Build AWS Mock Integration with AWS SAM and OpenAPI

You can integrate an API method without the need for an integration backend using **Mock integration**.

Use AWS SAM and OpenAPI to create an API Gateway REST API with Mock integration

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

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

```yaml
AWSTemplateFormatVersion: '2010-09-09'
Transform: 'AWS::Serverless-2016-10-31'
Description: >
    module3-mock-rest-api: Sample SAM Template for mock-rest-api

Globals:

    # Enable Logs
    Api:
        MethodSettings:
          - ResourcePath: "/*"
            HttpMethod: "*"
            DataTraceEnabled: True
            LoggingLevel: INFO
            MetricsEnabled: True

Resources:

    # Example API Gateway
    MockApi:
        Type: AWS::Serverless::Api
        Properties:
          EndpointConfiguration:
            Type: "REGIONAL"
          StageName: "dev"
          DefinitionBody: # an OpenApi definition
            'Fn::Transform':
              Name: 'AWS::Include'
              Parameters:
                Location: './openapi.yaml'

Outputs:
    MockApi:
        Description: "API Gateway endpoint URL to call Mock resource method api"
        Value: !Sub "https://${MockApi}.execute-api.${AWS::Region}.amazonaws.com/dev/"
```

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

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

```yaml
1   openapi: "3.0.1"
2   info:
3     title: "module-3-mock-integration"
4     version: "1.0"
5   paths:
6     /pets:
7       get:
8         x-amazon-apigateway-integration:
9           responses:
10            default:
11              statusCode: "200"
12          requestTemplates:
13            application/json: "{\"statusCode\": 200}"
14          passthroughBehavior: "when_no_match"
15          type: "mock"
16  components: {}
```

These lines **(5 to 15)** define a single endpoint for the API, `/pets`, with an HTTP `GET` method. The endpoint is a mock endpoint, meaning it does not actually execute any code but simply returns a static response.

The x-amazon-apigateway-integration **(line 8)** section provides information about how the API Gateway should integrate with the mock endpoint. Specifically:

- `responses` **(line 9)** defines the expected HTTP response codes and their associated responses. In this case, there is only a single response, with a status code of 200 and no body.
- `requestTemplates` **(line 12)** defines the input to the mock endpoint. In this case, it is a JSON object with a single key, `statusCode`, whose value is also 200.
- `passthroughBehavior` **(line 14)** field is set to **"when_no_match"**, which means that if no response template matches the status code, the response from the backend is passed through to the client. It specifies how the API Gateway should handle requests that do not match any integration. In this case, it will pass the request through to the backend, since this is a mock endpoint.
- `type` **(line 15)** specifies the integration type. In this case, it is `mock`.

Here are some relevant AWS documentation links:

- OpenAPI Specification
- API Gateway OpenAPI extensions

> **Save your files!**
> Cloud9 does not automatically save your files so be sure to save them

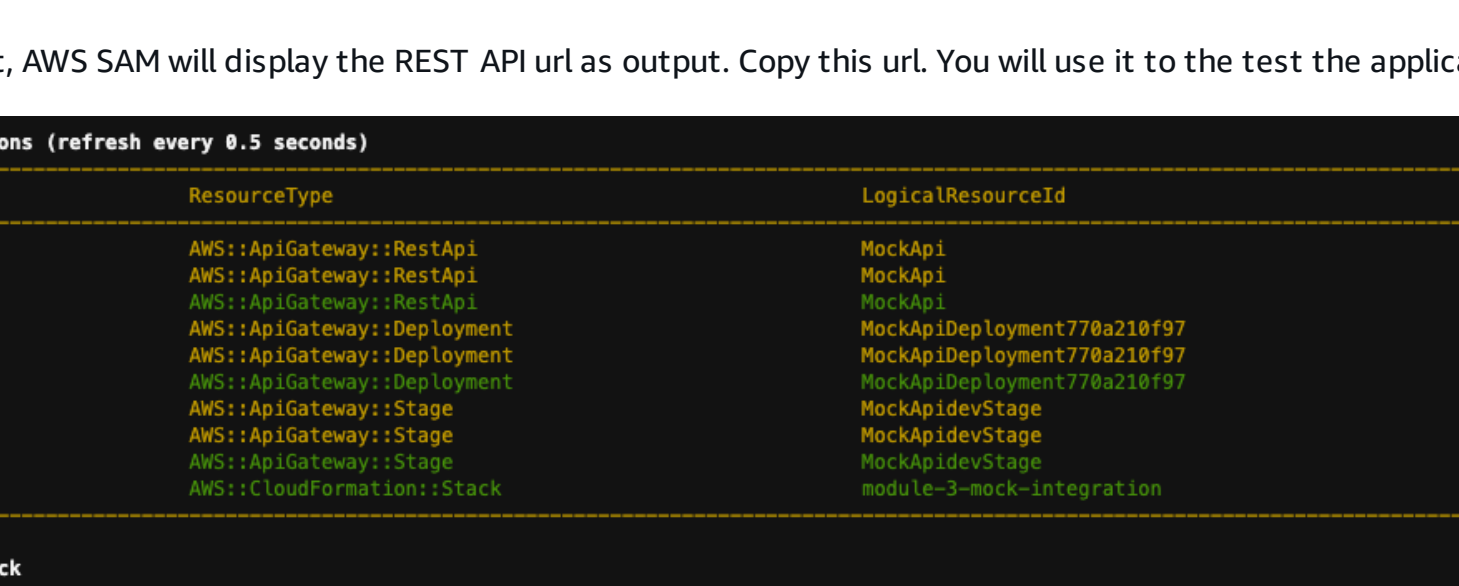
Deploy the project

1. To deploy the Amazon API Gateway to your AWS account, run the following commands from the application root `module-3/mock`, where the `template.yaml` file for the sample application is located:

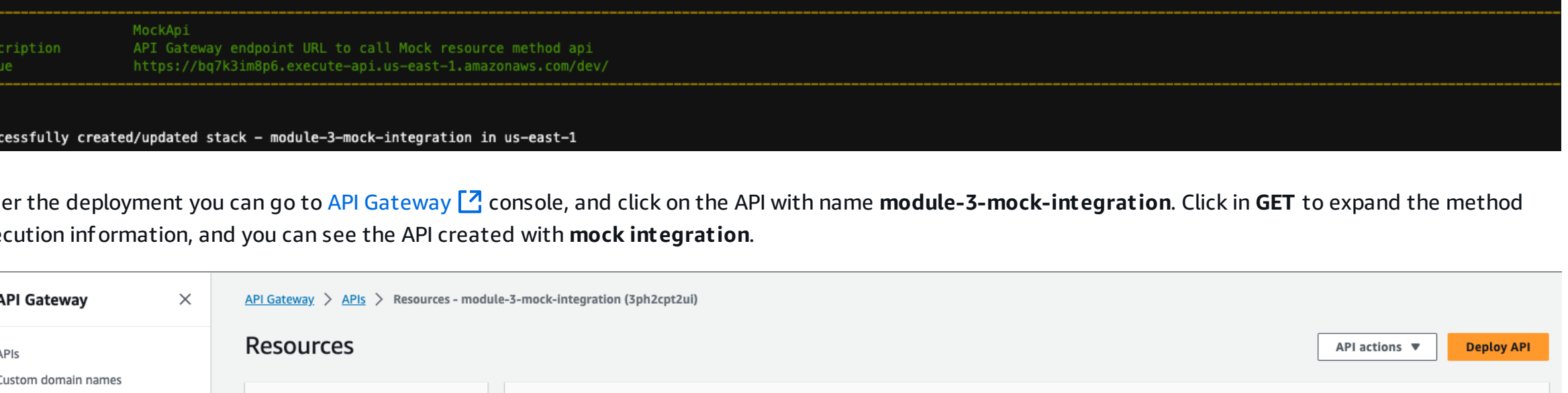
```
sam build && sam deploy --guided
```

The first time that you run the `sam deploy --guided` command, AWS SAM starts an AWS CloudFormation deployment. In this case, you needed to say what are the configurations that you want SAM to have in order to get the guided deployment. You can configure as it is above.

