aws workshop studio

Gateway Workshop Build Amazon SNS Integration with AWS SAM and OpenAPI

The Amazon API

▶ Module 1 - Introduction to Amazon

▶ Module 2 - Deploy your first API

▼ Module 3 - API Gateway REST

Introduction

Getting Started

API Gateway

Integrations

Mock

Module Goals

▶ HTTP Integration

AWS Step Functions

Setup your AWS SAM Project

Integration with AWS SAM

Build Amazon SNS

Test the Integration

and OpenAPI

Amazon Kinesis

Amazon DynamoDB

► Amazon EventBridge

► Module 4 - Observability in API

► Module 6 - Enable fine-grained

access control for your APIs

► Module 5 - WebSocket APIs

▶ Private Integration

► Amazon S3

Clean up

Gateway

Clean up

Resources

► AWS Lambda

Amazon SQS

▼ Amazon SNS

with IaC

<

You can integrate an API method directly with an Amazon SNS topic without the need for a compute layer such as Lambda in the middle. This is achieved through the **AWS service integration** type.

The Amazon API Gateway Workshop > Module 3 - API Gateway REST Integrations > Amazon SNS > Build Amazon SNS Integration with AWS SAM and OpenAPI

(2)

Use AWS SAM and OpenAPI to create an API Gateway REST API with a direct Amazon SNS integration

- 1. Using AWS Cloud9 console, return to the root folder module-3/SNS
- 2. This code belongs in your SAM [template file template.yaml

```
Review the code and then copy/paste it into the template.yaml file.
```

```
AWSTemplateFormatVersion: '2010-09-09'
     Transform: 'AWS::Serverless-2016-10-31'
     Description: >
        module3-sns: Sample SAM Template for module3-sns
     Globals:
9
         #Enable Logs
10
11
             MethodSettings:
12
                  ResourcePath: "/*"
13
                   HttpMethod: "*"
                   DataTraceEnabled: True
14
15
                   LoggingLevel: INFO
16
                   MetricsEnabled: True
17
18
     Resources:
19
20
         #Create the SNS Topic
21
         SNSTopicForAPIGW:
22
             Type: AWS::SNS::Topic
23
             Properties:
24
               TopicName: MyAPIGWSNSTopic
25
26
         #Create the IAM role
27
         IAMRoleForSNSIntegration:
28
             Type: AWS::IAM::Role
29
             Properties:
30
                 AssumeRolePolicyDocument:
                     Version: "2012-10-17"
31
32
                     Statement:
33
                       - Effect: Allow
34
                         Principal:
35
                           Service:
36

    apigateway.amazonaws.com

37
                         Action:
                              'sts:AssumeRole'
39
                 Path: /
                 Policies:
                     - PolicyName: PolicyForAPIGWSQSIntegration
41
                       PolicyDocument:
                         Version: "2012-10-17"
43
                         Statement:
                           - Effect: Allow
45
                             Action: 'sns:Publish'
47
                             Resource: !Ref SNSTopicForAPIGW
48
49
         # Create the API Gateway
50
         APIWithSNSIntegration:
             Type: AWS::Serverless::Api
51
52
             Properties:
53
                 StageName: dev
                 OpenApiVersion: 3.0.3
                 DefinitionBody: # an OpenApi definition
55
                     "Fn::Transform":
56
57
                         Name: "AWS::Include"
58
                         Parameters:
                             Location: "openapi.yaml"
59
                 EndpointConfiguration:
                     Type: REGIONAL
61
62
63
     Outputs:
64
         APIGatewayEndpoint:
65
                 Description: API Gateway Endpoint
```

The above template contains a mixture of both AWS SAM and CloudFormation resources. As SAM templates are an extenstion of CloudFormation, its possible to mix resources from both types as upon deployment, the SAM resources are converted to CloudFormation resources. Within this template, we have created an SNS topic, an IAM role that will give API Gateway permission to publish messages to the SNS service and an API Gateway. After a successful deployment, the template will output the API Gateway invoke URL as well as the ARN of the SNS topic.

