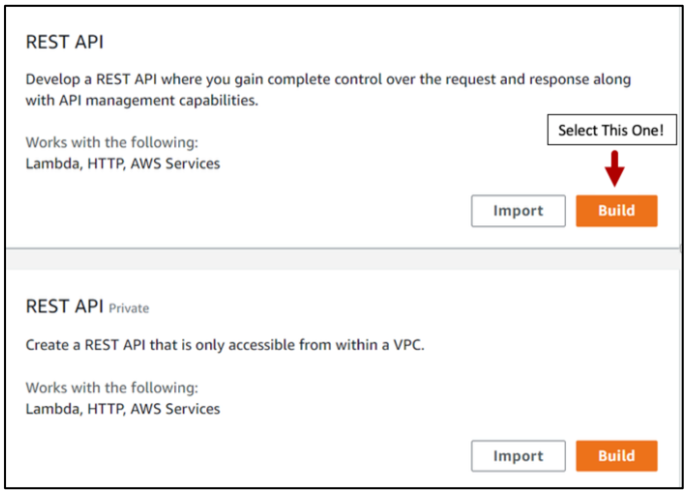
CS 470 Project One Guide

APIs to Lambda

- Please be careful with the case of the API resources. REST is a case-sensitive protocol, and the wrong case will prevent your application from working correctly.
- After every mapping of an API to a Lambda, you will be asked to OK the addition to the permission of the Lambda. Click OK. This allows each individual API you map to a Lambda to have access to that specific Lambda.

Part One – The Questions API

1. Navigate to the **API Gateway** in the AWS Console. See Module Four Assignment Two for a refresher.
2. Click the orange **Create API** button, then click **Build a REST API**.
   - Do not click the REST API marked "private".



REST API

Develop a REST API where you gain complete control over the request and response along with API management capabilities.

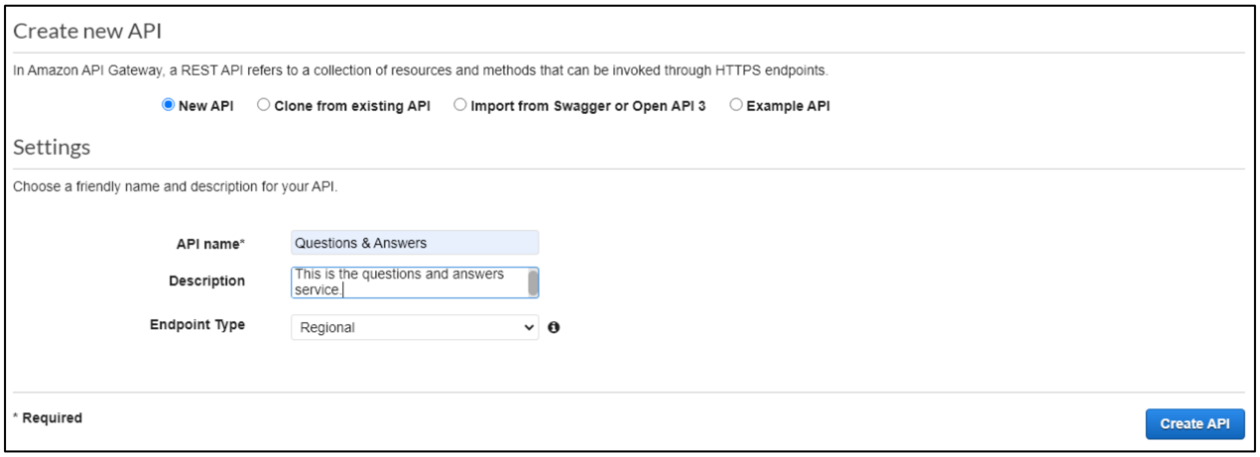
Works with the following:
Lambda, HTTP, AWS Services

REST API Private

Create a REST API that is only accessible from within a VPC.

Works with the following:
Lambda, HTTP, AWS Services

3. Your settings should be **REST**, **New API**, and an **API name** of "Questions & Answers".



Create new API

In Amazon API Gateway, a REST API refers to a collection of resources and methods that can be invoked through HTTPS endpoints.

[✓ New API] Clone from existing API Import from Swagger or Open API 3 Example API

Settings

Choose a friendly name and description for your API.

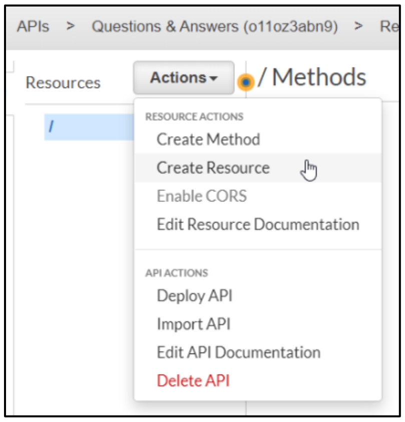
API name*	Questions & Answers
Description	This is the questions and answers service.
Endpoint Type	Regional

* Required

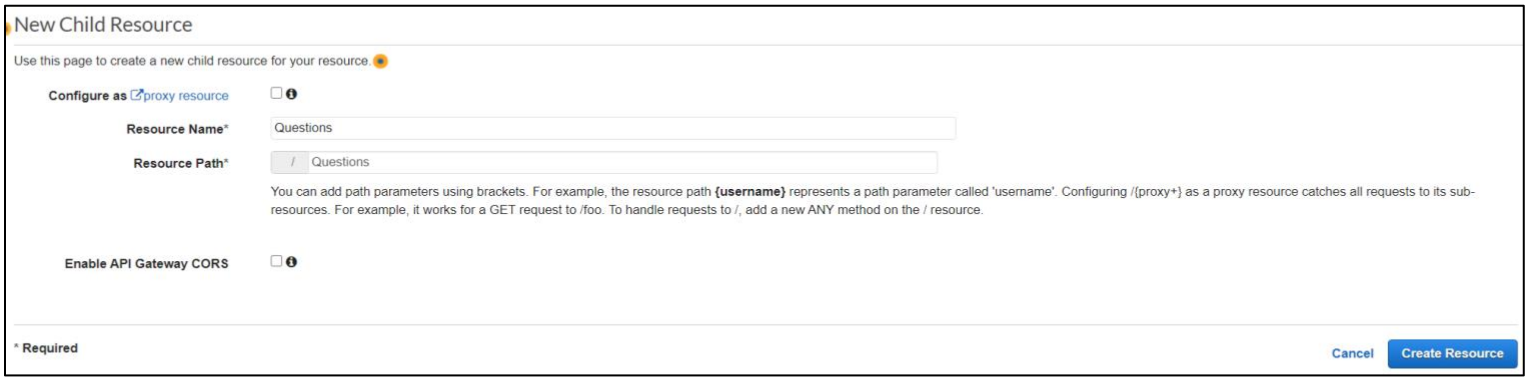
4. Click the **Create API** button.

Questions Resource

1. Click **Actions** drop-down menu and select **Create Resource**.



2. Create a resource named "Questions", and make sure the resource path has a capitalized "Q".
3. Leave the check boxes un-checked and click **Create Resource**.



New Child Resource

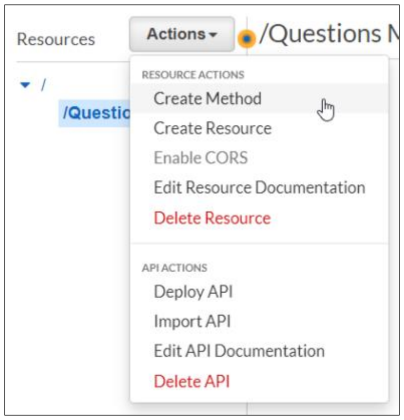
Use this page to create a new child resource for your resource.

Configure as proxy resource	☐
Resource Name*	Questions
Resource Path*	/ Questions
Enable API Gateway CORS	☐

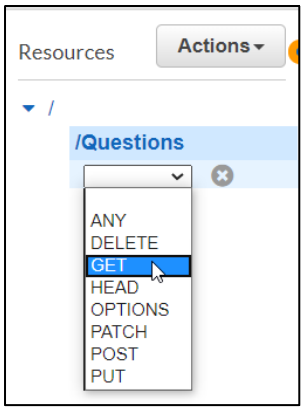
You can add path parameters using brackets. For example, the resource path **{username}** represents a path parameter called 'username'. Configuring /{proxy+} as a proxy resource catches all requests to its sub-resources. For example, it works for a GET request to /foo. To handle requests to /, add a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

* Required

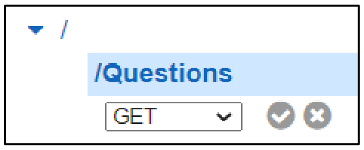
4. With **/Questions** selected, click **Create Method** in the **Actions** drop-down menu.



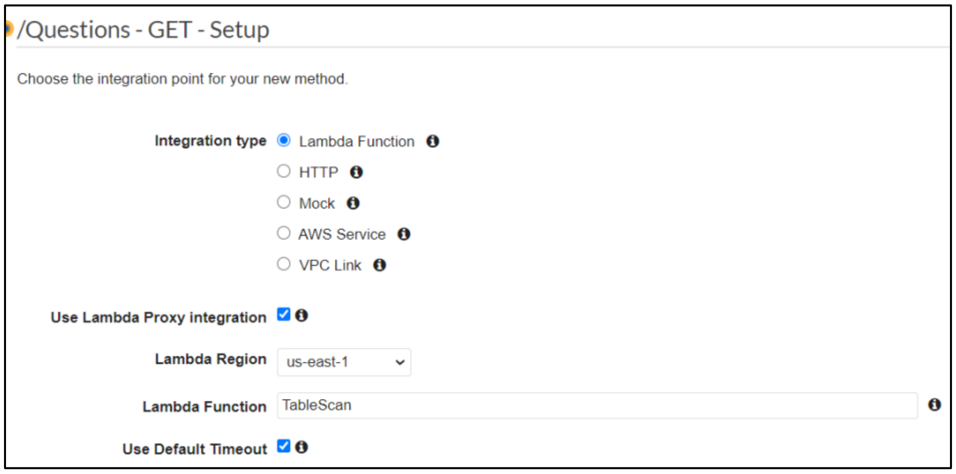
5. Select the GET method in the new drop-down box that is created.



