Creating a Serverless API

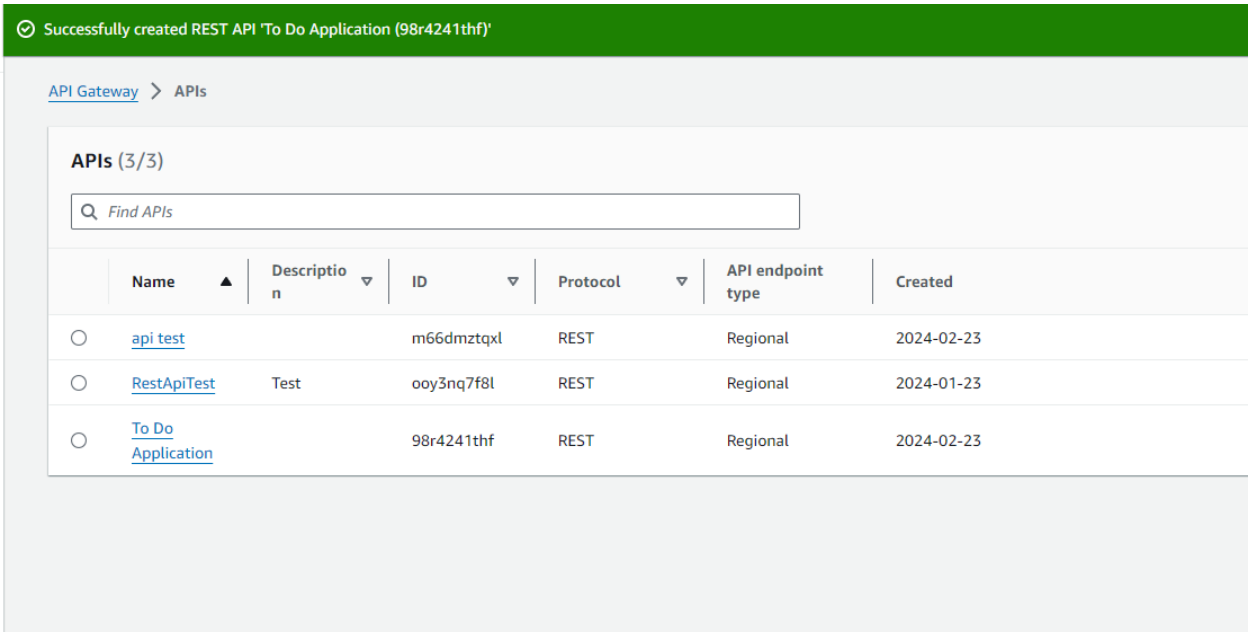
Objective: Develop a serverless API using AWS Lambda and API Gateway.

Approach:

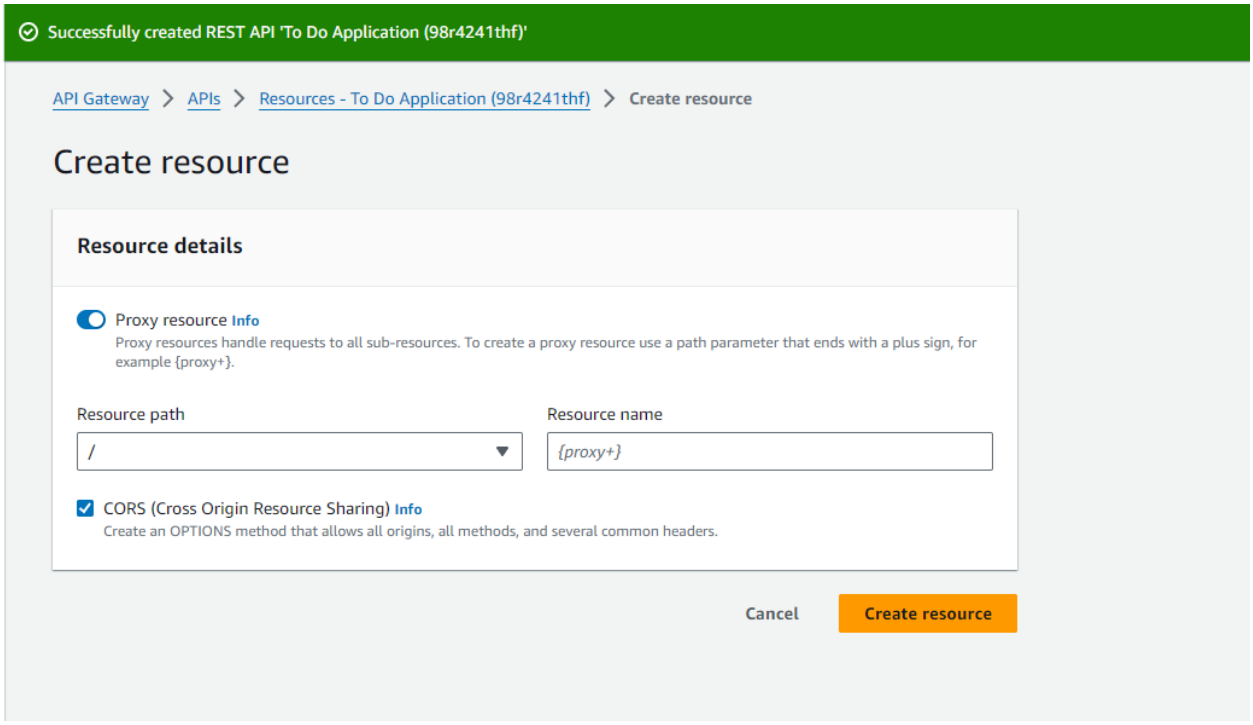
- **Define API**: Design a simple RESTful API (e.g., for a todo list application).
- **Lambda Functions**: Create Lambda functions for each API method (GET, POST, PUT, DELETE).
- **API Gateway Setup**: Use API Gateway to set up the API endpoints, connecting each endpoint to the corresponding Lambda function.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