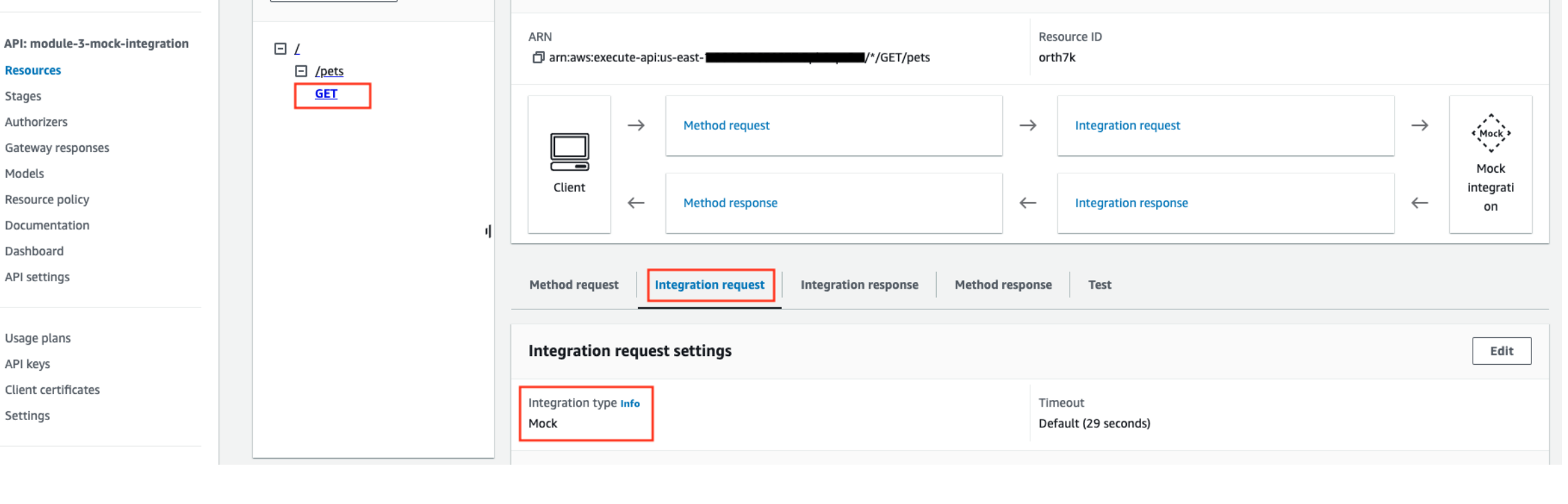
- **Stack Name**: `module-3-mock-integration`
- **AWS Region**: Put the chosen region to run the workshop. **e.g.** `us-east-1`
- **Confirm changes before deploy**: `Y`
- **Allow SAM CLI IAM role creation**: `Y`
- **Disable rollback**: `N`
- **Save arguments to configuration file**: `Y`
- **SAM configuration file** and **SAM configuration environment** leave blank


Setting default arguments for 'sam deploy'
===
Stack Name [sam-app]: module-3-mock-integration
AWS Region [us-east-1]: us-east-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]:

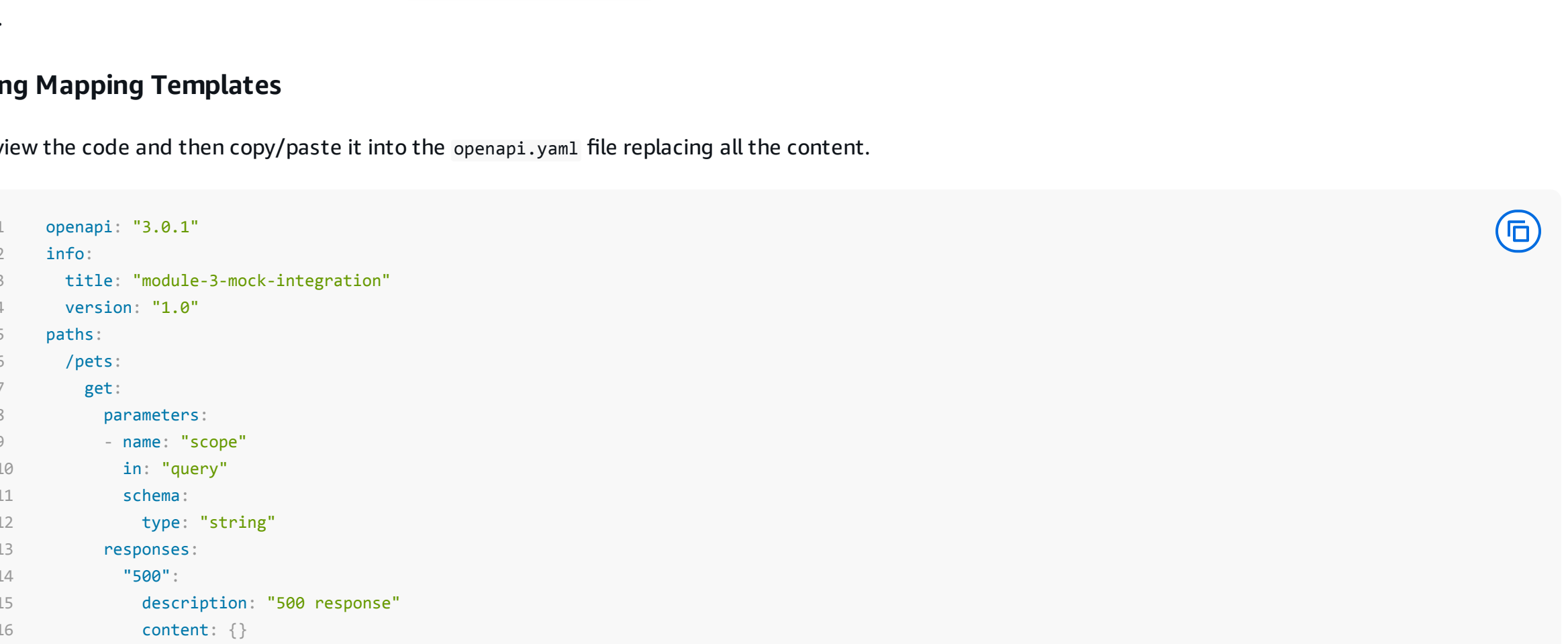
2. After configuring the deployment, AWS SAM will display assets that will be created. But first, it will automatically upload the template to a temporary bucket it creates. Then, it will ask you to confirm the changes. Type `y` to confirm.


CloudFormation stack changeset
Operation LogicalResourceId ResourceType Replacement
+ Add MockApiDeployment770a210f97 AWS::ApiGateway::Deployment N/A
+ Add MockApidevStage AWS::ApiGateway::Stage N/A
+ Add MockApi AWS::ApiGateway::RestApi N/A
Changeset created successfully. arn:aws:cloudformation:us-east-1:[redacted]:changeSet/samcli-deploy1679407039/e6d8529b-7308-48d1-8eb5-628450790321
Previewing CloudFormation changeset before deployment
Deploy this changeset? [y/N]: y

3. After completing the deployment, AWS SAM will display the REST API url as output. Copy this url. You will use it to the test the application in the next step.