6. Click the checkmark symbol that appears between the method drop-down menu and the X.



7. Set up your GET method as a Lambda Function, using Lambda Proxy Integration, in region **us-east-1** and with the Lambda Function **TableScan**.



/Questions - GET - Setup

Choose the integration point for your new method.

Integration type: (●) Lambda Function ( ) HTTP ( ) Mock ( ) AWS Service ( ) VPC Link

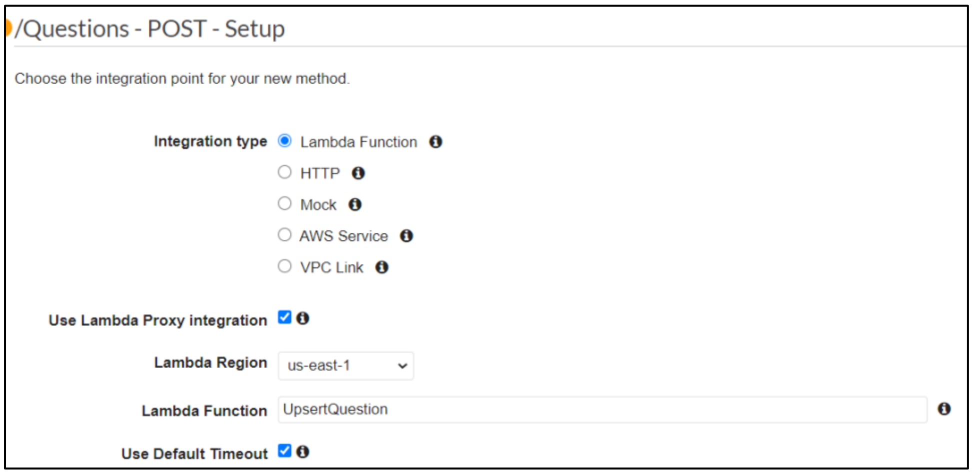
Use Lambda Proxy integration ✓

Lambda Region us-east-1

Lambda Function TableScan

Use Default Timeout ✓

8. Click the **Save** button and then click the **OK** button when asked about **Adding Permission to Lambda Function**.
9. Add the POST method to “Questions” with the following setup: a Lambda Function, using Lambda Proxy Integration, in region **us-east-1**, and with the Lambda Function **UpsertQuestion**.



/Questions - POST - Setup

Choose the integration point for your new method.

Integration type: [✓ Lambda Function] HTTP Mock AWS Service VPC Link

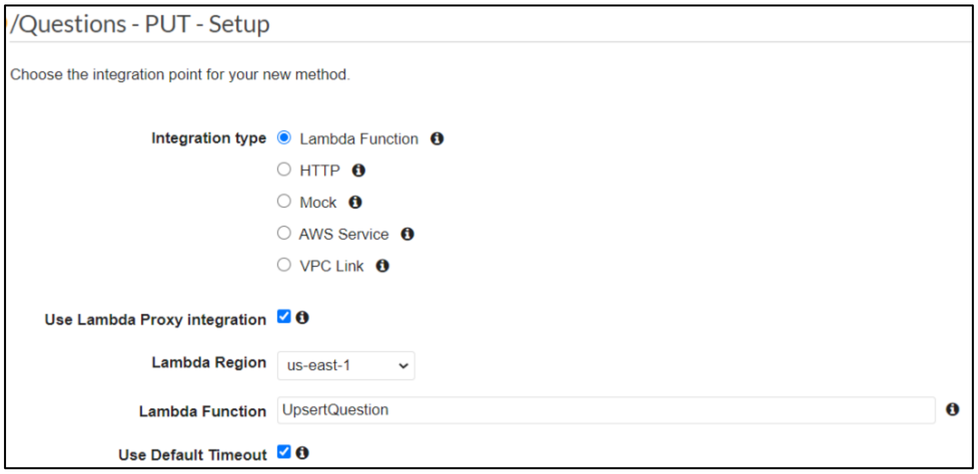
Use Lambda Proxy integration ✓

Lambda Region: us-east-1

Lambda Function: UpsertQuestion

Use Default Timeout ✓

10. Add the PUT method to "Questions" with the following setup: a Lambda Function, using Lambda Proxy Integration, in region **us-east-1**, and with the Lambda Function **UpsertQuestion**.



/Questions - PUT - Setup

Choose the integration point for your new method.

Integration type: [✓ Lambda Function] HTTP Mock AWS Service VPC Link

Use Lambda Proxy integration ✓

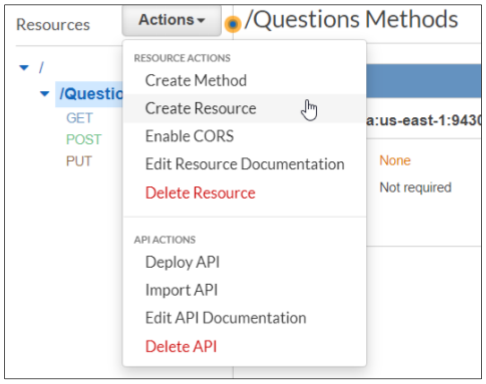
Lambda Region: us-east-1

Lambda Function: UpsertQuestion

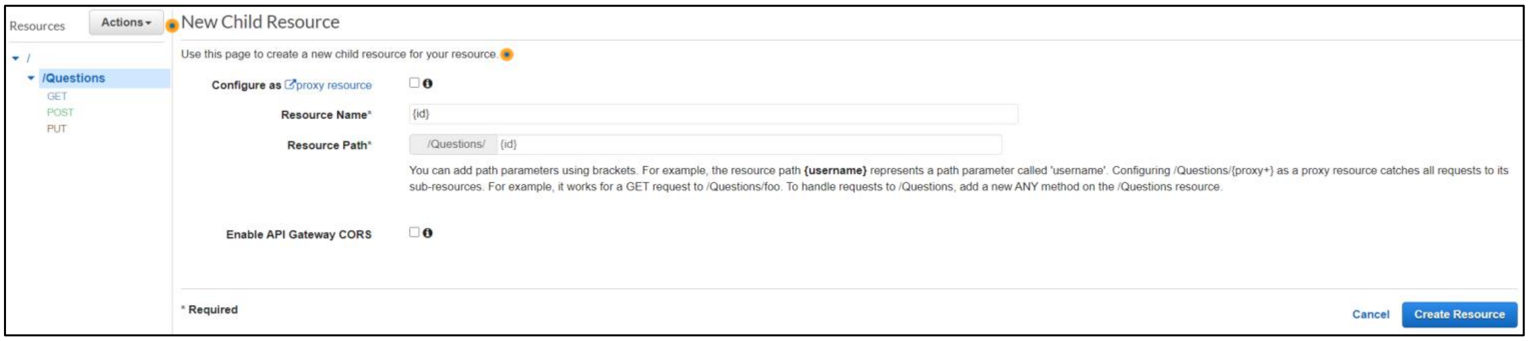
Use Default Timeout ✓

The {id} Sub-Resource

1. Select the **/Questions** resource and click **Create Resource** under the **Actions** menu.



2. Create a resource named **{id}**. Pay special attention to the resource path – it should be **/Questions/{id}**. Leave both check boxes unchecked.



Resources | Actions ▾

▾ /
▾ /Questions
GET
POST
PUT

New Child Resource

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

Configure as proxy resource	☐
Resource Name*	{id}
Resource Path*	/Questions/ {id}

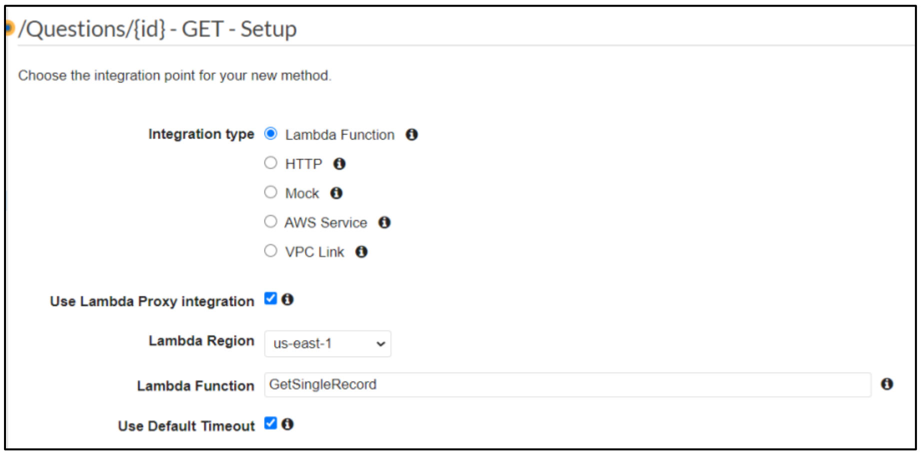
You can add path parameters using brackets. For example, the resource path **{username}** represents a path parameter called 'username'. Configuring /Questions/{proxy+} as a proxy resource catches all requests to its sub-resources. For example, it works for a GET request to /Questions/foo. To handle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

Enable API Gateway CORS	☐

* Required

Cancel | Create Resource

3. Click the **Create Resource** button.
4. Add a **GET** method to **/Questions/{id}** with the following setup: a Lambda Function, using Lambda Proxy Integration, in region **us-east-1**, and with the Lambda Function **GetSingleRecord**.