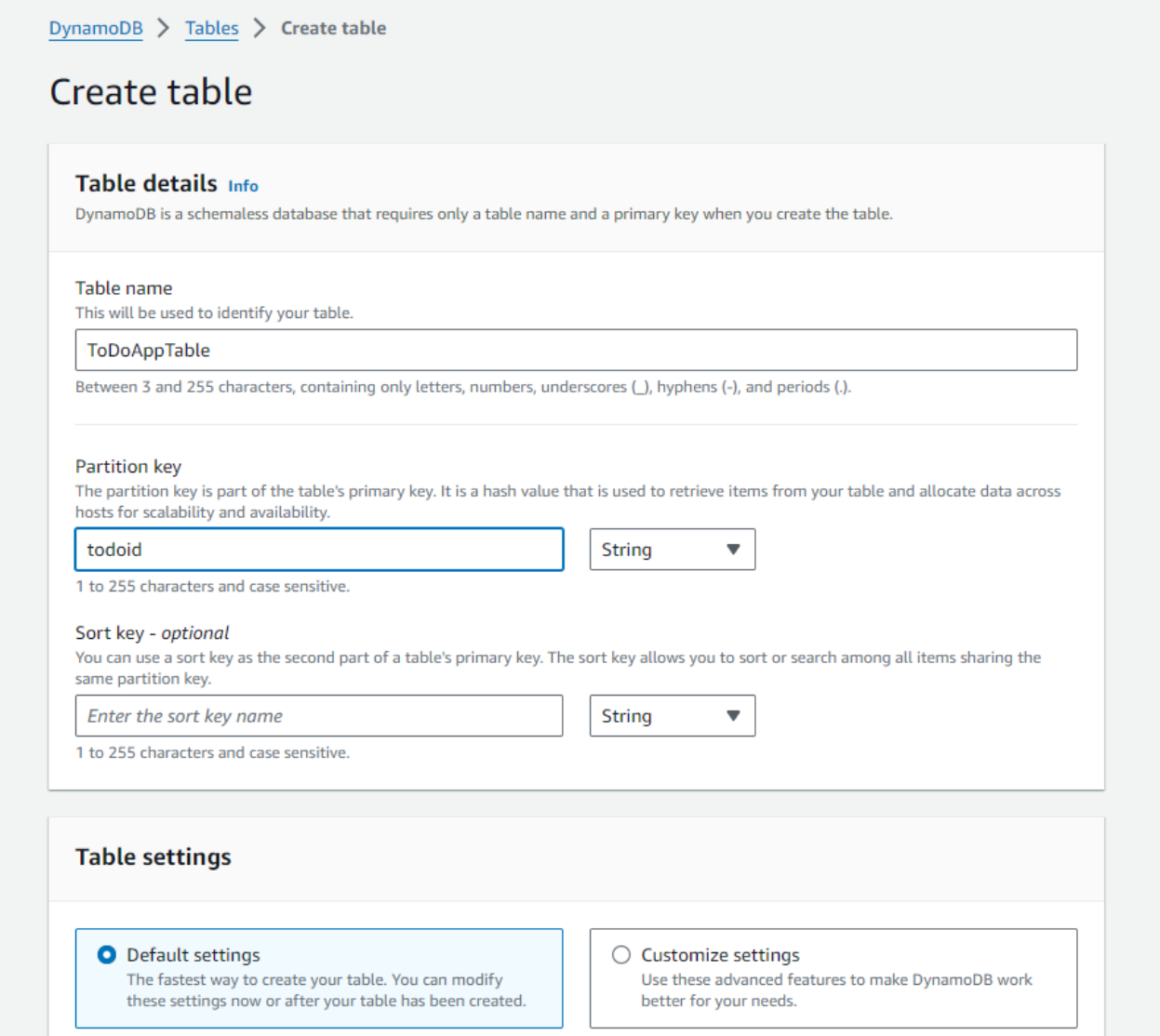
1. First of all we go the api getway and create an api to-do-application.



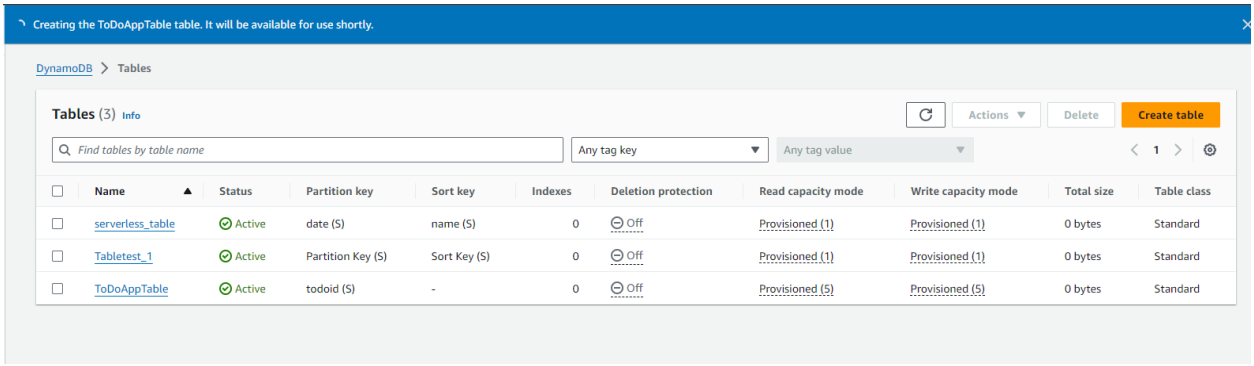
2. Create a new resource with following methods and using proxy resources on.



