2. Creating a Serverless API

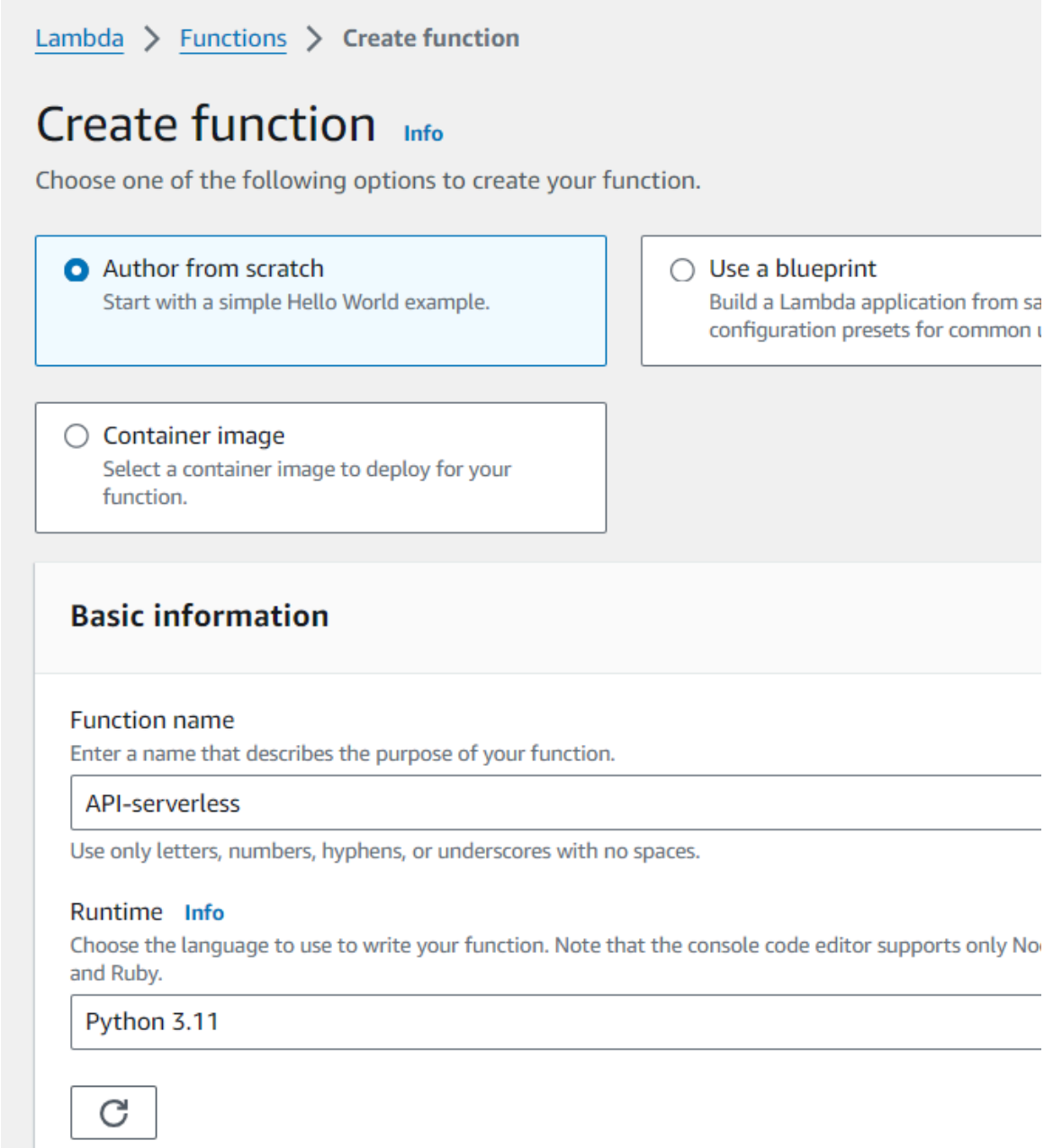
Objective: Develop a serverless API using AWS Lambda and API Gateway.

Approach:

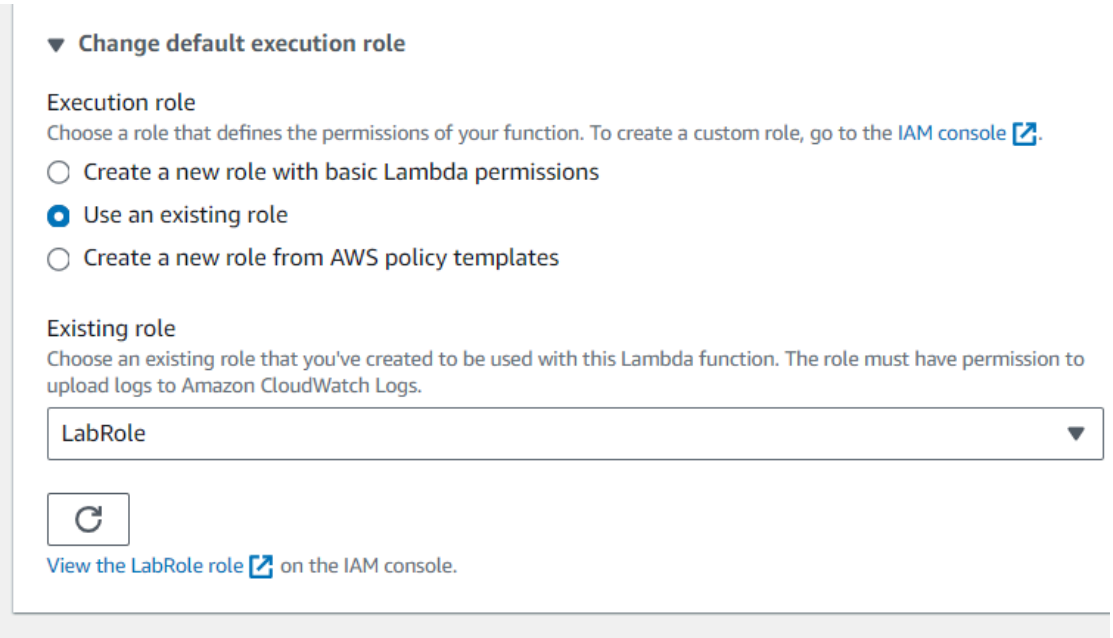
- **Define API**: Design a simple RESTful API (e.g., for a todo list application).
- **Lambda Functions**: Create Lambda functions for each API method (GET, POST, PUT, DELETE).
- **API Gateway Setup**: Use API Gateway to set up the API endpoints, connecting each endpoint to the corresponding Lambda function.
- **Testing**: Test the API using tools like Postman or AWS API Gateway test functionality.