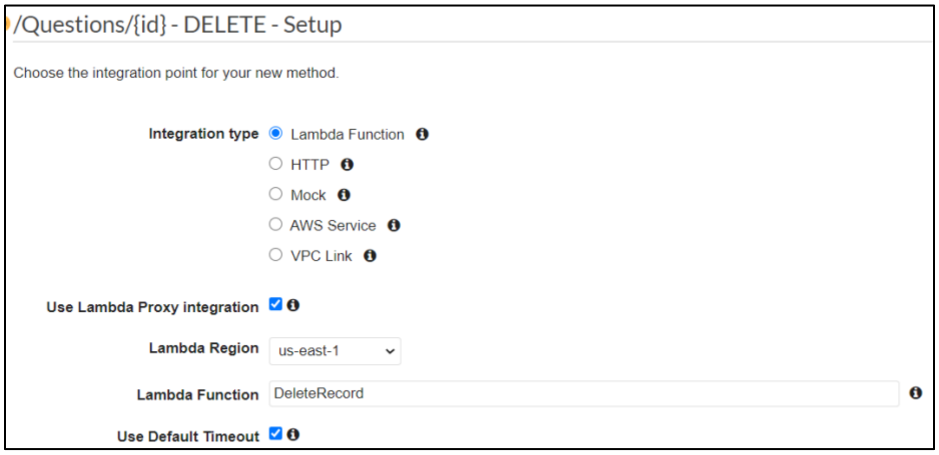


/Questions/{id} - GET - Setup

Choose the integration point for your new method.

Integration type	[✓ Lambda Function] HTTP Mock AWS Service VPC Link
Use Lambda Proxy integration	✓
Lambda Region	us-east-1
Lambda Function	GetSingleRecord
Use Default Timeout	✓

5. Add a **DELETE** method to **/Questions/{id}** with the following setup: a Lambda Function, using Lambda Proxy Integration, in region **us-east-1**, and with the Lambda Function **DeleteRecord**.

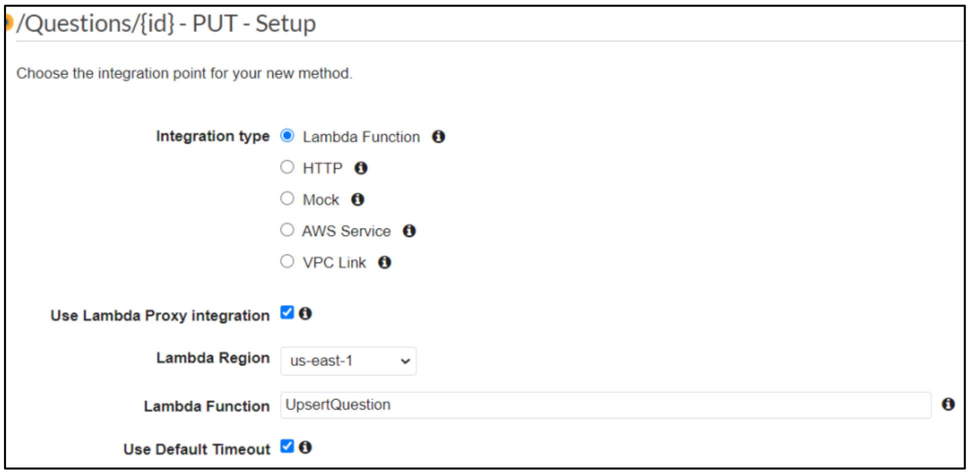


/Questions/{id} - DELETE - Setup

Choose the integration point for your new method.

Integration type	[✓ Lambda Function] HTTP Mock AWS Service VPC Link
Use Lambda Proxy integration	✓
Lambda Region	us-east-1
Lambda Function	DeleteRecord
Use Default Timeout	✓

6. Add a **PUT** method to **/Questions/{id}** with the following setup: a Lambda Function, using Lambda Proxy Integration, in region **us-east-1**, and with the Lambda Function **UpsertQuestion**.



/Questions/{id} - PUT - Setup

Choose the integration point for your new method.

Integration type: [✓ Lambda Function] HTTP Mock AWS Service VPC Link

Use Lambda Proxy integration: ✓

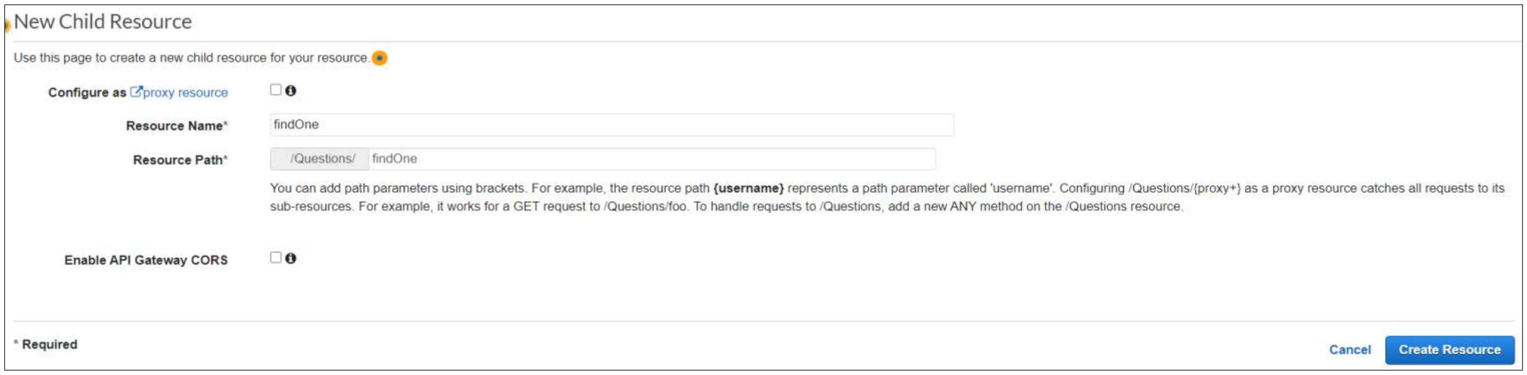
Lambda Region: us-east-1

Lambda Function: UpsertQuestion

Use Default Timeout: ✓

The findOne Sub-Resource

1. Create a new resource named "findOne" under **Questions**. Pay special attention to the casing of the resource name.



New Child Resource

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

Configure as proxy resource: ☐

Resource Name*: findOne

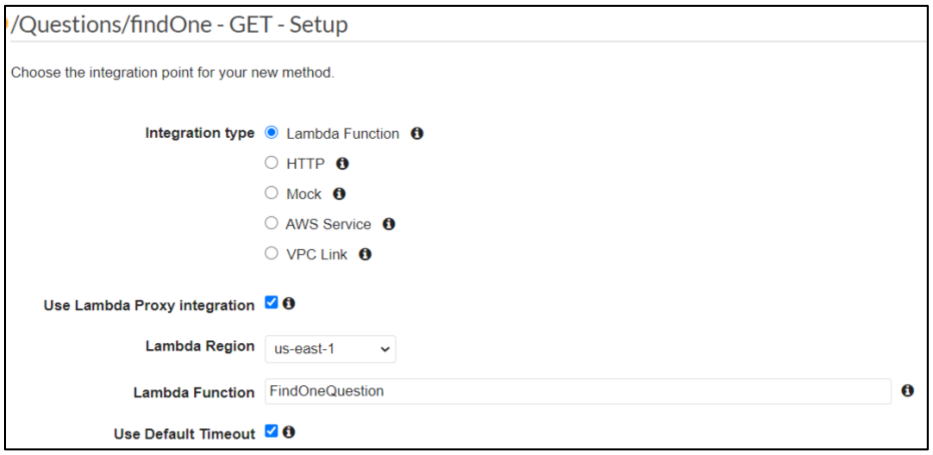
Resource Path*: /Questions/ findOne

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

Enable API Gateway CORS: ☐

* Required

2. Add a **GET** method to **/Questions/findOne** with the following setup: a Lambda Function, using Lambda Proxy Integration, in region **us-east-1**, and with the Lambda Function **FindOneQuestion**.



/Questions/findOne - GET - Setup

Choose the integration point for your new method.