CloudFormation events from stack operations (refresh every 0.5 seconds)
ResourceStatus ResourceType LogicalResourceId ResourceStatusReason
CREATE_IN_PROGRESS AWS::ApiGateway::RestApi MockApi -
CREATE_IN_PROGRESS AWS::ApiGateway::RestApi MockApi Resource creation Initiated
CREATE_COMPLETE AWS::ApiGateway::RestApi MockApi -
CREATE_IN_PROGRESS AWS::ApiGateway::Deployment MockApiDeployment770a210f97 -
CREATE_IN_PROGRESS AWS::ApiGateway::Deployment MockApiDeployment770a210f97 Resource creation Initiated
CREATE_COMPLETE AWS::ApiGateway::Deployment MockApiDeployment770a210f97 -
CREATE_IN_PROGRESS AWS::ApiGateway::Stage MockApidevStage -
CREATE_IN_PROGRESS AWS::ApiGateway::Stage MockApidevStage Resource creation Initiated
CREATE_COMPLETE AWS::ApiGateway::Stage MockApidevStage -
CREATE_COMPLETE AWS::CloudFormation::Stack module-3-mock-integration -
CloudFormation outputs from deployed stack
Outputs
Key MockApi
Description API Gateway endpoint URL to call Mock resource method api
Value https://bq7k3im8p6.execute-api.us-east-1.amazonaws.com/dev/
Successfully created/updated stack - module-3-mock-integration in us-east-1

4. After the deployment you can go to API Gateway console, and click on the API with name **module-3-mock-integration**. Click in **GET** to expand the method execution information, and you can see the API created with **mock integration**.


API Gateway
APIs
Custom domain names
VPC links
API: module-3-mock-integration
Resources
Stages
Authorizers
Gateway responses
Models
Resource policy
Documentation
Dashboard
API settings
Usage plans
API keys
Client certificates
Settings
API Gateway > APIs > Resources - module-3-mock-integration (3ph2cpt2ui)
Resources
API actions
Deploy API
Create resource
/pets
GET
/pets - GET - Method execution
Update documentation
Delete
ARN
Resource ID
orth7k
Client
Method request
Method response
Integration request
Integration response
Mock integration
Method request
Integration request
Integration response
Method response
Test
Integration request settings
Edit
Integration type Info
Mock
Timeout
Default (29 seconds)

Configuring the Mock integration created before

To update the stack on AWS, you'll need to run `sam build && sam deploy` each time you make changes to your files. To set up a **mock integration**, follow the steps below.

Adding Mapping Templates

1. Review the code and then copy/paste it into the `openapi.yaml` file replacing all the content.

```yaml
1   openapi: "3.0.1"
2   info:
3     title: "module-3-mock-integration"
4     version: "1.0"
5   paths:
6     /pets:
7       get:
8         parameters:
9         - name: "scope"
10          in: "query"
11          schema:
12            type: "string"
13        responses:
14          "500":
15            description: "500 response"
16            content: {}
17          "200":
18            description: "200 response"
19            content:
20              application/json:
21                schema:
22                  $ref: "#/components/schemas/Empty"
23        x-amazon-apigateway-integration:
24          responses:
25            default:
26              statusCode: "200"
27          requestTemplates:
28            application/json: "{\n  #if( $input.params('scope') == \"internal\" )\n\
29              \    \"statusCode\": 200\n  #else\n    \"statusCode\": 500\n  #end\n}"
30          passthroughBehavior: "when_no_match"
31          type: "mock"
32  components:
33    schemas:
34      Empty:
35        title: "Empty Schema"
36        type: "object"
```

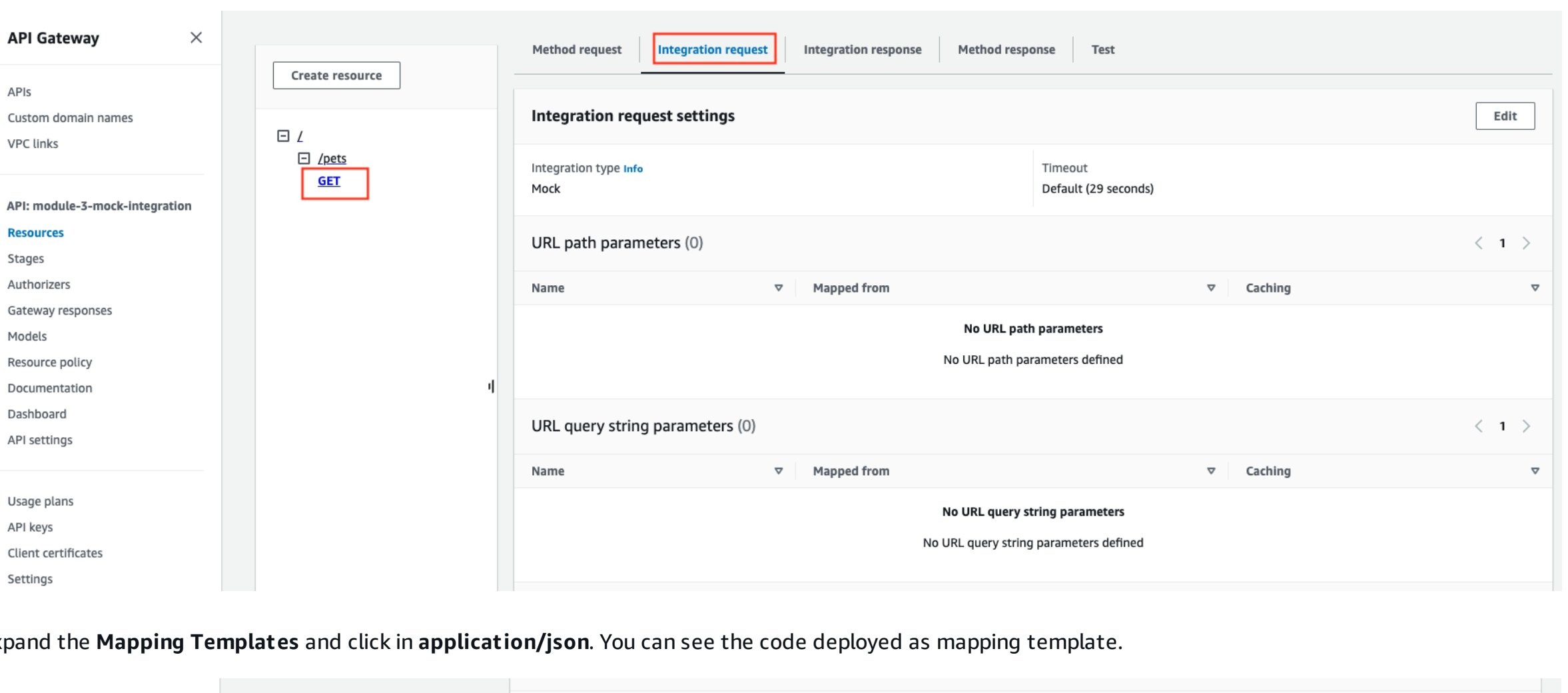
The code above is an OpenAPI specification in version 3.0.1 that describes a mock REST API. The API has one endpoint at the path `/pets` that accepts a `GET` request. The request can have a query parameter named `scope` of type string. The response to the request can be a 200 **OK** or a 500 **Internal Server Error**, depending on the value of the `scope` parameter.