Goal: Gain hands-on experience in building and deploying a serverless API, understanding the integration between Lambda and API Gateway.

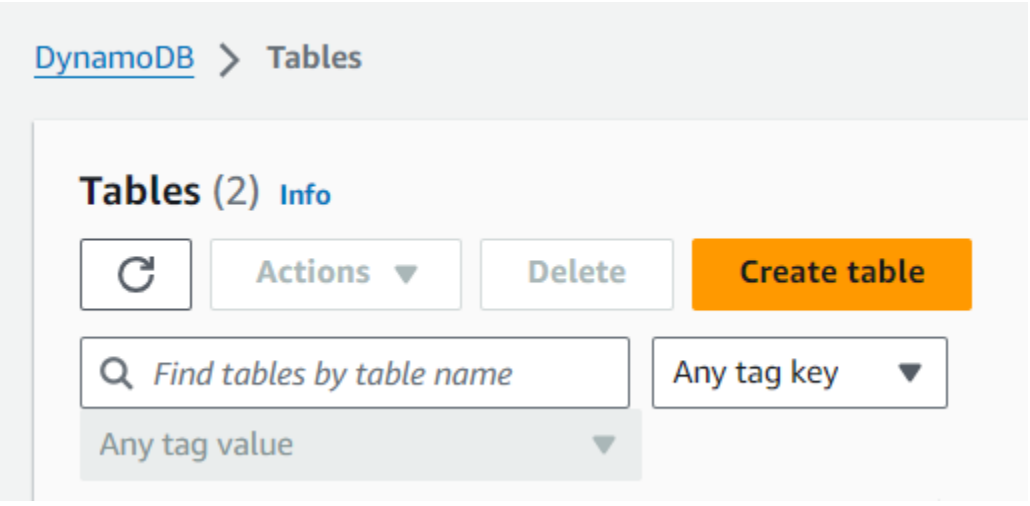
1) **Create Lambda function**



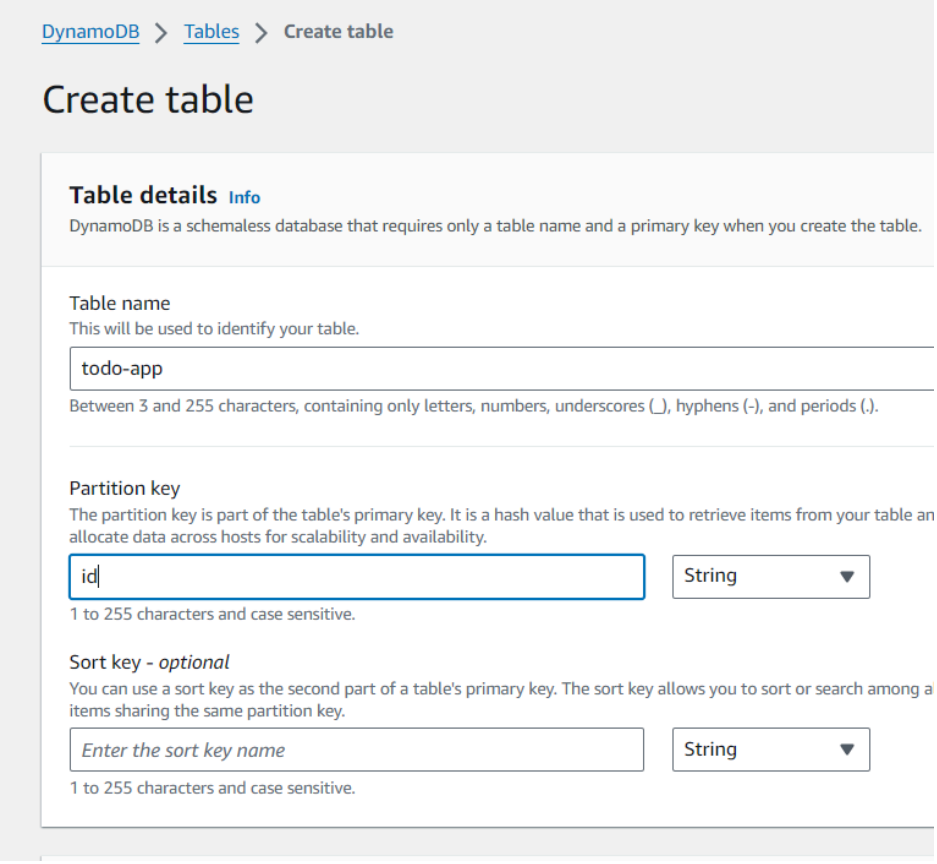
2) Assign Lambda function permission