Integration type: [✓ Lambda Function] HTTP Mock AWS Service VPC Link

Use Lambda Proxy integration: ✓

Lambda Region: us-east-1

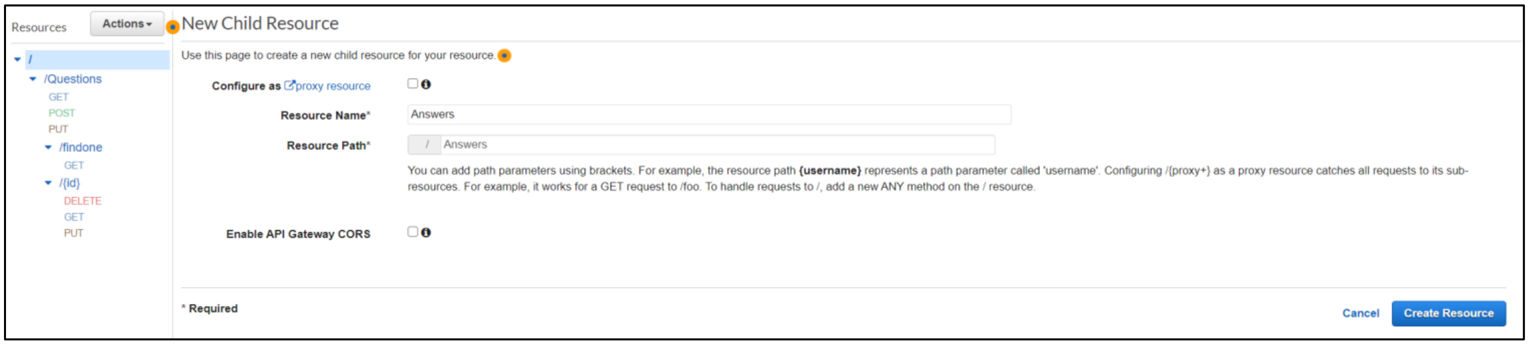
Lambda Function: FindOneQuestion

Use Default Timeout: ✓

Part Two – The Answers API

The Answers Resource

1. Create a new top-level resource (child of "/") named **Answers**. Make sure the resource path is **/Answers**.



New Child Resource

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

Configure as proxy resource ☐

Resource Name* Answers

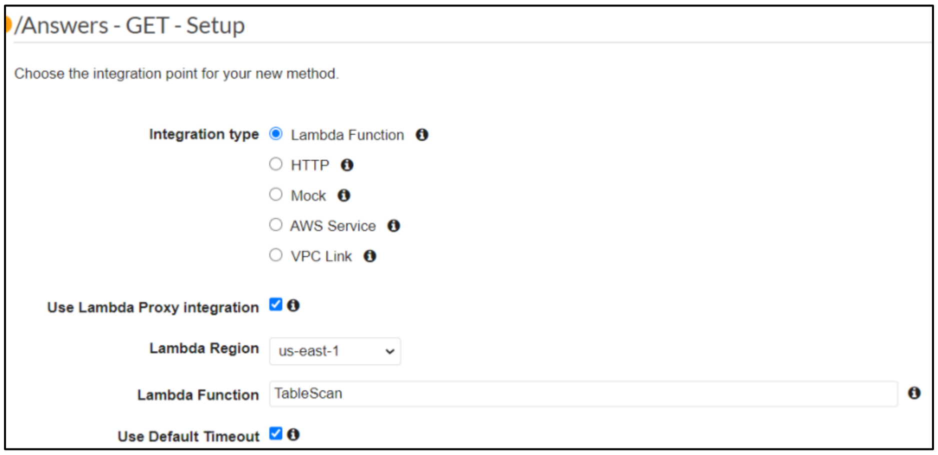
Resource Path* / Answers

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

2. Add a **GET** method to **/Answers** with the following setup: a Lambda Function, using Lambda Proxy Integration, in region **us-east-1**, and with the Lambda Function **TableScan**.



/Answers - GET - Setup

Choose the integration point for your new method.

Integration type ● Lambda Function ○ HTTP ○ Mock ○ AWS Service ○ VPC Link

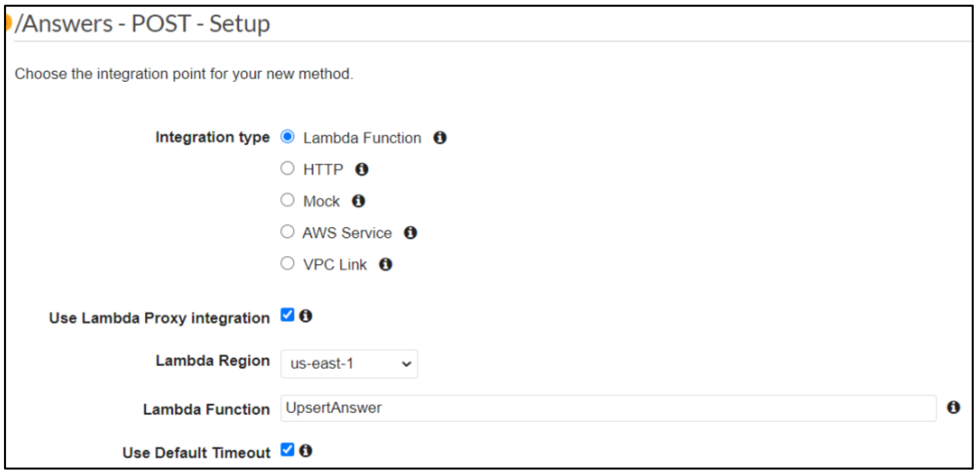
Use Lambda Proxy integration ✓

Lambda Region us-east-1

Lambda Function TableScan

Use Default Timeout ✓

3. Add a **POST** method to **/Answers** with the following setup: a Lambda Function, using Lambda Proxy Integration, in region **us-east-1**, and with the Lambda Function **UpsertAnswer**.



/Answers - POST - Setup

Choose the integration point for your new method.

Integration type: [✓ Lambda Function] HTTP Mock AWS Service VPC Link

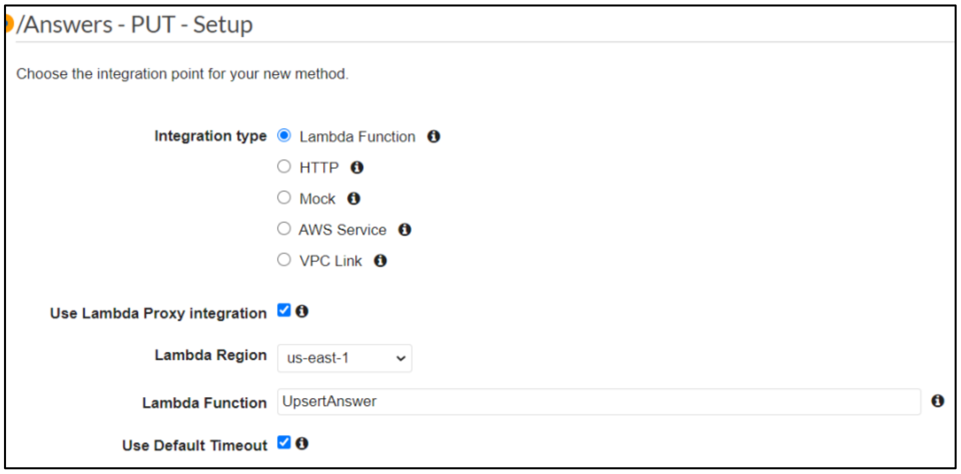
Use Lambda Proxy integration ✓

Lambda Region: us-east-1

Lambda Function: UpsertAnswer

Use Default Timeout ✓

4. Add a **PUT** method to **/Answers** with the following setup: a Lambda Function, using Lambda Proxy Integration, in region **us-east-1**, and with the Lambda Function **UpsertAnswer**.



/Answers - PUT - Setup

Choose the integration point for your new method.

Integration type: [✓ Lambda Function] HTTP Mock AWS Service VPC Link

Use Lambda Proxy integration ✓

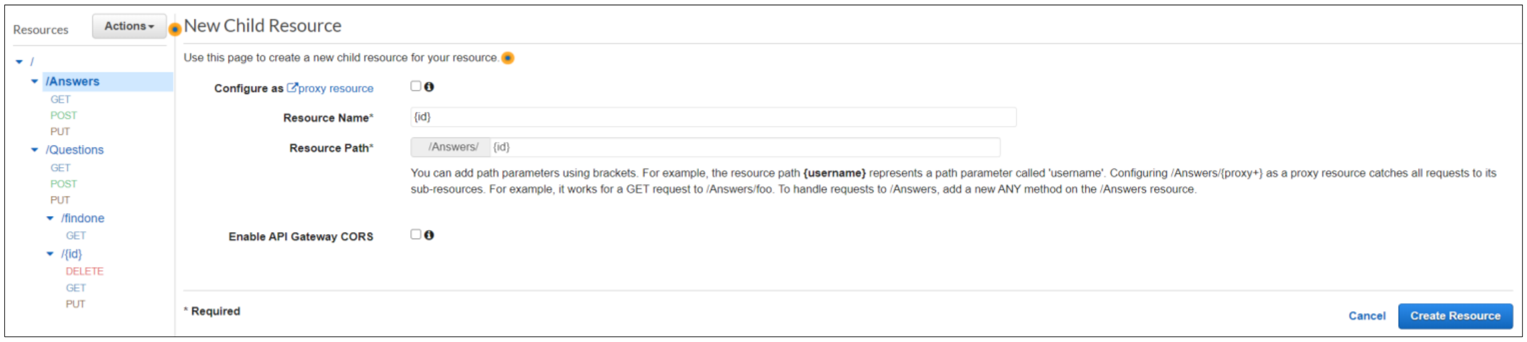
Lambda Region: us-east-1

Lambda Function: UpsertAnswer

Use Default Timeout ✓

The {id} Sub-Resource

1. Create an **{id}** resource under **Answers**. Make sure the Resource Path is **/Answers/{id}**.



New Child Resource

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

Configure as proxy resource ☐

Resource Name*: {id}

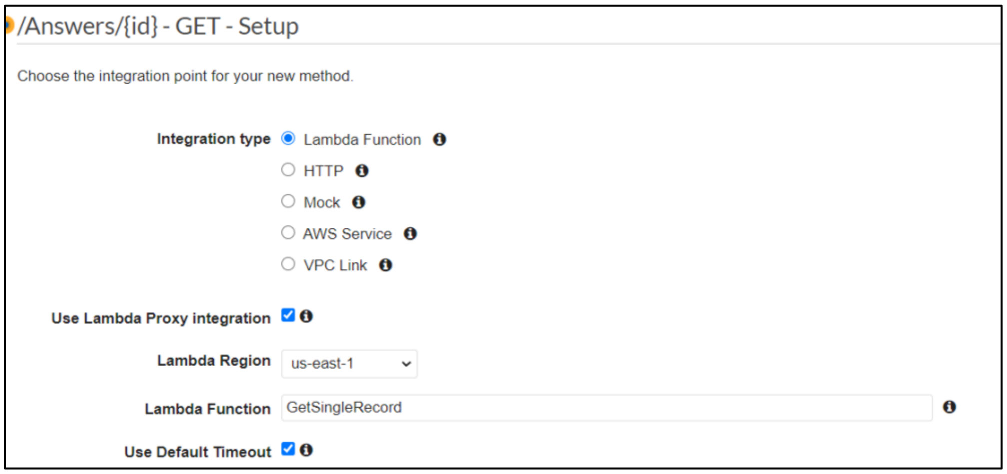
Resource Path*: /Answers/ {id}

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

Enable API Gateway CORS ☐

* Required

2. Add a **GET** method to **/Answers/{id}** with the following setup: a Lambda Function, using Lambda Proxy Integration, in region **us-east-1**, and with the Lambda Function **GetSingleRecord**.