If the `scope` parameter is "internal", the response will be a 200 **OK** with an empty body. If the `scope` parameter is anything other than "internal", the response will be a 500 **Internal Server Error** with an empty body.

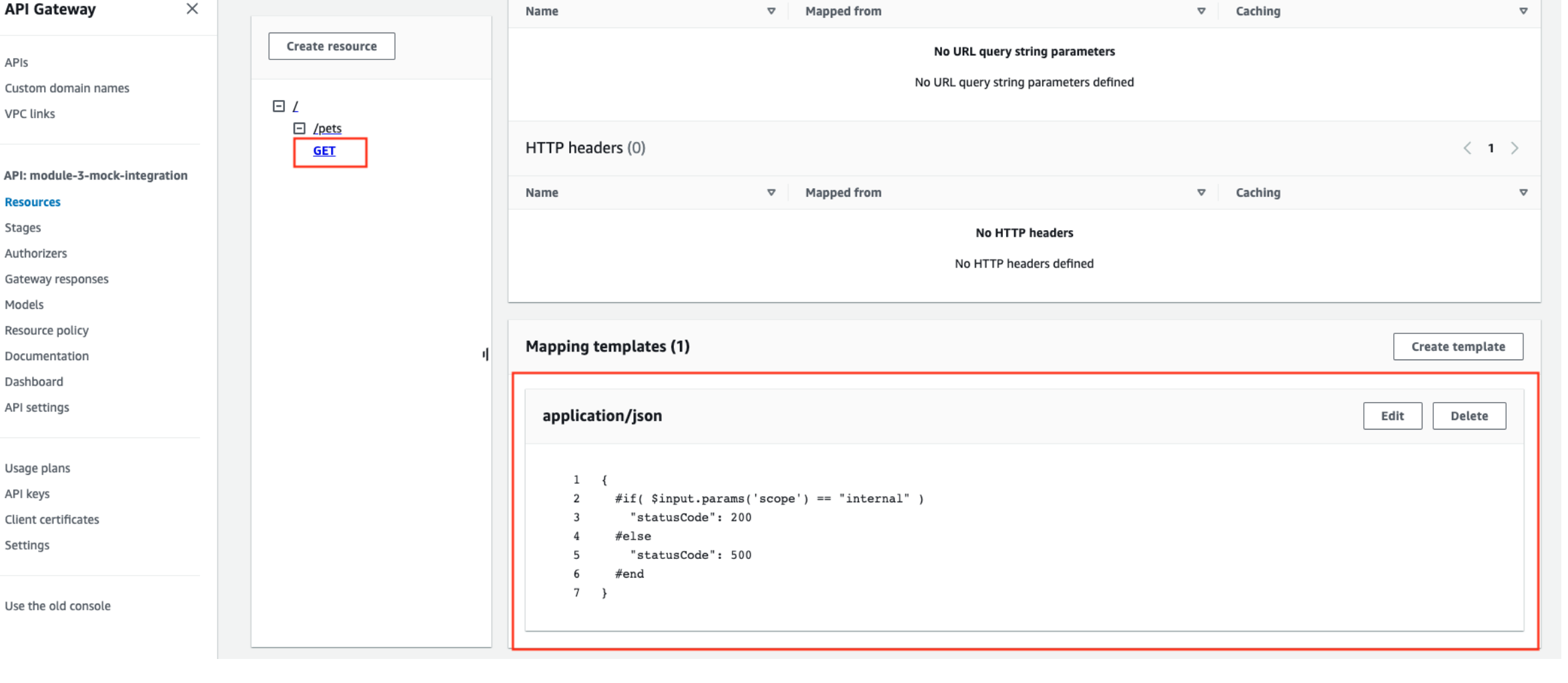
The API is integrated with Amazon API Gateway using the x-amazon-apigateway-integration **(line 23)** field. This integration specifies a mock integration type, which returns a predefined response without calling any backend service. The integration also sets the default response status code to 200 and defines a request template that conditionally sets the status code based on the `scope` parameter value.

Finally, the code defines a schema named `Empty` in the `components` section, which describes an empty JSON object. This schema is used to specify the content of the 200 OK response.

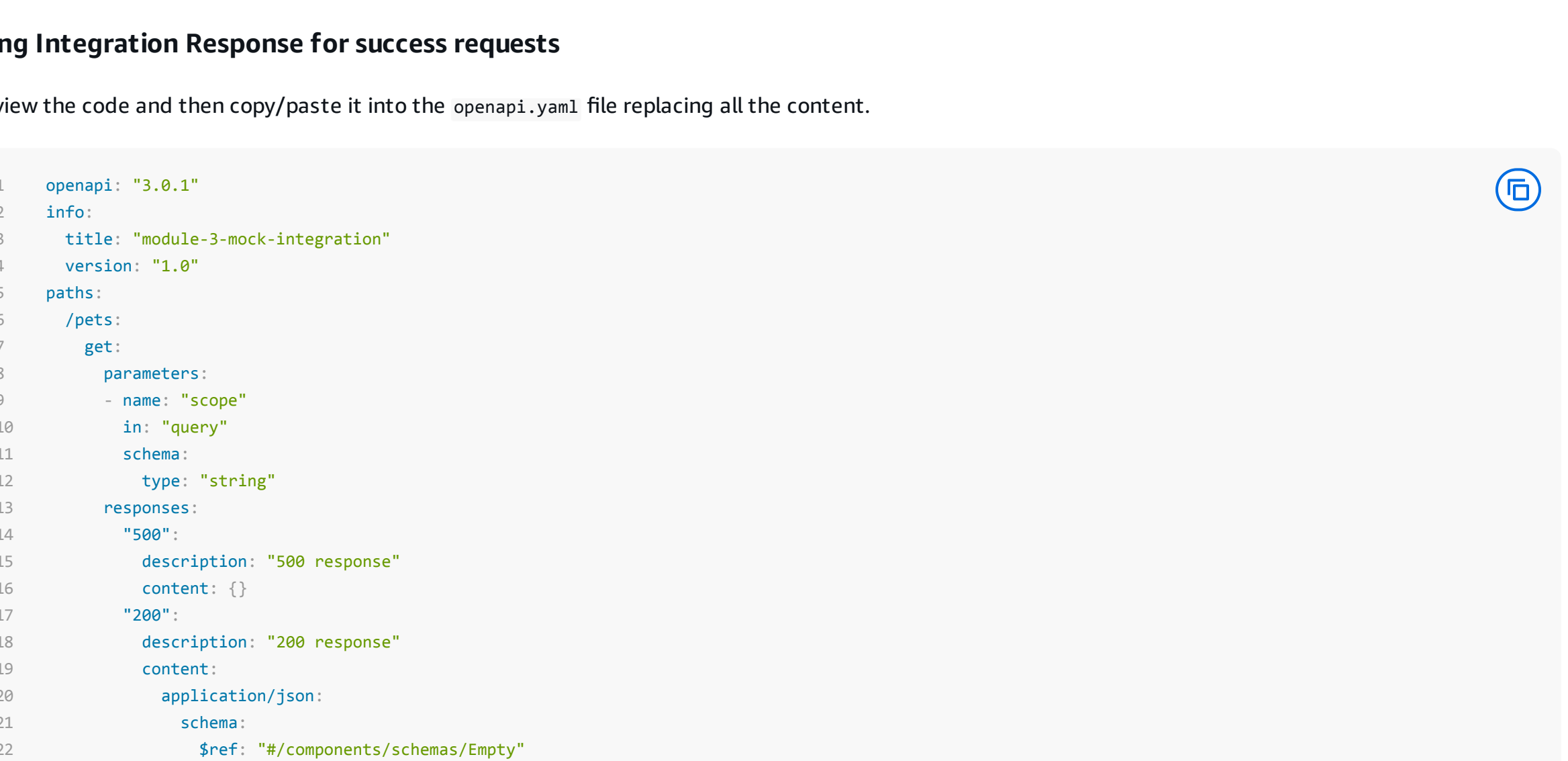
2. **Run** `sam build && sam deploy` on the Cloud9 console to update the Stack.
3. After update, go to the API again you can see the `scope` **Query String** created.


