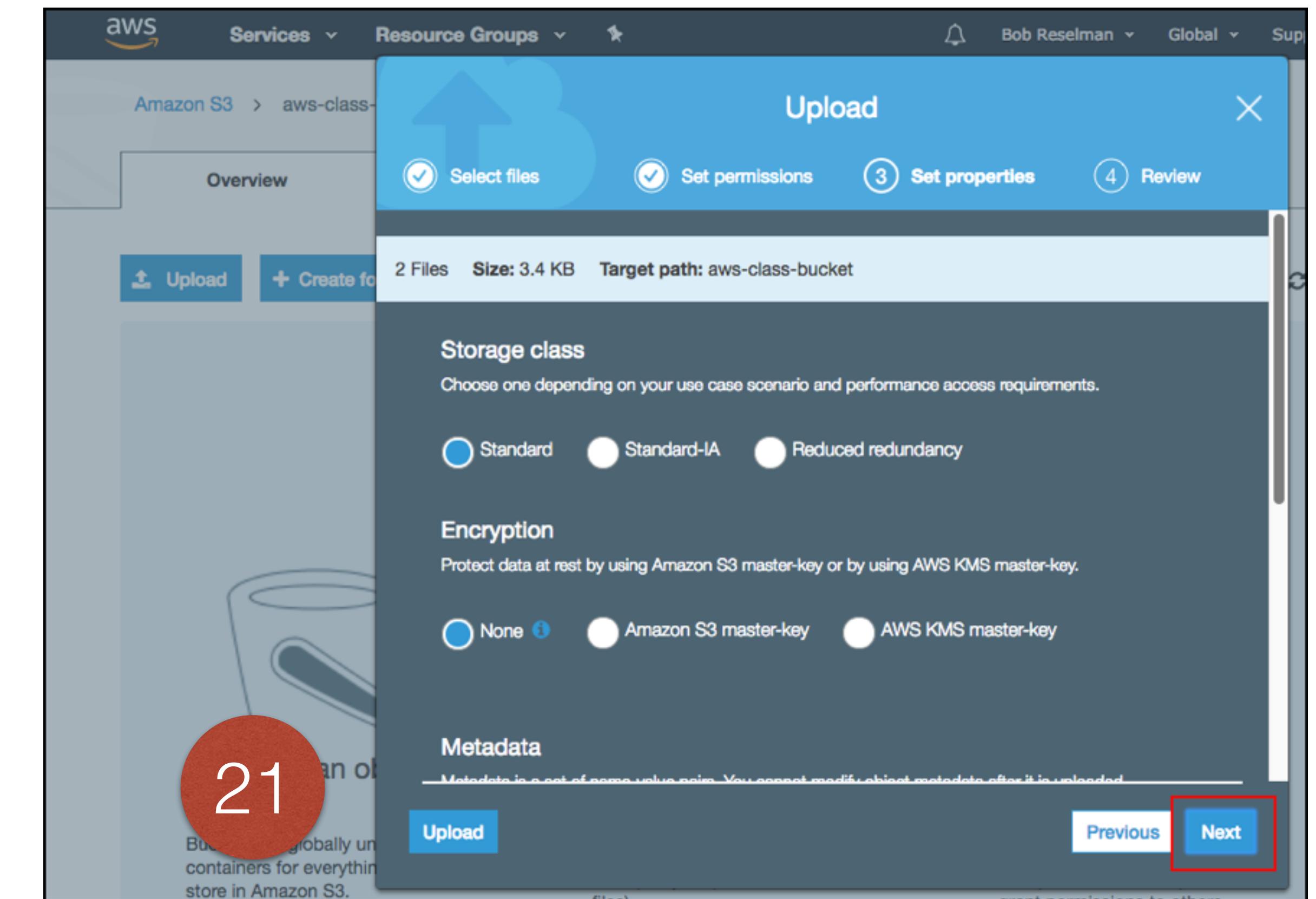
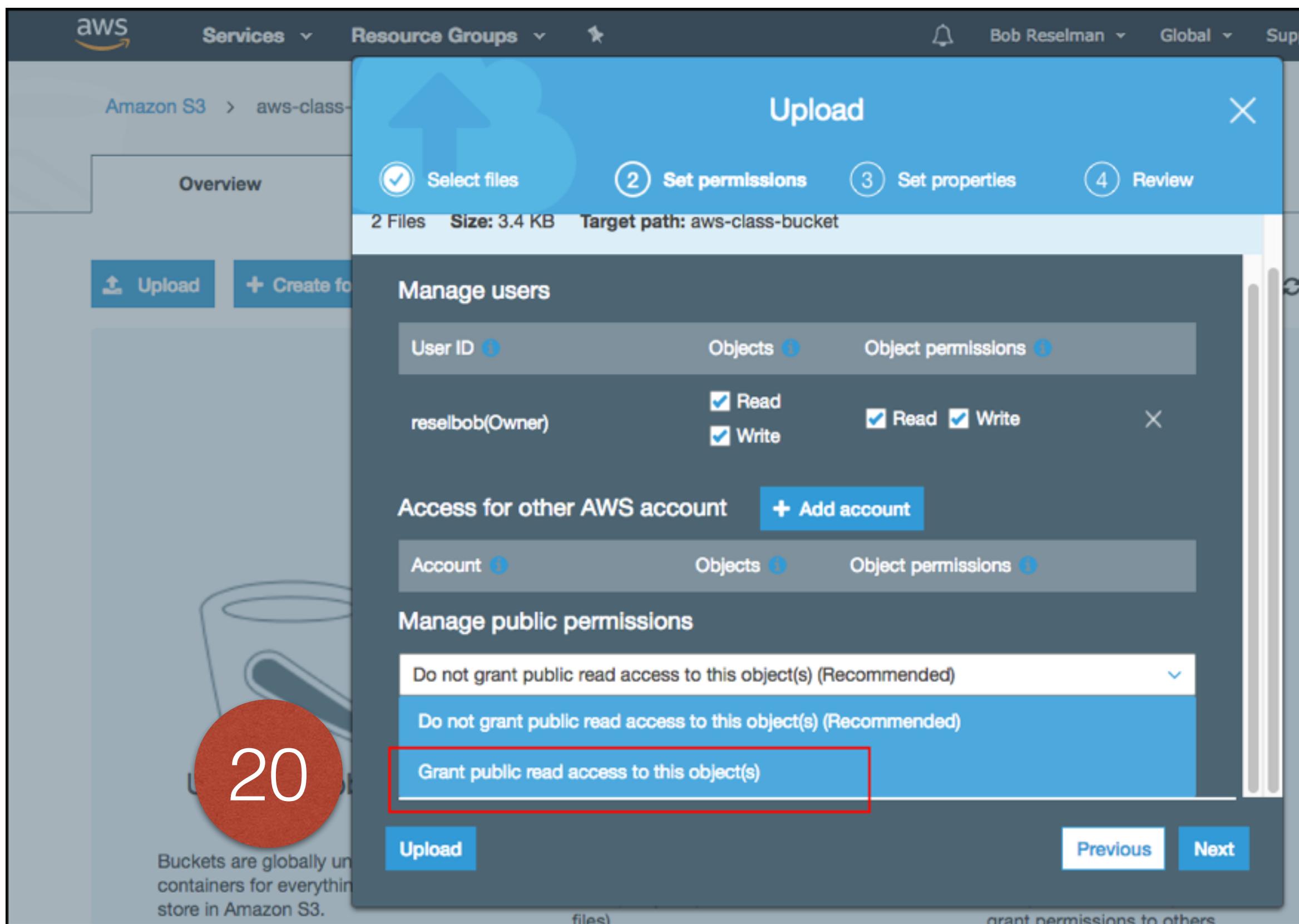
This screenshot shows the 'Upload' wizard in the AWS S3 console, currently at step 1: Select files. It displays a large blue area for dragging and dropping files, with the text 'Drag and drop here OR' and a 'Add files' button below it. To the right, a sidebar shows a file tree with files like '.git', '.idea', 'error.html', 'index.html', 'js', 'LICENSE', and 'README.md'. A red box highlights the file selection area, and a red arrow points from the 'error.html' file in the sidebar to this area.

18

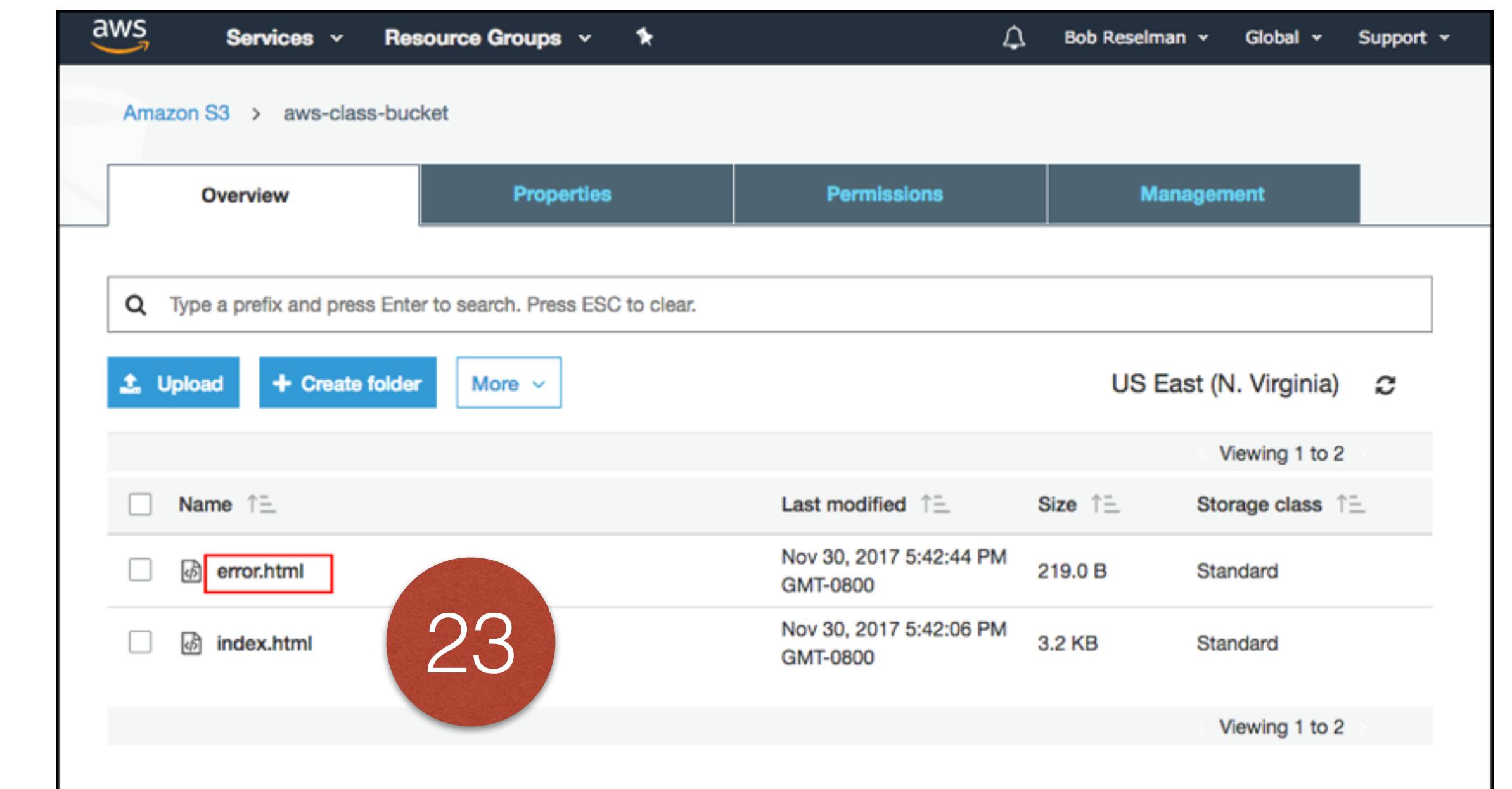
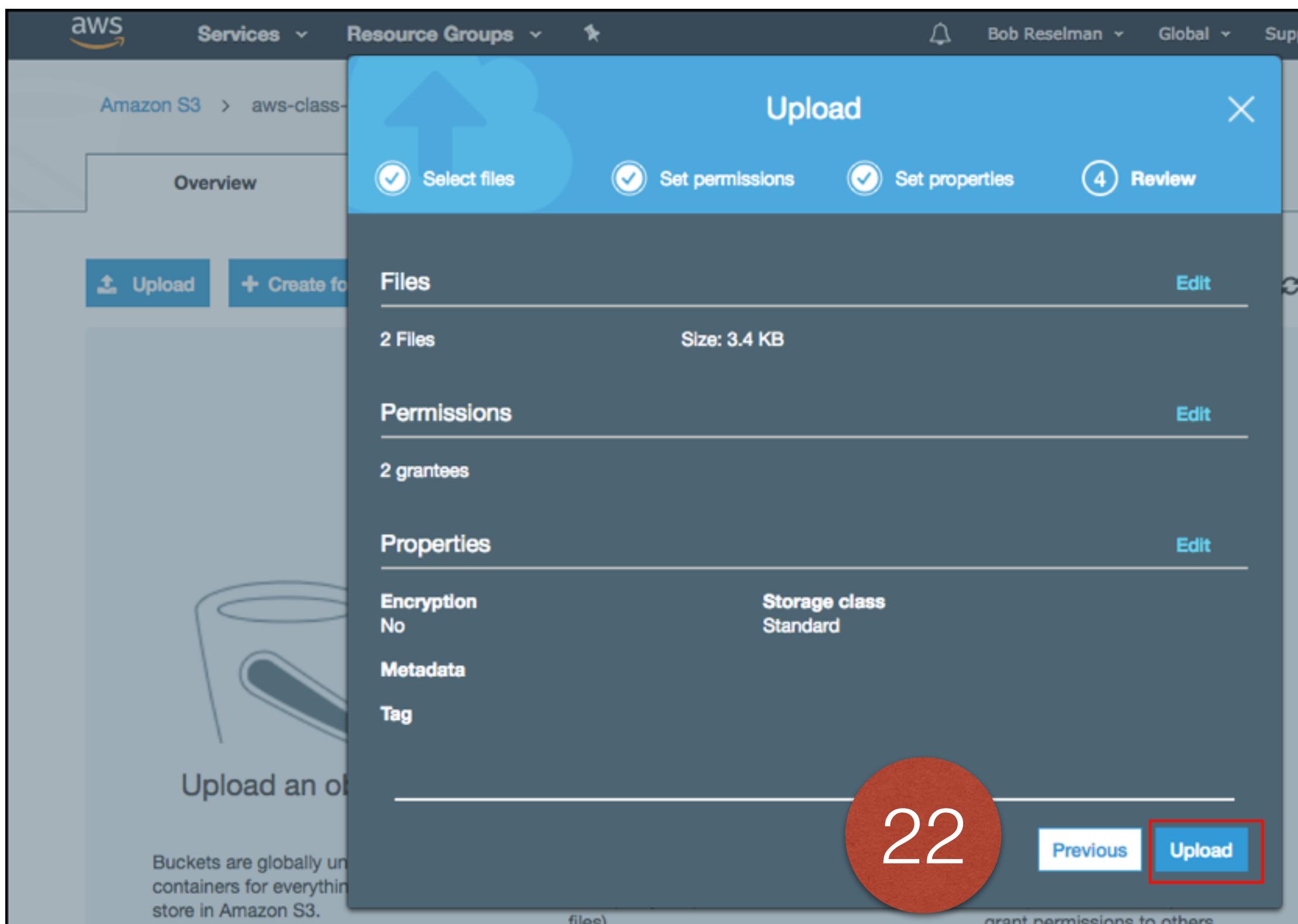
# Create the S3 Bucket



# Create the S3 Bucket



# Create the S3 Bucket



# Create the S3 Bucket

<https://s3.amazonaws.com/aws-class-bucket/index.html>

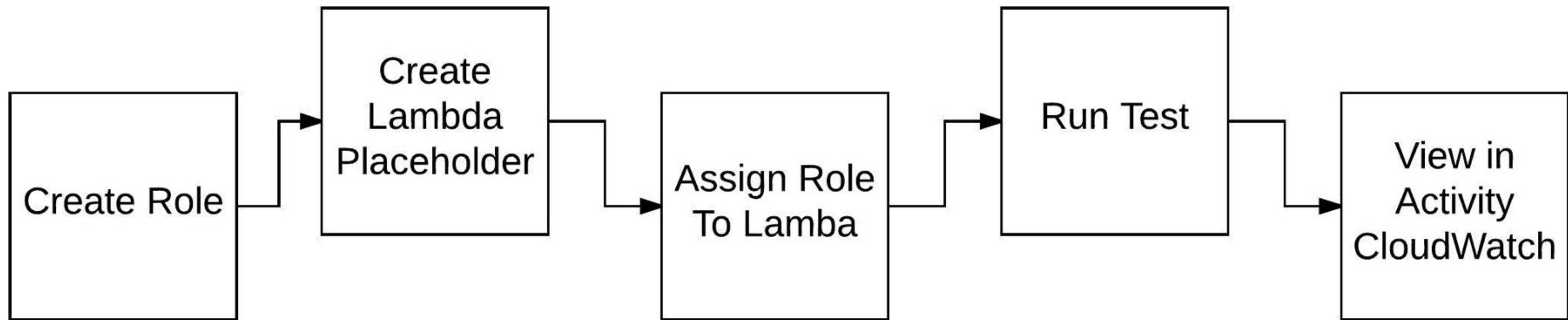
The screenshot shows the Amazon S3 console with the path `Amazon S3 > aws-class-bucket`. A file named `index.html` is selected, indicated by a blue border. The `Properties` tab is active. Below the tabs are buttons for `Open`, `Download`, `Download as`, `Make public`, and `Copy path`. The file details section includes fields for `Owner` (reselbob), `Last modified` (Nov 30, 2017 5:42:06 PM GMT-0800), `Etag` (ddfe1f47bcb21842886121443ddeaac), `Storage class` (Standard), `Server side encryption` (None), and `Size` (3226). At the bottom, a red box highlights the `Link` button, which has the URL `https://s3.amazonaws.com/aws-class-bucket/index.html`.

24

The screenshot shows a web form titled **Code District Aws Class Movie Rater**. It contains four input fields: `Movie` (a dropdown menu labeled "Select a Movie"), `Rating` (a dropdown menu labeled "Select a rating"), `Rater Email` (an input field), and a `Submit` button.

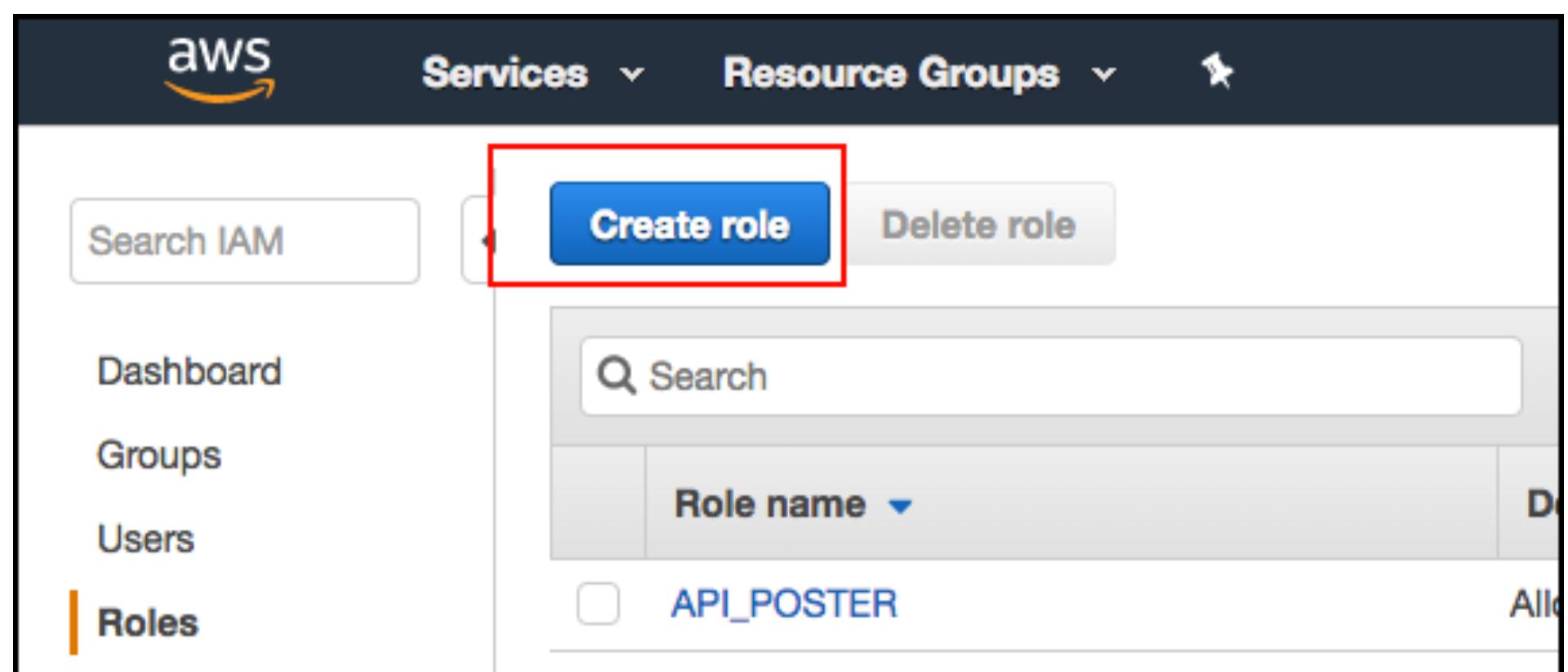
25

# Session 2

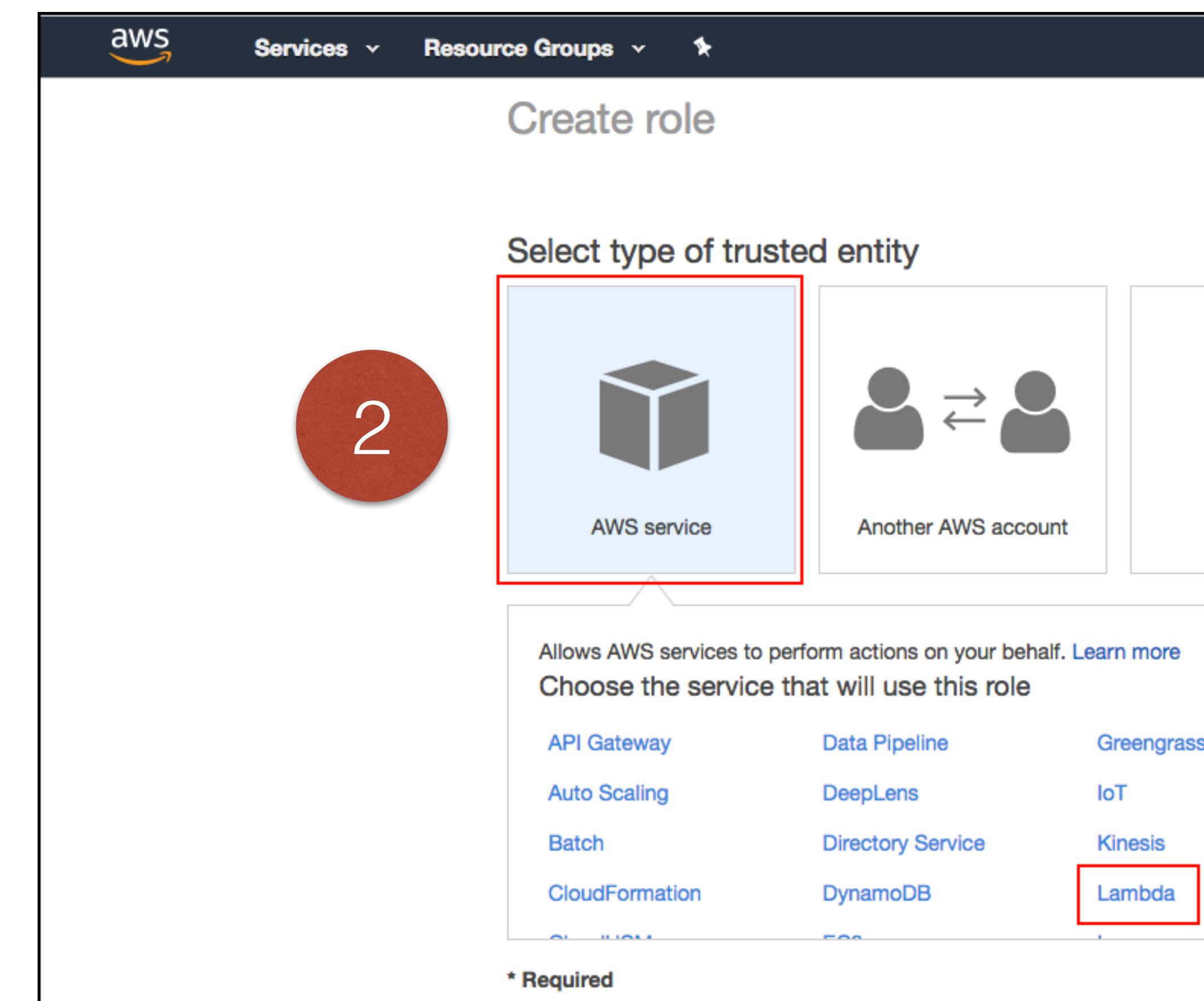


# Creating the Resource Role

1



2



# Creating the Resource Role

aws Services Resource Groups ★ Bob Reselman Global Support

Create role

3

1 Trust      2 Permissions      3 Review

Attach permissions policies

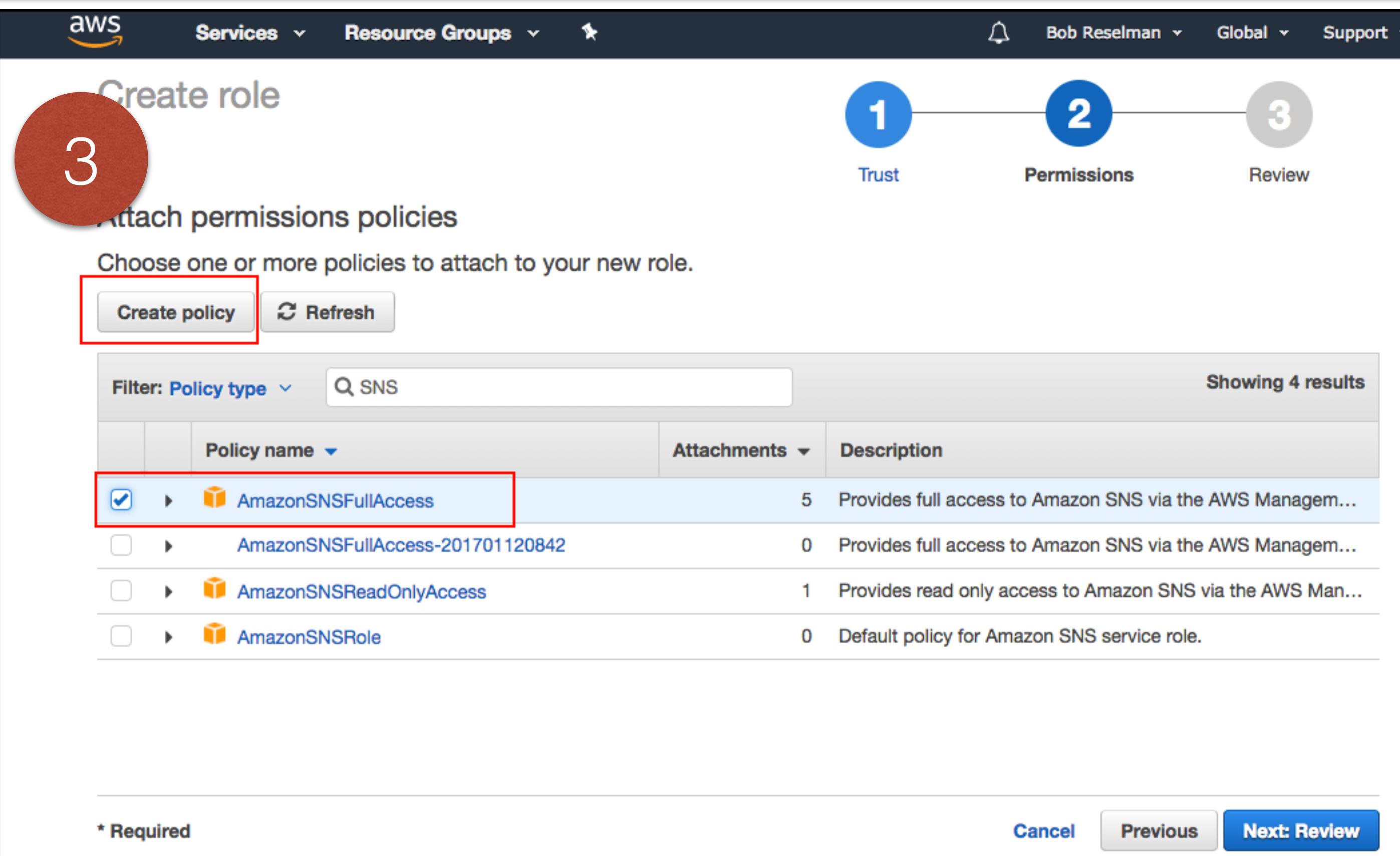
Choose one or more policies to attach to your new role.

Filter: Policy type ▾  Showing 4 results

	Policy name ▾	Attachments ▾	Description
<input checked="" type="checkbox"/>	▶ AmazonSNSFullAccess	5	Provides full access to Amazon SNS via the AWS Managem...
<input type="checkbox"/>	▶ AmazonSNSFullAccess-201701120842	0	Provides full access to Amazon SNS via the AWS Managem...
<input type="checkbox"/>	▶ AmazonSNSReadOnlyAccess	1	Provides read only access to Amazon SNS via the AWS Man...
<input type="checkbox"/>	▶ AmazonSNSRole	0	Default policy for Amazon SNS service role.

\* Required

Cancel Previous Next: Review

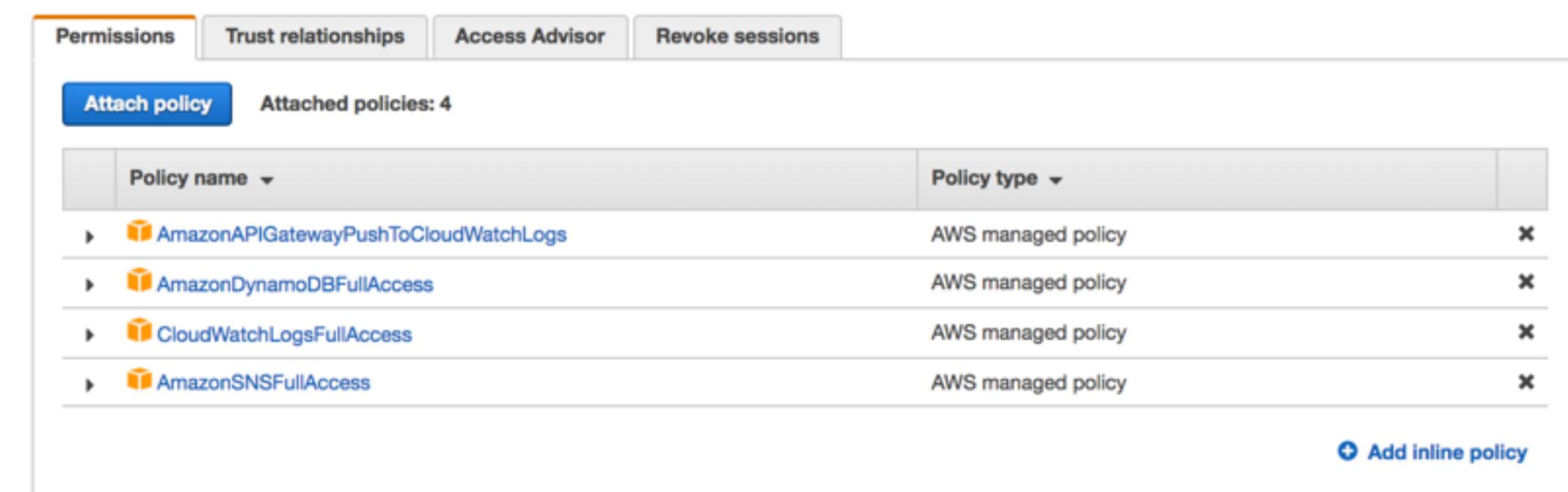


Permissions Trust relationships Access Advisor Revoke sessions

Attach policy Attached policies: 4

Policy name ▾	Policy type ▾
▶ AmazonAPIGatewayPushToCloudWatchLogs	AWS managed policy
▶ AmazonDynamoDBFullAccess	AWS managed policy
▶ CloudWatchLogsFullAccess	AWS managed policy
▶ AmazonSNSFullAccess	AWS managed policy

+ Add inline policy



# Creating the Resource Role

5

reselbob / CD AwsClass

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Repository that contains artifacts use in the Code Class, District AWS for Developers

Topics

4 commits 3 branches 0 releases 1 contributor MIT

Branch: Session-2 New pull request

This branch is 1 commit ahead of master.

Pull request Compare

Latest commit fd0c070 an hour ago

File	Commit Message	Time
.gitignore	Initial commit	22 hours ago
AwsClass.role.json	Created AwsClass.role.json	an hour ago
LICENSE	Initial commit	22 hours ago
README.md	Initial commit	22 hours ago
error.html	Created error.html	22 hours ago
index.html	Created index.html	22 hours ago

6

reselbob / CD AwsClass

Code Issues 0 Pull requests 0 Projects 0

Branch: Session-2 CDAwsClass / AwsClass.role.json

reselbob Created AwsClass.role.json

1 contributor

15 lines (14 sloc) | 308 Bytes

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Effect": "Allow",  
6       "Action": [  
7         "logs>CreateLogGroup",  
8         "logs>CreateLogStream",  
9         "logs:PutLogEvents"  
10      ],  
11      "Resource": "arn:aws:logs:*****"  
12    }  
13  ]  
14 }
```

# Creating the Resource Role

7

**Review policy**

Before you create this policy, provide the required information and review this policy.

Name\*  Maximum 64 characters. Use alphanumeric and '+,-,@,\_' characters.

Description

Maximum 1000 characters. Use alphanumeric and '+,-,@,\_' characters.

**Summary**

This policy defines some actions, resources, or conditions that do not provide permissions. To grant access, policies must have an action that has an applicable resource or condition. For details, choose Show remaining. [Learn more](#)

Service	Access level	Resource	Request condition
Allow (1 of 121 services) <a href="#">Show remaining 120</a>	Limited: Write	arn:aws:logs:/*:/*	None
CloudWatch Logs	Limited: Write	arn:aws:logs:/*:/*	None

Cancel Previous **Create policy**

8

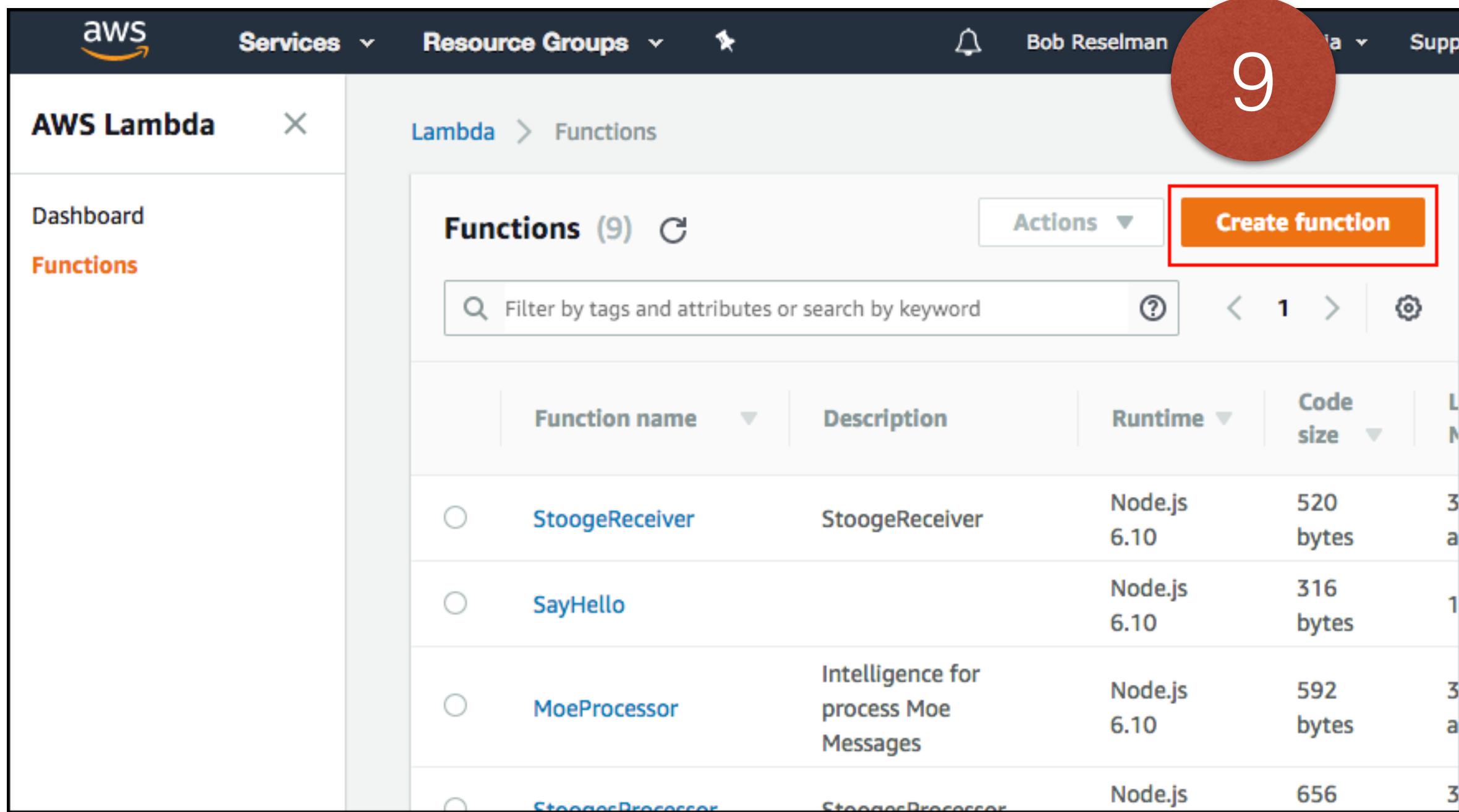
**Create role**

Search IAM

Dashboard Groups Users Roles Policies Identity providers Account settings Credential report Encryption keys

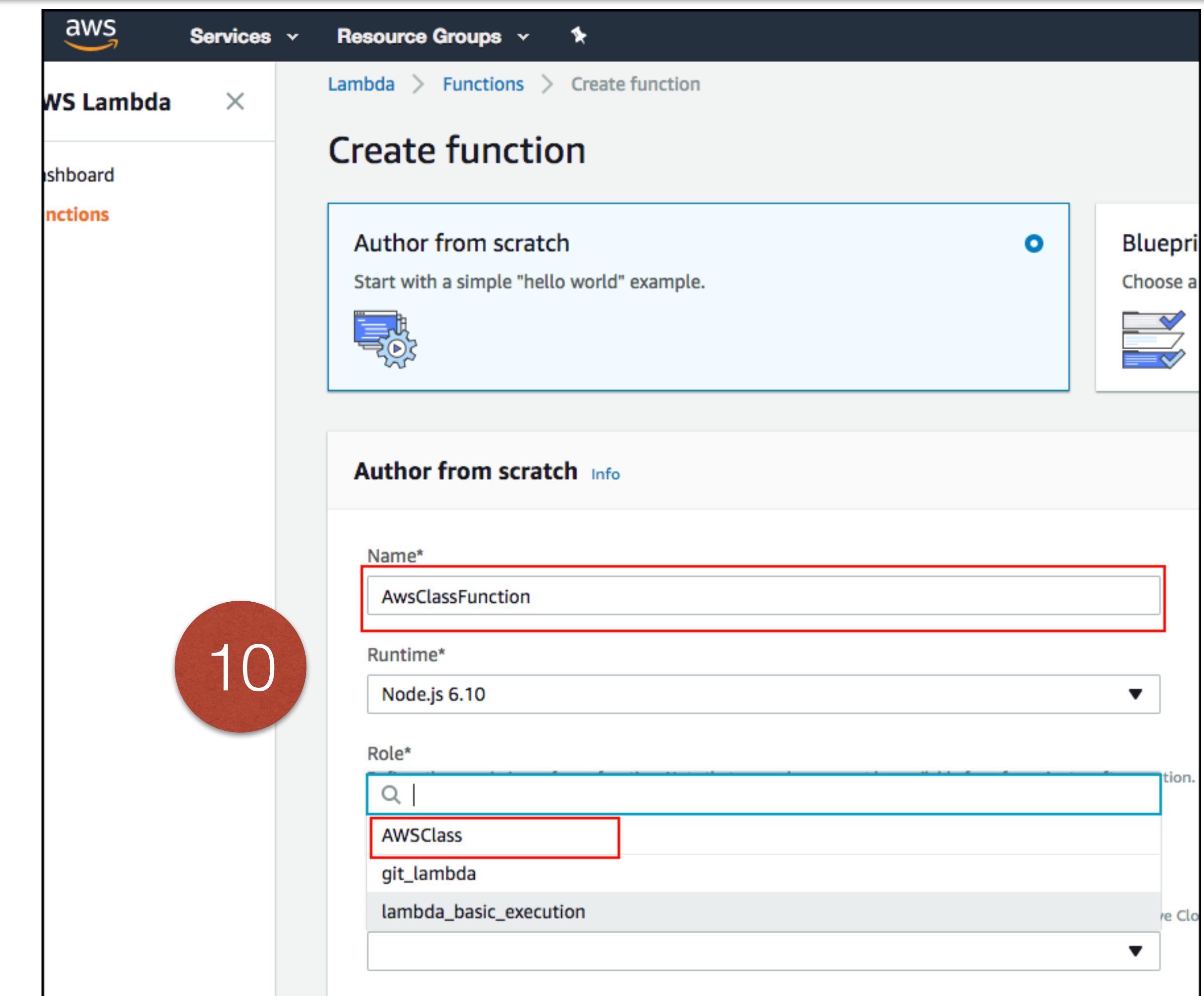
Role name	Description	Trusted entities
API_POSTER	Allows API Gateway to call AWS resources on yo...	AWS service: apigateway
aws-opsworks-ec2-role		AWS service: ec2
aws-opsworks-service-...		AWS service: opsworks
<b>AWSClass</b>		AWS service: lambda
CloudTrail_CloudWatch...		AWS service: cloudtrail
git_lambda	Allows git to fiddle with lambda	AWS service: lambda
lambda_basic_execution		AWS service: lambda
SNSFailureFeedback		AWS service: sns
SNSSuccessFeedback		AWS service: sns

# Creating Lambda Function



The screenshot shows the AWS Lambda service dashboard under the 'Functions' tab. There are 9 functions listed:

Function name	Description	Runtime	Code size
StoogeReceiver	StoogeReceiver	Node.js 6.10	520 bytes
SayHello		Node.js 6.10	316 bytes
MoeProcessor	Intelligence for process Moe Messages	Node.js 6.10	592 bytes
StoogesProcessor	StoogesProcessor	Node.js	656



The screenshot shows the 'Create function' wizard. The first step, 'Author from scratch', is selected. The form fields are as follows:

- Name\*:
- Runtime\*:
- Role\*:

# Creating Lambda Function

Author from scratch Info

Name\*  
AwsClassFunction

Runtime\*  
Node.js 6.10

Role\*  
Defines the permissions of your function. Note that new roles may not be available for a few minutes after creation. [Learn more](#) about Lambda execution roles.  
Choose an existing role

Existing role\*  
You may use an existing role with this function. Note that the role must be assumable by Lambda and must have Cloudwatch Logs permissions.  
AWSClass

Cancel **Create function**

11

Qualifiers ▾ Actions ▾ Select a test event.. ▾ Test Save

Runtime Handler Info  
Node.js 6.10 index.handler

Goto Tools Window

index.js

```
1 exports.handler = (event, context, callback) => {
2     // TODO implement
3     const dt = new Date();
4     let msg = `Hello from AwsClassFunction() at ${dt.toString()}`;
5     msg += ` From ${event.firstName} ${event.lastName} ... ${event.status}`;
6     console.log(msg);
7     callback(null, 'Hello from Lambda');
8 };
```

8:3 JavaScript Spaces: 4

12

# Creating Lambda Function

A function can have up to 10 test events. The events are persisted so you can switch to another computer or web browser and test your function with the same events.

- Create new test event
- Edit saved test events

Event template

Hello World

Event name

MyEvent

```
1: {  
2:   "firstName": "YOUR_FIRST_NAME",  
3:   "lastName": "YOUR_LAST_NAME",  
4:   "status": "Rocks!"  
5: }
```

13

Cancel

Create

14

Qualifiers ▾ Actions ▾ MyEvent ▾ Test Save

Runtime Node.js 6.10 Handler Info index.handler