/Answers/{id} - GET - Setup

Choose the integration point for your new method.

Integration type (● Lambda Function) ○ HTTP ○ Mock ○ AWS Service ○ VPC Link

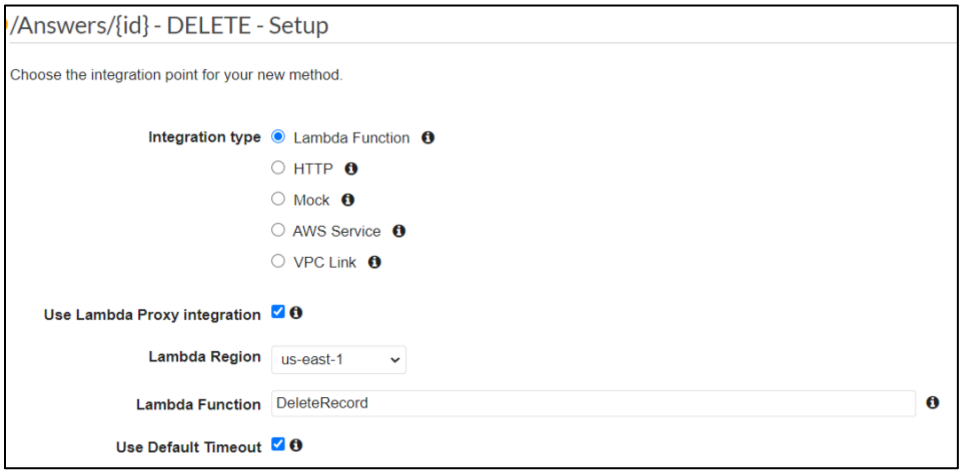
Use Lambda Proxy integration ✓

Lambda Region us-east-1

Lambda Function GetSingleRecord

Use Default Timeout ✓

3. Add a **DELETE** method to **/Answers/{id}** with the following setup: a Lambda Function, using Lambda Proxy Integration, in region **us-east-1**, and with the Lambda Function **DeleteRecord**.



/Answers/{id} - DELETE - Setup

Choose the integration point for your new method.

Integration type (● Lambda Function) ○ HTTP ○ Mock ○ AWS Service ○ VPC Link

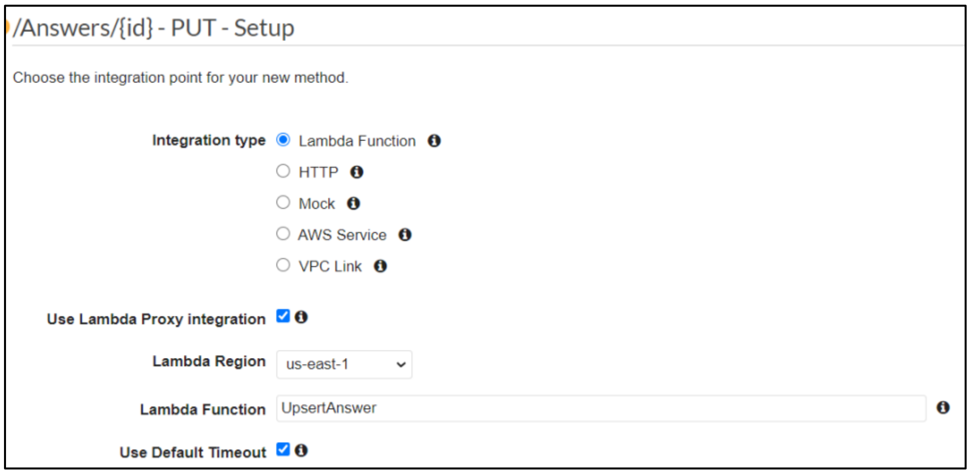
Use Lambda Proxy integration ✓

Lambda Region us-east-1

Lambda Function DeleteRecord

Use Default Timeout ✓

4. Add a **PUT** method to **/Answers/{id}** with the following setup: a Lambda Function, using Lambda Proxy Integration, in region **us-east-1**, and with the Lambda Function **UpsertAnswer**.



/Answers/{id} - PUT - Setup

Choose the integration point for your new method.

Integration type (● Lambda Function) ○ HTTP ○ Mock ○ AWS Service ○ VPC Link

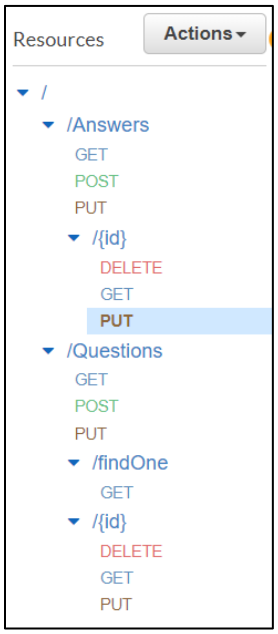
Use Lambda Proxy integration ✓

Lambda Region us-east-1

Lambda Function UpsertAnswer

Use Default Timeout ✓

5. You should now have an API Resource tree that looks like this:

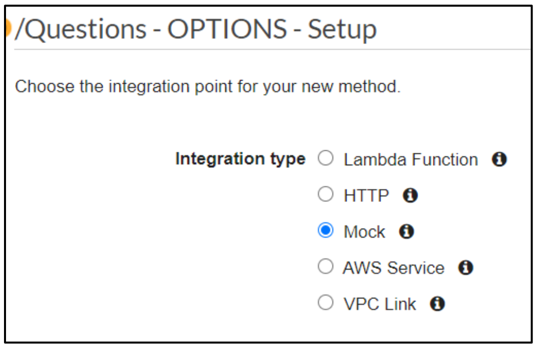


Supporting CORS

Cross-Origin Resource Sharing (CORS) is a mechanism to use HTTP headers to allow resource sharing across different domains. AWS Gateway has an action to enable CORS, but we will *not* be using it. This action is only partially successful, so we will do this manually as defined on the CORS on AWS API Gateway website. The requirements are to have an **OPTIONS** method for each resource with specific headers returned, and for some specific headers to be returned on each **REST** method called.

Adding CORS to Questions Resource

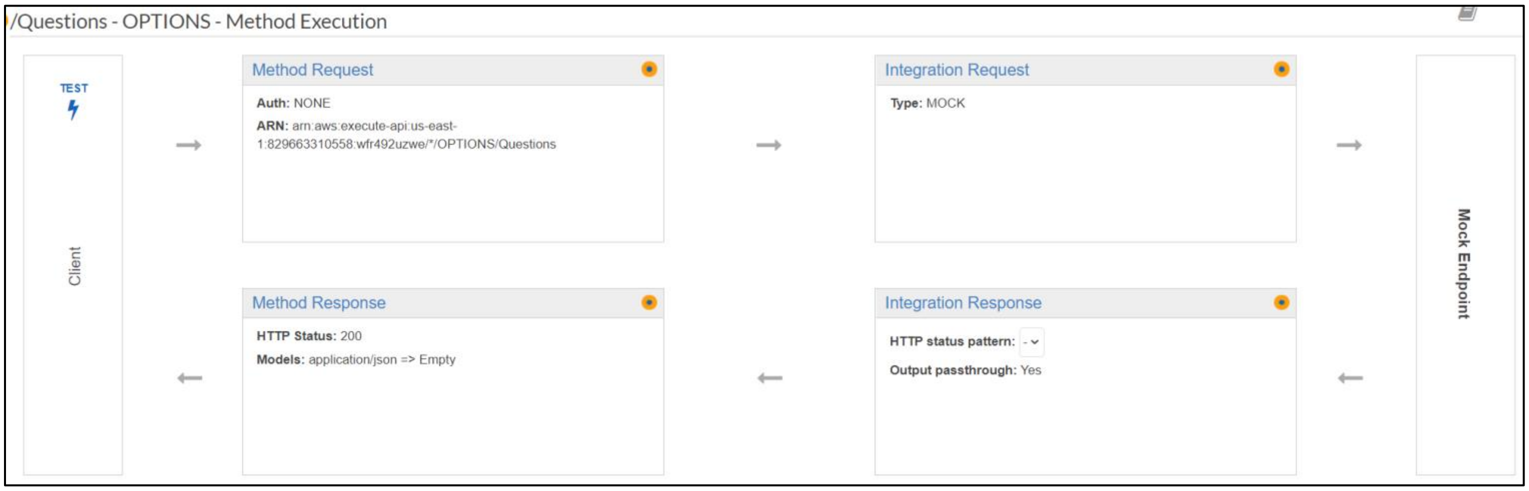
1. Select the **/Questions** resource and add an **OPTIONS** method.
2. Select **Mock** as the Integration type and click the **Save** button.


/Questions - OPTIONS - Setup

Choose the integration point for your new method.

