**Requirement:**

You have a GET endpoint that has a query param and you want your lambda function to be able to read that query param.

Unlike to POST endpoint, GET endpoint cannot have a json body that can be easily read by Lambda function. You need to map these query params to a json using API Gateway’s tools.

API Gateway can intercept incoming request and can validate and transform it to some other form before it is submitted to your backend service. Here, we will learn how to validate and transform GET endpoint’s query param into json that can be read by your Lambda function.

Create a GET endpoint using API Gateway. API should be mapped to your Lambda Function (Hello.java’s myHandlerTakingParams handler).

You can have a function code as shown below. You can have a Map or java object that can be mapped to submitted Json.

public class Hello {

public String myHandlerTakingParams(**Map<String, Integer> params**, Context context) throws Exception {

if(params == null || params.size() == 0 || !params.containsKey("myCount")) throw new BadRequestException("BAD\_REQ: Check your count again. It can't be <=0");

Integer myCount = params.get("myCount");

if(myCount <= 0) throw new BadRequestException("BAD\_REQ: myCount is a required parameter.");

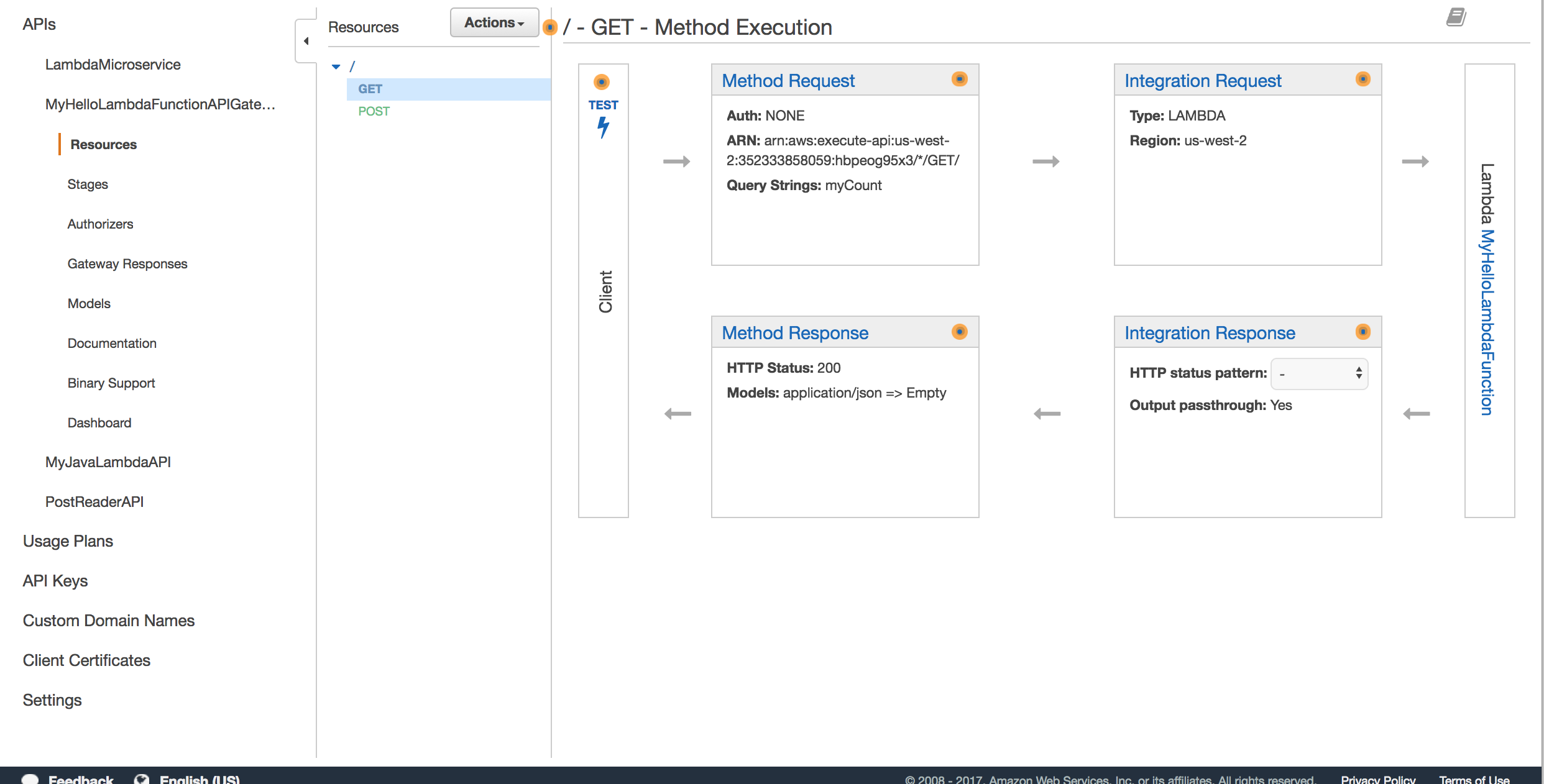
LambdaLogger logger = context.getLogger();

logger.log("received : " + myCount);