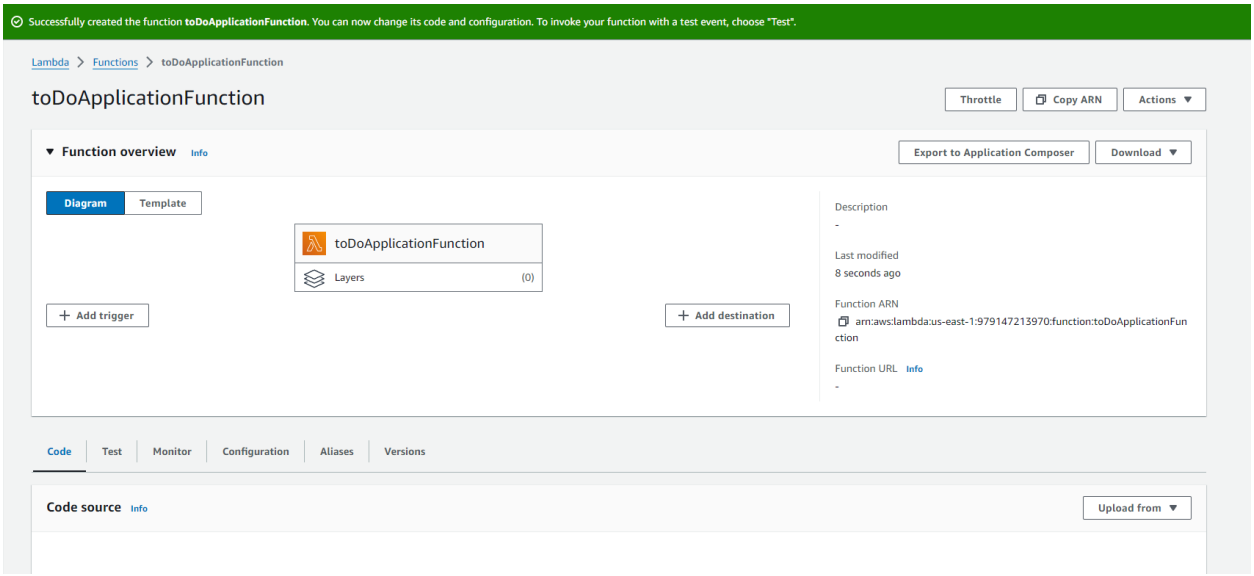
3. After that we have to create a new table from dynamo db from the left control panel of the aws menus.



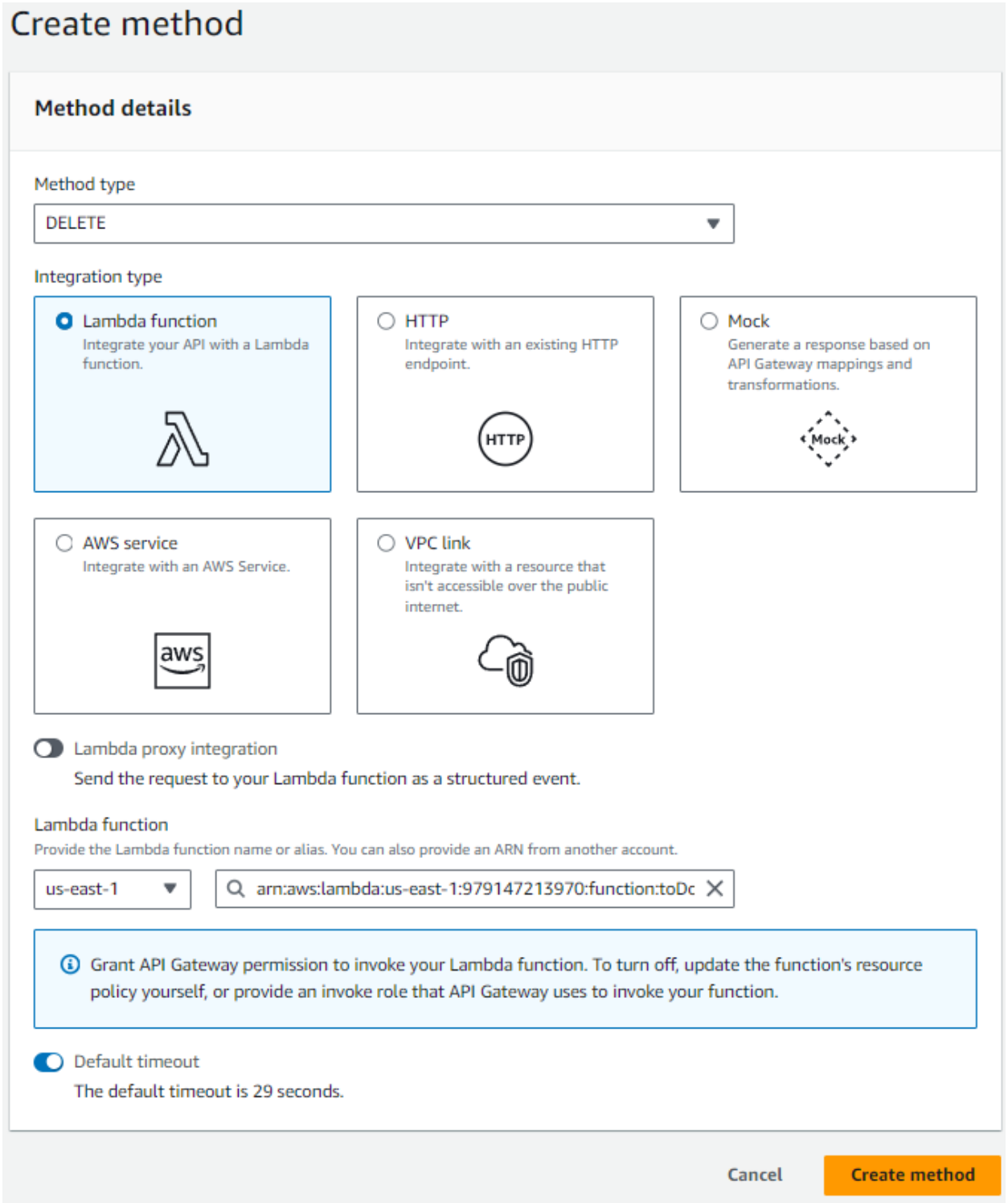
4. We can see the following table after the table is created.