```
index.js  x +  
1 exports.handler = (event, context, callback) => {  
2   // TODO implement  
3   const dt = new Date();  
4   let msg = `Hello from AwsClassFunction() at ${dt.toString()}`;  
5   msg += ` From ${event.firstName} ${event.lastName} ... ${event.status}`;  
6   console.log(msg);  
7   callback(null, 'Hello from Lambda');
```

# Creating a Lambda Function

A screenshot of the AWS Lambda function editor and execution results interface. The top half shows the code editor with an index.js file containing a handler function. A red arrow points from the highlighted 'console.log' line in the code to the corresponding log entry in the execution results. The bottom half shows the execution results panel with a success status, memory usage, and execution time. A red arrow points from the highlighted 'Response' value in the results to the same value in the logs.

index.js

```
1 exports.handler = (event, context, callback) => {
2     // TODO implement
3     const dt = new Date();
4     let msg = `Hello from AwsClassFunction() at ${dt.toString()}`;
5     msg += ` From ${event.firstName} ${event.lastName} ... ${event.status}`;
6     console.log(msg);
7     callback(null, 'Hello from Lambda');
8 };
```

Execution Result

Status: Succeeded Max Memory Used: 20 MB Time: 18.09 ms

Response:  
"Hello from Lambda"

Request ID:  
"6e020981-d6f8-11e7-aaab-617d2f8c30dc"

Function Logs:

```
START RequestId: 6e020981-d6f8-11e7-aaab-617d2f8c30dc Version: $LATEST
2017-12-02T00:33:34.915Z 6e020981-d6f8-11e7-aaab-617d2f8c30dc Hello from AwsClassFunction() at Sat Dec 02 2017 00:33:34 GMT+0000 (UTC)
END RequestId: 6e020981-d6f8-11e7-aaab-617d2f8c30dc
REPORT RequestId: 6e020981-d6f8-11e7-aaab-617d2f8c30dc Duration: 18.09 ms Billed Duration: 100 ms Memory Size: 128 MB Max Memory Used: 20 MB
```

15

# View Activity in CloudWatch

The screenshot shows the left sidebar of the AWS Management Console. It includes sections for Compute (EC2, Lightsail, Elastic Container Service, Lambda, Batch, Elastic Beanstalk), Storage (S3, EFS, Glacier, Storage Gateway), and Database (RDS, DynamoDB, ElastiCache, Amazon Redshift). On the right, there are three main categories: Developer Tools (CodeStar, CodeCommit, CodeBuild, CodeDeploy, CodePipeline, Cloud9, X-Ray), Management Tools (CloudWatch, CloudFormation, CloudTrail, Config, OpsWorks, Service Catalog, Systems Manager, Trusted Advisor, Managed Services), and a red-highlighted item, CloudWatch.

16

The screenshot shows two pages from the CloudWatch Metrics service. The top page is the Metric Summary, which displays operational and performance metrics for the US East (N. Virginia) region. It includes sections for Dashboards, Alarms (with 0 ALARM, 0 INSUFFICIENT, and 0 OK metrics), Billing, Events, Rules, Event Buses, Logs (highlighted with a red box), Metrics, Favorites, and a button to Add a dashboard. The bottom page is the Alarm Summary, which states that no alarms have been created in the region and provides instructions for setting up alarms based on CloudWatch metrics. It also includes a section for setting up billing alarms.

17

# View Activity in CloudWatch

The screenshot shows the AWS CloudWatch service dashboard. The left sidebar has links for CloudWatch, Dashboards, Alarms, ALARM (0), INSUFFICIENT (0), OK (0), Billing, Events, Rules, Event Buses, and Logs. The Logs link is highlighted with an orange bar. The main area shows the CloudWatch > Log Groups page. It includes a 'Create Metric Filter' button, an 'Actions' dropdown, and a 'Filter: Log Group Name Prefix' input field. A red box highlights the first log group listed: '/aws/lambda/AwsClassFunction'. Below it are other log groups: '/aws/lambda/CurlyProcessor', '/aws/lambda/LarryProcessor', '/aws/lambda/MoeProcessor', and '/aws/lambda/SayHello'. A red circle with the number 18 is in the bottom-left corner.

The screenshot shows the CloudWatch > Log Groups > Streams for /aws/lambda/AwsClassFunction page. It includes buttons for 'Search Log Group', 'Create Log Stream', and 'Delete Log Stream'. A 'Filter: Log Stream Name Prefix' input field is present. A red box highlights the first log stream listed: '2017/12/02/[\$LATEST]2e3d231bbd314fffa055033eec463960'. Below it are other log streams: '2017/12/02/[\$LATEST]263a57b2eef84b1d981cd44e21b99e61', '2017/12/02/[\$LATEST]677fe246cbf54279993237e4e8d36f79', '2017/12/02/[\$LATEST]0bb8d76fb2346f9b7351b559002cdf1', and '2017/12/02/[\$LATEST]9fe7a586369f43c4ac1273e85fd2b53a'. A red circle with the number 19 is in the top-right corner.

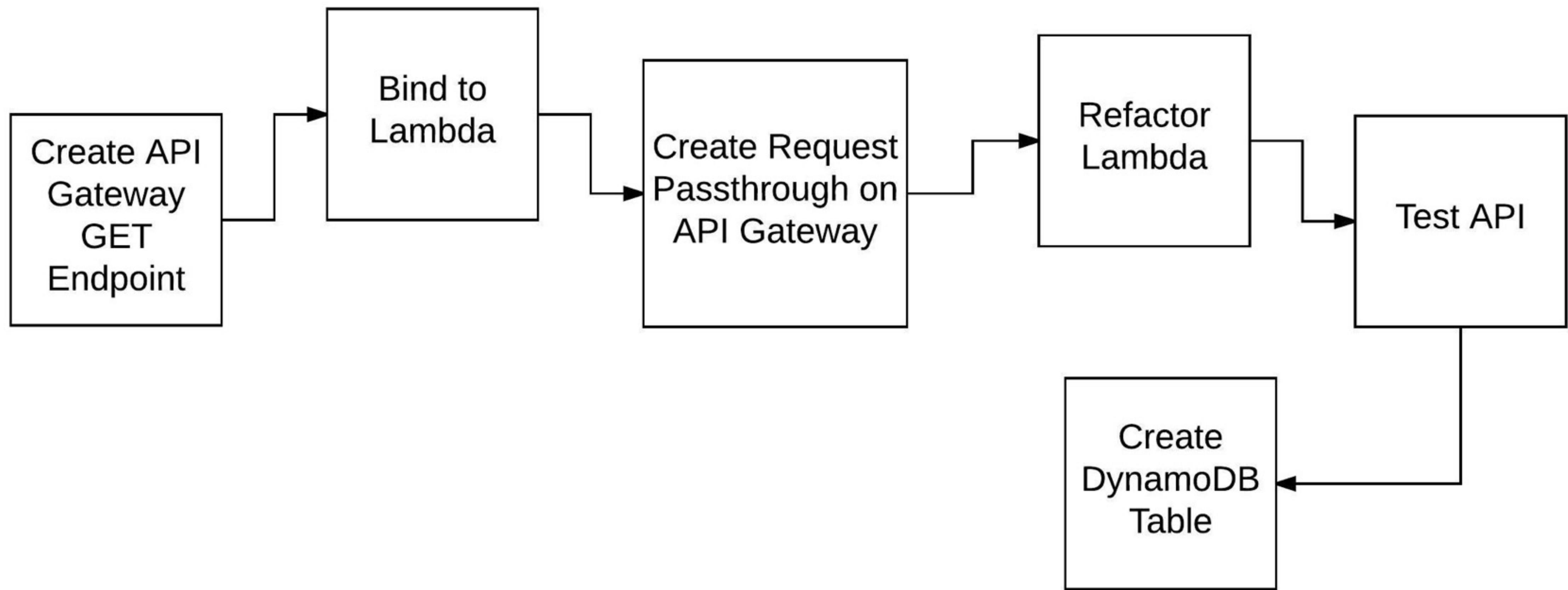
Log Streams	Last Event Time
2017/12/02/[\$LATEST]2e3d231bbd314fffa055033eec463960	2017-12-01 16:33 UTC-8
2017/12/02/[\$LATEST]263a57b2eef84b1d981cd44e21b99e61	2017-12-01 16:03 UTC-8
2017/12/02/[\$LATEST]677fe246cbf54279993237e4e8d36f79	2017-12-01 16:02 UTC-8
2017/12/02/[\$LATEST]0bb8d76fb2346f9b7351b559002cdf1	2017-12-01 16:01 UTC-8
2017/12/02/[\$LATEST]9fe7a586369f43c4ac1273e85fd2b53a	2017-12-01 16:00 UTC-8

# View Activity in CloudWatch

The screenshot shows the AWS CloudWatch Logs interface. The left sidebar is collapsed, showing navigation links: CloudWatch, Dashboards, Alarms, ALARM (0), INSUFFICIENT (0), OK (0), Billing, Events, Rules, Event Buses, Logs (selected), Metrics, and Favorites. The main area displays a breadcrumb trail: CloudWatch > Log Groups > /aws/lambda/AwsClassFunction > 2017/12/02/[LATEST]2e3d231bbd314fffa055033eec463960. Below the trail are buttons for 'Expand all' (radio button selected), 'Row' (radio button unselected), and 'Text'. There is also a filter bar with 'Filter events' and time range buttons for 'all' (selected) and 30s, 5m, 1h, 6h, 1d, 1w, custom. The log table has columns for 'Time (UTC +00:00)' and 'Message'. The table shows one log entry for December 2, 2017, at 00:33:34. The message content is: "No older events found at the moment. Retry." followed by "START RequestId: 6e020981-d6f8-11e7-aaab-617d2f8c30dc Version: \$LATEST". A second row for the same timestamp shows the full message: "2017-12-02T00:33:34.915Z 6e020981-d6f8-11e7-aaab-617d2f8c30dc Hello from AwsClassFunction() at Sat Dec 02 2017 00:33:34 GMT+0000 (UTC) From YOUR\_FIRST\_NAME YOUR\_LAST\_NAME ... Rocks!". This message is highlighted with a red border. A red circle with the number 20 is overlaid on the timeline above the second log entry. The bottom of the table shows two more entries: "END RequestId: 6e020981-d6f8-11e7-aaab-617d2f8c30dc" and "REPORT RequestId: 6e020981-d6f8-11e7-aaab-617d2f8c30dc Duration: 18.09 ms Billed Duration: 100 ms".

Time (UTC +00:00)	Message
2017-12-02	No older events found at the moment. Retry.
00:33:34	START RequestId: 6e020981-d6f8-11e7-aaab-617d2f8c30dc Version: \$LATEST
00:33:34	2017-12-02T00:33:34.915Z 6e020981-d6f8-11e7-aaab-617d2f8c30dc Hello from AwsClassFunction() at Sat Dec 02 2017 00:33:34 GMT+0000 (UTC) From YOUR_FIRST_NAME YOUR_LAST_NAME ... Rocks!
00:33:34	END RequestId: 6e020981-d6f8-11e7-aaab-617d2f8c30dc
00:33:34	REPORT RequestId: 6e020981-d6f8-11e7-aaab-617d2f8c30dc Duration: 18.09 ms Billed Duration: 100 ms

# Session 3



# Creating API GET Endpoint

Services ▾ Resource Groups ▾

Find a service by name or feature (for example, EC2, S3 or VM, storage).

Amazon Rekognition

Migration

AWS Migration Hub

Application Discovery Service

Database Migration Service

Server Migration Service

Snowball

Networking & Content Delivery

VPC

CloudFront

Route 53

**API Gateway**

Direct Connect

Media Services

Elastic Transcoder

MediaConvert

MediaLive

MediaPackage

MediaStore

MediaTailor

Machine Learning

Amazon SageMaker

Amazon Comprehend

AWS DeepLens

Amazon Lex

Machine Learning

Amazon Polly

Rekognition

Amazon Transcribe

CloudWatch Metrics

Directory Service

WAF & Shield

Artifact

Mobile Services

Mobile Hub

Pinpoint

AWS AppSync

Device Farm

Mobile Analytics

AR & VR

Amazon Sumerian

Application Integration

Step Functions

Amazon MQ

aws Services ▾ Resource Groups ▾

Amazon API Gateway APIs

**+ Create API**

Usage Plans

API Keys

Custom Domain Names

Client Certificates

VPC Links

# Creating API GET Endpoint

The screenshot shows the 'Create new API' page in the Amazon API Gateway console. On the left sidebar, under the 'APIs' section, there are links for Usage Plans, API Keys, Custom Domain Names, Client Certificates, VPC Links, and Settings. The main area is titled 'Create new API' and contains the following fields:

- API name\***: AwsClassApi (highlighted with a red box)
- Description**: API for the Aws Class Project (highlighted with a red box)
- Endpoint Type**: Edge optimized

At the bottom right of the form is a blue 'Create API' button. A large red circle with the number '3' is overlaid on the 'API name\*' field.

# Creating API GET Endpoint

The screenshot shows the AWS API Gateway interface. The top navigation bar includes the AWS logo, Services dropdown, Resource Groups dropdown, a bell icon, Bob Reselman, N. Virginia, and Support. The main navigation on the left lists APIs, AWSClassApi, Resources (which is selected), Stages, Authorizers, Gateway Responses, Models, Documentation, and Binary Support. The current view is under the 'Methods' tab for the root resource '/'. A red circle with the number '4' is overlaid on the left side of the screen.

No methods defined for the resource.

The screenshot shows the same AWS API Gateway interface, but the focus is on the 'Actions' dropdown menu. The menu includes RESOURCE ACTIONS: Create Method (highlighted with a red box), Create Resource (highlighted with a red box), Enable CORS, and Edit Resource Documentation. It also includes API ACTIONS: Deploy API, Import API, Edit API Documentation, and Delete API. A red circle with the number '5' is overlaid on the right side of the screen.

# Creating API GET Endpoint

New Child Resource

Use this page to create a new child resource for your resource.

Configure as proxy resource

Resource Name\*   

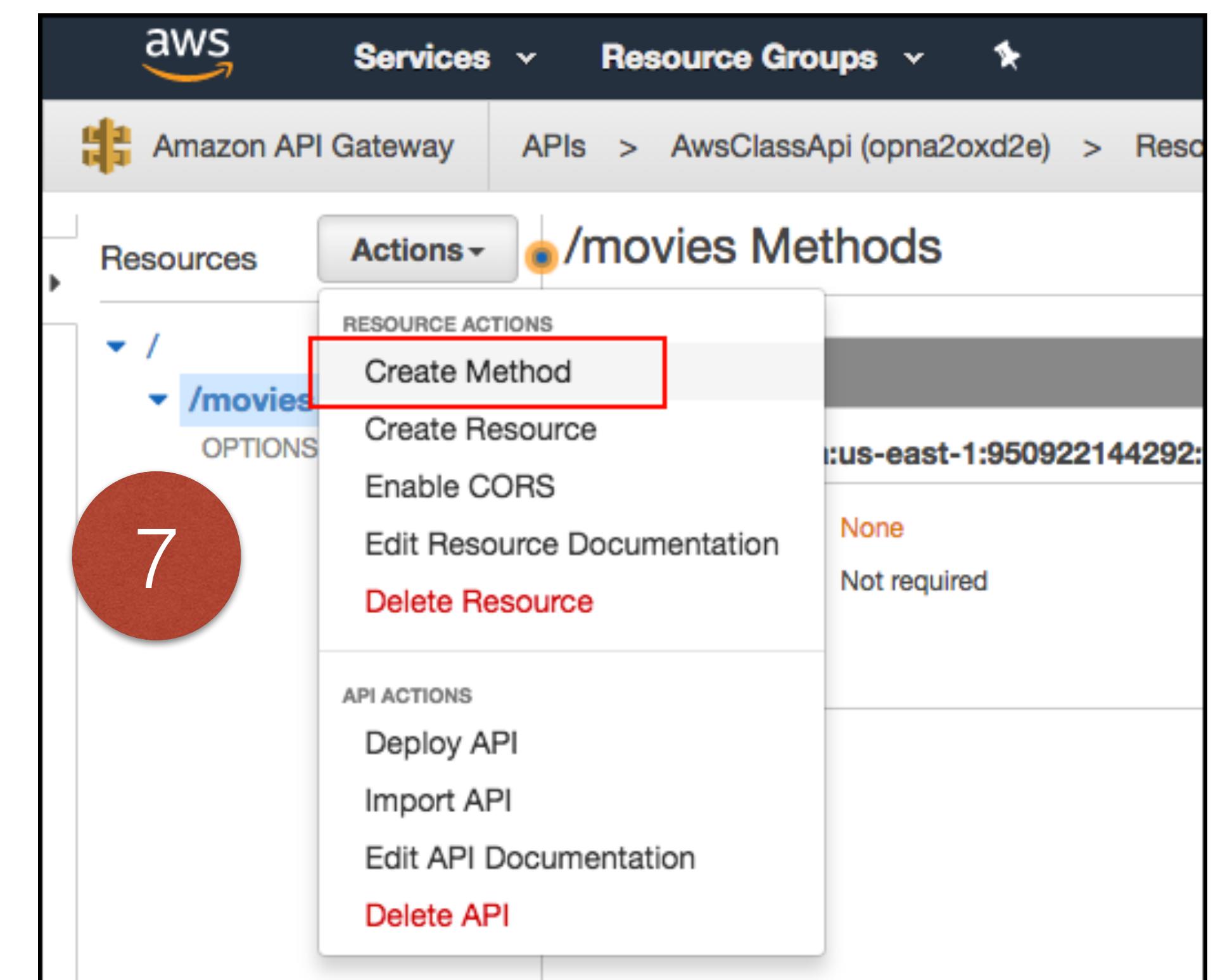
Resource Path\*   

You can add path parameters using brackets. For example, the resource path `{username}` represents a path parameter called 'username'. Configuring `/proxy+` as a proxy resource catches all requests to its sub-resources. For example, it works for a GET request to `/foo`. To handle requests to `/`, add a new ANY method on the `/` resource.

Enable API Gateway CORS   

\* Required

Cancel Create Resource



# Creating API GET Endpoint

The screenshot shows the AWS API Gateway console. The navigation bar at the top says "Services" and "Resource Groups". Below that, it shows "Amazon API Gateway" and "APIs > AwsClassApi (opna2ox)". On the left, there's a sidebar with "Resources" and a tree view showing a root node with a child node "/movies". In the main area, there's a "Actions" button and a title "(/movies Methods)". Below that, under the "OPTIONS" section, the Lambda ARN is listed as "arn:aws:lambda:us-east-1:950922144292:function:...". Under "Authorization", it says "None". Under "API Key", it says "Not required". To the right of the Lambda ARN, there are three icons: a checkmark, a question mark, and a close button. A red box highlights the "GET" icon in the "Actions" dropdown menu. A red circle with the number "8" is in the bottom right corner.

The screenshot shows the AWS API Gateway console, similar to the previous one but with a different URL: "APIs > AwsClassApi (opna2oxd2e) > Resources > /movies Methods". The main area shows the same "OPTIONS" configuration as step 8. However, the "Actions" dropdown menu now has "GET" selected, indicated by a red box around the "GET" icon. A red circle with the number "9" is in the bottom right corner.

# Creating API GET Endpoint

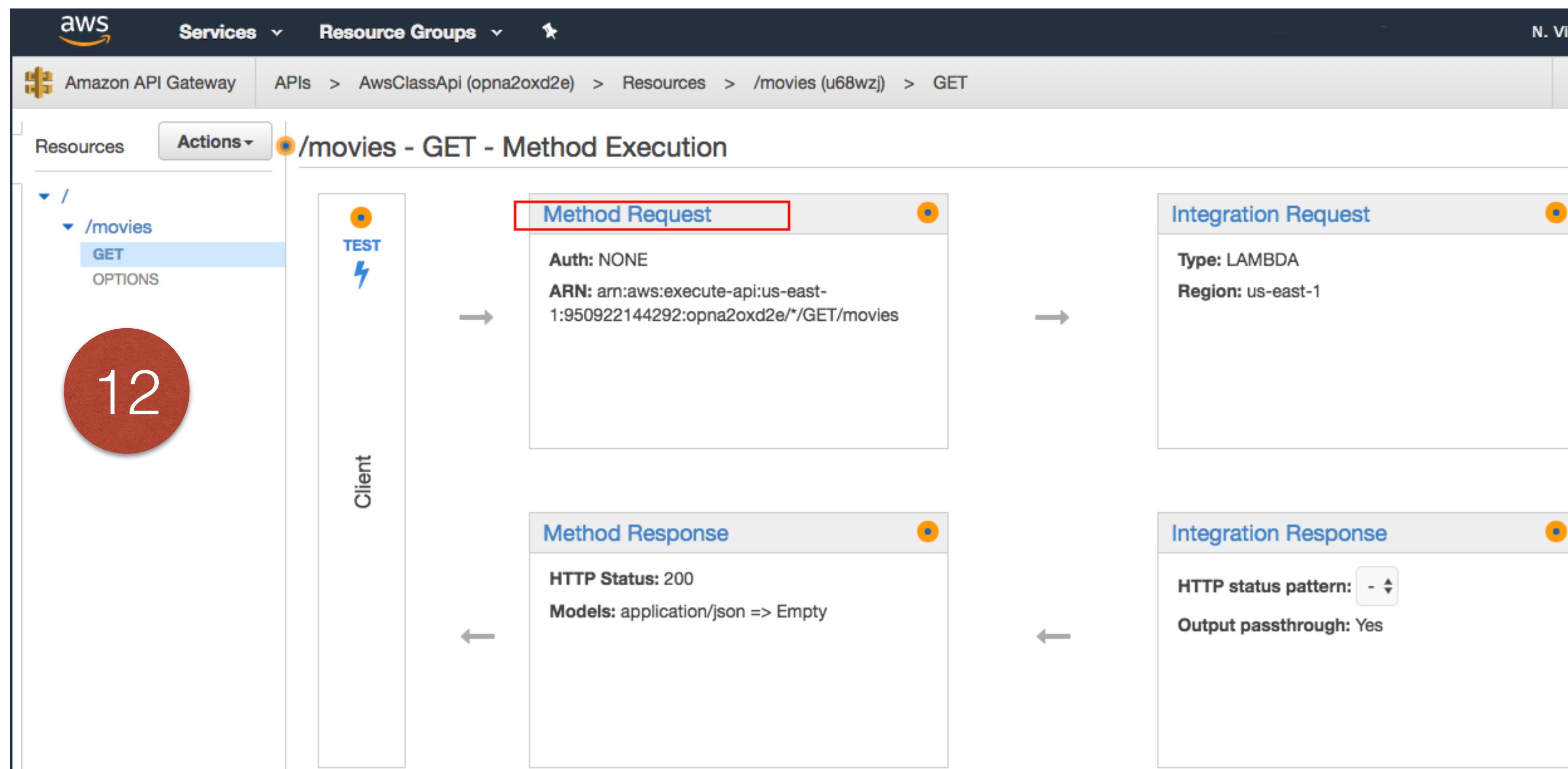
The screenshot shows the AWS API Gateway console. The URL in the address bar is `https://console.aws.amazon.com/apigateway/v2/resources/.../methods`. The left sidebar shows a tree structure with `/`, `/movies`, and `/movies/{id}`. The `/movies/{id}` node is selected. On the right, under the heading `/movies/{id} Methods`, there is a section titled `OPTIONS` with a sub-section `Mock Endpoint`. The `Authorization` field is set to `None` and the `API Key` field is set to `Not required`. A dropdown menu is open over the `OPTIONS` button, listing `ANY`, `DELETE`, `GET`, `HEAD`, `PATCH`, `POST`, and `PUT`. The `GET` option is highlighted with a blue background and a red box is drawn around it.

10

The screenshot shows the same AWS API Gateway interface as the previous one, but now the `GET` method is selected in the dropdown menu. The red box from the previous screenshot is now highlighting the checked checkbox next to the `GET` button in the dropdown menu.

11

# Creating API GET Endpoint



# Creating API GET Endpoint

The screenshot shows the AWS API Gateway console. In the top navigation bar, 'Services' is selected under 'Resource Groups'. The main area displays a tree structure: 'Amazon API Gateway' > 'APIs' > 'AwsClassApi (opna2oxd2e)' > 'Resources' > '/movies (u68wz...)'.

The left sidebar shows the 'Resources' list with '/movies' expanded, showing 'GET' and 'OPTIONS' methods. A red circle with the number '13' is overlaid on the left side of the screen.

The right panel shows the 'Method Execution /movies - GET - Method Request' configuration. It includes sections for 'Settings' (Authorization: NONE, Request Validator: NONE, API Key Required: false), 'URL Query String Parameters' (with a table showing a single entry for 'firstName'), and 'HTTP Request Headers' (with a table showing a single entry for 'Caching'). A red box highlights the 'Add query string' button at the bottom.

The screenshot shows the 'Method Execution /movies - GET - Method Request' configuration. A red circle with the number '14' is overlaid on the right side of the screen.

The 'URL Query String Parameters' section is expanded, showing a table with one row for 'firstName'. The 'Name' column contains 'firstName', the 'Required' column has a checked checkbox, and the 'Caching' column has a checked checkbox with a red border around it.

# Creating API GET Endpoint

The screenshot shows the AWS API Gateway console with the following details:

- Path:** APIs > AwsClassApi (opna2oxd2e) > Resources > /movies (u68wzj) > GET
- Method Request:** /movies - GET - Method Request
- Authorization:** NONE
- Request Validator:** NONE
- API Key Required:** false
- URL Query String Parameters:**

Name	Required	Caching
firstName	<input type="checkbox"/>	<input type="checkbox"/>
lastName	<input type="checkbox"/>	<input type="checkbox"/>
status	<input type="checkbox"/>	<input type="checkbox"/>

15

# Creating API GET Endpoint

The screenshot shows the AWS API Gateway Method Execution interface. On the left, the navigation path is: APIs > AwsClassApi (opna2oxd2e) > Resources > /movies (u68wzj) > GET. The main area displays the flow of data between the Client and a Lambda function named 'Lambda AwsClassFunction'. The flow consists of four boxes: 'Method Request' (Client), 'Integration Request' (Lambda), 'Integration Response' (Lambda), and 'Method Response' (Client). A red circle labeled '16' highlights the 'Integration Request' box.

**Method Request**  
Auth: NONE  
ARN: arn:aws:execute-api:us-east-1:950922144292:opna2oxd2e/  
Query Strings: firstName, lastName, status

**Integration Request**  
Type: LAMBDA  
Region: us-east-1

**Integration Response**  
HTTP status pattern: -  
Output passthrough: Yes

**Method Response**  
HTTP Status: 200  
Models: application/json => Empty

**Lambda AwsClassFunction**

Provide information about the target backend that this method will call and whether the incoming request is passed through to the Lambda function.

**Integration type**  Lambda Function  HTTP  Mock  AWS Service  VPC Link

**Use Lambda Proxy integration**

**Lambda Region** us-east-1

**Lambda Function** AwsClassFunction

**Invoke with caller credentials**

# Creating API GET Endpoint

Lambda Function AwsClassFunction

Invoke with caller credentials

Credentials cache Do not add caller credentials to cache key

Use Default Timeout

▶ URL Path Parameters

▶ URL Query String Parameters

▶ HTTP Headers

▶ Body Mapping Templates

18

Use Default Timeout

▶ URL Path Parameters

▶ URL Query String Parameters

▶ HTTP Headers

▼ Body Mapping Templates

Request body passthrough  When no template matches the request Content-Type header   
 When there are no templates defined (recommended)   
 Never

Content-Type

No mapping templates defined. The request body will be passed through to the integration endpoint

+ Add mapping template

19

# Creating API GET Endpoint

The screenshot shows the 'Content-Type' mapping section for a GET endpoint. The 'application/json' entry is selected and highlighted with a red box. A dropdown menu is open under 'Generate template', showing options: 'Method Request passthrough' (selected and highlighted with a red box), 'Models', 'Empty', and 'Error'. A large red circle with the number '20' is overlaid on the bottom right of the window.

The screenshot shows the generated mapping template code for 'Method Request passthrough'. The code is as follows:

```
1 ## See http://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-mapping-template-reference.html
2 ## This template will pass through all parameters including path, querystring, header, stage variables,
3 ## and context through to the integration endpoint via the body/payload
4 #set($allParams = $input.params())
5 "body-json" : $input.json('$'),
6 "params" : {
7     #foreach($type in $allParams.keySet())
8         #set($params = $allParams.get($type))
9         "$type" : {
10             #foreach($paramName in $params.keySet())
11                 "$paramName" : "$util.escapeJavaScript($params.get($paramName))"
12                     #if($foreach.hasNext),#end
13             #end
14         }
15         #if($foreach.hasNext),#end
16     #end
17 }
```

A large red circle with the number '21' is overlaid on the bottom right of the window. At the bottom right of the code editor, there are 'Cancel' and 'Save' buttons, with 'Save' being highlighted by a red box.

# Binding API Gateway to Lambda

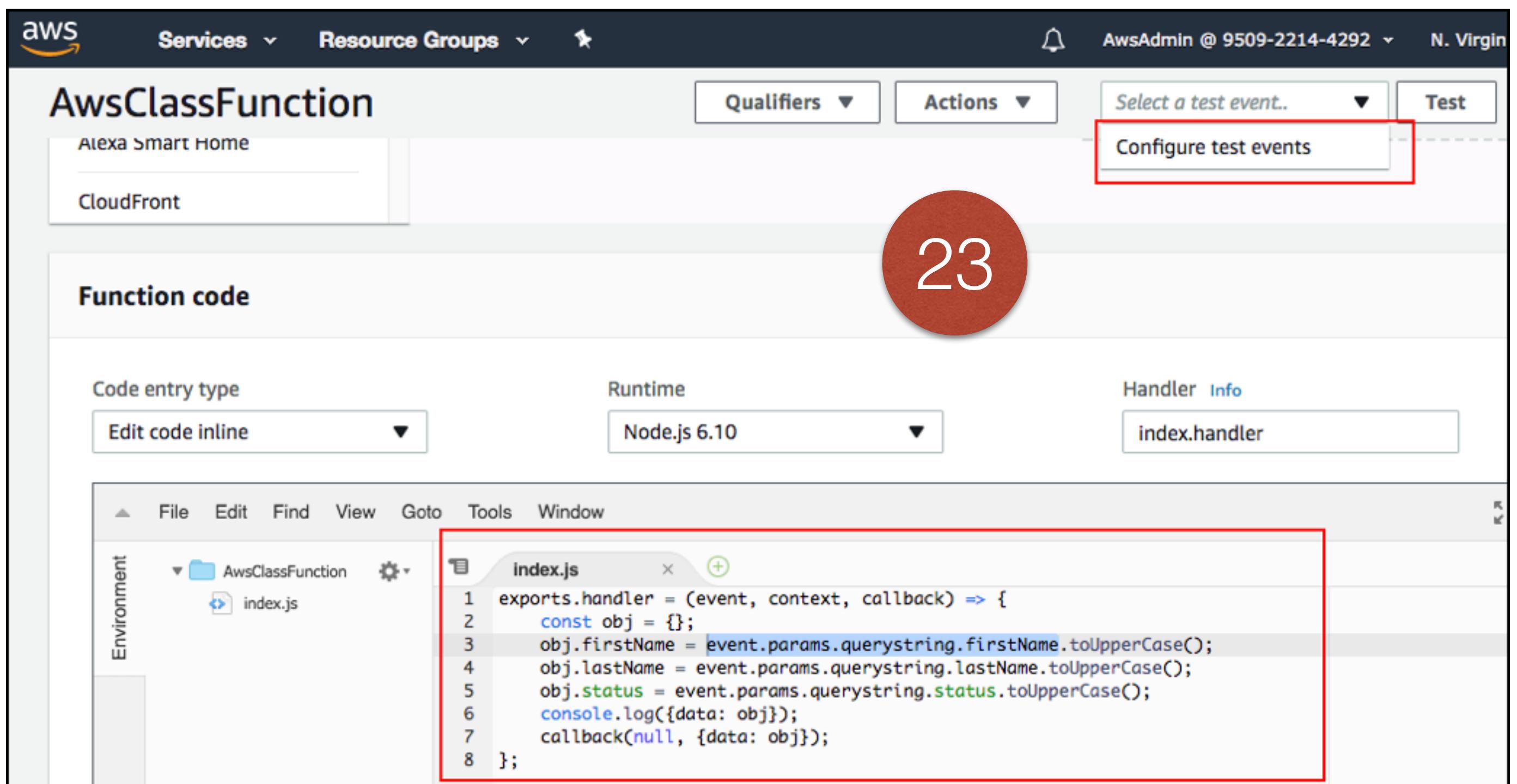
## Mapping Template Info

<http://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-mapping-template-reference.html>

The screenshot shows the AWS Lambda Functions page. At the top, there are navigation links for Services, Resource Groups, and a user profile for AwsAdmin. Below the header, it says "Lambda > Functions". A red circle with the number "22" is overlaid on the left side of the page. The main area displays a table titled "Functions (10)". The table has columns for Function name, Description, Runtime, Code size, and Last Modified. One row, "AwsClassFunction", is highlighted with a red box. The "AwsClassFunction" row contains the following information: Function name (AwsClassFunction), Description (empty), Runtime (Node.js 6.10), Code size (277 bytes), and Last Modified (30 seconds ago).

Function name	Description	Runtime	Code size	Last Modified
AwsClassFunction		Node.js 6.10	277 bytes	30 seconds ago

# Refactor Lambda



23

A screenshot of the 'Configure test event' dialog. It starts with a heading 'Configure test event' and a note: 'A function can have up to 10 test events. The events are persisted so you can switch to another computer or web browser and test your function with the same events.' Two radio buttons are shown: 'Create new test event' (unselected) and 'Edit saved test events' (selected). Below this is a section for 'Saved Test Event' with a dropdown menu set to 'MyEvent'. The bottom half of the dialog shows a JSON representation of a test event:

```
1 {
2     "params": {
3         "queryString": {
4             "firstName": "Test",
5             "lastName": "User",
6             "status": "Rocks!"
7         }
8     }
9 }
```

24

<https://github.com/reselbob/CDAwsClass/blob/Session-3/AwsClassLambda.js>

<https://github.com/reselbob/CDAwsClass/blob/Session-3/AwsClassLambdaEvent.json>

# Refactor Lambda

Lambda > Functions > AwsClassFunction ARN - arn:aws:lambda:us-east-1:950922144292:function:AwsClassFunction

AwsClassF... Qualifiers Actions MyEvent Test Save

Execution result: succeeded (logs)

Details

The area below shows the result returned by your function execution. [Learn more](#) about returning results from your function.

```
{  
  "data": {  
    "firstName": "TEST",  
    "lastName": "USER",  
    "status": "ROCKS!"  
  }  
}
```

Summary

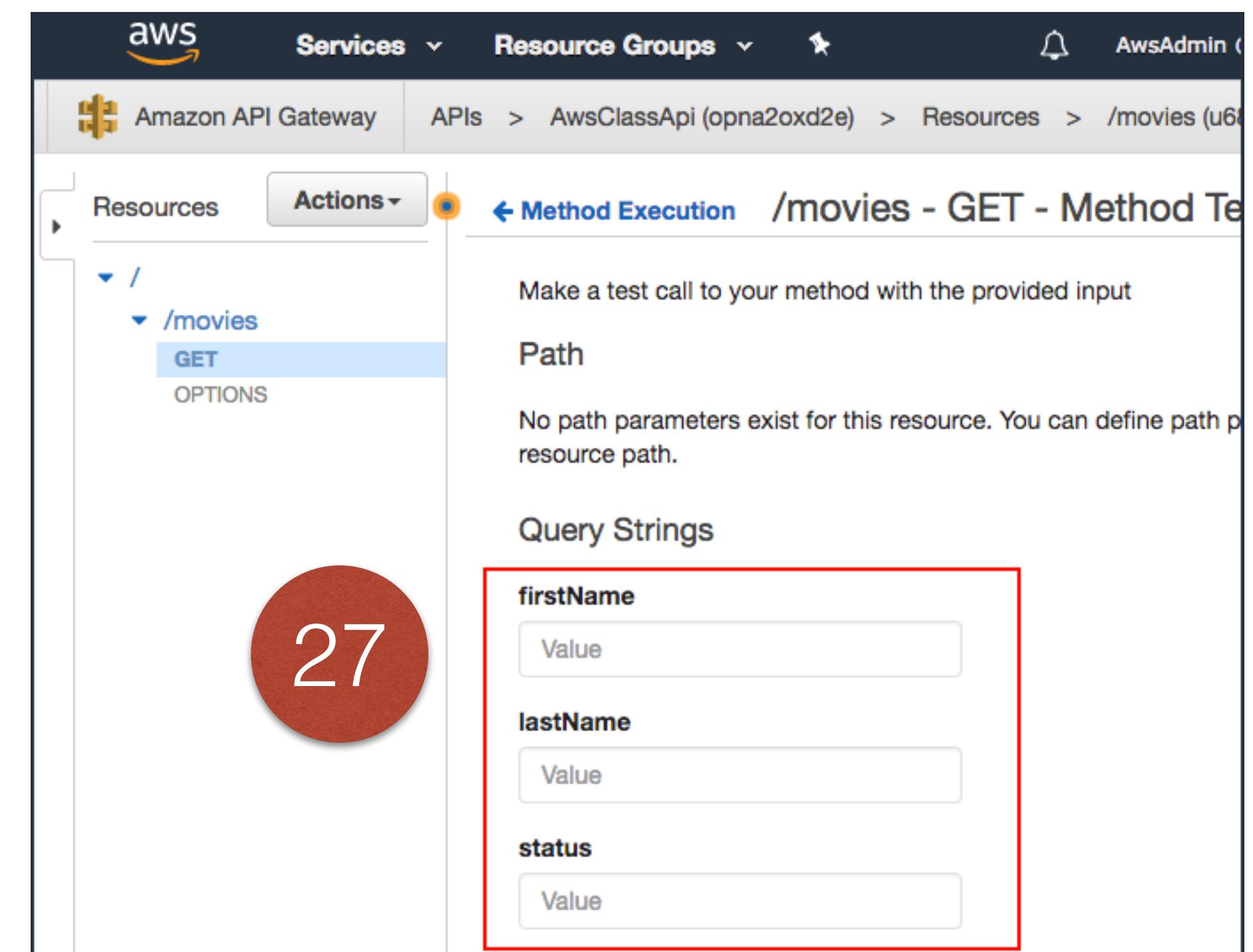
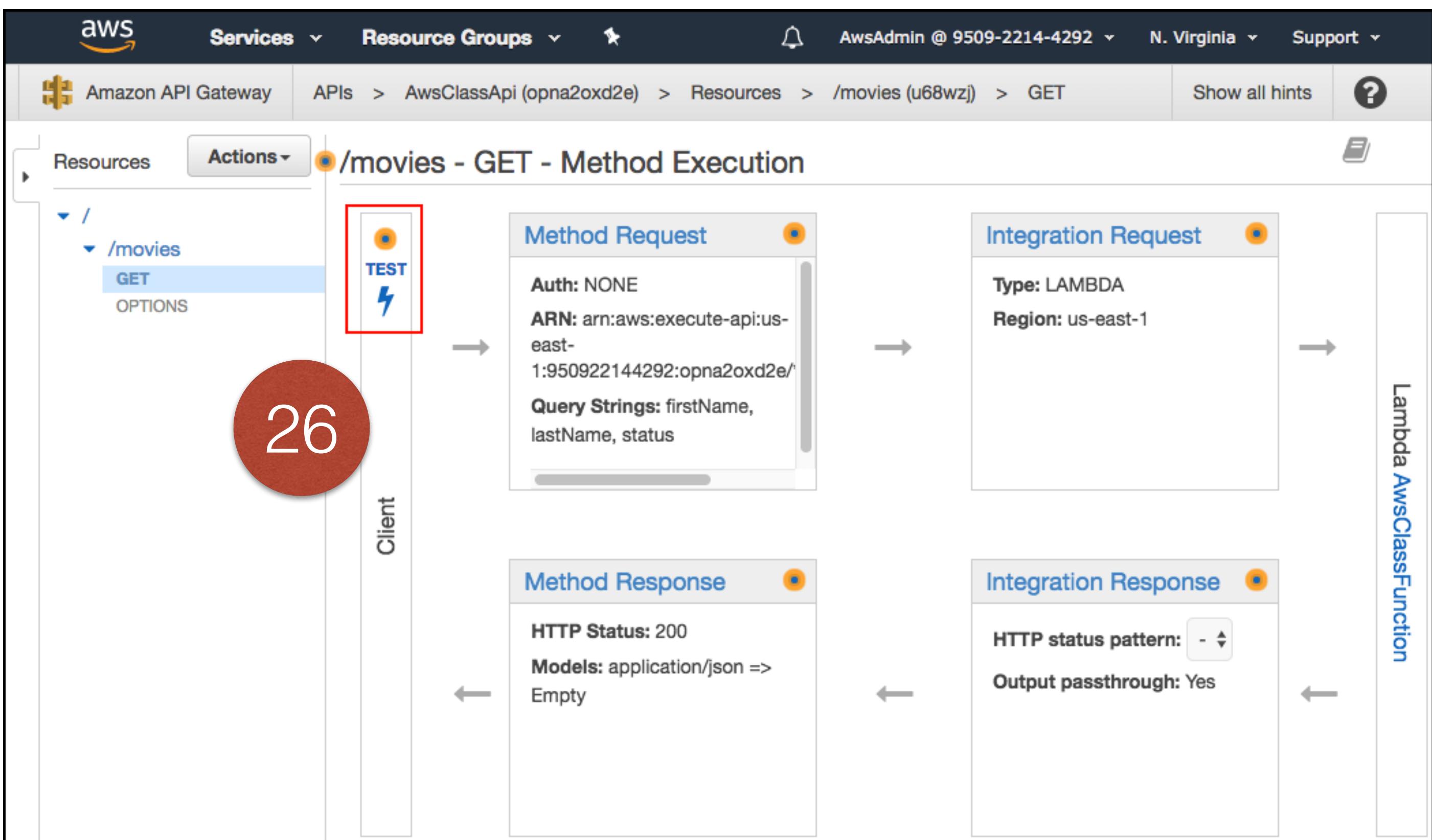
Code SHA-256	q169+AbQSC+JonRrrawgzZqlo3z6	Request ID	b53d8a0b-d966-11e7-aa99-8348e9e52d63
Duration	6.24 ms	Billed duration	100 ms
Resources configured	128 MB	Max memory used	20 MB

Log output

The area below shows the logging calls in your code. These correspond to a single row within the CloudWatch log group corresponding to this Lambda function. [Click here](#) to view the CloudWatch log group.

25

# Testing the API



# Testing the API

The screenshot shows the AWS API Gateway Method Test interface. The URL is `/movies - GET - Method Test`. The left sidebar shows a tree structure with `/movies` expanded, showing `GET` and `OPTIONS` methods. A red circle with the number 28 is overlaid on the left side of the interface. The main area contains a form for testing the `GET` method. It has sections for **Path** (empty), **Query Strings**, and **Headers**. The **Query Strings** section is highlighted with a red border and contains three fields: `firstName` (Cool), `lastName` (User), and `status` (Rocks).

The screenshot shows the AWS API Gateway Method Test interface for the same endpoint. The URL is `/movies - GET - Method Test`. The left sidebar shows the same tree structure. A red circle with the number 29 is overlaid on the right side of the interface. The main area shows the test results. It indicates that no path parameters exist. The **Query Strings** section is populated with `firstName: Cool`, `lastName: User`, and `status: Rocks`. The **Headers** section is empty. The **Stage Variables** section states there are no stage variables. The **Request Body** section notes that request bodies are not supported for GET methods. A blue button labeled **Test** is visible at the bottom right.

# Testing the API

30

Request: /movies?lastName=Cool&status=User&firstName=Rocks

Status: 200

Latency: 381 ms

Response Body