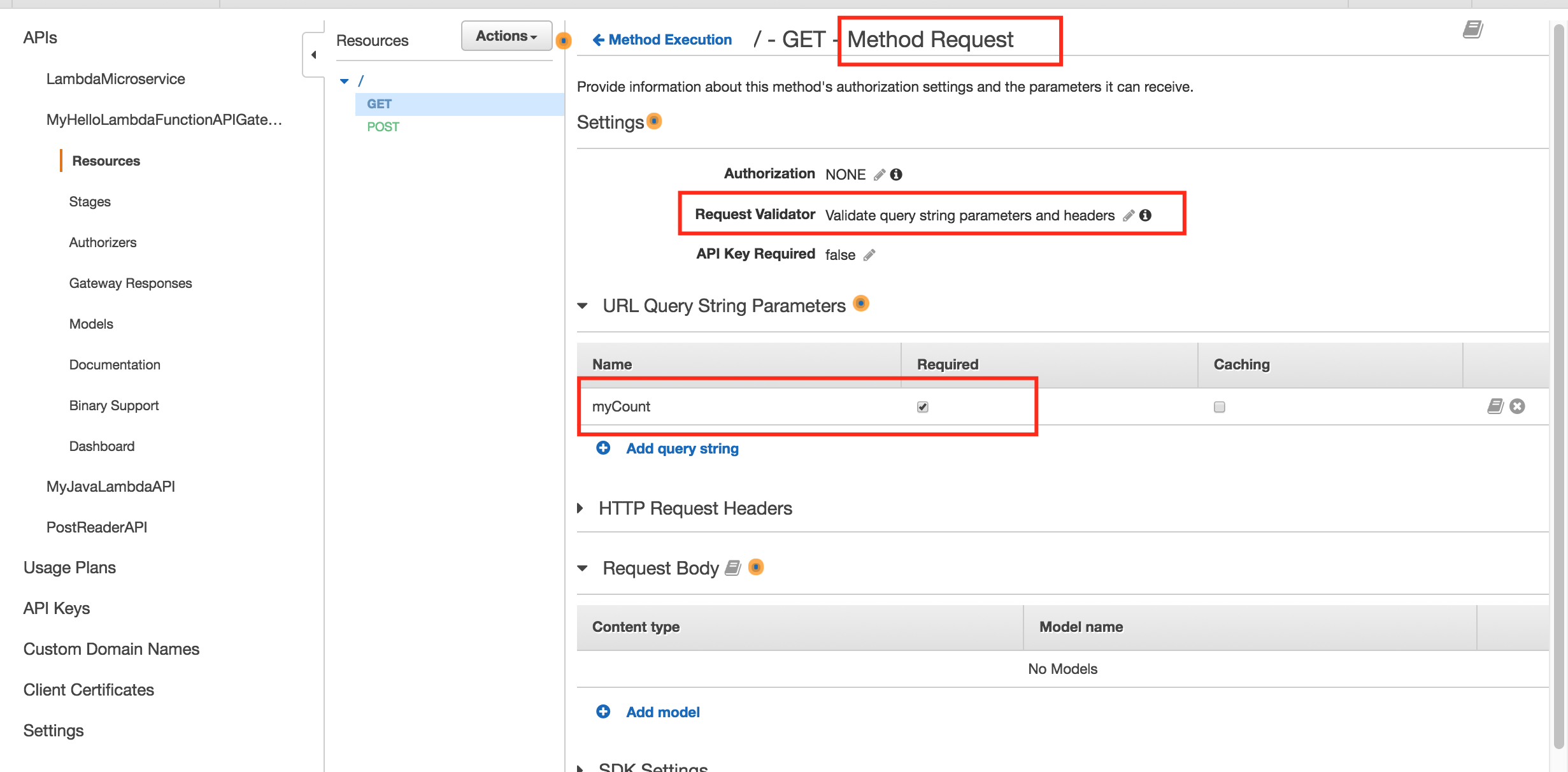
return String.valueOf(myCount);