API Gateway
APIs
Custom domain names
VPC links
API: module-3-mock-integration
Resources
Stages
Authorizers
Gateway responses
Models
Resource policy
Documentation
Dashboard
API settings
Usage plans
API keys
Client certificates
Settings
Create resource
/pets
GET
Method request
Integration request
Integration response
Method response
Test
Method request settings
Edit
Authorization
NONE
API key required
False
Request validator
None
SDK operation name
Generated based on method and path
Request paths (0)
Name
Caching
No request paths
No request paths defined
URL query string parameters (1)
Name
Required
Caching
scope
False
Inactive
HTTP request headers (0)

4. Now click on the **Integration Request** on the same screen.


API Gateway
APIs
Custom domain names
VPC links
API: module-3-mock-integration
Resources
Stages
Authorizers
Gateway responses
Models
Resource policy
Documentation
Dashboard
API settings
Usage plans
API keys
Client certificates
Settings
Create resource
/pets
GET
Method request
Integration request
Integration response
Method response
Test
Integration request settings
Edit
Integration type Info
Mock
Timeout
Default (29 seconds)
URL path parameters (0)
Name
Mapped from
Caching
No URL path parameters
No URL path parameters defined
URL query string parameters (0)
Name
Mapped from
Caching
No URL query string parameters
No URL query string parameters defined

5. Expand the **Mapping Templates** and click in **application/json**. You can see the code deployed as mapping template.


API Gateway
APIs
Custom domain names
VPC links
API: module-3-mock-integration
Resources
Stages
Authorizers
Gateway responses
Models
Resource policy
Documentation
Dashboard
API settings
Usage plans
API keys
Client certificates
Settings
Use the old console
Create resource
/pets
GET
Name
Mapped from
Caching
No URL query string parameters
No URL query string parameters defined
HTTP headers (0)
No HTTP headers
No HTTP headers defined
Mapping templates (1)
Create template
application/json
Edit
Delete
1 {
2 #if($input.params('scope') == "internal")
3 "statusCode": 200
4 #else
5 "statusCode": 500
6 #end
7 }

Adding Integration Response for success requests

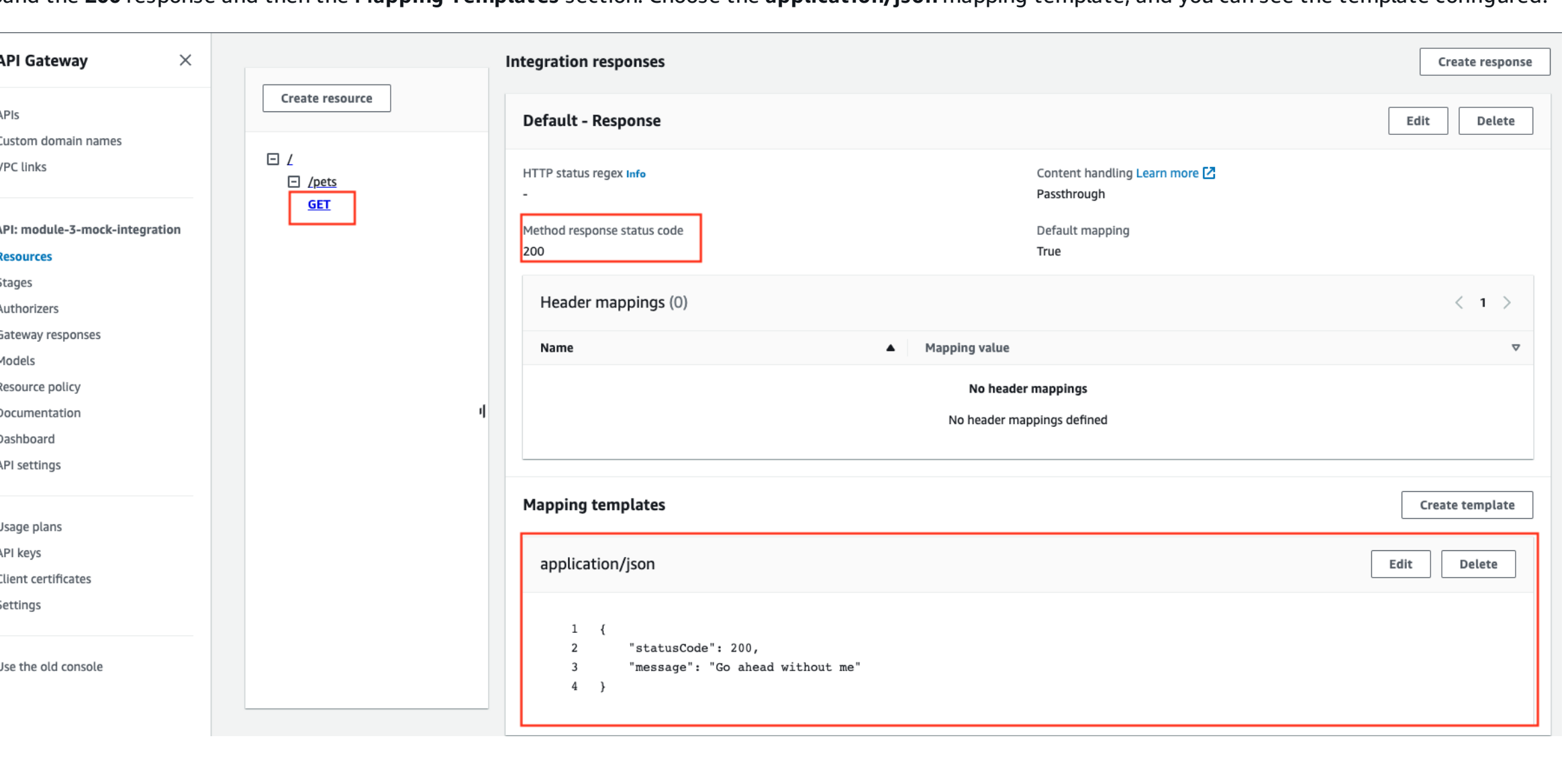
1. Review the code and then copy/paste it into the `openapi.yaml` file replacing all the content.

```yaml
1   openapi: "3.0.1"
2   info:
3     title: "module-3-mock-integration"
4     version: "1.0"
5   paths:
6     /pets:
7       get:
8         parameters:
9         - name: "scope"
10          in: "query"
11          schema:
12            type: "string"
13        responses:
14          "500":
15            description: "500 response"
16            content: {}
17          "200":
18            description: "200 response"
19            content:
20              application/json:
21                schema:
22                  $ref: "#/components/schemas/Empty"
23        x-amazon-apigateway-integration:
24          responses:
25            default:
26              statusCode: "200"
27              responseTemplates:
28                application/json: "{\n    \"statusCode\": 200,\n    \"message\": \"\
29                  Go ahead without me\"\n}"
30          requestTemplates:
31            application/json: "{\n  #if( $input.params('scope') == \"internal\" )\n\
32              \    \"statusCode\": 200\n  #else\n    \"statusCode\": 500\n  #end\n}"
33          passthroughBehavior: "when_no_match"
34          type: "mock"
35  components:
36    schemas:
37      Empty:
38        title: "Empty Schema"
39        type: "object"
```