```
{  
  "data": {  
    "firstName": "ROCKS",  
    "lastName": "COOL",  
    "status": "USER"  
  }  
}
```

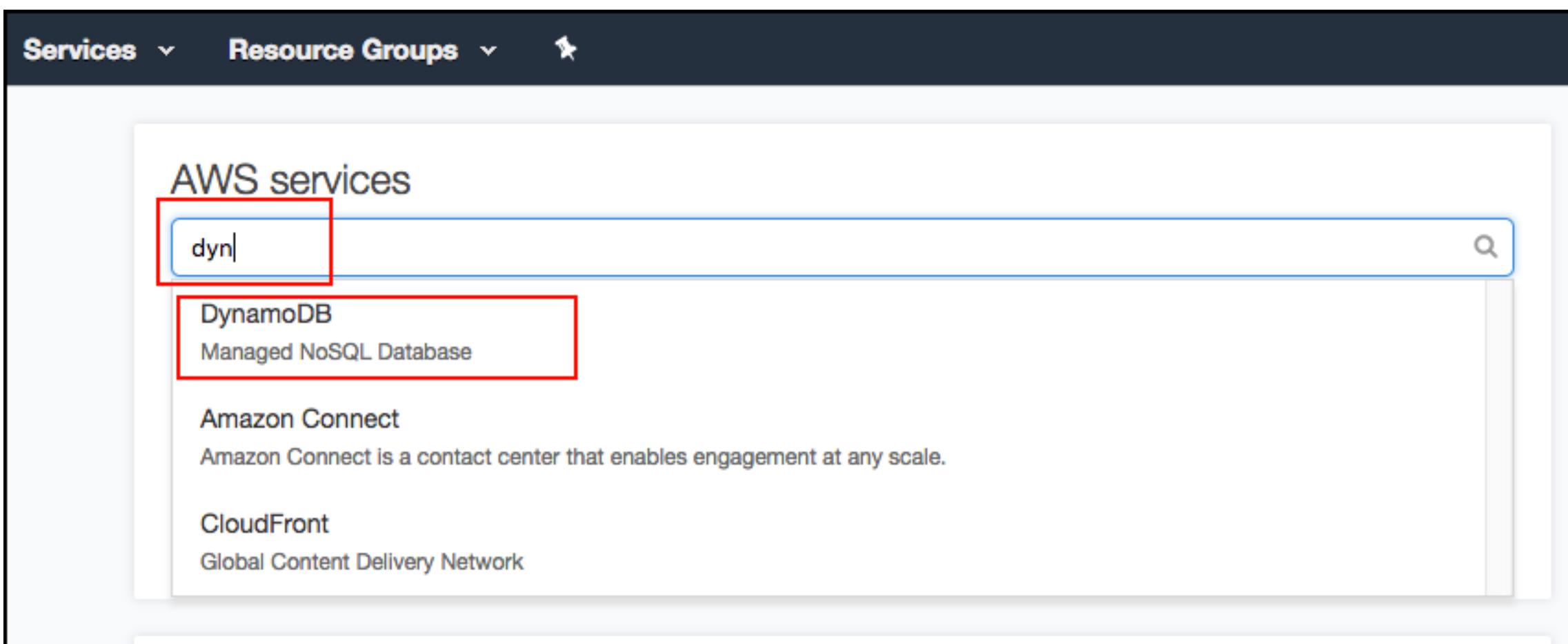
Response Headers

```
{"X-Amzn-Trace-Id": "sampled=0;root=1-5a2618d7-efde4c7547217b6929a63bee", "Content-Type": "application/json"}
```

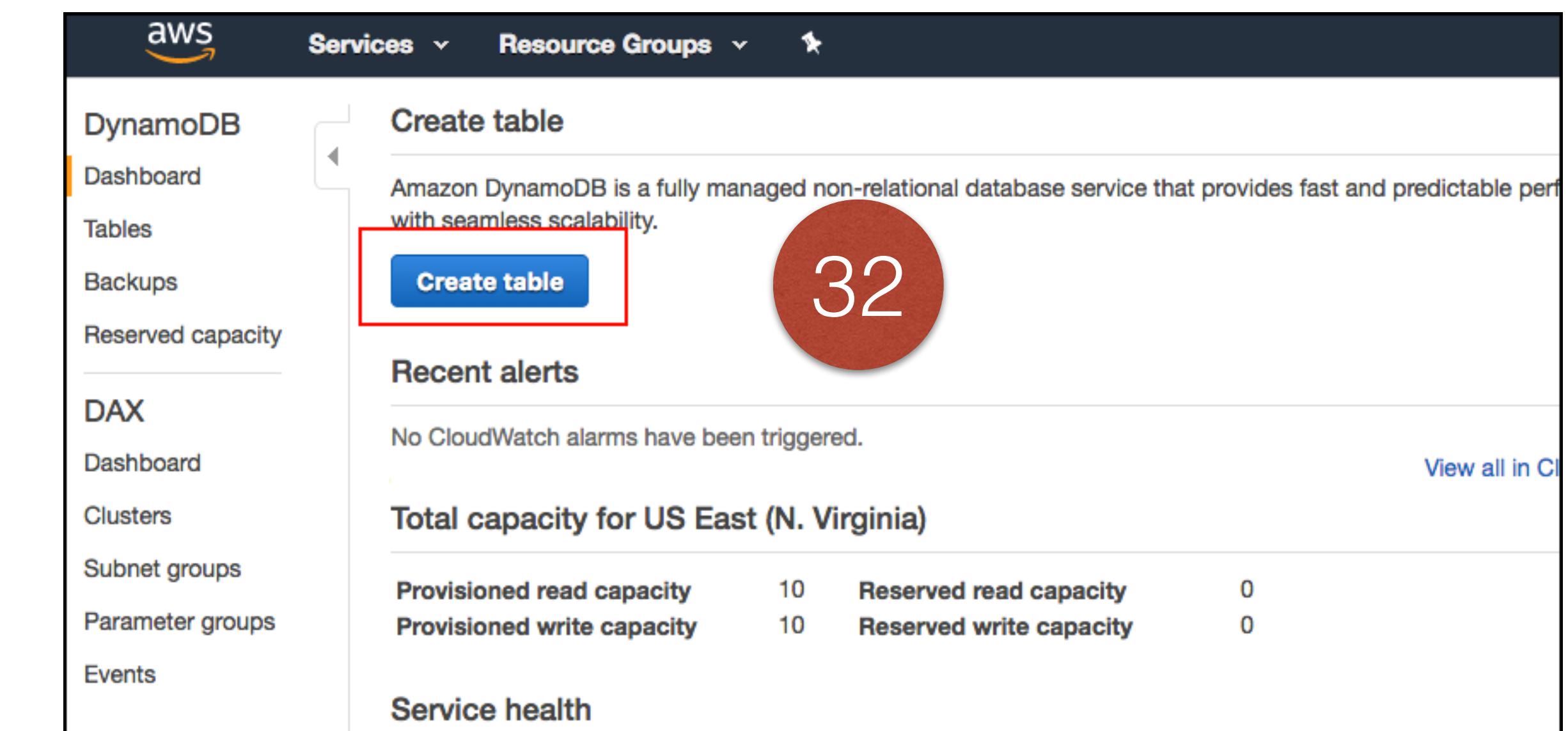
Logs

```
Execution log for request test-request
```

# Creating the DynamoDB Table



31



@reselbob

# Creating the DynamoDB Table

Create DynamoDB table

DynamoDB is a schema-less database that only requires a table name and primary key. The table's primary key is made up of one or two attributes that uniquely identify items, partition the data, and sort data within each partition.

Table name\* Ratings

Primary key\* Partition key movieTitle String

Add sort key

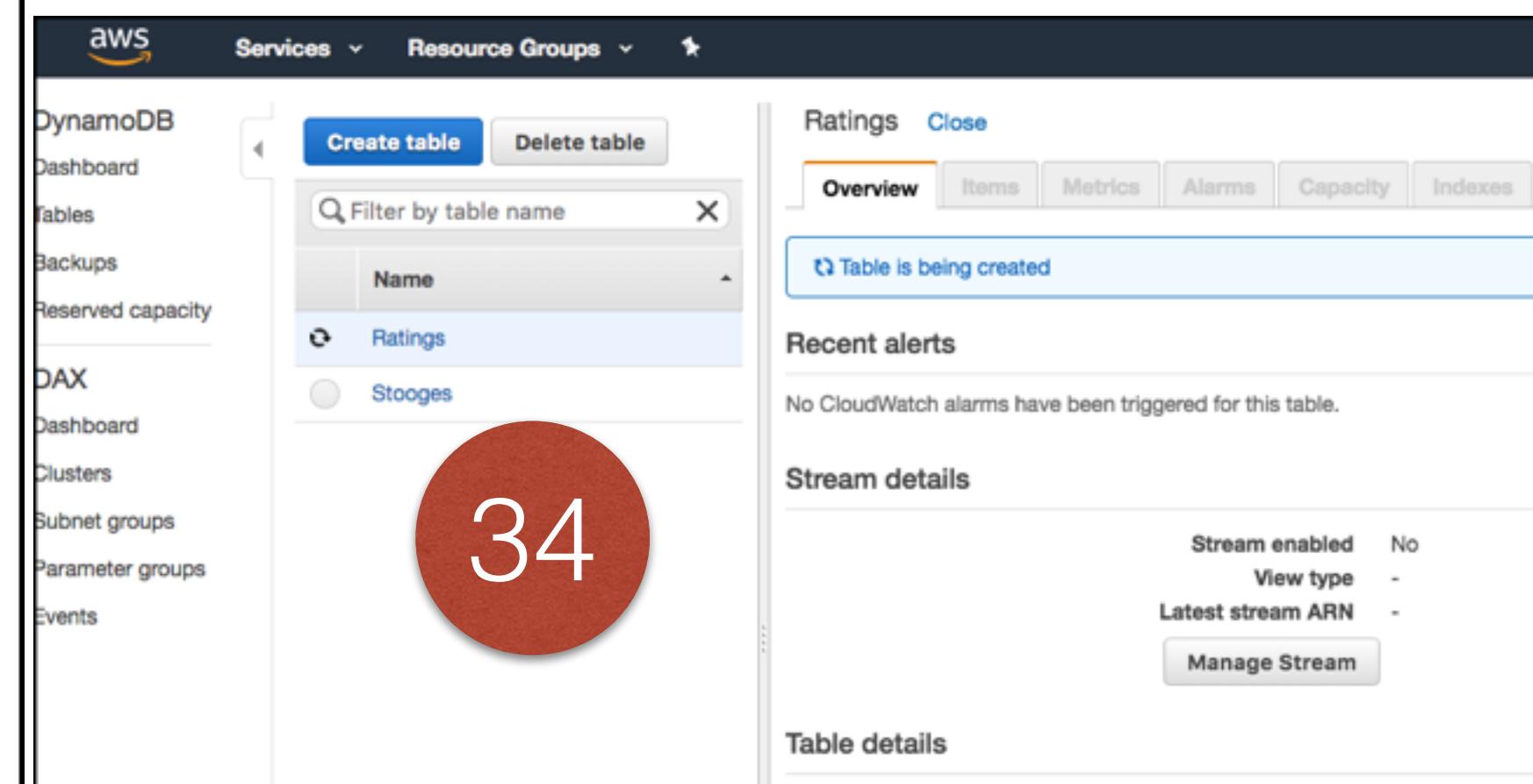
Table settings

Default settings provide the fastest way to get started with your table. You can modify these default settings now or after your table has been created.

Use default settings

- No secondary indexes.
- Provisioned capacity set to 5 reads and 5 writes.
- Basic alarms with 80% upper threshold using SNS topic "dynamodb".
- On-Demand Backup and Restore Enabled NEW!

Cancel Create



Ratings Close

Overview Items Metrics Alarms Capacity Indexes

Table is being created

Recent alerts

No CloudWatch alarms have been triggered for this table.

Stream details

Stream enabled No

View type -

Latest stream ARN -

Manage Stream

Table details

Name
Ratings
Stooges

# Questions and Answers

End of Day 1  
Review and Evaluation

**Please take the Day 1 Evaluation:**

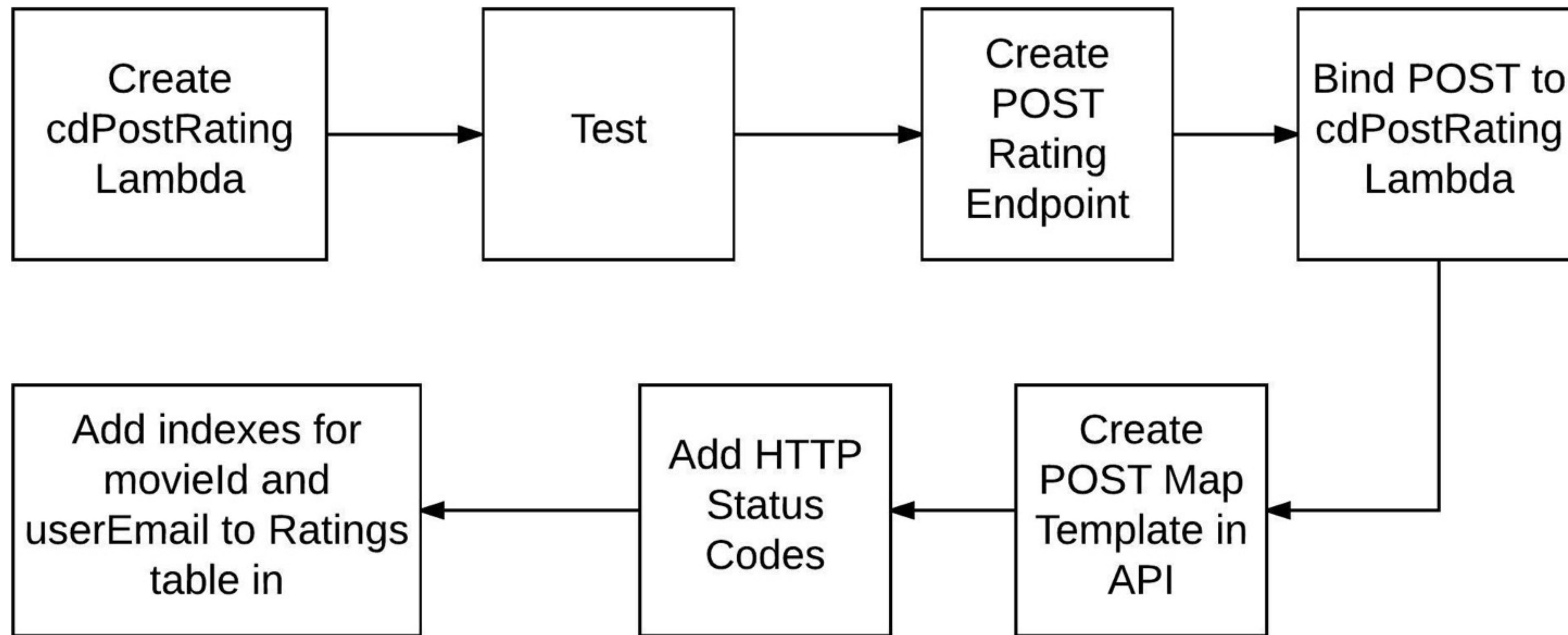
<https://www.surveymonkey.com/r/25H6BSB>

Bob Reselman  
[reselbob@gmail.com](mailto:reselbob@gmail.com)

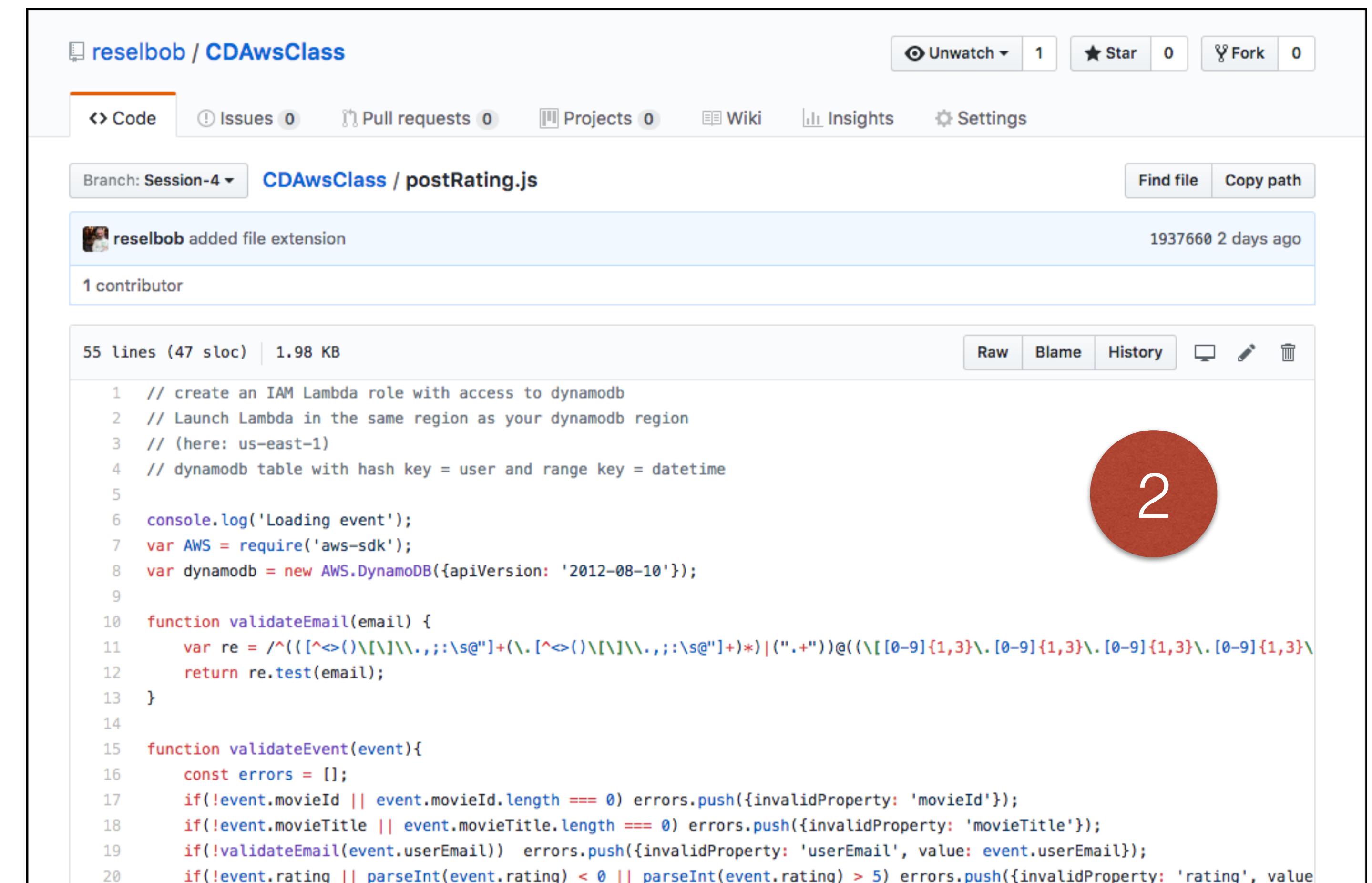
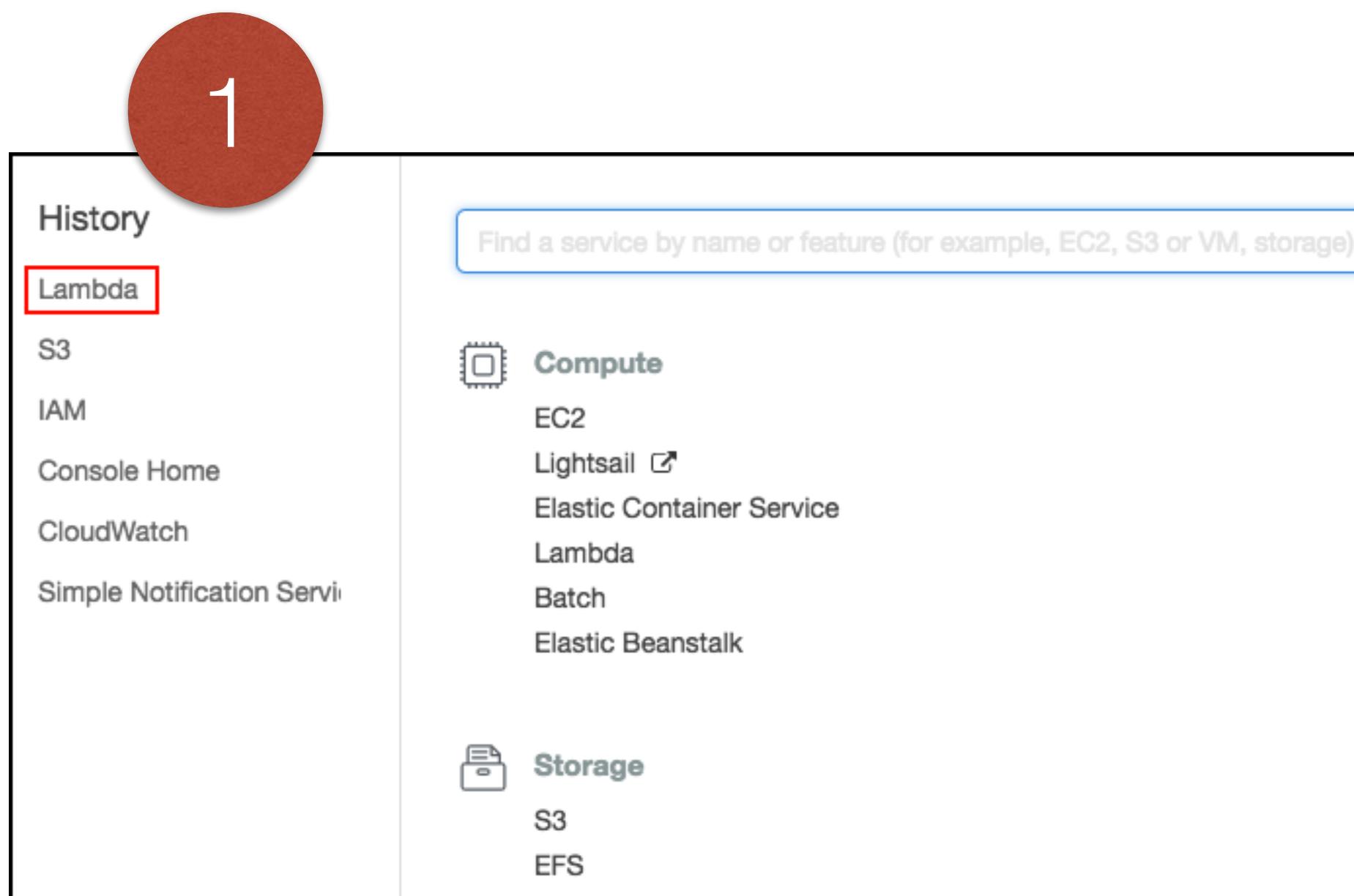
# Agenda for Day 2

- **Session 4**
  - Creating the POST Rating API Endpoint
  - Creating the postRating Lambda Function
  - Creating the Rating Index in DynamoDB
  - Supporting Status Codes
- **Session 5**
  - Creating the getMovies Lambda Function
  - Refactor the GET Movies API Endpoint
  - Creating the getRating Lambda Function
  - Creating the GET Ratings API Endpoint
  - Publishing the API to a Test Deployment
- **Session 6**
  - Refactoring the Movie Rater UI
  - Creating and Binding the SNS Topic
  - Subscribing to the MovieRater Topic

# Session 4



# Create POST Rating Lambda



# Create POST Rating Lambda

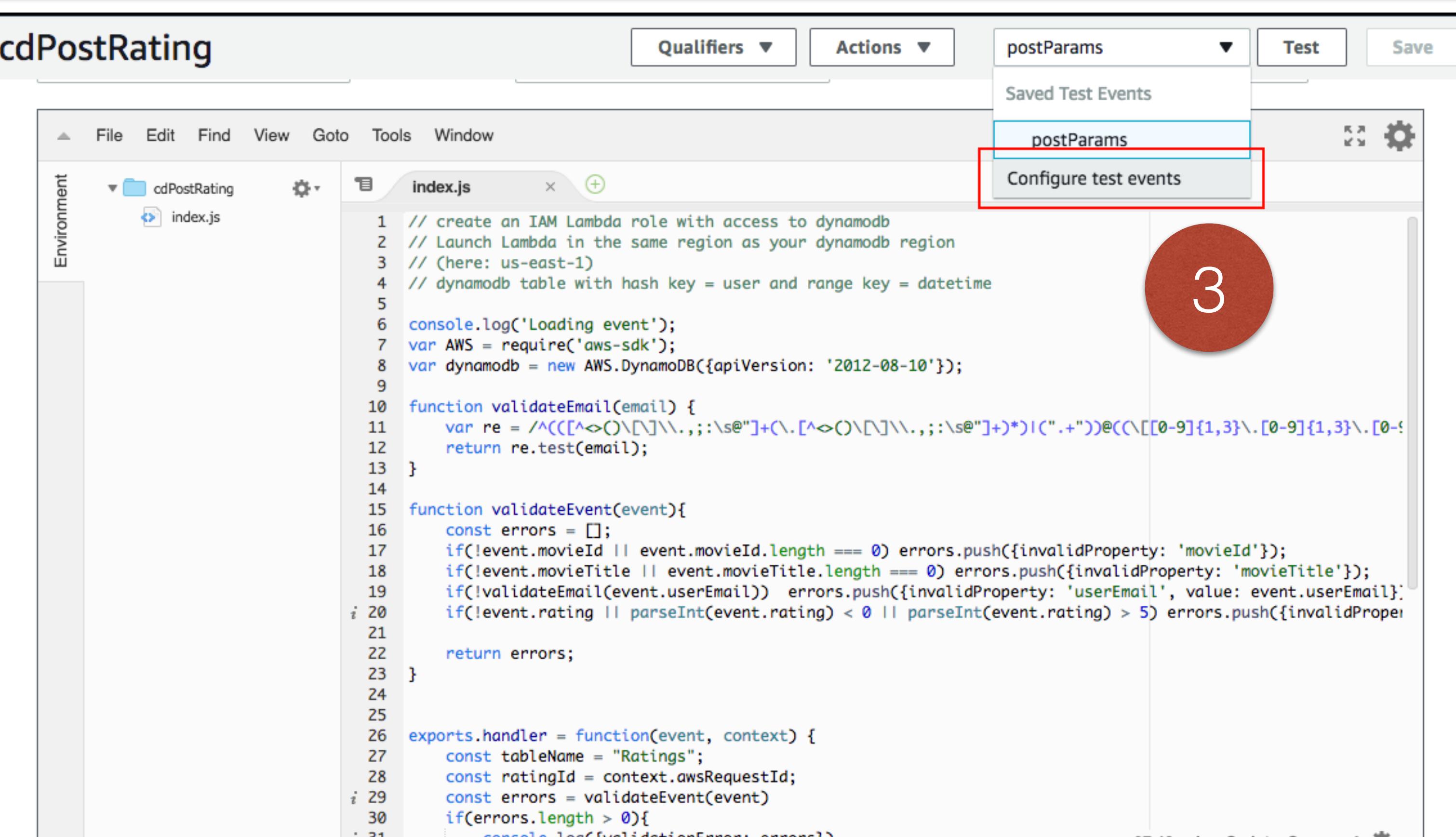
cdPostRating

Qualifiers ▾ Actions ▾ postParams ▾ Test Save

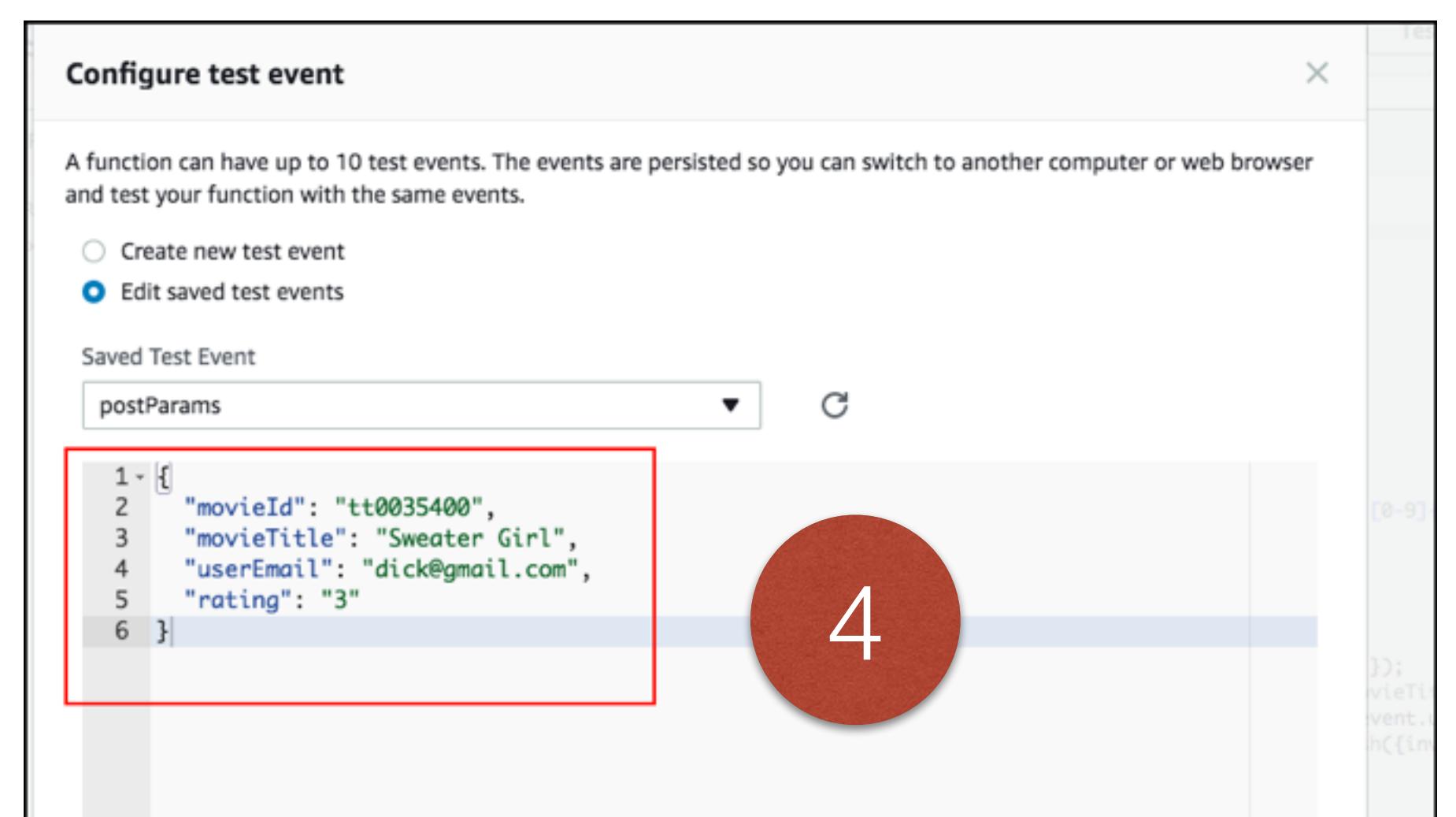
Saved Test Events

postParams

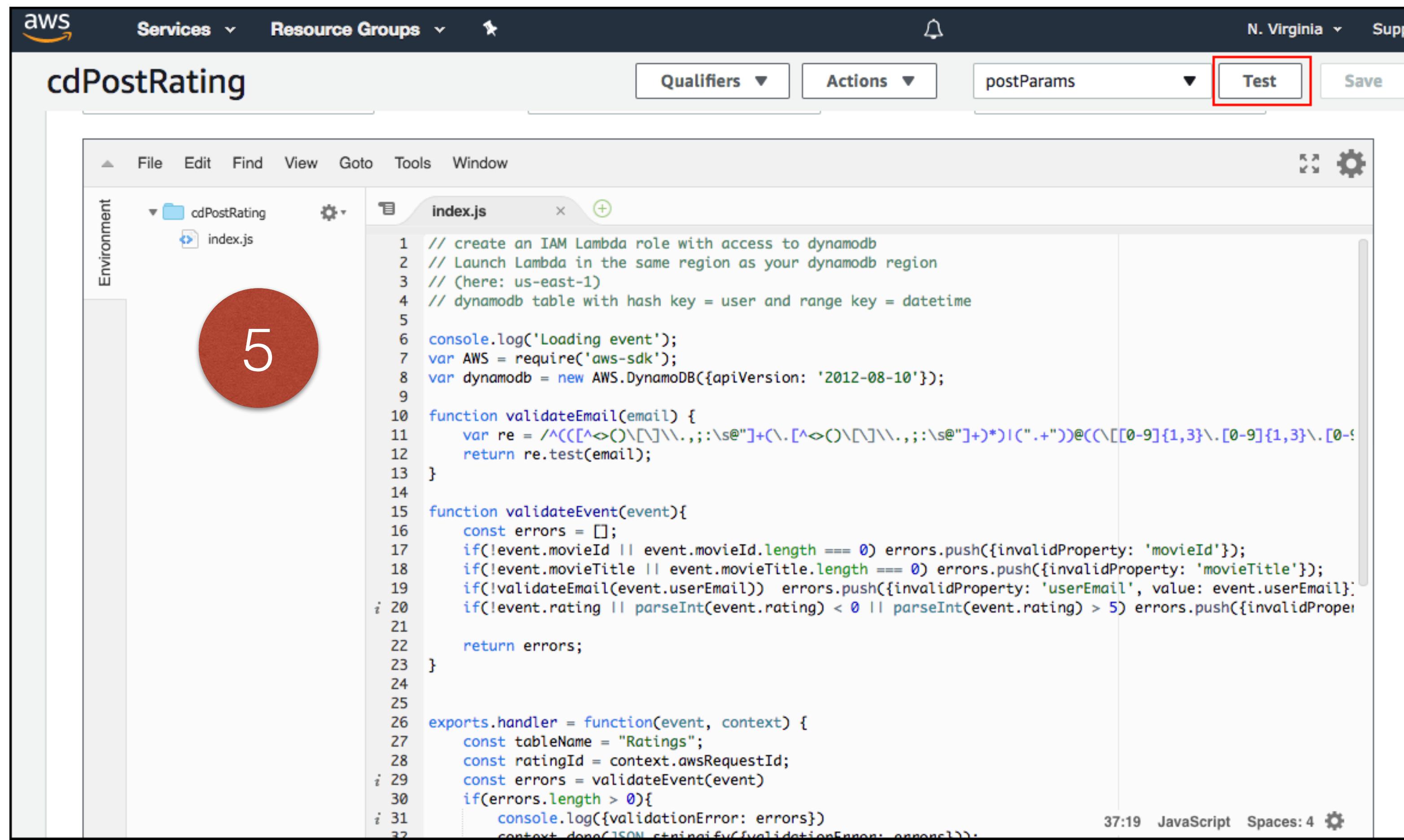
Configure test events



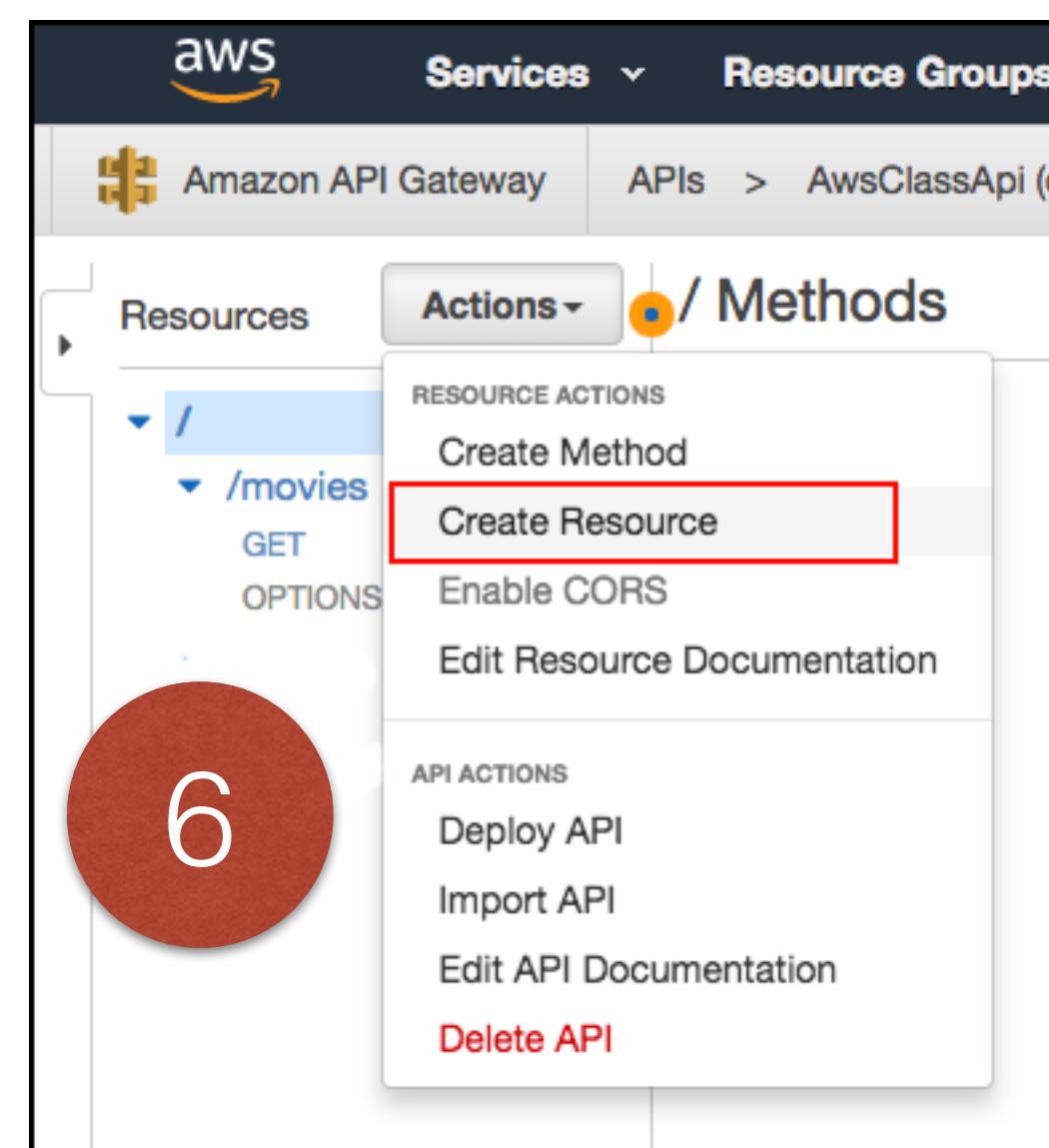
```
1 // create an IAM Lambda role with access to dynamodb
2 // Launch Lambda in the same region as your dynamodb region
3 // (here: us-east-1)
4 // dynamodb table with hash key = user and range key = datetime
5
6 console.log('Loading event');
7 var AWS = require('aws-sdk');
8 var dynamodb = new AWS.DynamoDB({apiVersion: '2012-08-10'});
9
10 function validateEmail(email) {
11     var re = /^[^@]+@[^\.\n]+\.\w+$/;
12     return re.test(email);
13 }
14
15 function validateEvent(event){
16     const errors = [];
17     if(!event.movieId || event.movieId.length === 0) errors.push({invalidProperty: 'movieId'});
18     if(!event.movieTitle || event.movieTitle.length === 0) errors.push({invalidProperty: 'movieTitle'});
19     if(!validateEmail(event.userEmail)) errors.push({invalidProperty: 'userEmail', value: event.userEmail});
20     if(!event.rating || parseInt(event.rating) < 0 || parseInt(event.rating) > 5) errors.push({invalidProperty: 'rating', value: event.rating});
21
22     return errors;
23 }
24
25
26 exports.handler = function(event, context) {
27     const tableName = "Ratings";
28     const ratingId = context.awsRequestId;
29     const errors = validateEvent(event);
30     if(errors.length > 0){
31         console.log(`Validation Errors: ${errors}`);
32     }
33 }
```



# Test POST Rating Lambda

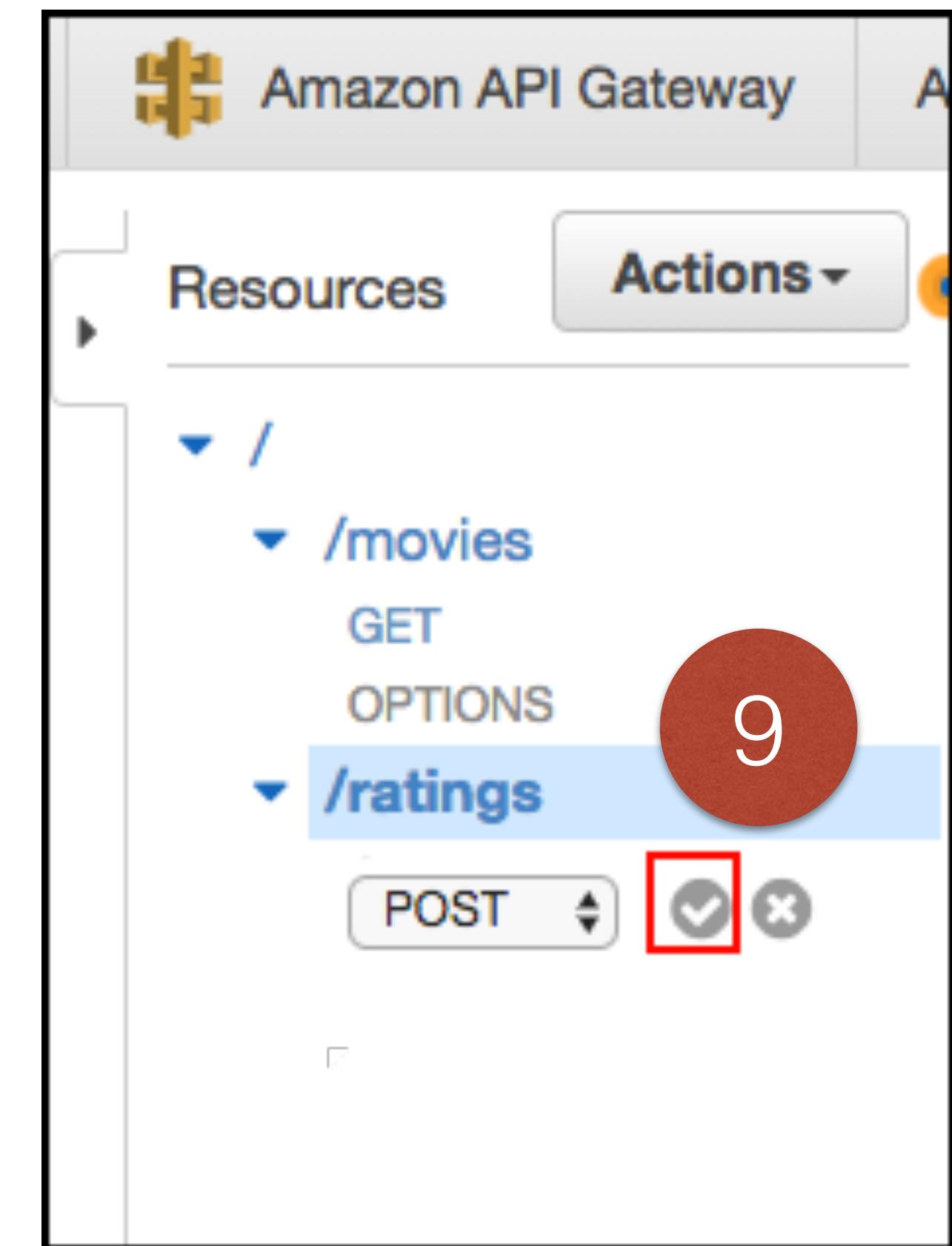
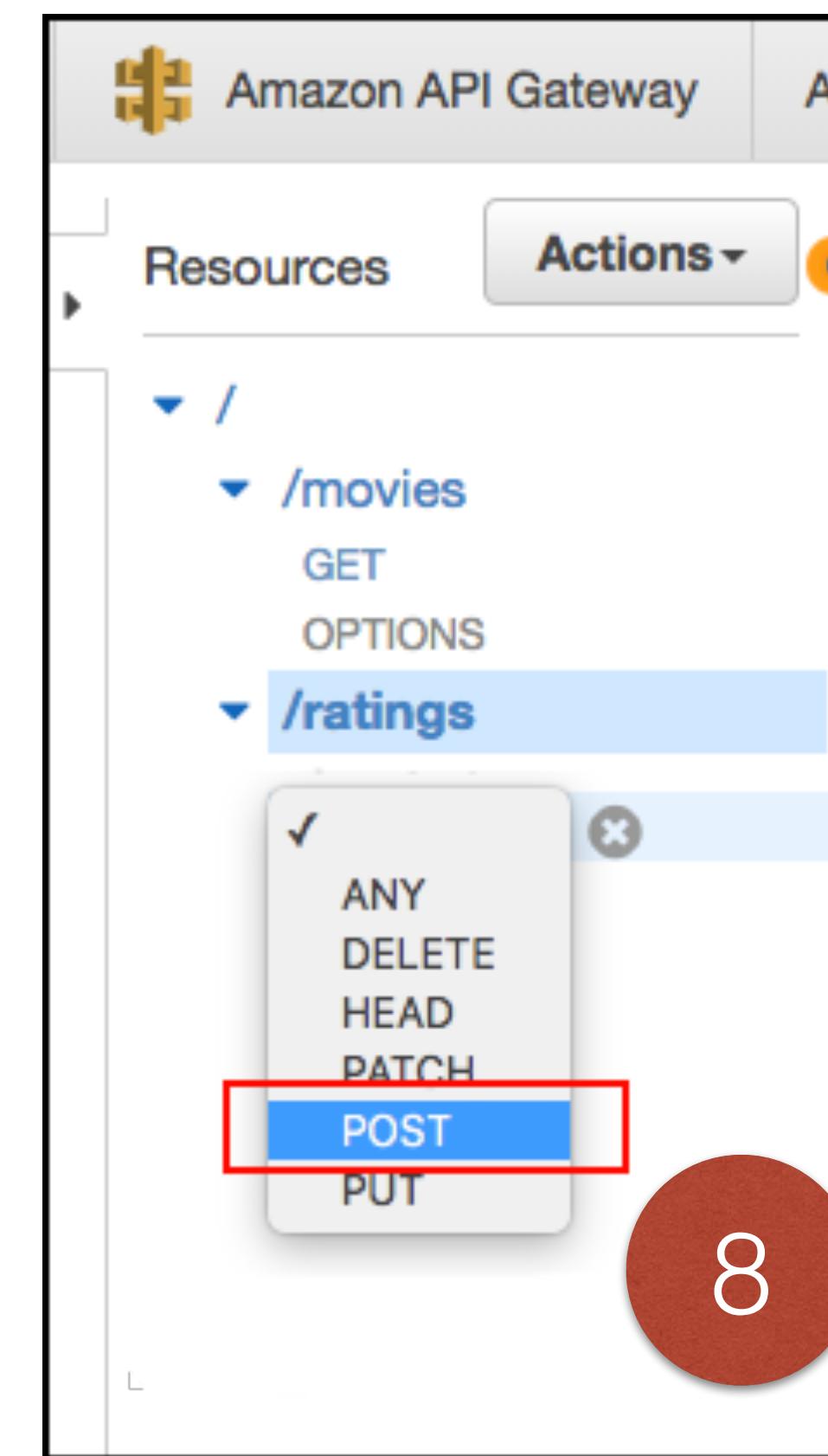
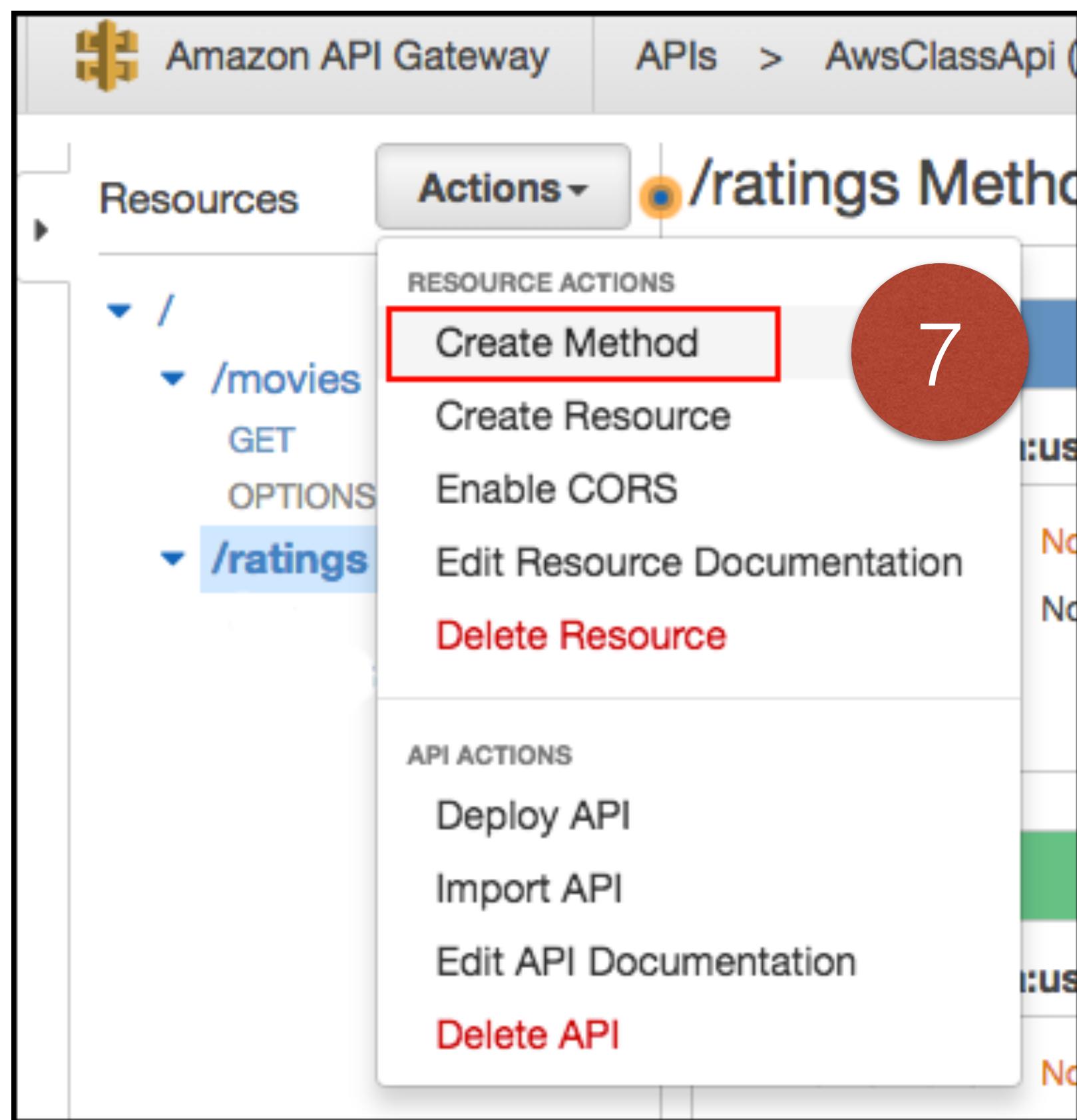


# Create POST Rating Endpoint



A screenshot of the Amazon API Gateway 'New Child Resource' creation page. The URL in the browser is `APIs > AwsClassApi (opna2oxd2e) > Resources > / (685iyb8e0b) > Create`. The left sidebar shows a tree structure with a node expanded to show 'GET' and 'OPTIONS' methods. The main form has a heading 'New Child Resource' and instructions: 'Use this page to create a new child resource for your resource.' It includes fields for 'Resource Name\*' (set to 'ratings') and 'Resource Path\*' (set to '/ ratings'). A note explains path parameters like '{username}'. There's a checkbox for 'Enable API Gateway CORS' which is checked. At the bottom are 'Cancel' and 'Create Resource' buttons, with 'Create Resource' highlighted with a red box. A large red circle with the number '7' is overlaid on the bottom-left corner of the screenshot.

# Create POST Rating Endpoint



# Create POST Rating Endpoint

Amazon API Gateway   APIs > AwsClassApi (opna20xd2e) > Resources > /ratings (ivryrv) > POST   Show all hints ?

Resources Actions ▾

Choose the integration point for your new method.

Integration type  Lambda Function i  
 HTTP i  
 Mock i  
 AWS Service i  
 VPC Link i

Use Lambda Proxy integration  i

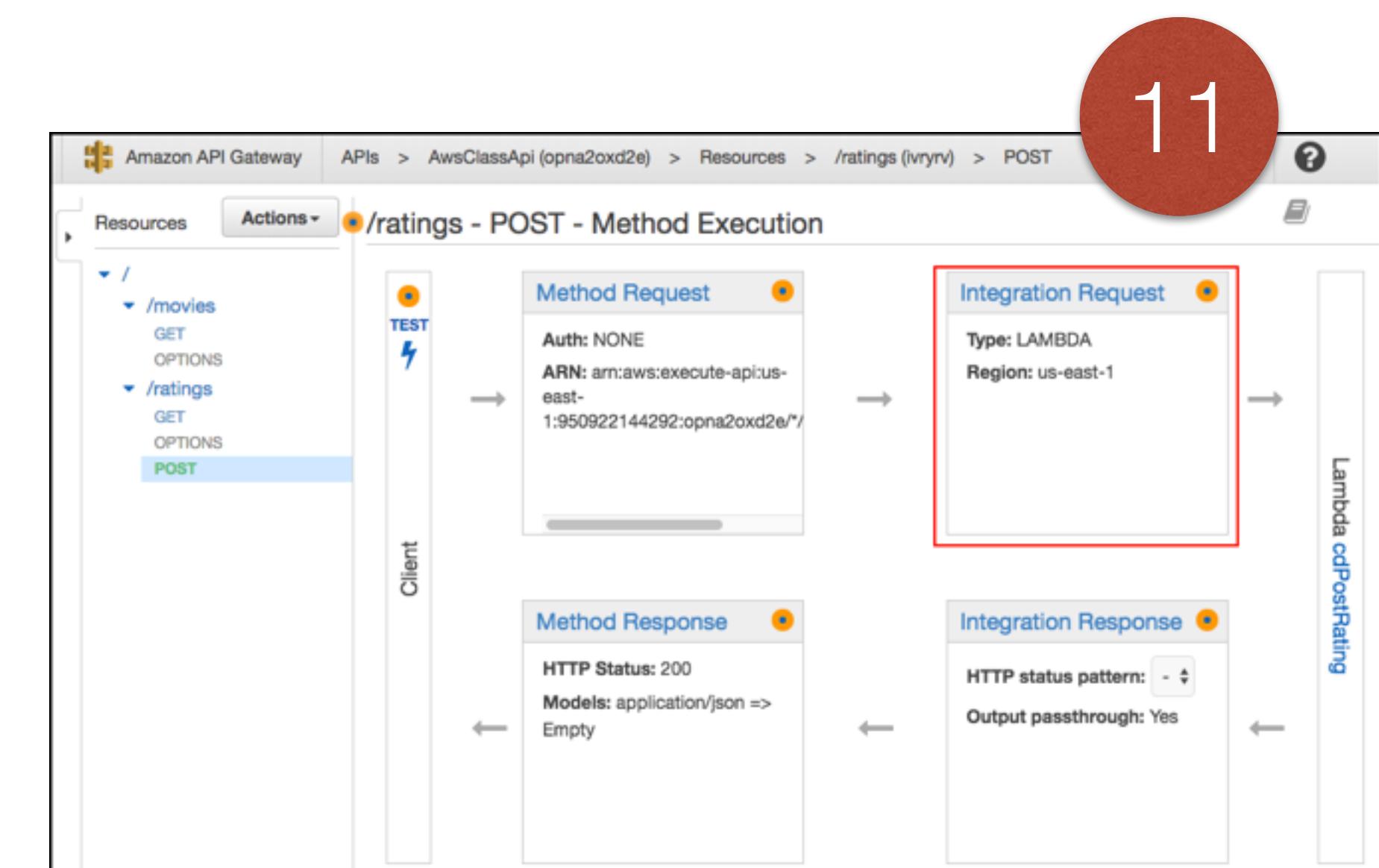
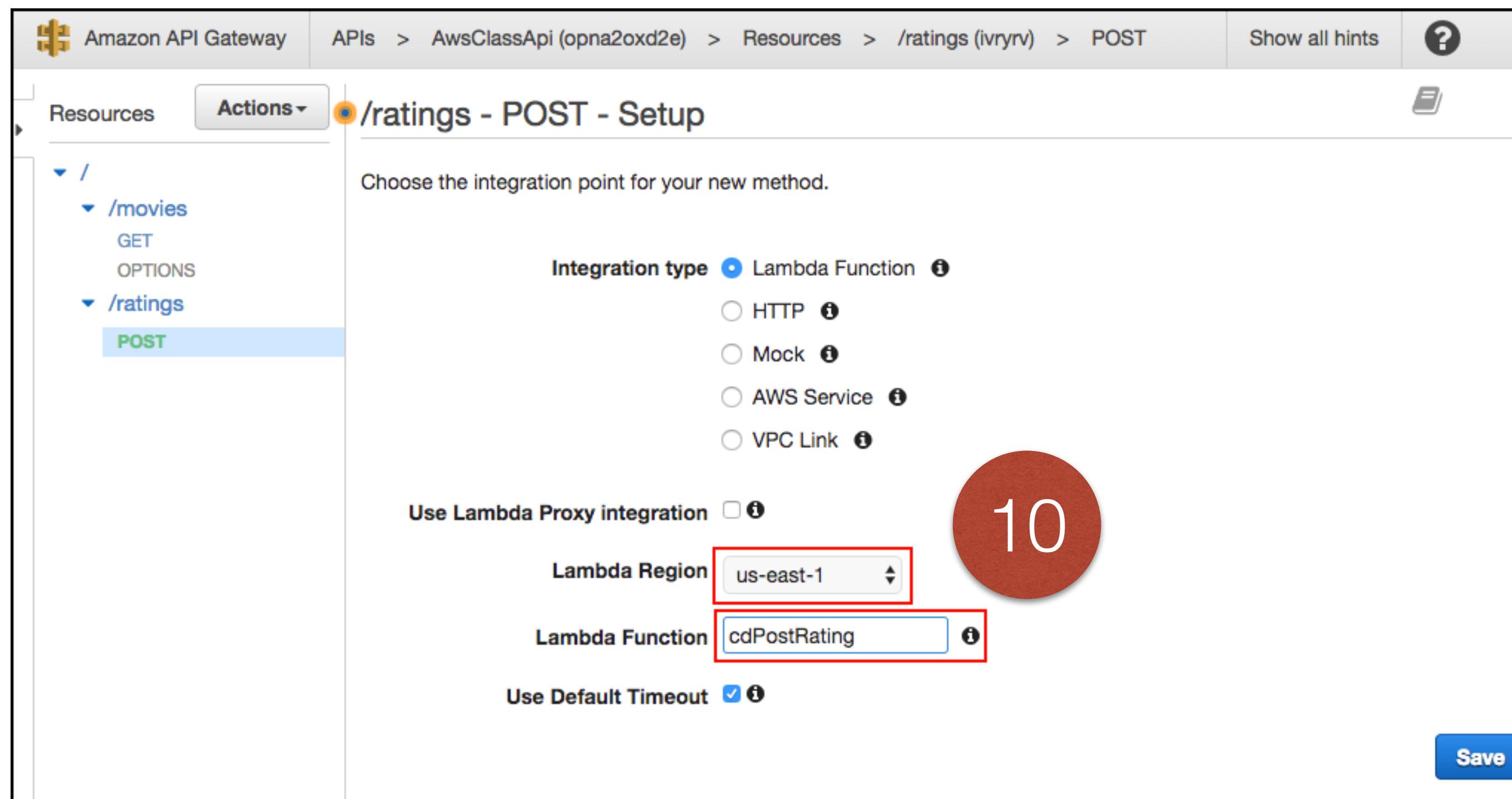
Lambda Region us-east-1

Lambda Function cdPostRating i

Use Default Timeout  i

Save

10



# Create POST Rating Endpoint

The screenshot shows the AWS Lambda function configuration interface. In the left sidebar, there are sections for 'HTTP Headers' and 'Body Mapping Templates'. Under 'Body Mapping Templates', there is a dropdown for 'Request body passthrough' with three options: 'When no template matches the request Content-Type header' (selected), 'When there are no templates defined (recommended)', and 'Never'. Below this is a 'Content-Type' field set to 'application/json', which is highlighted with a red box. A red box also highlights the 'Add mapping template' button.

12

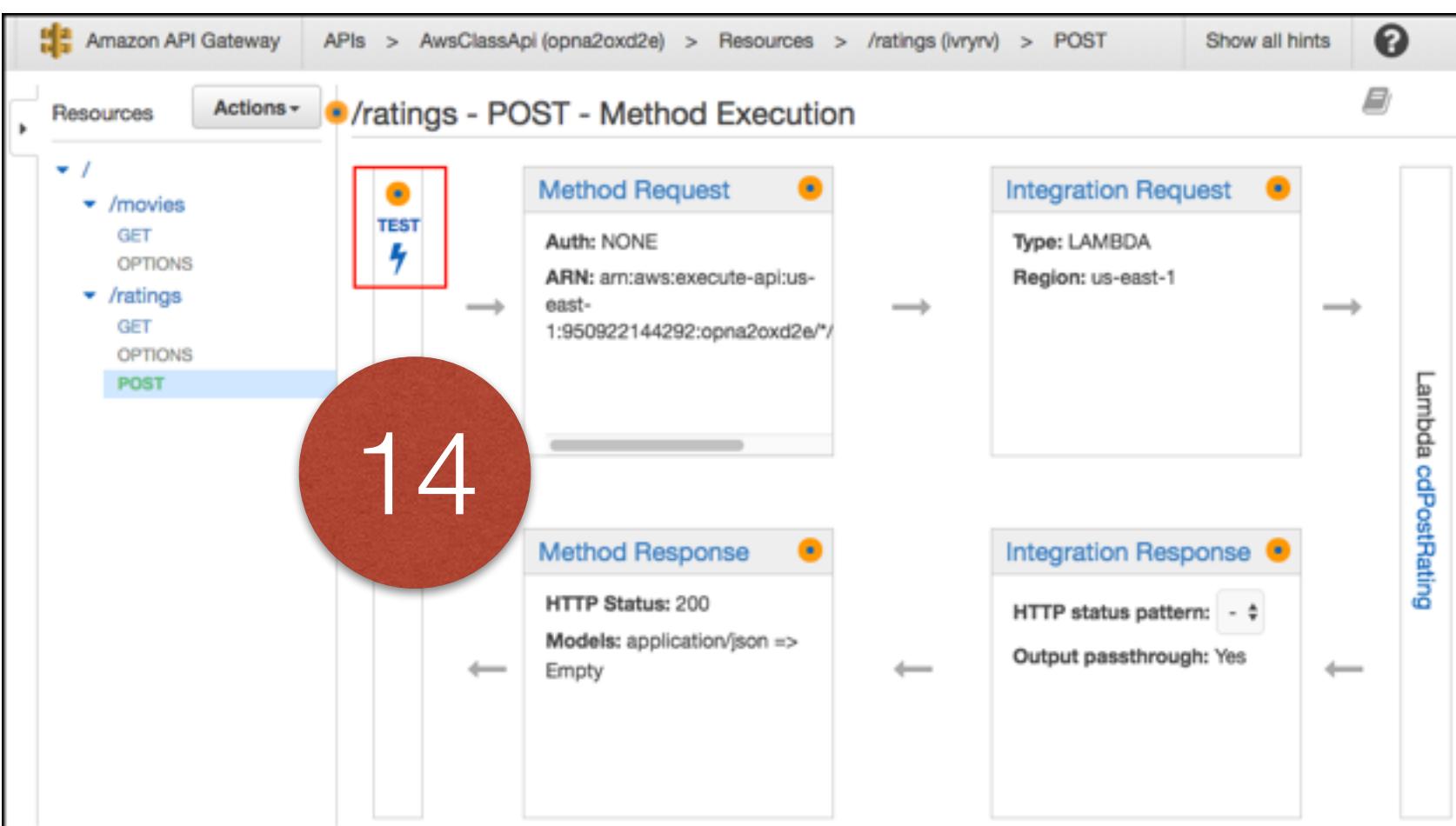
The screenshot shows the Amazon API Gateway POST method configuration interface. The path is 'APIs > AwsClassApi (opna2oxd2e) > Resources > /ratings (ivryrv) > POST'. The 'Actions' tab is selected. On the right, there is a 'Content-Type' field set to 'application/json'. Below it is a 'Generate template:' dropdown and a code editor containing a JSON template. The code is highlighted with a red box:

```
1 {  
2   "movieId" : $input.json('$.movieId'),  
3   "movieTitle": $input.json('$.movieTitle'),  
4   "userEmail": $input.json('$ userEmail'),  
5   "rating": $input.json('$ rating')  
6 }
```

At the bottom right are 'Cancel' and 'Save' buttons, with 'Save' highlighted by a red box.

13

# Create POST Rating Endpoint



The screenshot shows the 'Request Body' section with a red box around the JSON input field. The JSON content is:

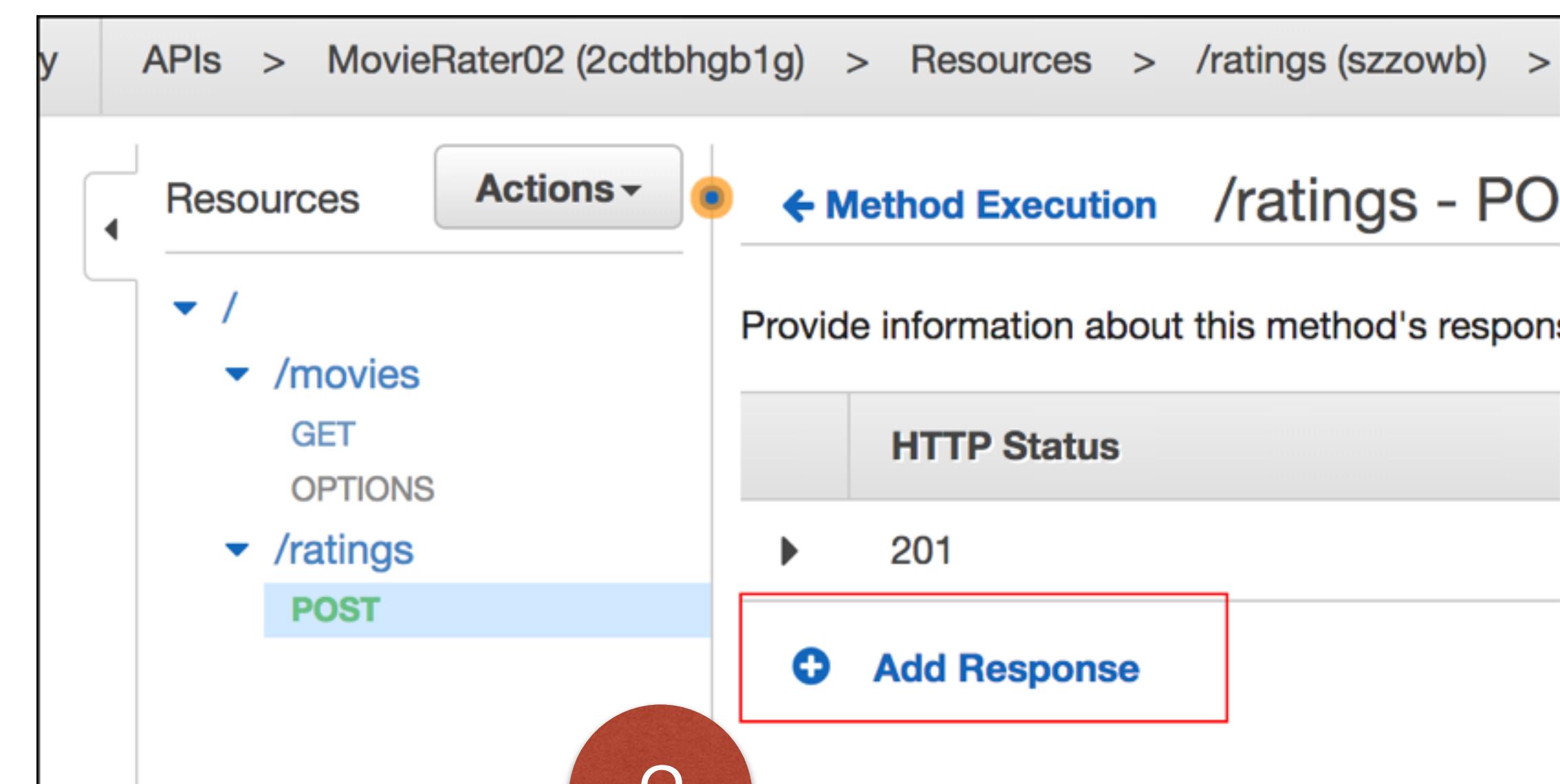
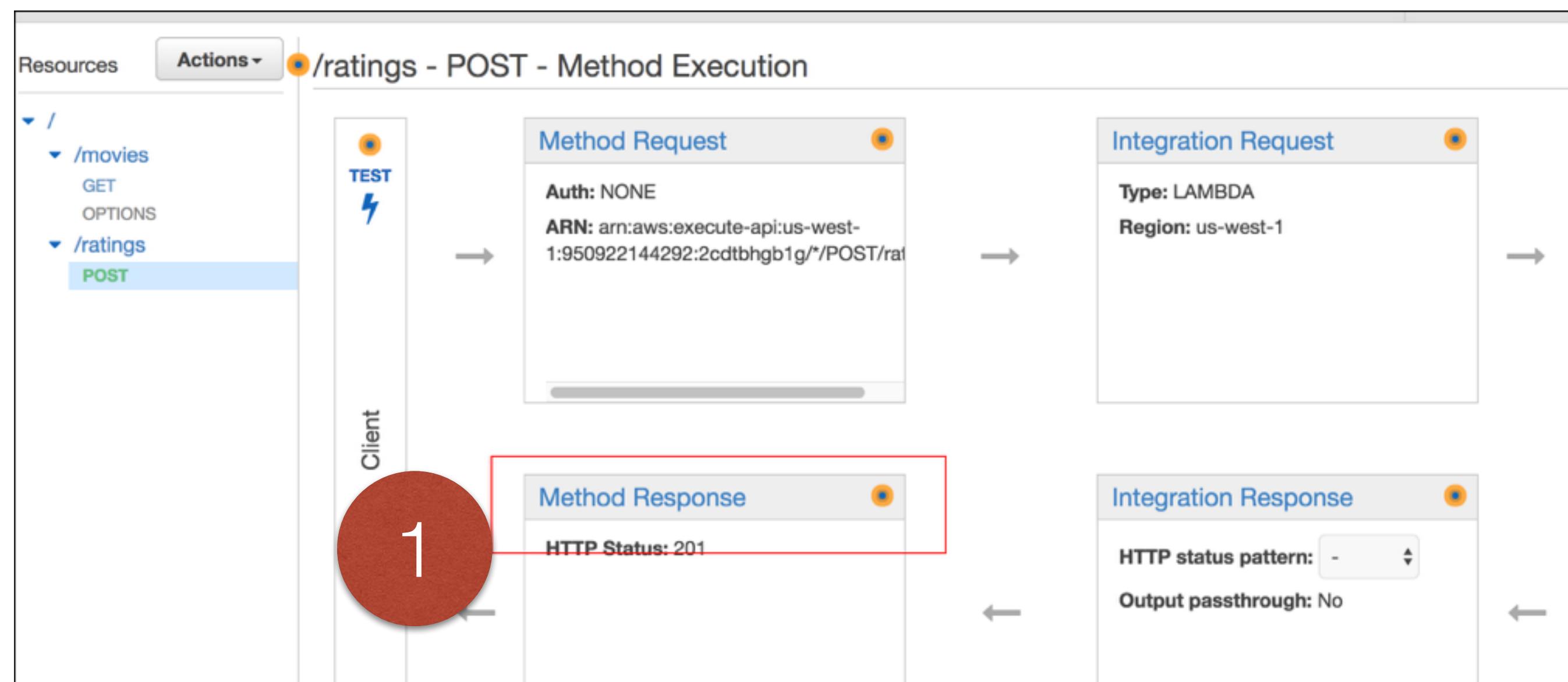
```
1 {  
2   "movieId" : "tt0035400",  
3   "movieTitle": "Sweater Girl",  
4   "userEmail": "dick@gmail.com",  
5   "rating": "2"  
6 }
```

A red box also surrounds the 'Test' button at the bottom right. A large red circle with the number '15' is overlaid on the right side of the interface.

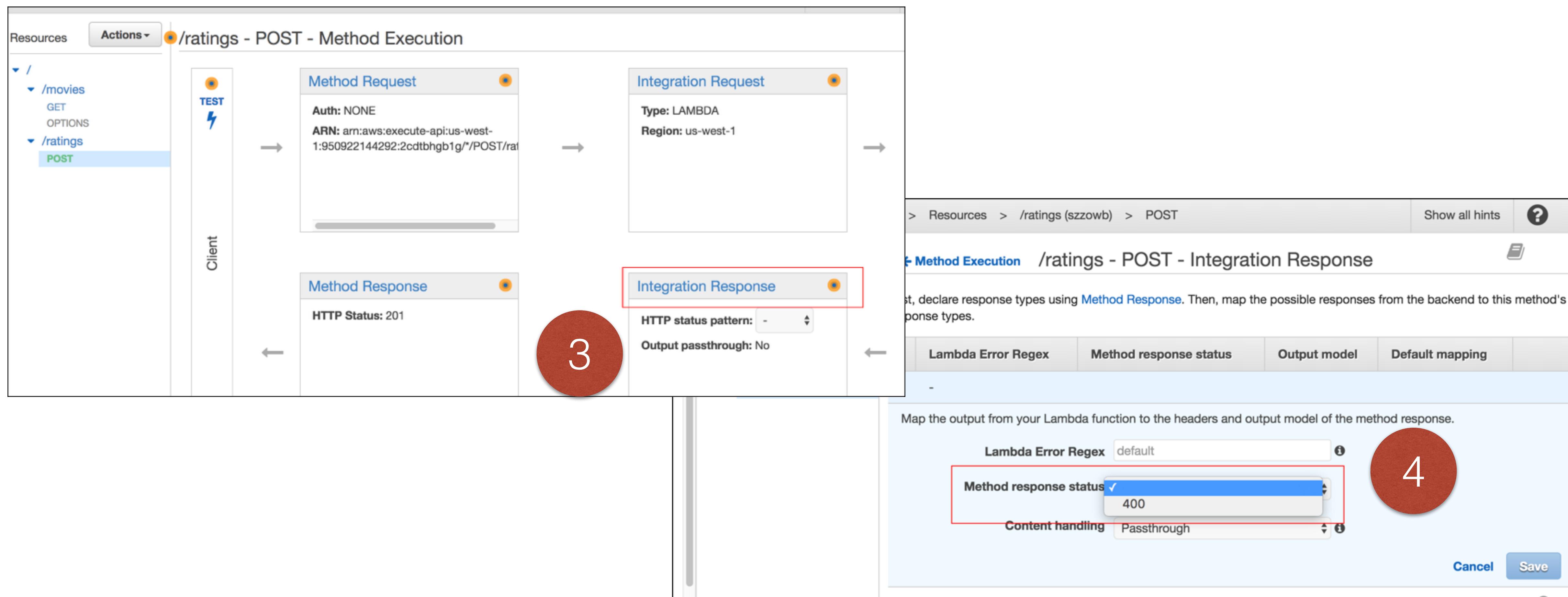
# Create POST Rating Endpoint

16

# Supporting HTTP Status Codes



# Supporting HTTP Status Codes



# Supporting HTTP Status Codes

The screenshot shows the AWS API Gateway console. The path is APIs > MovieRater02 (2cdtbhgb1g) > Resources > /ratings (szzowb) > POST. The left sidebar shows resources: /, /movies (GET, OPTIONS), and /ratings (POST). The main area is titled 'Method Execution /ratings - POST - Integration Response'. It says 'First, declare response types using Method Response. Then, map the possible responses from the backend to this method's response types.' A table has four columns: Lambda Error Regex, Method response status, Output model, and Default mapping. The Lambda Error Regex field contains '.\*Error.\*' and is highlighted with a red box. The Method response status dropdown is set to 400. The Content handling dropdown is set to Passthrough. At the bottom are 'Cancel' and 'Save' buttons, with 'Save' highlighted with a red box.

5

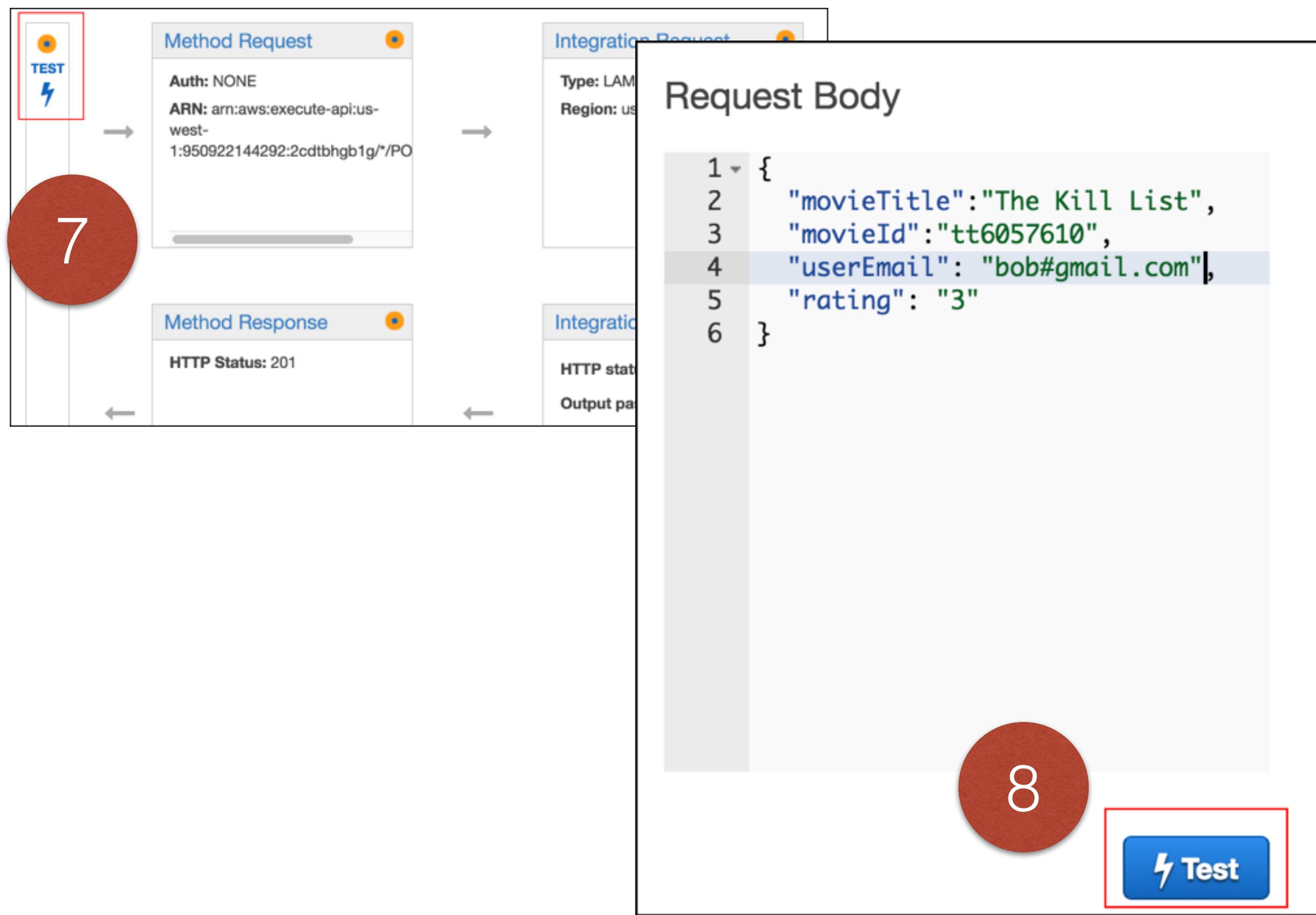
The screenshot shows the AWS API Gateway console with the same path and sidebar as the previous screenshot. The main area is titled 'Method Execution /ratings - POST - Integration Response'. It displays the configured integration response mapping:

Lambda Error Regex	Method response status	Output model	Default mapping
-	201		Yes
.*Error.*	400		No

At the bottom is a 'Add integration response' button.

6

# Supporting HTTP Status Codes



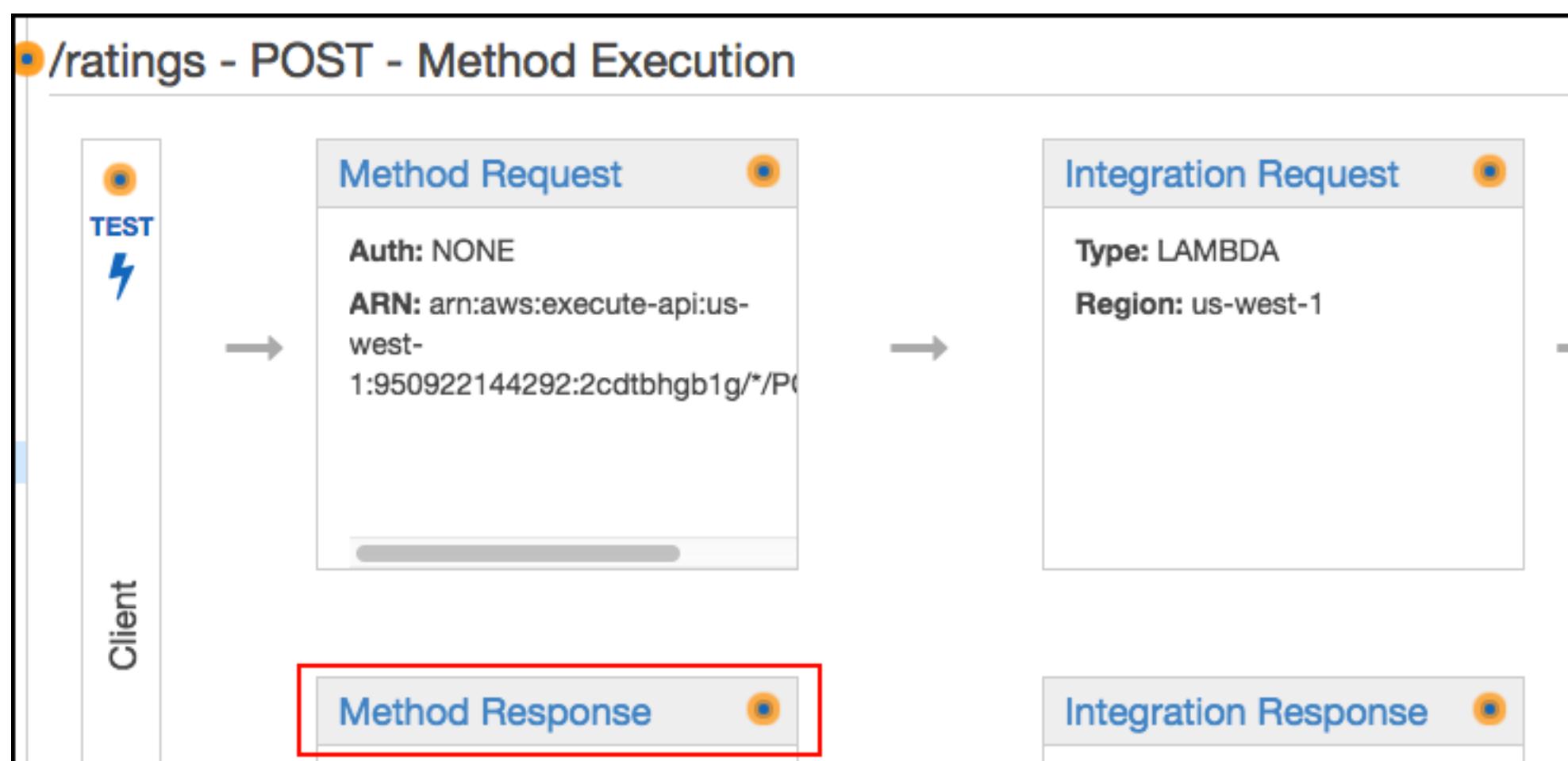
Request: /ratings  
Status: 400  
Latency: 249 ms

Response Body