Integration type: Lambda Function / HTTP / [✓ Mock] / AWS Service / VPC Link

3. Click on the **Method Response** link.


/Questions - OPTIONS - Method Execution

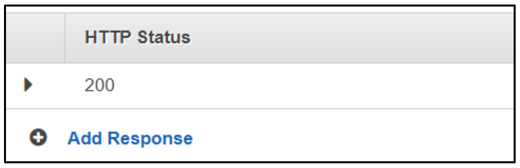
Method Request
Auth: NONE
ARN: arn:aws:execute-api:us-east-1:829663310558:wfr492uzwe/*/OPTIONS/Questions

Integration Request
Type: MOCK

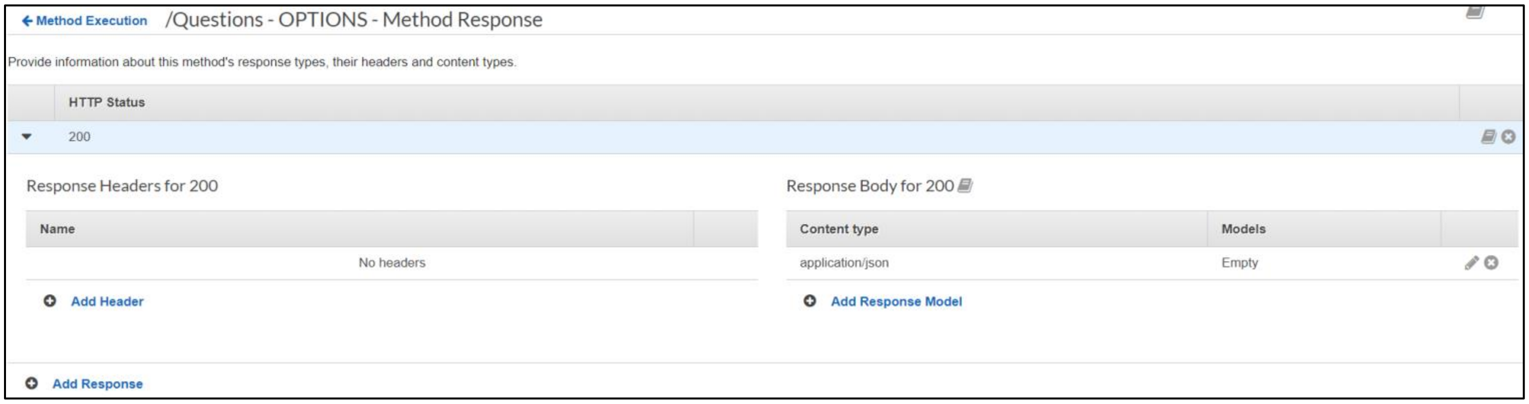
Method Response
HTTP Status: 200
Models: application/json => Empty

Integration Response
HTTP status pattern: -
Output passthrough: Yes

4. There should be an HTTP Status of “200”, but if not, add it by clicking **Add Response**, entering “200” in the box, and clicking the checkmark on the right side.



5. Click on the arrow to the left of the HTTP status to open the 200 status details.



/Questions - OPTIONS - Method Response

Provide information about this method's response types, their headers and content types.

HTTP Status
200

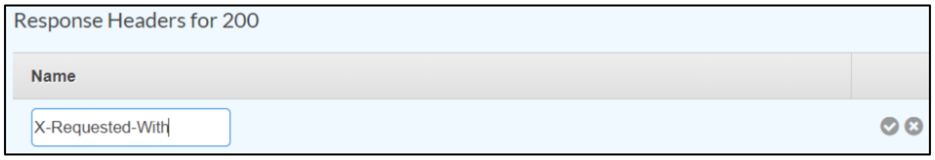
Response Headers for 200

Name
No headers

Response Body for 200

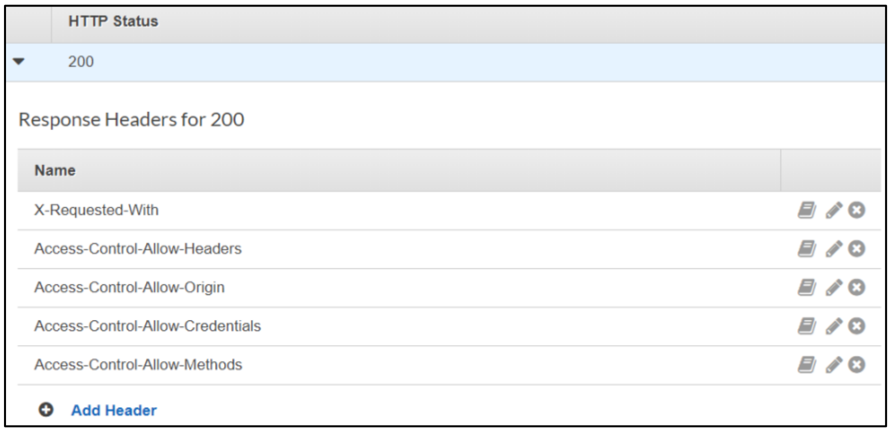
Content type	Models
application/json	Empty

6. Now you need to add some headers. Click **Add Header**.
7. Enter “X-Requested-With” in the box and click the checkmark. Be careful to use the correct case, and not to add spaces.


Response Headers for 200

Name
X-Requested-With

8. Repeat the process to add "Access-Control-Allow-Headers", "Access-Control-Allow-Origin", "Access-Control-Allow-Methods", and "Access-Control-Allow-Credentials". When complete, the **Response Headers** section will look like this:

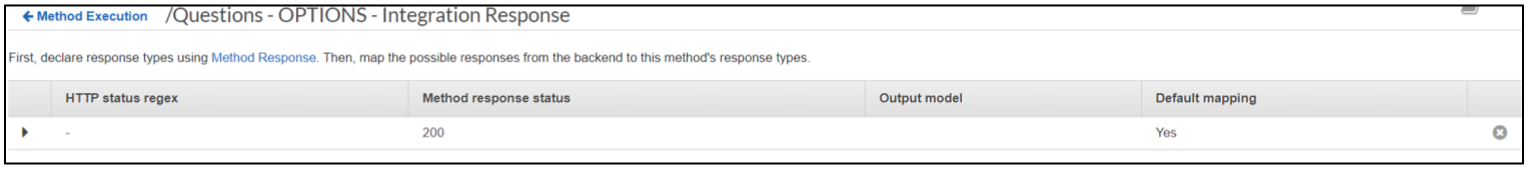


HTTP Status
200

Response Headers for 200

Name
X-Requested-With
Access-Control-Allow-Headers
Access-Control-Allow-Origin
Access-Control-Allow-Credentials
Access-Control-Allow-Methods

9. Click on "Method Execution" in the upper right to return to the **Method Execution** page.
10. Click the "Integration Response" link to go to the **Integration Response** page.

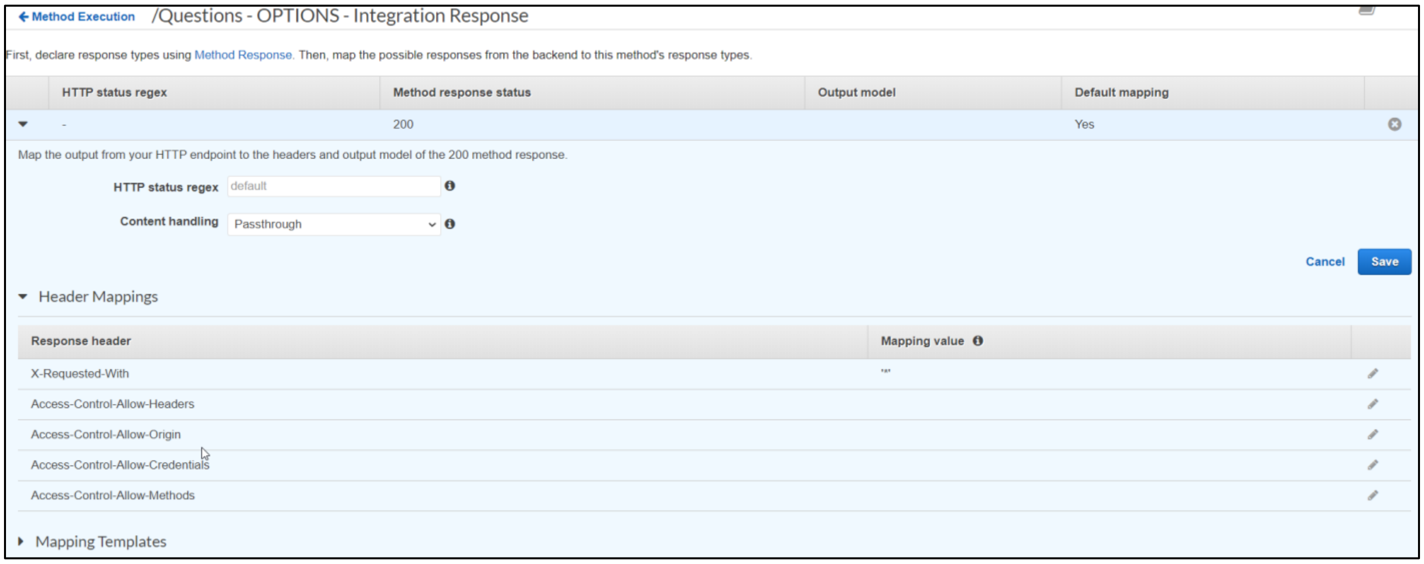


/Questions - OPTIONS - Integration Response

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

HTTP status regex	Method response status	Output model	Default mapping
-	200		Yes

11. Open the "200" method response status by clicking the arrow to the left of the row.
12. Click on **Header Mappings** to expand that section.