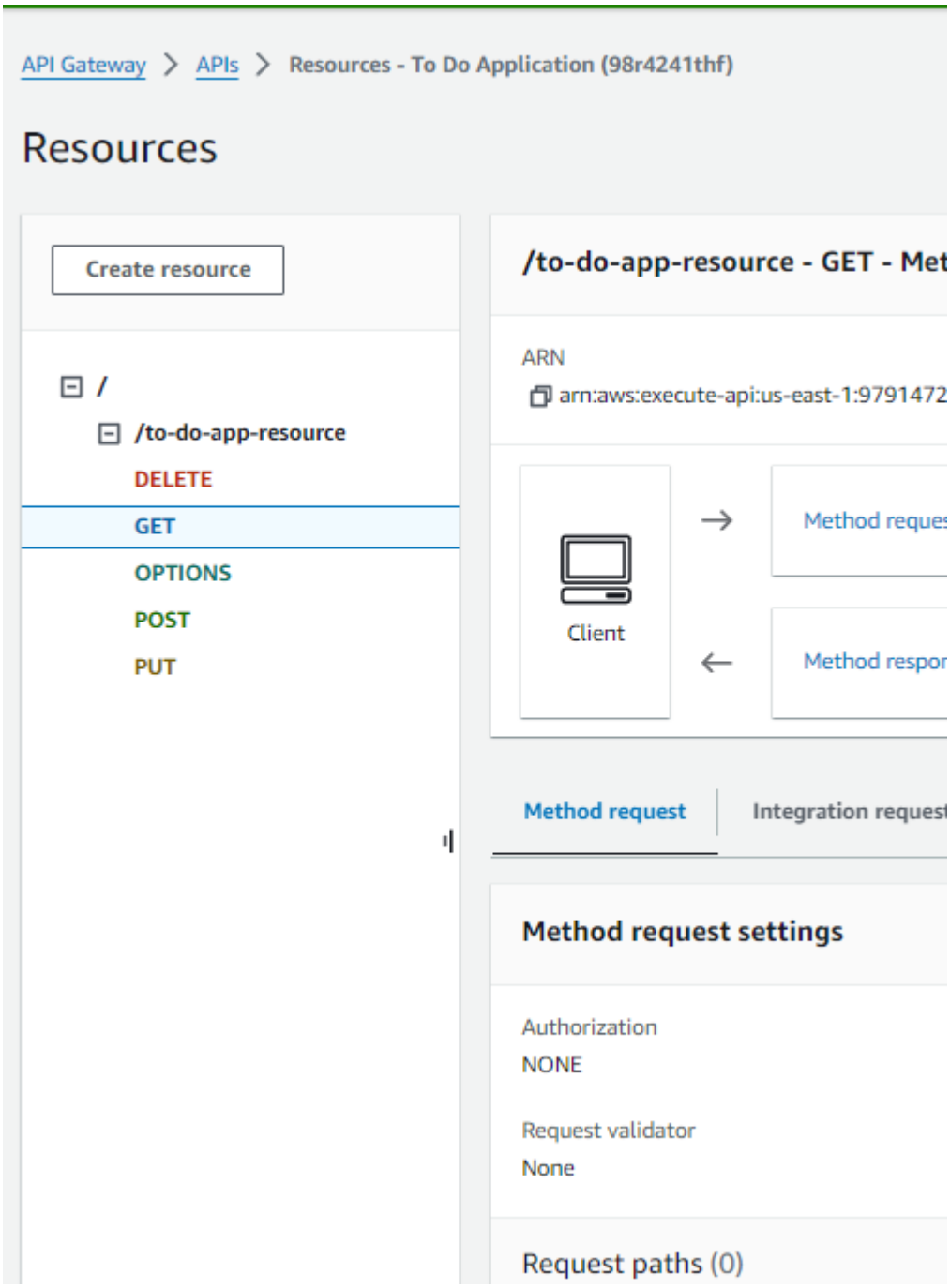


5. Then create a new lambda function for the to do app.

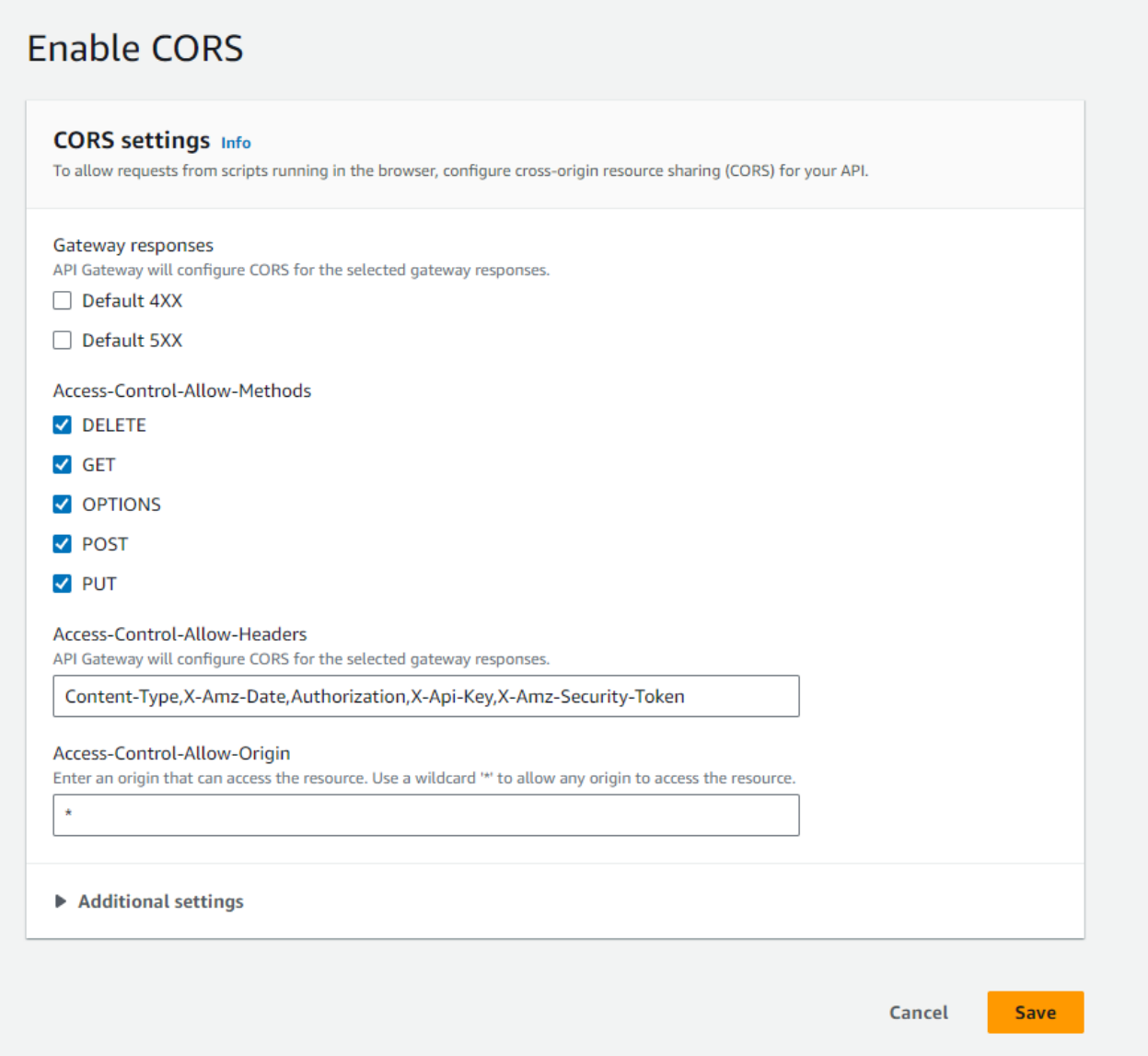


6. Then we have to create the 4 methods GET, PUT, POST, DELETE and it should be inside our todoApplication.

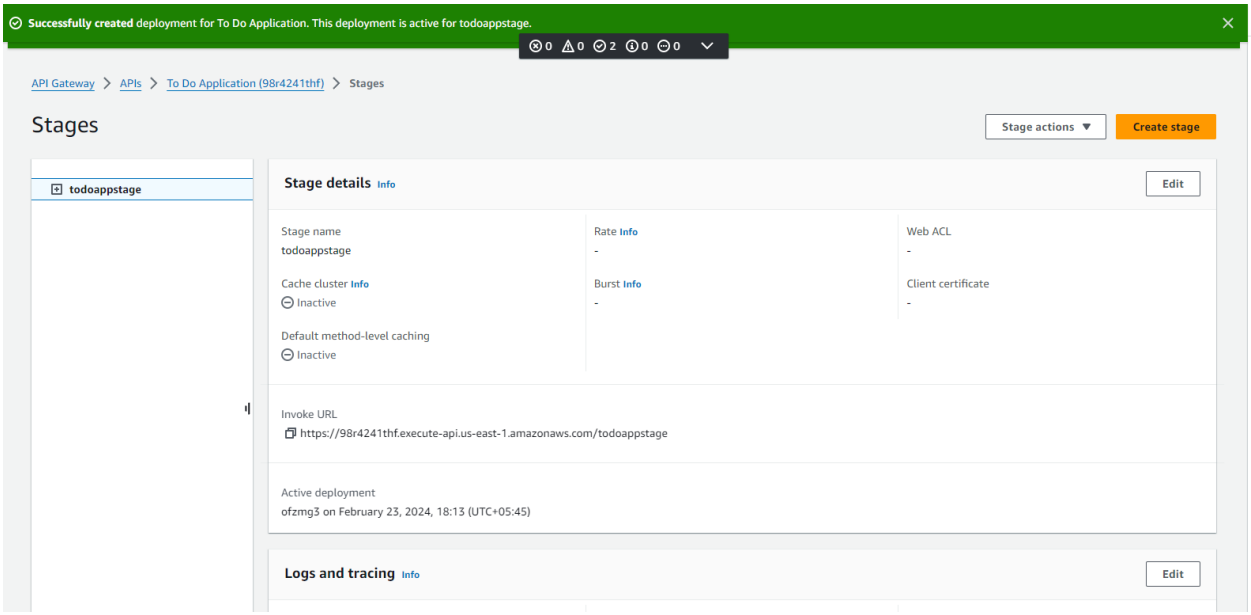



API Gateway
APIs
Resources - To Do Application (98r4241thf)
Resources
Create resource
/
/to-do-app-resource
DELETE
GET
OPTIONS
POST
PUT
/to-do-app-resource - GET - Met
ARN
arn:aws:execute-api:us-east-1:9791472
Client
Method reques
Method respor
Method request
Integration request
Method request settings
Authorization
NONE
Request validator
None
Request paths (0)