```
{  
  "errorMessage": "{\"validationError\": [{\"invalidProperty\": \"userEmail\", \"value\": \"bob#gmail.com\"}]}",  
  "errorType": "Error",  
  "stackTrace": [  
    "exports.handler (/var/task/index.js:29:22)"  
  ]  
}
```

A red circle with the number 9 is overlaid on the top-right corner of the response body panel.

# Supporting HTTP Status Codes



# Supporting HTTP Status Codes

The screenshot shows two overlapping configuration panels for a POST method on the '/ratings' resource.

**Left Panel (Step 13):** Shows the 'Actions' tab selected. The 'HTTP Status' section lists 201 and 400. A red box highlights the '201' entry. The URL path on the left is: APIs > MovieRater02 (2cdtbhgb1g) > Resources > /ratings (szzowb) > POST.

**Right Panel (Step 14):** Shows the 'Actions' tab selected. The 'HTTP Status' section lists 201. A red box highlights the '201' entry. Below it, the 'Response Headers for 201' and 'Response Body for 201' sections are shown, both currently empty. The URL path on the left is: APIs > MovieRater02 (2cdtbhgb1g) > Resources > /movies > POST.

# Supporting HTTP Status Codes

The image consists of three vertically stacked screenshots from a REST API configuration tool, likely Swagger or a similar API management platform. Each screenshot shows a different step in the process of adding a response header to a POST method.

**Screenshot 15:** The user is on the 'Method Response' configuration page for the '/ratings - POST' method. The left sidebar shows resources for '/movies' and '/ratings'. Under '/ratings', the 'POST' method is selected. The main area displays the 'HTTP Status' section, which currently lists '201'. A red circle with the number '15' is overlaid on the bottom-left corner of this screen.

**Screenshot 16:** The user has opened the 'Response Headers for 201' configuration panel. It shows a table with a single row labeled 'Name' and a value of 'No header'. Below the table is a button labeled '+ Add Header'. A red circle with the number '16' is overlaid on the bottom-left corner of this screen. A red box highlights the '+ Add Header' button.

**Screenshot 17:** The user has added a new header named 'Control-Allow-Headers'. This header is listed in the 'Response Headers for 201' table. The value is set to 'Access-Control-Allow-Headers'. A red circle with the number '17' is overlaid on the bottom-left corner of this screen. A red box highlights the newly added header row.

# Supporting HTTP Status Codes

Screenshot of an API management tool showing the configuration for the POST method of the /ratings resource. The left sidebar shows resources for /movies and /ratings. The right panel shows the configuration for the POST method of /ratings.

**HTTP Status**

201
-----

**Response Headers for 201**

Name
Access-Control-Allow-Headers
Access-Control-Allow-Methods
Access-Control-Allow-Origin

A red box highlights the "Access-Control-Allow-Origin" header entry. A red circle with the number 18 is overlaid on the bottom-left corner of the screenshot.

Screenshot of the same API management tool after changes have been made. The configuration for the POST method of /ratings now includes the "Access-Control-Allow-Origin" header.

**HTTP Status**

201
-----

**Response Headers for 201**

Name
Access-Control-Allow-Headers
Access-Control-Allow-Methods
Access-Control-Allow-Origin

A red box highlights the "Access-Control-Allow-Origin" header entry. A red circle with the number 19 is overlaid on the bottom-left corner of the screenshot.

# Supporting HTTP Status Codes



The screenshot shows the AWS API Gateway Method Execution interface for a POST request to '/ratings'. The 'Actions' tab is selected. In the 'Resources' tree, the 'POST' method under the '/ratings' resource is selected. A red box highlights the 'Add integration response' button. A large red circle with the number '21' is overlaid on the bottom right of the screen.

20

The screenshot shows the AWS API Gateway Method Execution interface for a POST request to '/ratings'. The 'Actions' tab is selected. In the 'Resources' tree, the 'POST' method under the '/ratings' resource is selected. The 'Lambda Error Regex' field is set to 'default'. The 'Method response status' dropdown is set to '201'. The 'Content handling' dropdown is set to 'Passthrough'. A red box highlights the 'Method response status' dropdown. A large red circle with the number '22' is overlaid on the bottom right of the screen.

@reselbob

# Supporting HTTP Status Codes

The screenshot shows the AWS API Gateway Method Execution interface for a POST request to the '/ratings' endpoint. The left sidebar lists resources: '/', '/movies', and '/ratings'. The '/ratings' section is expanded, showing methods: GET, OPTIONS, and POST. The POST method is selected and highlighted with a blue background. The main area displays a table for mapping Lambda error regexes to method response statuses:

Lambda Error Regex	Method response status	Output model	Default mapping
-	201		Yes
.*Error.*	400		No

A red box highlights the first row's 'Lambda Error Regex' column.

23

The screenshot shows the AWS API Gateway Method Execution interface for a POST request to the '/ratings' endpoint. The left sidebar lists resources: '/', '/movies', and '/ratings'. The '/ratings' section is expanded, showing methods: GET, OPTIONS, and POST. The POST method is selected and highlighted with a blue background. The main area displays a table for mapping Lambda error regexes to method response statuses:

Lambda Error Regex	Method response status	Output model	Default mapping
-	201		Yes

Below the table, instructions say: "Map the output from your Lambda function to the headers and output model of the 201 method response." A red box highlights the 'Header Mappings' button at the bottom of the interface.

24

# Supporting HTTP Status Codes

Resources Actions ▾

First, declare response types using [Method Response](#). Then, map the possible responses from the backend to this method's response types.

Lambda Error Regex	Method response status	Output model	Default mapping
-	201	Yes	

Map the output from your Lambda function to the headers and output model of the 201 method response.

Lambda Error Regex default

Content handling Passthrough

Cancel Save

Header Mappings

Response header	Mapping value ⓘ
Access-Control-Allow-Headers	<input type="text"/> <input type="button" value=""/>
Access-Control-Allow-Methods	<input type="text"/> <input type="button" value=""/>
Access-Control-Allow-Origin	<input type="text"/> <input type="button" value=""/>

25

'Content-Type,X-Amz-Date,Authorization,X-Api-Key,X-Amz-Security-Token'

, declare response types using [Method Response](#). Then, map the possible responses from the backend to this method's response types.

Lambda Error Regex	Method response status	Output model	Default mapping
-	201	Yes	<input type="checkbox"/>

Map the output from your Lambda function to the headers and output model of the 201 method response.

Lambda Error Regex default

Content handling Passthrough

Cancel Save

Header Mappings

Response header	Mapping value ⓘ
Access-Control-Allow-Headers	<input type="text" value="X-Amz-Security-Token"/> <input type="button" value=""/>

26

# Supporting HTTP Status Codes

The screenshot shows the AWS API Gateway configuration interface for a POST method on the /ratings resource of the MovieRater02 API. The left sidebar shows the API structure: APIs > MovieRater02 (2cdtbhgb1g) > Resources > /ratings (szzowb) > POST.

The main panel displays the configuration for handling Lambda errors. It includes a table for mapping Lambda Error Regex to Method response status, Output model, and Default mapping. A note indicates that users should declare response types using Method Response and map possible responses from the backend to this method's response types.

Below this, there is a section for mapping the output from the Lambda function to headers and an output model. The Lambda Error Regex is set to 'default', Content handling is set to 'Passthrough', and the Method response status is '201'. A note says to map the output from the Lambda function to the headers and output model of the 201 method response.

At the bottom, there is a 'Header Mappings' section where response headers are mapped to specific values. The 'Access-Control-Allow-Headers' header is mapped to the value 'Content-Type,X-Amz-Date,Authorization,X-Api-Key,X-Amz-Security-Token'. The 'Access-Control-Allow-Methods' and 'Access-Control-Allow-Origin' headers are also listed but have no visible mappings.

A red circle with the number 27 is overlaid on the bottom-left corner of the screenshot.

On the right side of the screenshot, there is a preview or continuation of the configuration, showing the same tables and sections. A red circle with the number 28 is overlaid on the bottom-right corner of the screenshot.

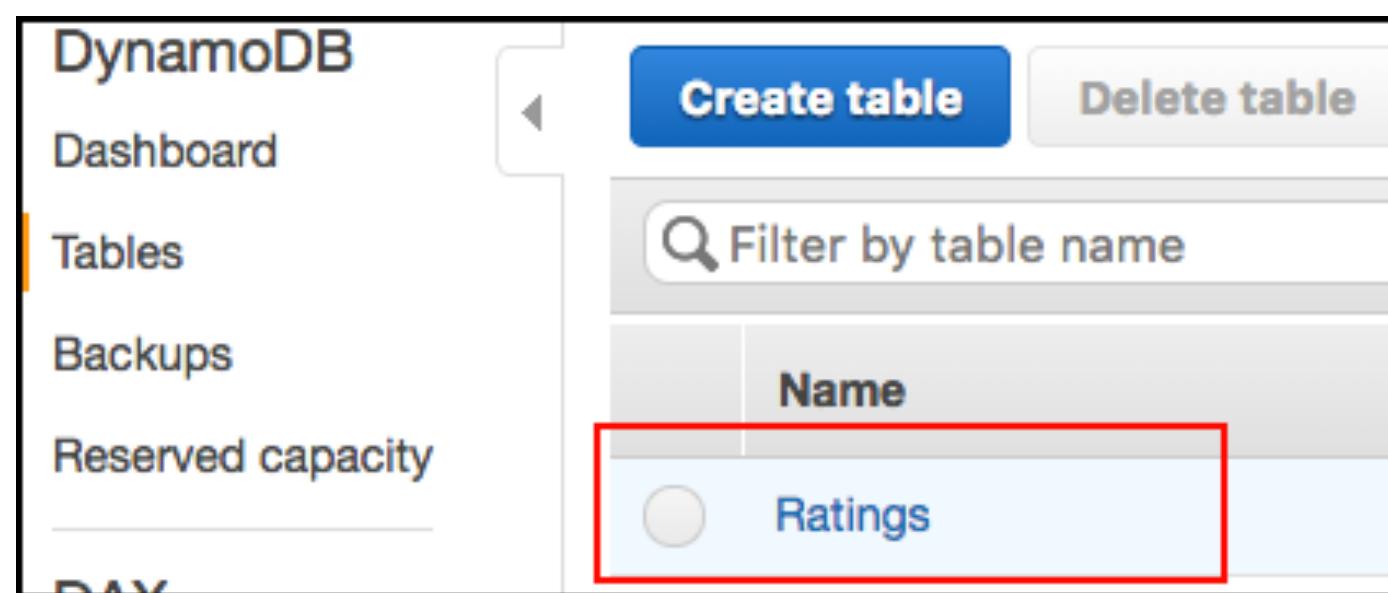
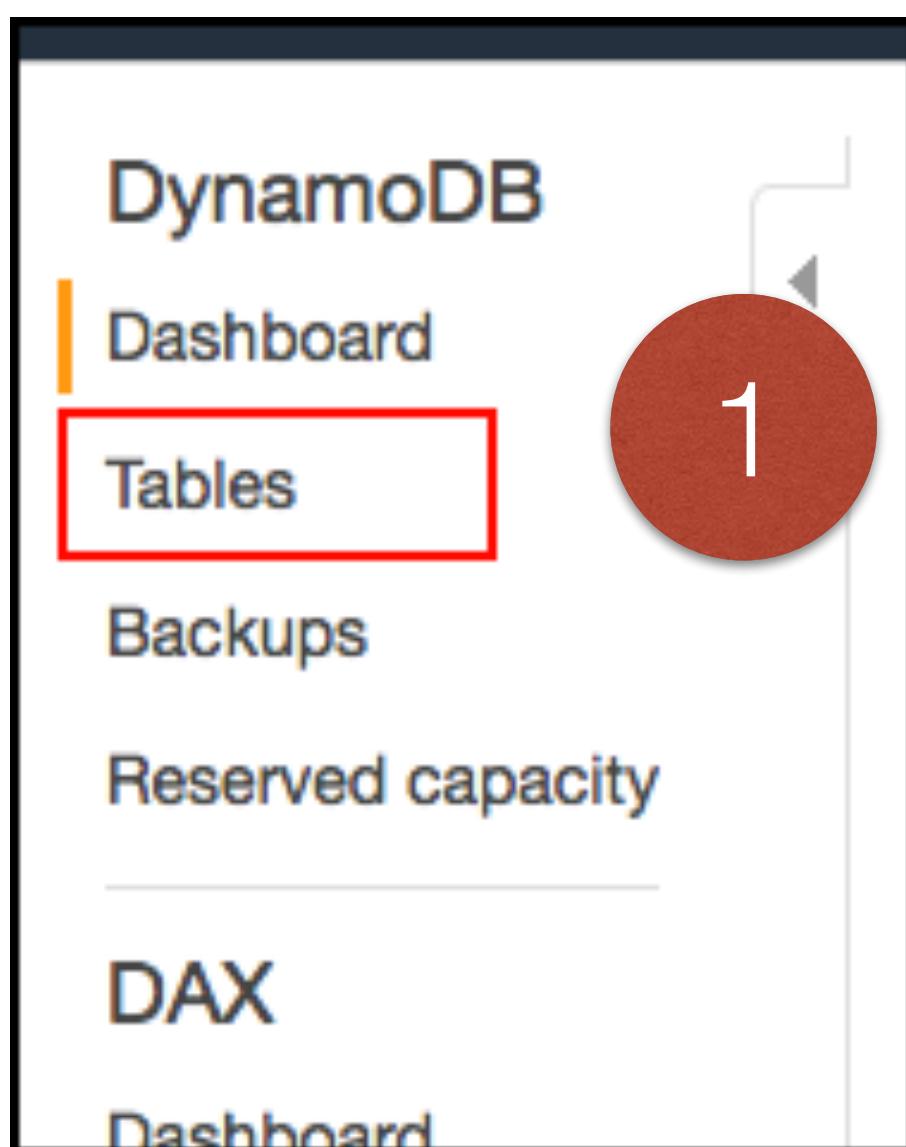
# Supporting HTTP Status Codes

The screenshot shows the AWS API Gateway interface for configuring a POST method response. The path is APIs > MovieRater02 (2cdtbhgb1g) > Resources > /ratings (szzowb) > POST. The 'Actions' dropdown is open, and the 'Header Mappings' section is highlighted with a red box. It shows three mappings for CORS headers:

Response header	Mapping value
Access-Control-Allow-Headers	'Content-Type,X-Amz-Date,Authorization,X-Api-Key,X-Amz-Security-Token'
Access-Control-Allow-Methods	**
Access-Control-Allow-Origin	**

A red circle containing the number 29 is overlaid on the left side of the screenshot.

# Create DynamoDB Index



The screenshot shows the 'Create DynamoDB table' configuration page. A red circle labeled '3' highlights the 'Primary key\*' section, specifically the 'ratingId.' input field, which is defined as a String type. A red circle labeled '4' highlights the 'Create' button at the bottom right of the form.

**Create DynamoDB table**

DynamoDB is a schema-less database that only requires a table name and primary key. The table's primary key is made up of one or two attributes that uniquely identify items, partition the data, and sort data within each partition.

Table name\* Ratings

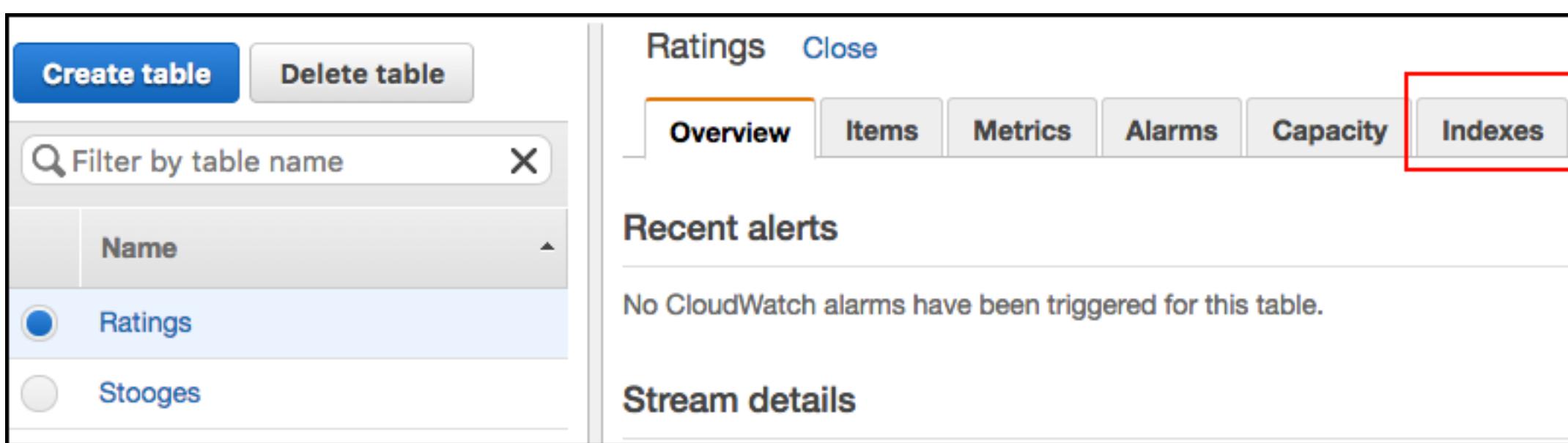
Primary key\* Partition key ratingId. String

Add sort key

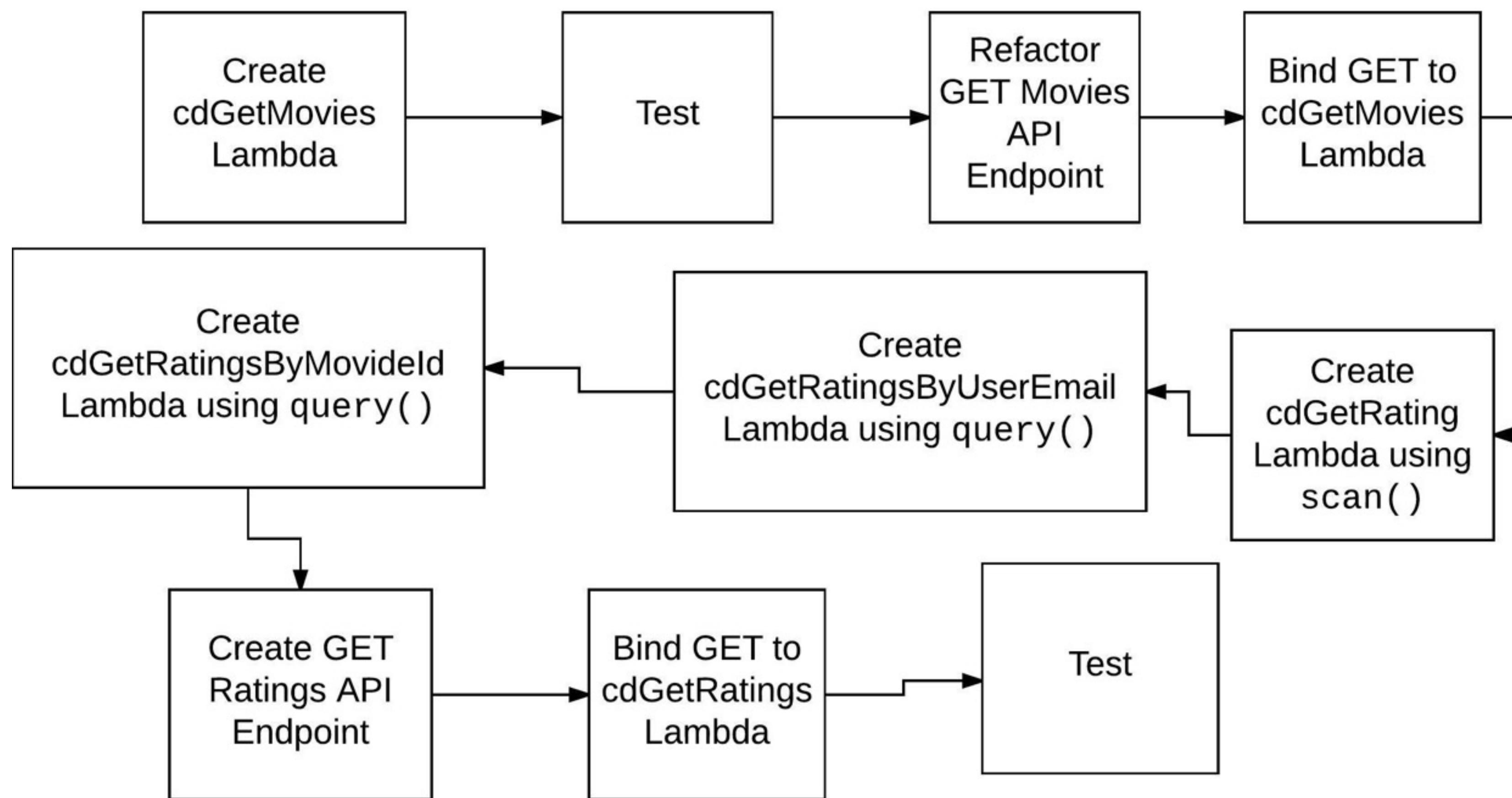
Use default settings

- No secondary indexes.
- Provisioned capacity set to 5 reads and 5 writes.
- Basic alarms with 80% upper threshold using SNS topic "dynamodb".
- On-Demand Backup and Restore Enabled NEW

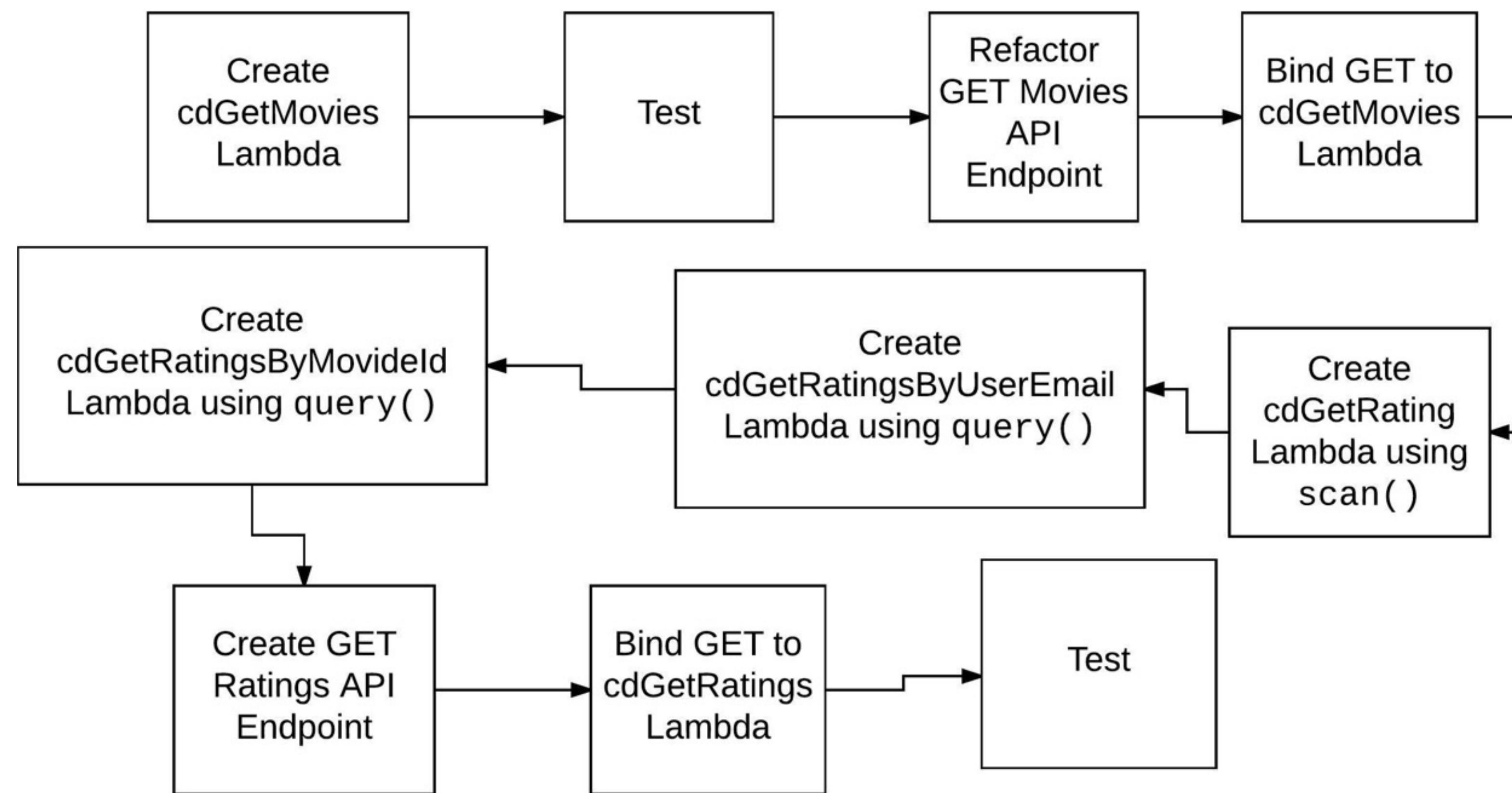
Cancel Create



# Create DynamoDB Index



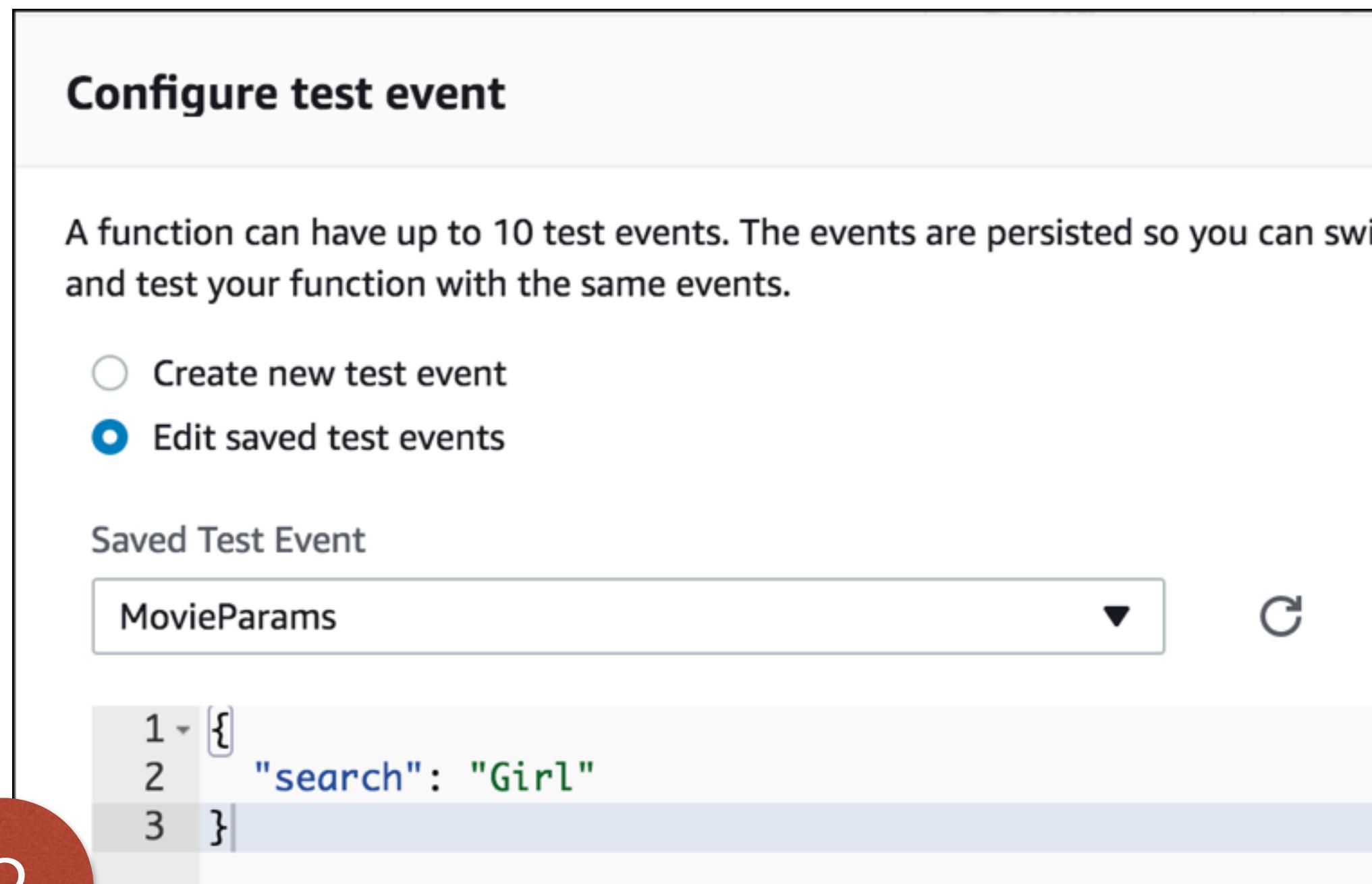
# Session 5



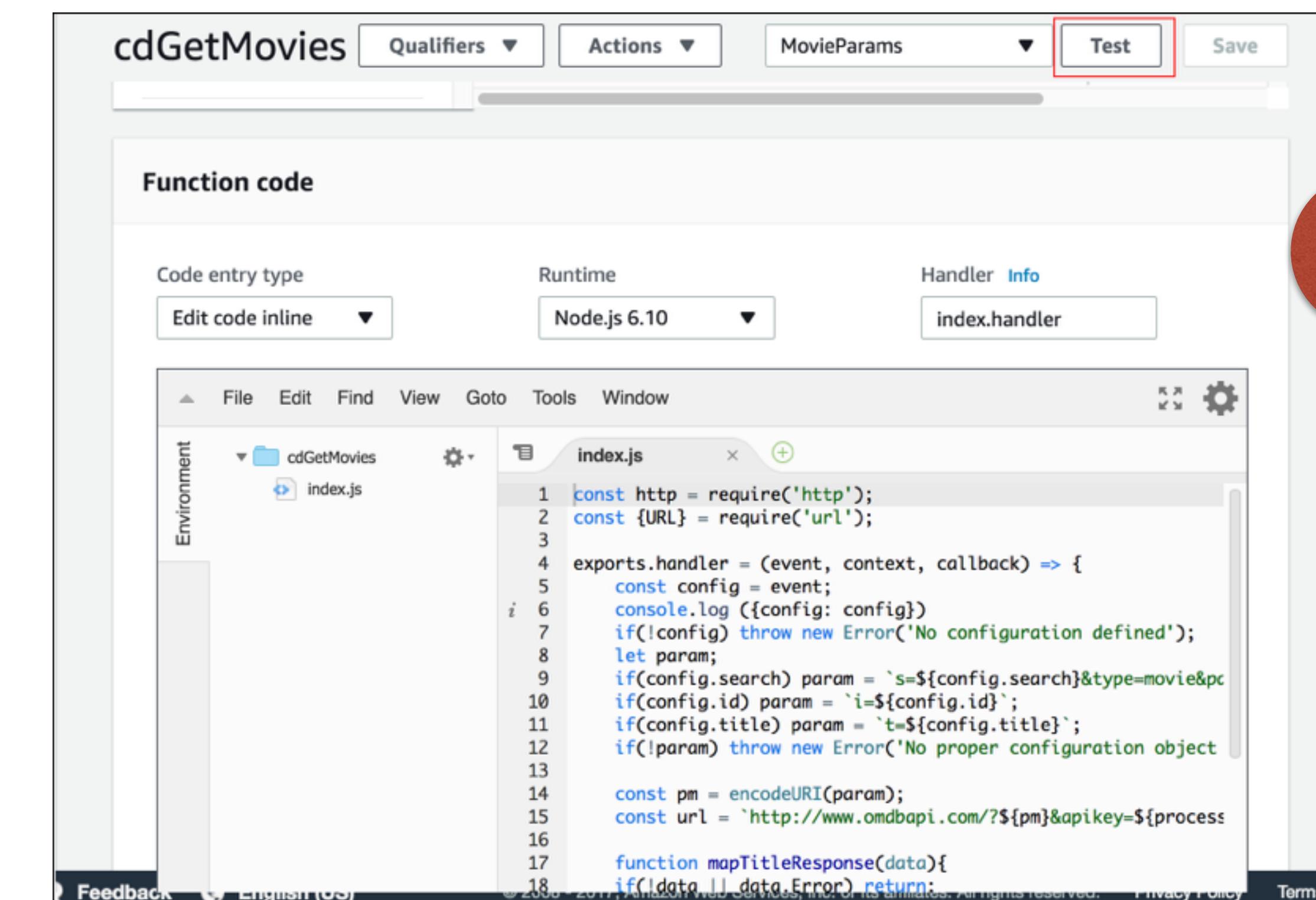
# Create cdGetMovies Lambda

1

<https://github.com/reselbob/CDAwsClass/blob/Session-5/getMovies.js>



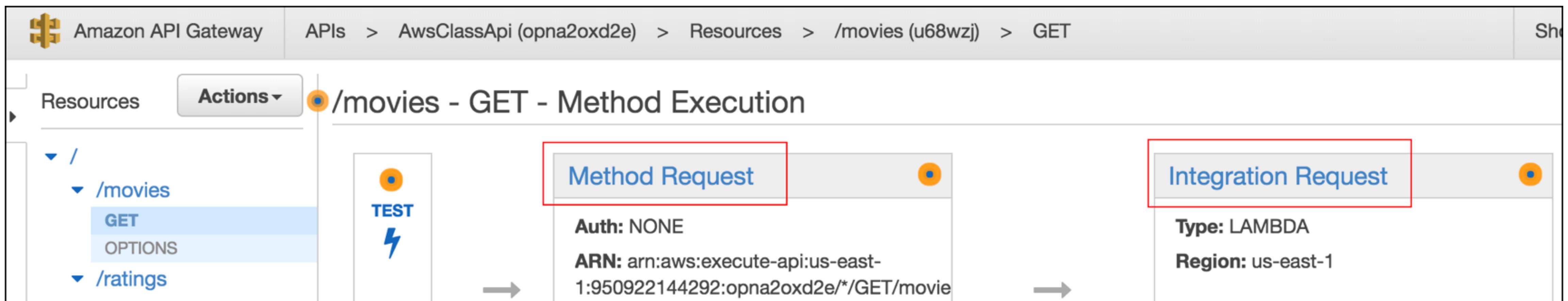
2



3

# Refactor GET Movies Endpoint

4



# Refactor GET Movies Endpoint

Method Execution /movies - GET - Method Request

Provide information about this method's authorization settings and the parameters it can receive.

Settings

Authorization: NONE  

Request Validator: NONE  

API Key Required: false 

URL Query String Parameters 

Name	Required	Caching
id	<input type="checkbox"/>	<input type="checkbox"/>
search	<input type="checkbox"/>	<input type="checkbox"/>
title	<input type="checkbox"/>	<input type="checkbox"/>

+ Add query string

5

Method Execution /movies - GET - Integration Request

Provide information about the target backend that this method will call and whether the incoming request

Integration type:  Lambda Function   
 HTTP   
 Mock   
 AWS Service   
 VPC Link 

Use Lambda Proxy integration  

Lambda Region: us-east-1 

Lambda Function: cdGetMovies 

6

# Refactor GET Movies Endpoint

Body Mapping Templates

**Request body passthrough**

- When no template matches the request Content-Type header i
- When there are no templates defined (recommended) i
- Never i

Content-Type

application/json

+ Add mapping template

application/json

Generate template:

```
1 {  
2   "title" : "$input.params('title')",  
3   "id" : "$input.params('id')",  
4   "search" : "$input.params('search')"  
5 }
```

7

/movies - GE

TEST

8

Method Execution /movies - GET - Method

Make a test call to your method with the provided input

Path

No path parameters exist for this resource. You can define path parameters by using the syntax {myPathParam} in a resource path.

Query Strings

search

Value

title

Sounder

id

Value

Headers

No header parameters exist for this method. You can add them via Method Request.

Stage Variables

No stage variables exist for this method.

Request Body

Request Body is not supported for GET methods.

Test

9

# Refactor GET Movies Endpoint

10

Request: /movies?title=Sounder

Status: 200

Latency: 105 ms

Response Body

```
{  
  "genre": "",  
  "id": "tt0069303",  
  "title": "Sounder",  
  "releaseDate": "10 May 1973",  
  "studio": "Rainbow Group / KOCH Entertainment"  
}
```

Response Headers

```
{"X-Amzn-Trace-Id":"sampled=0;root=1-5a2cafcb-95e58a3f7d7c142c  
b851ba64", "Content-Type": "application/json"}
```

# Created getRatings Lambda

1

[http://docs.aws.amazon.com/AWSJavaScriptSDK/latest/AWS/  
DynamoDB.html#query-property](http://docs.aws.amazon.com/AWSJavaScriptSDK/latest/AWS/DynamoDB.html#query-property)

2

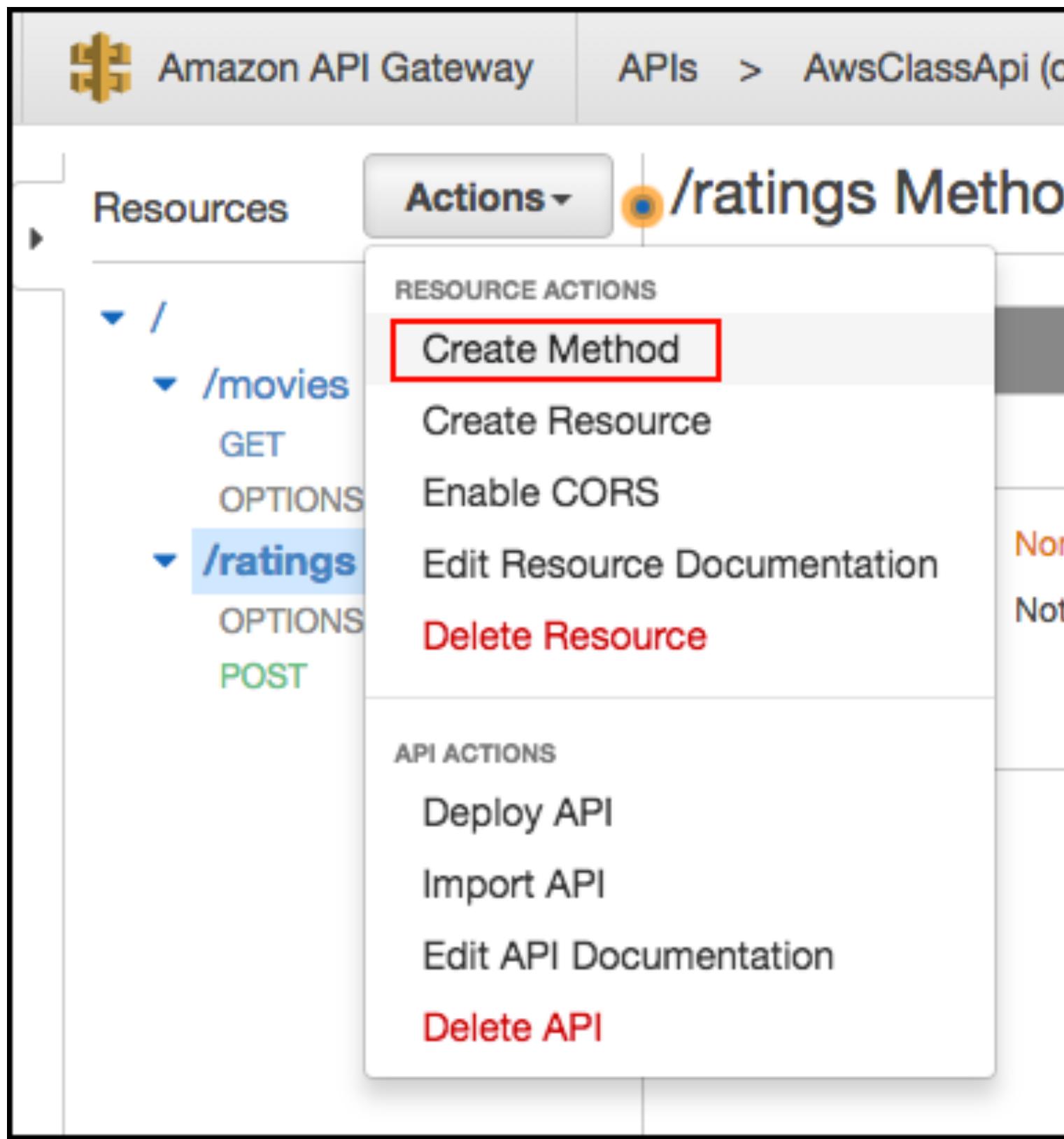
<https://github.com/reselbob/CDAwsClass/blob/Session-5/getRatingsByMovieId.js>

3

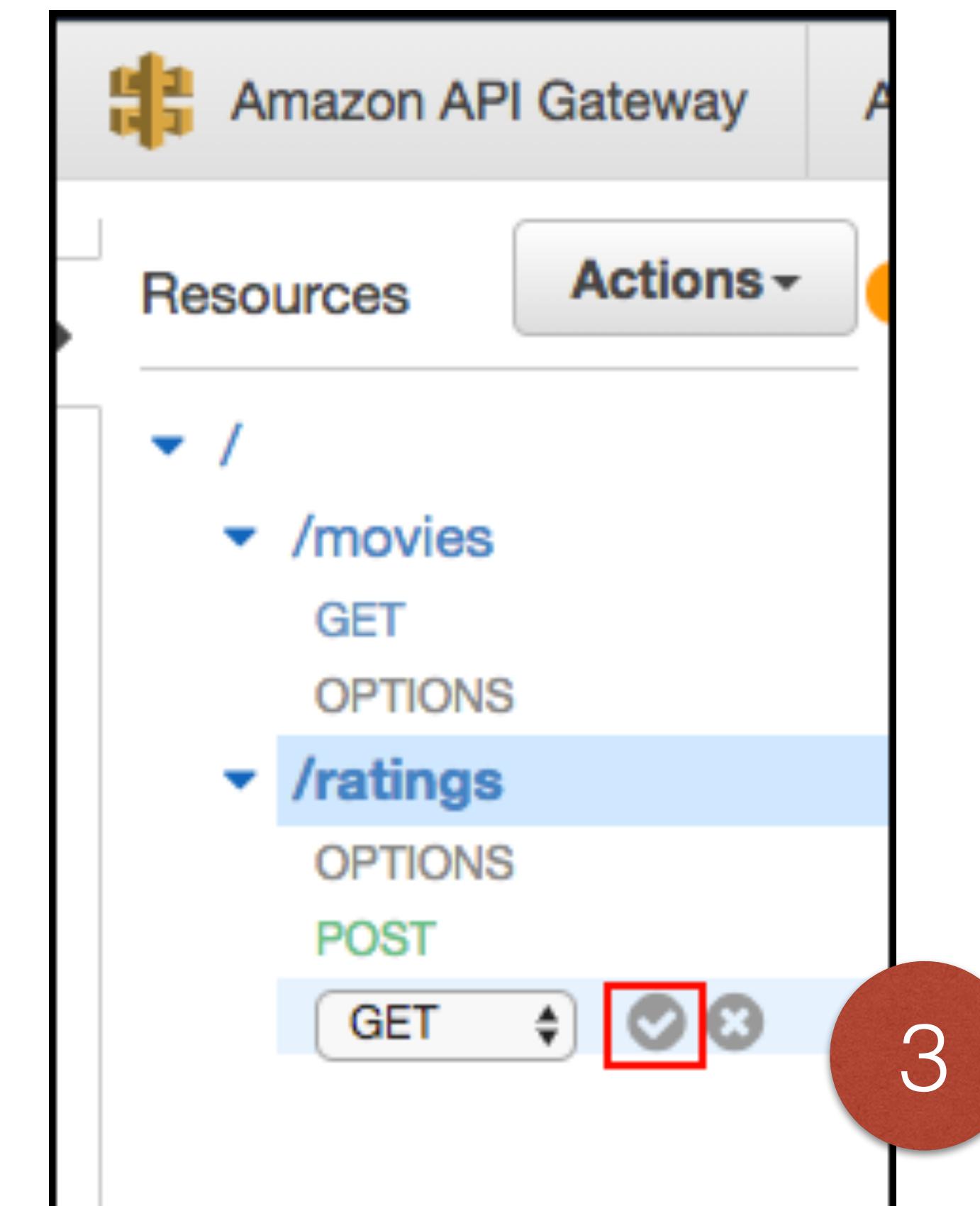
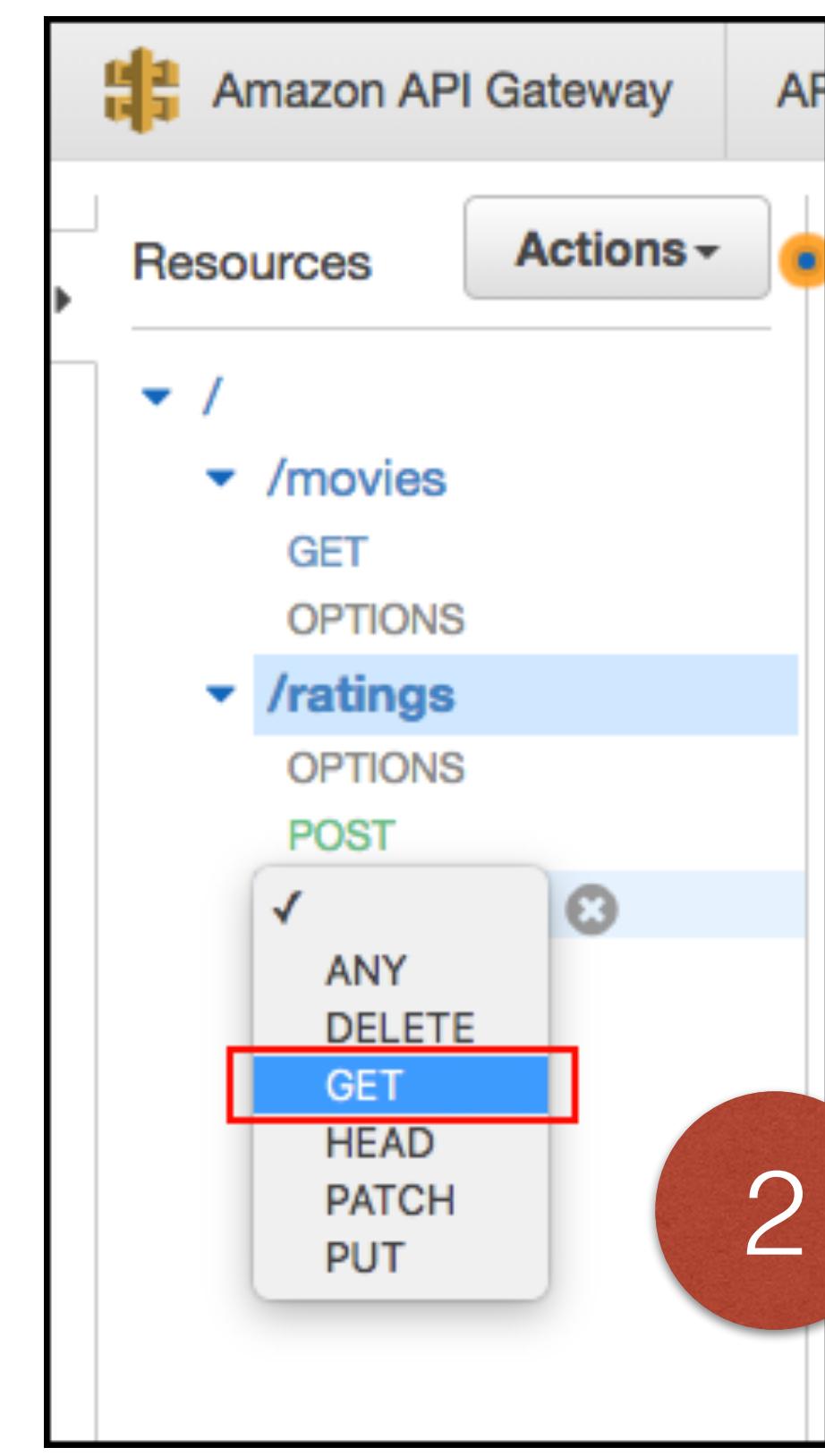
<https://github.com/reselbob/CDAwsClass/blob/Session-5/getRatingsByEmailUser.js>

# Create the GET Ratings Endpoint

1



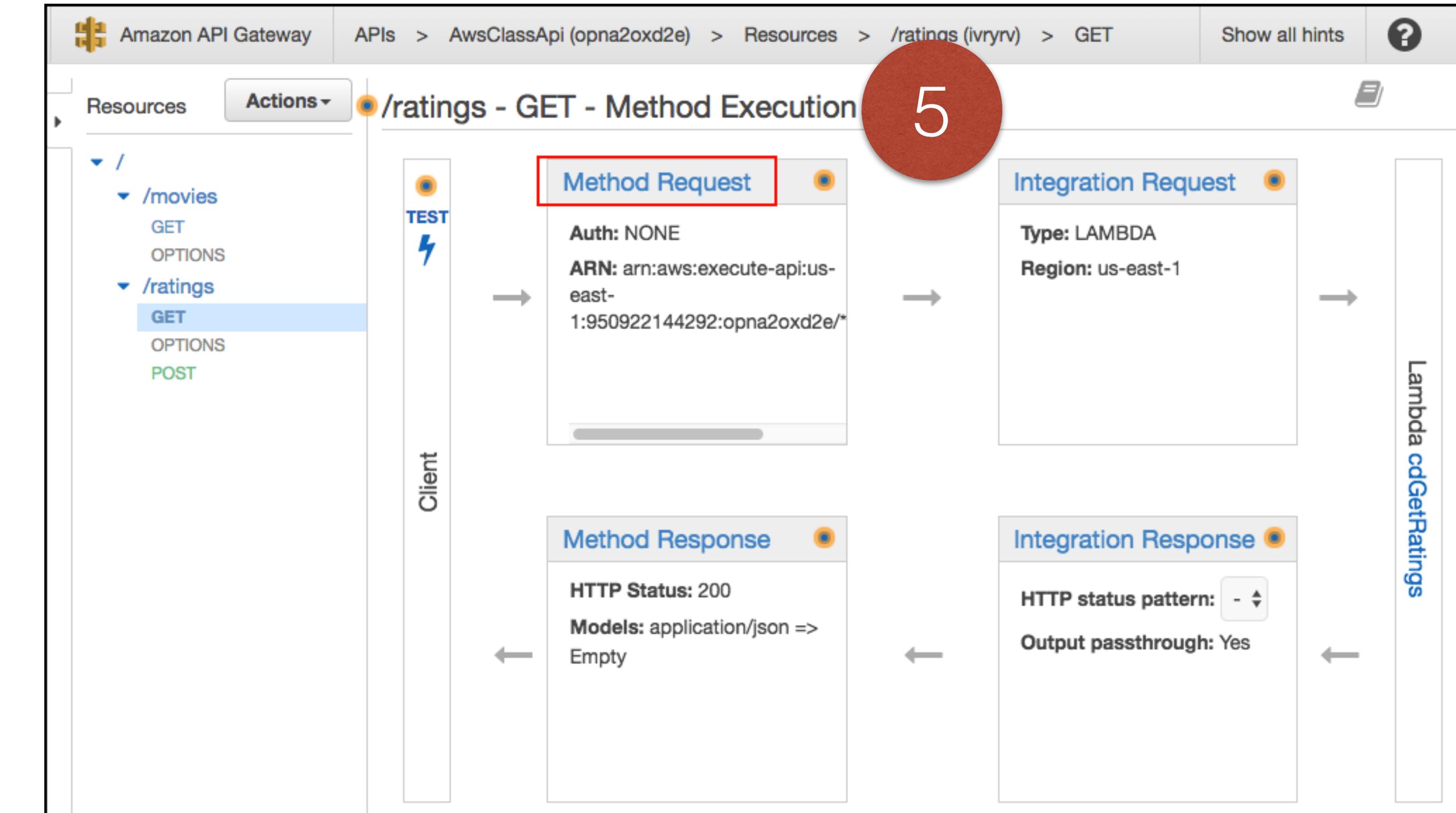
2



3

# Create the GET Ratings Endpoint

The screenshot shows the 'Actions' tab selected in the top navigation bar. Under the 'Resources' section, the path 'AwsClassApi / /ratings (ivryrv) / /ratings - GET - Setup' is shown. On the left, a tree view lists resources: '/movies' (GET, OPTIONS), '/ratings' (selected, GET, OPTIONS, POST). In the main area, the 'Integration type' dropdown is set to 'Lambda Function' (radio button selected). Below it, 'Use Lambda Proxy integration' is unchecked. The 'Lambda Region' dropdown is set to 'us-east-1'. The 'Lambda Function' dropdown contains several options, with 'cdGetRatings' highlighted and surrounded by a red box. A red circle with the number '4' is overlaid at the bottom left.



# Create the GET Ratings Endpoint

The screenshot shows the Amazon API Gateway interface. The path is APIs > AwsClassApi (opna20xd2e) > Resources > /rating. A red circle labeled '6' highlights the 'Actions' dropdown menu. The 'GET' method for the '/ratings' resource is selected. A red box highlights the 'URL Query String Parameters' section, which contains a table with one row for 'movield'. A red button labeled '+ Add query string' is at the bottom.

Amazon API Gateway

APIs > AwsClassApi (opna20xd2e) > Resources > /rating

Resources Actions ▾

Method Execution /ratings - GET - Method

Provide information about this method's authorization settings and the parameters it can receive.

Settings

Authorization NONE

Request Validator NONE

API Key Required false

URL Query String Parameters

Name	Required
movield	

+ Add query string

The screenshot shows the Amazon API Gateway interface. The path is APIs > AwsClassApi (opna20xd2e) > Resources > /ratings (ivryrv) > GET. A red circle labeled '7' highlights the 'Actions' dropdown menu. The 'GET' method for the '/ratings' resource is selected. A red box highlights the 'URL Query String Parameters' section, which contains a table with one row for 'movield'. The 'Required' column has a checked checkbox. A red button labeled '+ Add query string' is at the bottom.

Amazon API Gateway

APIs > AwsClassApi (opna20xd2e) > Resources > /ratings (ivryrv) > GET

Resources Actions ▾

Method Execution /ratings - GET - Method Request

Provide information about this method's authorization settings and the parameters it can receive.

Settings

Authorization NONE

Request Validator NONE

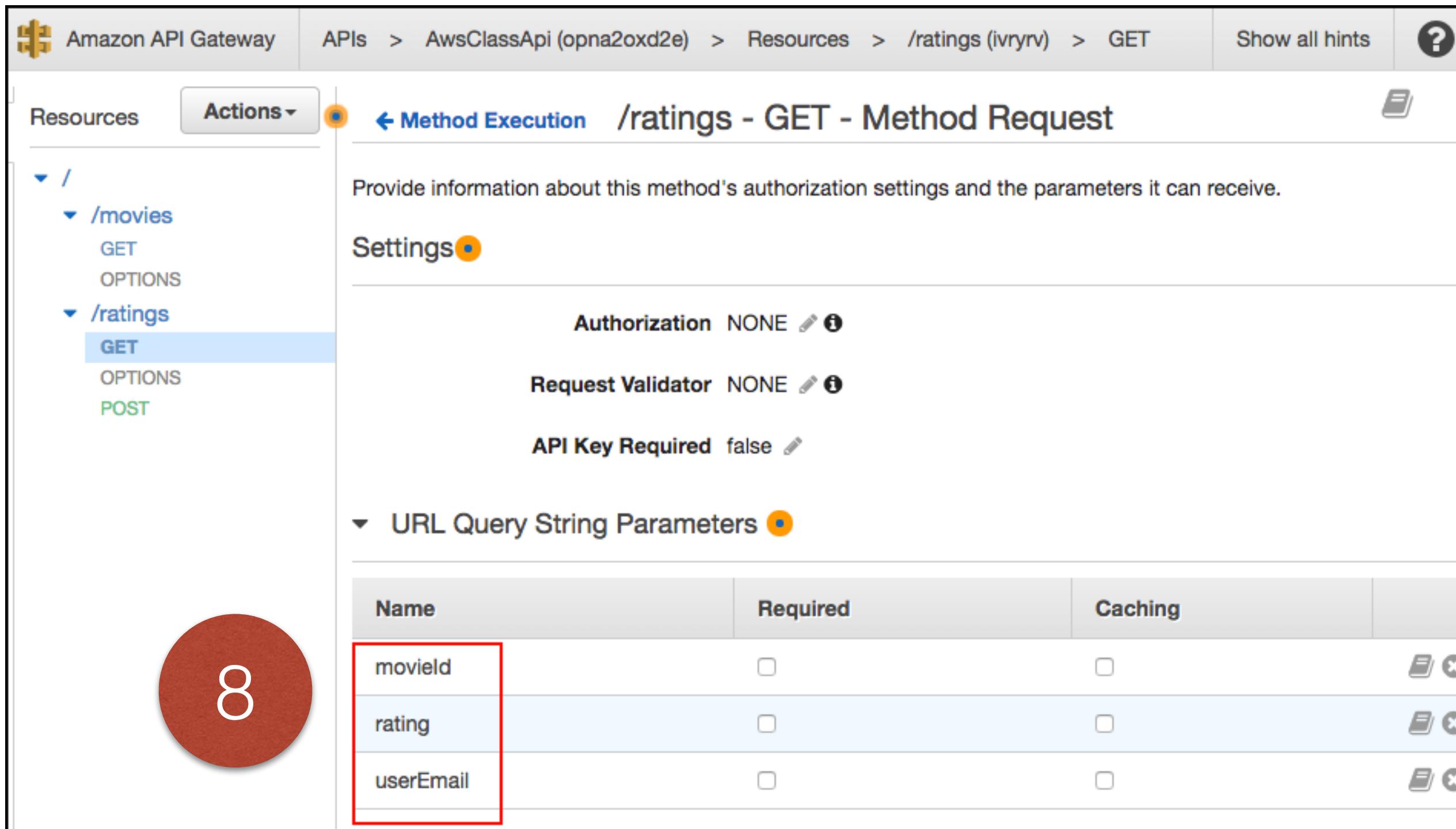
API Key Required false

URL Query String Parameters

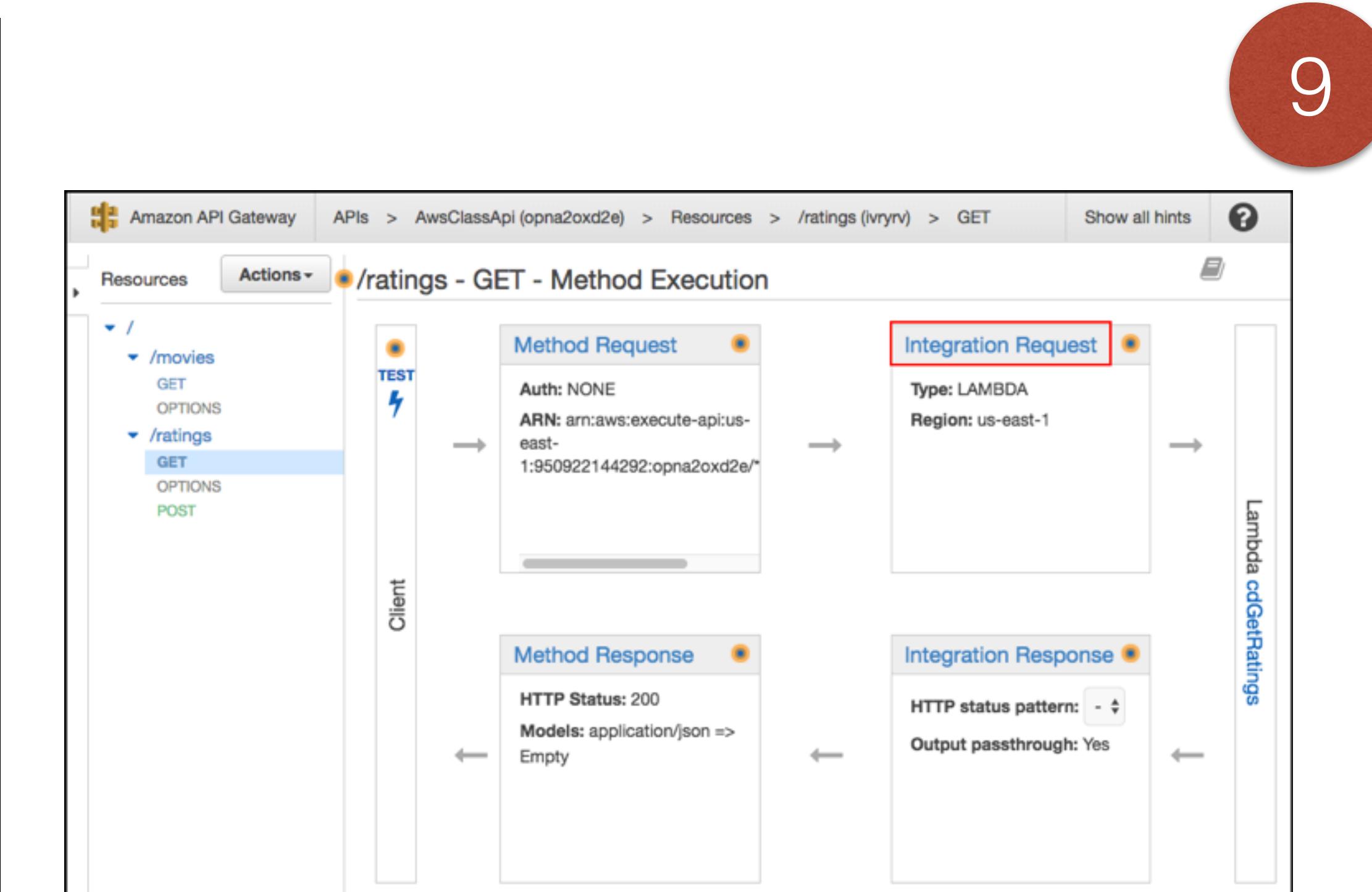
Name	Required	Caching
movield	<input checked="" type="checkbox"/>	<input type="checkbox"/>

+ Add query string

# Create the GET Ratings Endpoint



The screenshot shows the Amazon API Gateway interface for creating a GET method request. The URL path is /ratings. The method settings show 'Authorization: NONE', 'Request Validator: NONE', and 'API Key Required: false'. Under 'URL Query String Parameters', three parameters are listed: 'movield' (Required: no), 'rating' (Required: no), and 'userEmail' (Required: no). A red circle with the number '8' is overlaid on the bottom-left corner of the screenshot.



# Create the GET Ratings Endpoint

The screenshot shows the configuration for the GET /ratings endpoint. The 'Body Mapping Templates' section is highlighted with a red box and contains a dropdown menu for 'Content-Type' set to 'application/json'. A red circle with the number 10 is overlaid on the bottom right of the interface.

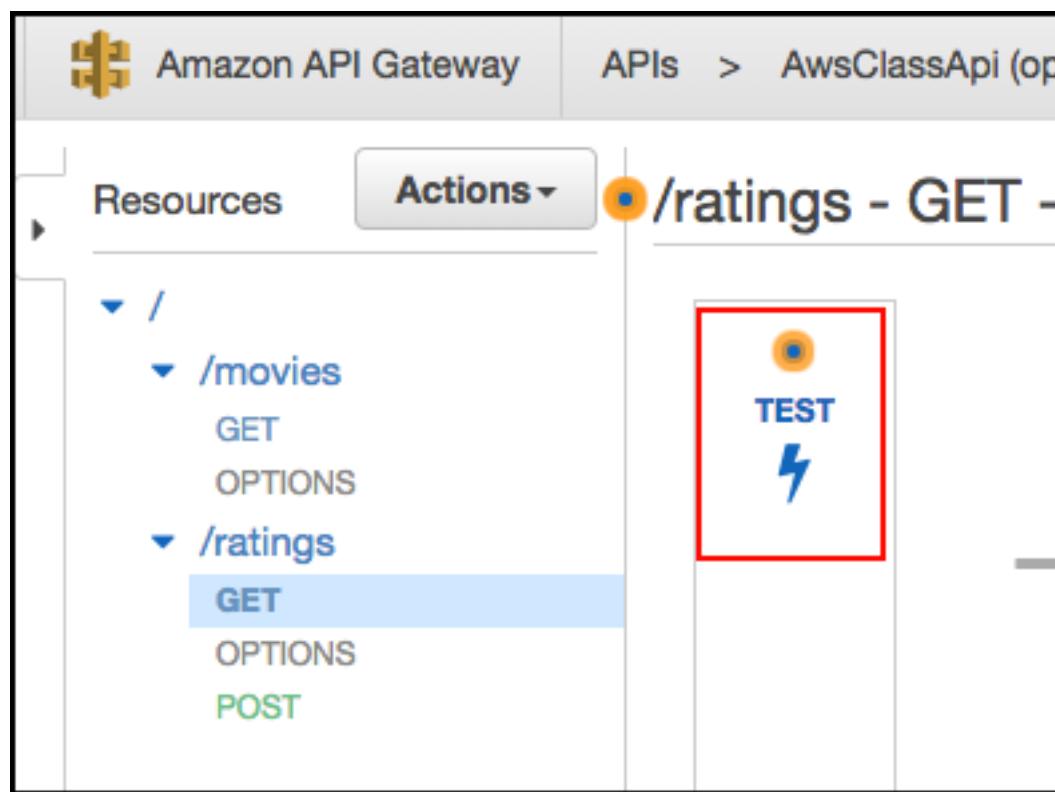
The screenshot shows the configuration for the GET /ratings endpoint. A mapping template is defined for 'application/json' content-type, containing the following JSON path template:

```
1 {  
2   "movieId" : "$input.params('movieId')",  
3   "userEmail" : "$input.params('userEmail')",  
4   "rating" : "$input.params('rating')"  
5 }
```

A red box highlights the 'application/json' Content-Type selection. A red arrow points from the 'application/json' selection in step 10 to this Content-Type field. A red circle with the number 11 is overlaid on the bottom right of the interface.

# Create the GET Ratings Endpoint

12



The screenshot shows the 'Method Execution /ratings - GET - Method Test' page. The left sidebar lists the resources and methods: '/' (with '/movies' and 'GET'), '/ratings' (with 'GET', 'OPTIONS', and 'POST'). The 'GET' method under '/ratings' is selected. The main area contains fields for testing: 'Path', 'Query Strings', 'Headers', 'Stage Variables', and 'Request Body'. The 'userEmail' field has the value 'tom@gmail.com' entered, and this field is highlighted with a red box. At the bottom right, there is a 'Test' button with a lightning bolt icon, which is also highlighted with a red box.

13

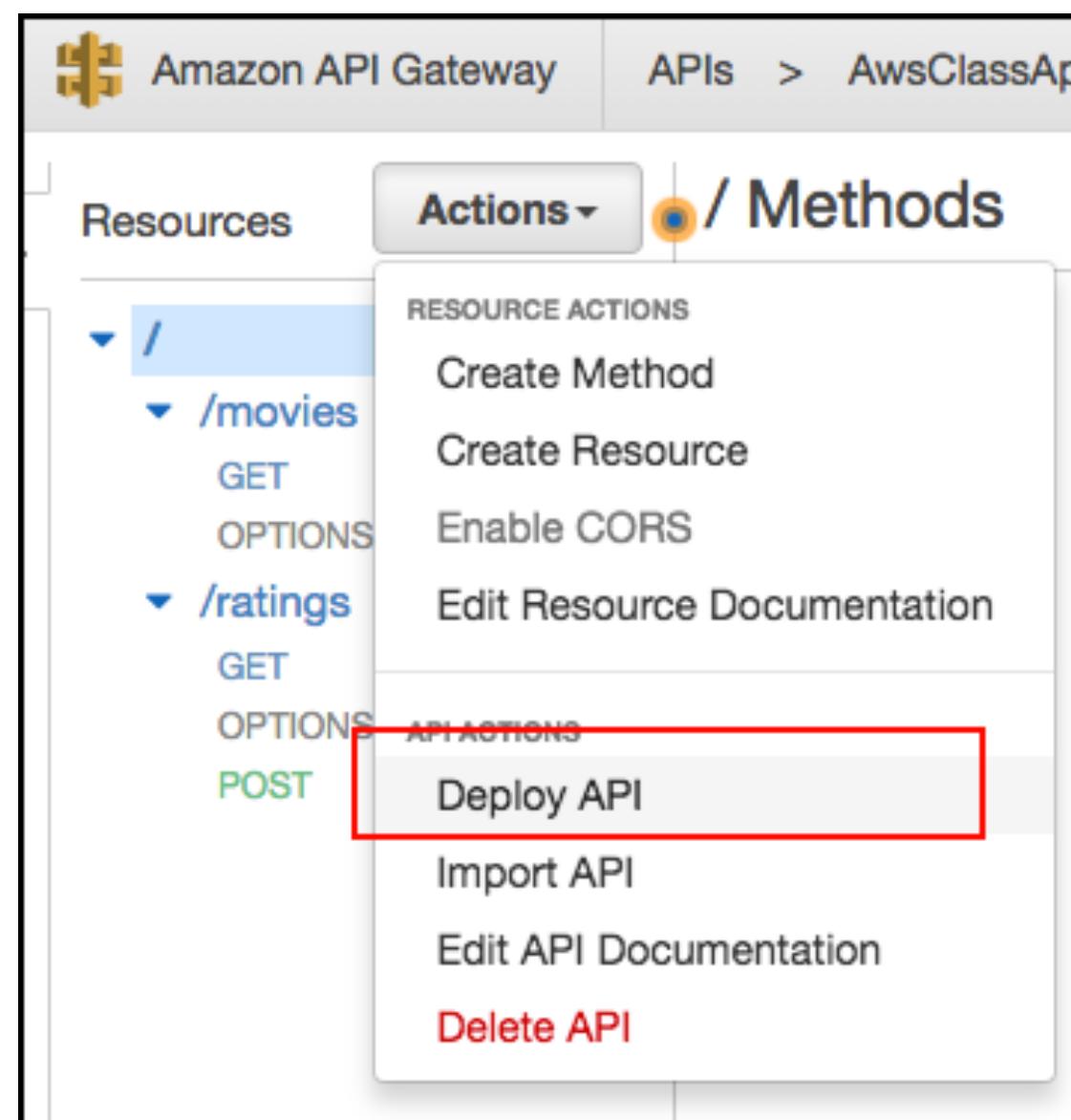
# Create the GET Ratings Endpoint

14

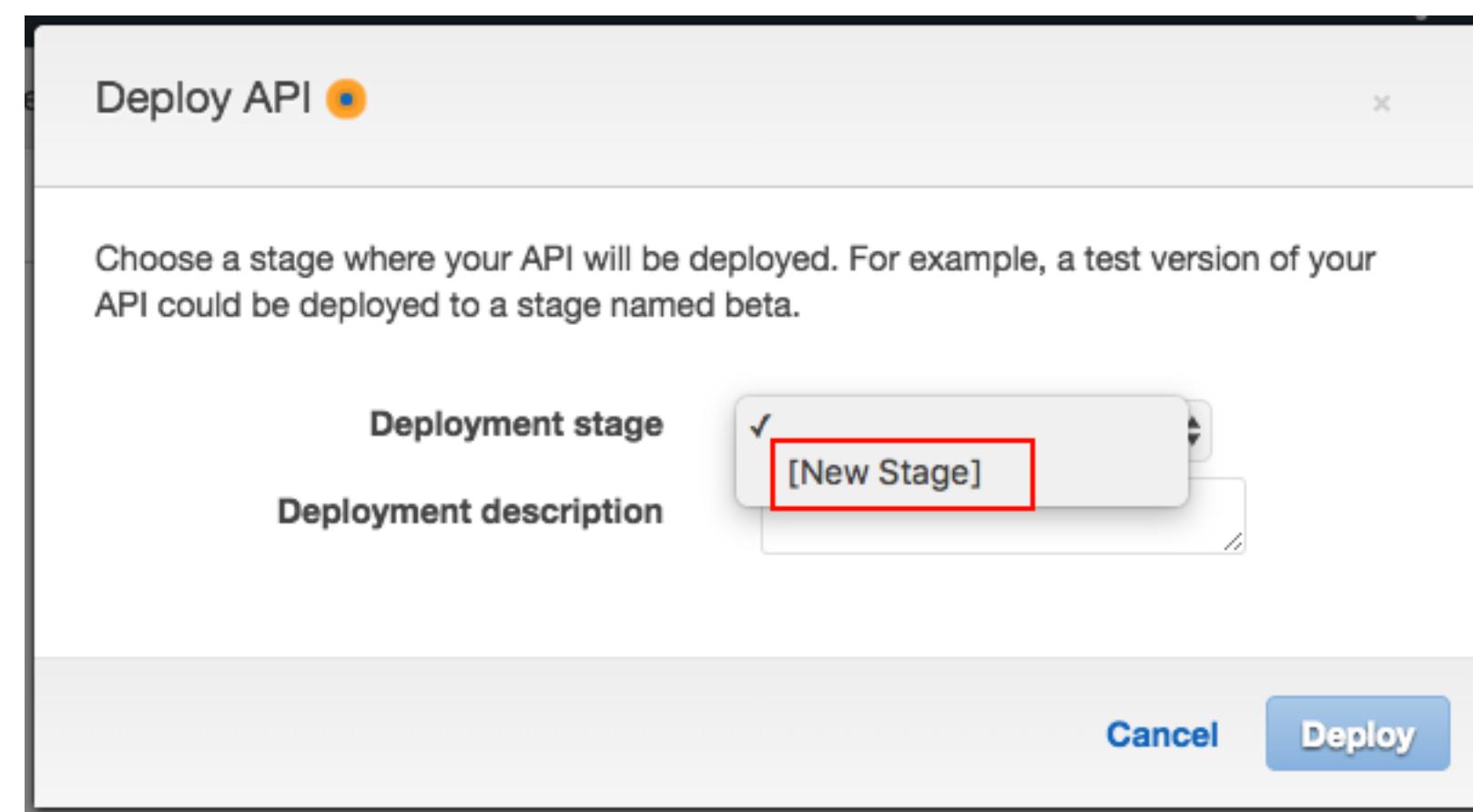
The screenshot shows a REST API testing interface with the following details:

- Resources:** /movies /movies GET /movies OPTIONS /ratings /ratings GET /ratings OPTIONS POST
- Method Execution:** /ratings - GET - Method Test
- Path:** No path parameters exist for this resource. You can define path parameters by using the syntax {myPathParam} in a resource path.
- Query Strings:**
  - moviedId:** Value (input field)
  - userEmail:** tom@gmail.com (input field, highlighted with a red box)
  - rating:** Value (input field)
- Headers:** No header parameters exist for this method. You can add them via Method Request.
- Stage Variables:** No stage variables exist for this method.
- Request Body:** Request Body is not supported for GET methods.
- Test:** A blue button with a lightning bolt icon.
- Response:**
  - Request:** /ratings?userEmail=tom@gmail.com
  - Status:** 200
  - Latency:** 1799 ms
  - Response Body:** [ { "movieId": { "S": "tt6057610" }, "movieTitle": { "S": "The Kill List" }, "userEmail": { "S": "tom@gmail.com" }, "rating": { "N": "2" }, "ratingId": { "S": "cdf4b6c3-dafc-11e7-a5bb-b124db12c854" } }, { "movieId": { "S": "tt6057610" }, "movieTitle": { "S": "The Kill List" }, "userEmail": { "S": "tom@gmail.com" }, "rating": { "N": "2" } ]

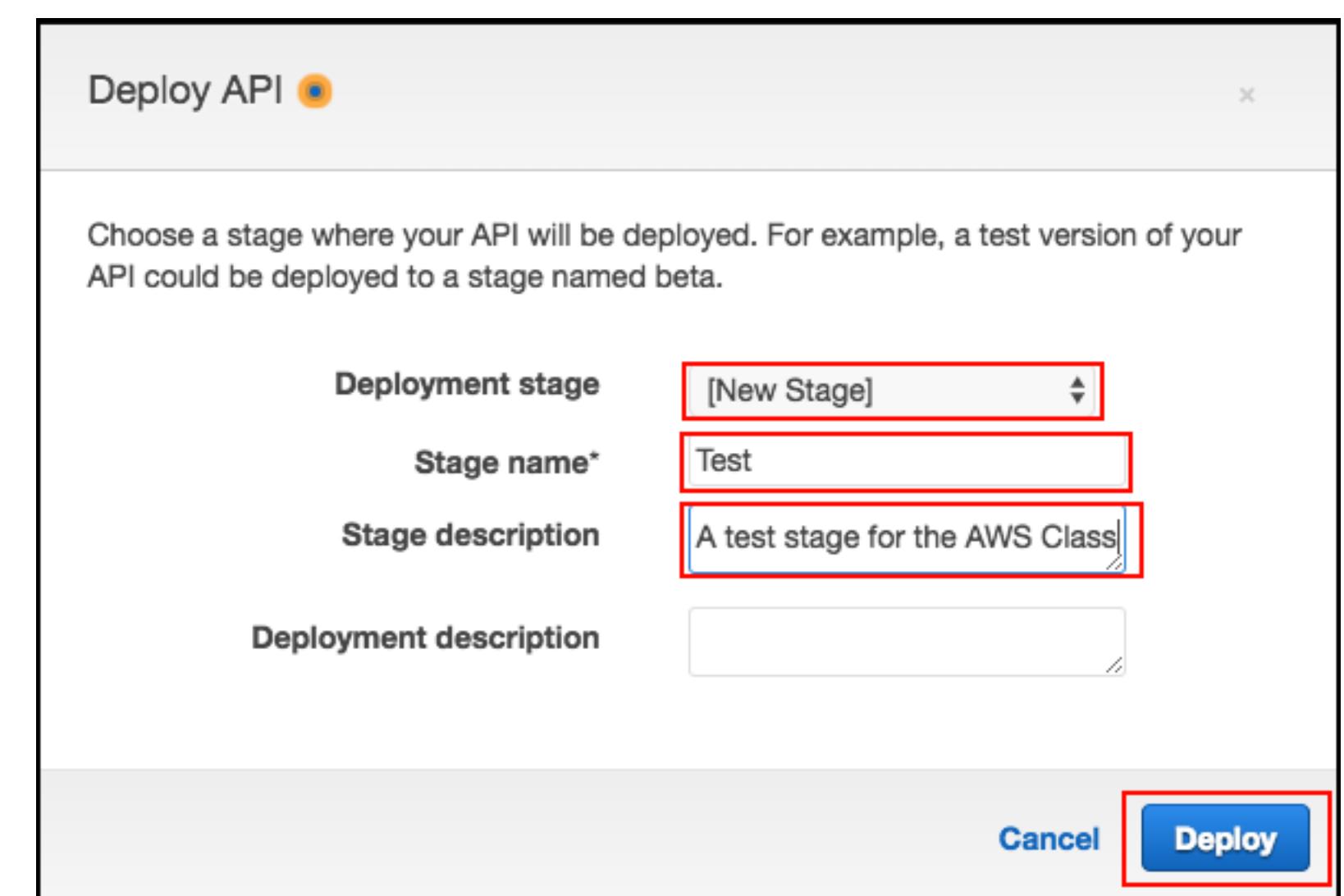
# Deploy API to Test



1

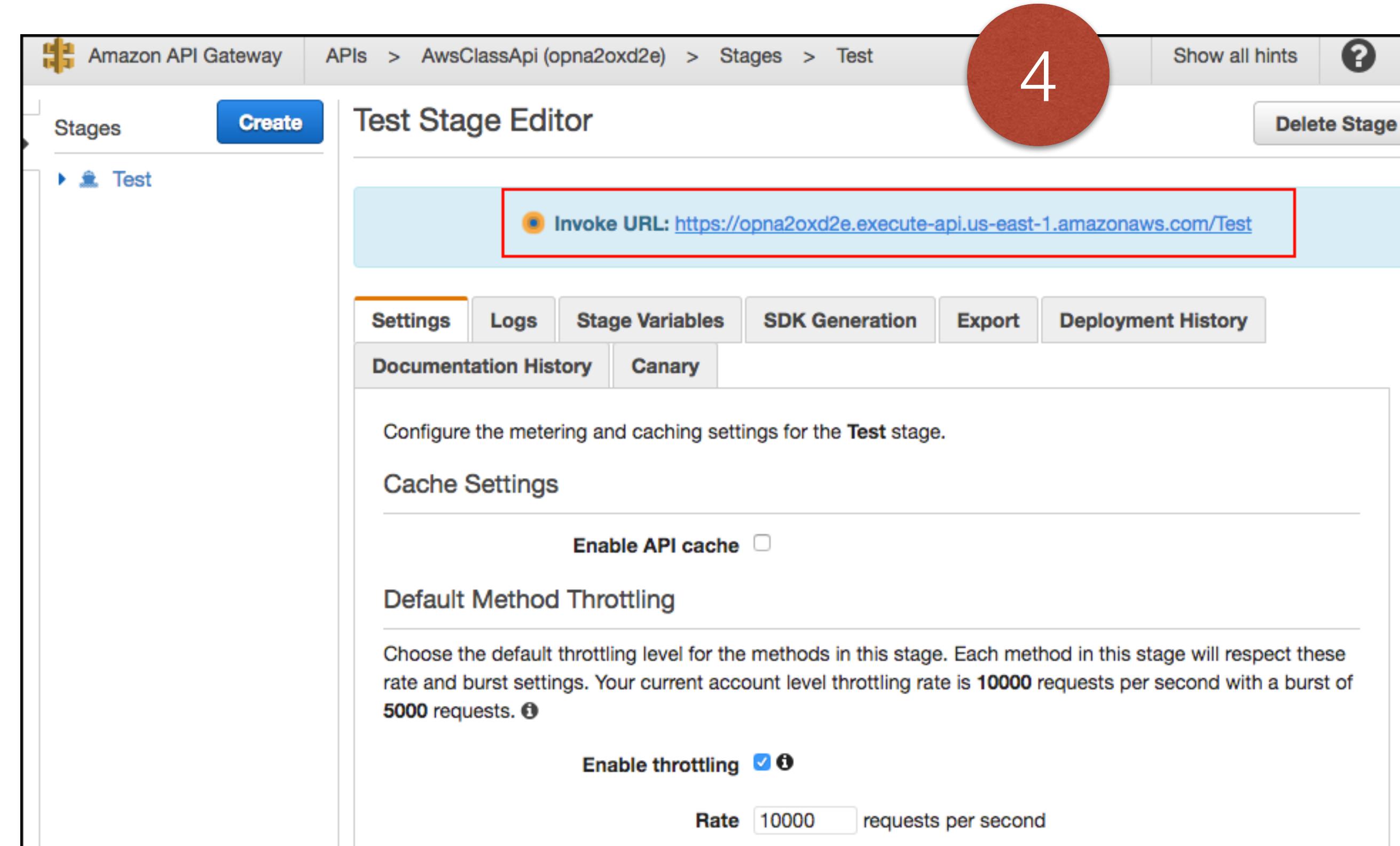


2

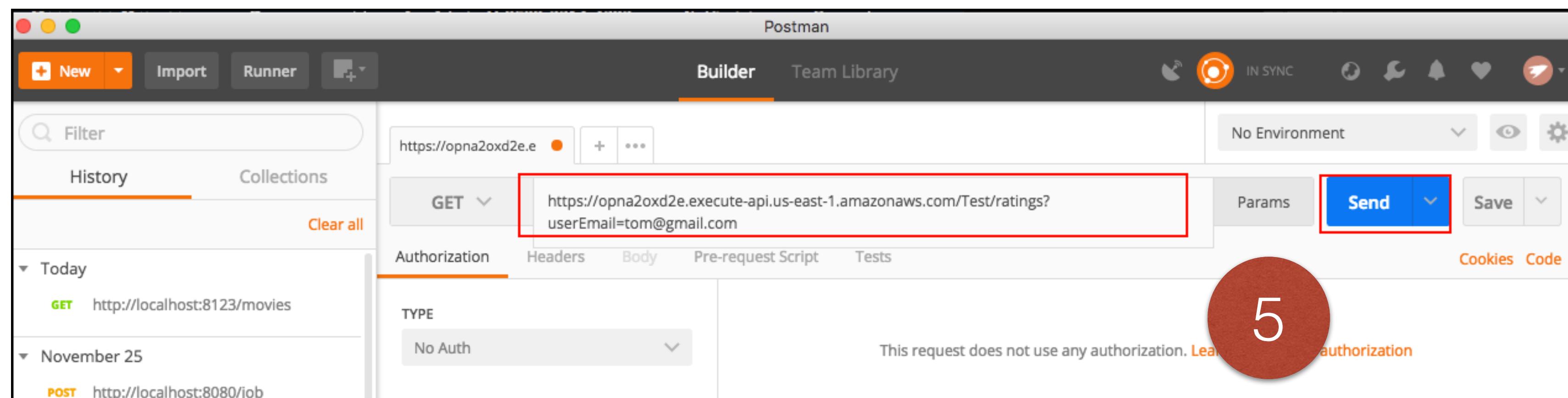


3

# Deploy API to Test



# Deploy API to Test



# Deploy API to Test

The screenshot shows the Postman application interface. At the top, the URL is set to `https://opna20xd2e.e...`. The method is selected as `GET`, and the full URL is `https://opna20xd2e.execute-api.us-east-1.amazonaws.com/Test/ratings?userEmail=tom@gmail.com`. The status bar at the bottom indicates a `200 OK` response with a time of `2029 ms` and a size of `1.44 KB`.

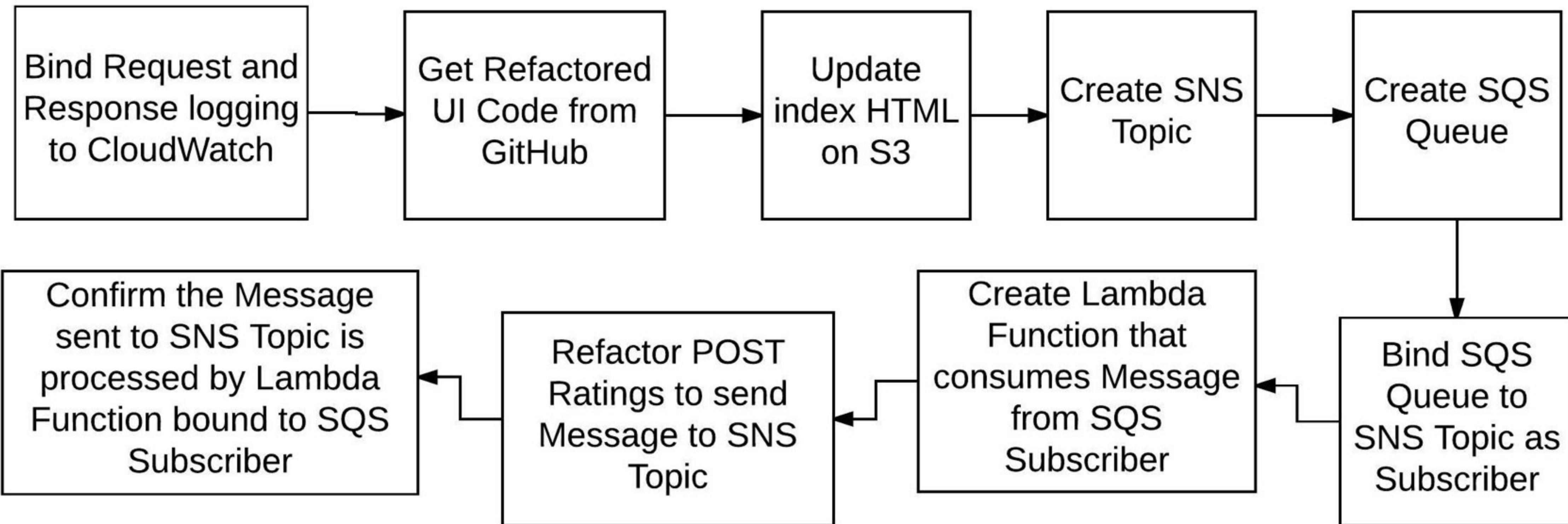
The `Authorization` tab is active, showing `No Auth` selected. A note states: "This request does not use any authorization. [Learn more about authorization](#)".

The `Body` tab is active, displaying the JSON response. The response body is:

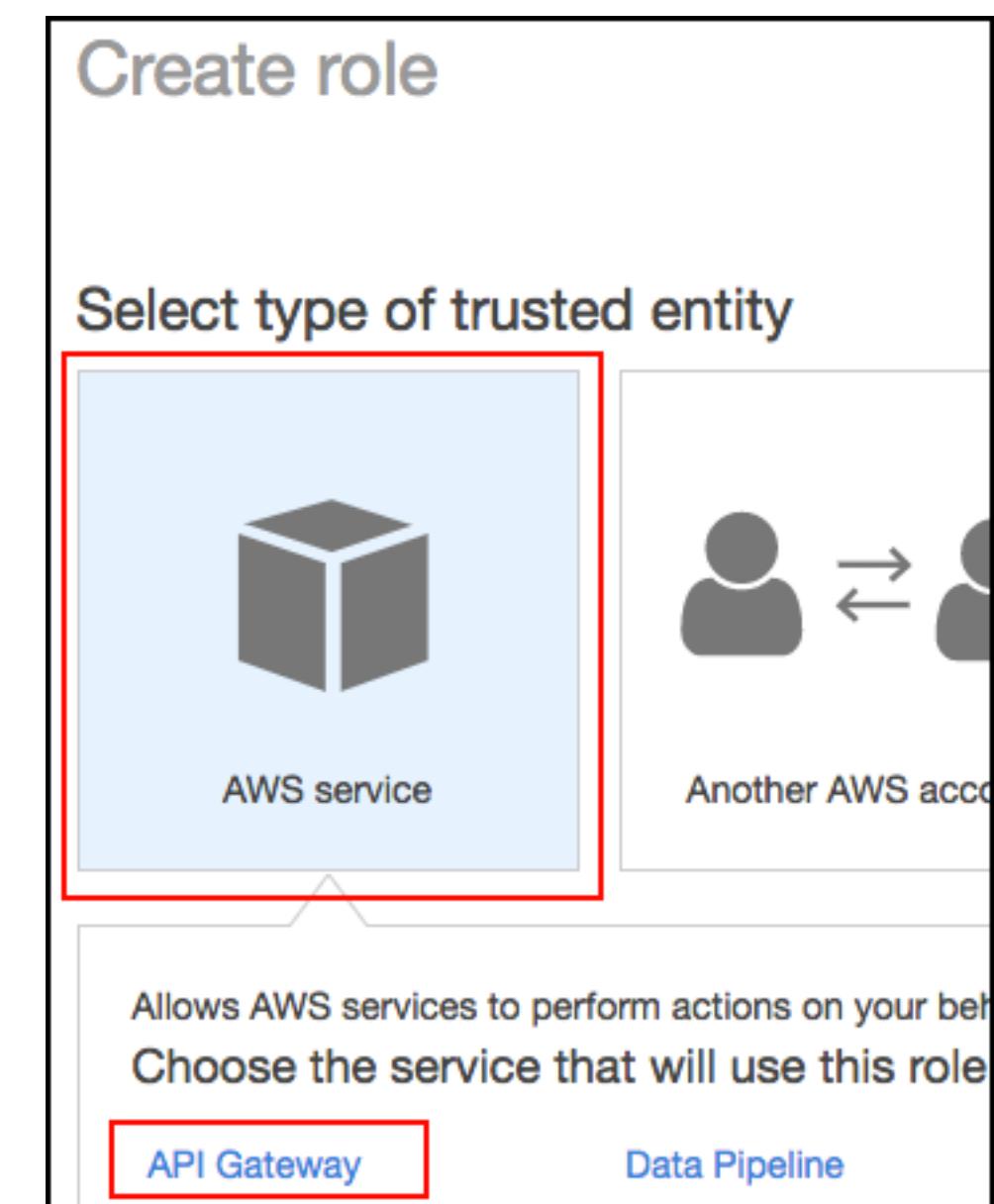
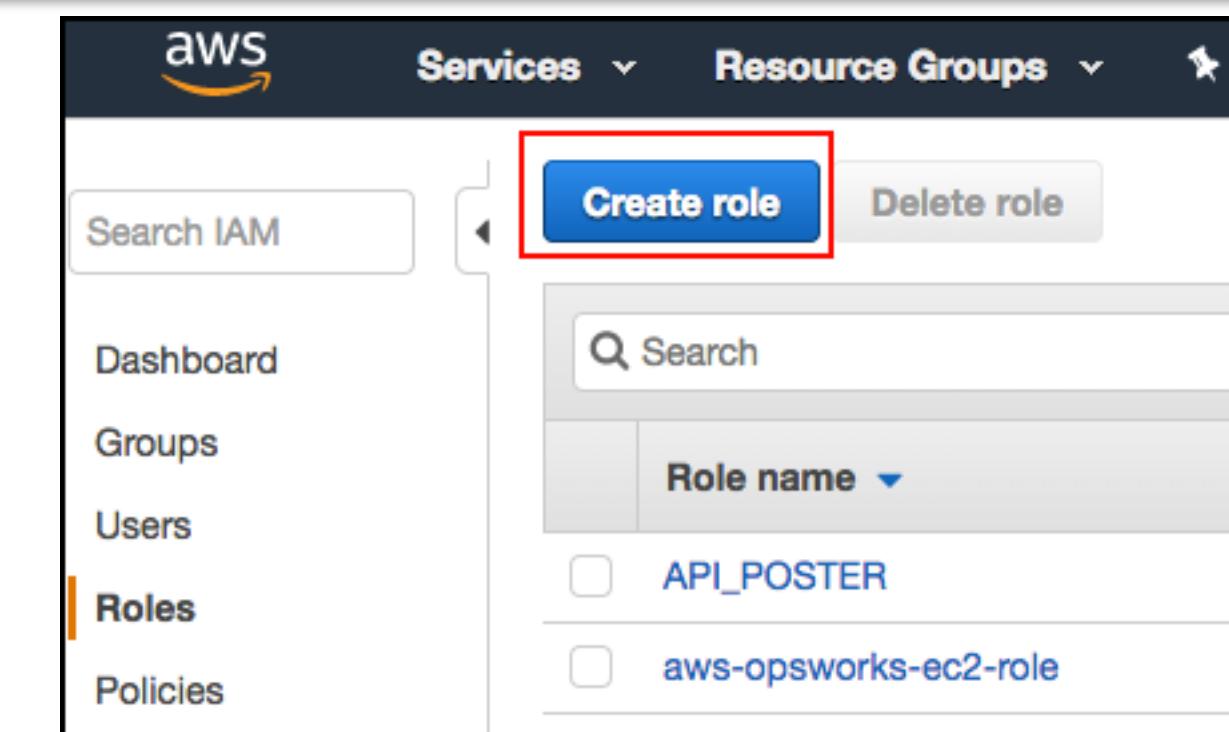
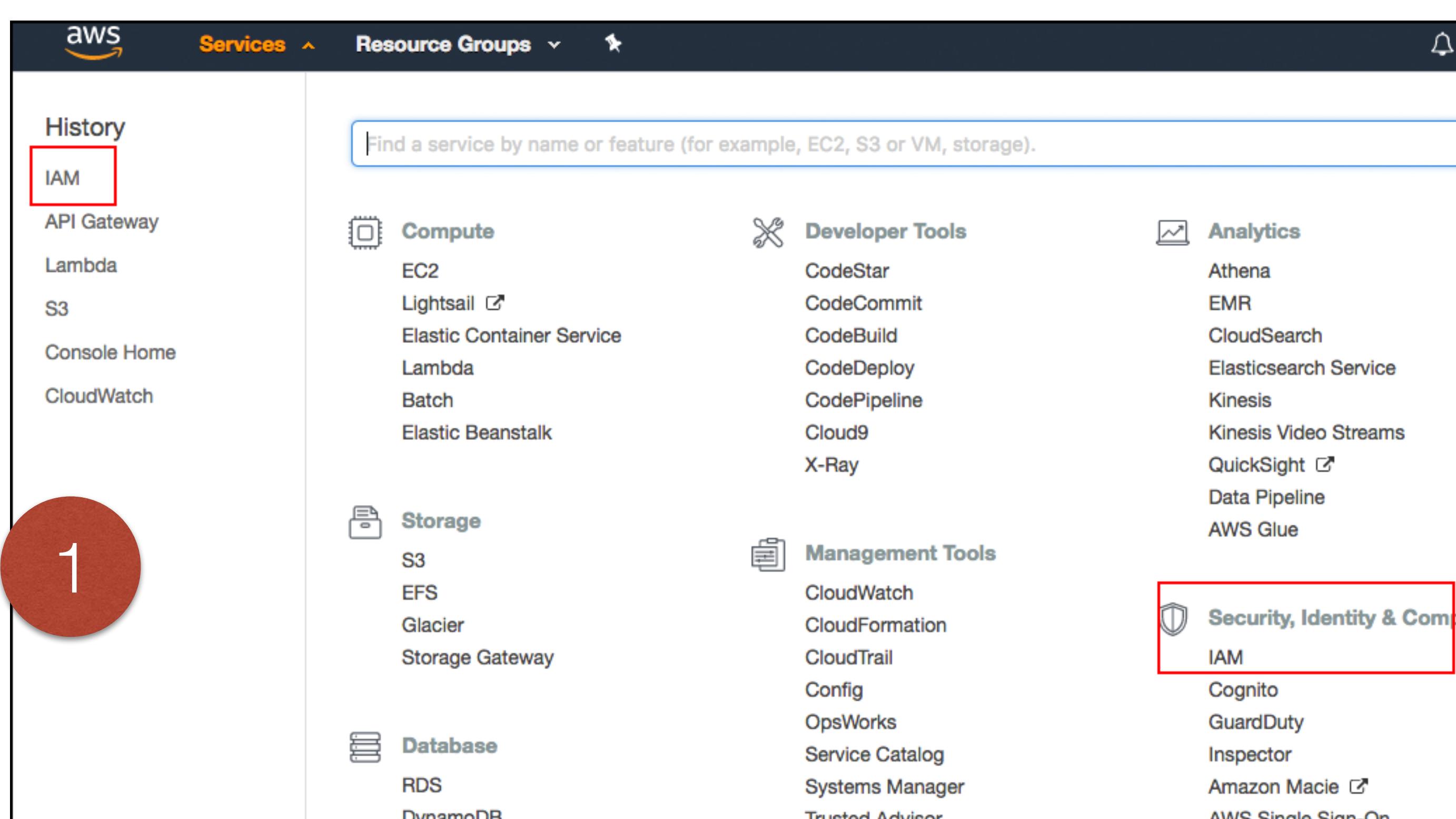
```
1 [
2   {
3     "movieId": {
4       "S": "tt6057610"
5     },
6     "movieTitle": {
7       "S": "The Kill List"
8     },
9     "userEmail": {
10      "S": "tom@gmail.com"
11    },
12    "rating": {
13      "N": "2"
14    }
15  ]
16 ]
```

A red circle with the number **6** is overlaid on the JSON response area.

# Session 6



# Bind Request/Response to CloudWatch



# Bind Request/Response to CloudWatch

Create role

Attached permissions policy

The type of role that you selected requires the following policy.

Filter: Policy type ▾

Policy name	Attachments	Description
AmazonAPIGatewayPushToCloudWatchLogs	2	Allows API Gateway to push logs to user's account.

Showing 1 result

1 Trust    2 Permissions    3 Review

4

Create role

Review

Provide the required information below and review this role before you create it.

Role name\*  Maximum 64 characters. Use alphanumeric and '+,-,@-\_' characters.

Role description

Maximum 1000 characters. Use alphanumeric and '+,-,@-\_' characters.

Trusted entities AWS service: apigateway.amazonaws.com

Policies  AmazonAPIGatewayPushToCloudWatchLogs

\* Required    Cancel    Previous    **Create role**

1 Trust    2 Permissions    3 Review

5

# Bind Request/Response to CloudWatch

6

The screenshot shows the AWS Resource Groups console. On the left, there's a sidebar with links to History, IAM, Console Home, S3, Lambda, CloudWatch, and Simple Notification Service. A red circle with the number 6 is in the top-left corner. In the main area, a search bar contains the text "api". Below it, the "API Gateway" service is listed with the subtext "Build, Deploy and Manage APIs", which is also highlighted with a red box.

7

The screenshot shows the "Deploy API" dialog box. It has fields for "Deployment stage" (set to "[New Stage]"), "Stage name" (set to "TestWithLogging", highlighted with a red box), "Stage description", and "Deployment description". At the bottom are "Cancel" and "Deploy" buttons, with "Deploy" highlighted with a red box. A red circle with the number 7 is in the bottom-right corner.

8

The screenshot shows the "TestWithLogging Stage Editor" page. It includes sections for "Settings", "Logs" (highlighted with a red box), "Stage Variables", and "SDK Generation". Under "CloudWatch Settings", there are checkboxes for "Enable CloudWatch Logs" and "Enable Detailed CloudWatch Metrics", both of which have red boxes around them. A red circle with the number 8 is in the bottom-right corner.

# Bind Request/Response to CloudWatch

The screenshot shows the AWS API Gateway Stage Editor for the 'TestWithLogging' stage of the 'AwsClassApi'. The 'Logs' tab is selected. A red box highlights the 'CloudWatch Settings' section, which includes options for enabling CloudWatch Logs (checked), setting the log level to 'INFO' (selected), and enabling detailed metrics (checked). Another red box highlights the 'Save Changes' button at the bottom right.

aws Services Resource Groups N. Virginia Support

Amazon API Gateway APIs > AwsClassApi (opna20xd2e) > Stages > TestWithLogging Show all hints ?

Stages Create Delete Stage

TestWithLogging Stage Editor

Invoke URL: <https://opna20xd2e.execute-api.us-east-1.amazonaws.com/TestWithLogging>

Settings Logs Stage Variables SDK Generation Export Deployment History

Documentation History Canary

Configure the metering and caching settings for the stage.

CloudWatch Settings

Enable CloudWatch Logs  ⓘ

Log level

Log full requests/responses data

Enable Detailed CloudWatch Metrics  ⓘ

Custom Access Logging

Enable Access Logging

Save Changes

9

# Get Refactored UI Code from GitHub

reselbob / CDAwsClass

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Repository that contains artifacts use in the Code Class, District AWS for Developers

21 commits 7 branches 0 releases 1 contributor MIT

Your recently pushed branches:

Session-6 (4 minutes ago) Compare & pull request

Branch: Session-6 New pull request Create new file Upload files Find file Clone or download

This branch is 18 commits ahead of master.

reselbob Added rater listing

File	Description	Time Ago
.gitignore	Initial commit	13 days ago
AwsClass.role.json	Created AwsClass.role.json	12 days ago
AwsClassLambda.js	removed unnecessary code	9 days ago
AwsClassLambdaEvent.json	created AwsClassLambdaEvent.json	9 days ago
LICENSE	Initial commit	13 days ago
README.md	Initial commit	13 days ago
error.html	Created error.html	13 days ago
getMovies.js	created getMovies.js	10 days ago
getRatings.js	added retrieval using Scan()	6 days ago
getRatingsByMovieId.js	added file extension	6 days ago
getRatingsByEmail.js	created getRatingsByEmail.js	6 days ago
getRatingsParamsMapTemplate.json	added param: rating	6 days ago
index.html	Added rater listing	3 days ago
postRating.js	added file extension	7 days ago
postRatingMap.json	created postRatingMap.json	7 days ago
requestPassthrough.js	Create requestPassthrough.js	9 days ago

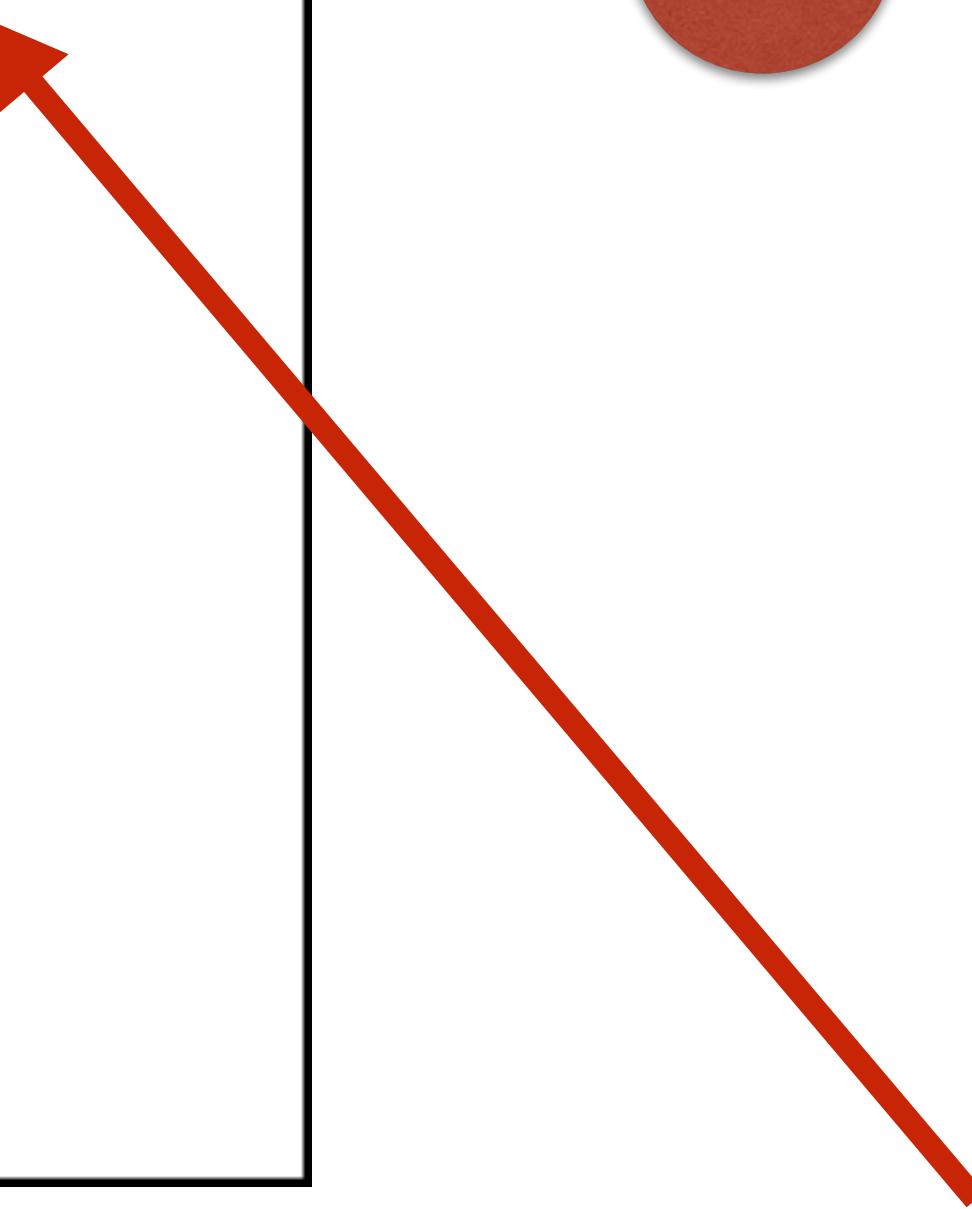
10

# Get Refactored UI Code from GitHub

