**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: Check your count again. It can't be <=0");

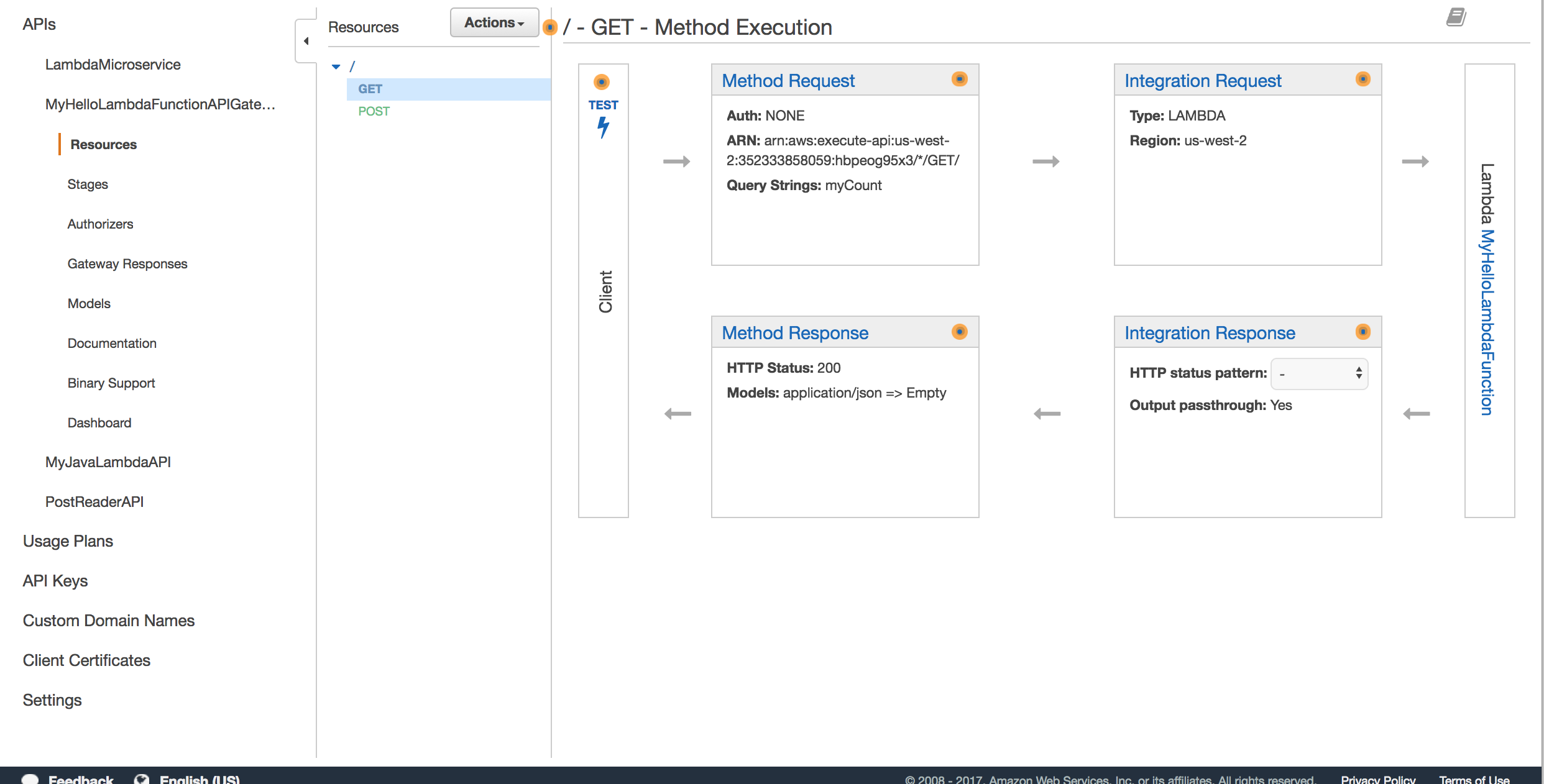
LambdaLogger logger = context.getLogger();