3) Create Table in DynamoDB

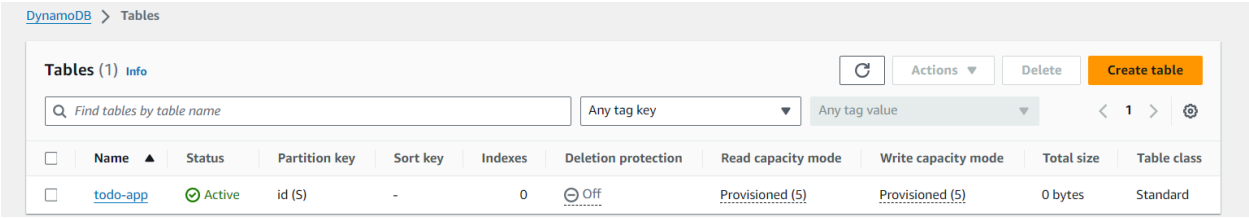


4) **Provide table-name and donot forget to give partition key.**

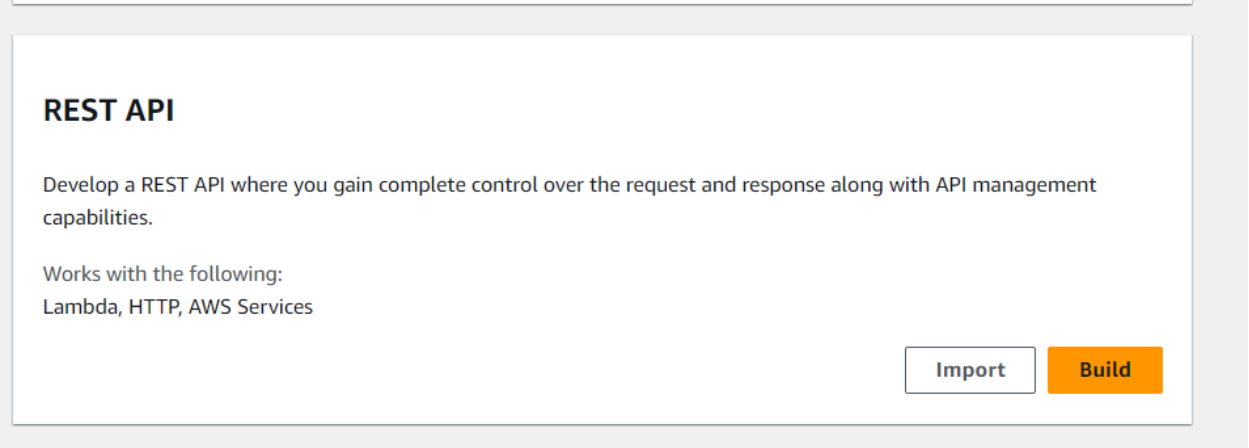


5)

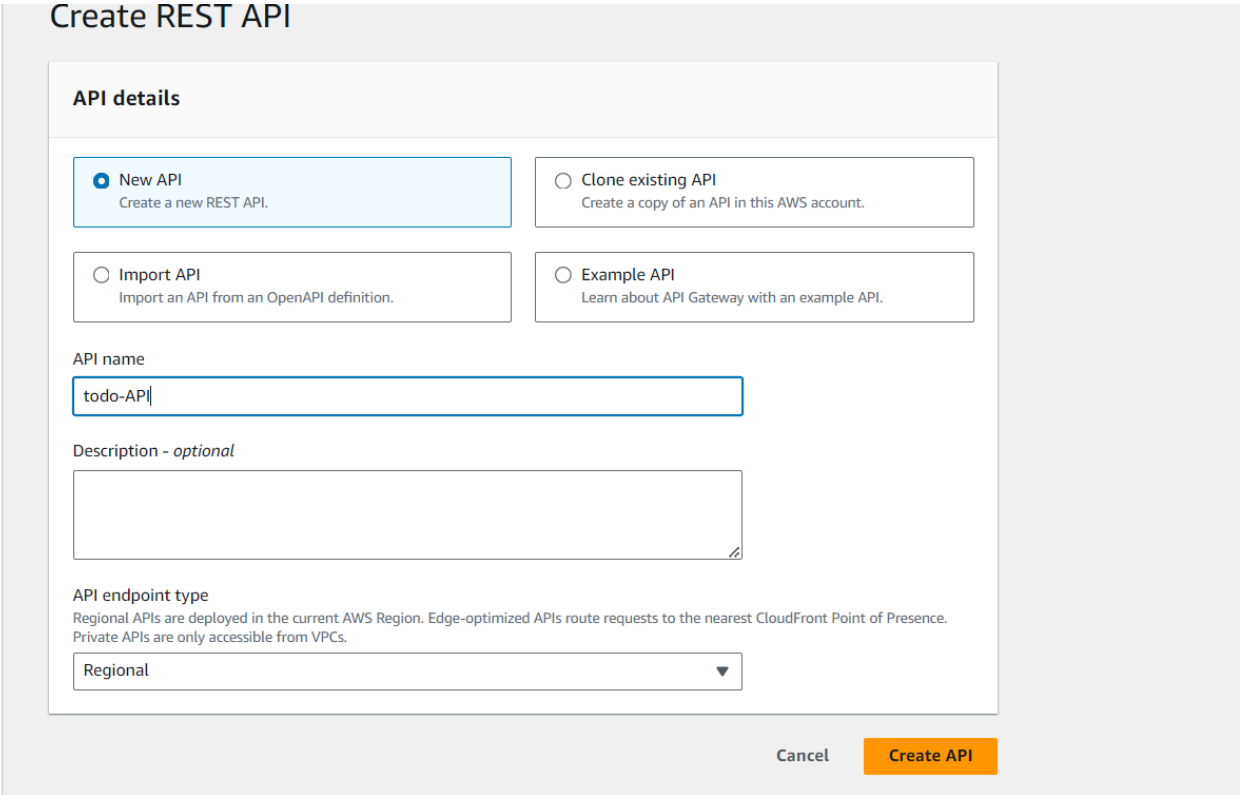
6) **Table has been successfully created.**