```
81  </table>
82  <script>
83
84    var API_URL = 'https://opna20xd2e.execute-api.us-east-1.amazonaws.com/Test';
85
86
87    function populateRatingTable(data){
88
89      //alert(JSON.stringify(data));
90      var tbl = document.getElementById("myRatings");
91      data.forEach(function(row){
92        var str = '<td>' + row.movieTitle.S + '</td>';
93        str += '<td>' + row.rating.N + '</td>'
94        str += '<td>' + row.userEmail.S + '</td>';
95        $("#myRatings").last().append('<tr>' + str + '</tr>');
96      });
97    }

```

11



Change to **TestWithLogging**

# Get Refactored UI Code from GitHub

**Code District Aws Class Movie Rater**

Search for a Movie

Movie

Rating

Rater Email

Submit

Your Rating ID

**Movie Rating Rater**

12

**Code District Aws Class Movie Rater**

Search for a Movie

Movie  Select a Movie

Rating

Rater Email

Submit

Your Rating ID

**Movie Rating Rater**

A Girl of Solbakken

- La Blue Girl 3
- A Girl Fighter
- Girl in the Street
- Sweater Girl
- A Nearly Decent Girl
- A Girl of Solbakken
- Girl Without a Room
- Girl in Tails
- What a Girl Wants
- Girl on Girl 2

13

**Code District Aws Class Movie Rater**

Search for a Movie

Movie

Rating

Rater Email

Submit

Your Rating ID

Movie	Rating	Rater
Sweater Girl	3	dick@gmail.com
In the Line of Duty 5: Middle Man 2	2	dick@gmail.com
Dog Wars	4	dick@gmail.com
Sweater Girl	2	dick@gmail.com
Shark Girl	0	dick@gmail.com
West of Nowhere	0	dick@gmail.com
Sweater Girl	3	dick@gmail.com
The Kill List	3	dick@gmail.com
The Bee (Gentle Hero of the West) 2	2	dick@gmail.com
Sweater Girl	2	dick@gmail.com
Strange Dream	4	dick@gmail.com
A Girl of Solbakken	4	dick@gmail.com
A Man Called Autumn Flower	4	dick@gmail.com
Dog Wars	4	dick@gmail.com

14

# Update Index HTML on S3

The image consists of three side-by-side screenshots of the AWS S3 console, each with a large red numbered circle indicating a specific step:

- Step 1:** Shows the AWS Services menu with "S3" selected. A red box highlights the "S3" link in the sidebar.
- Step 2:** Shows the Amazon S3 dashboard with three buckets listed: "aws-class-bucket" (selected), "movie-rater", and "sqsbucket2". A red box highlights the bucket name "aws-class-bucket".
- Step 3:** Shows the "aws-class-bucket" bucket details page. A red box highlights the "Index.html" object in the list, which is checked. A red box also highlights the "Rename" option in the context menu for this object.

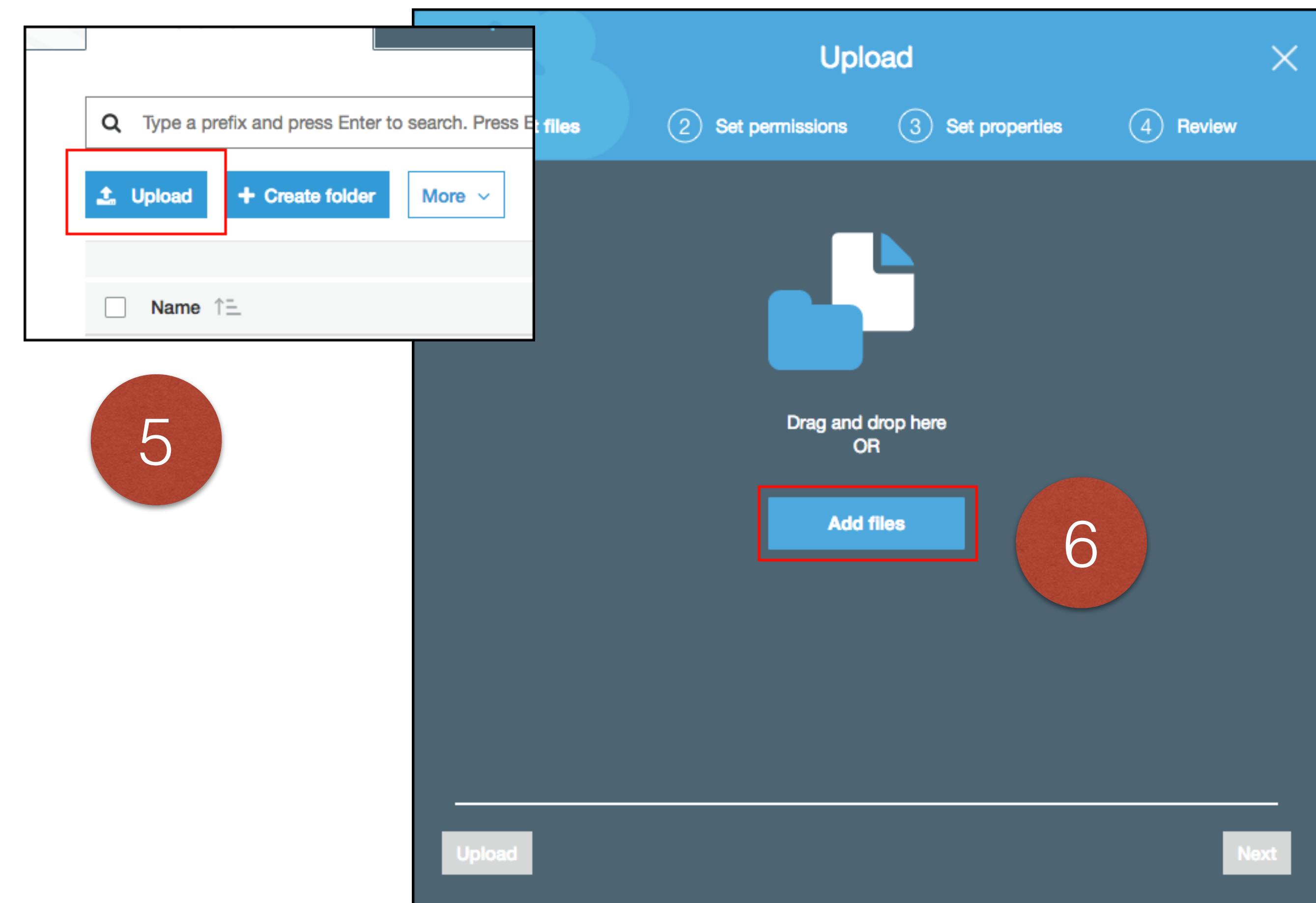
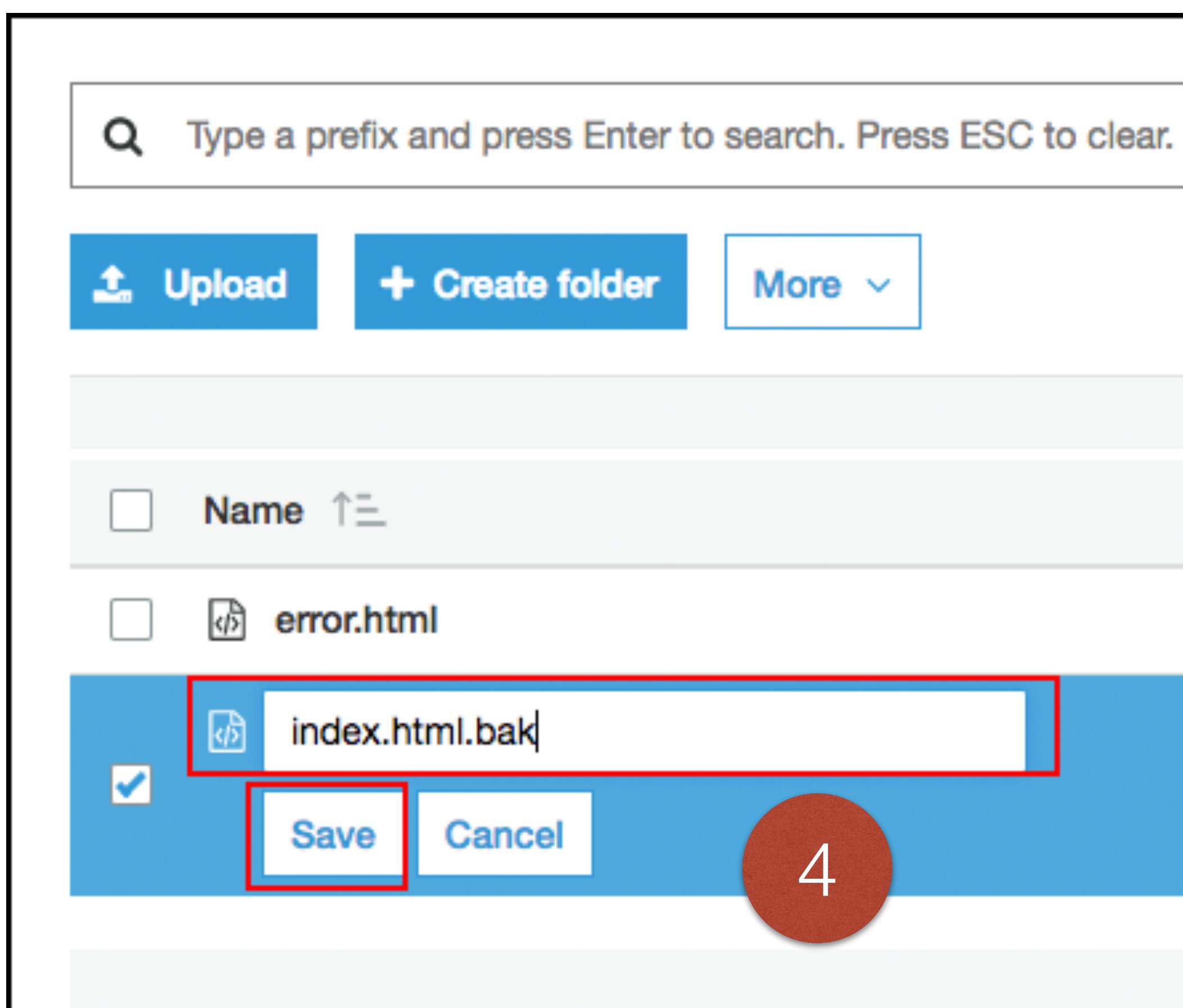
**Screenshot 1: AWS Services Menu**

**Screenshot 2: Amazon S3 Dashboard**

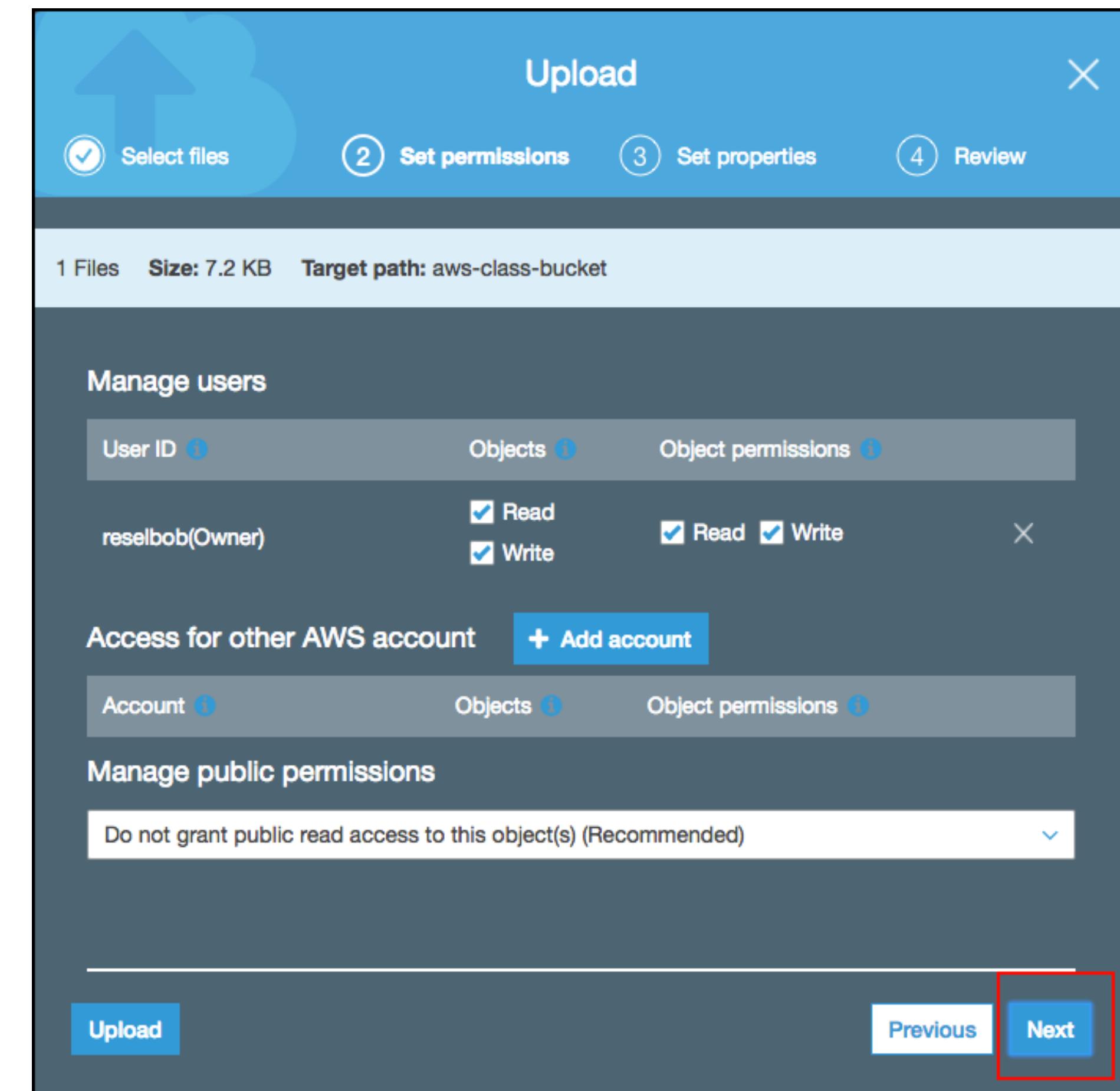
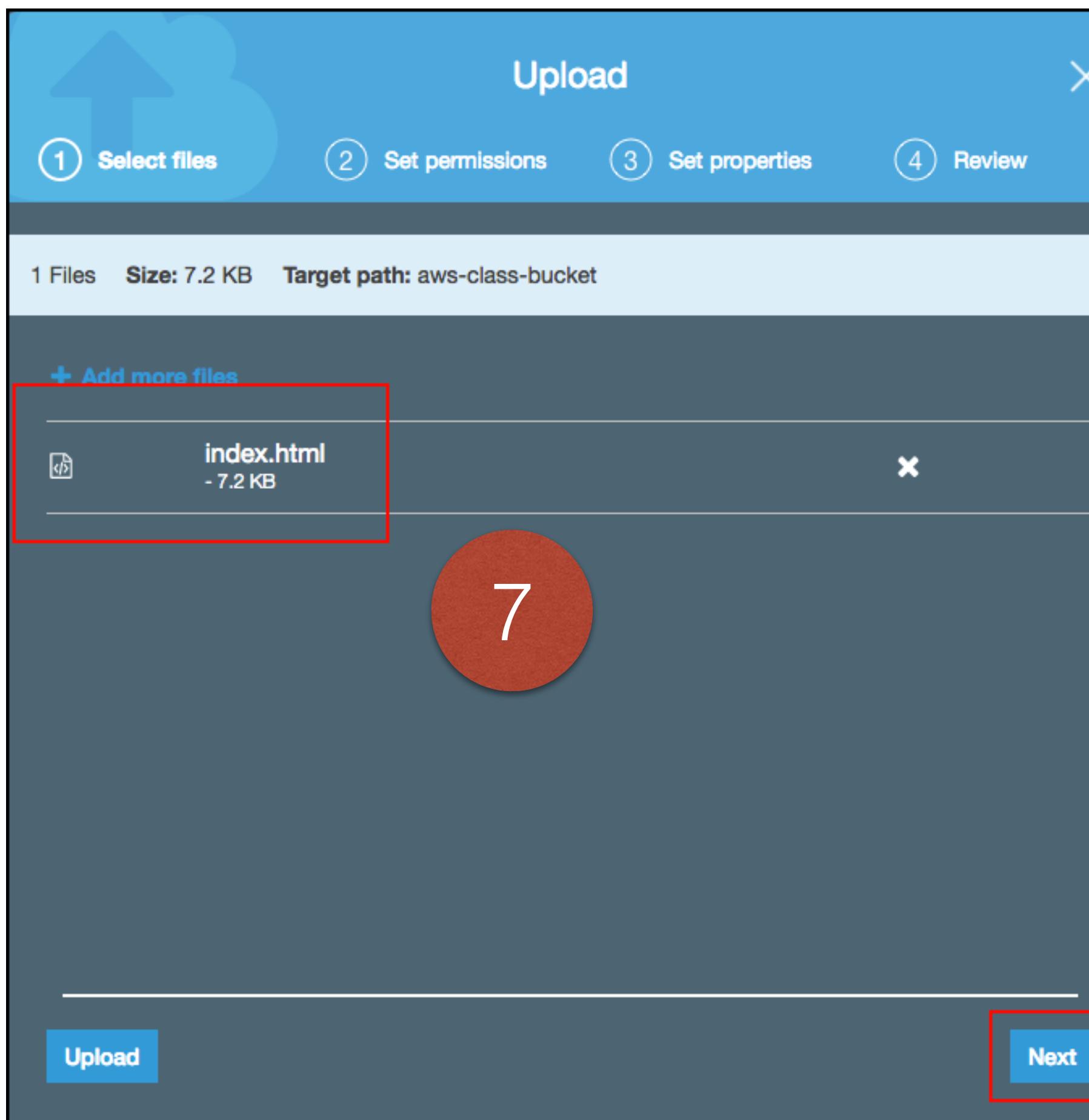
**Screenshot 3: Bucket Details Page**

\* Objects might still be publicly accessible due to object ACLs. [Learn more](#)

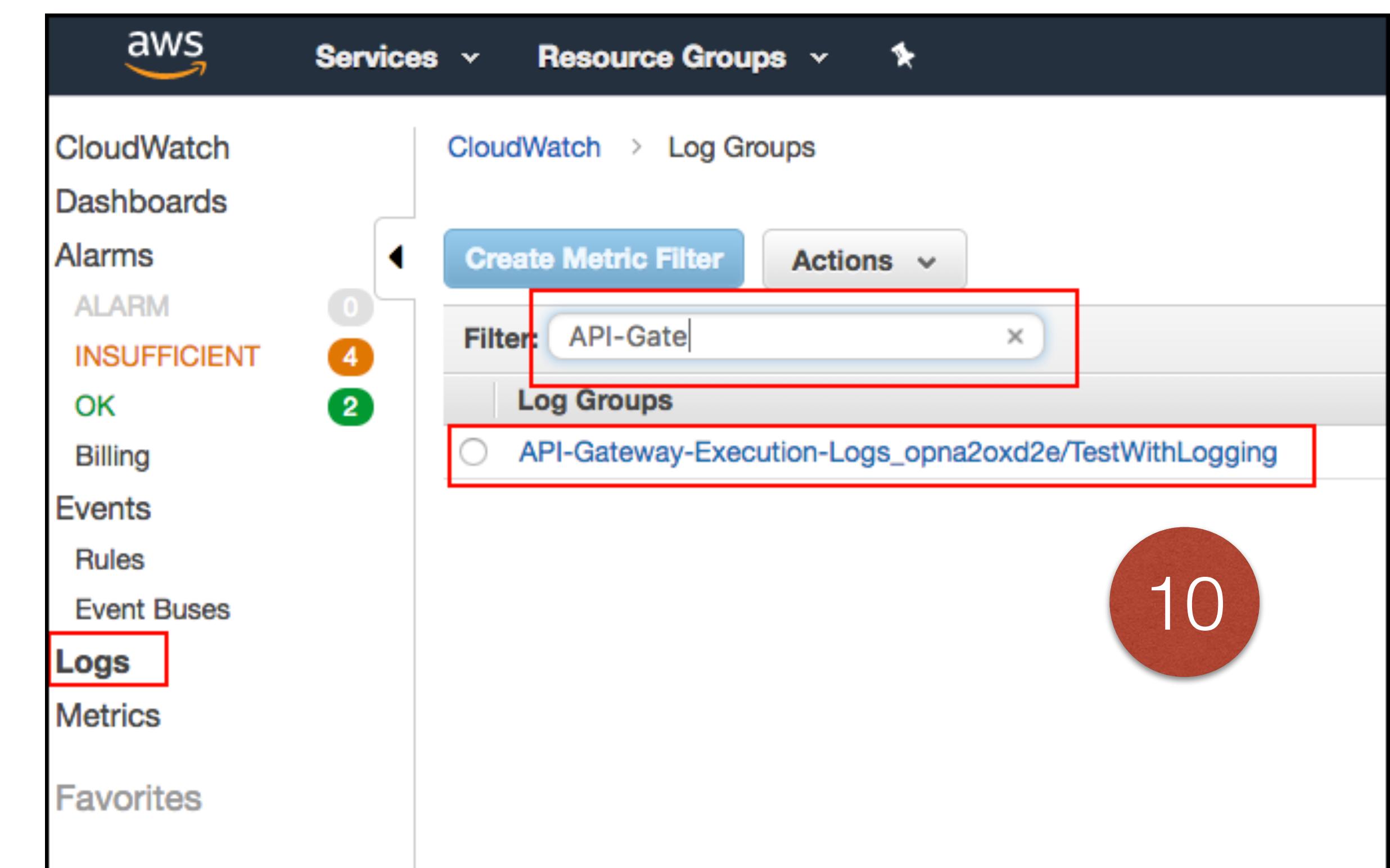
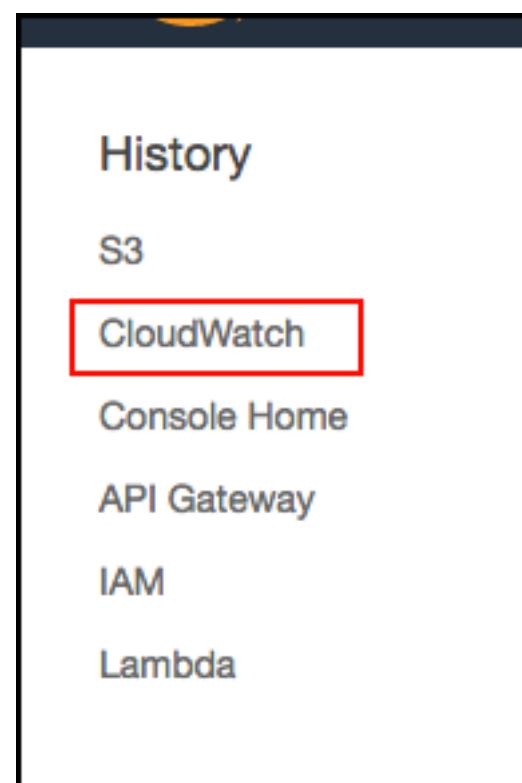
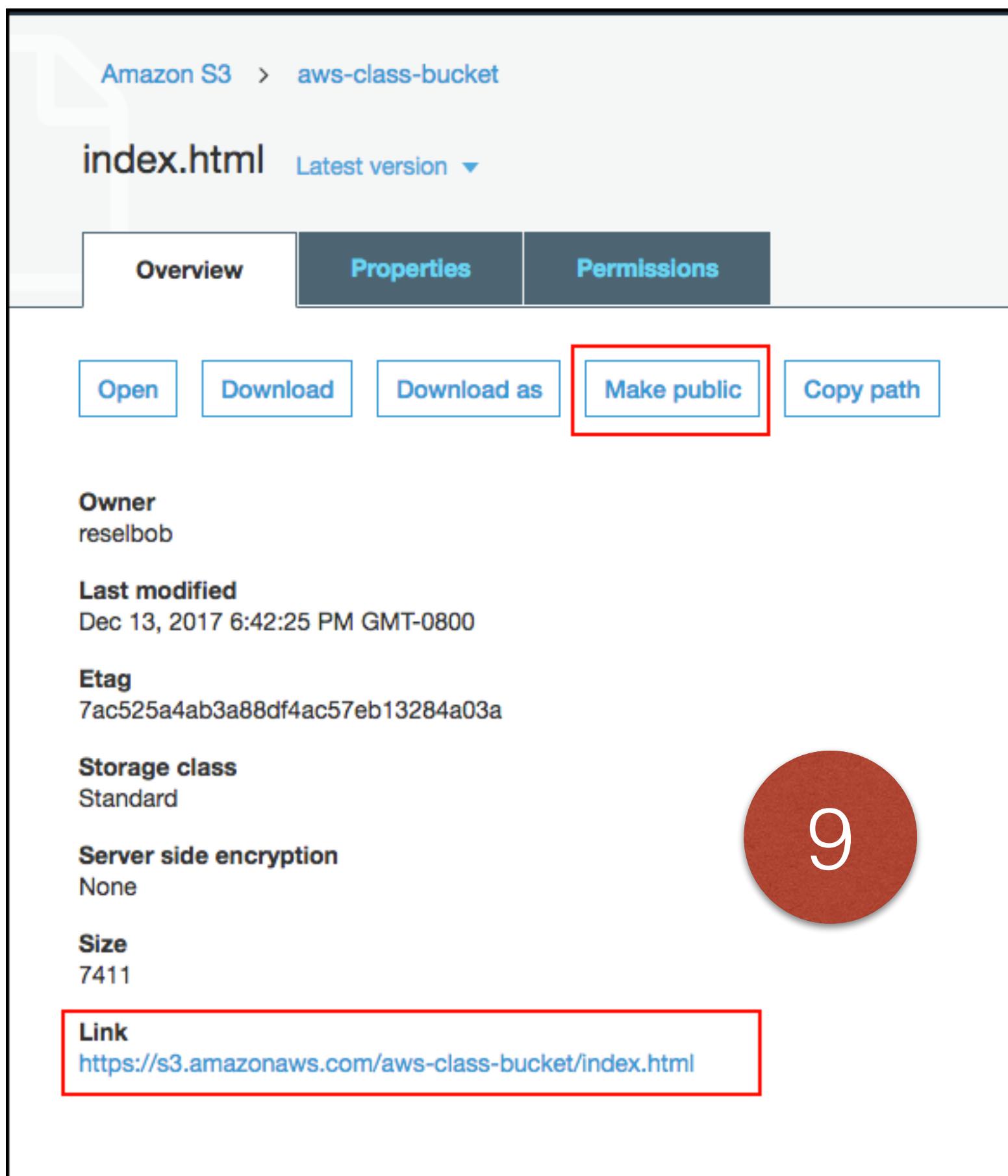
# Update Index HTML on S3



# Update Index HTML on S3



# Update Index HTML on S3



# Update Index HTML on S3

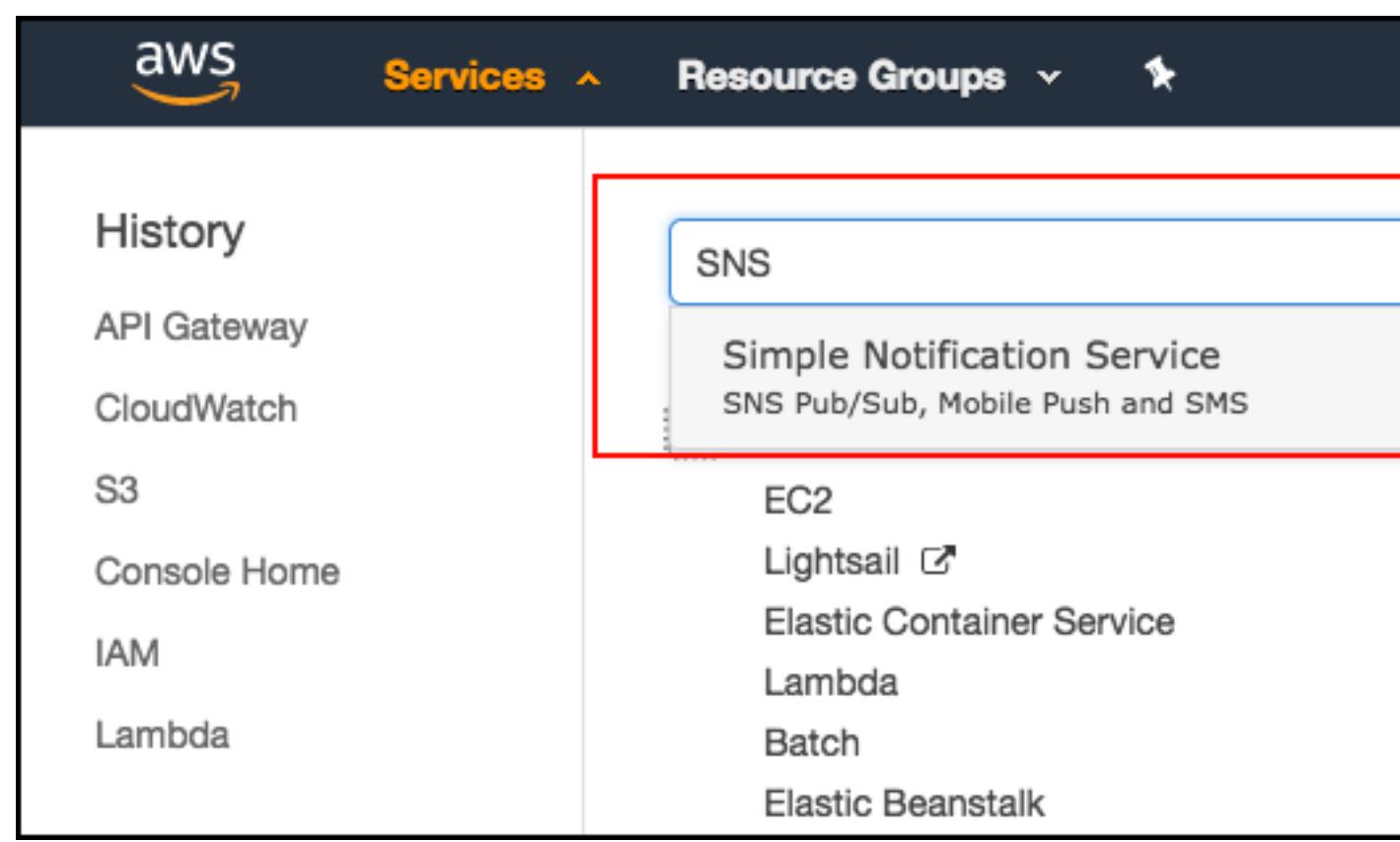
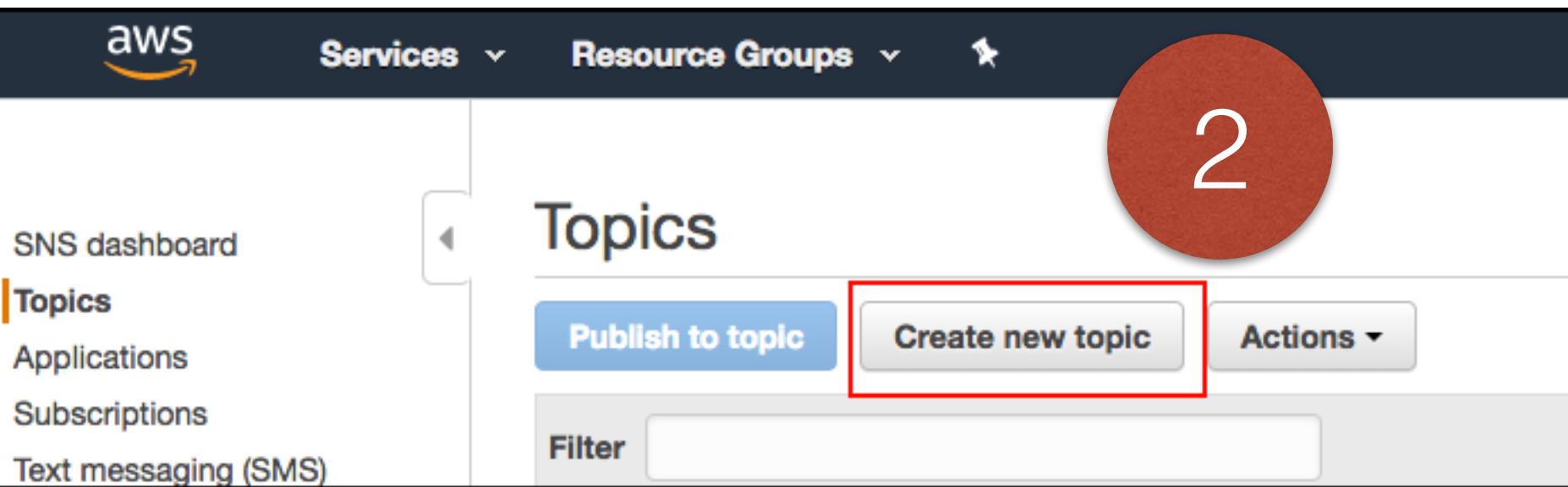
11

The screenshot shows the AWS CloudWatch Logs interface. The left sidebar has a red circle with the number 11. The main area shows a log group path: CloudWatch > Log Groups > API-Gateway-Execution-Logs\_opna2oxd2e/TestWithLogging > 46ba9f2a6976570b035320. The log table has two columns: Time (UTC +00:00) and Message. The table header says "Filter events". The date 2017-12-14 is selected. The log entries show the execution of an API request, starting with verifying the usage plan and API key, followed by starting the execution, sending the request to Lambda, receiving the response, and finally completing the execution with status 200.

Time (UTC +00:00)	Message
2017-12-14	No older events found at the moment. <a href="#">Retry</a> .
02:25:26	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Verifying Usage Plan for request: 0ba8483d-e076-11e7-bffc-7dbcf035ecc6
02:25:26	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) API Key authorized because method 'GET /rating'
02:25:26	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Usage Plan check succeeded for API Key and AP
02:25:26	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Starting execution for request: 0ba8483d-e076-11e7-bffc-7dbcf035ecc6
02:25:26	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) HTTP Method: GET, Resource Path: /ratings
02:25:26	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Method request path: {}
02:25:26	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Method request query string: {userEmail=dick@g
02:25:26	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Method request headers: {Accept=*, CloudFront-Forwarded-SSL=0}
02:25:26	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Method request body before transformations:
02:25:26	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Endpoint request URI: https://lambda.us-east-1.amazonaws.com/functions/test-with-logging
02:25:26	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Endpoint request headers: {x-amzn-lambda-integration-type=HTTP}
02:25:26	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Endpoint request body after transformations: { "name": "dick", "rating": 5 }
02:25:26	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Sending request to https://lambda.us-east-1.amazonaws.com/functions/test-with-logging
02:25:27	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Received response. Integration latency: 1446 ms
02:25:27	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Endpoint response body before transformations:
02:25:27	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Endpoint response headers: {X-Amz-Executed-Version=1.0, Content-Type=application/json}
02:25:27	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Method response body after transformations: [{"rating": 5}])
02:25:27	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Method response headers: {Access-Control-Allow-Origin=*, Content-Type=application/json}
02:25:27	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Successfully completed execution
02:25:27	(0ba8483d-e076-11e7-bffc-7dbcf035ecc6) Method completed with status: 200
	No newer events found at the moment. <a href="#">Retry</a> .

# Bind SQS Queue to SNS Topic as Subscriber

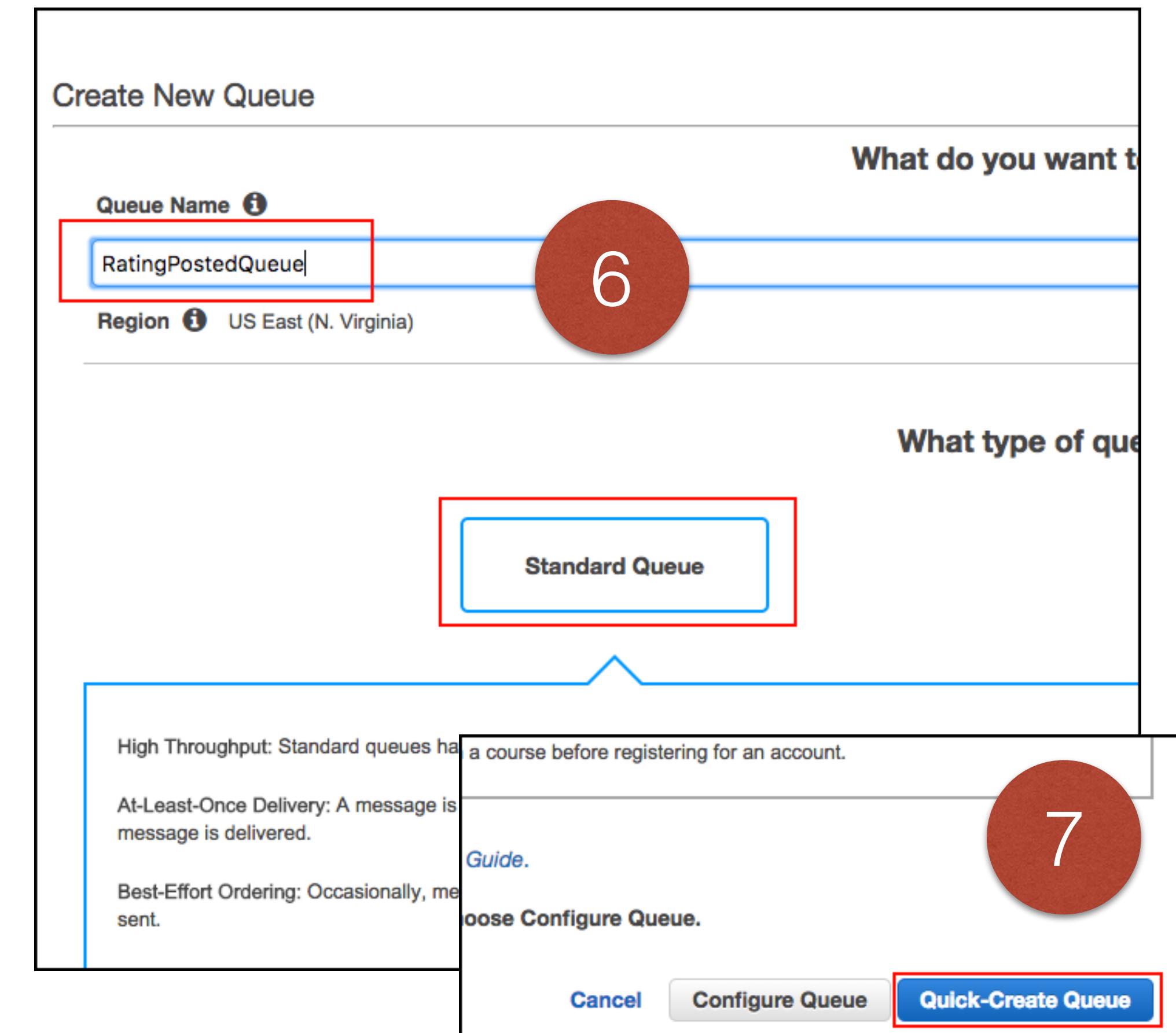
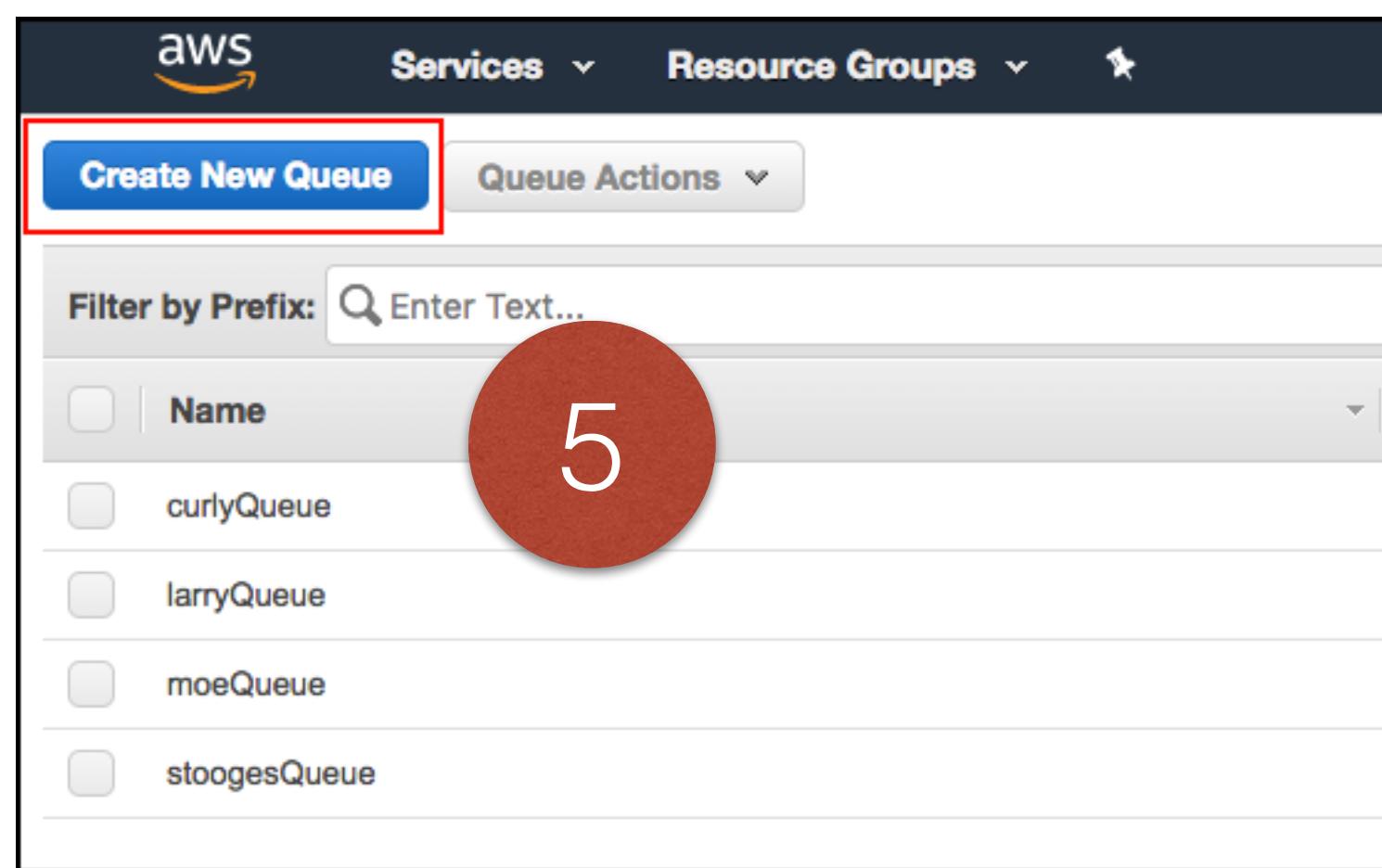
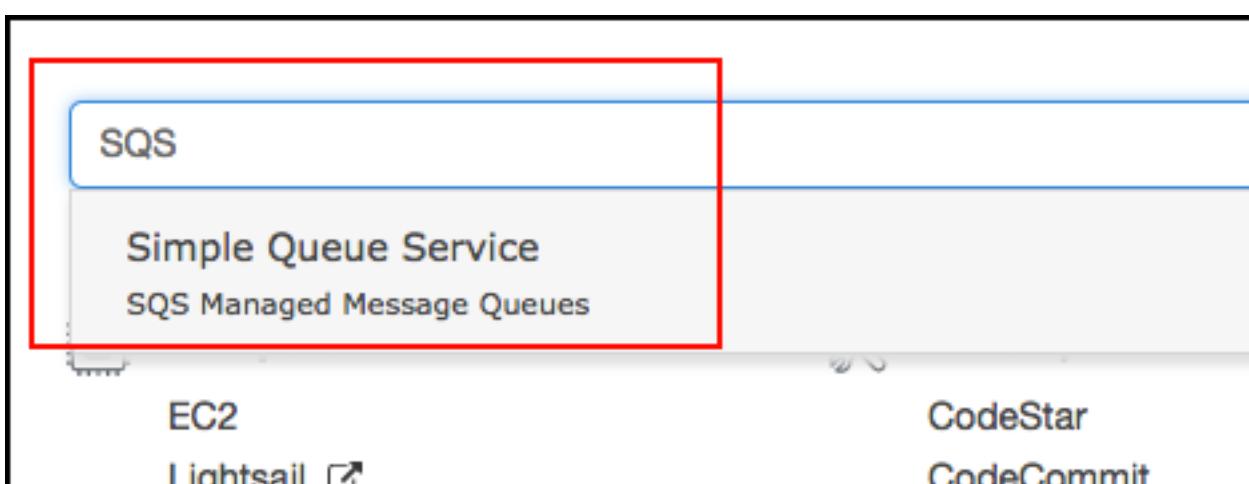
1



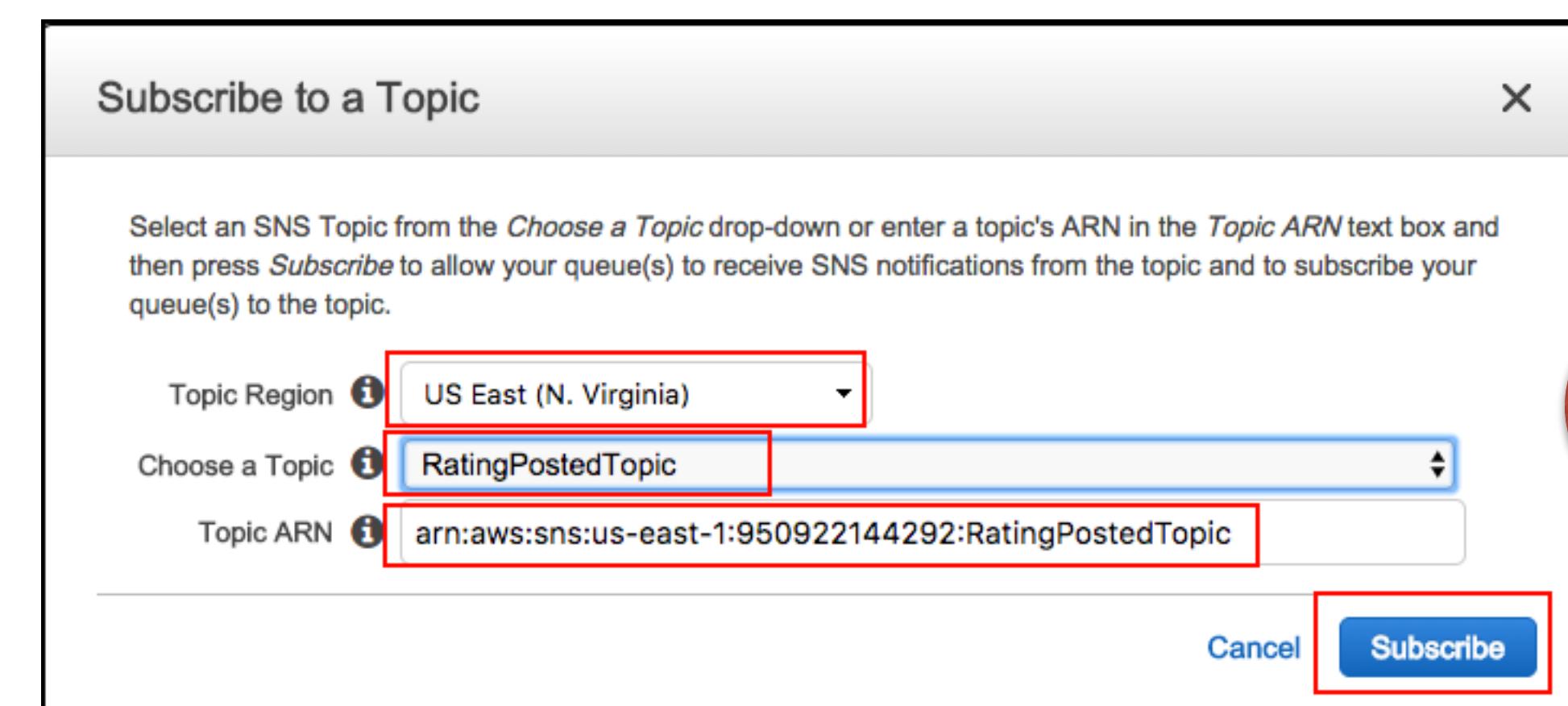
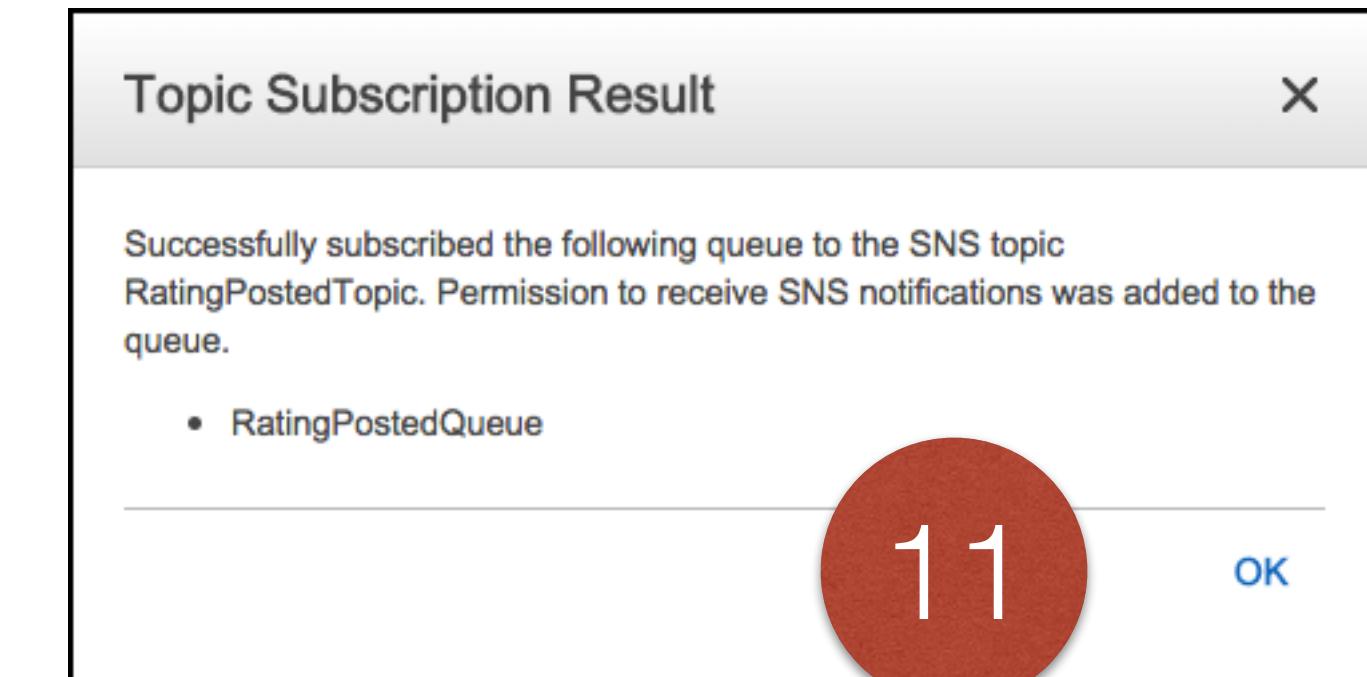
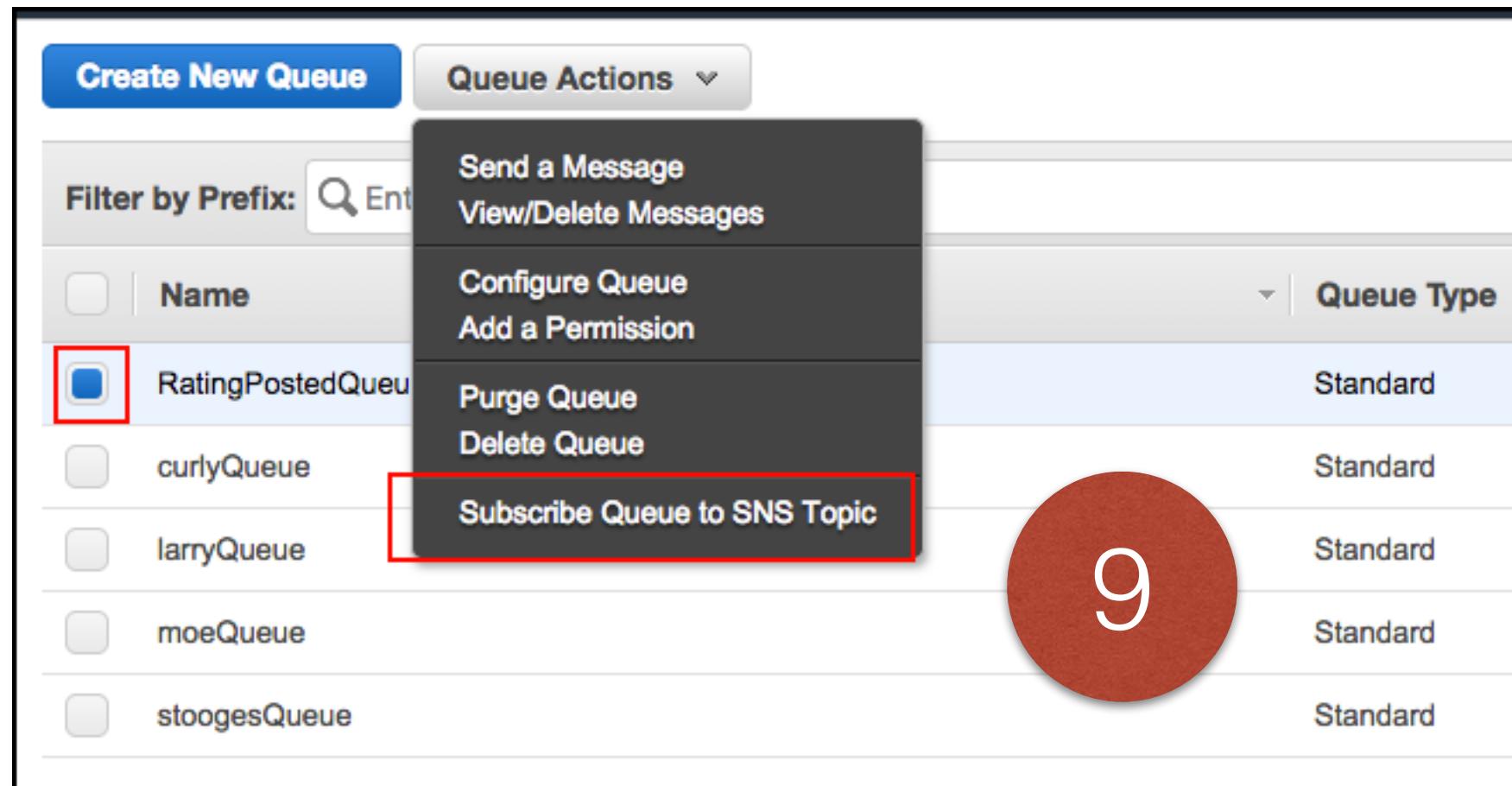
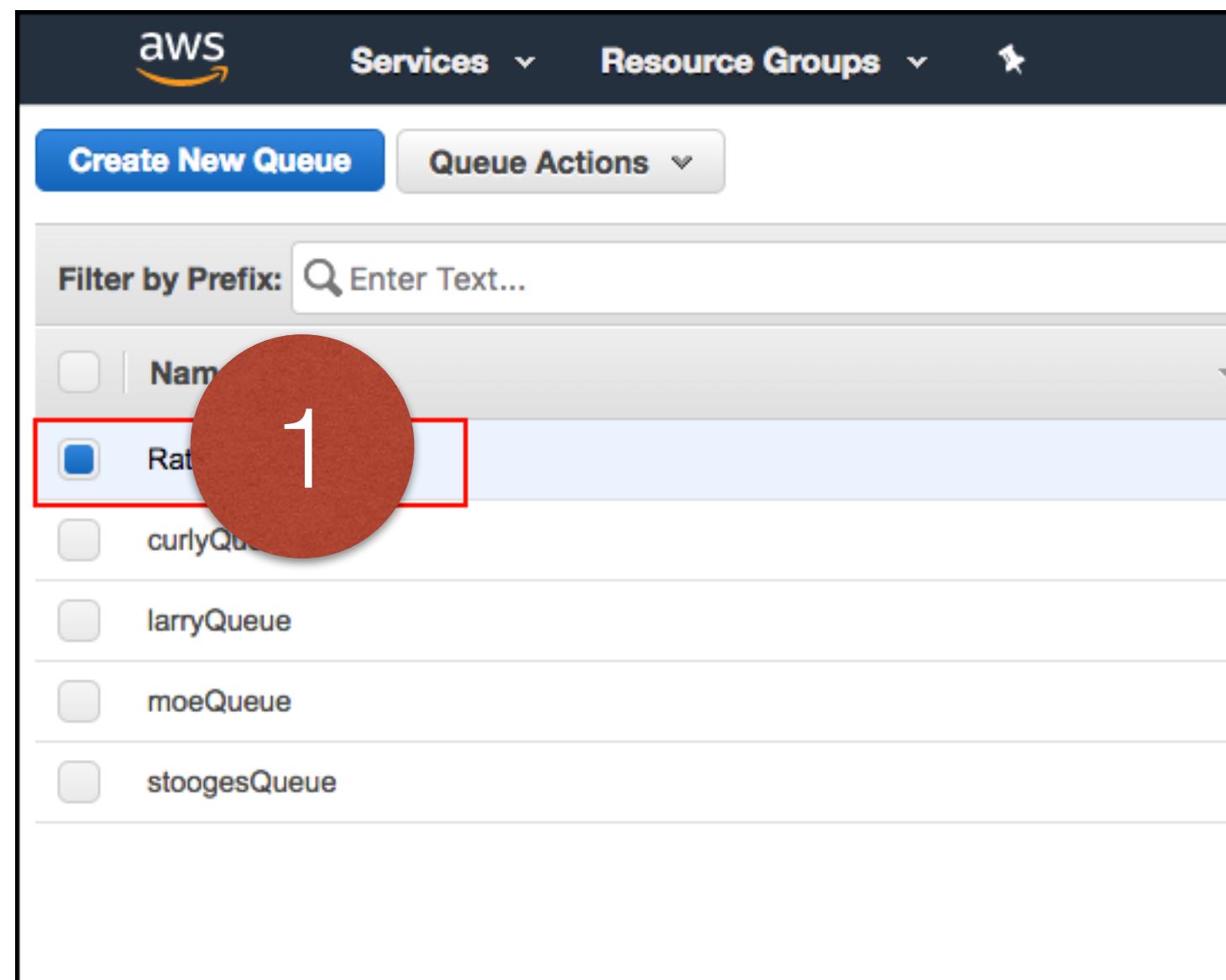
3

The screenshot shows the 'Create new topic' dialog box. It has a title bar 'Create new topic'. Below it, a note says: 'A topic name will be used to create a permanent unique identifier called an Amazon Resource Name (ARN.)'. There are two input fields: 'Topic name' containing 'RatingPostedTopic' (highlighted with a red box) and 'Display name' containing 'Enter topic display name. Required for topics with SMS subscriptions.' At the bottom right are 'Cancel' and 'Create topic' buttons, with 'Create topic' also highlighted with a red box.

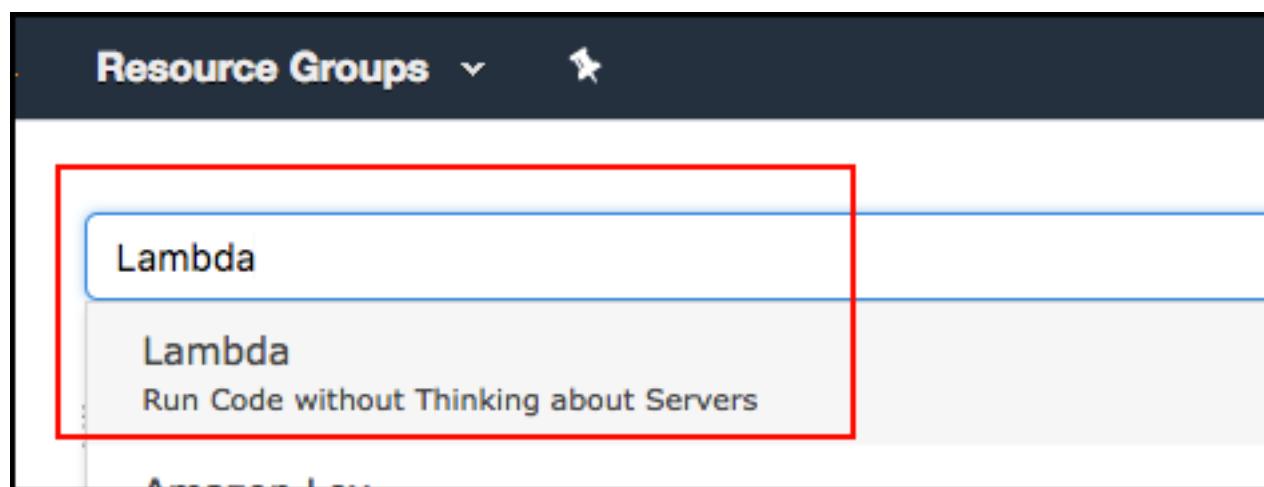
# Bind SQS Queue to SNS Topic as Subscriber



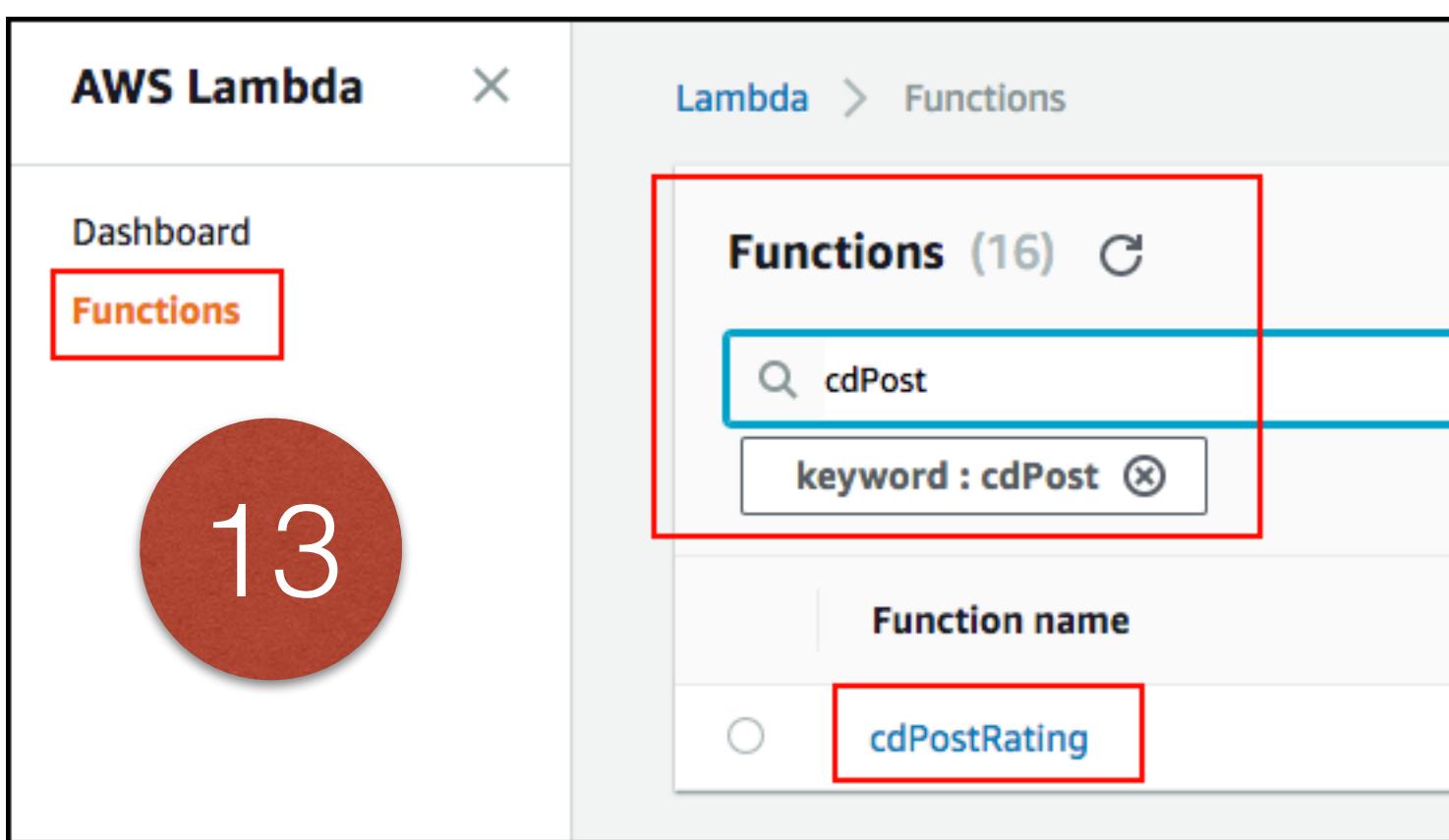
# Bind SQS Queue to SNS Topic as Subscriber



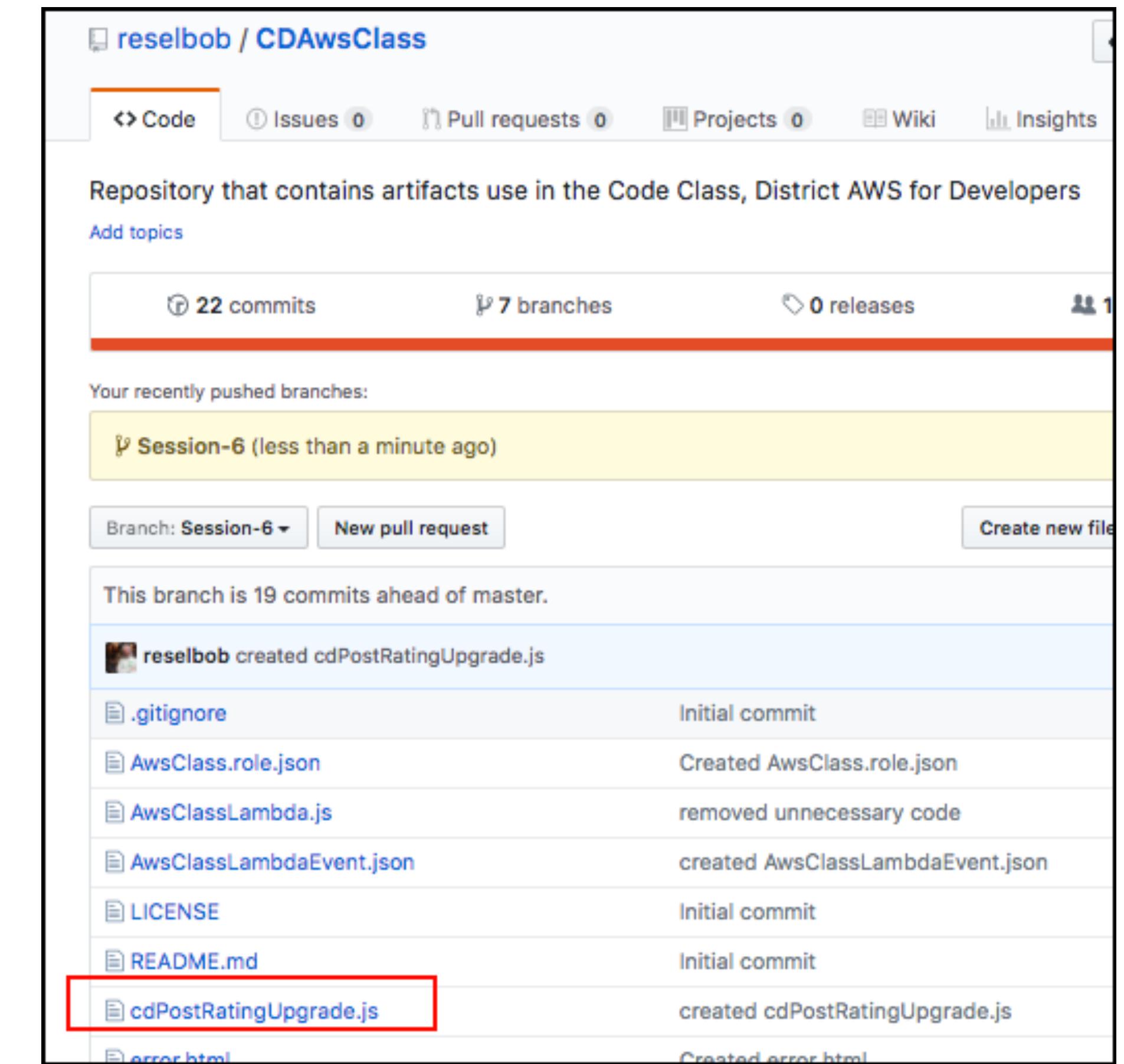
# Bind SQS Queue to SNS Topic as Subscriber



12



13



14

# Bind SQS Queue to SNS Topic as Subscriber