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

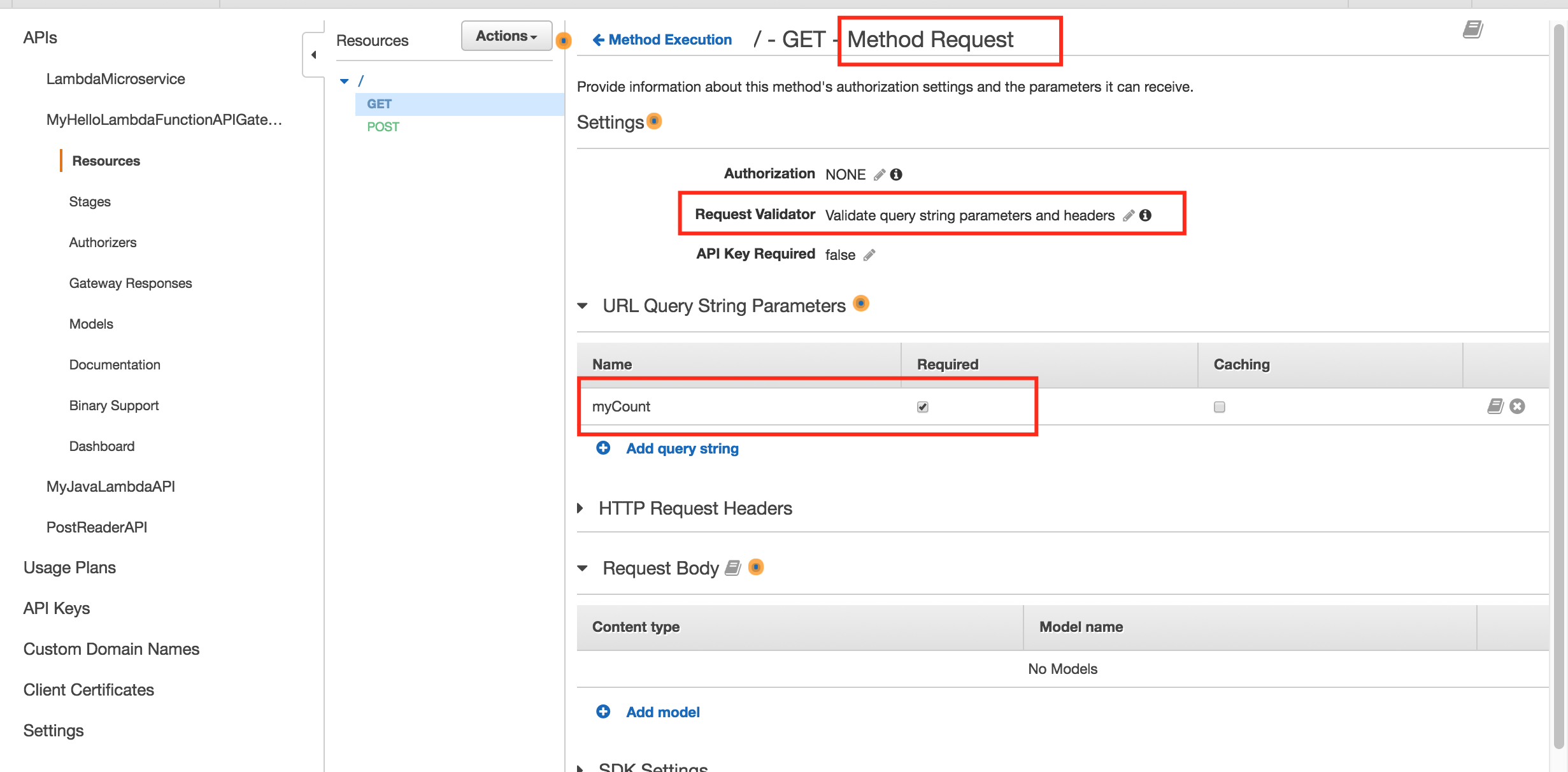
return String.valueOf(myCount);