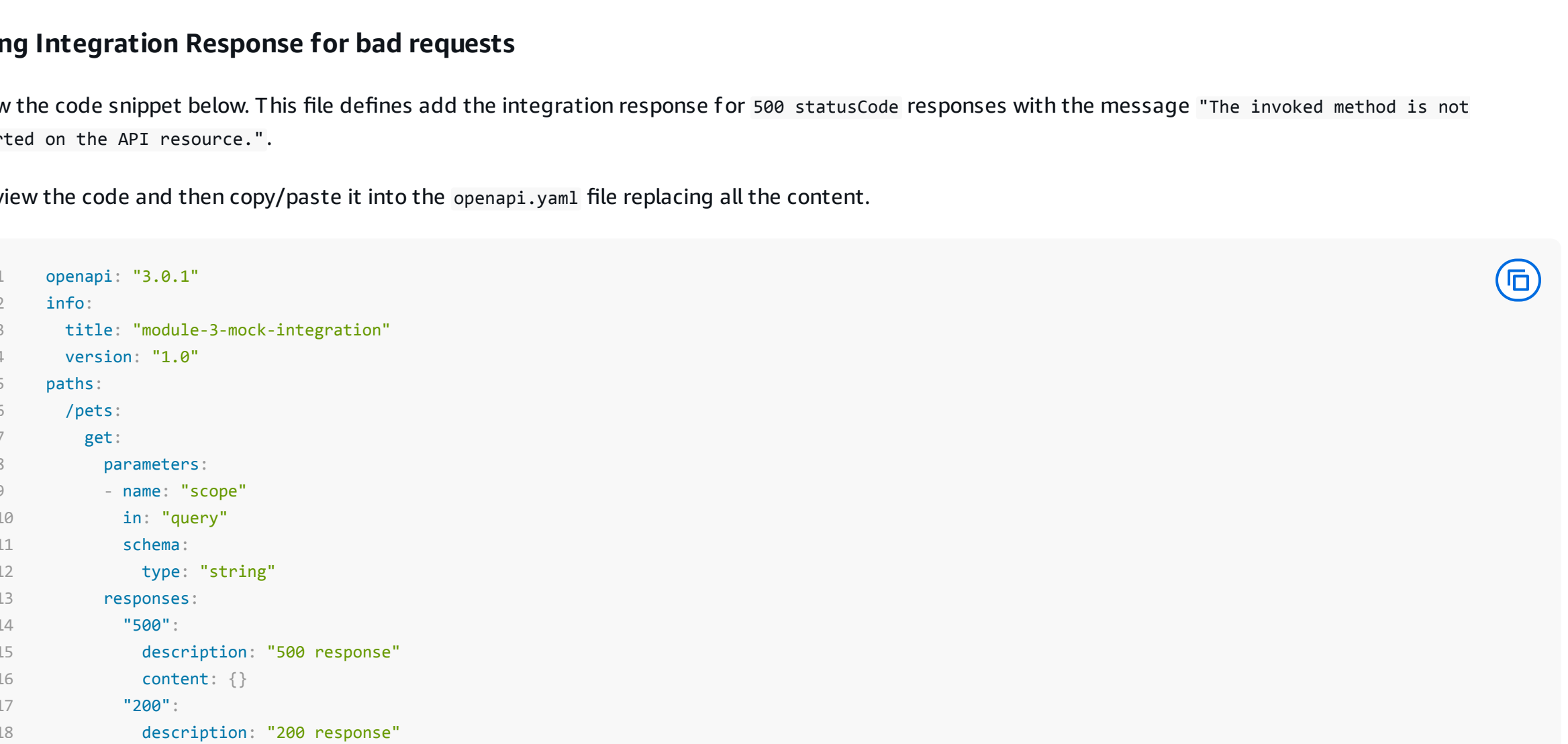
This template has an additional field in the x-amazon-apigateway-integration **(line 23)** section. Specifically, it adds a `responseTemplates` field that specifies a template for the response body. This template sets a message property with the value **"Go ahead without me"**, and a `statusCode` property with the value 200.

This means that when the API receives a request to the `/pets` endpoint, and the scope parameter is not "internal", the API will respond with a 200 `OK` status code and a JSON body that includes the message property set to **"Go ahead without me"**.

2. **Run** `sam build && sam deploy` on the Cloud9 console to update the Stack.
3. After the update, go to the API again on the `GET` method and click on **Integration Response**.


API Gateway
APIs
Custom domain names
VPC links
API: module-3-mock-integration
Resources
Stages
Authorizers
Gateway responses
Models
Resource policy
Documentation
Dashboard
API settings
Usage plans
API keys
Client certificates
Settings
Use the old console
API Gateway > APIs > Resources - module-3-mock-integration (3ph2cpt2ui)
Resources
API actions
Deploy API
Create resource
/pets
GET
/pets - GET - Method execution
Update documentation
Delete
ARN
arn:aws:execute-api:us-east-1:964612631677:3ph2cpt2ui/*/GET/pets
Resource ID
orth7k
Client
Method request
Method response
Integration request
Integration response
Mock integration
Method request
Integration request
Integration response
Method response
Test
Integration responses
Create response
Default - Response
Edit
Delete
HTTP status regex Info
Content handling Learn more
Passthrough
Method response status code
Default mapping

4. Expand the **200** response and then the **Mapping Templates** section. Choose the **application/json** mapping template, and you can see the template configured.


API Gateway
APIs
Custom domain names
VPC links
API: module-3-mock-integration
Resources
Stages
Authorizers
Gateway responses
Models
Resource policy
Documentation
Dashboard
API settings
Usage plans
API keys
Client certificates
Settings
Use the old console
Create resource
/pets
GET
Integration responses
Create response
Default - Response
Edit
Delete
HTTP status regex Info
Content handling Learn more
Passthrough
Method response status code
200
Default mapping
True
Header mappings (0)
Name
Mapping value
No header mappings
No header mappings defined
Mapping templates
Create template
application/json
Edit
Delete
1 {
2 "statusCode": 200,
3 "message": "Go ahead without me"
4 }

Adding Integration Response for bad requests

Review the code snippet below. This file defines add the integration response for `500 statusCode` responses with the message `"The invoked method is not supported on the API resource."`.