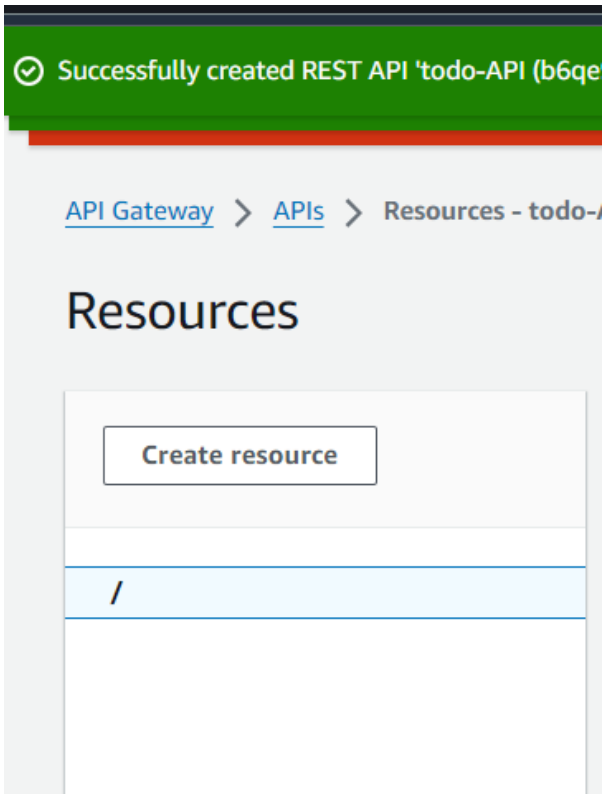
7) Build RestAPI



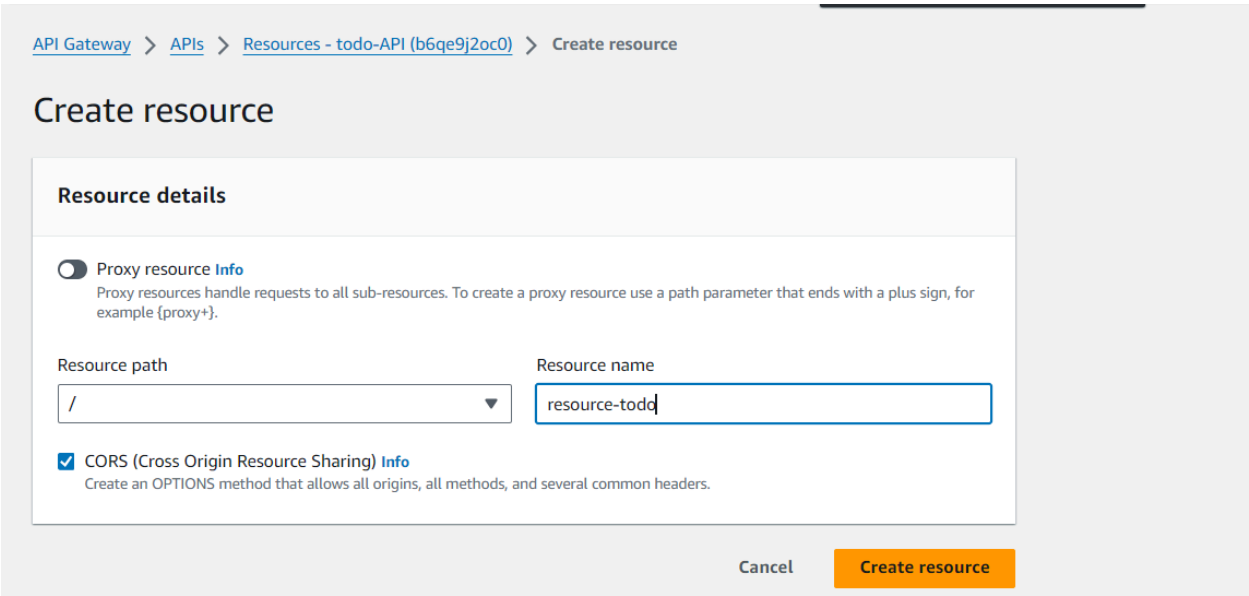
8) Create API with the API name.



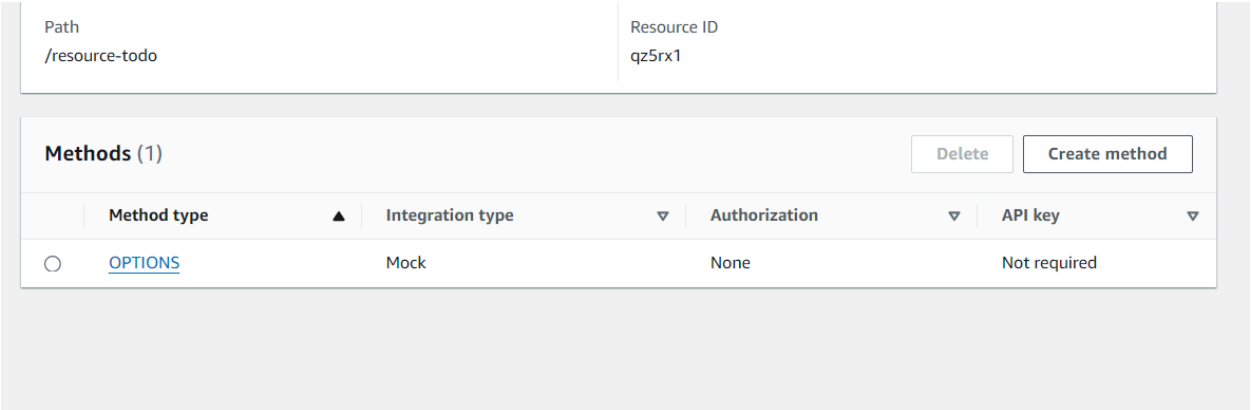
9) **After successfully creation of API,add resource.**



10) **Add resource name and enable CORS.**

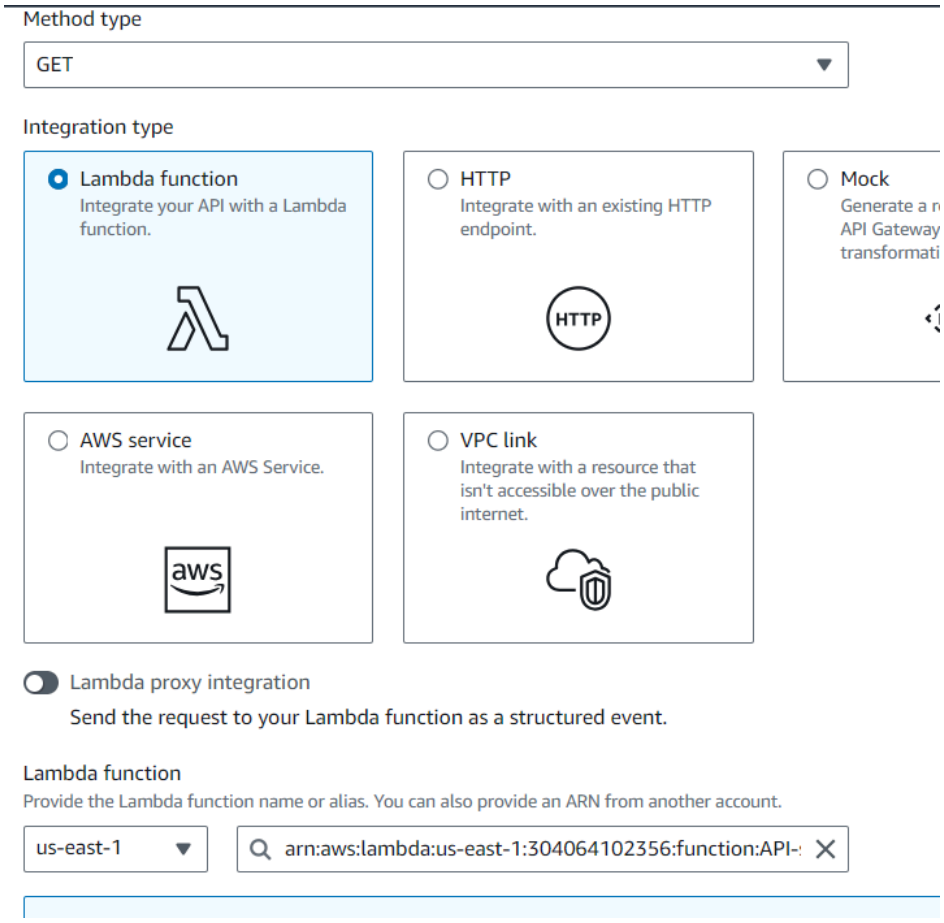


11) Now after successful creation of resources, methods has to be created.