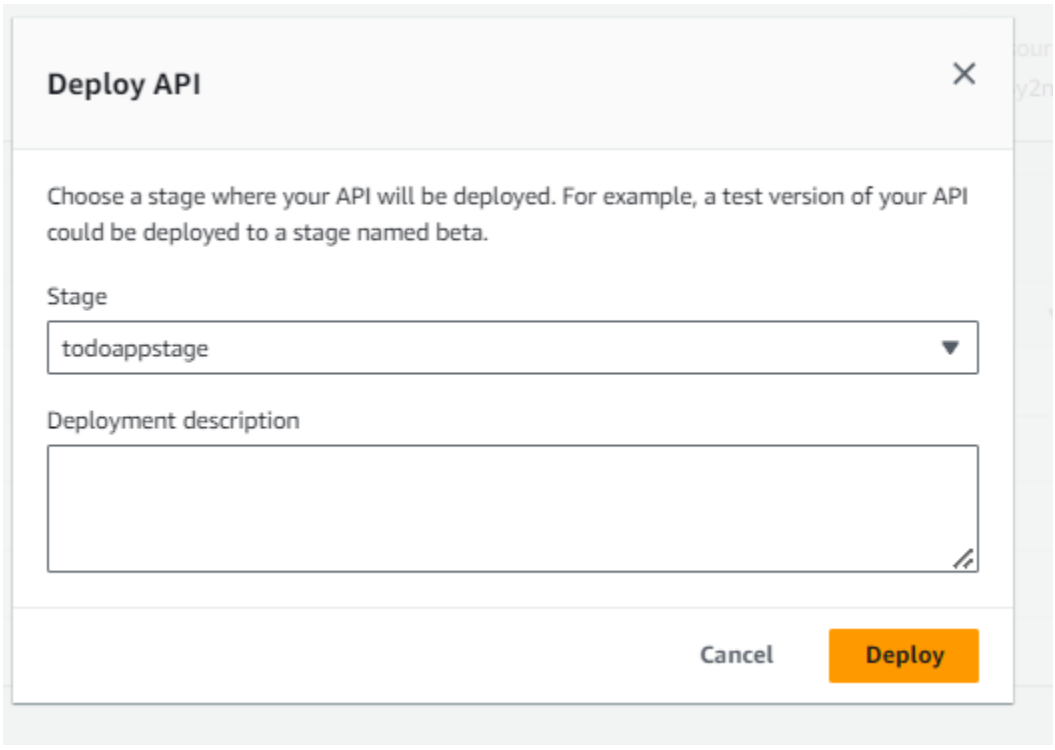
7. Enable cors.



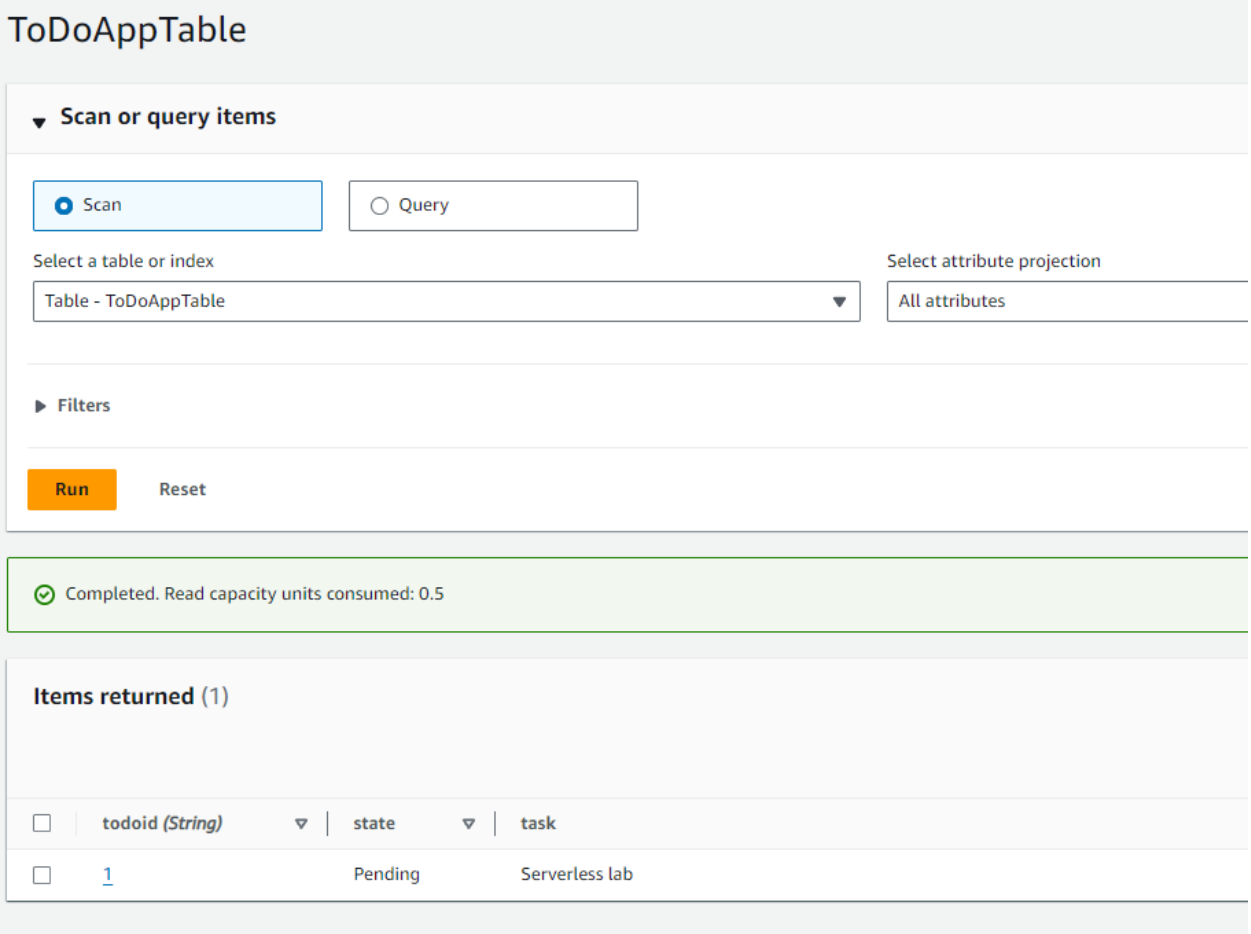
8. Create a new stage.