1. Review the code and then copy/paste it into the `openapi.yaml` file replacing all the content.

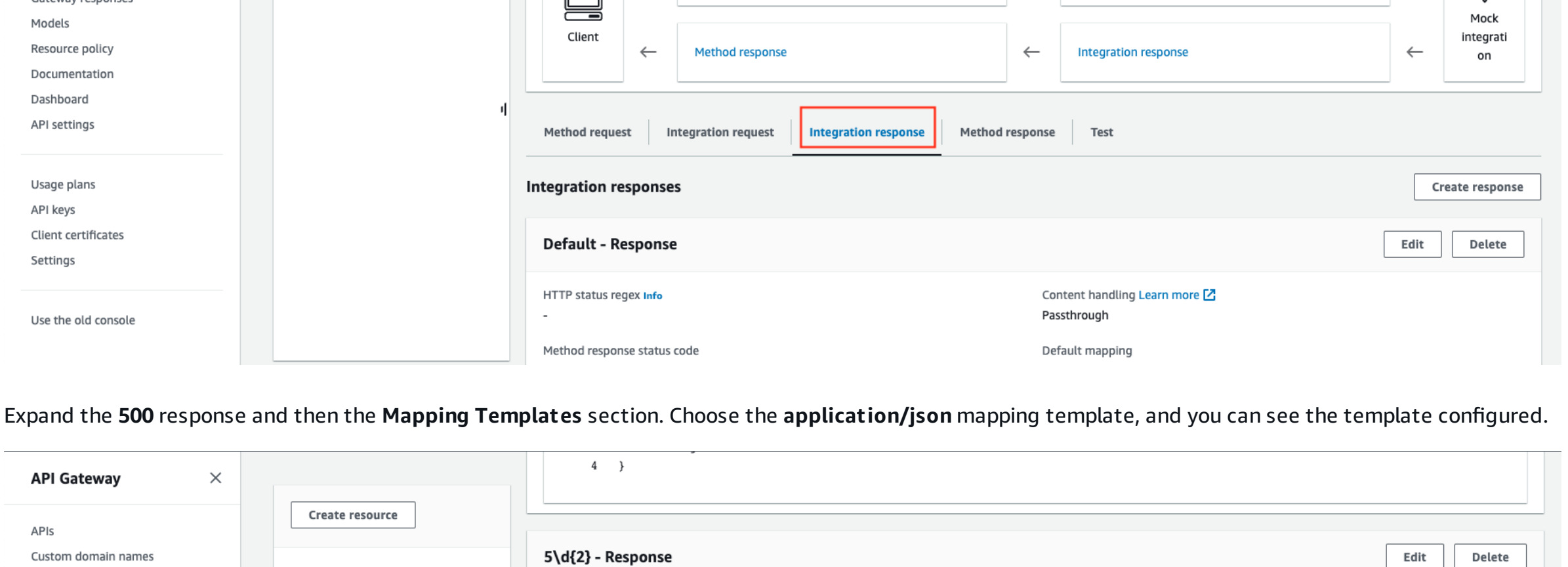
```yaml
1   openapi: "3.0.1"
2   info:
3     title: "module-3-mock-integration"
4     version: "1.0"
5   paths:
6     /pets:
7       get:
8         parameters:
9         - name: "scope"
10          in: "query"
11          schema:
12            type: "string"
13        responses:
14          "500":
15            description: "500 response"
16            content: {}
17          "200":
18            description: "200 response"
19            content:
20              application/json:
21                schema:
22                  $ref: "#/components/schemas/Empty"
23        x-amazon-apigateway-integration:
24          responses:
25            default:
26              statusCode: "200"
27              responseTemplates:
28                application/json: "{\n    \"statusCode\": 200,\n    \"message\": \"\
29                  Go ahead without me\"\n}"
30            "5\\d{2}":
31              statusCode: "500"
32              responseTemplates:
33                application/json: "{\n    \"statusCode\": 500,\n    \"message\": \"\
34                  The invoked method is not supported on the API resource.\"\n}"
35          requestTemplates:
36            application/json: "{\n  #if( $input.params('scope') == \"internal\" )\n\
37              \    \"statusCode\": 200\n  #else\n    \"statusCode\": 500\n  #end\n}"
38          passthroughBehavior: "when_no_match"
39          type: "mock"
40  components:
41    schemas:
42      Empty:
43        title: "Empty Schema"
44        type: "object"
```

The `responses` **(line 24)** field specifies the possible HTTP responses for the endpoint. In this template, it defines two response templates: the `default` response, which is used when the status code doesn't match any of the other response templates, and a specific response for all status codes in the 500 range.

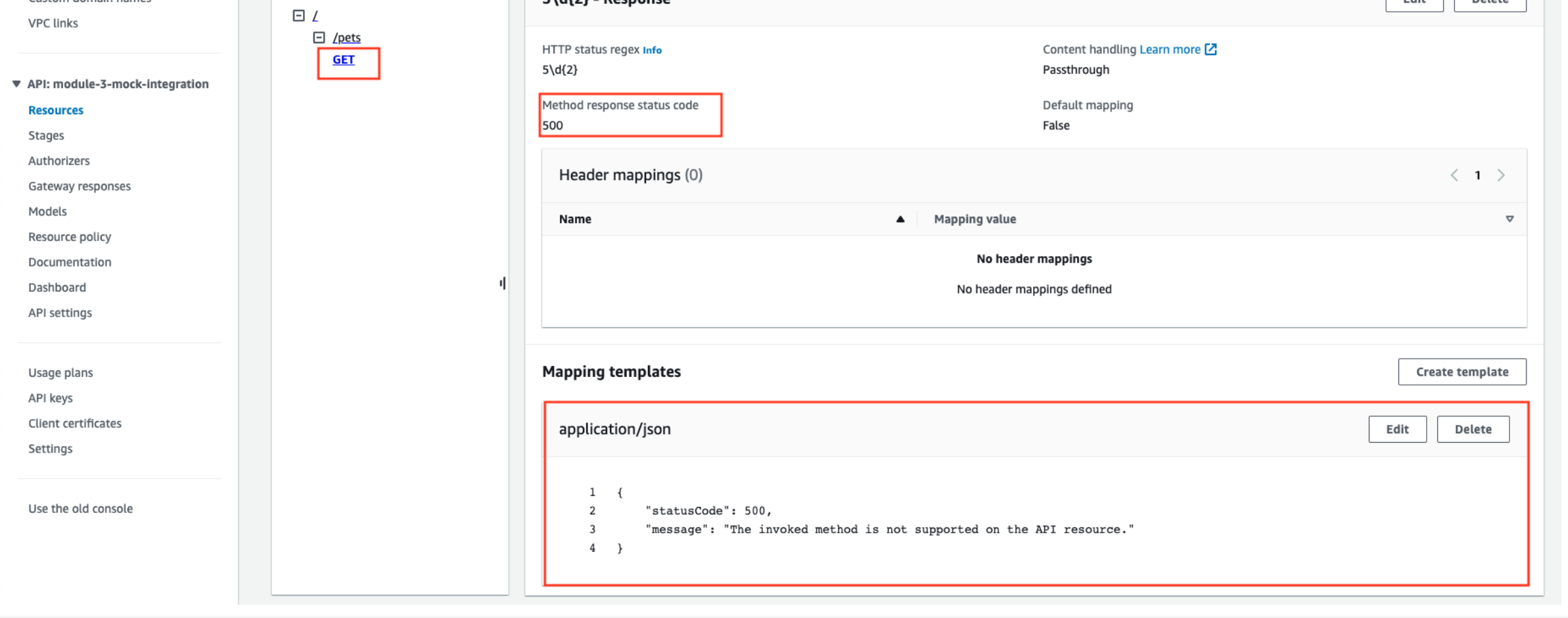
The first response template **(default)** specifies a status code of 200 and returns a JSON message with a **"statusCode"** and **"message"** field.

The second response template specifies a regular expression pattern `5\\d{2}` that matches any status code in the 500 range (e.g., 500, 501, 502, etc.). The template sets the status code to 500 and returns a JSON message (**"The invoked method is not supported on the API resource."**) indicating that the invoked method is not supported on the API resource.

2. **Run** `sam build && sam deploy` on the Cloud9 console to update the Stack.
3. After the update, go to the API again on the `GET` method and click on **Integration Response**.