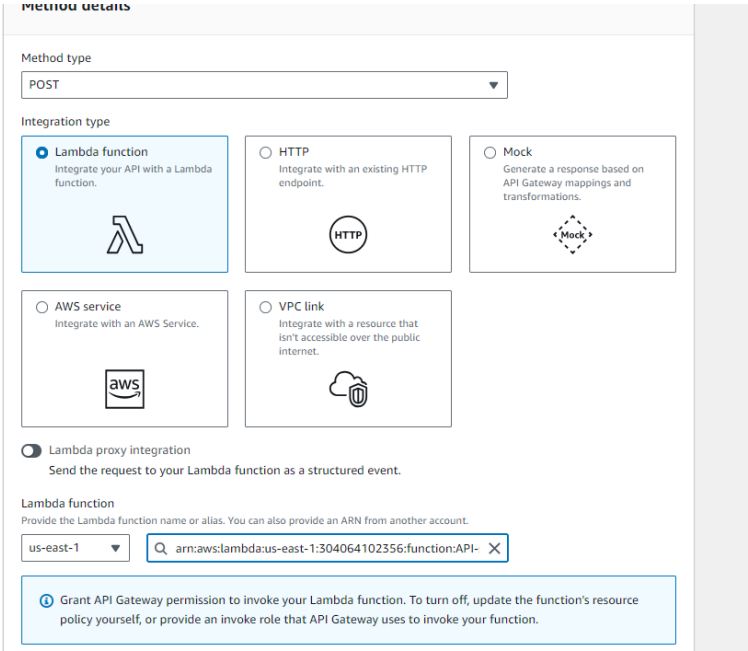


12) Lambda function created in previous steps are used.