}

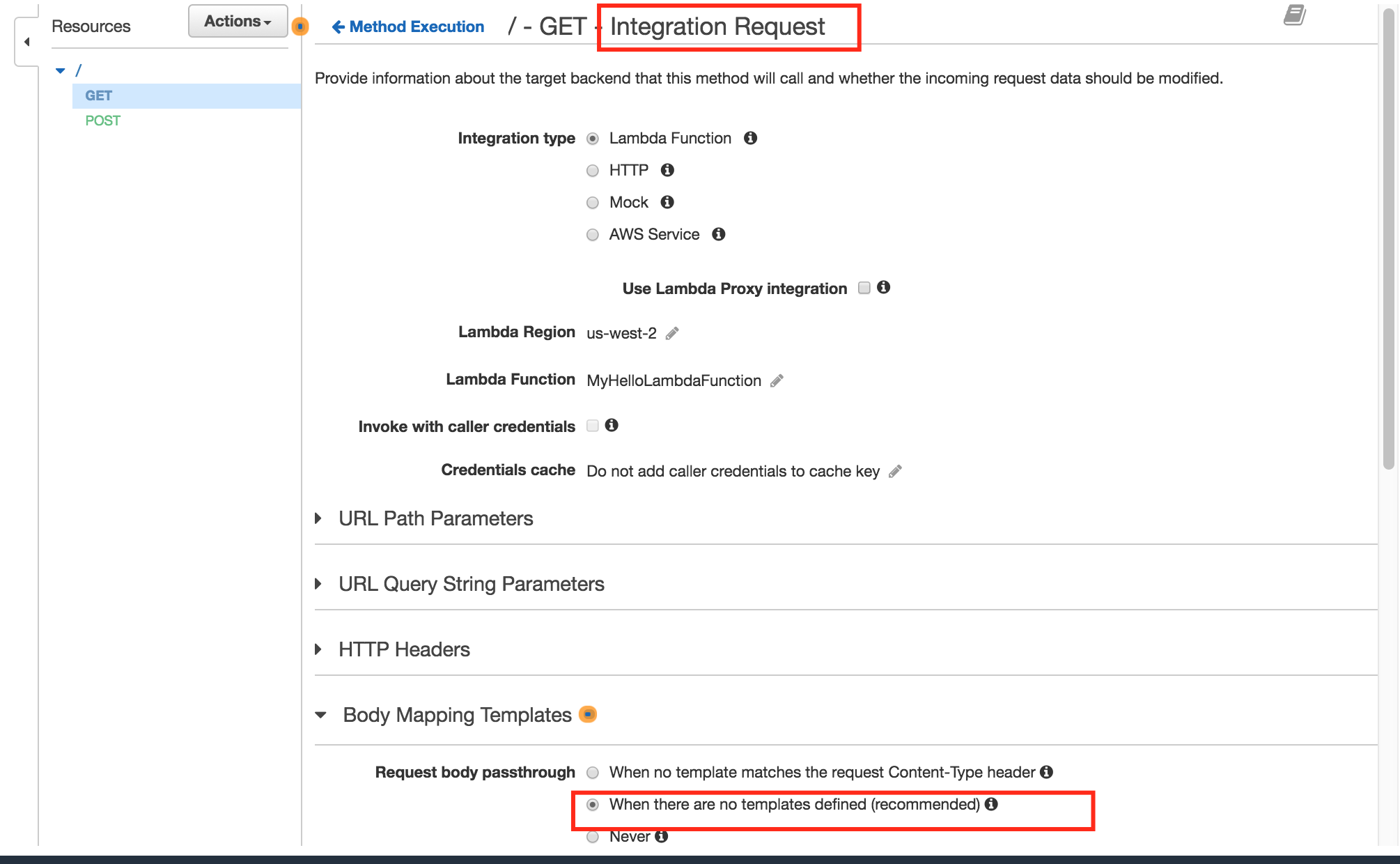
}

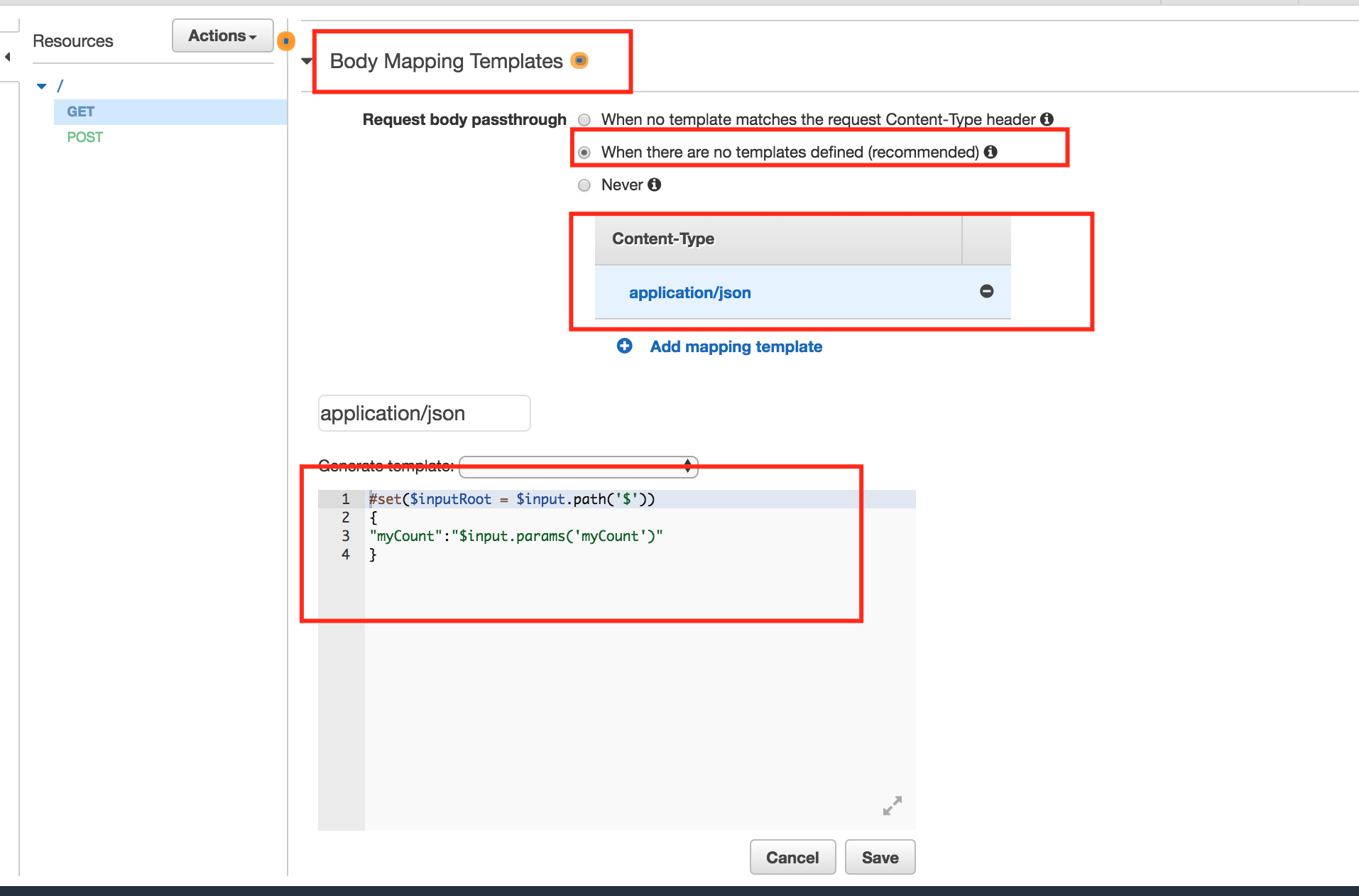


Add Query Param validation in ‘Method Request’ section.



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’.

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.