Fn::Sub: https://\${APIWithSNSIntegration}.execute-api.\${AWS::Region}.amazonaws.com/dev/email-me

Review the code and then copy/paste it into the openapi.yaml file

3. This code belongs in your OpenAPI definition file openapi.yaml

Description: ARN for SNS topic

Fn::Sub: \${SNSTopicForAPIGW}

67

68 69

70 71 SNSTopicARN:

```
openapi: "3.0.1"
        title: "module-3-sns-integration"
        description: "API Gateway example for using SNS as direct service integration"
        version: "1.0"
     paths:
      /email-me:
10
         post:
           consumes:
           - "application/json"
        produces:
14
           - "application/json"
15
           responses:
16
17
              description: "200 response"
18
               schema:
19
                 $ref: "#/definitions/Empty"
20
           x-amazon-apigateway-integration:
21
             uri: "arn:aws:apigateway:${AWS::Region}:sns:path//"
22
             credentials:
              Fn::Sub: ${IAMRoleForSNSIntegration.Arn}
23
24
             httpMethod: "POST"
25
             responses:
              default:
26
27
                 statusCode: "200"
28
             requestTemplates:
29
              application/json:
                 Fn::Sub: "Action=Publish&TopicArn=$util.urlEncode('${SNSTopicForAPIGW}')&Message=$util.urlEncode($input.body)"
30
31
             requestParameters:
              integration.request.header.Content-Type: "'application/x-www-form-urlencoded'"
32
33
             passthroughBehavior: "never"
34
             type: "aws"
35
```

Lines (19-33) are particulary important as these are what create the direct service integration for SNS. The x-amazon-apigateway-integration [2] is an OpenAPI extension that allows us to integrate our APIs to AWS backends.

the SNS Publish API expects. Within this template, we also define the intended action (which is to call the Publish API [2]) as well as the topic we want to publish the

```
(i) Info: The SNS Publish API call expects a POST request where the relevant parameters are passed in the body as form data.
Line (29) defines our integration request template, which transforms the incoming HTTP request, including the request body into the URL encoded format that
```

configurations that you want SAM to have in order to get the guided deployment. You can configure it as below.

message to, and finally what message we want to send. Line (31) sets the Content-Type header that is sent to the backend. As the SNS endpoint expects the request to be URL encoded, we set this value to 'application/x-www-form-urlencoded'.

```
Deploy the project
1. To deploy the API Gateway and SNS resources to your AWS account, run the following commands from the application root module-3/SNS, where the
```

template.yaml file for the sample application is located: sam build && sam deploy --guided

The first time that you run the sam deploy --guided command, AWS SAM starts an AWS CloudFormation deployment. In this case, you need to say what are the

Stack Name: module-3-sns-integration

- AWS Region: Put the chosen region to run the workshop. e.g. us-east-1 🗖 Confirm changes before deploy: Y
- Allow SAM CLI IAM role creation: Y Disable rollback: N
- Save arguments to configuration file: Y SAM configuration file and SAM configuration environment leave blank

Outputs

Configuring SAM deploy

```
Looking for config file [samconfig.toml] : Not found
           Setting default arguments for 'sam deploy'
           Stack Name [sam-app]: module-3-sns-integration
           AWS Region [eu-west-1]:
           #Shows you resources changes to be deployed and require a 'Y' to initiate deploy
           Confirm changes before deploy [y/N]: Y
           #SAM needs permission to be able to create roles to connect to the resources in your template
           Allow SAM CLI IAM role creation [Y/n]: Y
           #Preserves the state of previously provisioned resources when an operation fails
           Disable rollback [y/N]: N
           Save arguments to configuration file [Y/n]: Y
           SAM configuration file [samconfig.toml]:
           SAM configuration environment [default]:
2. After configuring the deployment, AWS SAM will display assets that will be created. But first, it will automatically upload the template to a temporary bucket it
```

creates. Then, it will ask you to confirm the changes. Type y to confirm.

```
APIGatewayEndpoint
 Key
 Description
                           API Gateway Endpoint
                           https://k
                                                 j.execute-api.eu-west-1.amazonaws.com/dev/email-me
 Value
                           SNSTopicARN
 Key
 Description
                           ARN for SNS topic
                                                                     B9:MyAPIGWSNSTopic
 Value
                           arn:aws:sns:eu-west-1:02
3. Once the deployment has been successful, you will see an 'Outputs' section that contains the API Gateway invoke URL and the ARN of the SNS topic you just
 created - take note of the SNS topic ARN as you will need this for the next step.
```

- 4. With our SNS topic created, you now need to create a subscription in order to receive the messages sent to this topic. For the purposes of this demo, you will use your email address as the notification endpoint. Return to your Cloud9 environment and paste in the below command, being sure to replace SNS Topic ARN with the ARN provided in the Outputs section of the SAM deployment and YOUR EMAIL HERE with an email address you have ownership of.
- aws sns subscribe --topic-arn <!SNS Topic ARN!> --protocol email --notification-endpoint <!YOUR EMAIL HERE!>