}

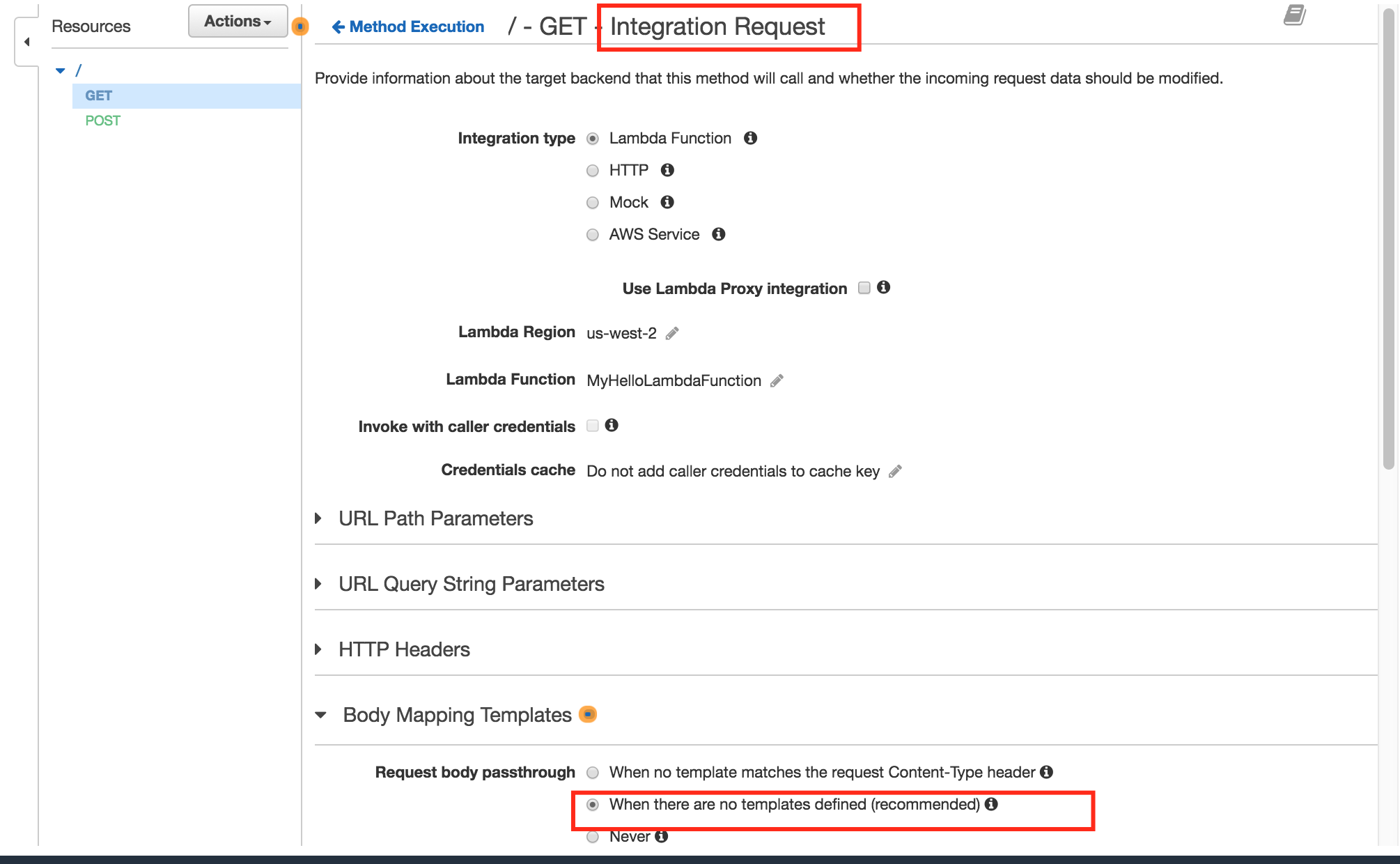
}

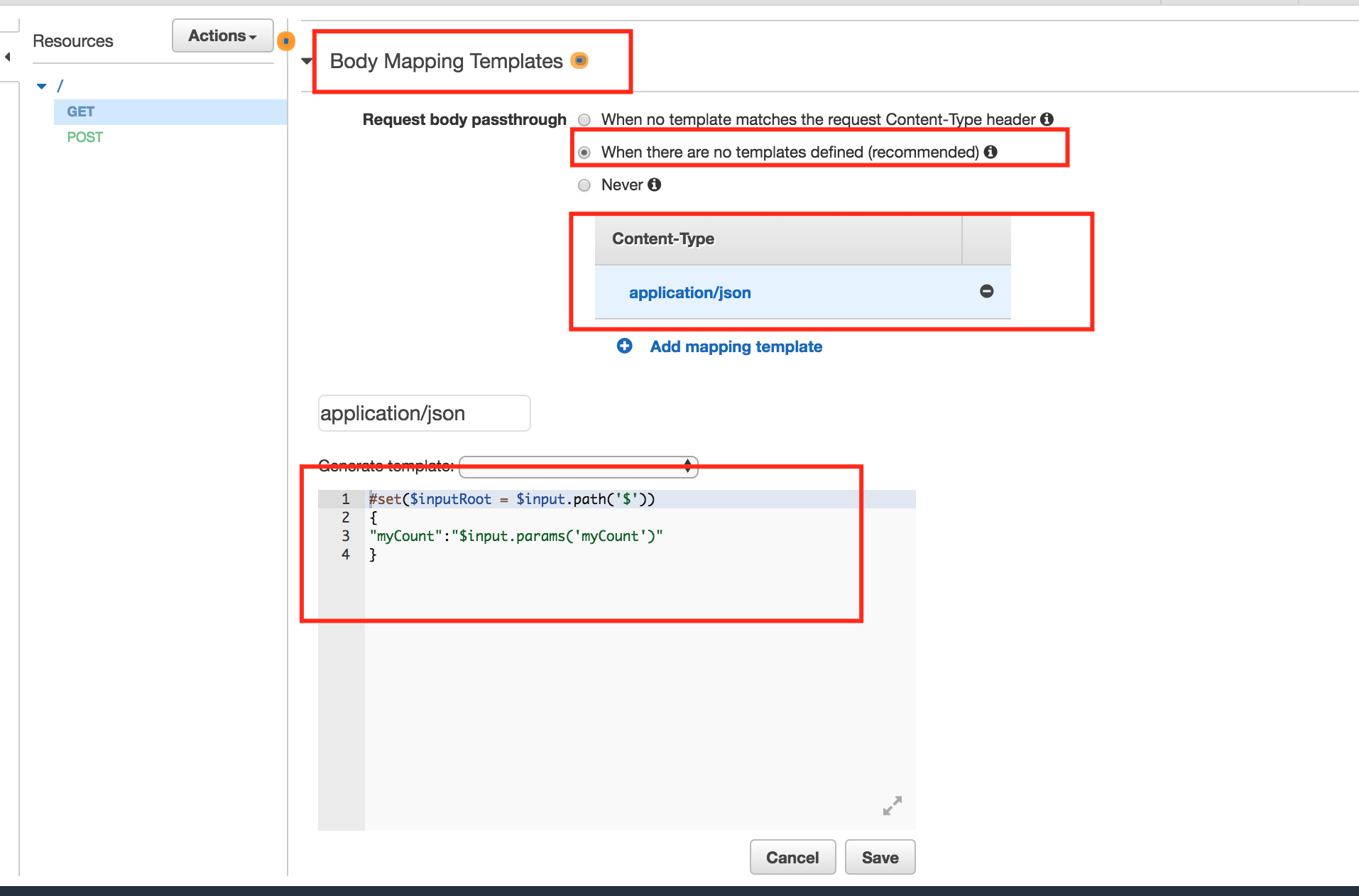


Add Query Param validation in ‘Method Request’ section.