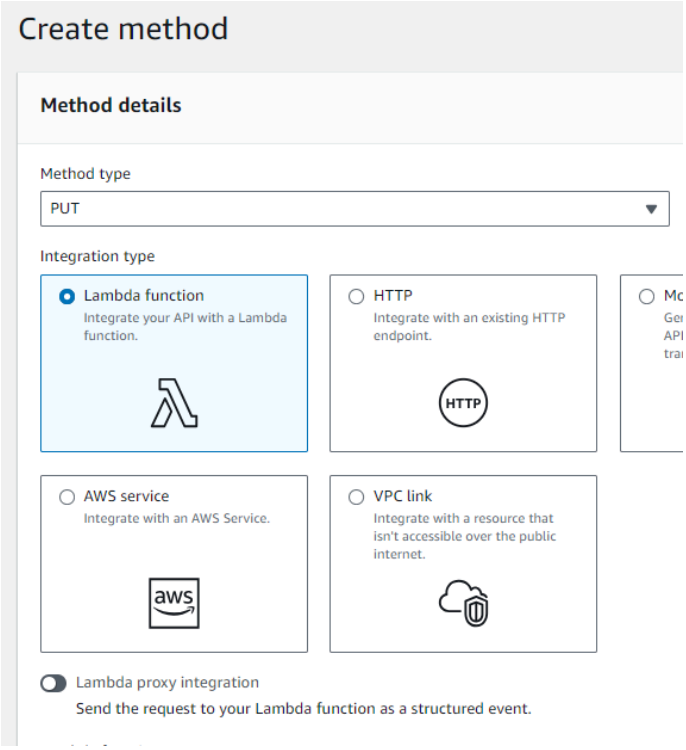
For GET method



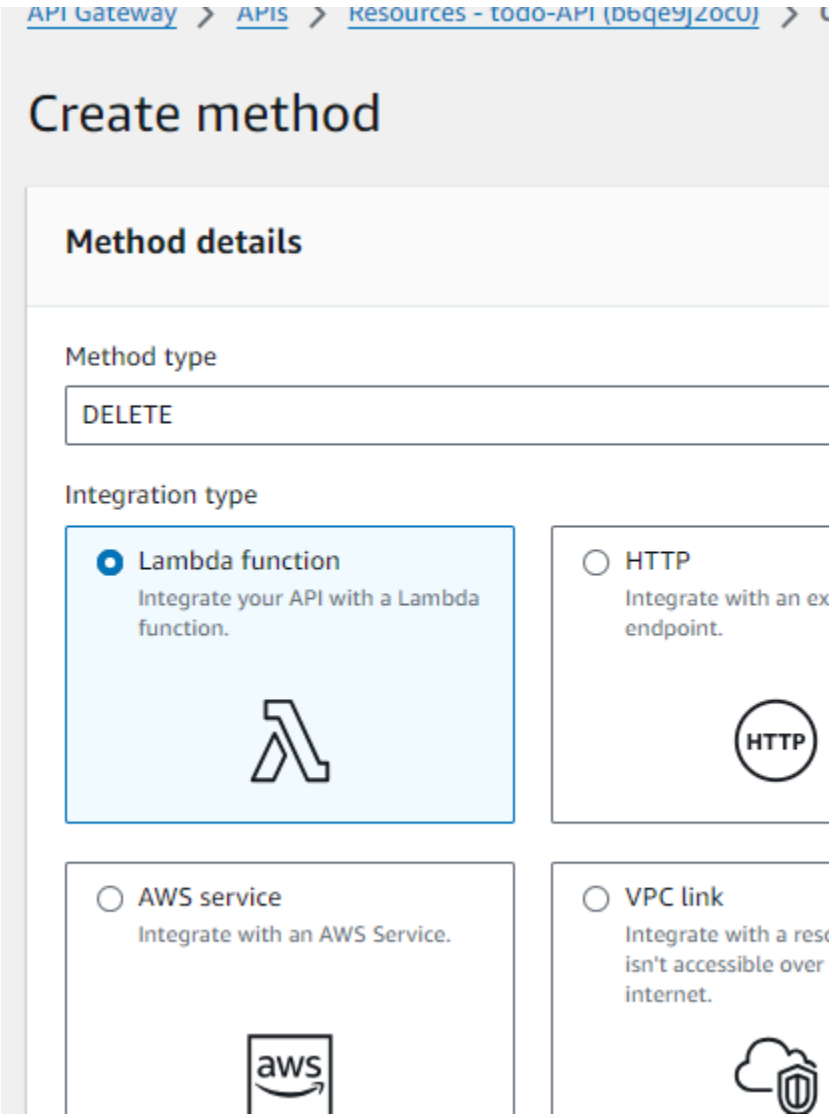
For POST Method



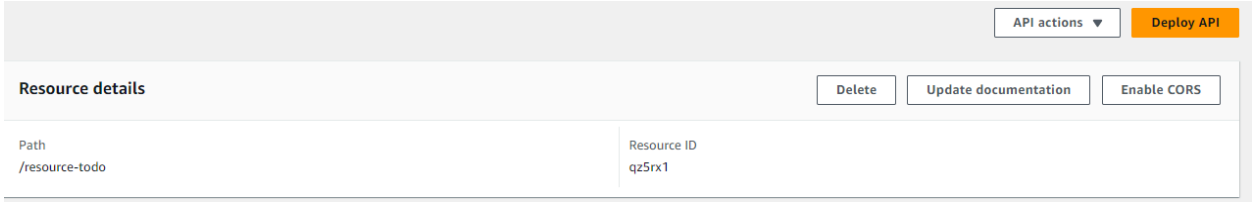
For PUT method



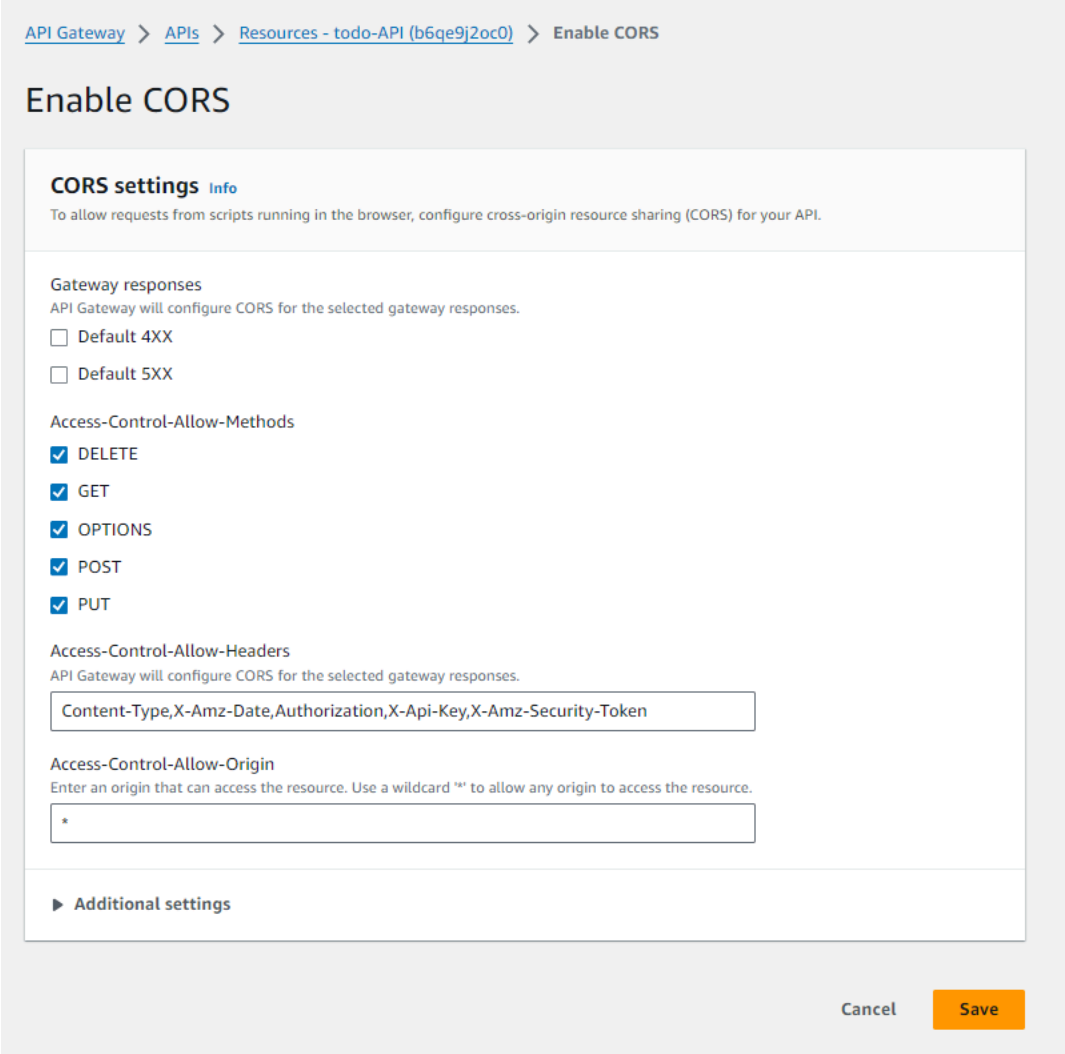
For Delete Method



13) Enabling CORS.