5. After running the above command, you should get the response 'Pending Confirmation'. To confirm the subscription, navigate to the email address you used to

```
subscribe to the topic and look for an email from 'no-reply@sns.amazonaws.com ''. From this email, click 'Confirm subscription'.
AWS Notification - Subscription Confirmation
```

You have chosen to subscribe to the topic:

o AWS Notifications <no-reply@sns.amazonaws.com>

Thursday 23 March 2023 at 11:06

B9:MyAPIGWSNSTopic arn:aws:sns:eu-west-1:0 To confirm this subscription, click or visit the link below (If this was in error no action is necessary):

Confirm subscription

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

advertising. To accept or decline all non-essential cookies, choose "Accept" or "Decline." To make more detailed choices, choose "Customize."

Customize

aws workshop studio (3)

1. Navigate to the API Gateway console and select the API that has been created. It will be named **module-3-sns-integration**.

Update documentation

aws

2. Select the POST method from the resources section to the left of the screen to bring up the API request flow window.

:bfsb6no6l4/*/POST/email-me

The Amazon API Gateway Workshop > Module 3 - API Gateway REST Integrations > Amazon SNS > **Test the Integration**

Now the API has been deployed and subscription confirmed, its time to test the integration.

Gateway Workshop

The Amazon API

<

Test the Integration

3. Click in Integration Request and then Edit.

API Gateway > APIs > Resources - module-3-sns-integration (bfsb6no6l4)

Resources

⊡ ∠

Create resource

- /email-me

<u>POST</u>

Test SNS Integration using API Gateway console

/email-me - POST - Method execution

🗖 arn:aws:execute-api:us-east-1:8

Integration request settings

AWS

Content-Type

Default (29 seconds)

URL path parameters (0)

- Introduction
- Getting Started ▶ Module 1 - Introduction to Amazon
- **API Gateway** ► Module 2 - Deploy your first API
- with IaC ▼ Module 3 - API Gateway REST
- Integrations
- Module Goals
- Mock
- ► HTTP Integration AWS Lambda
- Amazon SQS Amazon SNS

► AWS Step Functions

Build Amazon SNS Integration with AWS SAM and OpenAPI

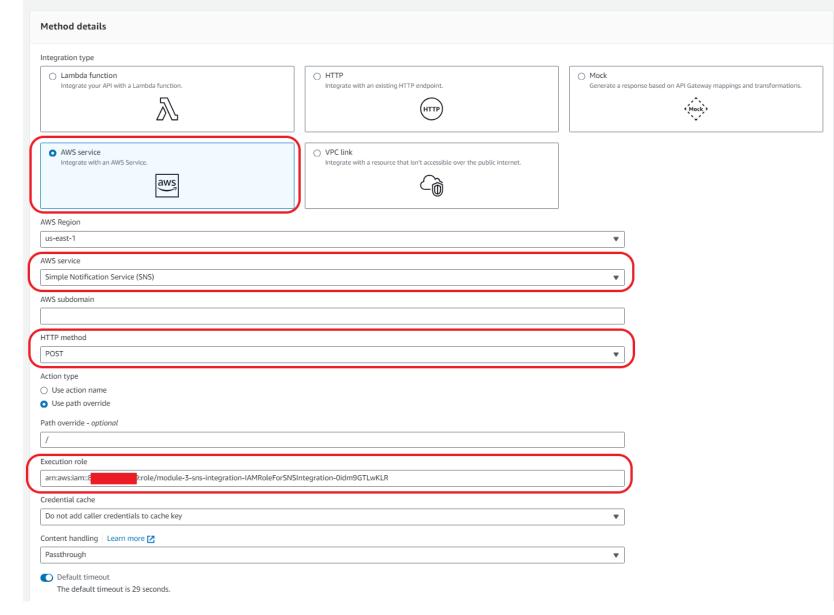
Setup your AWS SAM Project

- Test the Integration
- ► Amazon DynamoDB
- Amazon Kinesis
- ► Amazon EventBridge
- ▶ Private Integration
- ► Amazon S3
- Clean up
- ► Module 4 Observability in API Gateway
- ► Module 5 WebSocket APIs
- ► Module 6 Enable fine-grained access control for your APIs
- Clean up

Resources

API Gateway > APIs > Resources - module-3-sns-integration (bfsb6no6l4) > Edit integration request Edit integration request