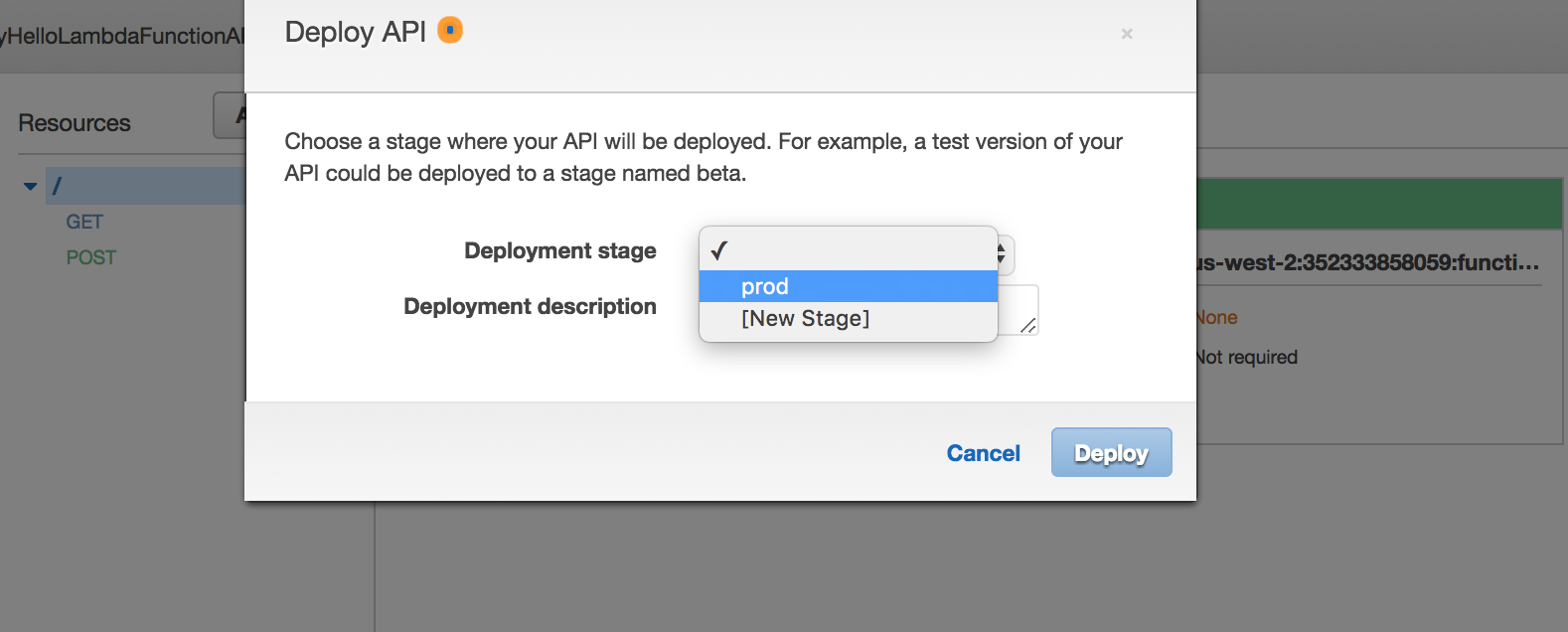
In ‘Integration Request’ section, map this query param ‘myCount’ to a json body that will be submitted a lambda function.