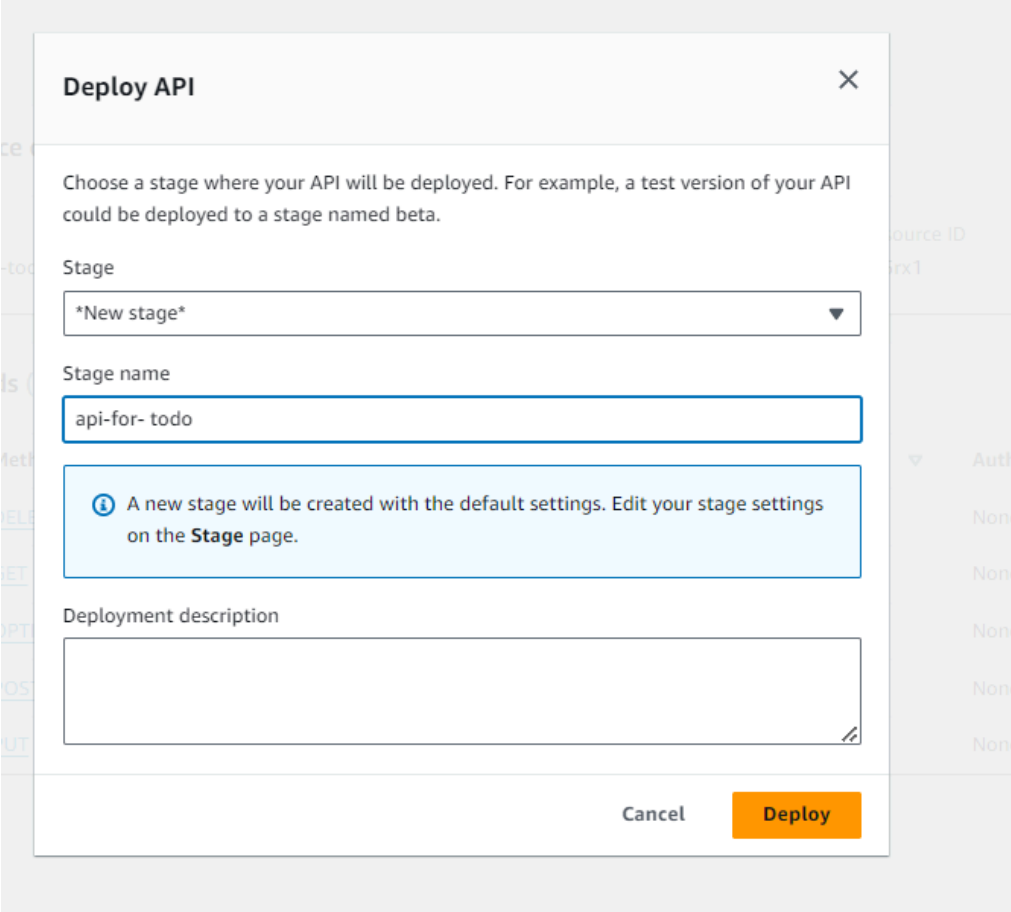


14) For all of the methods, CORS are enabled.

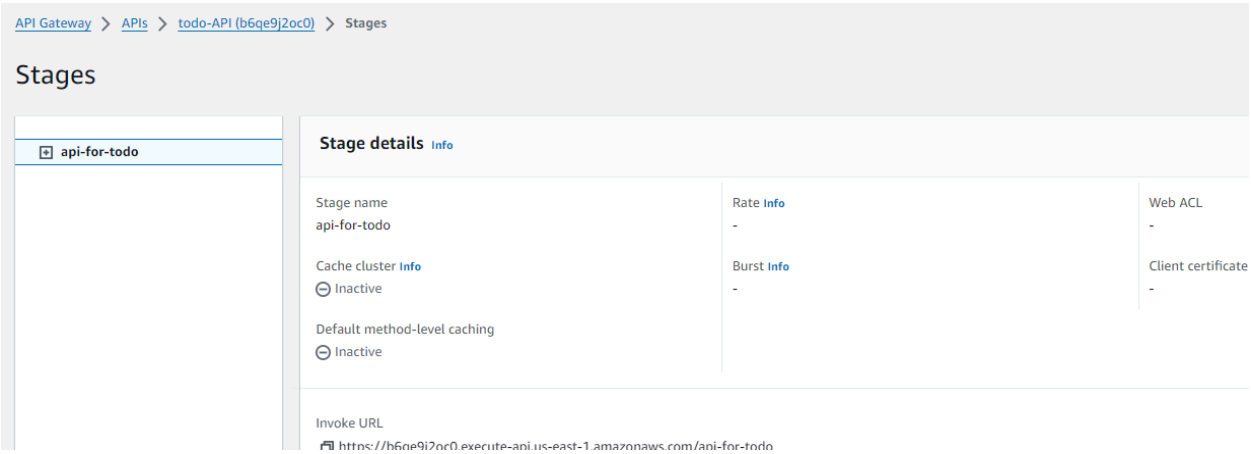


15) Now Click on Deploy API button on right-most side.

16) A new dialog box appears. Give stage name



17) Create a new stage



18) Testing for POST method.

For pending status(test-1)

Request body

```
1 {
2     "httpMethod":"POST",
3     "id": "2",
4     "task":"Serverless lab",
5     "status":"Pending"
6 }
```

Test

/resource-todo - POST method test results

Request

/resource-todo

Latency

247

Response body

```
{"statusCode": 200, "message": "New method has been added"}
```

For Completed Status(test-2)

Request body

```
{
    "httpMethod":"POST",
    "id": "3",
    "task":"Serverless lab",
    "status":"Completed"
}
```

Test

/resource-todo - POST method test results

Request
/resource-todo

Latency
24

Response body

```
{"statusCode": 200, "message": "New method has been added"}
```

For Empty Status

Request body

```
{
    "httpMethod":"POST",
    "id": "4",
    "task":"Server",
    "status":"Empty"
}
```