8. Deploy the API



9.Post method test



10.API TEST

/to-do-app-resource - POST method test results

Request	Latency	Status
/to-do-app-resource	1223	200

Response body

```
{"statusCode": 200, "message": "New task added"}
```

Response headers

```
{
  "Access-Control-Allow-Origin": "*",
  "Content-Type": "application/json",
  "X-Amzn-Trace-Id": "Root=1-65d89ea1-3f289fbcaa7e226ebc221d8d;Parent=795dc6bb7e09f1bc;Sampled=0;lineage=b62a6d32:0"
}
```

Log

```
Execution log for request 9fde651c-5ac1-4c72-88f5-bfd6fb019c16
Fri Feb 23 13:33:21 UTC 2024 : Starting execution for request: 9fde651c-5ac1-4c72-88f5-bfd6fb019c16
Fri Feb 23 13:33:21 UTC 2024 : HTTP Method: POST, Resource Path: /to-do-app-resource
Fri Feb 23 13:33:21 UTC 2024 : Method request path: {}
Fri Feb 23 13:33:21 UTC 2024 : Method request query string: {}
Fri Feb 23 13:33:21 UTC 2024 : Method request headers: {}
Fri Feb 23 13:33:21 UTC 2024 : Method request body before transformations: {
    "httpMethod":"POST",
    "todoid": "1",
    "task":"Serverless lab",
    "state":"Pending"
}
Fri Feb 23 13:33:21 UTC 2024 : Endpoint request URI: https://lambda.us-east-1.amazonaws.com/2015-03-31/functions/arn:aws:lambda:us-east-
1:979147213970:function:toDoApplicationFunction/invocations
Fri Feb 23 13:33:21 UTC 2024 : Endpoint request headers: {X-Amz-Date=20240223T133321Z, x-amzn-apigateway-api-id=98r4241thf, Accept=application/json, User-
Agent=AmazonAPIGateway_98r4241thf, Host=lambda.us-east-1.amazonaws.com, X-Amz-Content-
Sha256=b488fe751a232b6238eb812e6184528f2d3beebb661a33b96f61f98e11c2029f, X-Amzn-Trace-Id=Root=1-65d89ea1-3f289fbcaa7e226ebc221d8d, x-amzn-lambda-
```

Test method

Make a test call to your method. When you make a test call, API Gateway skips authorization and directly invokes your method.

Query strings

param1=value1&param2=value2

Headers

Enter a header name and value separated by a colon (:). Use a new line for each header.

header1:value1
header2:value2

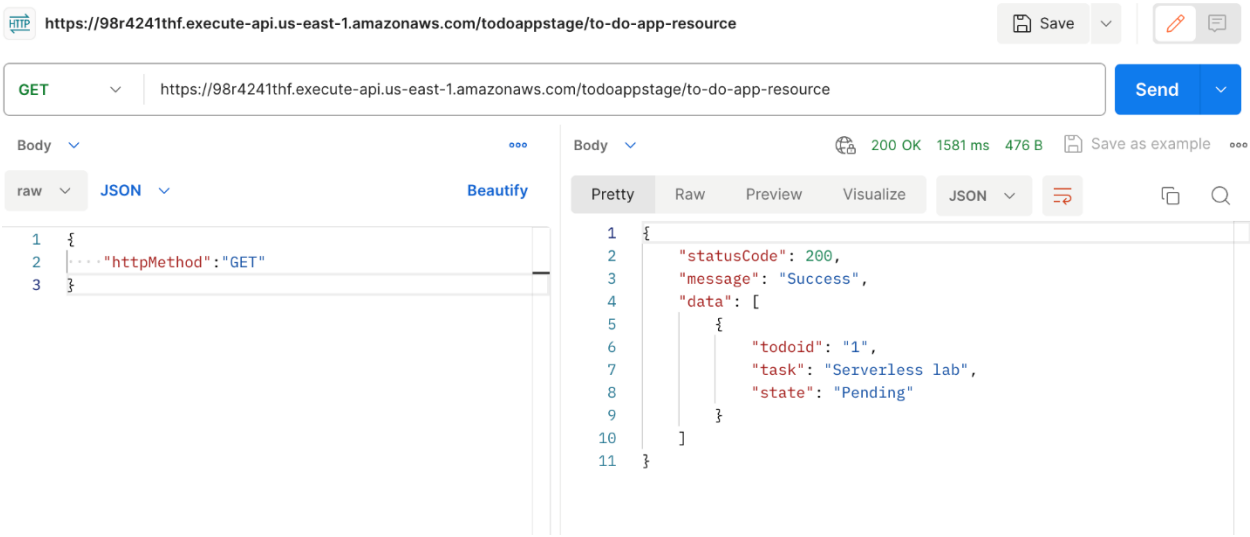
Client certificate

No client certificates have been generated.

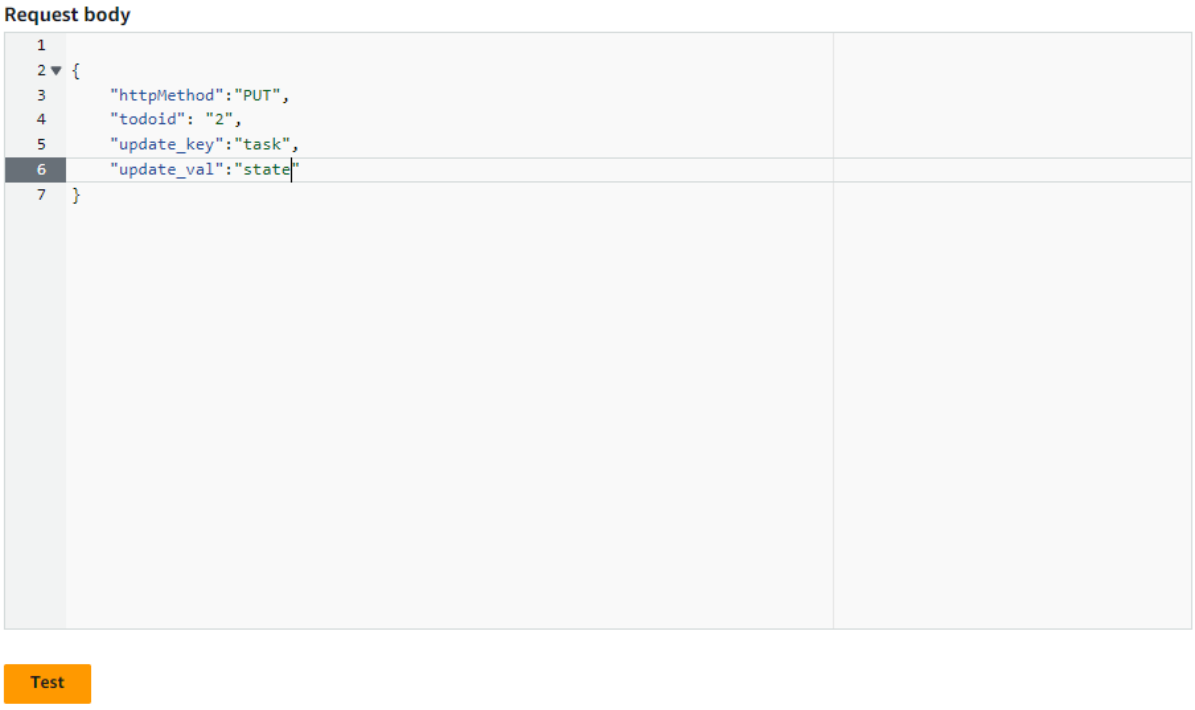
Request body

```
{
    "httpMethod":"POST",
    "todoid": "1",
    "task":"Serverless lab",
    "state":"Pending"
}
```