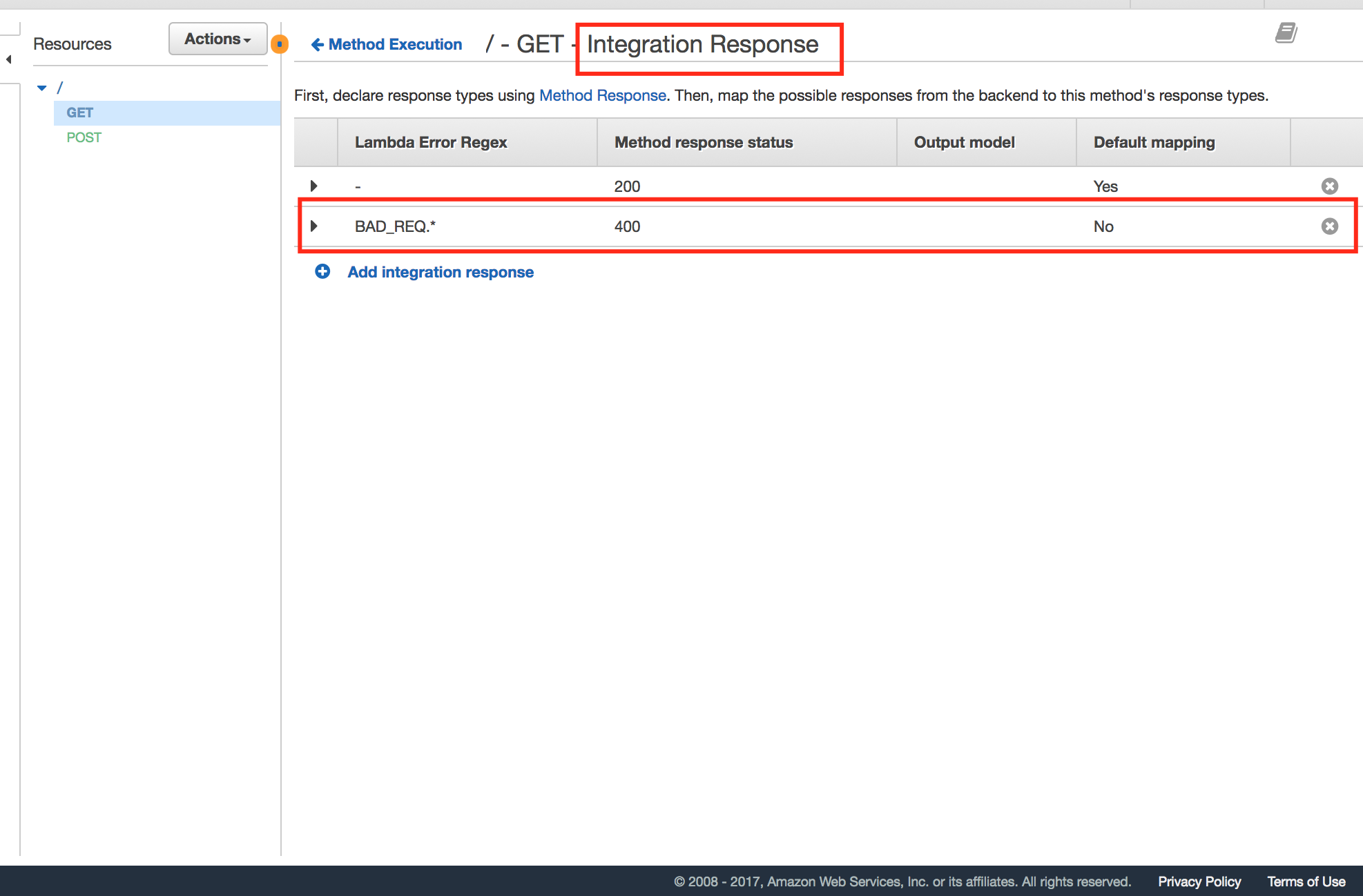


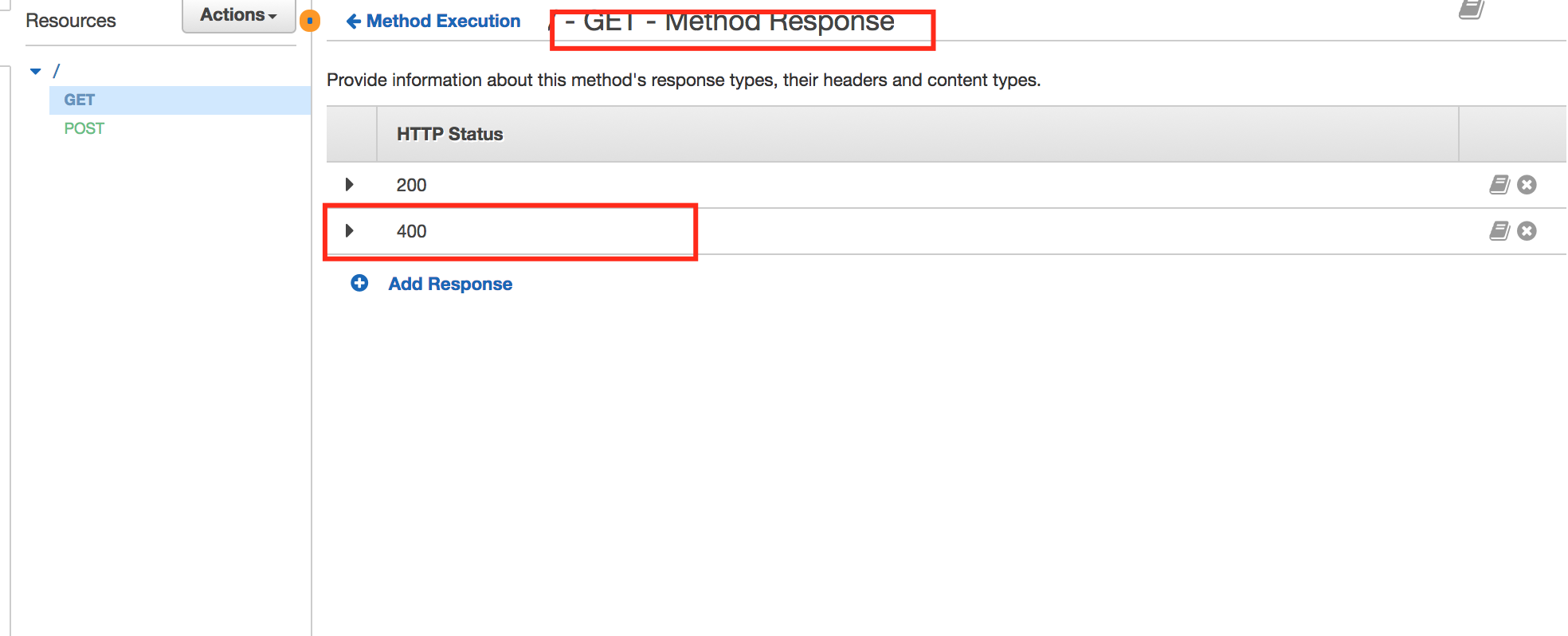
This mapping read query param ‘myCount’ and assigns it to json variable ‘myCount’.

DON’T FORGET to deploy api

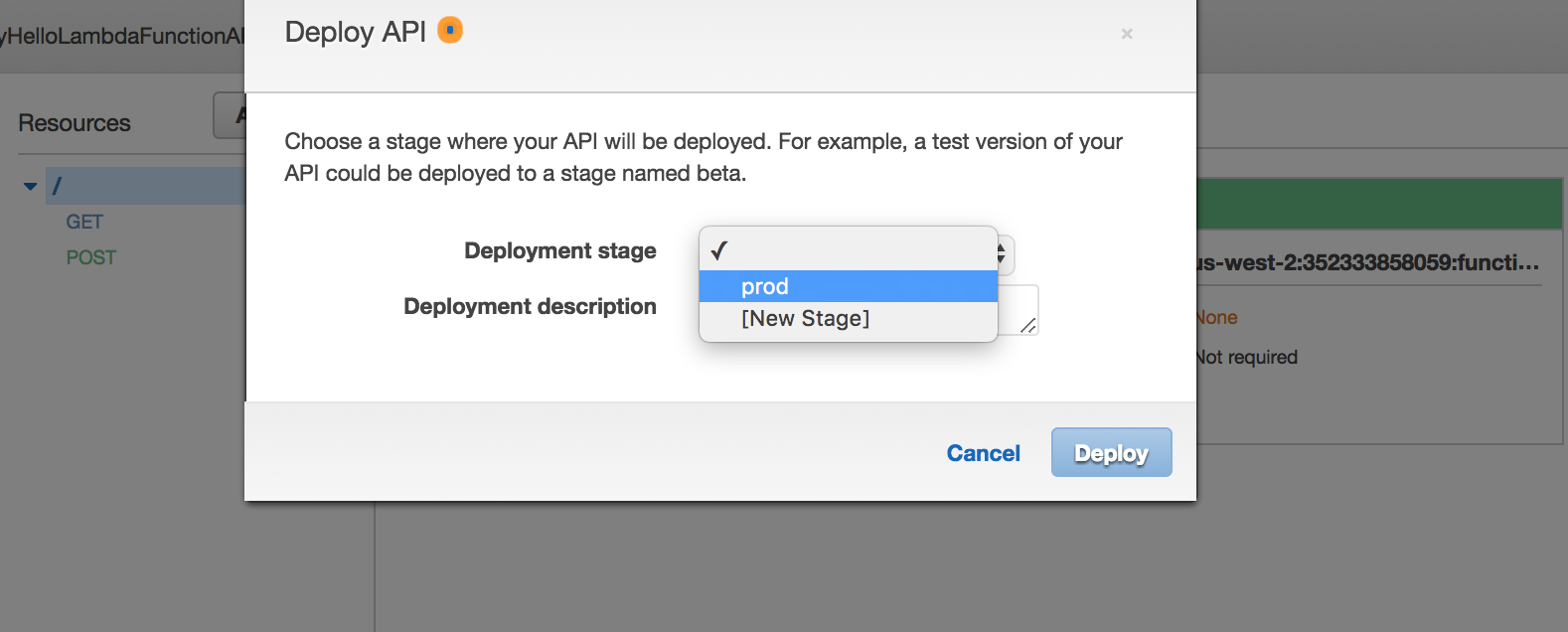


You can also map exceptions thrown by lambda function to a HttpStatus Code. Lambda Function doesn’t return HttpStatus Code, but you can map its exceptions to appropriate HttpStatus Codes using API Gateway.