Test

/resource-todo - POST method test results

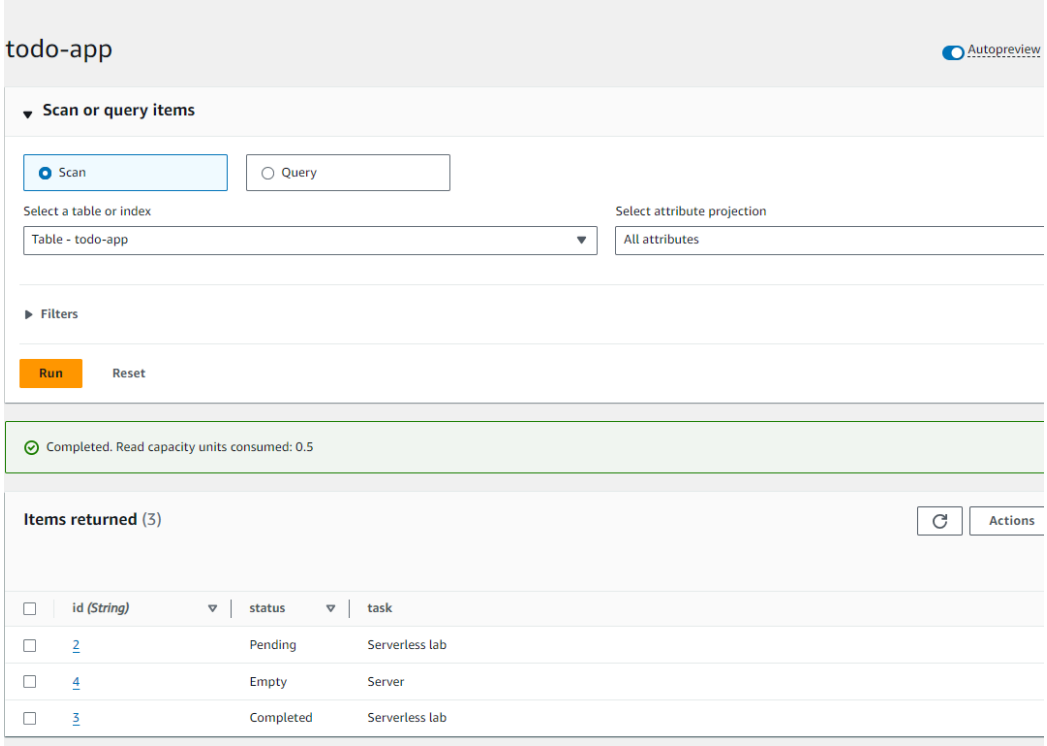
Request
/resource-todo

Latency
72

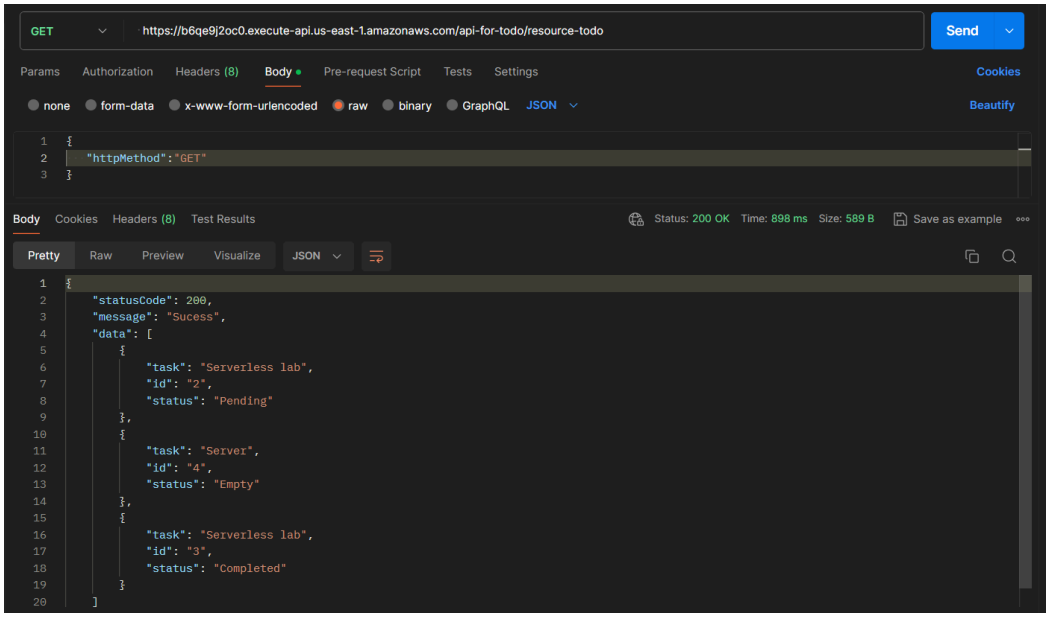
Response body

```
{"statusCode": 200, "message": "New method has been added"}
```

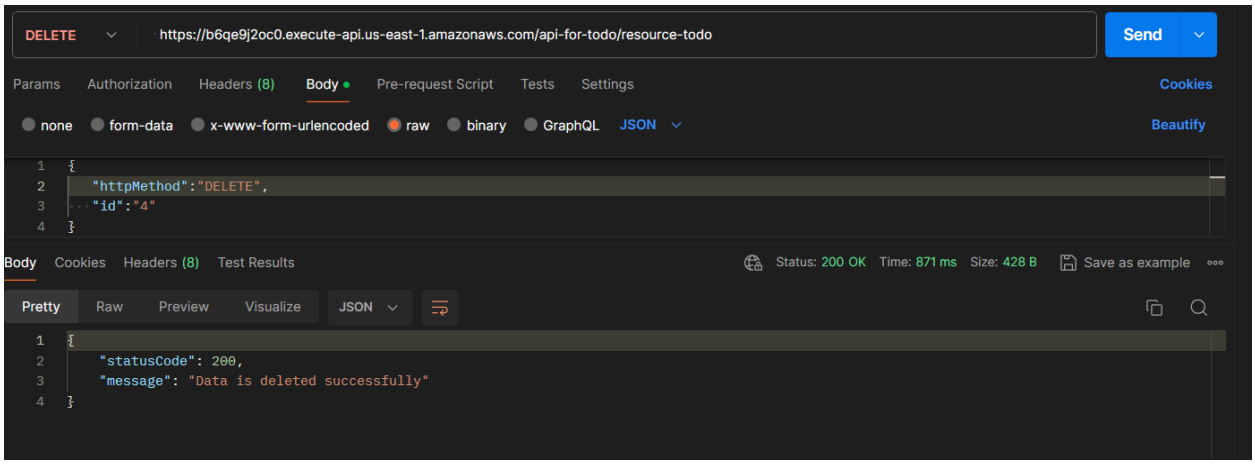
19) Successful POST method. Its seen under items returned in DynamoDB table.



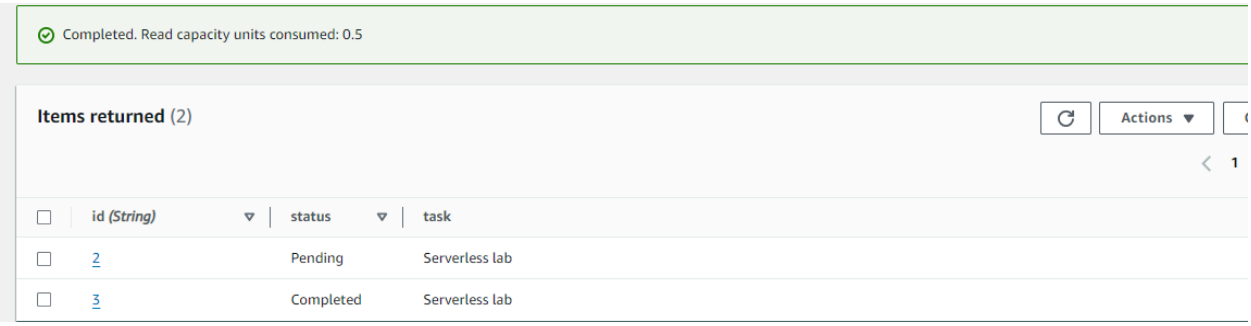
20) Testing GET method in Postman



21) Testing DELETE Method



22) After deleting, the results in table is:



23) Testing PUT Method

Request body

```
{
    "httpMethod":"PUT",
    "id":"2",
    "key":"score",
    "new_value":"your score is 90"
}
```

Test

/resource-todo - PUT method test results

Request	Latency	Status
/resource-todo	77	200

Response body

```
{"statusCode": 200, "message": "Task Updated"}
```

Response headers

The results in DynamoDB table is shown as:

Items returned (1)

id *(String)*	score	status	task
3	your score is 90	Completed	Serverless lab