```
24 /*  
25 sends the rating onto the SNS Topic  
26 */  
27 function sendRatingToTopic(rating, callback){  
28     var sns = new AWS.SNS();  
29     console.log({message: 'Send sending rating to topic: arn:aws:sns:us-east-1:950922144292:RatingPostedTopic.',rating });  
30     sns.publish({  
31         TopicArn: "arn:aws:sns:us-east-1:950922144292:RatingPostedTopic",  
32         Subject: "From cdPostRating",  
33         Message: JSON.stringify(rating)  
34     }, callback);  
35 }  
36
```

15

```
42 const rating = {  
43     "movieId": {"S": event.movieId },  
44     "movieTitle": {"S":event. movieTitle },  
45     "userEmail": {"S": event.userEmail},  
46     "rating": {"N": event.rating},  
47     "ratingId": {"S": ratingId},  
48 };  
49
```

16

```
54 dynamodb.putItem({  
55     "TableName": tableName,  
56     "Item" : rating  
57 },  
58     function(err, data) {  
59         if (err) {  
60             context.done('error','putting item into dynamodb failed: '+ err);  
61         }  
62         else {  
63             console.log({message: 'Data written.',rating });  
64             sendRatingToTopic(rating, function(err, data){  
65                 if (err) {  
66                     console.log({message: 'Post to topic failed',rating, err });  
67                     context.fail(JSON.stringify({message: 'Post to topic failed',rating, err}));  
68                 }  
69             })  
70         }  
71     }  
72});
```

17

# Bind SQS Queue to SNS Topic as Subscriber

**Configure test event**

A function can have up to 10 test events. The events are persisted so you can switch to and test your function with the same events.

Create new test event  
 Edit saved test events

**Saved Test Event**

postParams

```
1 [{}  
2   "movieId": "tt0035400",  
3   "movieTitle": "Sweater Girl",  
4   "userEmail": "dick@gmail.com",  
5   "rating": "3"  
6 ]
```

18

aws Services Resource Groups AwsAdmin @ 9509-2214-4292 N. Virginia

**cdPostRating** Qualifiers Actions postParams **Test** Save

**FUNCTION CODE**

Code entry type: Edit code inline Runtime: Node.js 6.10 Handler: index.handler

File Edit Find View Goto Tools Window

Environment cdPostRating index.js

```
54 dynamodb.putItem({  
55   "TableName": tableName,  
56   "Item" : rating  
57 },  
58   function(err, data) {  
59     if (err) {  
60       context.done('error','putting item into dynamodb failed');  
61     } else {  
62       console.log({message: 'Data written.',rating});  
63       sendRatingToTopic(rating, function(err, data){  
64         if (err) {  
65           console.log({message: 'Post to topic failed',rating});  
66           context.fail(JSON.stringify({message: 'Post to topic failed'}));  
67         }  
68       });  
69     }  
70   }  
71 }  
72 );  
73 };
```

(2855 Bytes) 73:3 JavaScript Spaces: 4

Execution Result: +

Execution Results: Status: Succeeded Max Memory Used: 36 MB Time: 640.66 ms

Feedback

19

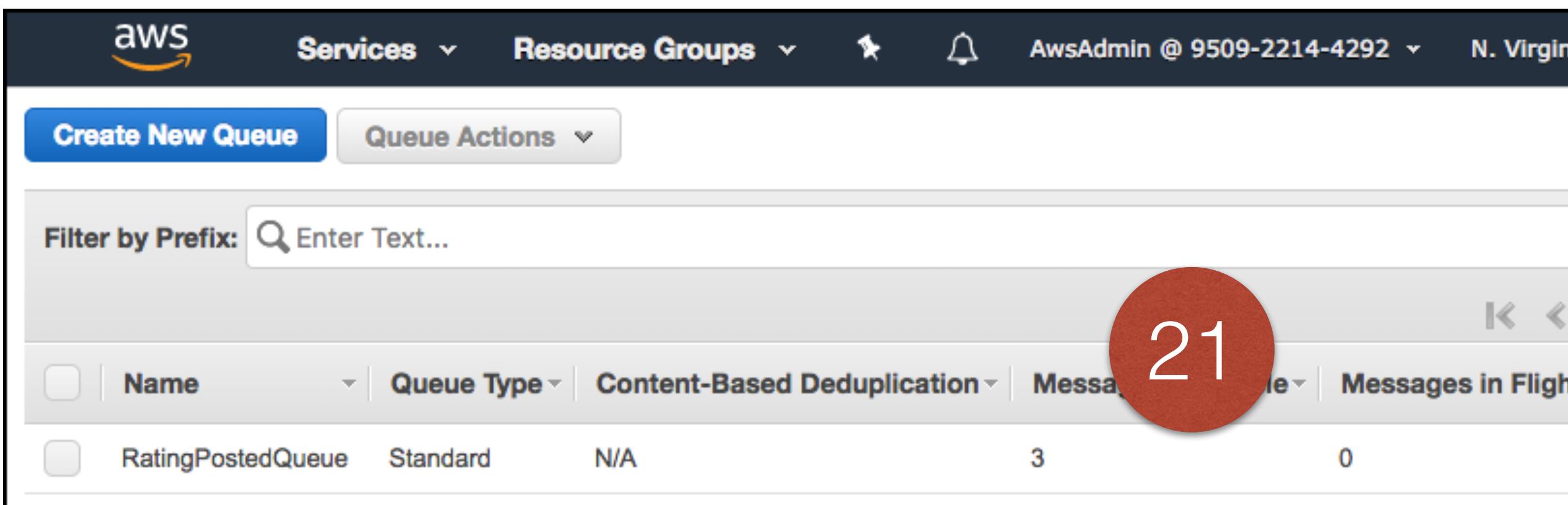
# Bind SQS Queue to SNS Topic as Subscriber

20



The screenshot shows the AWS Lambda 'Execution Result' interface. The status is 'Succeeded'. The logs display the execution of a Lambda function named 'handler'. The log entries show the function starting, processing a message from an SQS queue, validating the event, and then publishing a message to an SNS topic. The message content includes the movie ID and title.

```
Execution Result x +  
Status: Succeeded Max Memo  
Execution Results  
Function Logs:  
START RequestId: 451e2c95-e08f-11e7-be5f-5d750bcdcbf8 Version: $LATEST  
2017-12-14T05:26:00.452Z 451e2c95-e08f-11e7-be5f-5d750bcdcbf8 { function: 'handler',  
  data:  
    { movieId: 'tt0035400',  
      movieTitle: 'Sweater Girl',  
      userEmail: 'dick@gmail.com',  
      rating: '3' } }  
2017-12-14T05:26:00.453Z 451e2c95-e08f-11e7-be5f-5d750bcdcbf8 {"function":"validateEvent","data":{"movieId":"tt0035400","movieTitle":"Sweater Girl","userEmail":"dick@gmail.com","rating":3}}  
2017-12-14T05:26:00.629Z 451e2c95-e08f-11e7-be5f-5d750bcdcbf8 { message: 'Data written.',  
  data:  
    { movieId: { S: 'tt0035400' },  
      movieTitle: { S: 'Sweater Girl' },  
      userEmail: { S: 'dick@gmail.com' },  
      rating: { S: '3' } } }
```



# Bind SQS Queue to SNS Topic as Subscriber

**Code District Aws Class Movie Rater**

Search for a Movie  Search

Movie

Rating

Rater Email

Submit

Your Rating: **22**

CloudWatch > Log Groups > API-Gateway-Execution-Logs\_opna20xd2e/TestWithLogging > 37a749d808e46495a8da1e5

Filter events

Time (UTC +00:00)	Message
2017-12-14 05:38:11	(f91fe9f7-e090-11e7-aa58-052518ddb732) Endpoint request URI: https://lambda.us-east-1.amazonaws.com/functions/cdPostRating
05:38:11	(f91fe9f7-e090-11e7-aa58-052518ddb732) Endpoint request headers: {x-amzn-lambda-integration-tag: f91fe9f7-e090-11e7-aa58-052518ddb732, Authorization: *****, Date: 20171214T053811Z, x-amzn-apigateway-api-id: opna20xd2e, X-Amz-Source-Arn: arn:aws:execute-api:us-east-1:950981384111:opna20xd2e, Accept: application/json, User-Agent: AmazonAPIGateway_opna20xd2e, X-Amz-Security-Token: FQoDYXdzED4aDKVtrp3S1ESTeApEyK3Az94YnP4KFyKRLkogv8Ik1544UKRpAmYX0h1DgVo84RNUWz527lq7MHhJN61qMhJ4+rHkbqCnbnJr3jYXrVCzf0wj8fUUFRSH/x2DsIT/8M9S43FGIH/krh8npVpErVXi88KFqCAgeGx5mrmHEfW+EuNxGAUijc3qAUjq26zk0hbH0QeE6B2cqnjYhN [TRUNCATED]
05:38:11	(f91fe9f7-e090-11e7-aa58-052518ddb732) Endpoint request body after transformations: { "movieId": "", "userEmail": "phil@gmail.com", "rating": "" }
05:38:11	(f91fe9f7-e090-11e7-aa58-052518ddb732) Sending request to https://lambda.us-east-1.amazonaws.com/2015-03-31/functions/opna20xd2e/invocations
05:38:11	(f91fe9f7-e090-11e7-aa58-052518ddb732) Sending request to https://lambda.us-east-1.amazonaws.com/2015-03-31/functions/opna20xd2e/invocations

23

CloudWatch > Log Groups > /aws/lambda/cdPostRating > 2017/12/14/[LATEST]b362741bf5b5495fa9327d7e3f7a2f76

Filter events

Time (UTC +00:00)	Message
2017-12-14 05:38:10	START RequestId: f8973216-e090-11e7-b163-83b00e05ba40 Version: \$LATEST
05:38:11	START RequestId: f8973216-e090-11e7-b163-83b00e05ba40 Version: \$LATEST
05:38:11	2017-12-14T05:38:10.955Z f8973216-e090-11e7-b163-83b00e05ba40 { function: 'handler', data: { movieId: 'tt0058515', movieTitle: 'Rattle of a Simple Man', userEmail: 'phil@gmail.com', rating: '1' } }
05:38:11	2017-12-14T05:38:10.955Z f8973216-e090-11e7-b163-83b00e05ba40 {"function": "validateEvent", "message": "Data written to database"}
05:38:11	2017-12-14T05:38:11.212Z f8973216-e090-11e7-b163-83b00e05ba40 {"function": "validateEvent", "message": "Data written to database"}

24

# Day 2 Evaluation

**Please take the Day 2 Evaluation:**

<https://www.surveymonkey.com/r/CK8K6BJ>