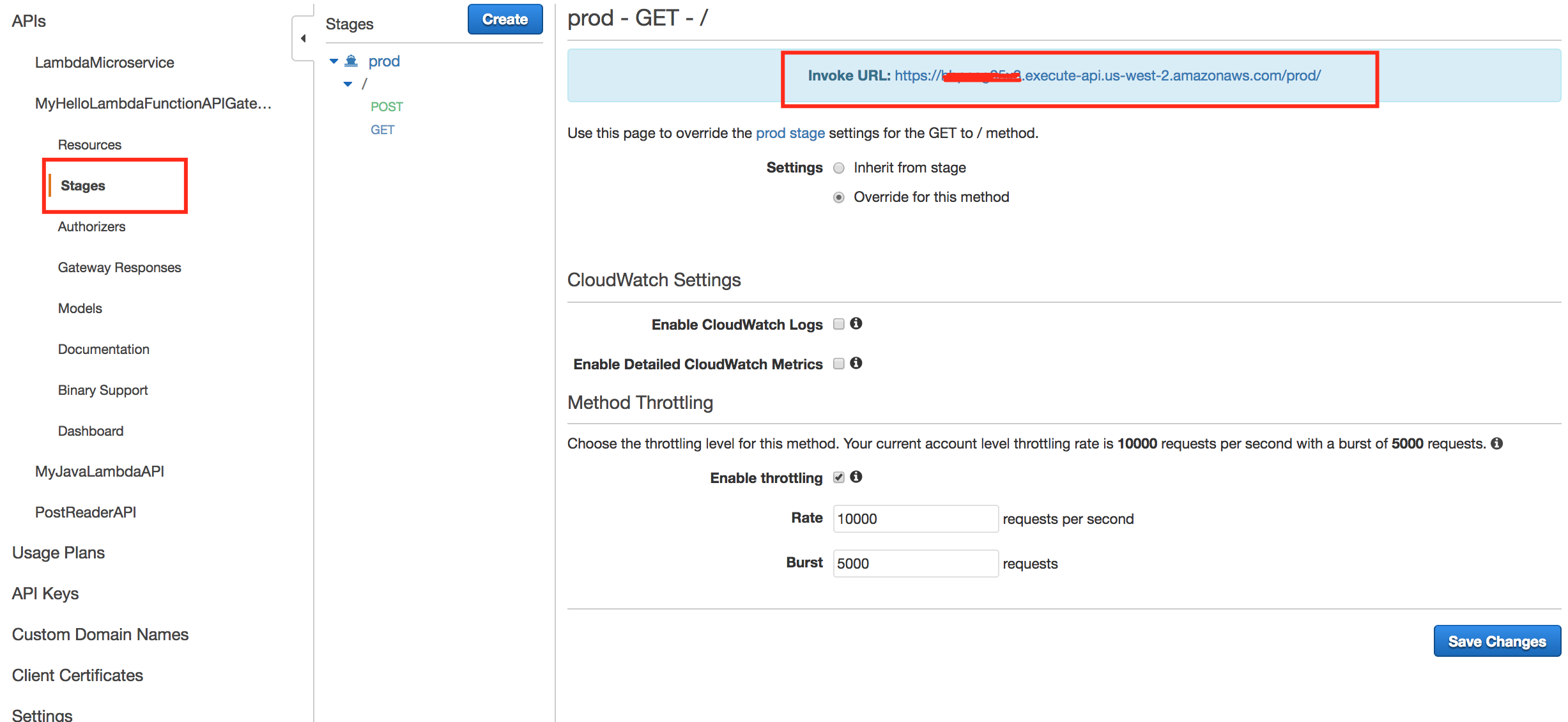




DON’T FORGET to deploy api



Retrieve endpoint url



try this endpoint using GET and ‘?myCount=1’ query param. Lambda function should return valid response and API Gateway should return HttpStatus Code 200 with lambda function’s response.

**Let’s try POST endpoint.**

Create a model of json that comes from client. Model is normally used to generate api documentation. It also helps you to create a mapping to convert client’s request to the request that should be submitted to lambda function. You can use a template that can be generated from model.

Below configuration will allow you to submit below body using POST endpoint of API Gateway