11.Get request test



12. Put test



/to-do-app-resource - PUT method test results

Request	Latency	Status
/to-do-app-resource	1152	200

Response body

```
{"statusCode": 200, "message": "Task is updated successfully"}
```

Response headers

```
{
  "Access-Control-Allow-Origin": "*",
  "Content-Type": "application/json",
  "X-Amzn-Trace-Id": "Root=1-65d8a1a0-d8780f334680500082e0108f;Parent=12c17aa0a4e445c0;Sampled=0;lineage=b62a6d32:0"
}
```

Log

```
Execution log for request d8a5b691-9a89-4d03-b376-fd7320b19e80
Fri Feb 23 13:46:08 UTC 2024 : Starting execution for request: d8a5b691-9a89-4d03-b376-fd7320b19e80
Fri Feb 23 13:46:08 UTC 2024 : HTTP Method: PUT, Resource Path: /to-do-app-resource
Fri Feb 23 13:46:08 UTC 2024 : Method request path: {}
Fri Feb 23 13:46:08 UTC 2024 : Method request query string: {}
Fri Feb 23 13:46:08 UTC 2024 : Method request headers: {}
Fri Feb 23 13:46:08 UTC 2024 : Method request body before transformations:
{
    "httpMethod":"PUT",
    "todoid": "2",
    "update_key":"task",
    "update_val":"state"
}
Fri Feb 23 13:46:08 UTC 2024 : Endpoint request URI: https://lambda.us-east-1.amazonaws.com/2015-03-31/functions/arn:aws:lambda:us-east-
1:979147213970:function:toDoApplicationFunction/invocations
Fri Feb 23 13:46:08 UTC 2024 : Endpoint request headers: {X-Amz-Date=20240223T134608Z, x-amzn-apigateway-api-id=98r4241thf, Accept=application/json, User-
```

13. We can see that the record is now updated.

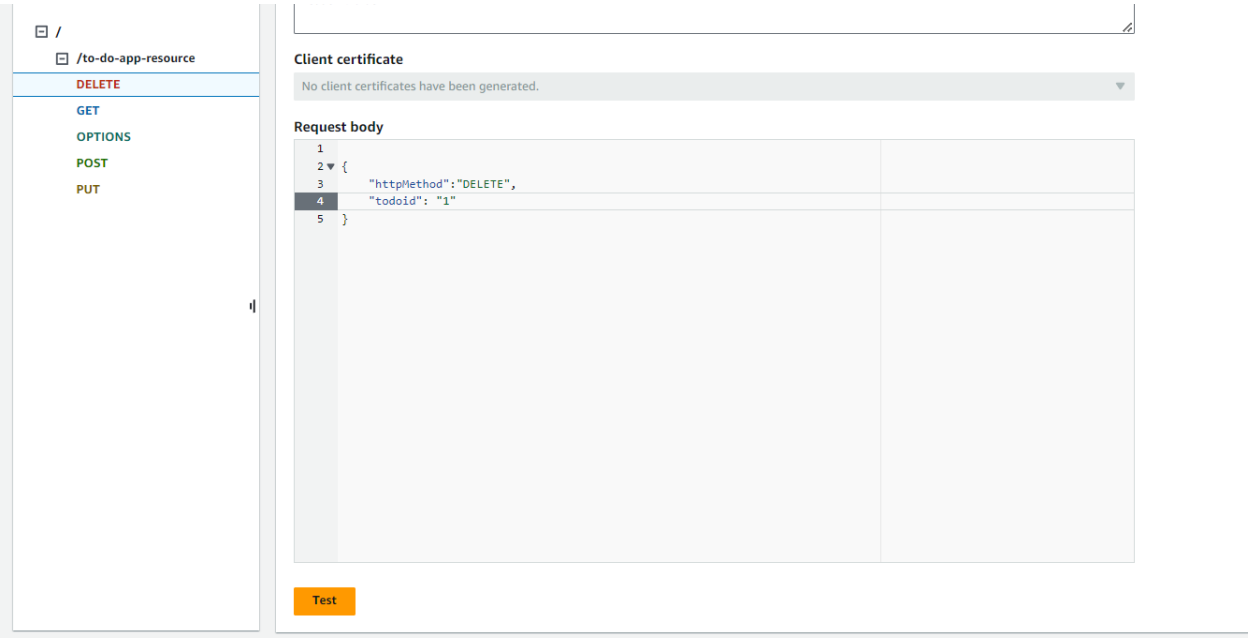
Items returned (2)

Actions ▼ | Create item

1

todoid *(String)*	state	task
2		state
1	Pending	Serverless lab

14. Delete method test



15. Data deleted successfully.