/Questions - OPTIONS - Integration Response

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

HTTP status regex	Method response status	Output model	Default mapping
-	200		Yes

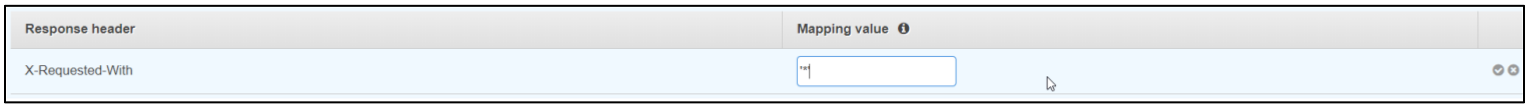
Map the output from your HTTP endpoint to the headers and output model of the 200 method response.

HTTP status regex: default

Content handling: Passthrough

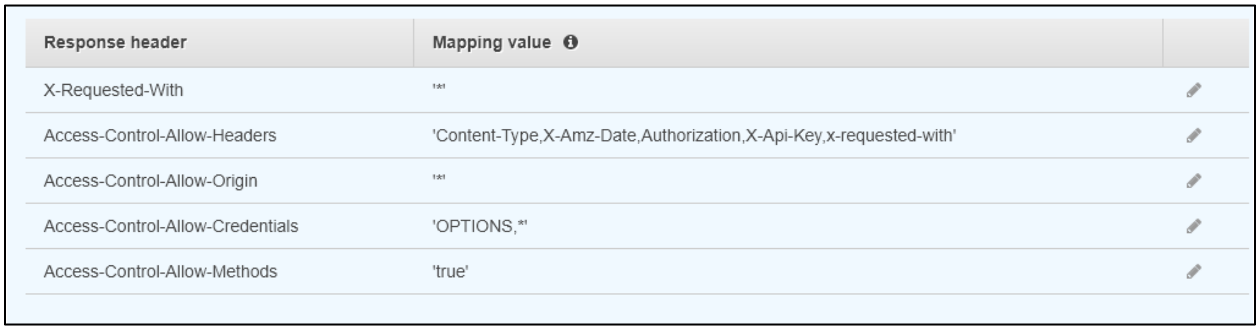
Response header	Mapping value
X-Requested-With	'*'
Access-Control-Allow-Headers	
Access-Control-Allow-Origin	
Access-Control-Allow-Credentials	
Access-Control-Allow-Methods	

13. Set the Mapping Value for each of the response headers by clicking the pencil icon on the right, entering the value, and clicking the checkmark.



Response header	Mapping value
X-Requested-With	'*'

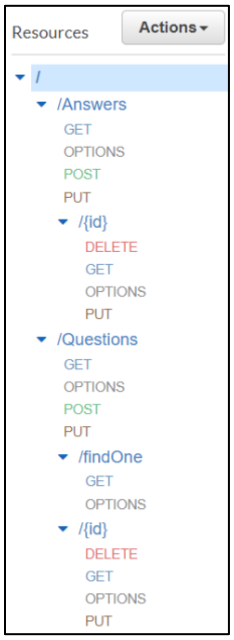
14. For **X-Requested-With**, enter '*', with the quotes.
15. For **Access-Control-Allow-Headers**, enter 'Content-Type,X-Amz-Date,Authorization,X-Api-Key,x-requested-with', with the quotes.
16. For **Access-Control-Allow-Origin**, enter '*', with the quotes.
17. For **Access-Control-Allow-Methods**, enter 'OPTIONS,*', with the quotes.
18. For **Access-Control-Allow-Credentials**, enter 'true', with the quotes.
19. When done, the **Header Mappings** page should look like this:



Response header	Mapping value
X-Requested-With	'*'
Access-Control-Allow-Headers	'Content-Type,X-Amz-Date,Authorization,X-Api-Key,x-requested-with'
Access-Control-Allow-Origin	'*'
Access-Control-Allow-Credentials	'OPTIONS,*'
Access-Control-Allow-Methods	'true'

Adding CORS to the Remaining Resources

1. Repeat this process for each resource and sub-resource:
   - /Answers
   - /Answers/{id}
   - /Questions/findOne
   - /Questions/{id}
2. When complete, your resource tree will look like this:



```
Resources  Actions
/
  /Answers
    GET
    OPTIONS
    POST
    PUT
    /{id}
      DELETE
      GET
      OPTIONS
      PUT
  /Questions
    GET
    OPTIONS
    POST
    PUT
    /findOne
      GET
      OPTIONS
    /{id}
      DELETE
      GET
      OPTIONS
      PUT
```

CORS Headers for the other Methods

The other methods need to return CORS headers as well. The good news is there is nothing for you to do now. Because we configured our APIs as proxies to the Lambdas, we can handle this in the Lambda code – and you did that already. Near the start of each Lambda you created in Module Five is the following code:

```
const responseHeaders = (headers) => {
  const origin = headers.origin || headers.Origin;

  return {
    // HTTP headers to pass back to the client
    "Content-Type": "application/json",
    // the next headers support CORS
    "X-Requested-With": "*",
    "Access-Control-Allow-Headers":
      "Content-Type,X-Amz-Date,Authorization,X-Api-Key,x-requested-
with",
    "Access-Control-Allow-Origin": origin,
    "Access-Control-Allow-Methods": "OPTIONS,*",
    Vary: "Origin", // for proxies
    // the "has-cors" library used by the Angular application wants this
set
    "Access-Control-Allow-Credentials": "true",
  };
};
```

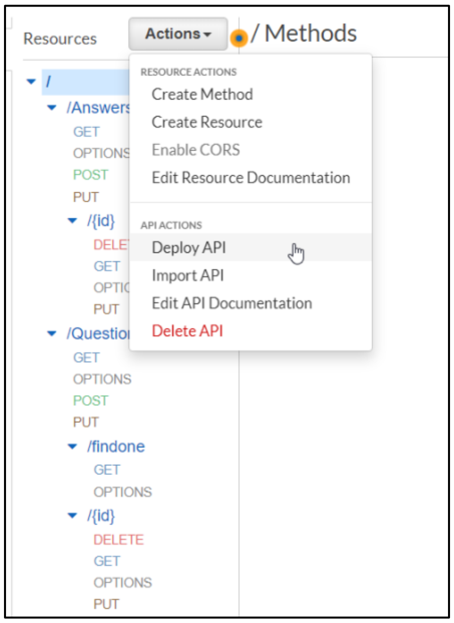
Towards the bottom of the Lambda is the code where the response to the client is created. You had this line of code:

```
// HTTP headers to pass back to the client
            headers: responseHeaders(event.headers),
```

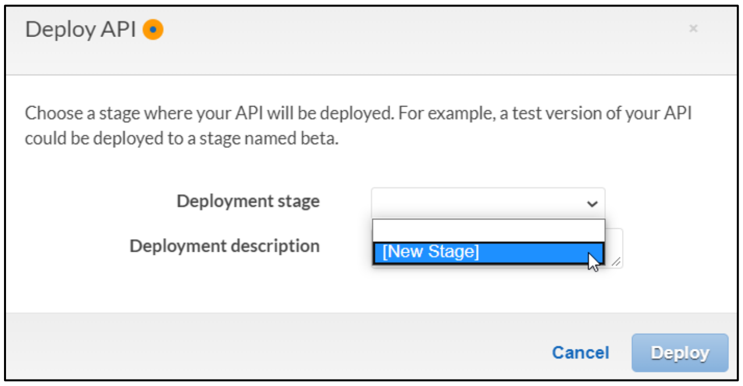
Between these two blocks of code and the APIs being configured as proxies, you are handling the CORS requirements for the other methods.

Deployment

1. Deploy the API using the “api” deployment name by clicking **Deploy API** from the **Actions** menu. If you need a refresher, refer to Module Four.

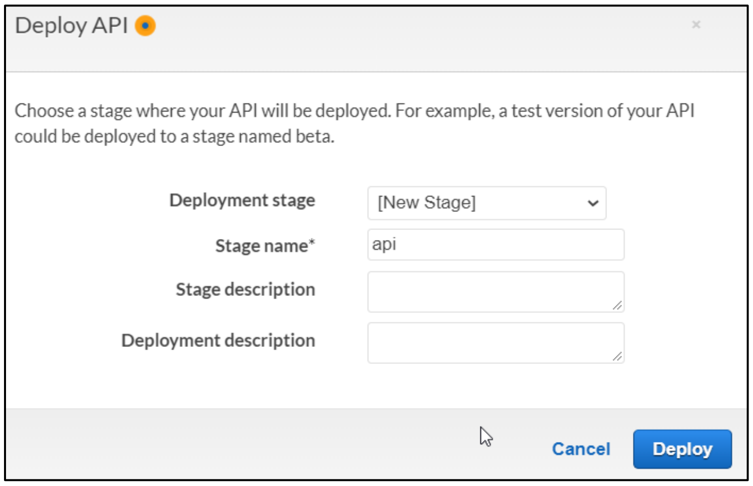


2. Create a new stage from the **Deployment Stage** drop-down menu.



Choose a stage where your API will be deployed. For example, a test version of your API could be deployed to a stage named beta.

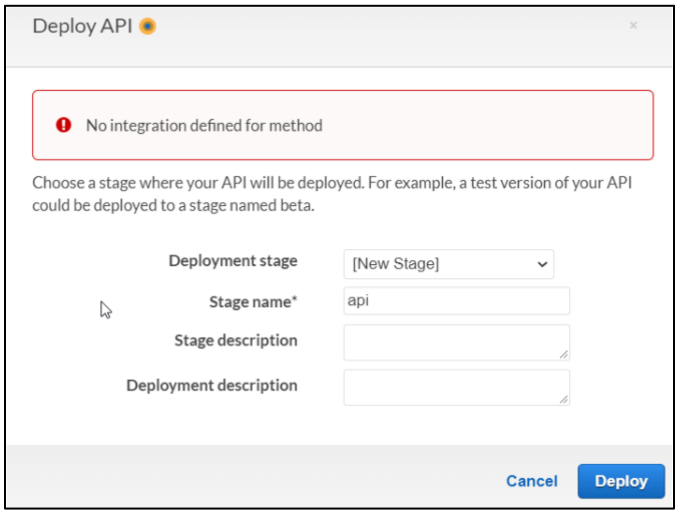
3. Enter a **Stage name** of “api” and click **Deploy**.