{

"someCount":-1

}

and it will convert to

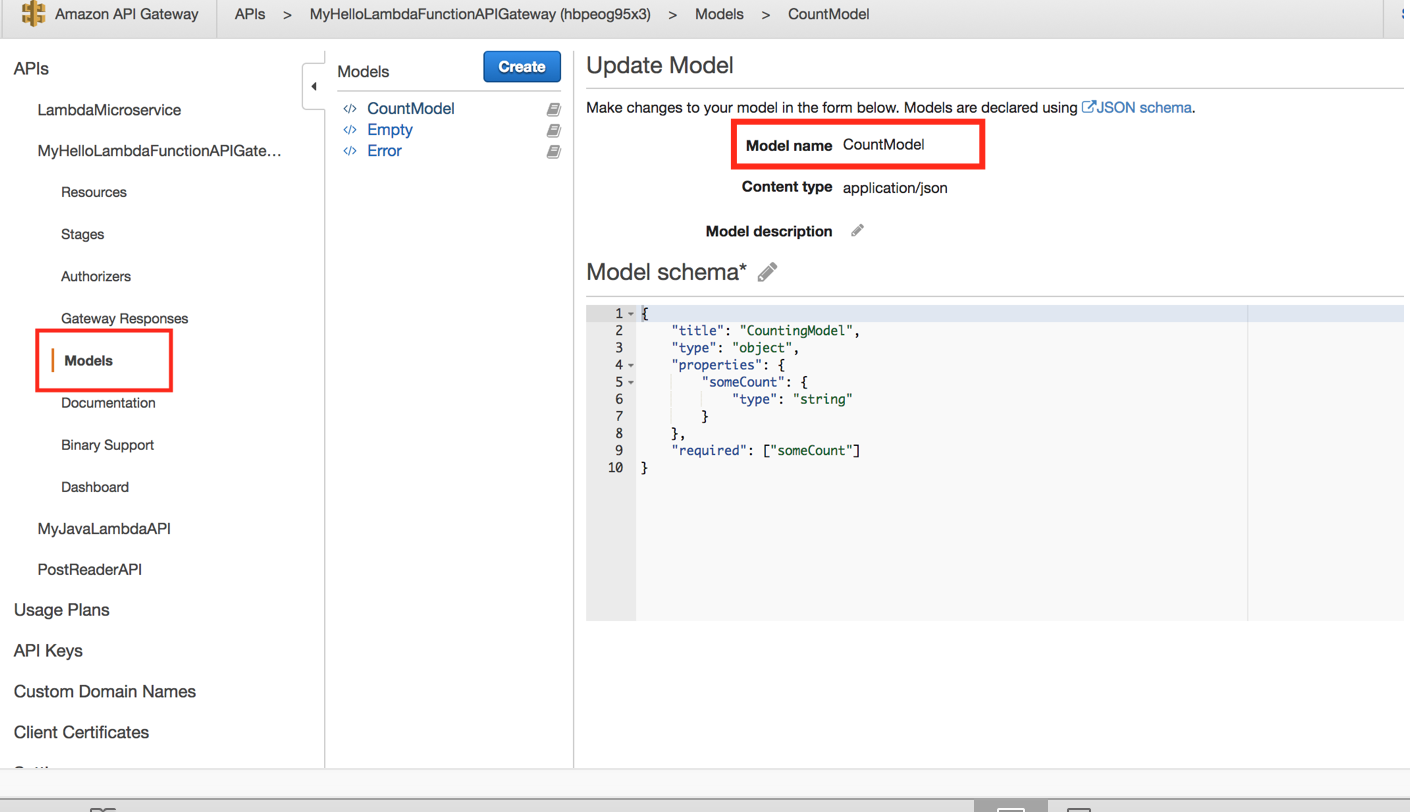
{

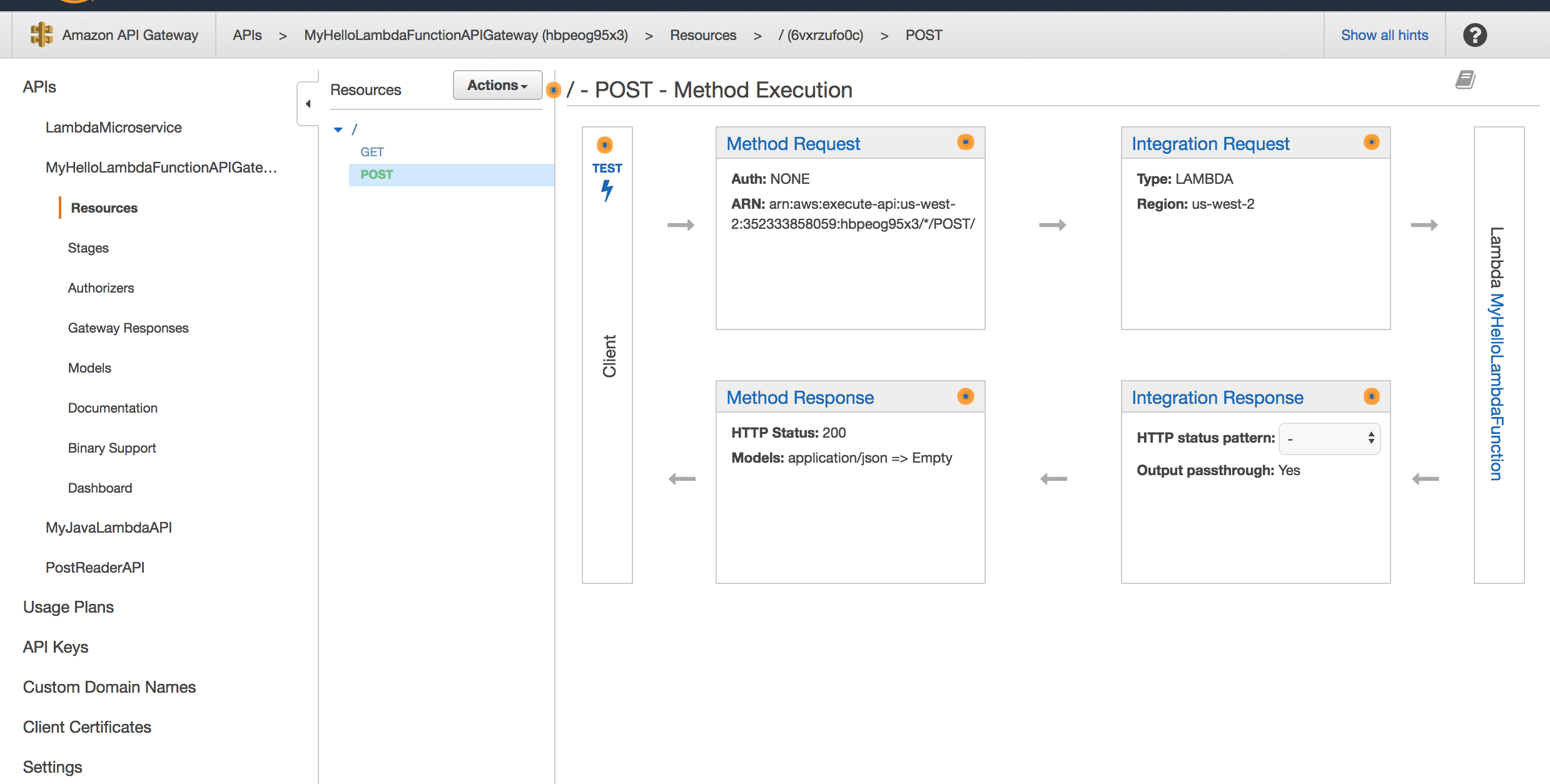
"myCount":-1

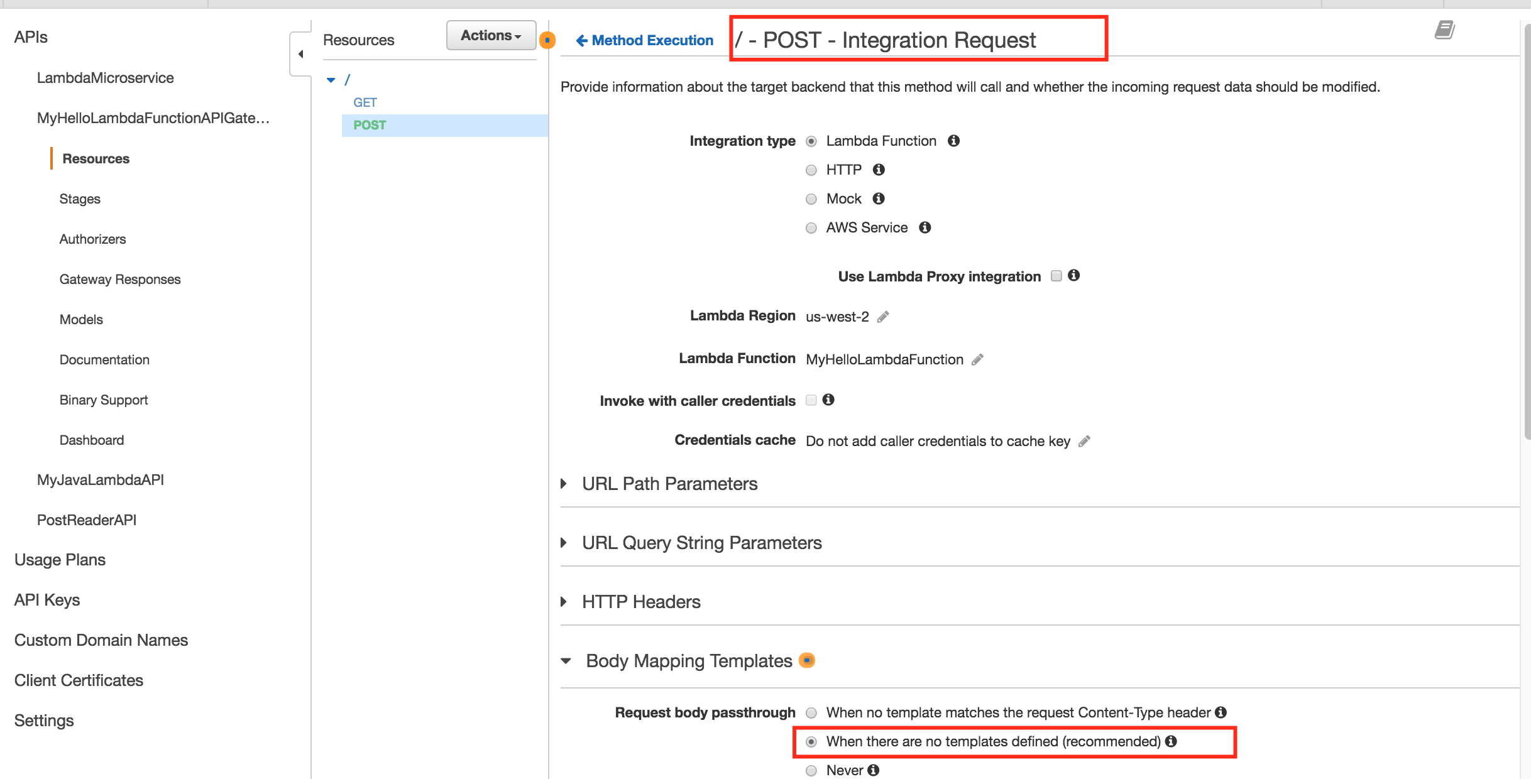
}

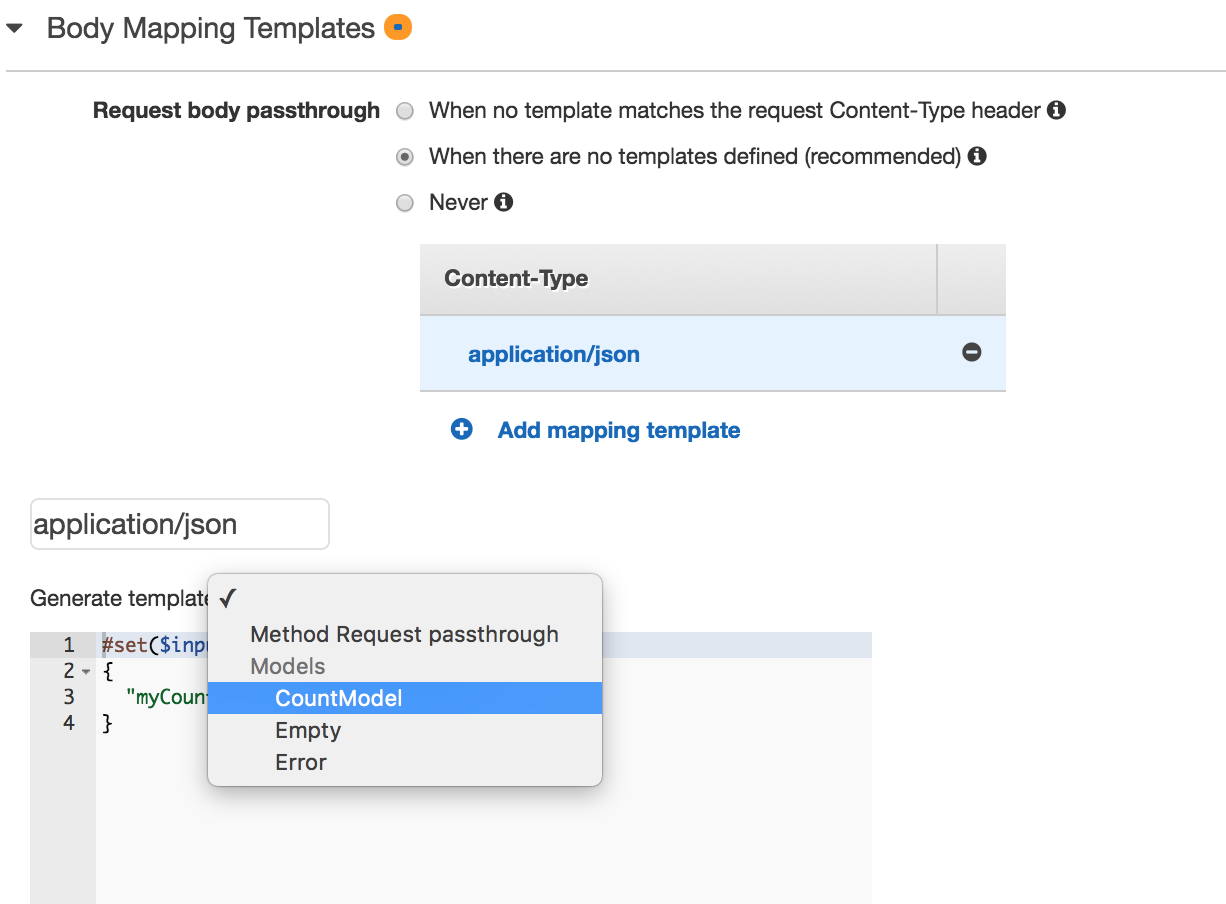
using mapping that you will create.

Let’s create a client request model first.

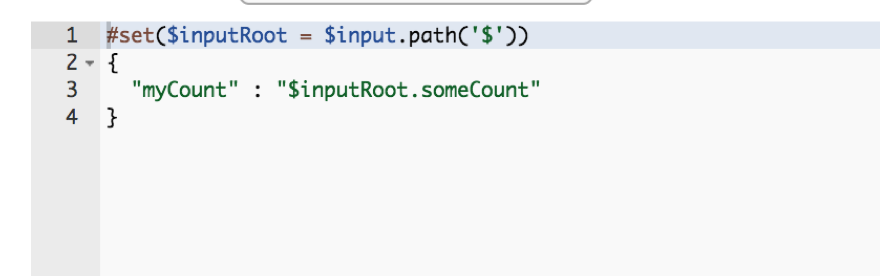








While creating this mapping, you can use the client request model that you created. This is not necessary, but it can give you some idea about how should your code look like. Change the template code as you need.



This code is converting client request body

{

"someCount":-1

}

to request body that should be submitted lambda function

{

"myCount":-1

}

Similar to GET endpoint as shown before in this document, you can map Lambda function’s exceptions to HttpStatus Codes.

DO NOT FORGET to deploy API after all these changes.

Retrieve API endpoint url just like how you did for GET endpoint.