API Gateway
APIs
Custom domain names
VPC links
API: module-3-mock-integration
Resources
Stages
Authorizers
Gateway responses
Models
Resource policy
Documentation
Dashboard
API settings
Usage plans
API keys
Client certificates
Settings
Use the old console
API Gateway > APIs > Resources - module-3-mock-integration (3ph2cpt2ui)
Resources
API actions
Deploy API
Create resource
/pets
GET
/pets - GET - Method execution
Update documentation
Delete
ARN
arn:aws:execute-api:us-east-1:964612631677:3ph2cpt2ui/*/GET/pets
Resource ID
orth7k
Client
Method request
Method response
Integration request
Integration response
Mock integration
Method request
Integration request
Integration response
Method response
Test
Integration responses
Create response
Default - Response
Edit
Delete
HTTP status regex Info
Content handling Learn more
Passthrough
Method response status code
Default mapping

4. Expand the **500** response and then the **Mapping Templates** section. Choose the **application/json** mapping template, and you can see the template configured.


API Gateway
APIs
Custom domain names
VPC links
API: module-3-mock-integration
Resources
Stages
Authorizers
Gateway responses
Models
Resource policy
Documentation
Dashboard
API settings
Usage plans
API keys
Client certificates
Settings
Use the old console
Create resource
/pets
GET
5\d{2} - Response
Edit
Delete
HTTP status regex Info
5\d{2}
Content handling Learn more
Passthrough
Method response status code
500
Default mapping
False
Header mappings (0)
Name
Mapping value
No header mappings
No header mappings defined
Mapping templates
Create template
application/json
Edit
Delete
1 {
2 "statusCode": 500,
3 "message": "The invoked method is not supported on the API resource."
4 }

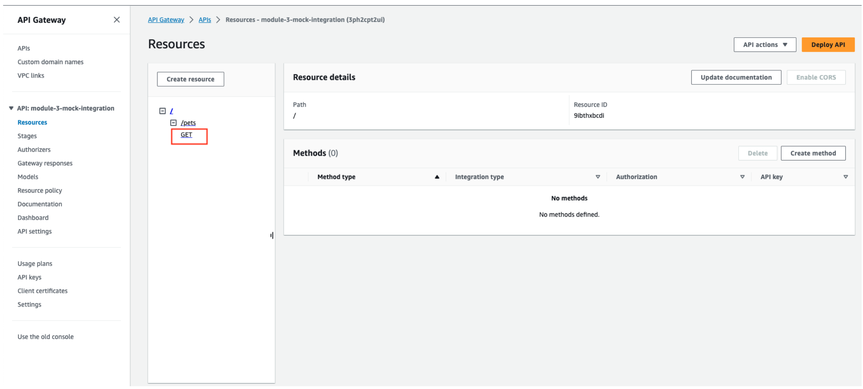
Test Integration

Testing the project

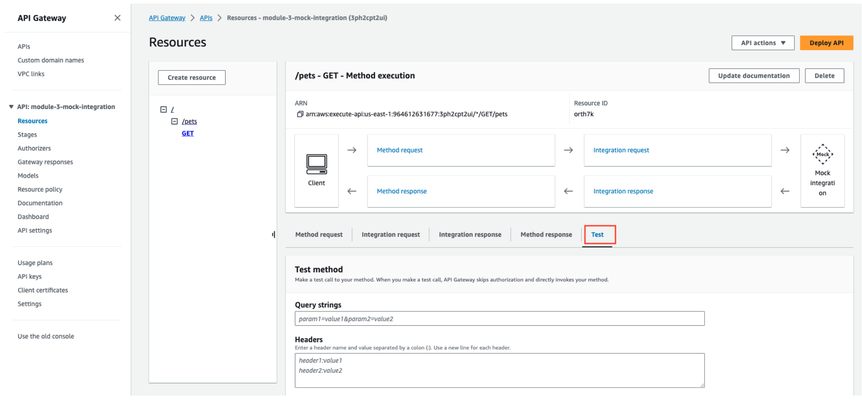
After you create your API Gateway REST API with Mock integration, you can test the API Gateway.

Test the deployed API Gateway using API Gateway console

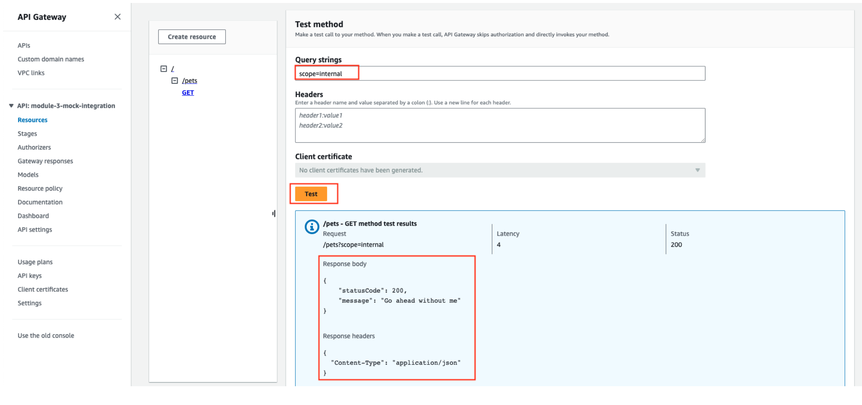
1. Open the Amazon API Gateway console and sign in.
2. Choose your REST API named, `module-3-mock-integration`.
3. In the **Resources** pane, you can select the method you want to test. Click on the `GET` method.



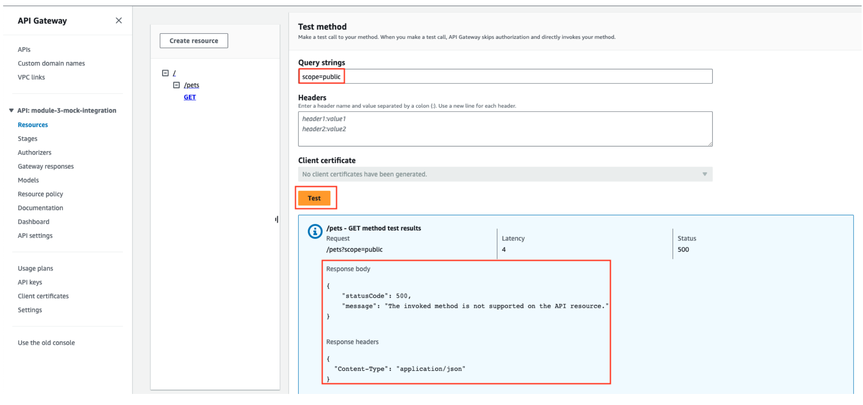
4. In the **Method Execution** pane, in the **Client** box, choose **TEST**.



5. Enter `scope=internal` under Query Strings. Choose **Test**. The test result shows:



6. Enter `scope=public` under Query Strings or leave it blank. Choose **Test**. The test result shows:



You can also return headers in a mock integration response by first adding a header to the method response and then setting up a header mapping in the integration response. In fact, this is how the API Gateway console enables CORS support by returning CORS required headers.

Test the deployed API using cURL

- Open a new terminal window in your AWS Cloud9 environment.
- Copy the following 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 may have copied this URL in from the CloudFormation output in the last step. You can also find the full invoke URL in the API Gateway console by navigating to **Stages > dev**.

```
curl -X GET \
  'https://<api-id>.execute-api.<region>.amazonaws.com/dev/pets' \
  -G -d 'scope=internal'
```

The **Response Body** output should be:

```
{
    "statusCode": 200,
    "message": "Go ahead without me"
}
```