Choose a stage where your API will be deployed. For example, a test version of your API could be deployed to a stage named beta.

Deployment stage	[New Stage]
Stage name*	api
Stage description	
Deployment description	

4. A common error to see at this stage is “No integration defined for method”. If you see this error, then you forgot click the **Save** button when mapping API methods to your Lambdas. Check each API method and fix as needed. Then try again.



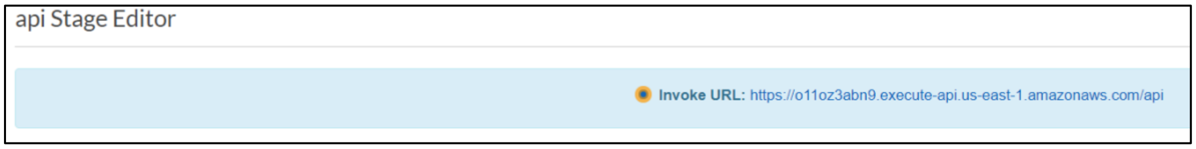
Deploy API

No integration defined for method

Choose a stage where your API will be deployed. For example, a test version of your API could be deployed to a stage named beta.

Deployment stage: [New Stage]
Stage name*: api
Stage description:
Deployment description:

5. Record the **Invoke URL** displayed after deployment. It will look like this:

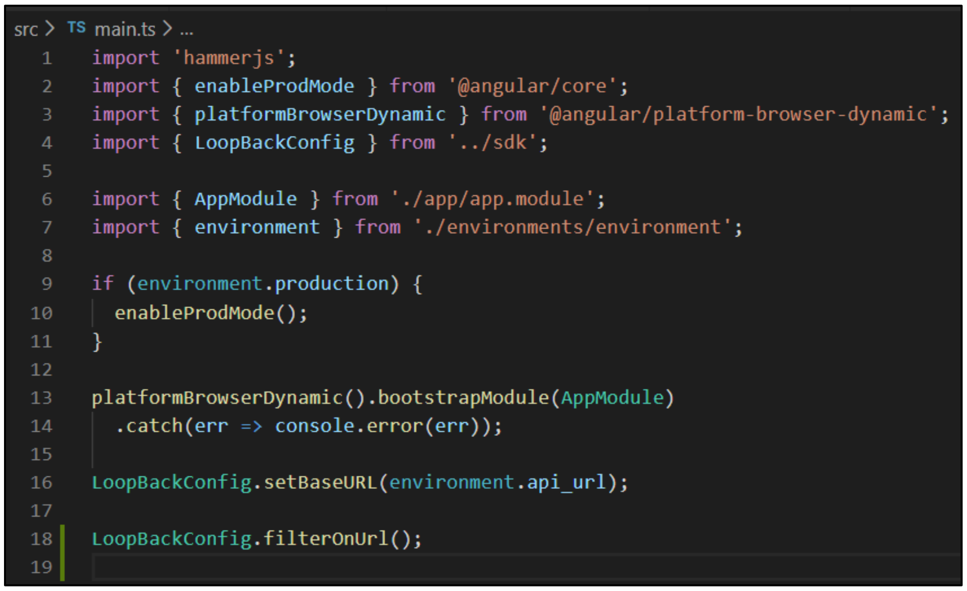


api Stage Editor

Invoke URL: https://o11oz3abn9.execute-api.us-east-1.amazonaws.com/api

Modifying the Angular Application

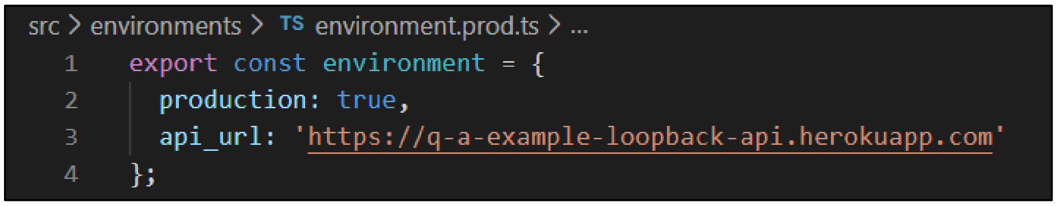
1. Modify **main.ts**. Add the line **LoopBackConfig.filterOnUrl();** to the end of the file. The resulting change will look like this:



src > TS main.ts > ...

```
import 'hammerjs';
import { enableProdMode } from '@angular/core';
import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';
import { LoopBackConfig } from '../sdk';

import { AppModule } from './app/app.module';
import { environment } from './environments/environment';

if (environment.production) {
  enableProdMode();
}

platformBrowserDynamic().bootstrapModule(AppModule)
  .catch(err => console.error(err));

LoopBackConfig.setBaseURL(environment.api_url);

LoopBackConfig.filterOnUrl();

```

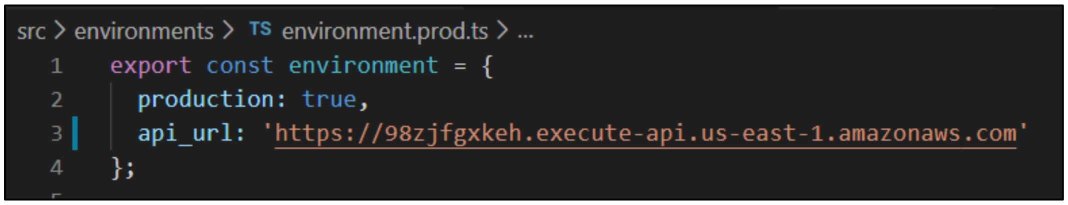
2. Modify **environment.prod.ts**.
   - Change the line **api_url: 'https://q-a-example-loopback-api.herokuapp.com**:



```
src > environments > TS environment.prod.ts > ...
export const environment = {
  production: true,
  api_url: 'https://q-a-example-loopback-api.herokuapp.com'
};
```

   - To **api_url: 'https://{deployed-url}.us-east-1.amazonaws.com'**

Remember: Your deployed URL is the one you created in the "Deployment" section of this guide. Be careful: Do not include the 'api' or a trailing '/'. The Angular application adds those.



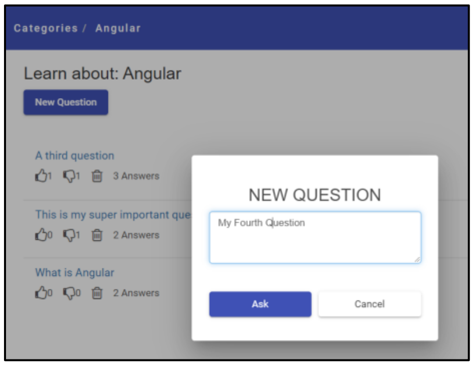
```
src > environments > TS environment.prod.ts > ...
export const environment = {
  production: true,
  api_url: 'https://98zjfgxkeh.execute-api.us-east-1.amazonaws.com'
};
```

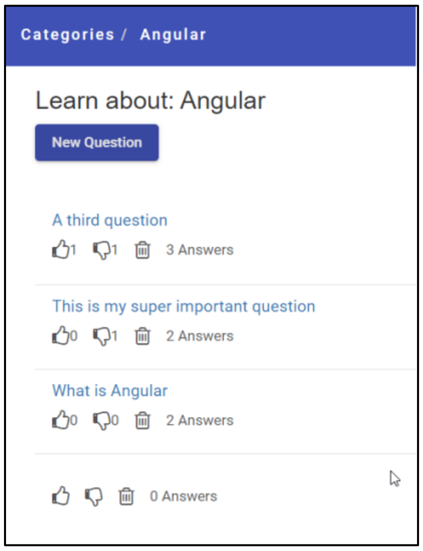
3. Build and deploy the Angular application (see the Module Three Assignment Guide).
   - Optional: You can safely delete the existing elements in your S3 bucket prior to copying new files over if you choose.

You Are Done!

Try testing your application. Some notes to consider:

1. You shifted the application from using the Loopback SDK on the client that was tied to the Loopback REST API on the server.
2. In completing the transfer, you would change the client side to not depend on the Loopback SDK and its assumption of a Loopback REST server.
3. Why does this matter? When you add a new Question or Answer, the text of the add is not always seen right away. The call is successful, and a new entry is visible – just without the title.





Categories / Angular

Learn about: Angular

New Question

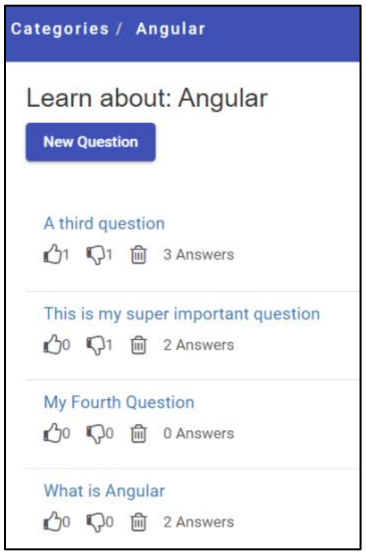
A third question
1 1 3 Answers

This is my super important question
0 1 2 Answers

What is Angular
0 0 2 Answers

0 Answers

4. If you click on **Categories** and back into **Angular**, the new content will show up.



Categories / Angular

Learn about: Angular

New Question

A third question
1 1 3 Answers

This is my super important question
0 1 2 Answers

My Fourth Question
0 0 0 Answers

What is Angular
0 0 2 Answers

However, changing that behavior may be beyond the scope of this class.

Time to celebrate a complete project!