4. Review the configuration details for integration. Once done, click **Cancel**.



skwx43

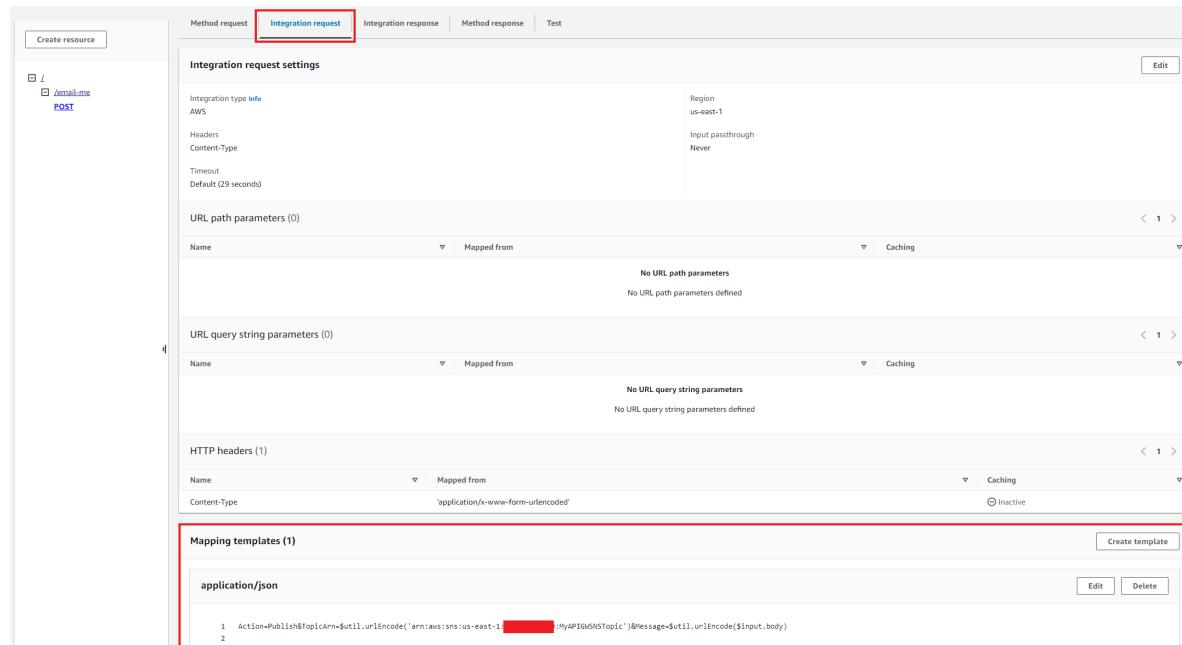
us-east-1

No URL path parameters

Input passthrough

Integration response Method response Test

5. Scroll down, and look for the Mapping templates, see the input transformation request template:

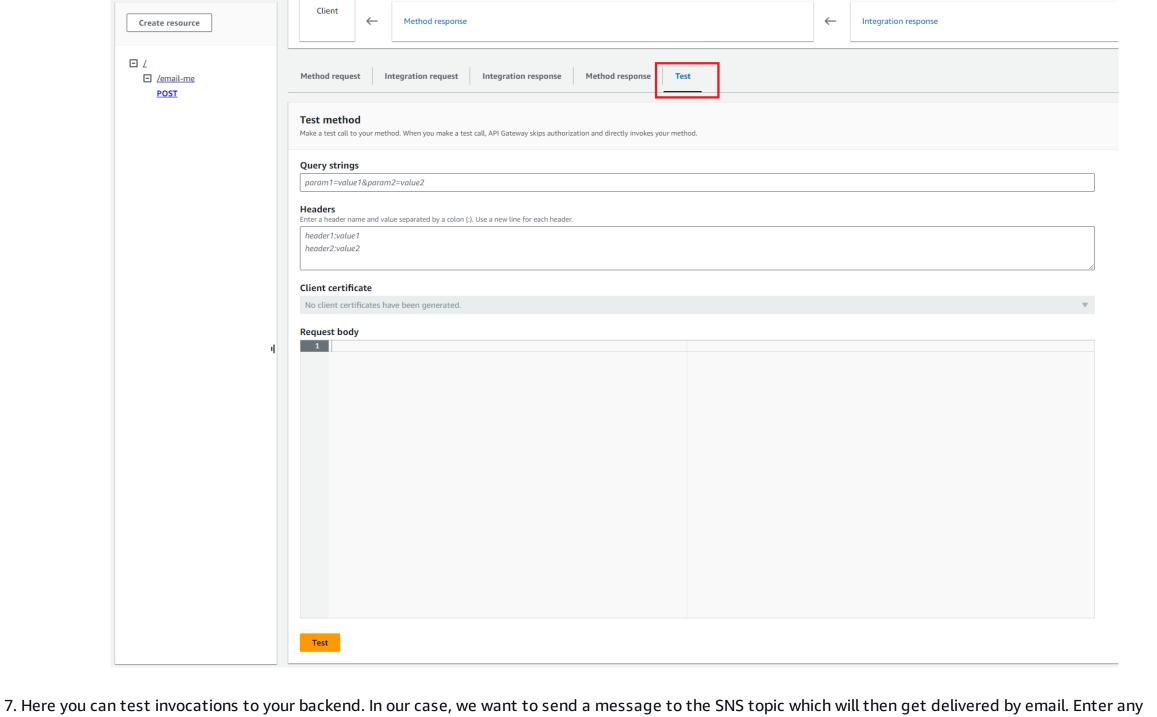


Marning! Testing methods with the API Gateway console may result in changes to resources that cannot be undone. Testing a method with the API Gateway console is the

message you want within the Request Body section.

6. Scroll up, select the Test tab.

same as calling the method outside of the API Gateway console. For example, if you use the API Gateway console to call a method that deletes an API's resources, if the method call is successful, the API's resources will be deleted.



□ / Method respon **POST** Test method

Make a test call to your method. When you make a test call, API Gateway skips authorization and directly invokes your method

```
Query strings
                                                     param1=value1&param2=value2
                                                     header1:value1
                                                      header2:value2
                                                    Client certificate
                                                     No client certificates have been generated.
                                                     ⊗1 Hello from the API Gateway workshop
                                                      Test
8. Hit the Test button. From this, you should receive a Status code of 200 meaning the message was successfully delivered to the SNS topic.
```

 \Box \angle

Request body

Create resource

■ Hello from the API Gateway workshop

```
/email-me - POST method test results
                                                                                             Latency
                                                                                                                                              Status
                                                                                             80
                                                                                                                                              200
                                           Response headers
                                             "Content-Type": "application/json",
                                             "X-Amzn-Trace-Id": "Root=1-6531042f-6f2281a5dbec23fad98fa6e2"
                                           Execution log for request 0d421c1e-398d-4fa5-a6d1-2749440847c1
                                           Thu Oct 19 10:25:51 UTC 2023 : Starting execution for request: 0d421c1e-398d-4fa5-a6d1-2749440847c1
                                           Thu Oct 19 10:25:51 UTC 2023 : HTTP Method: POST, Resource Path: /email-me
                                           Thu Oct 19 10:25:51 UTC 2023 : Method request path: \{\}
                                           Thu Oct 19 10:25:51 UTC 2023 : Method request query string: {}
9. After a short period of time, you should receive an email from no-reply@sns.amazonaws.com 🔼 that contains the message sent from the API Gateway.
             AWS Notification Message
```

AWS Notifications <no-reply@sns.amazonaws.com>

Hello from API Gateway SNS module

To: O Burns, Zac

1. Open a new terminal window in your AWS Cloud9 environment. 2. Copy the below cURL command and paste it into the terminal window, replacing <api-id> with your API ID and <region> with the region where your API is deployed. You can also get the full URL from the Outputs section from your SAM deployment. You can also find the full invoke URL in the API Gateway console by

Test the deployed API using cURL

navigating to **Stages > dev**. 1 curl -X POST \

-d 'Hello from API Gateway workshop'

'https://<api-id>.execute-api.<region>.amazonaws.com/dev/email-me' \ -H 'Content-Type: application/json' \

```
(i) Info: We need to ensure we send the 'application/json' Content-Type header due to the fact our Request Body passthrough settings are set to Never meaning the request
        will not be sent to the backend unless the request is using this supported Content Type.
3. The Response Body output should be:
```

```
2 "PublishResponse": {
3 "PublishResult": {
                 "MessageId": "f9f3e5e7-58b4-5e4d-84bc-b97923a2ee97",
      },
"ResponseMetadata": {
    "RequestId": "d26
                 "SequenceNumber": null
                 "RequestId": "d26b09c6-99e0-5213-8e39-5558e2f612c9"
10
```

4. You will receive an email from no-reply@sns.amazonaws.com that contains the message sent from the API Gateway.

```
Find the payload logs before and after the request and response transformations in CloudWatch.
```

We use essential cookies and similar tools that are necessary to provide our site and services. We use performance cookies to collect anonymous statistics, so we can understand

Select your cookie preferences

Accept